

libcaer

2.4.0

Generated by Doxygen 1.8.13



# Contents

<b>1</b>	<b>Data Structure Index</b>	<b>1</b>
1.1	Data Structures . . . . .	1
<b>2</b>	<b>File Index</b>	<b>3</b>
2.1	File List . . . . .	3
<b>3</b>	<b>Data Structure Documentation</b>	<b>5</b>
3.1	caer_bias_coarsefine Struct Reference . . . . .	5
3.1.1	Detailed Description . . . . .	5
3.2	caer_bias_dynapse Struct Reference . . . . .	5
3.2.1	Detailed Description . . . . .	6
3.3	caer_bias_shiftedsources Struct Reference . . . . .	6
3.3.1	Detailed Description . . . . .	6
3.4	caer_bias_vdac Struct Reference . . . . .	7
3.4.1	Detailed Description . . . . .	7
3.5	caer_davis_info Struct Reference . . . . .	7
3.5.1	Detailed Description . . . . .	8
3.6	caer_dvs128_info Struct Reference . . . . .	8
3.6.1	Detailed Description . . . . .	9
3.7	caer_dynapse_info Struct Reference . . . . .	9
3.7.1	Detailed Description . . . . .	10
3.8	caer_edvs_info Struct Reference . . . . .	10
3.8.1	Detailed Description . . . . .	10

<b>4 File Documentation</b>	<b>11</b>
4.1 devices/davis.h File Reference	11
4.1.1 Detailed Description	21
4.1.2 Macro Definition Documentation	21
4.1.2.1 CAER_DEVICE_DAVIS	21
4.1.2.2 CAER_DEVICE_DAVIS_FX2	21
4.1.2.3 CAER_DEVICE_DAVIS_FX3	21
4.1.2.4 DAVIS128_CONFIG_BIAS_ADCCOMPBP	21
4.1.2.5 DAVIS128_CONFIG_BIAS_ADCREFHIGH	22
4.1.2.6 DAVIS128_CONFIG_BIAS_ADCREFLOW	22
4.1.2.7 DAVIS128_CONFIG_BIAS_AEPDBN	22
4.1.2.8 DAVIS128_CONFIG_BIAS_AEPUXBP	23
4.1.2.9 DAVIS128_CONFIG_BIAS_AEPUYBP	23
4.1.2.10 DAVIS128_CONFIG_BIAS_APSCAS	23
4.1.2.11 DAVIS128_CONFIG_BIAS_APSEVERFLOWLEVEL	24
4.1.2.12 DAVIS128_CONFIG_BIAS_APSROSFBN	24
4.1.2.13 DAVIS128_CONFIG_BIAS_BIASBUFFER	24
4.1.2.14 DAVIS128_CONFIG_BIAS_COLSELLOWBN	25
4.1.2.15 DAVIS128_CONFIG_BIAS_DACBUFBP	25
4.1.2.16 DAVIS128_CONFIG_BIAS_DIFFBN	25
4.1.2.17 DAVIS128_CONFIG_BIAS_IFREFRBN	26
4.1.2.18 DAVIS128_CONFIG_BIAS_IFTHRBN	26
4.1.2.19 DAVIS128_CONFIG_BIAS_LCOLTIMEOUTBN	26
4.1.2.20 DAVIS128_CONFIG_BIAS_LOCALBUFBN	27
4.1.2.21 DAVIS128_CONFIG_BIAS_OFFBN	27
4.1.2.22 DAVIS128_CONFIG_BIAS_ONBN	27
4.1.2.23 DAVIS128_CONFIG_BIAS_PADFOLLBN	28
4.1.2.24 DAVIS128_CONFIG_BIAS_PIXINVBN	28
4.1.2.25 DAVIS128_CONFIG_BIAS_PRBP	28
4.1.2.26 DAVIS128_CONFIG_BIAS_PRSFBN	29

4.1.2.27	DAVIS128_CONFIG_BIAS_READOUTBUFBP . . . . .	29
4.1.2.28	DAVIS128_CONFIG_BIAS_REFRBP . . . . .	29
4.1.2.29	DAVIS128_CONFIG_BIAS_SSN . . . . .	30
4.1.2.30	DAVIS128_CONFIG_BIAS_SSP . . . . .	30
4.1.2.31	DAVIS128_CONFIG_CHIP_AERNAROW . . . . .	30
4.1.2.32	DAVIS128_CONFIG_CHIP_ANALOGMUX0 . . . . .	30
4.1.2.33	DAVIS128_CONFIG_CHIP_ANALOGMUX1 . . . . .	31
4.1.2.34	DAVIS128_CONFIG_CHIP_ANALOGMUX2 . . . . .	31
4.1.2.35	DAVIS128_CONFIG_CHIP_BIASMUX0 . . . . .	31
4.1.2.36	DAVIS128_CONFIG_CHIP_DIGITALMUX0 . . . . .	31
4.1.2.37	DAVIS128_CONFIG_CHIP_DIGITALMUX1 . . . . .	31
4.1.2.38	DAVIS128_CONFIG_CHIP_DIGITALMUX2 . . . . .	31
4.1.2.39	DAVIS128_CONFIG_CHIP_DIGITALMUX3 . . . . .	32
4.1.2.40	DAVIS128_CONFIG_CHIP_GLOBAL_SHUTTER . . . . .	32
4.1.2.41	DAVIS128_CONFIG_CHIP_RESETCALIBNEURON . . . . .	32
4.1.2.42	DAVIS128_CONFIG_CHIP_RESETTESTPIXEL . . . . .	32
4.1.2.43	DAVIS128_CONFIG_CHIP_SELECTGRAYCOUNTER . . . . .	32
4.1.2.44	DAVIS128_CONFIG_CHIP_TYPCALIBNEURON . . . . .	32
4.1.2.45	DAVIS128_CONFIG_CHIP_USEAOUT . . . . .	33
4.1.2.46	DAVIS208_CONFIG_BIAS_ADCCOMPBP . . . . .	33
4.1.2.47	DAVIS208_CONFIG_BIAS_ADCREFHIGH . . . . .	33
4.1.2.48	DAVIS208_CONFIG_BIAS_ADCREFLOW . . . . .	33
4.1.2.49	DAVIS208_CONFIG_BIAS_AEPDBN . . . . .	34
4.1.2.50	DAVIS208_CONFIG_BIAS_AEPUXBP . . . . .	34
4.1.2.51	DAVIS208_CONFIG_BIAS_AEPUYBP . . . . .	34
4.1.2.52	DAVIS208_CONFIG_BIAS_APSCAS . . . . .	35
4.1.2.53	DAVIS208_CONFIG_BIAS_APSEVERFLOWLEVEL . . . . .	35
4.1.2.54	DAVIS208_CONFIG_BIAS_APSROSFBN . . . . .	35
4.1.2.55	DAVIS208_CONFIG_BIAS_BIASBUFFER . . . . .	36
4.1.2.56	DAVIS208_CONFIG_BIAS_COLSELLOWBN . . . . .	36

4.1.2.57	DAVIS208_CONFIG_BIAS_DACBUFBP . . . . .	36
4.1.2.58	DAVIS208_CONFIG_BIAS_DIFFBN . . . . .	37
4.1.2.59	DAVIS208_CONFIG_BIAS_IFREFRBN . . . . .	37
4.1.2.60	DAVIS208_CONFIG_BIAS_IFTHRBN . . . . .	37
4.1.2.61	DAVIS208_CONFIG_BIAS_LCOLTIMEOUTBN . . . . .	38
4.1.2.62	DAVIS208_CONFIG_BIAS_LOCALBUFBN . . . . .	38
4.1.2.63	DAVIS208_CONFIG_BIAS_OFFBN . . . . .	38
4.1.2.64	DAVIS208_CONFIG_BIAS_ONBN . . . . .	39
4.1.2.65	DAVIS208_CONFIG_BIAS_PADFOLLBN . . . . .	39
4.1.2.66	DAVIS208_CONFIG_BIAS_PIXINBN . . . . .	39
4.1.2.67	DAVIS208_CONFIG_BIAS_PRBP . . . . .	40
4.1.2.68	DAVIS208_CONFIG_BIAS_PRSFBP . . . . .	40
4.1.2.69	DAVIS208_CONFIG_BIAS_READOUTBUFBP . . . . .	40
4.1.2.70	DAVIS208_CONFIG_BIAS_REFRBP . . . . .	41
4.1.2.71	DAVIS208_CONFIG_BIAS_REFSS . . . . .	41
4.1.2.72	DAVIS208_CONFIG_BIAS_REFSSBN . . . . .	41
4.1.2.73	DAVIS208_CONFIG_BIAS_REGBIASBP . . . . .	42
4.1.2.74	DAVIS208_CONFIG_BIAS_RESETHIGHPASS . . . . .	42
4.1.2.75	DAVIS208_CONFIG_BIAS_SSN . . . . .	42
4.1.2.76	DAVIS208_CONFIG_BIAS_SSP . . . . .	43
4.1.2.77	DAVIS208_CONFIG_CHIP_AERNAROW . . . . .	43
4.1.2.78	DAVIS208_CONFIG_CHIP_ANALOGMUX0 . . . . .	43
4.1.2.79	DAVIS208_CONFIG_CHIP_ANALOGMUX1 . . . . .	43
4.1.2.80	DAVIS208_CONFIG_CHIP_ANALOGMUX2 . . . . .	43
4.1.2.81	DAVIS208_CONFIG_CHIP_BIASMUX0 . . . . .	44
4.1.2.82	DAVIS208_CONFIG_CHIP_DIGITALMUX0 . . . . .	44
4.1.2.83	DAVIS208_CONFIG_CHIP_DIGITALMUX1 . . . . .	44
4.1.2.84	DAVIS208_CONFIG_CHIP_DIGITALMUX2 . . . . .	44
4.1.2.85	DAVIS208_CONFIG_CHIP_DIGITALMUX3 . . . . .	44
4.1.2.86	DAVIS208_CONFIG_CHIP_GLOBAL_SHUTTER . . . . .	44

4.1.2.87	DAVIS208_CONFIG_CHIP_RESETCALIBNEURON . . . . .	45
4.1.2.88	DAVIS208_CONFIG_CHIP_RESETTESTPIXEL . . . . .	45
4.1.2.89	DAVIS208_CONFIG_CHIP_SELECTBIASREFSS . . . . .	45
4.1.2.90	DAVIS208_CONFIG_CHIP_SELECTGRAYCOUNTER . . . . .	45
4.1.2.91	DAVIS208_CONFIG_CHIP_SELECTHIGHPASS . . . . .	45
4.1.2.92	DAVIS208_CONFIG_CHIP_SELECTPOSFB . . . . .	45
4.1.2.93	DAVIS208_CONFIG_CHIP_SELECTPREMPAVG . . . . .	46
4.1.2.94	DAVIS208_CONFIG_CHIP_SELECTSENSE . . . . .	46
4.1.2.95	DAVIS208_CONFIG_CHIP_TYPENCALIBNEURON . . . . .	46
4.1.2.96	DAVIS208_CONFIG_CHIP_USEAOUT . . . . .	46
4.1.2.97	DAVIS240_CONFIG_BIAS_AEPDBN . . . . .	46
4.1.2.98	DAVIS240_CONFIG_BIAS_AEPUXBP . . . . .	47
4.1.2.99	DAVIS240_CONFIG_BIAS_AEPUYBP . . . . .	47
4.1.2.100	DAVIS240_CONFIG_BIAS_APSCASEPC . . . . .	47
4.1.2.101	DAVIS240_CONFIG_BIAS_APSEVERFLOWLEVELBN . . . . .	47
4.1.2.102	DAVIS240_CONFIG_BIAS_APSROSFBN . . . . .	48
4.1.2.103	DAVIS240_CONFIG_BIAS_BIASBUFFER . . . . .	48
4.1.2.104	DAVIS240_CONFIG_BIAS_DIFFBN . . . . .	48
4.1.2.105	DAVIS240_CONFIG_BIAS_DIFFCASBNC . . . . .	48
4.1.2.106	DAVIS240_CONFIG_BIAS_IFREFRBN . . . . .	49
4.1.2.107	DAVIS240_CONFIG_BIAS_IFTHRBN . . . . .	49
4.1.2.108	DAVIS240_CONFIG_BIAS_LCOLTIMEOUTBN . . . . .	49
4.1.2.109	DAVIS240_CONFIG_BIAS_LOCALBUFBN . . . . .	49
4.1.2.110	DAVIS240_CONFIG_BIAS_OFFBN . . . . .	50
4.1.2.111	DAVIS240_CONFIG_BIAS_ONBN . . . . .	50
4.1.2.112	DAVIS240_CONFIG_BIAS_PADFOLLBN . . . . .	50
4.1.2.113	DAVIS240_CONFIG_BIAS_PIXINVBN . . . . .	50
4.1.2.114	DAVIS240_CONFIG_BIAS_PRBP . . . . .	51
4.1.2.115	DAVIS240_CONFIG_BIAS_PRSFBN . . . . .	51
4.1.2.116	DAVIS240_CONFIG_BIAS_REFRBN . . . . .	51

4.1.2.117 DAVIS240_CONFIG_BIAS_SSN . . . . .	51
4.1.2.118 DAVIS240_CONFIG_BIAS_SSP . . . . .	52
4.1.2.119 DAVIS240_CONFIG_CHIP_AERNAROW . . . . .	52
4.1.2.120 DAVIS240_CONFIG_CHIP_ANALOGMUX0 . . . . .	52
4.1.2.121 DAVIS240_CONFIG_CHIP_ANALOGMUX1 . . . . .	52
4.1.2.122 DAVIS240_CONFIG_CHIP_ANALOGMUX2 . . . . .	52
4.1.2.123 DAVIS240_CONFIG_CHIP_BIASMUX0 . . . . .	53
4.1.2.124 DAVIS240_CONFIG_CHIP_DIGITALMUX0 . . . . .	53
4.1.2.125 DAVIS240_CONFIG_CHIP_DIGITALMUX1 . . . . .	53
4.1.2.126 DAVIS240_CONFIG_CHIP_DIGITALMUX2 . . . . .	53
4.1.2.127 DAVIS240_CONFIG_CHIP_DIGITALMUX3 . . . . .	53
4.1.2.128 DAVIS240_CONFIG_CHIP_GLOBAL_SHUTTER . . . . .	53
4.1.2.129 DAVIS240_CONFIG_CHIP_RESETCALIBNEURON . . . . .	54
4.1.2.130 DAVIS240_CONFIG_CHIP_RESETTESTPIXEL . . . . .	54
4.1.2.131 DAVIS240_CONFIG_CHIP_SPECIALPIXELCONTROL . . . . .	54
4.1.2.132 DAVIS240_CONFIG_CHIP_TYPCALIBNEURON . . . . .	54
4.1.2.133 DAVIS240_CONFIG_CHIP_USEAOUT . . . . .	54
4.1.2.134 DAVIS346_CONFIG_BIAS_ADCCOMPBP . . . . .	55
4.1.2.135 DAVIS346_CONFIG_BIAS_ADCREFHIGH . . . . .	55
4.1.2.136 DAVIS346_CONFIG_BIAS_ADCREFLOW . . . . .	55
4.1.2.137 DAVIS346_CONFIG_BIAS_ADCTESTVOLTAGE . . . . .	56
4.1.2.138 DAVIS346_CONFIG_BIAS_AEPDBN . . . . .	56
4.1.2.139 DAVIS346_CONFIG_BIAS_AEPUXBP . . . . .	56
4.1.2.140 DAVIS346_CONFIG_BIAS_AEPUYBP . . . . .	57
4.1.2.141 DAVIS346_CONFIG_BIAS_APSCAS . . . . .	57
4.1.2.142 DAVIS346_CONFIG_BIAS_APSEVERFLOWLEVEL . . . . .	57
4.1.2.143 DAVIS346_CONFIG_BIAS_APSROSFBN . . . . .	58
4.1.2.144 DAVIS346_CONFIG_BIAS_BIASBUFFER . . . . .	58
4.1.2.145 DAVIS346_CONFIG_BIAS_COLSELLOWBN . . . . .	58
4.1.2.146 DAVIS346_CONFIG_BIAS_DACBUFBP . . . . .	59



4.1.2.147 DAVIS346_CONFIG_BIAS_DIFFBN . . . . .	59
4.1.2.148 DAVIS346_CONFIG_BIAS_IFREFRBN . . . . .	59
4.1.2.149 DAVIS346_CONFIG_BIAS_IFTHRBN . . . . .	60
4.1.2.150 DAVIS346_CONFIG_BIAS_LCOLTIMEOUTBN . . . . .	60
4.1.2.151 DAVIS346_CONFIG_BIAS_LOCALBUFBN . . . . .	60
4.1.2.152 DAVIS346_CONFIG_BIAS_OFFBN . . . . .	61
4.1.2.153 DAVIS346_CONFIG_BIAS_ONBN . . . . .	61
4.1.2.154 DAVIS346_CONFIG_BIAS_PADFOLLBN . . . . .	61
4.1.2.155 DAVIS346_CONFIG_BIAS_PIXINVBN . . . . .	62
4.1.2.156 DAVIS346_CONFIG_BIAS_PRBP . . . . .	62
4.1.2.157 DAVIS346_CONFIG_BIAS_PRSFBP . . . . .	62
4.1.2.158 DAVIS346_CONFIG_BIAS_READOUTBUFBP . . . . .	63
4.1.2.159 DAVIS346_CONFIG_BIAS_REFRBP . . . . .	63
4.1.2.160 DAVIS346_CONFIG_BIAS_SSN . . . . .	63
4.1.2.161 DAVIS346_CONFIG_BIAS_SSP . . . . .	64
4.1.2.162 DAVIS346_CONFIG_CHIP_AERNAROW . . . . .	64
4.1.2.163 DAVIS346_CONFIG_CHIP_ANALOGMUX0 . . . . .	64
4.1.2.164 DAVIS346_CONFIG_CHIP_ANALOGMUX1 . . . . .	64
4.1.2.165 DAVIS346_CONFIG_CHIP_ANALOGMUX2 . . . . .	64
4.1.2.166 DAVIS346_CONFIG_CHIP_BIASMUX0 . . . . .	65
4.1.2.167 DAVIS346_CONFIG_CHIP_DIGITALMUX0 . . . . .	65
4.1.2.168 DAVIS346_CONFIG_CHIP_DIGITALMUX1 . . . . .	65
4.1.2.169 DAVIS346_CONFIG_CHIP_DIGITALMUX2 . . . . .	65
4.1.2.170 DAVIS346_CONFIG_CHIP_DIGITALMUX3 . . . . .	65
4.1.2.171 DAVIS346_CONFIG_CHIP_GLOBAL_SHUTTER . . . . .	65
4.1.2.172 DAVIS346_CONFIG_CHIP_RESETCALIBNEURON . . . . .	66
4.1.2.173 DAVIS346_CONFIG_CHIP_RESETTESTPIXEL . . . . .	66
4.1.2.174 DAVIS346_CONFIG_CHIP_SELECTGRAYCOUNTER . . . . .	66
4.1.2.175 DAVIS346_CONFIG_CHIP_TESTADC . . . . .	66
4.1.2.176 DAVIS346_CONFIG_CHIP_TYPENCALIBNEURON . . . . .	66

4.1.2.177 DAVIS346_CONFIG_CHIP_USEAOUT . . . . .	66
4.1.2.178 DAVIS640_CONFIG_BIAS_ADCCOMPBP . . . . .	67
4.1.2.179 DAVIS640_CONFIG_BIAS_ADCREFHIGH . . . . .	67
4.1.2.180 DAVIS640_CONFIG_BIAS_ADCREFLOW . . . . .	67
4.1.2.181 DAVIS640_CONFIG_BIAS_ADCTESTVOLTAGE . . . . .	68
4.1.2.182 DAVIS640_CONFIG_BIAS_AEPDBN . . . . .	68
4.1.2.183 DAVIS640_CONFIG_BIAS_AEPUXBP . . . . .	68
4.1.2.184 DAVIS640_CONFIG_BIAS_AEPUYBP . . . . .	69
4.1.2.185 DAVIS640_CONFIG_BIAS_APSCAS . . . . .	69
4.1.2.186 DAVIS640_CONFIG_BIAS_APSOEVERFLOWLEVEL . . . . .	69
4.1.2.187 DAVIS640_CONFIG_BIAS_APSROSFBN . . . . .	70
4.1.2.188 DAVIS640_CONFIG_BIAS_BIASBUFFER . . . . .	70
4.1.2.189 DAVIS640_CONFIG_BIAS_COLSELLOWBN . . . . .	70
4.1.2.190 DAVIS640_CONFIG_BIAS_DACBUFBP . . . . .	71
4.1.2.191 DAVIS640_CONFIG_BIAS_DIFFBN . . . . .	71
4.1.2.192 DAVIS640_CONFIG_BIAS_IFREFRBN . . . . .	71
4.1.2.193 DAVIS640_CONFIG_BIAS_IFTHRBN . . . . .	72
4.1.2.194 DAVIS640_CONFIG_BIAS_LCOLTIMEOUTBN . . . . .	72
4.1.2.195 DAVIS640_CONFIG_BIAS_LOCALBUFBN . . . . .	72
4.1.2.196 DAVIS640_CONFIG_BIAS_OFFBN . . . . .	73
4.1.2.197 DAVIS640_CONFIG_BIAS_ONBN . . . . .	73
4.1.2.198 DAVIS640_CONFIG_BIAS_PADFOLLBN . . . . .	73
4.1.2.199 DAVIS640_CONFIG_BIAS_PIXINVBN . . . . .	74
4.1.2.200 DAVIS640_CONFIG_BIAS_PRBP . . . . .	74
4.1.2.201 DAVIS640_CONFIG_BIAS_PRSFBP . . . . .	74
4.1.2.202 DAVIS640_CONFIG_BIAS_READOUTBUFBP . . . . .	75
4.1.2.203 DAVIS640_CONFIG_BIAS_REFRBP . . . . .	75
4.1.2.204 DAVIS640_CONFIG_BIAS_SSN . . . . .	75
4.1.2.205 DAVIS640_CONFIG_BIAS_SSP . . . . .	76
4.1.2.206 DAVIS640_CONFIG_CHIP_AERNAROW . . . . .	76

4.1.2.207 DAVIS640_CONFIG_CHIP_ANALOGMUX0 . . . . .	76
4.1.2.208 DAVIS640_CONFIG_CHIP_ANALOGMUX1 . . . . .	76
4.1.2.209 DAVIS640_CONFIG_CHIP_ANALOGMUX2 . . . . .	76
4.1.2.210 DAVIS640_CONFIG_CHIP_BIASMUX0 . . . . .	77
4.1.2.211 DAVIS640_CONFIG_CHIP_DIGITALMUX0 . . . . .	77
4.1.2.212 DAVIS640_CONFIG_CHIP_DIGITALMUX1 . . . . .	77
4.1.2.213 DAVIS640_CONFIG_CHIP_DIGITALMUX2 . . . . .	77
4.1.2.214 DAVIS640_CONFIG_CHIP_DIGITALMUX3 . . . . .	77
4.1.2.215 DAVIS640_CONFIG_CHIP_GLOBAL_SHUTTER . . . . .	77
4.1.2.216 DAVIS640_CONFIG_CHIP_RESETCALIBNEURON . . . . .	78
4.1.2.217 DAVIS640_CONFIG_CHIP_RESETTESTPIXEL . . . . .	78
4.1.2.218 DAVIS640_CONFIG_CHIP_SELECTGRAYCOUNTER . . . . .	78
4.1.2.219 DAVIS640_CONFIG_CHIP_TESTADC . . . . .	78
4.1.2.220 DAVIS640_CONFIG_CHIP_TYPCALIBNEURON . . . . .	78
4.1.2.221 DAVIS640_CONFIG_CHIP_USEAOUT . . . . .	78
4.1.2.222 DAVIS_CHIP_DAVIS128 . . . . .	79
4.1.2.223 DAVIS_CHIP_DAVIS208 . . . . .	79
4.1.2.224 DAVIS_CHIP_DAVIS240A . . . . .	79
4.1.2.225 DAVIS_CHIP_DAVIS240B . . . . .	79
4.1.2.226 DAVIS_CHIP_DAVIS240C . . . . .	79
4.1.2.227 DAVIS_CHIP_DAVIS346A . . . . .	79
4.1.2.228 DAVIS_CHIP_DAVIS346B . . . . .	79
4.1.2.229 DAVIS_CHIP_DAVIS346C . . . . .	79
4.1.2.230 DAVIS_CHIP_DAVIS640 . . . . .	80
4.1.2.231 DAVIS_CHIP_DAVISRGB . . . . .	80
4.1.2.232 DAVIS_CONFIG_APS . . . . .	80
4.1.2.233 DAVIS_CONFIG_APS_ADC_TEST_MODE . . . . .	80
4.1.2.234 DAVIS_CONFIG_APS_AUTOEXPOSURE . . . . .	80
4.1.2.235 DAVIS_CONFIG_APS_COLOR_FILTER . . . . .	80
4.1.2.236 DAVIS_CONFIG_APS_COLUMN_SETTLE . . . . .	80

4.1.2.237 DAVIS_CONFIG_APS_END_COLUMN_0 . . . . .	81
4.1.2.238 DAVIS_CONFIG_APS_END_COLUMN_1 . . . . .	81
4.1.2.239 DAVIS_CONFIG_APS_END_COLUMN_2 . . . . .	81
4.1.2.240 DAVIS_CONFIG_APS_END_COLUMN_3 . . . . .	81
4.1.2.241 DAVIS_CONFIG_APS_END_ROW_0 . . . . .	81
4.1.2.242 DAVIS_CONFIG_APS_END_ROW_1 . . . . .	81
4.1.2.243 DAVIS_CONFIG_APS_END_ROW_2 . . . . .	81
4.1.2.244 DAVIS_CONFIG_APS_END_ROW_3 . . . . .	82
4.1.2.245 DAVIS_CONFIG_APS_EXPOSURE . . . . .	82
4.1.2.246 DAVIS_CONFIG_APS_FRAME_DELAY . . . . .	82
4.1.2.247 DAVIS_CONFIG_APS_GLOBAL_SHUTTER . . . . .	82
4.1.2.248 DAVIS_CONFIG_APS_HAS_EXTERNAL_ADC . . . . .	82
4.1.2.249 DAVIS_CONFIG_APS_HAS_GLOBAL_SHUTTER . . . . .	82
4.1.2.250 DAVIS_CONFIG_APS_HAS_INTERNAL_ADC . . . . .	83
4.1.2.251 DAVIS_CONFIG_APS_HAS_QUAD_ROI . . . . .	83
4.1.2.252 DAVIS_CONFIG_APS_NULL_SETTLE . . . . .	83
4.1.2.253 DAVIS_CONFIG_APS_ORIENTATION_INFO . . . . .	83
4.1.2.254 DAVIS_CONFIG_APS_RAMP_RESET . . . . .	83
4.1.2.255 DAVIS_CONFIG_APS_RAMP_SHORT_RESET . . . . .	83
4.1.2.256 DAVIS_CONFIG_APS_RESET_READ . . . . .	84
4.1.2.257 DAVIS_CONFIG_APS_RESET_SETTLE . . . . .	84
4.1.2.258 DAVIS_CONFIG_APS_ROW_SETTLE . . . . .	84
4.1.2.259 DAVIS_CONFIG_APS_RUN . . . . .	84
4.1.2.260 DAVIS_CONFIG_APS_SAMPLE_ENABLE . . . . .	84
4.1.2.261 DAVIS_CONFIG_APS_SAMPLE_SETTLE . . . . .	84
4.1.2.262 DAVIS_CONFIG_APS_SIZE_COLUMNS . . . . .	84
4.1.2.263 DAVIS_CONFIG_APS_SIZE_ROWS . . . . .	85
4.1.2.264 DAVIS_CONFIG_APS_SNAPSHOT . . . . .	85
4.1.2.265 DAVIS_CONFIG_APS_START_COLUMN_0 . . . . .	85
4.1.2.266 DAVIS_CONFIG_APS_START_COLUMN_1 . . . . .	85

4.1.2.267 DAVIS_CONFIG_APS_START_COLUMN_2 . . . . .	85
4.1.2.268 DAVIS_CONFIG_APS_START_COLUMN_3 . . . . .	85
4.1.2.269 DAVIS_CONFIG_APS_START_ROW_0 . . . . .	86
4.1.2.270 DAVIS_CONFIG_APS_START_ROW_1 . . . . .	86
4.1.2.271 DAVIS_CONFIG_APS_START_ROW_2 . . . . .	86
4.1.2.272 DAVIS_CONFIG_APS_START_ROW_3 . . . . .	86
4.1.2.273 DAVIS_CONFIG_APS_USE_INTERNAL_ADC . . . . .	86
4.1.2.274 DAVIS_CONFIG_APS_WAIT_ON_TRANSFER_STALL . . . . .	86
4.1.2.275 DAVIS_CONFIG_BIAS . . . . .	87
4.1.2.276 DAVIS_CONFIG_CHIP . . . . .	87
4.1.2.277 DAVIS_CONFIG_DVS . . . . .	87
4.1.2.278 DAVIS_CONFIG_DVS_ACK_DELAY_COLUMN . . . . .	87
4.1.2.279 DAVIS_CONFIG_DVS_ACK_DELAY_ROW . . . . .	87
4.1.2.280 DAVIS_CONFIG_DVS_ACK_EXTENSION_COLUMN . . . . .	87
4.1.2.281 DAVIS_CONFIG_DVS_ACK_EXTENSION_ROW . . . . .	87
4.1.2.282 DAVIS_CONFIG_DVS_EXTERNAL_AER_CONTROL . . . . .	88
4.1.2.283 DAVIS_CONFIG_DVS_FILTER_BACKGROUND_ACTIVITY . . . . .	88
4.1.2.284 DAVIS_CONFIG_DVS_FILTER_BACKGROUND_ACTIVITY_DELTAT . . . . .	88
4.1.2.285 DAVIS_CONFIG_DVS_FILTER_PIXEL_0_COLUMN . . . . .	88
4.1.2.286 DAVIS_CONFIG_DVS_FILTER_PIXEL_0_ROW . . . . .	88
4.1.2.287 DAVIS_CONFIG_DVS_FILTER_PIXEL_1_COLUMN . . . . .	88
4.1.2.288 DAVIS_CONFIG_DVS_FILTER_PIXEL_1_ROW . . . . .	89
4.1.2.289 DAVIS_CONFIG_DVS_FILTER_PIXEL_2_COLUMN . . . . .	89
4.1.2.290 DAVIS_CONFIG_DVS_FILTER_PIXEL_2_ROW . . . . .	89
4.1.2.291 DAVIS_CONFIG_DVS_FILTER_PIXEL_3_COLUMN . . . . .	89
4.1.2.292 DAVIS_CONFIG_DVS_FILTER_PIXEL_3_ROW . . . . .	89
4.1.2.293 DAVIS_CONFIG_DVS_FILTER_PIXEL_4_COLUMN . . . . .	89
4.1.2.294 DAVIS_CONFIG_DVS_FILTER_PIXEL_4_ROW . . . . .	89
4.1.2.295 DAVIS_CONFIG_DVS_FILTER_PIXEL_5_COLUMN . . . . .	90
4.1.2.296 DAVIS_CONFIG_DVS_FILTER_PIXEL_5_ROW . . . . .	90

4.1.2.297 DAVIS_CONFIG_DVS_FILTER_PIXEL_6_COLUMN . . . . .	90
4.1.2.298 DAVIS_CONFIG_DVS_FILTER_PIXEL_6_ROW . . . . .	90
4.1.2.299 DAVIS_CONFIG_DVS_FILTER_PIXEL_7_COLUMN . . . . .	90
4.1.2.300 DAVIS_CONFIG_DVS_FILTER_PIXEL_7_ROW . . . . .	90
4.1.2.301 DAVIS_CONFIG_DVS_FILTER_ROW_ONLY_EVENTS . . . . .	90
4.1.2.302 DAVIS_CONFIG_DVS_HAS_BACKGROUND_ACTIVITY_FILTER . . . . .	91
4.1.2.303 DAVIS_CONFIG_DVS_HAS_PIXEL_FILTER . . . . .	91
4.1.2.304 DAVIS_CONFIG_DVS_HAS_TEST_EVENT_GENERATOR . . . . .	91
4.1.2.305 DAVIS_CONFIG_DVS_ORIENTATION_INFO . . . . .	91
4.1.2.306 DAVIS_CONFIG_DVS_RUN . . . . .	91
4.1.2.307 DAVIS_CONFIG_DVS_SIZE_COLUMNS . . . . .	91
4.1.2.308 DAVIS_CONFIG_DVS_SIZE_ROWS . . . . .	92
4.1.2.309 DAVIS_CONFIG_DVS_TEST_EVENT_GENERATOR_ENABLE . . . . .	92
4.1.2.310 DAVIS_CONFIG_DVS_WAIT_ON_TRANSFER_STALL . . . . .	92
4.1.2.311 DAVIS_CONFIG_EXTINPUT . . . . .	92
4.1.2.312 DAVIS_CONFIG_EXTINPUT_DETECT_FALLING_EDGES . . . . .	92
4.1.2.313 DAVIS_CONFIG_EXTINPUT_DETECT_FALLING_EDGES1 . . . . .	92
4.1.2.314 DAVIS_CONFIG_EXTINPUT_DETECT_FALLING_EDGES2 . . . . .	93
4.1.2.315 DAVIS_CONFIG_EXTINPUT_DETECT_PULSE_LENGTH . . . . .	93
4.1.2.316 DAVIS_CONFIG_EXTINPUT_DETECT_PULSE_LENGTH1 . . . . .	93
4.1.2.317 DAVIS_CONFIG_EXTINPUT_DETECT_PULSE_LENGTH2 . . . . .	93
4.1.2.318 DAVIS_CONFIG_EXTINPUT_DETECT_PULSE_POLARITY . . . . .	93
4.1.2.319 DAVIS_CONFIG_EXTINPUT_DETECT_PULSE_POLARITY1 . . . . .	93
4.1.2.320 DAVIS_CONFIG_EXTINPUT_DETECT_PULSE_POLARITY2 . . . . .	94
4.1.2.321 DAVIS_CONFIG_EXTINPUT_DETECT_PULSES . . . . .	94
4.1.2.322 DAVIS_CONFIG_EXTINPUT_DETECT_PULSES1 . . . . .	94
4.1.2.323 DAVIS_CONFIG_EXTINPUT_DETECT_PULSES2 . . . . .	94
4.1.2.324 DAVIS_CONFIG_EXTINPUT_DETECT_RISING_EDGES . . . . .	94
4.1.2.325 DAVIS_CONFIG_EXTINPUT_DETECT_RISING_EDGES1 . . . . .	94
4.1.2.326 DAVIS_CONFIG_EXTINPUT_DETECT_RISING_EDGES2 . . . . .	95

4.1.2.327 DAVIS_CONFIG_EXTINPUT_GENERATE_INJECT_ON_FALLING_EDGE . . .	95
4.1.2.328 DAVIS_CONFIG_EXTINPUT_GENERATE_INJECT_ON_RISING_EDGE . . . .	95
4.1.2.329 DAVIS_CONFIG_EXTINPUT_GENERATE_PULSE_INTERVAL . . . . .	95
4.1.2.330 DAVIS_CONFIG_EXTINPUT_GENERATE_PULSE_LENGTH . . . . .	95
4.1.2.331 DAVIS_CONFIG_EXTINPUT_GENERATE_PULSE_POLARITY . . . . .	95
4.1.2.332 DAVIS_CONFIG_EXTINPUT_GENERATE_USE_CUSTOM_SIGNAL . . . . .	96
4.1.2.333 DAVIS_CONFIG_EXTINPUT_HAS_EXTRA_DETECTORS . . . . .	96
4.1.2.334 DAVIS_CONFIG_EXTINPUT_HAS_GENERATOR . . . . .	96
4.1.2.335 DAVIS_CONFIG_EXTINPUT_RUN_DETECTOR . . . . .	96
4.1.2.336 DAVIS_CONFIG_EXTINPUT_RUN_DETECTOR1 . . . . .	96
4.1.2.337 DAVIS_CONFIG_EXTINPUT_RUN_DETECTOR2 . . . . .	96
4.1.2.338 DAVIS_CONFIG_EXTINPUT_RUN_GENERATOR . . . . .	97
4.1.2.339 DAVIS_CONFIG_IMU . . . . .	97
4.1.2.340 DAVIS_CONFIG_IMU_ACCEL_FULL_SCALE . . . . .	97
4.1.2.341 DAVIS_CONFIG_IMU_ACCEL_STANDBY . . . . .	97
4.1.2.342 DAVIS_CONFIG_IMU_DIGITAL_LOW_PASS_FILTER . . . . .	97
4.1.2.343 DAVIS_CONFIG_IMU_GYRO_FULL_SCALE . . . . .	97
4.1.2.344 DAVIS_CONFIG_IMU_GYRO_STANDBY . . . . .	98
4.1.2.345 DAVIS_CONFIG_IMU_LP_CYCLE . . . . .	98
4.1.2.346 DAVIS_CONFIG_IMU_LP_WAKEUP . . . . .	98
4.1.2.347 DAVIS_CONFIG_IMU_ORIENTATION_INFO . . . . .	98
4.1.2.348 DAVIS_CONFIG_IMU_RUN . . . . .	98
4.1.2.349 DAVIS_CONFIG_IMU_SAMPLE_RATE_DIVIDER . . . . .	98
4.1.2.350 DAVIS_CONFIG_IMU_TEMP_STANDBY . . . . .	99
4.1.2.351 DAVIS_CONFIG_MICROPHONE . . . . .	99
4.1.2.352 DAVIS_CONFIG_MICROPHONE_RUN . . . . .	99
4.1.2.353 DAVIS_CONFIG_MICROPHONE_SAMPLE_FREQUENCY . . . . .	99
4.1.2.354 DAVIS_CONFIG_MUX . . . . .	99
4.1.2.355 DAVIS_CONFIG_MUX_DROP_APS_ON_TRANSFER_STALL . . . . .	99
4.1.2.356 DAVIS_CONFIG_MUX_DROP_DVS_ON_TRANSFER_STALL . . . . .	100

4.1.2.357 DAVIS_CONFIG_MUX_DROP_EXTINPUT_ON_TRANSFER_STALL . . . . .	100
4.1.2.358 DAVIS_CONFIG_MUX_DROP_IMU_ON_TRANSFER_STALL . . . . .	100
4.1.2.359 DAVIS_CONFIG_MUX_DROP_MIC_ON_TRANSFER_STALL . . . . .	100
4.1.2.360 DAVIS_CONFIG_MUX_FORCE_CHIP_BIAS_ENABLE . . . . .	100
4.1.2.361 DAVIS_CONFIG_MUX_RUN . . . . .	100
4.1.2.362 DAVIS_CONFIG_MUX_TIMESTAMP_RESET . . . . .	101
4.1.2.363 DAVIS_CONFIG_MUX_TIMESTAMP_RUN . . . . .	101
4.1.2.364 DAVIS_CONFIG_SYSINFO . . . . .	101
4.1.2.365 DAVIS_CONFIG_SYSINFO_ADC_CLOCK . . . . .	101
4.1.2.366 DAVIS_CONFIG_SYSINFO_CHIP_IDENTIFIER . . . . .	101
4.1.2.367 DAVIS_CONFIG_SYSINFO_DEVICE_IS_MASTER . . . . .	101
4.1.2.368 DAVIS_CONFIG_SYSINFO_LOGIC_CLOCK . . . . .	102
4.1.2.369 DAVIS_CONFIG_SYSINFO_LOGIC_VERSION . . . . .	102
4.1.2.370 DAVIS_CONFIG_USB . . . . .	102
4.1.2.371 DAVIS_CONFIG_USB_EARLY_PACKET_DELAY . . . . .	102
4.1.2.372 DAVIS_CONFIG_USB_RUN . . . . .	102
4.1.2.373 DAVISRGB_CONFIG_APS_GSFDRESET . . . . .	102
4.1.2.374 DAVISRGB_CONFIG_APS_GSPDRESET . . . . .	103
4.1.2.375 DAVISRGB_CONFIG_APS_GSRESETFALL . . . . .	103
4.1.2.376 DAVISRGB_CONFIG_APS_GSTXFALL . . . . .	103
4.1.2.377 DAVISRGB_CONFIG_APS_RSFDSETTLE . . . . .	103
4.1.2.378 DAVISRGB_CONFIG_APS_TRANSFER . . . . .	103
4.1.2.379 DAVISRGB_CONFIG_BIAS_ADCCOMPBP . . . . .	103
4.1.2.380 DAVISRGB_CONFIG_BIAS_ADCREFHIGH . . . . .	104
4.1.2.381 DAVISRGB_CONFIG_BIAS_ADCREFLOW . . . . .	104
4.1.2.382 DAVISRGB_CONFIG_BIAS_ADCTESTVOLTAGE . . . . .	104
4.1.2.383 DAVISRGB_CONFIG_BIAS_AEPDBN . . . . .	105
4.1.2.384 DAVISRGB_CONFIG_BIAS_AEPUXBP . . . . .	105
4.1.2.385 DAVISRGB_CONFIG_BIAS_AEPUYBP . . . . .	105
4.1.2.386 DAVISRGB_CONFIG_BIAS_APSCAS . . . . .	106



4.1.2.387 DAVISRGB_CONFIG_BIAS_APSROSFBN . . . . .	106
4.1.2.388 DAVISRGB_CONFIG_BIAS_ARRAYBIASBUFFERBN . . . . .	106
4.1.2.389 DAVISRGB_CONFIG_BIAS_ARRAYLOGICBUFFERBN . . . . .	107
4.1.2.390 DAVISRGB_CONFIG_BIAS_BIASBUFFER . . . . .	107
4.1.2.391 DAVISRGB_CONFIG_BIAS_DACBUFBP . . . . .	107
4.1.2.392 DAVISRGB_CONFIG_BIAS_DIFFBN . . . . .	108
4.1.2.393 DAVISRGB_CONFIG_BIAS_FALLTIMEBN . . . . .	108
4.1.2.394 DAVISRGB_CONFIG_BIAS_GND07 . . . . .	108
4.1.2.395 DAVISRGB_CONFIG_BIAS_IFREFRBN . . . . .	109
4.1.2.396 DAVISRGB_CONFIG_BIAS_IFTHRBN . . . . .	109
4.1.2.397 DAVISRGB_CONFIG_BIAS_LCOLTIMEOUTBN . . . . .	109
4.1.2.398 DAVISRGB_CONFIG_BIAS_LOCALBUFBN . . . . .	110
4.1.2.399 DAVISRGB_CONFIG_BIAS_OFFBN . . . . .	110
4.1.2.400 DAVISRGB_CONFIG_BIAS_ONBN . . . . .	110
4.1.2.401 DAVISRGB_CONFIG_BIAS_OVG1LO . . . . .	111
4.1.2.402 DAVISRGB_CONFIG_BIAS_OVG2LO . . . . .	111
4.1.2.403 DAVISRGB_CONFIG_BIAS_PADFOLLBN . . . . .	111
4.1.2.404 DAVISRGB_CONFIG_BIAS_PIXINVBN . . . . .	112
4.1.2.405 DAVISRGB_CONFIG_BIAS_PRBP . . . . .	112
4.1.2.406 DAVISRGB_CONFIG_BIAS_PRSFBP . . . . .	112
4.1.2.407 DAVISRGB_CONFIG_BIAS_READOUTBUFBP . . . . .	113
4.1.2.408 DAVISRGB_CONFIG_BIAS_REFRBP . . . . .	113
4.1.2.409 DAVISRGB_CONFIG_BIAS_RISETIMEBP . . . . .	113
4.1.2.410 DAVISRGB_CONFIG_BIAS_SSN . . . . .	114
4.1.2.411 DAVISRGB_CONFIG_BIAS_SSP . . . . .	114
4.1.2.412 DAVISRGB_CONFIG_BIAS_TX2OVG2HI . . . . .	114
4.1.2.413 DAVISRGB_CONFIG_CHIP_ADJUSTOVG1LO . . . . .	115
4.1.2.414 DAVISRGB_CONFIG_CHIP_ADJUSTOVG2LO . . . . .	115
4.1.2.415 DAVISRGB_CONFIG_CHIP_ADJUSTTX2OVG2HI . . . . .	115
4.1.2.416 DAVISRGB_CONFIG_CHIP_AERNAROW . . . . .	115

4.1.2.417	DAVISRGB_CONFIG_CHIP_ANALOGMUX0 . . . . .	115
4.1.2.418	DAVISRGB_CONFIG_CHIP_ANALOGMUX1 . . . . .	115
4.1.2.419	DAVISRGB_CONFIG_CHIP_ANALOGMUX2 . . . . .	116
4.1.2.420	DAVISRGB_CONFIG_CHIP_BIASMUX0 . . . . .	116
4.1.2.421	DAVISRGB_CONFIG_CHIP_DIGITALMUX0 . . . . .	116
4.1.2.422	DAVISRGB_CONFIG_CHIP_DIGITALMUX1 . . . . .	116
4.1.2.423	DAVISRGB_CONFIG_CHIP_DIGITALMUX2 . . . . .	116
4.1.2.424	DAVISRGB_CONFIG_CHIP_DIGITALMUX3 . . . . .	116
4.1.2.425	DAVISRGB_CONFIG_CHIP_RESETCALIBNEURON . . . . .	117
4.1.2.426	DAVISRGB_CONFIG_CHIP_RESETTESTPIXEL . . . . .	117
4.1.2.427	DAVISRGB_CONFIG_CHIP_SELECTGRAYCOUNTER . . . . .	117
4.1.2.428	DAVISRGB_CONFIG_CHIP_TESTADC . . . . .	117
4.1.2.429	DAVISRGB_CONFIG_CHIP_TYPENCALIBNEURON . . . . .	117
4.1.2.430	DAVISRGB_CONFIG_CHIP_USEAOUT . . . . .	117
4.1.2.431	IS_DAVIS128 . . . . .	118
4.1.2.432	IS_DAVIS208 . . . . .	118
4.1.2.433	IS_DAVIS240 . . . . .	118
4.1.2.434	IS_DAVIS240A . . . . .	118
4.1.2.435	IS_DAVIS240B . . . . .	118
4.1.2.436	IS_DAVIS240C . . . . .	118
4.1.2.437	IS_DAVIS346 . . . . .	119
4.1.2.438	IS_DAVIS346A . . . . .	119
4.1.2.439	IS_DAVIS346B . . . . .	119
4.1.2.440	IS_DAVIS346C . . . . .	119
4.1.2.441	IS_DAVIS640 . . . . .	119
4.1.2.442	IS_DAVISRGB . . . . .	119
4.1.3	Enumeration Type Documentation . . . . .	119
4.1.3.1	caer_bias_shiftedsource_operating_mode . . . . .	119
4.1.3.2	caer_bias_shiftedsource_voltage_level . . . . .	120
4.1.4	Function Documentation . . . . .	120

4.1.4.1	<a href="#">caerBiasCoarseFineGenerate()</a>	120
4.1.4.2	<a href="#">caerBiasCoarseFineParse()</a>	120
4.1.4.3	<a href="#">caerBiasShiftedSourceGenerate()</a>	122
4.1.4.4	<a href="#">caerBiasShiftedSourceParse()</a>	122
4.1.4.5	<a href="#">caerBiasVDACGenerate()</a>	123
4.1.4.6	<a href="#">caerBiasVDACParse()</a>	123
4.1.4.7	<a href="#">caerDavisInfoGet()</a>	123
4.2	<a href="#">devices/device.h File Reference</a>	124
4.2.1	<a href="#">Detailed Description</a>	124
4.2.2	<a href="#">Macro Definition Documentation</a>	124
4.2.2.1	<a href="#">CAER_HOST_CONFIG_DATAEXCHANGE</a>	125
4.2.2.2	<a href="#">CAER_HOST_CONFIG_DATAEXCHANGE_BLOCKING</a>	125
4.2.2.3	<a href="#">CAER_HOST_CONFIG_DATAEXCHANGE_BUFFER_SIZE</a>	125
4.2.2.4	<a href="#">CAER_HOST_CONFIG_DATAEXCHANGE_START_PRODUCERS</a>	125
4.2.2.5	<a href="#">CAER_HOST_CONFIG_DATAEXCHANGE_STOP_PRODUCERS</a>	125
4.2.2.6	<a href="#">CAER_HOST_CONFIG_LOG</a>	125
4.2.2.7	<a href="#">CAER_HOST_CONFIG_LOG_LEVEL</a>	126
4.2.2.8	<a href="#">CAER_HOST_CONFIG_PACKETS</a>	126
4.2.2.9	<a href="#">CAER_HOST_CONFIG_PACKETS_MAX_CONTAINER_INTERVAL</a>	126
4.2.2.10	<a href="#">CAER_HOST_CONFIG_PACKETS_MAX_CONTAINER_PACKET_SIZE</a>	126
4.2.3	<a href="#">Typedef Documentation</a>	126
4.2.3.1	<a href="#">caerDeviceHandle</a>	126
4.2.4	<a href="#">Function Documentation</a>	126
4.2.4.1	<a href="#">caerDeviceClose()</a>	126
4.2.4.2	<a href="#">caerDeviceConfigGet()</a>	127
4.2.4.3	<a href="#">caerDeviceConfigSet()</a>	127
4.2.4.4	<a href="#">caerDeviceDataGet()</a>	128
4.2.4.5	<a href="#">caerDeviceDataStart()</a>	128
4.2.4.6	<a href="#">caerDeviceDataStop()</a>	129
4.2.4.7	<a href="#">caerDeviceSendDefaultConfig()</a>	129

4.3	devices/dvs128.h File Reference	130
4.3.1	Detailed Description	130
4.3.2	Macro Definition Documentation	131
4.3.2.1	CAER_DEVICE_DVS128	131
4.3.2.2	DVS128_CONFIG_BIAS	131
4.3.2.3	DVS128_CONFIG_BIAS_CAS	131
4.3.2.4	DVS128_CONFIG_BIAS_DIFF	131
4.3.2.5	DVS128_CONFIG_BIAS_DIFFOFF	131
4.3.2.6	DVS128_CONFIG_BIAS_DIFFON	131
4.3.2.7	DVS128_CONFIG_BIAS_FOLL	132
4.3.2.8	DVS128_CONFIG_BIAS_INJGND	132
4.3.2.9	DVS128_CONFIG_BIAS_PR	132
4.3.2.10	DVS128_CONFIG_BIAS_PUX	132
4.3.2.11	DVS128_CONFIG_BIAS_PUY	132
4.3.2.12	DVS128_CONFIG_BIAS_REFR	132
4.3.2.13	DVS128_CONFIG_BIAS_REQ	132
4.3.2.14	DVS128_CONFIG_BIAS_REQPD	133
4.3.2.15	DVS128_CONFIG_DVS	133
4.3.2.16	DVS128_CONFIG_DVS_ARRAY_RESET	133
4.3.2.17	DVS128_CONFIG_DVS_RUN	133
4.3.2.18	DVS128_CONFIG_DVS_TIMESTAMP_RESET	133
4.3.2.19	DVS128_CONFIG_DVS_TS_MASTER	133
4.3.3	Function Documentation	133
4.3.3.1	caerDVS128InfoGet()	133
4.4	devices/dynapse.h File Reference	134
4.4.1	Detailed Description	138
4.4.2	Macro Definition Documentation	139
4.4.2.1	CAER_DEVICE_DYNAPSE	139
4.4.2.2	DYNAPSE_CHIP_DYNAPSE	139
4.4.2.3	DYNAPSE_CONFIG_AER	139

4.4.2.4	DYNAPSE_CONFIG_AER_ACK_DELAY . . . . .	139
4.4.2.5	DYNAPSE_CONFIG_AER_ACK_EXTENSION . . . . .	139
4.4.2.6	DYNAPSE_CONFIG_AER_EXTERNAL_AER_CONTROL . . . . .	139
4.4.2.7	DYNAPSE_CONFIG_AER_RUN . . . . .	140
4.4.2.8	DYNAPSE_CONFIG_AER_WAIT_ON_TRANSFER_STALL . . . . .	140
4.4.2.9	DYNAPSE_CONFIG_BIAS_C0_PULSE_PWLK_P . . . . .	140
4.4.2.10	DYNAPSE_CONFIG_CHIP . . . . .	140
4.4.2.11	DYNAPSE_CONFIG_CHIP_CONTENT . . . . .	140
4.4.2.12	DYNAPSE_CONFIG_CHIP_ID . . . . .	140
4.4.2.13	DYNAPSE_CONFIG_CHIP_REQ_DELAY . . . . .	141
4.4.2.14	DYNAPSE_CONFIG_CHIP_REQ_EXTENSION . . . . .	141
4.4.2.15	DYNAPSE_CONFIG_CHIP_RUN . . . . .	141
4.4.2.16	DYNAPSE_CONFIG_CLEAR_CAM . . . . .	141
4.4.2.17	DYNAPSE_CONFIG_DEFAULT_SRAM . . . . .	141
4.4.2.18	DYNAPSE_CONFIG_DEFAULT_SRAM_EMPTY . . . . .	141
4.4.2.19	DYNAPSE_CONFIG_MONITOR_NEU . . . . .	141
4.4.2.20	DYNAPSE_CONFIG_MUX . . . . .	142
4.4.2.21	DYNAPSE_CONFIG_MUX_DROP_AER_ON_TRANSFER_STALL . . . . .	142
4.4.2.22	DYNAPSE_CONFIG_MUX_FORCE_CHIP_BIAS_ENABLE . . . . .	142
4.4.2.23	DYNAPSE_CONFIG_MUX_RUN . . . . .	142
4.4.2.24	DYNAPSE_CONFIG_MUX_TIMESTAMP_RESET . . . . .	142
4.4.2.25	DYNAPSE_CONFIG_MUX_TIMESTAMP_RUN . . . . .	142
4.4.2.26	DYNAPSE_CONFIG_POISSONSPIKEGEN . . . . .	143
4.4.2.27	DYNAPSE_CONFIG_POISSONSPIKEGEN_CHIPID . . . . .	143
4.4.2.28	DYNAPSE_CONFIG_POISSONSPIKEGEN_RUN . . . . .	143
4.4.2.29	DYNAPSE_CONFIG_POISSONSPIKEGEN_WRITEADDRESS . . . . .	143
4.4.2.30	DYNAPSE_CONFIG_POISSONSPIKEGEN_WRITEDATA . . . . .	143
4.4.2.31	DYNAPSE_CONFIG_SPIKEGEN . . . . .	143
4.4.2.32	DYNAPSE_CONFIG_SPIKEGEN_BASEADDR . . . . .	143
4.4.2.33	DYNAPSE_CONFIG_SPIKEGEN_ISI . . . . .	144

4.4.2.34	DYNAPSE_CONFIG_SPIKEGEN_ISIBASE . . . . .	144
4.4.2.35	DYNAPSE_CONFIG_SPIKEGEN_REPEAT . . . . .	144
4.4.2.36	DYNAPSE_CONFIG_SPIKEGEN_RUN . . . . .	144
4.4.2.37	DYNAPSE_CONFIG_SPIKEGEN_STIMCOUNT . . . . .	144
4.4.2.38	DYNAPSE_CONFIG_SPIKEGEN_VARMODE . . . . .	144
4.4.2.39	DYNAPSE_CONFIG_SRAM . . . . .	144
4.4.2.40	DYNAPSE_CONFIG_SRAM_ADDRESS . . . . .	145
4.4.2.41	DYNAPSE_CONFIG_SRAM_BURSTMODE . . . . .	145
4.4.2.42	DYNAPSE_CONFIG_SRAM_DIRECTION_POS . . . . .	145
4.4.2.43	DYNAPSE_CONFIG_SRAM_READ . . . . .	145
4.4.2.44	DYNAPSE_CONFIG_SRAM_READDATA . . . . .	145
4.4.2.45	DYNAPSE_CONFIG_SRAM_RWCOMMAND . . . . .	145
4.4.2.46	DYNAPSE_CONFIG_SRAM_WRITE . . . . .	146
4.4.2.47	DYNAPSE_CONFIG_SRAM_WRITEDATA . . . . .	146
4.4.2.48	DYNAPSE_CONFIG_SYNAPSERECONFIG . . . . .	146
4.4.2.49	DYNAPSE_CONFIG_SYNAPSERECONFIG_CHIPSELECT . . . . .	146
4.4.2.50	DYNAPSE_CONFIG_SYNAPSERECONFIG_GLOBAKERNEL . . . . .	146
4.4.2.51	DYNAPSE_CONFIG_SYNAPSERECONFIG_RUN . . . . .	146
4.4.2.52	DYNAPSE_CONFIG_SYNAPSERECONFIG_SRAMBASEADDR . . . . .	147
4.4.2.53	DYNAPSE_CONFIG_SYNAPSERECONFIG_USESRAMKERNELS . . . . .	147
4.4.2.54	DYNAPSE_CONFIG_SYSINFO . . . . .	147
4.4.2.55	DYNAPSE_CONFIG_SYSINFO_CHIP_IDENTIFIER . . . . .	147
4.4.2.56	DYNAPSE_CONFIG_SYSINFO_DEVICE_IS_MASTER . . . . .	147
4.4.2.57	DYNAPSE_CONFIG_SYSINFO_LOGIC_CLOCK . . . . .	147
4.4.2.58	DYNAPSE_CONFIG_SYSINFO_LOGIC_VERSION . . . . .	148
4.4.2.59	DYNAPSE_CONFIG_USB . . . . .	148
4.4.2.60	DYNAPSE_CONFIG_USB_EARLY_PACKET_DELAY . . . . .	148
4.4.2.61	DYNAPSE_CONFIG_USB_RUN . . . . .	148
4.4.2.62	DYNAPSE_X4BOARD_COREX . . . . .	148
4.4.2.63	DYNAPSE_X4BOARD_COREY . . . . .	148

4.4.2.64	DYNAPSE_X4BOARD_NEUX . . . . .	148
4.4.2.65	DYNAPSE_X4BOARD_NEUY . . . . .	149
4.4.2.66	DYNAPSE_X4BOARD_NUMCHIPS . . . . .	149
4.4.3	Function Documentation . . . . .	149
4.4.3.1	caerBiasDynapseGenerate() . . . . .	149
4.4.3.2	caerBiasDynapseParse() . . . . .	149
4.4.3.3	caerDynapseCoreAddrToNeuronId() . . . . .	150
4.4.3.4	caerDynapseCoreXYToNeuronId() . . . . .	150
4.4.3.5	caerDynapseGenerateCamBits() . . . . .	150
4.4.3.6	caerDynapseGenerateSramBits() . . . . .	151
4.4.3.7	caerDynapseInfoGet() . . . . .	152
4.4.3.8	caerDynapseSendDataToUSB() . . . . .	152
4.4.3.9	caerDynapseSpikeEventFromXY() . . . . .	153
4.4.3.10	caerDynapseSpikeEventGetX() . . . . .	153
4.4.3.11	caerDynapseSpikeEventGetY() . . . . .	153
4.4.3.12	caerDynapseWriteCam() . . . . .	154
4.4.3.13	caerDynapseWritePoissonSpikeRate() . . . . .	154
4.4.3.14	caerDynapseWriteSram() . . . . .	155
4.4.3.15	caerDynapseWriteSramN() . . . . .	156
4.4.3.16	caerDynapseWriteSramWords() . . . . .	156
4.5	devices/edvs.h File Reference . . . . .	157
4.5.1	Detailed Description . . . . .	158
4.5.2	Macro Definition Documentation . . . . .	158
4.5.2.1	CAER_DEVICE_EDVS . . . . .	158
4.5.2.2	EDVS_CONFIG_BIAS . . . . .	158
4.5.2.3	EDVS_CONFIG_BIAS_CAS . . . . .	158
4.5.2.4	EDVS_CONFIG_BIAS_DIFF . . . . .	158
4.5.2.5	EDVS_CONFIG_BIAS_DIFFOFF . . . . .	158
4.5.2.6	EDVS_CONFIG_BIAS_DIFFON . . . . .	158
4.5.2.7	EDVS_CONFIG_BIAS_FOLL . . . . .	159

4.5.2.8	EDVS_CONFIG_BIAS_INJGND	159
4.5.2.9	EDVS_CONFIG_BIAS_PR	159
4.5.2.10	EDVS_CONFIG_BIAS_PUX	159
4.5.2.11	EDVS_CONFIG_BIAS_PUY	159
4.5.2.12	EDVS_CONFIG_BIAS_REFR	159
4.5.2.13	EDVS_CONFIG_BIAS_REQ	159
4.5.2.14	EDVS_CONFIG_BIAS_REQPD	160
4.5.2.15	EDVS_CONFIG_DVS	160
4.5.2.16	EDVS_CONFIG_DVS_RUN	160
4.5.2.17	EDVS_CONFIG_DVS_TIMESTAMP_RESET	160
4.5.3	Function Documentation	160
4.5.3.1	caerEDVSInfoGet()	160
4.6	devices/serial.h File Reference	161
4.6.1	Detailed Description	161
4.6.2	Macro Definition Documentation	161
4.6.2.1	CAER_HOST_CONFIG_SERIAL	161
4.6.2.2	CAER_HOST_CONFIG_SERIAL_BAUD_RATE_12M	161
4.6.2.3	CAER_HOST_CONFIG_SERIAL_BAUD_RATE_2M	162
4.6.2.4	CAER_HOST_CONFIG_SERIAL_BAUD_RATE_4M	162
4.6.2.5	CAER_HOST_CONFIG_SERIAL_BAUD_RATE_8M	162
4.6.2.6	CAER_HOST_CONFIG_SERIAL_READ_SIZE	162
4.6.3	Function Documentation	162
4.6.3.1	caerDeviceOpenSerial()	162
4.7	devices/usb.h File Reference	163
4.7.1	Detailed Description	163
4.7.2	Macro Definition Documentation	163
4.7.2.1	CAER_HOST_CONFIG_USB	163
4.7.2.2	CAER_HOST_CONFIG_USB_BUFFER_NUMBER	163
4.7.2.3	CAER_HOST_CONFIG_USB_BUFFER_SIZE	164
4.7.3	Function Documentation	164



4.7.3.1	<a href="#">caerDeviceOpen()</a>	164
4.8	<a href="#">events/common.h File Reference</a>	164
4.8.1	<a href="#">Detailed Description</a>	166
4.8.2	<a href="#">Macro Definition Documentation</a>	166
4.8.2.1	<a href="#">CAER_DEFAULT_EVENT_TYPES_COUNT</a>	166
4.8.2.2	<a href="#">CAER_EVENT_PACKET_HEADER_SIZE</a>	166
4.8.2.3	<a href="#">CAER_ITERATOR_ALL_END</a>	167
4.8.2.4	<a href="#">CAER_ITERATOR_ALL_START</a>	167
4.8.2.5	<a href="#">CAER_ITERATOR_VALID_END</a>	167
4.8.2.6	<a href="#">CAER_ITERATOR_VALID_START</a>	167
4.8.2.7	<a href="#">TS_OVERFLOW_SHIFT</a>	168
4.8.2.8	<a href="#">VALID_MARK_MASK</a>	168
4.8.2.9	<a href="#">VALID_MARK_SHIFT</a>	168
4.8.3	<a href="#">Typedef Documentation</a>	168
4.8.3.1	<a href="#">caerEventPacketHeader</a>	168
4.8.4	<a href="#">Enumeration Type Documentation</a>	168
4.8.4.1	<a href="#">caer_default_event_types</a>	168
4.8.5	<a href="#">Function Documentation</a>	169
4.8.5.1	<a href="#">caerEventPacketAppend()</a>	169
4.8.5.2	<a href="#">caerEventPacketClean()</a>	169
4.8.5.3	<a href="#">caerEventPacketClear()</a>	170
4.8.5.4	<a href="#">caerEventPacketCopy()</a>	170
4.8.5.5	<a href="#">caerEventPacketCopyOnlyEvents()</a>	170
4.8.5.6	<a href="#">caerEventPacketCopyOnlyValidEvents()</a>	171
4.8.5.7	<a href="#">caerEventPacketEquals()</a>	171
4.8.5.8	<a href="#">caerEventPacketGetDataSize()</a>	171
4.8.5.9	<a href="#">caerEventPacketGetSize()</a>	172
4.8.5.10	<a href="#">caerEventPacketGrow()</a>	172
4.8.5.11	<a href="#">caerEventPacketHeaderGetEventCapacity()</a>	173
4.8.5.12	<a href="#">caerEventPacketHeaderGetEventNumber()</a>	173

4.8.5.13	<a href="#">caerEventPacketHeaderGetEventSize()</a>	173
4.8.5.14	<a href="#">caerEventPacketHeaderGetEventSource()</a>	174
4.8.5.15	<a href="#">caerEventPacketHeaderGetEventTSOffset()</a>	174
4.8.5.16	<a href="#">caerEventPacketHeaderGetEventTSOverflow()</a>	174
4.8.5.17	<a href="#">caerEventPacketHeaderGetEventType()</a>	175
4.8.5.18	<a href="#">caerEventPacketHeaderGetEventValid()</a>	175
4.8.5.19	<a href="#">caerEventPacketHeaderSetEventCapacity()</a>	175
4.8.5.20	<a href="#">caerEventPacketHeaderSetEventNumber()</a>	176
4.8.5.21	<a href="#">caerEventPacketHeaderSetEventSize()</a>	176
4.8.5.22	<a href="#">caerEventPacketHeaderSetEventSource()</a>	176
4.8.5.23	<a href="#">caerEventPacketHeaderSetEventTSOffset()</a>	177
4.8.5.24	<a href="#">caerEventPacketHeaderSetEventTSOverflow()</a>	177
4.8.5.25	<a href="#">caerEventPacketHeaderSetEventType()</a>	177
4.8.5.26	<a href="#">caerEventPacketHeaderSetEventValid()</a>	178
4.8.5.27	<a href="#">caerEventPacketResize()</a>	178
4.8.5.28	<a href="#">caerGenericEventCopy()</a>	179
4.8.5.29	<a href="#">caerGenericEventGetEvent()</a>	179
4.8.5.30	<a href="#">caerGenericEventGetTimestamp()</a>	179
4.8.5.31	<a href="#">caerGenericEventGetTimestamp64()</a>	180
4.8.5.32	<a href="#">caerGenericEventIsValid()</a>	180
4.8.5.33	<a href="#">PACKED_STRUCT()</a>	181
4.9	<a href="#">events/config.h File Reference</a>	181
4.9.1	<a href="#">Detailed Description</a>	182
4.9.2	<a href="#">Macro Definition Documentation</a>	182
4.9.2.1	<a href="#">CAER_CONFIGURATION_CONST_ITERATOR_ALL_START</a>	182
4.9.2.2	<a href="#">CAER_CONFIGURATION_CONST_ITERATOR_VALID_START</a>	183
4.9.2.3	<a href="#">CAER_CONFIGURATION_CONST_REVERSE_ITERATOR_ALL_START</a>	183
4.9.2.4	<a href="#">CAER_CONFIGURATION_CONST_REVERSE_ITERATOR_VALID_START</a>	184
4.9.2.5	<a href="#">CAER_CONFIGURATION_ITERATOR_ALL_END</a>	184
4.9.2.6	<a href="#">CAER_CONFIGURATION_ITERATOR_ALL_START</a>	184

4.9.2.7	CAER_CONFIGURATION_ITERATOR_VALID_END . . . . .	184
4.9.2.8	CAER_CONFIGURATION_ITERATOR_VALID_START . . . . .	185
4.9.2.9	CAER_CONFIGURATION_REVERSE_ITERATOR_ALL_END . . . . .	185
4.9.2.10	CAER_CONFIGURATION_REVERSE_ITERATOR_ALL_START . . . . .	185
4.9.2.11	CAER_CONFIGURATION_REVERSE_ITERATOR_VALID_END . . . . .	185
4.9.2.12	CAER_CONFIGURATION_REVERSE_ITERATOR_VALID_START . . . . .	186
4.9.2.13	CONFIG_MODULE_ADDR_MASK . . . . .	186
4.9.2.14	CONFIG_MODULE_ADDR_SHIFT . . . . .	186
4.9.3	Typedef Documentation . . . . .	186
4.9.3.1	caerConfigurationEvent . . . . .	186
4.9.3.2	caerConfigurationEventPacket . . . . .	186
4.9.4	Function Documentation . . . . .	187
4.9.4.1	caerConfigurationEventGetModuleAddress() . . . . .	187
4.9.4.2	caerConfigurationEventGetParameter() . . . . .	187
4.9.4.3	caerConfigurationEventGetParameterAddress() . . . . .	187
4.9.4.4	caerConfigurationEventGetTimestamp() . . . . .	188
4.9.4.5	caerConfigurationEventGetTimestamp64() . . . . .	188
4.9.4.6	caerConfigurationEventInvalidate() . . . . .	188
4.9.4.7	caerConfigurationEventIsValid() . . . . .	189
4.9.4.8	caerConfigurationEventPacketAllocate() . . . . .	189
4.9.4.9	caerConfigurationEventPacketGetEvent() . . . . .	190
4.9.4.10	caerConfigurationEventPacketGetEventConst() . . . . .	190
4.9.4.11	caerConfigurationEventSetModuleAddress() . . . . .	190
4.9.4.12	caerConfigurationEventSetParameter() . . . . .	191
4.9.4.13	caerConfigurationEventSetParameterAddress() . . . . .	191
4.9.4.14	caerConfigurationEventSetTimestamp() . . . . .	191
4.9.4.15	caerConfigurationEventValidate() . . . . .	192
4.9.4.16	PACKED_STRUCT() [1/2] . . . . .	192
4.9.4.17	PACKED_STRUCT() [2/2] . . . . .	192
4.10	events/ear.h File Reference . . . . .	192

4.10.1 Detailed Description . . . . .	194
4.10.2 Macro Definition Documentation . . . . .	194
4.10.2.1 CAER_EAR_CONST_ITERATOR_ALL_START . . . . .	194
4.10.2.2 CAER_EAR_CONST_ITERATOR_VALID_START . . . . .	194
4.10.2.3 CAER_EAR_CONST_REVERSE_ITERATOR_ALL_START . . . . .	195
4.10.2.4 CAER_EAR_CONST_REVERSE_ITERATOR_VALID_START . . . . .	195
4.10.2.5 CAER_EAR_ITERATOR_ALL_END . . . . .	195
4.10.2.6 CAER_EAR_ITERATOR_ALL_START . . . . .	196
4.10.2.7 CAER_EAR_ITERATOR_VALID_END . . . . .	196
4.10.2.8 CAER_EAR_ITERATOR_VALID_START . . . . .	196
4.10.2.9 CAER_EAR_REVERSE_ITERATOR_ALL_END . . . . .	196
4.10.2.10 CAER_EAR_REVERSE_ITERATOR_ALL_START . . . . .	197
4.10.2.11 CAER_EAR_REVERSE_ITERATOR_VALID_END . . . . .	197
4.10.2.12 CAER_EAR_REVERSE_ITERATOR_VALID_START . . . . .	197
4.10.2.13 EAR_CHANNEL_MASK . . . . .	197
4.10.2.14 EAR_CHANNEL_SHIFT . . . . .	198
4.10.2.15 EAR_FILTER_MASK . . . . .	198
4.10.2.16 EAR_FILTER_SHIFT . . . . .	198
4.10.2.17 EAR_MASK . . . . .	198
4.10.2.18 EAR_NEURON_MASK . . . . .	198
4.10.2.19 EAR_NEURON_SHIFT . . . . .	198
4.10.2.20 EAR_SHIFT . . . . .	199
4.10.3 Typedef Documentation . . . . .	199
4.10.3.1 caerEarEvent . . . . .	199
4.10.3.2 caerEarEventPacket . . . . .	199
4.10.4 Function Documentation . . . . .	199
4.10.4.1 caerEarEventGetChannel() . . . . .	199
4.10.4.2 caerEarEventGetEar() . . . . .	200
4.10.4.3 caerEarEventGetTimestamp() . . . . .	200
4.10.4.4 caerEarEventGetTimestamp64() . . . . .	200

4.10.4.5	<a href="#">caerEarEventInvalidate()</a>	201
4.10.4.6	<a href="#">caerEarEventIsValid()</a>	201
4.10.4.7	<a href="#">caerEarEventPacketAllocate()</a>	201
4.10.4.8	<a href="#">caerEarEventPacketGetEvent()</a>	202
4.10.4.9	<a href="#">caerEarEventPacketGetEventConst()</a>	202
4.10.4.10	<a href="#">caerEarEventSetChannel()</a>	203
4.10.4.11	<a href="#">caerEarEventSetEar()</a>	203
4.10.4.12	<a href="#">caerEarEventSetTimestamp()</a>	203
4.10.4.13	<a href="#">caerEarEventValidate()</a>	204
4.10.4.14	<a href="#">PACKED_STRUCT()</a> [1/2]	204
4.10.4.15	<a href="#">PACKED_STRUCT()</a> [2/2]	204
4.11	<a href="#">events/frame.h File Reference</a>	204
4.11.1	<a href="#">Detailed Description</a>	207
4.11.2	<a href="#">Macro Definition Documentation</a>	207
4.11.2.1	<a href="#">CAER_FRAME_CONST_ITERATOR_ALL_START</a>	207
4.11.2.2	<a href="#">CAER_FRAME_CONST_ITERATOR_VALID_START</a>	207
4.11.2.3	<a href="#">CAER_FRAME_CONST_REVERSE_ITERATOR_ALL_START</a>	208
4.11.2.4	<a href="#">CAER_FRAME_CONST_REVERSE_ITERATOR_VALID_START</a>	208
4.11.2.5	<a href="#">CAER_FRAME_ITERATOR_ALL_END</a>	208
4.11.2.6	<a href="#">CAER_FRAME_ITERATOR_ALL_START</a>	209
4.11.2.7	<a href="#">CAER_FRAME_ITERATOR_VALID_END</a>	209
4.11.2.8	<a href="#">CAER_FRAME_ITERATOR_VALID_START</a>	209
4.11.2.9	<a href="#">CAER_FRAME_REVERSE_ITERATOR_ALL_END</a>	209
4.11.2.10	<a href="#">CAER_FRAME_REVERSE_ITERATOR_ALL_START</a>	210
4.11.2.11	<a href="#">CAER_FRAME_REVERSE_ITERATOR_VALID_END</a>	210
4.11.2.12	<a href="#">CAER_FRAME_REVERSE_ITERATOR_VALID_START</a>	210
4.11.2.13	<a href="#">FRAME_COLOR_CHANNELS_MASK</a>	210
4.11.2.14	<a href="#">FRAME_COLOR_CHANNELS_SHIFT</a>	211
4.11.2.15	<a href="#">FRAME_COLOR_FILTER_MASK</a>	211
4.11.2.16	<a href="#">FRAME_COLOR_FILTER_SHIFT</a>	211

4.11.2.17	FRAME_ROI_IDENTIFIER_MASK	211
4.11.2.18	FRAME_ROI_IDENTIFIER_SHIFT	211
4.11.3	Typedef Documentation	212
4.11.3.1	caerFrameEvent	212
4.11.3.2	caerFrameEventPacket	212
4.11.4	Enumeration Type Documentation	212
4.11.4.1	caer_frame_event_color_channels	212
4.11.4.2	caer_frame_event_color_filter	212
4.11.5	Function Documentation	213
4.11.5.1	caerFrameEventGetChannelNumber()	213
4.11.5.2	caerFrameEventGetColorFilter()	213
4.11.5.3	caerFrameEventGetExposureLength()	214
4.11.5.4	caerFrameEventGetLengthX()	214
4.11.5.5	caerFrameEventGetLengthY()	214
4.11.5.6	caerFrameEventGetPixel()	215
4.11.5.7	caerFrameEventGetPixelArrayUnsafe()	215
4.11.5.8	caerFrameEventGetPixelArrayUnsafeConst()	215
4.11.5.9	caerFrameEventGetPixelForChannel()	216
4.11.5.10	caerFrameEventGetPixelForChannelUnsafe()	216
4.11.5.11	caerFrameEventGetPixelsMaxIndex()	217
4.11.5.12	caerFrameEventGetPixelsSize()	217
4.11.5.13	caerFrameEventGetPixelUnsafe()	218
4.11.5.14	caerFrameEventGetPositionX()	218
4.11.5.15	caerFrameEventGetPositionY()	218
4.11.5.16	caerFrameEventGetROIIdentifier()	219
4.11.5.17	caerFrameEventGetTimestamp()	219
4.11.5.18	caerFrameEventGetTimestamp64()	220
4.11.5.19	caerFrameEventGetTSEndOfExposure()	220
4.11.5.20	caerFrameEventGetTSEndOfExposure64()	220
4.11.5.21	caerFrameEventGetTSEndOfFrame()	221

4.11.5.22 caerFrameEventGetTSEndOfFrame64()	221
4.11.5.23 caerFrameEventGetTSSStartOfExposure()	222
4.11.5.24 caerFrameEventGetTSSStartOfExposure64()	222
4.11.5.25 caerFrameEventGetTSSStartOfFrame()	222
4.11.5.26 caerFrameEventGetTSSStartOfFrame64()	223
4.11.5.27 caerFrameEventInvalidate()	223
4.11.5.28 caerFrameEventIsValid()	224
4.11.5.29 caerFrameEventPacketAllocate()	224
4.11.5.30 caerFrameEventPacketGetEvent()	225
4.11.5.31 caerFrameEventPacketGetEventConst()	225
4.11.5.32 caerFrameEventPacketGetPixelsMaxIndex()	225
4.11.5.33 caerFrameEventPacketGetPixelsSize()	226
4.11.5.34 caerFrameEventSetColorFilter()	226
4.11.5.35 caerFrameEventSetLengthXLengthYChannelNumber()	226
4.11.5.36 caerFrameEventSetPixel()	227
4.11.5.37 caerFrameEventSetPixelForChannel()	227
4.11.5.38 caerFrameEventSetPixelForChannelUnsafe()	228
4.11.5.39 caerFrameEventSetPixelUnsafe()	228
4.11.5.40 caerFrameEventSetPositionX()	229
4.11.5.41 caerFrameEventSetPositionY()	229
4.11.5.42 caerFrameEventSetROIIdentifier()	229
4.11.5.43 caerFrameEventSetTSEndOfExposure()	230
4.11.5.44 caerFrameEventSetTSEndOfFrame()	230
4.11.5.45 caerFrameEventSetTSSStartOfExposure()	230
4.11.5.46 caerFrameEventSetTSSStartOfFrame()	230
4.11.5.47 caerFrameEventValidate()	231
4.11.5.48 PACKED_STRUCT() [1/2]	231
4.11.5.49 PACKED_STRUCT() [2/2]	231
4.12 events/imu6.h File Reference	232
4.12.1 Detailed Description	233

4.12.2	Macro Definition Documentation	233
4.12.2.1	CAER_IMU6_CONST_ITERATOR_ALL_START	233
4.12.2.2	CAER_IMU6_CONST_ITERATOR_VALID_START	233
4.12.2.3	CAER_IMU6_CONST_REVERSE_ITERATOR_ALL_START	234
4.12.2.4	CAER_IMU6_CONST_REVERSE_ITERATOR_VALID_START	234
4.12.2.5	CAER_IMU6_ITERATOR_ALL_END	234
4.12.2.6	CAER_IMU6_ITERATOR_ALL_START	235
4.12.2.7	CAER_IMU6_ITERATOR_VALID_END	235
4.12.2.8	CAER_IMU6_ITERATOR_VALID_START	235
4.12.2.9	CAER_IMU6_REVERSE_ITERATOR_ALL_END	235
4.12.2.10	CAER_IMU6_REVERSE_ITERATOR_ALL_START	236
4.12.2.11	CAER_IMU6_REVERSE_ITERATOR_VALID_END	236
4.12.2.12	CAER_IMU6_REVERSE_ITERATOR_VALID_START	236
4.12.3	Typedef Documentation	236
4.12.3.1	caerIMU6Event	237
4.12.3.2	caerIMU6EventPacket	237
4.12.4	Function Documentation	237
4.12.4.1	caerIMU6EventGetAccelX()	237
4.12.4.2	caerIMU6EventGetAccelY()	237
4.12.4.3	caerIMU6EventGetAccelZ()	238
4.12.4.4	caerIMU6EventGetGyroX()	238
4.12.4.5	caerIMU6EventGetGyroY()	238
4.12.4.6	caerIMU6EventGetGyroZ()	239
4.12.4.7	caerIMU6EventGetTemp()	239
4.12.4.8	caerIMU6EventGetTimestamp()	239
4.12.4.9	caerIMU6EventGetTimestamp64()	240
4.12.4.10	caerIMU6EventInvalidate()	240
4.12.4.11	caerIMU6EventIsValid()	241
4.12.4.12	caerIMU6EventPacketAllocate()	241
4.12.4.13	caerIMU6EventPacketGetEvent()	241



4.12.4.14 caerIMU6EventPacketGetEventConst()	242
4.12.4.15 caerIMU6EventSetAccelX()	242
4.12.4.16 caerIMU6EventSetAccelY()	243
4.12.4.17 caerIMU6EventSetAccelZ()	243
4.12.4.18 caerIMU6EventSetGyroX()	243
4.12.4.19 caerIMU6EventSetGyroY()	243
4.12.4.20 caerIMU6EventSetGyroZ()	244
4.12.4.21 caerIMU6EventSetTemp()	244
4.12.4.22 caerIMU6EventSetTimestamp()	244
4.12.4.23 caerIMU6EventValidate()	245
4.12.4.24 PACKED_STRUCT() [1/2]	245
4.12.4.25 PACKED_STRUCT() [2/2]	245
4.13 events/imu9.h File Reference	245
4.13.1 Detailed Description	247
4.13.2 Macro Definition Documentation	247
4.13.2.1 CAER_IMU9_CONST_ITERATOR_ALL_START	247
4.13.2.2 CAER_IMU9_CONST_ITERATOR_VALID_START	247
4.13.2.3 CAER_IMU9_CONST_REVERSE_ITERATOR_ALL_START	248
4.13.2.4 CAER_IMU9_CONST_REVERSE_ITERATOR_VALID_START	248
4.13.2.5 CAER_IMU9_ITERATOR_ALL_END	248
4.13.2.6 CAER_IMU9_ITERATOR_ALL_START	249
4.13.2.7 CAER_IMU9_ITERATOR_VALID_END	249
4.13.2.8 CAER_IMU9_ITERATOR_VALID_START	249
4.13.2.9 CAER_IMU9_REVERSE_ITERATOR_ALL_END	249
4.13.2.10 CAER_IMU9_REVERSE_ITERATOR_ALL_START	250
4.13.2.11 CAER_IMU9_REVERSE_ITERATOR_VALID_END	250
4.13.2.12 CAER_IMU9_REVERSE_ITERATOR_VALID_START	250
4.13.3 Typedef Documentation	250
4.13.3.1 caerIMU9Event	251
4.13.3.2 caerIMU9EventPacket	251

4.13.4	Function Documentation	251
4.13.4.1	caerIMU9EventGetAccelX()	251
4.13.4.2	caerIMU9EventGetAccelY()	251
4.13.4.3	caerIMU9EventGetAccelZ()	252
4.13.4.4	caerIMU9EventGetCompX()	252
4.13.4.5	caerIMU9EventGetCompY()	252
4.13.4.6	caerIMU9EventGetCompZ()	253
4.13.4.7	caerIMU9EventGetGyroX()	253
4.13.4.8	caerIMU9EventGetGyroY()	253
4.13.4.9	caerIMU9EventGetGyroZ()	255
4.13.4.10	caerIMU9EventGetTemp()	255
4.13.4.11	caerIMU9EventGetTimestamp()	255
4.13.4.12	caerIMU9EventGetTimestamp64()	256
4.13.4.13	caerIMU9EventInvalidate()	256
4.13.4.14	caerIMU9EventIsValid()	257
4.13.4.15	caerIMU9EventPacketAllocate()	257
4.13.4.16	caerIMU9EventPacketGetEvent()	257
4.13.4.17	caerIMU9EventPacketGetEventConst()	258
4.13.4.18	caerIMU9EventSetAccelX()	258
4.13.4.19	caerIMU9EventSetAccelY()	259
4.13.4.20	caerIMU9EventSetAccelZ()	259
4.13.4.21	caerIMU9EventSetCompX()	259
4.13.4.22	caerIMU9EventSetCompY()	259
4.13.4.23	caerIMU9EventSetCompZ()	260
4.13.4.24	caerIMU9EventSetGyroX()	260
4.13.4.25	caerIMU9EventSetGyroY()	260
4.13.4.26	caerIMU9EventSetGyroZ()	261
4.13.4.27	caerIMU9EventSetTemp()	261
4.13.4.28	caerIMU9EventSetTimestamp()	261
4.13.4.29	caerIMU9EventValidate()	262

4.13.4.30	PACKED_STRUCT() [1/2]	262
4.13.4.31	PACKED_STRUCT() [2/2]	262
4.14	events/packetContainer.h File Reference	262
4.14.1	Detailed Description	264
4.14.2	Macro Definition Documentation	264
4.14.2.1	CAER_EVENT_PACKET_CONTAINER_CONST_ITERATOR_START	264
4.14.2.2	CAER_EVENT_PACKET_CONTAINER_ITERATOR_END	265
4.14.2.3	CAER_EVENT_PACKET_CONTAINER_ITERATOR_START	265
4.14.3	Typedef Documentation	265
4.14.3.1	caerEventPacketContainer	265
4.14.4	Function Documentation	265
4.14.4.1	caerEventPacketContainerAllocate()	265
4.14.4.2	caerEventPacketContainerCopyAllEvents()	266
4.14.4.3	caerEventPacketContainerCopyValidEvents()	266
4.14.4.4	caerEventPacketContainerFindEventPacketByType()	267
4.14.4.5	caerEventPacketContainerFindEventPacketByTypeConst()	267
4.14.4.6	caerEventPacketContainerFree()	267
4.14.4.7	caerEventPacketContainerGetEventPacket()	268
4.14.4.8	caerEventPacketContainerGetEventPacketConst()	268
4.14.4.9	caerEventPacketContainerGetEventPacketsNumber()	268
4.14.4.10	caerEventPacketContainerGetEventsNumber()	269
4.14.4.11	caerEventPacketContainerGetEventsValidNumber()	269
4.14.4.12	caerEventPacketContainerGetHighestEventTimestamp()	270
4.14.4.13	caerEventPacketContainerGetLowestEventTimestamp()	270
4.14.4.14	caerEventPacketContainerSetEventPacket()	270
4.14.4.15	caerEventPacketContainerSetEventPacketsNumber()	271
4.14.4.16	caerEventPacketContainerUpdateStatistics()	271
4.14.4.17	PACKED_STRUCT()	271
4.15	events/point1d.h File Reference	271
4.15.1	Detailed Description	273

4.15.2	Macro Definition Documentation	273
4.15.2.1	CAER_POINT1D_CONST_ITERATOR_ALL_START	273
4.15.2.2	CAER_POINT1D_CONST_ITERATOR_VALID_START	273
4.15.2.3	CAER_POINT1D_CONST_REVERSE_ITERATOR_ALL_START	274
4.15.2.4	CAER_POINT1D_CONST_REVERSE_ITERATOR_VALID_START	274
4.15.2.5	CAER_POINT1D_ITERATOR_ALL_END	274
4.15.2.6	CAER_POINT1D_ITERATOR_ALL_START	275
4.15.2.7	CAER_POINT1D_ITERATOR_VALID_END	275
4.15.2.8	CAER_POINT1D_ITERATOR_VALID_START	275
4.15.2.9	CAER_POINT1D_REVERSE_ITERATOR_ALL_END	275
4.15.2.10	CAER_POINT1D_REVERSE_ITERATOR_ALL_START	276
4.15.2.11	CAER_POINT1D_REVERSE_ITERATOR_VALID_END	276
4.15.2.12	CAER_POINT1D_REVERSE_ITERATOR_VALID_START	276
4.15.2.13	POINT1D_SCALE_MASK	276
4.15.2.14	POINT1D_SCALE_SHIFT	277
4.15.2.15	POINT1D_TYPE_MASK	277
4.15.2.16	POINT1D_TYPE_SHIFT	277
4.15.3	Typedef Documentation	277
4.15.3.1	caerPoint1DEvent	277
4.15.3.2	caerPoint1DEventPacket	277
4.15.4	Function Documentation	277
4.15.4.1	caerPoint1DEventGetScale()	277
4.15.4.2	caerPoint1DEventGetTimestamp()	278
4.15.4.3	caerPoint1DEventGetTimestamp64()	278
4.15.4.4	caerPoint1DEventGetType()	279
4.15.4.5	caerPoint1DEventGetX()	279
4.15.4.6	caerPoint1DEventInvalidate()	279
4.15.4.7	caerPoint1DEventIsValid()	280
4.15.4.8	caerPoint1DEventPacketAllocate()	280
4.15.4.9	caerPoint1DEventPacketGetEvent()	280

4.15.4.10 caerPoint1DEventPacketGetEventConst()	281
4.15.4.11 caerPoint1DEventSetScale()	281
4.15.4.12 caerPoint1DEventSetTimestamp()	282
4.15.4.13 caerPoint1DEventSetType()	282
4.15.4.14 caerPoint1DEventSetX()	282
4.15.4.15 caerPoint1DEventValidate()	282
4.15.4.16 PACKED_STRUCT() [1/2]	283
4.15.4.17 PACKED_STRUCT() [2/2]	283
4.16 events/point2d.h File Reference	283
4.16.1 Detailed Description	284
4.16.2 Macro Definition Documentation	285
4.16.2.1 CAER_POINT2D_CONST_ITERATOR_ALL_START	285
4.16.2.2 CAER_POINT2D_CONST_ITERATOR_VALID_START	285
4.16.2.3 CAER_POINT2D_CONST_REVERSE_ITERATOR_ALL_START	286
4.16.2.4 CAER_POINT2D_CONST_REVERSE_ITERATOR_VALID_START	286
4.16.2.5 CAER_POINT2D_ITERATOR_ALL_END	286
4.16.2.6 CAER_POINT2D_ITERATOR_ALL_START	287
4.16.2.7 CAER_POINT2D_ITERATOR_VALID_END	287
4.16.2.8 CAER_POINT2D_ITERATOR_VALID_START	287
4.16.2.9 CAER_POINT2D_REVERSE_ITERATOR_ALL_END	287
4.16.2.10 CAER_POINT2D_REVERSE_ITERATOR_ALL_START	288
4.16.2.11 CAER_POINT2D_REVERSE_ITERATOR_VALID_END	288
4.16.2.12 CAER_POINT2D_REVERSE_ITERATOR_VALID_START	288
4.16.2.13 POINT2D_SCALE_MASK	288
4.16.2.14 POINT2D_SCALE_SHIFT	289
4.16.2.15 POINT2D_TYPE_MASK	289
4.16.2.16 POINT2D_TYPE_SHIFT	289
4.16.3 Typedef Documentation	289
4.16.3.1 caerPoint2DEvent	289
4.16.3.2 caerPoint2DEventPacket	289

4.16.4	Function Documentation	289
4.16.4.1	caerPoint2DEventGetScale()	289
4.16.4.2	caerPoint2DEventGetTimestamp()	290
4.16.4.3	caerPoint2DEventGetTimestamp64()	290
4.16.4.4	caerPoint2DEventGetType()	291
4.16.4.5	caerPoint2DEventGetX()	291
4.16.4.6	caerPoint2DEventGetY()	291
4.16.4.7	caerPoint2DEventInvalidate()	292
4.16.4.8	caerPoint2DEventIsValid()	292
4.16.4.9	caerPoint2DEventPacketAllocate()	292
4.16.4.10	caerPoint2DEventPacketGetEvent()	293
4.16.4.11	caerPoint2DEventPacketGetEventConst()	293
4.16.4.12	caerPoint2DEventSetScale()	294
4.16.4.13	caerPoint2DEventSetTimestamp()	294
4.16.4.14	caerPoint2DEventSetType()	294
4.16.4.15	caerPoint2DEventSetX()	295
4.16.4.16	caerPoint2DEventSetY()	295
4.16.4.17	caerPoint2DEventValidate()	295
4.16.4.18	PACKED_STRUCT() [1/2]	296
4.16.4.19	PACKED_STRUCT() [2/2]	296
4.17	events/point3d.h File Reference	296
4.17.1	Detailed Description	297
4.17.2	Macro Definition Documentation	297
4.17.2.1	CAER_POINT3D_CONST_ITERATOR_ALL_START	298
4.17.2.2	CAER_POINT3D_CONST_ITERATOR_VALID_START	298
4.17.2.3	CAER_POINT3D_CONST_REVERSE_ITERATOR_ALL_START	298
4.17.2.4	CAER_POINT3D_CONST_REVERSE_ITERATOR_VALID_START	299
4.17.2.5	CAER_POINT3D_ITERATOR_ALL_END	299
4.17.2.6	CAER_POINT3D_ITERATOR_ALL_START	299
4.17.2.7	CAER_POINT3D_ITERATOR_VALID_END	299

4.17.2.8	CAER_POINT3D_ITERATOR_VALID_START . . . . .	300
4.17.2.9	CAER_POINT3D_REVERSE_ITERATOR_ALL_END . . . . .	300
4.17.2.10	CAER_POINT3D_REVERSE_ITERATOR_ALL_START . . . . .	300
4.17.2.11	CAER_POINT3D_REVERSE_ITERATOR_VALID_END . . . . .	300
4.17.2.12	CAER_POINT3D_REVERSE_ITERATOR_VALID_START . . . . .	301
4.17.2.13	POINT3D_SCALE_MASK . . . . .	301
4.17.2.14	POINT3D_SCALE_SHIFT . . . . .	301
4.17.2.15	POINT3D_TYPE_MASK . . . . .	301
4.17.2.16	POINT3D_TYPE_SHIFT . . . . .	301
4.17.3	Typedef Documentation . . . . .	302
4.17.3.1	caerPoint3DEvent . . . . .	302
4.17.3.2	caerPoint3DEventPacket . . . . .	302
4.17.4	Function Documentation . . . . .	302
4.17.4.1	caerPoint3DEventGetScale() . . . . .	302
4.17.4.2	caerPoint3DEventGetTimestamp() . . . . .	302
4.17.4.3	caerPoint3DEventGetTimestamp64() . . . . .	303
4.17.4.4	caerPoint3DEventGetType() . . . . .	303
4.17.4.5	caerPoint3DEventGetX() . . . . .	304
4.17.4.6	caerPoint3DEventGetY() . . . . .	304
4.17.4.7	caerPoint3DEventGetZ() . . . . .	304
4.17.4.8	caerPoint3DEventInvalidate() . . . . .	305
4.17.4.9	caerPoint3DEventIsValid() . . . . .	305
4.17.4.10	caerPoint3DEventPacketAllocate() . . . . .	305
4.17.4.11	caerPoint3DEventPacketGetEvent() . . . . .	306
4.17.4.12	caerPoint3DEventPacketGetEventConst() . . . . .	306
4.17.4.13	caerPoint3DEventSetScale() . . . . .	306
4.17.4.14	caerPoint3DEventSetTimestamp() . . . . .	307
4.17.4.15	caerPoint3DEventSetType() . . . . .	307
4.17.4.16	caerPoint3DEventSetX() . . . . .	307
4.17.4.17	caerPoint3DEventSetY() . . . . .	308

4.17.4.18 caerPoint3DEventSetZ()	308
4.17.4.19 caerPoint3DEventValidate()	308
4.17.4.20 PACKED_STRUCT() [1/2]	309
4.17.4.21 PACKED_STRUCT() [2/2]	309
4.18 events/point4d.h File Reference	309
4.18.1 Detailed Description	310
4.18.2 Macro Definition Documentation	310
4.18.2.1 CAER_POINT4D_CONST_ITERATOR_ALL_START	311
4.18.2.2 CAER_POINT4D_CONST_ITERATOR_VALID_START	311
4.18.2.3 CAER_POINT4D_CONST_REVERSE_ITERATOR_ALL_START	311
4.18.2.4 CAER_POINT4D_CONST_REVERSE_ITERATOR_VALID_START	312
4.18.2.5 CAER_POINT4D_ITERATOR_ALL_END	312
4.18.2.6 CAER_POINT4D_ITERATOR_ALL_START	312
4.18.2.7 CAER_POINT4D_ITERATOR_VALID_END	312
4.18.2.8 CAER_POINT4D_ITERATOR_VALID_START	313
4.18.2.9 CAER_POINT4D_REVERSE_ITERATOR_ALL_END	313
4.18.2.10 CAER_POINT4D_REVERSE_ITERATOR_ALL_START	313
4.18.2.11 CAER_POINT4D_REVERSE_ITERATOR_VALID_END	313
4.18.2.12 CAER_POINT4D_REVERSE_ITERATOR_VALID_START	314
4.18.2.13 POINT4D_SCALE_MASK	314
4.18.2.14 POINT4D_SCALE_SHIFT	314
4.18.2.15 POINT4D_TYPE_MASK	314
4.18.2.16 POINT4D_TYPE_SHIFT	314
4.18.3 Typedef Documentation	315
4.18.3.1 caerPoint4DEvent	315
4.18.3.2 caerPoint4DEventPacket	315
4.18.4 Function Documentation	315
4.18.4.1 caerPoint4DEventGetScale()	315
4.18.4.2 caerPoint4DEventGetTimestamp()	315
4.18.4.3 caerPoint4DEventGetTimestamp64()	316



4.18.4.4	<a href="#">caerPoint4DEventGetType()</a>	316
4.18.4.5	<a href="#">caerPoint4DEventGetW()</a>	317
4.18.4.6	<a href="#">caerPoint4DEventGetX()</a>	317
4.18.4.7	<a href="#">caerPoint4DEventGetY()</a>	317
4.18.4.8	<a href="#">caerPoint4DEventGetZ()</a>	318
4.18.4.9	<a href="#">caerPoint4DEventInvalidate()</a>	318
4.18.4.10	<a href="#">caerPoint4DEventIsValid()</a>	318
4.18.4.11	<a href="#">caerPoint4DEventPacketAllocate()</a>	319
4.18.4.12	<a href="#">caerPoint4DEventPacketGetEvent()</a>	319
4.18.4.13	<a href="#">caerPoint4DEventPacketGetEventConst()</a>	319
4.18.4.14	<a href="#">caerPoint4DEventSetScale()</a>	320
4.18.4.15	<a href="#">caerPoint4DEventSetTimestamp()</a>	320
4.18.4.16	<a href="#">caerPoint4DEventSetType()</a>	320
4.18.4.17	<a href="#">caerPoint4DEventSetW()</a>	321
4.18.4.18	<a href="#">caerPoint4DEventSetX()</a>	321
4.18.4.19	<a href="#">caerPoint4DEventSetY()</a>	321
4.18.4.20	<a href="#">caerPoint4DEventSetZ()</a>	322
4.18.4.21	<a href="#">caerPoint4DEventValidate()</a>	322
4.18.4.22	<a href="#">PACKED_STRUCT()</a> [1/2]	322
4.18.4.23	<a href="#">PACKED_STRUCT()</a> [2/2]	323
4.19	<a href="#">events/polarity.h File Reference</a>	323
4.19.1	<a href="#">Detailed Description</a>	324
4.19.2	<a href="#">Macro Definition Documentation</a>	324
4.19.2.1	<a href="#">CAER_POLARITY_CONST_ITERATOR_ALL_START</a>	324
4.19.2.2	<a href="#">CAER_POLARITY_CONST_ITERATOR_VALID_START</a>	325
4.19.2.3	<a href="#">CAER_POLARITY_CONST_REVERSE_ITERATOR_ALL_START</a>	325
4.19.2.4	<a href="#">CAER_POLARITY_CONST_REVERSE_ITERATOR_VALID_START</a>	325
4.19.2.5	<a href="#">CAER_POLARITY_ITERATOR_ALL_END</a>	326
4.19.2.6	<a href="#">CAER_POLARITY_ITERATOR_ALL_START</a>	326
4.19.2.7	<a href="#">CAER_POLARITY_ITERATOR_VALID_END</a>	326

4.19.2.8	CAER_POLARITY_ITERATOR_VALID_START . . . . .	326
4.19.2.9	CAER_POLARITY_REVERSE_ITERATOR_ALL_END . . . . .	327
4.19.2.10	CAER_POLARITY_REVERSE_ITERATOR_ALL_START . . . . .	327
4.19.2.11	CAER_POLARITY_REVERSE_ITERATOR_VALID_END . . . . .	327
4.19.2.12	CAER_POLARITY_REVERSE_ITERATOR_VALID_START . . . . .	327
4.19.2.13	POLARITY_MASK . . . . .	328
4.19.2.14	POLARITY_SHIFT . . . . .	328
4.19.2.15	POLARITY_X_ADDR_MASK . . . . .	328
4.19.2.16	POLARITY_X_ADDR_SHIFT . . . . .	328
4.19.2.17	POLARITY_Y_ADDR_MASK . . . . .	328
4.19.2.18	POLARITY_Y_ADDR_SHIFT . . . . .	328
4.19.3	Typedef Documentation . . . . .	328
4.19.3.1	caerPolarityEvent . . . . .	329
4.19.3.2	caerPolarityEventPacket . . . . .	329
4.19.4	Function Documentation . . . . .	329
4.19.4.1	caerPolarityEventGetPolarity() . . . . .	329
4.19.4.2	caerPolarityEventGetTimestamp() . . . . .	329
4.19.4.3	caerPolarityEventGetTimestamp64() . . . . .	330
4.19.4.4	caerPolarityEventGetX() . . . . .	330
4.19.4.5	caerPolarityEventGetY() . . . . .	330
4.19.4.6	caerPolarityEventInvalidate() . . . . .	331
4.19.4.7	caerPolarityEventIsValid() . . . . .	331
4.19.4.8	caerPolarityEventPacketAllocate() . . . . .	331
4.19.4.9	caerPolarityEventPacketGetEvent() . . . . .	332
4.19.4.10	caerPolarityEventPacketGetEventConst() . . . . .	332
4.19.4.11	caerPolarityEventSetPolarity() . . . . .	333
4.19.4.12	caerPolarityEventSetTimestamp() . . . . .	333
4.19.4.13	caerPolarityEventSetX() . . . . .	333
4.19.4.14	caerPolarityEventSetY() . . . . .	334
4.19.4.15	caerPolarityEventValidate() . . . . .	334

4.19.4.16 PACKED_STRUCT() [1/2]	334
4.19.4.17 PACKED_STRUCT() [2/2]	335
4.20 events/sample.h File Reference	335
4.20.1 Detailed Description	336
4.20.2 Macro Definition Documentation	336
4.20.2.1 CAER_SAMPLE_CONST_ITERATOR_ALL_START	336
4.20.2.2 CAER_SAMPLE_CONST_ITERATOR_VALID_START	337
4.20.2.3 CAER_SAMPLE_CONST_REVERSE_ITERATOR_ALL_START	337
4.20.2.4 CAER_SAMPLE_CONST_REVERSE_ITERATOR_VALID_START	337
4.20.2.5 CAER_SAMPLE_ITERATOR_ALL_END	338
4.20.2.6 CAER_SAMPLE_ITERATOR_ALL_START	338
4.20.2.7 CAER_SAMPLE_ITERATOR_VALID_END	338
4.20.2.8 CAER_SAMPLE_ITERATOR_VALID_START	338
4.20.2.9 CAER_SAMPLE_REVERSE_ITERATOR_ALL_END	339
4.20.2.10 CAER_SAMPLE_REVERSE_ITERATOR_ALL_START	339
4.20.2.11 CAER_SAMPLE_REVERSE_ITERATOR_VALID_END	339
4.20.2.12 CAER_SAMPLE_REVERSE_ITERATOR_VALID_START	339
4.20.2.13 SAMPLE_MASK	340
4.20.2.14 SAMPLE_SHIFT	340
4.20.2.15 SAMPLE_TYPE_MASK	340
4.20.2.16 SAMPLE_TYPE_SHIFT	340
4.20.3 Typedef Documentation	340
4.20.3.1 caerSampleEvent	340
4.20.3.2 caerSampleEventPacket	340
4.20.4 Function Documentation	341
4.20.4.1 caerSampleEventGetSample()	341
4.20.4.2 caerSampleEventGetTimestamp()	341
4.20.4.3 caerSampleEventGetTimestamp64()	341
4.20.4.4 caerSampleEventGetType()	342
4.20.4.5 caerSampleEventInvalidate()	342

4.20.4.6	<a href="#">caerSampleEventsValid()</a>	342
4.20.4.7	<a href="#">caerSampleEventPacketAllocate()</a>	343
4.20.4.8	<a href="#">caerSampleEventPacketGetEvent()</a>	343
4.20.4.9	<a href="#">caerSampleEventPacketGetEventConst()</a>	344
4.20.4.10	<a href="#">caerSampleEventSetSample()</a>	344
4.20.4.11	<a href="#">caerSampleEventSetTimestamp()</a>	344
4.20.4.12	<a href="#">caerSampleEventSetType()</a>	345
4.20.4.13	<a href="#">caerSampleEventValidate()</a>	345
4.20.4.14	<a href="#">PACKED_STRUCT()</a> [1/2]	345
4.20.4.15	<a href="#">PACKED_STRUCT()</a> [2/2]	346
4.21	<a href="#">events/special.h File Reference</a>	346
4.21.1	<a href="#">Detailed Description</a>	347
4.21.2	<a href="#">Macro Definition Documentation</a>	347
4.21.2.1	<a href="#">CAER_SPECIAL_CONST_ITERATOR_ALL_START</a>	348
4.21.2.2	<a href="#">CAER_SPECIAL_CONST_ITERATOR_VALID_START</a>	348
4.21.2.3	<a href="#">CAER_SPECIAL_CONST_REVERSE_ITERATOR_ALL_START</a>	348
4.21.2.4	<a href="#">CAER_SPECIAL_CONST_REVERSE_ITERATOR_VALID_START</a>	349
4.21.2.5	<a href="#">CAER_SPECIAL_ITERATOR_ALL_END</a>	349
4.21.2.6	<a href="#">CAER_SPECIAL_ITERATOR_ALL_START</a>	349
4.21.2.7	<a href="#">CAER_SPECIAL_ITERATOR_VALID_END</a>	349
4.21.2.8	<a href="#">CAER_SPECIAL_ITERATOR_VALID_START</a>	350
4.21.2.9	<a href="#">CAER_SPECIAL_REVERSE_ITERATOR_ALL_END</a>	350
4.21.2.10	<a href="#">CAER_SPECIAL_REVERSE_ITERATOR_ALL_START</a>	350
4.21.2.11	<a href="#">CAER_SPECIAL_REVERSE_ITERATOR_VALID_END</a>	350
4.21.2.12	<a href="#">CAER_SPECIAL_REVERSE_ITERATOR_VALID_START</a>	351
4.21.2.13	<a href="#">SPECIAL_DATA_MASK</a>	351
4.21.2.14	<a href="#">SPECIAL_DATA_SHIFT</a>	351
4.21.2.15	<a href="#">SPECIAL_TYPE_MASK</a>	351
4.21.2.16	<a href="#">SPECIAL_TYPE_SHIFT</a>	351
4.21.3	<a href="#">Typedef Documentation</a>	352

4.21.3.1	<a href="#">caerSpecialEvent</a>	352
4.21.3.2	<a href="#">caerSpecialEventPacket</a>	352
4.21.4	<a href="#">Enumeration Type Documentation</a>	352
4.21.4.1	<a href="#">caer_special_event_types</a>	352
4.21.5	<a href="#">Function Documentation</a>	353
4.21.5.1	<a href="#">caerSpecialEventGetData()</a>	353
4.21.5.2	<a href="#">caerSpecialEventGetTimestamp()</a>	353
4.21.5.3	<a href="#">caerSpecialEventGetTimestamp64()</a>	354
4.21.5.4	<a href="#">caerSpecialEventGetType()</a>	354
4.21.5.5	<a href="#">caerSpecialEventInvalidate()</a>	355
4.21.5.6	<a href="#">caerSpecialEventIsValid()</a>	355
4.21.5.7	<a href="#">caerSpecialEventPacketAllocate()</a>	355
4.21.5.8	<a href="#">caerSpecialEventPacketFindEventByType()</a>	356
4.21.5.9	<a href="#">caerSpecialEventPacketFindEventByTypeConst()</a>	356
4.21.5.10	<a href="#">caerSpecialEventPacketFindValidEventByType()</a>	356
4.21.5.11	<a href="#">caerSpecialEventPacketFindValidEventByTypeConst()</a>	357
4.21.5.12	<a href="#">caerSpecialEventPacketGetEvent()</a>	357
4.21.5.13	<a href="#">caerSpecialEventPacketGetEventConst()</a>	358
4.21.5.14	<a href="#">caerSpecialEventSetData()</a>	358
4.21.5.15	<a href="#">caerSpecialEventSetTimestamp()</a>	358
4.21.5.16	<a href="#">caerSpecialEventSetType()</a>	359
4.21.5.17	<a href="#">caerSpecialEventValidate()</a>	359
4.21.5.18	<a href="#">PACKED_STRUCT()</a> [1/2]	359
4.21.5.19	<a href="#">PACKED_STRUCT()</a> [2/2]	360
4.22	<a href="#">events/spike.h File Reference</a>	360
4.22.1	<a href="#">Detailed Description</a>	361
4.22.2	<a href="#">Macro Definition Documentation</a>	361
4.22.2.1	<a href="#">CAER_SPIKE_CONST_ITERATOR_ALL_START</a>	361
4.22.2.2	<a href="#">CAER_SPIKE_CONST_ITERATOR_VALID_START</a>	362
4.22.2.3	<a href="#">CAER_SPIKE_CONST_REVERSE_ITERATOR_ALL_START</a>	362

4.22.2.4	CAER_SPIKE_CONST_REVERSE_ITERATOR_VALID_START . . . . .	362
4.22.2.5	CAER_SPIKE_ITERATOR_ALL_END . . . . .	363
4.22.2.6	CAER_SPIKE_ITERATOR_ALL_START . . . . .	363
4.22.2.7	CAER_SPIKE_ITERATOR_VALID_END . . . . .	363
4.22.2.8	CAER_SPIKE_ITERATOR_VALID_START . . . . .	363
4.22.2.9	CAER_SPIKE_REVERSE_ITERATOR_ALL_END . . . . .	364
4.22.2.10	CAER_SPIKE_REVERSE_ITERATOR_ALL_START . . . . .	364
4.22.2.11	CAER_SPIKE_REVERSE_ITERATOR_VALID_END . . . . .	364
4.22.2.12	CAER_SPIKE_REVERSE_ITERATOR_VALID_START . . . . .	364
4.22.2.13	SPIKE_CHIP_ID_MASK . . . . .	365
4.22.2.14	SPIKE_CHIP_ID_SHIFT . . . . .	365
4.22.2.15	SPIKE_NEURON_ID_MASK . . . . .	365
4.22.2.16	SPIKE_NEURON_ID_SHIFT . . . . .	365
4.22.2.17	SPIKE_SOURCE_CORE_ID_MASK . . . . .	365
4.22.2.18	SPIKE_SOURCE_CORE_ID_SHIFT . . . . .	365
4.22.3	Typedef Documentation . . . . .	365
4.22.3.1	caerSpikeEvent . . . . .	366
4.22.3.2	caerSpikeEventPacket . . . . .	366
4.22.4	Function Documentation . . . . .	366
4.22.4.1	caerSpikeEventGetChipID() . . . . .	366
4.22.4.2	caerSpikeEventGetNeuronID() . . . . .	366
4.22.4.3	caerSpikeEventGetSourceCoreID() . . . . .	367
4.22.4.4	caerSpikeEventGetTimestamp() . . . . .	367
4.22.4.5	caerSpikeEventGetTimestamp64() . . . . .	367
4.22.4.6	caerSpikeEventInvalidate() . . . . .	368
4.22.4.7	caerSpikeEventIsValid() . . . . .	368
4.22.4.8	caerSpikeEventPacketAllocate() . . . . .	368
4.22.4.9	caerSpikeEventPacketGetEvent() . . . . .	369
4.22.4.10	caerSpikeEventPacketGetEventConst() . . . . .	369
4.22.4.11	caerSpikeEventSetChipID() . . . . .	370

4.22.4.12 caerSpikeEventSetNeuronID()	370
4.22.4.13 caerSpikeEventSetSourceCoreID()	370
4.22.4.14 caerSpikeEventSetTimestamp()	371
4.22.4.15 caerSpikeEventValidate()	371
4.22.4.16 PACKED_STRUCT() [1/2]	371
4.22.4.17 PACKED_STRUCT() [2/2]	371
4.23 frame_utils.h File Reference	372
4.23.1 Detailed Description	372
4.24 libcaer.h File Reference	372
4.24.1 Detailed Description	373
4.24.2 Macro Definition Documentation	374
4.24.2.1 CLEAR_NUMBITS16	374
4.24.2.2 CLEAR_NUMBITS32	374
4.24.2.3 CLEAR_NUMBITS8	374
4.24.2.4 GET_NUMBITS16	374
4.24.2.5 GET_NUMBITS32	374
4.24.2.6 GET_NUMBITS8	375
4.24.2.7 I16T	375
4.24.2.8 I32T	375
4.24.2.9 I64T	375
4.24.2.10 I8T	375
4.24.2.11 LIBCAER_HAVE_OPENCV	375
4.24.2.12 LIBCAER_HAVE_SERIALDEV	375
4.24.2.13 LIBCAER_NAME_STRING	376
4.24.2.14 LIBCAER_VERSION	376
4.24.2.15 LIBCAER_VERSION_STRING	376
4.24.2.16 MASK_NUMBITS32	376
4.24.2.17 MASK_NUMBITS64	376
4.24.2.18 SET_NUMBITS16	376
4.24.2.19 SET_NUMBITS32	376

4.24.2.20 SET_NUMBITS8 . . . . .	377
4.24.2.21 SWAP_VAR . . . . .	377
4.24.2.22 U16T . . . . .	377
4.24.2.23 U32T . . . . .	377
4.24.2.24 U64T . . . . .	377
4.24.2.25 U8T . . . . .	377
4.24.3 Function Documentation . . . . .	378
4.24.3.1 caerByteArrayToInteger() . . . . .	378
4.24.3.2 caerIntegerToByteArray() . . . . .	378
4.24.3.3 caerStrEquals() . . . . .	378
4.24.3.4 caerStrEqualsUpTo() . . . . .	380
4.25 log.h File Reference . . . . .	380
4.25.1 Detailed Description . . . . .	381
4.25.2 Enumeration Type Documentation . . . . .	381
4.25.2.1 caer_log_level . . . . .	381
4.25.3 Function Documentation . . . . .	381
4.25.3.1 caerLog() . . . . .	381
4.25.3.2 caerLogFileDescriptorsGetFirst() . . . . .	382
4.25.3.3 caerLogFileDescriptorsGetSecond() . . . . .	382
4.25.3.4 caerLogFileDescriptorsSet() . . . . .	382
4.25.3.5 caerLogLevelGet() . . . . .	383
4.25.3.6 caerLogLevelSet() . . . . .	383
4.25.3.7 caerLogVA() . . . . .	383
4.25.3.8 caerLogVAFull() . . . . .	384
4.26 network.h File Reference . . . . .	384
4.26.1 Detailed Description . . . . .	385
4.27 portable_endian.h File Reference . . . . .	385
4.27.1 Detailed Description . . . . .	385



# Chapter 1

## Data Structure Index

### 1.1 Data Structures

Here are the data structures with brief descriptions:

<a href="#">caer_bias_coarsefine</a>	5
<a href="#">caer_bias_dynapse</a>	5
<a href="#">caer_bias_shiftedsources</a>	6
<a href="#">caer_bias_vdac</a>	7
<a href="#">caer_davis_info</a>	7
<a href="#">caer_dvs128_info</a>	8
<a href="#">caer_dynapse_info</a>	9
<a href="#">caer_edvs_info</a>	10



## Chapter 2

# File Index

### 2.1 File List

Here is a list of all documented files with brief descriptions:

<a href="#">frame_utils.h</a>	372
<a href="#">libcaer.h</a>	372
<a href="#">log.h</a>	380
<a href="#">network.h</a>	384
<a href="#">portable_endian.h</a>	385
<b>ringbuffer.h</b>	<b>??</b>
devices/ <a href="#">davis.h</a>	11
devices/ <a href="#">device.h</a>	124
devices/ <a href="#">dvs128.h</a>	130
devices/ <a href="#">dynapse.h</a>	134
devices/ <a href="#">edvs.h</a>	157
devices/ <a href="#">serial.h</a>	161
devices/ <a href="#">usb.h</a>	163
events/ <a href="#">common.h</a>	164
events/ <a href="#">config.h</a>	181
events/ <a href="#">ear.h</a>	192
events/ <a href="#">frame.h</a>	204
events/ <a href="#">imu6.h</a>	232
events/ <a href="#">imu9.h</a>	245
events/ <a href="#">packetContainer.h</a>	262
events/ <a href="#">point1d.h</a>	271
events/ <a href="#">point2d.h</a>	283
events/ <a href="#">point3d.h</a>	296
events/ <a href="#">point4d.h</a>	309
events/ <a href="#">polarity.h</a>	323
events/ <a href="#">sample.h</a>	335
events/ <a href="#">special.h</a>	346
events/ <a href="#">spike.h</a>	360



## Chapter 3

# Data Structure Documentation

### 3.1 caer\_bias\_coarsefine Struct Reference

```
#include <davis.h>
```

#### Data Fields

- uint8\_t [coarseValue](#)  
*Coarse current, from 0 to 7, creates big variations in output current.*
- uint8\_t [fineValue](#)  
*Fine current, from 0 to 255, creates small variations in output current.*
- bool [enabled](#)  
*Whether this bias is enabled or not.*
- bool [sexN](#)  
*Bias sex: true for 'N' type, false for 'P' type.*
- bool [typeNormal](#)  
*Bias type: true for 'Normal', false for 'Cascode'.*
- bool [currentLevelNormal](#)  
*Bias current level: true for 'Normal', false for 'Low'.*

#### 3.1.1 Detailed Description

On-chip coarse-fine bias current configuration. See '<http://inilabs.com/support/biasing/>' for more details.

The documentation for this struct was generated from the following file:

- devices/[davis.h](#)

### 3.2 caer\_bias\_dynapse Struct Reference

```
#include <dynapse.h>
```

## Data Fields

- uint8\_t [biasAddress](#)  
*Address of bias to configure, see DYNAPSE\_CONFIG\_BIAS\_\* defines.*
- uint8\_t [coarseValue](#)  
*Coarse current, from 0 to 7, creates big variations in output current.*
- uint8\_t [fineValue](#)  
*Fine current, from 0 to 255, creates small variations in output current.*
- bool [enabled](#)  
*Whether this bias is enabled or not.*
- bool [sexN](#)  
*Bias sex: true for 'N' type, false for 'P' type.*
- bool [typeNormal](#)  
*Bias type: true for 'Normal', false for 'Cascode'.*
- bool [biasHigh](#)  
*Bias current level: true for 'HighBias', false for 'LowBias'.*

### 3.2.1 Detailed Description

On-chip coarse-fine bias current configuration for Dynap-se. See '<https://inilabs.com/support/hardware/user-guide>' section 'Neuron's behaviors and parameters tuning'.

The documentation for this struct was generated from the following file:

- [devices/dynapse.h](#)

## 3.3 caer\_bias\_shiftedsources Struct Reference

```
#include <davis.h>
```

## Data Fields

- uint8\_t [refValue](#)  
*Shifted-source bias level, from 0 to 63.*
- uint8\_t [regValue](#)  
*Shifted-source bias current for buffer amplifier, from 0 to 63.*
- enum [caer\\_bias\\_shiftedsources\\_operating\\_mode](#) [operatingMode](#)  
*Shifted-source operating mode (see 'enum caer\_bias\_shiftedsources\_operating\_mode').*
- enum [caer\\_bias\\_shiftedsources\\_voltage\\_level](#) [voltageLevel](#)  
*Shifted-source voltage level (see 'enum caer\_bias\_shiftedsources\_voltage\_level').*

### 3.3.1 Detailed Description

On-chip shifted-source bias current configuration. See '<http://inilabs.com/support/biasing/>' for more details.

The documentation for this struct was generated from the following file:

- [devices/davis.h](#)

## 3.4 caer\_bias\_vdac Struct Reference

```
#include <davis.h>
```

### Data Fields

- uint8\_t [voltageValue](#)  
*Voltage, between 0 and 63, as a fraction of 1/64th of VDD=3.3V.*
- uint8\_t [currentValue](#)  
*Current, between 0 and 7, that drives the voltage.*

### 3.4.1 Detailed Description

On-chip voltage digital-to-analog converter configuration. See '<http://inilabs.com/support/biasing/>' for more details.

The documentation for this struct was generated from the following file:

- devices/[davis.h](#)

## 3.5 caer\_davis\_info Struct Reference

```
#include <davis.h>
```

### Data Fields

- int16\_t [deviceId](#)  
*Unique device identifier. Also 'source' for events.*
- char [deviceSerialNumber](#) [8+1]  
*Device serial number.*
- uint8\_t [deviceUSBBusNumber](#)  
*Device USB bus number.*
- uint8\_t [deviceUSBDeviceAddress](#)  
*Device USB device address.*
- char \* [deviceString](#)  
*Device information string, for logging purposes.*
- int16\_t [logicVersion](#)  
*Logic (FPGA/CPLD) version.*
- bool [deviceIsMaster](#)  
*Whether the device is a time-stamp master or slave.*
- int16\_t [logicClock](#)  
*Clock in MHz for main logic (FPGA/CPLD).*
- int16\_t [adcClock](#)  
*Clock in MHz for ADC/APS logic (FPGA/CPLD).*
- int16\_t [chipID](#)  
*Chip identifier/type.*

- `int16_t dvsSizeX`  
*DVS X axis resolution.*
- `int16_t dvsSizeY`  
*DVS Y axis resolution.*
- `bool dvsHasPixelFilter`  
*Feature test: DVS pixel-level filtering.*
- `bool dvsHasBackgroundActivityFilter`  
*Feature test: DVS Background Activity filter.*
- `bool dvsHasTestEventGenerator`  
*Feature test: fake event generator (testing/debug).*
- `int16_t apsSizeX`  
*APS X axis resolution.*
- `int16_t apsSizeY`  
*APS Y axis resolution.*
- `enum caer_frame_event_color_filter apsColorFilter`  
*APS color filter type.*
- `bool apsHasGlobalShutter`  
*Feature test: APS supports Global Shutter.*
- `bool apsHasQuadROI`  
*Feature test: APS supports Quadruple Region-of-Interest readout.*
- `bool apsHasExternalADC`  
*Feature test: APS supports External ADC for getting the image.*
- `bool apsHasInternalADC`  
*Feature test: APS supports Internal (on-chip) ADC for getting the image.*
- `bool extInputHasGenerator`  
*Feature test: External Input module supports Signal-Generation.*
- `bool extInputHasExtraDetectors`  
*Feature test: External Input module supports extra detectors (1 & 2).*

### 3.5.1 Detailed Description

DAVIS device-related information.

The documentation for this struct was generated from the following file:

- `devices/davis.h`

## 3.6 caer\_dvs128\_info Struct Reference

```
#include <dvs128.h>
```



## Data Fields

- `int16_t deviceId`  
*Unique device identifier. Also 'source' for events.*
- `char deviceSerialNumber [8+1]`  
*Device serial number.*
- `uint8_t deviceUSBBusNumber`  
*Device USB bus number.*
- `uint8_t deviceUSBDeviceAddress`  
*Device USB device address.*
- `char * deviceString`  
*Device information string, for logging purposes.*
- `int16_t logicVersion`  
*Logic (FPGA/CPLD) version.*
- `bool deviceIsMaster`  
*Whether the device is a time-stamp master or slave.*
- `int16_t dvsSizeX`  
*DVS X axis resolution.*
- `int16_t dvsSizeY`  
*DVS Y axis resolution.*

### 3.6.1 Detailed Description

DVS128 device-related information.

The documentation for this struct was generated from the following file:

- `devices/dvs128.h`

## 3.7 caer\_dynapse\_info Struct Reference

```
#include <dynapse.h>
```

## Data Fields

- `int16_t deviceId`  
*Unique device identifier. Also 'source' for events.*
- `char deviceSerialNumber [8+1]`  
*Device serial number.*
- `uint8_t deviceUSBBusNumber`  
*Device USB bus number.*
- `uint8_t deviceUSBDeviceAddress`  
*Device USB device address.*
- `char * deviceString`  
*Device information string, for logging purposes.*
- `int16_t logicVersion`  
*Logic (FPGA/CPLD) version.*
- `bool deviceIsMaster`  
*Whether the device is a time-stamp master or slave.*
- `int16_t logicClock`  
*Clock in MHz for main logic (FPGA/CPLD).*
- `int16_t chipID`  
*Chip identifier/type.*

### 3.7.1 Detailed Description

Dynap-se device-related information.

The documentation for this struct was generated from the following file:

- [devices/dynapse.h](#)

## 3.8 caer\_edvs\_info Struct Reference

```
#include <edvs.h>
```

### Data Fields

- `int16_t deviceId`  
*Unique device identifier. Also 'source' for events.*
- `char * deviceString`  
*Device information string, for logging purposes.*
- `bool deviceIsMaster`  
*Whether the device is a time-stamp master or slave.*
- `int16_t dvsSizeX`  
*DVS X axis resolution.*
- `int16_t dvsSizeY`  
*DVS Y axis resolution.*

### 3.8.1 Detailed Description

EDVS device-related information.

The documentation for this struct was generated from the following file:

- [devices/edvs.h](#)

## Chapter 4

# File Documentation

### 4.1 devices/davis.h File Reference

```
#include "usb.h"
#include "../events/polarity.h"
#include "../events/special.h"
#include "../events/frame.h"
#include "../events/imu6.h"
#include "../events/sample.h"
```

#### Data Structures

- struct [caer\\_davis\\_info](#)
- struct [caer\\_bias\\_vdac](#)
- struct [caer\\_bias\\_coarsefine](#)
- struct [caer\\_bias\\_shiftedsources](#)

#### Macros

- #define [CAER\\_DEVICE\\_DAVIS\\_FX2](#) 1
- #define [CAER\\_DEVICE\\_DAVIS\\_FX3](#) 2
- #define [CAER\\_DEVICE\\_DAVIS](#) 4
- #define [DAVIS\\_CHIP\\_DAVIS240A](#) 0
- #define [DAVIS\\_CHIP\\_DAVIS240B](#) 1
- #define [DAVIS\\_CHIP\\_DAVIS240C](#) 2
- #define [DAVIS\\_CHIP\\_DAVIS128](#) 3
- #define [DAVIS\\_CHIP\\_DAVIS346A](#) 4
- #define [DAVIS\\_CHIP\\_DAVIS346B](#) 5
- #define [DAVIS\\_CHIP\\_DAVIS640](#) 6
- #define [DAVIS\\_CHIP\\_DAVISRGB](#) 7
- #define [DAVIS\\_CHIP\\_DAVIS208](#) 8
- #define [DAVIS\\_CHIP\\_DAVIS346C](#) 9
- #define [DAVIS\\_CONFIG\\_MUX](#) 0
- #define [DAVIS\\_CONFIG\\_DVS](#) 1
- #define [DAVIS\\_CONFIG\\_APS](#) 2

- `#define DAVIS_CONFIG_IMU` 3
- `#define DAVIS_CONFIG_EXTINPUT` 4
- `#define DAVIS_CONFIG_BIAS` 5
- `#define DAVIS_CONFIG_CHIP` 5
- `#define DAVIS_CONFIG_SYSINFO` 6
- `#define DAVIS_CONFIG_MICROPHONE` 7
- `#define DAVIS_CONFIG_USB` 9
- `#define DAVIS_CONFIG_MUX_RUN` 0
- `#define DAVIS_CONFIG_MUX_TIMESTAMP_RUN` 1
- `#define DAVIS_CONFIG_MUX_TIMESTAMP_RESET` 2
- `#define DAVIS_CONFIG_MUX_FORCE_CHIP_BIAS_ENABLE` 3
- `#define DAVIS_CONFIG_MUX_DROP_DVS_ON_TRANSFER_STALL` 4
- `#define DAVIS_CONFIG_MUX_DROP_APS_ON_TRANSFER_STALL` 5
- `#define DAVIS_CONFIG_MUX_DROP_IMU_ON_TRANSFER_STALL` 6
- `#define DAVIS_CONFIG_MUX_DROP_EXTINPUT_ON_TRANSFER_STALL` 7
- `#define DAVIS_CONFIG_MUX_DROP_MIC_ON_TRANSFER_STALL` 8
- `#define DAVIS_CONFIG_DVS_SIZE_COLUMNS` 0
- `#define DAVIS_CONFIG_DVS_SIZE_ROWS` 1
- `#define DAVIS_CONFIG_DVS_ORIENTATION_INFO` 2
- `#define DAVIS_CONFIG_DVS_RUN` 3
- `#define DAVIS_CONFIG_DVS_ACK_DELAY_ROW` 4
- `#define DAVIS_CONFIG_DVS_ACK_DELAY_COLUMN` 5
- `#define DAVIS_CONFIG_DVS_ACK_EXTENSION_ROW` 6
- `#define DAVIS_CONFIG_DVS_ACK_EXTENSION_COLUMN` 7
- `#define DAVIS_CONFIG_DVS_WAIT_ON_TRANSFER_STALL` 8
- `#define DAVIS_CONFIG_DVS_FILTER_ROW_ONLY_EVENTS` 9
- `#define DAVIS_CONFIG_DVS_EXTERNAL_AER_CONTROL` 10
- `#define DAVIS_CONFIG_DVS_HAS_PIXEL_FILTER` 11
- `#define DAVIS_CONFIG_DVS_FILTER_PIXEL_0_ROW` 12
- `#define DAVIS_CONFIG_DVS_FILTER_PIXEL_0_COLUMN` 13
- `#define DAVIS_CONFIG_DVS_FILTER_PIXEL_1_ROW` 14
- `#define DAVIS_CONFIG_DVS_FILTER_PIXEL_1_COLUMN` 15
- `#define DAVIS_CONFIG_DVS_FILTER_PIXEL_2_ROW` 16
- `#define DAVIS_CONFIG_DVS_FILTER_PIXEL_2_COLUMN` 17
- `#define DAVIS_CONFIG_DVS_FILTER_PIXEL_3_ROW` 18
- `#define DAVIS_CONFIG_DVS_FILTER_PIXEL_3_COLUMN` 19
- `#define DAVIS_CONFIG_DVS_FILTER_PIXEL_4_ROW` 20
- `#define DAVIS_CONFIG_DVS_FILTER_PIXEL_4_COLUMN` 21
- `#define DAVIS_CONFIG_DVS_FILTER_PIXEL_5_ROW` 22
- `#define DAVIS_CONFIG_DVS_FILTER_PIXEL_5_COLUMN` 23
- `#define DAVIS_CONFIG_DVS_FILTER_PIXEL_6_ROW` 24
- `#define DAVIS_CONFIG_DVS_FILTER_PIXEL_6_COLUMN` 25
- `#define DAVIS_CONFIG_DVS_FILTER_PIXEL_7_ROW` 26
- `#define DAVIS_CONFIG_DVS_FILTER_PIXEL_7_COLUMN` 27
- `#define DAVIS_CONFIG_DVS_HAS_BACKGROUND_ACTIVITY_FILTER` 28
- `#define DAVIS_CONFIG_DVS_FILTER_BACKGROUND_ACTIVITY` 29
- `#define DAVIS_CONFIG_DVS_FILTER_BACKGROUND_ACTIVITY_DELTAT` 30
- `#define DAVIS_CONFIG_DVS_HAS_TEST_EVENT_GENERATOR` 31
- `#define DAVIS_CONFIG_DVS_TEST_EVENT_GENERATOR_ENABLE` 32
- `#define DAVIS_CONFIG_APS_SIZE_COLUMNS` 0
- `#define DAVIS_CONFIG_APS_SIZE_ROWS` 1
- `#define DAVIS_CONFIG_APS_ORIENTATION_INFO` 2
- `#define DAVIS_CONFIG_APS_COLOR_FILTER` 3
- `#define DAVIS_CONFIG_APS_RUN` 4
- `#define DAVIS_CONFIG_APS_RESET_READ` 5

- #define [DAVIS\\_CONFIG\\_APS\\_WAIT\\_ON\\_TRANSFER\\_STALL](#) 6
- #define [DAVIS\\_CONFIG\\_APS\\_HAS\\_GLOBAL\\_SHUTTER](#) 7
- #define [DAVIS\\_CONFIG\\_APS\\_GLOBAL\\_SHUTTER](#) 8
- #define [DAVIS\\_CONFIG\\_APS\\_START\\_COLUMN\\_0](#) 9
- #define [DAVIS\\_CONFIG\\_APS\\_START\\_ROW\\_0](#) 10
- #define [DAVIS\\_CONFIG\\_APS\\_END\\_COLUMN\\_0](#) 11
- #define [DAVIS\\_CONFIG\\_APS\\_END\\_ROW\\_0](#) 12
- #define [DAVIS\\_CONFIG\\_APS\\_EXPOSURE](#) 13
- #define [DAVIS\\_CONFIG\\_APS\\_FRAME\\_DELAY](#) 14
- #define [DAVIS\\_CONFIG\\_APS\\_RESET\\_SETTLE](#) 15
- #define [DAVIS\\_CONFIG\\_APS\\_COLUMN\\_SETTLE](#) 16
- #define [DAVIS\\_CONFIG\\_APS\\_ROW\\_SETTLE](#) 17
- #define [DAVIS\\_CONFIG\\_APS\\_NULL\\_SETTLE](#) 18
- #define [DAVIS\\_CONFIG\\_APS\\_HAS\\_QUAD\\_ROI](#) 19
- #define [DAVIS\\_CONFIG\\_APS\\_START\\_COLUMN\\_1](#) 20
- #define [DAVIS\\_CONFIG\\_APS\\_START\\_ROW\\_1](#) 21
- #define [DAVIS\\_CONFIG\\_APS\\_END\\_COLUMN\\_1](#) 22
- #define [DAVIS\\_CONFIG\\_APS\\_END\\_ROW\\_1](#) 23
- #define [DAVIS\\_CONFIG\\_APS\\_START\\_COLUMN\\_2](#) 24
- #define [DAVIS\\_CONFIG\\_APS\\_START\\_ROW\\_2](#) 25
- #define [DAVIS\\_CONFIG\\_APS\\_END\\_COLUMN\\_2](#) 26
- #define [DAVIS\\_CONFIG\\_APS\\_END\\_ROW\\_2](#) 27
- #define [DAVIS\\_CONFIG\\_APS\\_START\\_COLUMN\\_3](#) 28
- #define [DAVIS\\_CONFIG\\_APS\\_START\\_ROW\\_3](#) 29
- #define [DAVIS\\_CONFIG\\_APS\\_END\\_COLUMN\\_3](#) 30
- #define [DAVIS\\_CONFIG\\_APS\\_END\\_ROW\\_3](#) 31
- #define [DAVIS\\_CONFIG\\_APS\\_HAS\\_EXTERNAL\\_ADC](#) 32
- #define [DAVIS\\_CONFIG\\_APS\\_HAS\\_INTERNAL\\_ADC](#) 33
- #define [DAVIS\\_CONFIG\\_APS\\_USE\\_INTERNAL\\_ADC](#) 34
- #define [DAVIS\\_CONFIG\\_APS\\_SAMPLE\\_ENABLE](#) 35
- #define [DAVIS\\_CONFIG\\_APS\\_SAMPLE\\_SETTLE](#) 36
- #define [DAVIS\\_CONFIG\\_APS\\_RAMP\\_RESET](#) 37
- #define [DAVIS\\_CONFIG\\_APS\\_RAMP\\_SHORT\\_RESET](#) 38
- #define [DAVIS\\_CONFIG\\_APS\\_ADC\\_TEST\\_MODE](#) 39
- #define [DAVISRGB\\_CONFIG\\_APS\\_TRANSFER](#) 50
- #define [DAVISRGB\\_CONFIG\\_APS\\_RSFDSETTLE](#) 51
- #define [DAVISRGB\\_CONFIG\\_APS\\_GSPDRESET](#) 52
- #define [DAVISRGB\\_CONFIG\\_APS\\_GSRESETFALL](#) 53
- #define [DAVISRGB\\_CONFIG\\_APS\\_GSTXFALL](#) 54
- #define [DAVISRGB\\_CONFIG\\_APS\\_GSFDRESET](#) 55
- #define [DAVIS\\_CONFIG\\_APS\\_SNAPSHOT](#) 80
- #define [DAVIS\\_CONFIG\\_APS\\_AUTOEXPOSURE](#) 81
- #define [DAVIS\\_CONFIG\\_IMU\\_RUN](#) 0
- #define [DAVIS\\_CONFIG\\_IMU\\_TEMP\\_STANDBY](#) 1
- #define [DAVIS\\_CONFIG\\_IMU\\_ACCEL\\_STANDBY](#) 2
- #define [DAVIS\\_CONFIG\\_IMU\\_GYRO\\_STANDBY](#) 3
- #define [DAVIS\\_CONFIG\\_IMU\\_LP\\_CYCLE](#) 4
- #define [DAVIS\\_CONFIG\\_IMU\\_LP\\_WAKEUP](#) 5
- #define [DAVIS\\_CONFIG\\_IMU\\_SAMPLE\\_RATE\\_DIVIDER](#) 6
- #define [DAVIS\\_CONFIG\\_IMU\\_DIGITAL\\_LOW\\_PASS\\_FILTER](#) 7
- #define [DAVIS\\_CONFIG\\_IMU\\_ACCEL\\_FULL\\_SCALE](#) 8
- #define [DAVIS\\_CONFIG\\_IMU\\_GYRO\\_FULL\\_SCALE](#) 9
- #define [DAVIS\\_CONFIG\\_IMU\\_ORIENTATION\\_INFO](#) 10
- #define [DAVIS\\_CONFIG\\_EXTINPUT\\_RUN\\_DETECTOR](#) 0
- #define [DAVIS\\_CONFIG\\_EXTINPUT\\_DETECT\\_RISING\\_EDGES](#) 1

- #define [DAVIS\\_CONFIG\\_EXTINPUT\\_DETECT\\_FALLING\\_EDGES](#) 2
  - #define [DAVIS\\_CONFIG\\_EXTINPUT\\_DETECT\\_PULSES](#) 3
  - #define [DAVIS\\_CONFIG\\_EXTINPUT\\_DETECT\\_PULSE\\_POLARITY](#) 4
  - #define [DAVIS\\_CONFIG\\_EXTINPUT\\_DETECT\\_PULSE\\_LENGTH](#) 5
  - #define [DAVIS\\_CONFIG\\_EXTINPUT\\_HAS\\_GENERATOR](#) 6
  - #define [DAVIS\\_CONFIG\\_EXTINPUT\\_RUN\\_GENERATOR](#) 7
  - #define [DAVIS\\_CONFIG\\_EXTINPUT\\_GENERATE\\_USE\\_CUSTOM\\_SIGNAL](#) 8
  - #define [DAVIS\\_CONFIG\\_EXTINPUT\\_GENERATE\\_PULSE\\_POLARITY](#) 9
  - #define [DAVIS\\_CONFIG\\_EXTINPUT\\_GENERATE\\_PULSE\\_INTERVAL](#) 10
  - #define [DAVIS\\_CONFIG\\_EXTINPUT\\_GENERATE\\_PULSE\\_LENGTH](#) 11
  - #define [DAVIS\\_CONFIG\\_EXTINPUT\\_GENERATE\\_INJECT\\_ON\\_RISING\\_EDGE](#) 12
  - #define [DAVIS\\_CONFIG\\_EXTINPUT\\_GENERATE\\_INJECT\\_ON\\_FALLING\\_EDGE](#) 13
  - #define [DAVIS\\_CONFIG\\_EXTINPUT\\_HAS\\_EXTRA\\_DETECTORS](#) 14
  - #define [DAVIS\\_CONFIG\\_EXTINPUT\\_RUN\\_DETECTOR1](#) 15
  - #define [DAVIS\\_CONFIG\\_EXTINPUT\\_DETECT\\_RISING\\_EDGES1](#) 16
  - #define [DAVIS\\_CONFIG\\_EXTINPUT\\_DETECT\\_FALLING\\_EDGES1](#) 17
  - #define [DAVIS\\_CONFIG\\_EXTINPUT\\_DETECT\\_PULSES1](#) 18
  - #define [DAVIS\\_CONFIG\\_EXTINPUT\\_DETECT\\_PULSE\\_POLARITY1](#) 19
  - #define [DAVIS\\_CONFIG\\_EXTINPUT\\_DETECT\\_PULSE\\_LENGTH1](#) 20
  - #define [DAVIS\\_CONFIG\\_EXTINPUT\\_RUN\\_DETECTOR2](#) 21
  - #define [DAVIS\\_CONFIG\\_EXTINPUT\\_DETECT\\_RISING\\_EDGES2](#) 22
  - #define [DAVIS\\_CONFIG\\_EXTINPUT\\_DETECT\\_FALLING\\_EDGES2](#) 23
  - #define [DAVIS\\_CONFIG\\_EXTINPUT\\_DETECT\\_PULSES2](#) 24
  - #define [DAVIS\\_CONFIG\\_EXTINPUT\\_DETECT\\_PULSE\\_POLARITY2](#) 25
  - #define [DAVIS\\_CONFIG\\_EXTINPUT\\_DETECT\\_PULSE\\_LENGTH2](#) 26
  - #define [DAVIS\\_CONFIG\\_SYSINFO\\_LOGIC\\_VERSION](#) 0
  - #define [DAVIS\\_CONFIG\\_SYSINFO\\_CHIP\\_IDENTIFIER](#) 1
  - #define [DAVIS\\_CONFIG\\_SYSINFO\\_DEVICE\\_IS\\_MASTER](#) 2
  - #define [DAVIS\\_CONFIG\\_SYSINFO\\_LOGIC\\_CLOCK](#) 3
  - #define [DAVIS\\_CONFIG\\_SYSINFO\\_ADC\\_CLOCK](#) 4
  - #define [DAVIS\\_CONFIG\\_MICROPHONE\\_RUN](#) 0
  - #define [DAVIS\\_CONFIG\\_MICROPHONE\\_SAMPLE\\_FREQUENCY](#) 1
  - #define [DAVIS\\_CONFIG\\_USB\\_RUN](#) 0
  - #define [DAVIS\\_CONFIG\\_USB\\_EARLY\\_PACKET\\_DELAY](#) 1
- 
- #define [IS\\_DAVIS128\(chipID\)](#) ((chipID) == [DAVIS\\_CHIP\\_DAVIS128](#))
  - #define [IS\\_DAVIS208\(chipID\)](#) ((chipID) == [DAVIS\\_CHIP\\_DAVIS208](#))
  - #define [IS\\_DAVIS240A\(chipID\)](#) ((chipID) == [DAVIS\\_CHIP\\_DAVIS240A](#))
  - #define [IS\\_DAVIS240B\(chipID\)](#) ((chipID) == [DAVIS\\_CHIP\\_DAVIS240B](#))
  - #define [IS\\_DAVIS240C\(chipID\)](#) ((chipID) == [DAVIS\\_CHIP\\_DAVIS240C](#))
  - #define [IS\\_DAVIS240\(chipID\)](#) ([IS\\_DAVIS240A\(chipID\)](#) || [IS\\_DAVIS240B\(chipID\)](#) || [IS\\_DAVIS240C\(chipID\)](#))
  - #define [IS\\_DAVIS346A\(chipID\)](#) ((chipID) == [DAVIS\\_CHIP\\_DAVIS346A](#))
  - #define [IS\\_DAVIS346B\(chipID\)](#) ((chipID) == [DAVIS\\_CHIP\\_DAVIS346B](#))
  - #define [IS\\_DAVIS346C\(chipID\)](#) ((chipID) == [DAVIS\\_CHIP\\_DAVIS346C](#))
  - #define [IS\\_DAVIS346\(chipID\)](#) ([IS\\_DAVIS346A\(chipID\)](#) || [IS\\_DAVIS346B\(chipID\)](#) || [IS\\_DAVIS346C\(chipID\)](#))
  - #define [IS\\_DAVIS640\(chipID\)](#) ((chipID) == [DAVIS\\_CHIP\\_DAVIS640](#))
  - #define [IS\\_DAVISRGB\(chipID\)](#) ((chipID) == [DAVIS\\_CHIP\\_DAVISRGB](#))

- #define [DAVIS128\\_CONFIG\\_BIAS\\_APSOVERFLOWLEVEL](#) 0
  - #define [DAVIS128\\_CONFIG\\_BIAS\\_APSCAS](#) 1
  - #define [DAVIS128\\_CONFIG\\_BIAS\\_ADCREFHIGH](#) 2
  - #define [DAVIS128\\_CONFIG\\_BIAS\\_ADCREFLOW](#) 3
  - #define [DAVIS128\\_CONFIG\\_BIAS\\_LOCALBUFBN](#) 8
  - #define [DAVIS128\\_CONFIG\\_BIAS\\_PADFOLLBN](#) 9
  - #define [DAVIS128\\_CONFIG\\_BIAS\\_DIFFBN](#) 10
  - #define [DAVIS128\\_CONFIG\\_BIAS\\_ONBN](#) 11
  - #define [DAVIS128\\_CONFIG\\_BIAS\\_OFFBN](#) 12
  - #define [DAVIS128\\_CONFIG\\_BIAS\\_PIXINBN](#) 13
  - #define [DAVIS128\\_CONFIG\\_BIAS\\_PRBP](#) 14
  - #define [DAVIS128\\_CONFIG\\_BIAS\\_PRSBP](#) 15
  - #define [DAVIS128\\_CONFIG\\_BIAS\\_REFRBP](#) 16
  - #define [DAVIS128\\_CONFIG\\_BIAS\\_READOUTBUFBP](#) 17
  - #define [DAVIS128\\_CONFIG\\_BIAS\\_APSROSBN](#) 18
  - #define [DAVIS128\\_CONFIG\\_BIAS\\_ADCCOMPBP](#) 19
  - #define [DAVIS128\\_CONFIG\\_BIAS\\_COLSELLOWBN](#) 20
  - #define [DAVIS128\\_CONFIG\\_BIAS\\_DACBUFBP](#) 21
  - #define [DAVIS128\\_CONFIG\\_BIAS\\_LCOLTIMEOUTBN](#) 22
  - #define [DAVIS128\\_CONFIG\\_BIAS\\_AEPDBN](#) 23
  - #define [DAVIS128\\_CONFIG\\_BIAS\\_AEPUXBP](#) 24
  - #define [DAVIS128\\_CONFIG\\_BIAS\\_AEPUYBP](#) 25
  - #define [DAVIS128\\_CONFIG\\_BIAS\\_IFREFRBN](#) 26
  - #define [DAVIS128\\_CONFIG\\_BIAS\\_IFTHRBN](#) 27
  - #define [DAVIS128\\_CONFIG\\_BIAS\\_BIASBUFFER](#) 34
  - #define [DAVIS128\\_CONFIG\\_BIAS\\_SSP](#) 35
  - #define [DAVIS128\\_CONFIG\\_BIAS\\_SSN](#) 36
- 
- #define [DAVIS128\\_CONFIG\\_CHIP\\_DIGITALMUX0](#) 128
  - #define [DAVIS128\\_CONFIG\\_CHIP\\_DIGITALMUX1](#) 129
  - #define [DAVIS128\\_CONFIG\\_CHIP\\_DIGITALMUX2](#) 130
  - #define [DAVIS128\\_CONFIG\\_CHIP\\_DIGITALMUX3](#) 131
  - #define [DAVIS128\\_CONFIG\\_CHIP\\_ANALOGMUX0](#) 132
  - #define [DAVIS128\\_CONFIG\\_CHIP\\_ANALOGMUX1](#) 133
  - #define [DAVIS128\\_CONFIG\\_CHIP\\_ANALOGMUX2](#) 134
  - #define [DAVIS128\\_CONFIG\\_CHIP\\_BIASMUX0](#) 135
  - #define [DAVIS128\\_CONFIG\\_CHIP\\_RESETCALIBNEURON](#) 136
  - #define [DAVIS128\\_CONFIG\\_CHIP\\_TYPENCALIBNEURON](#) 137
  - #define [DAVIS128\\_CONFIG\\_CHIP\\_RESETTESTPIXEL](#) 138
  - #define [DAVIS128\\_CONFIG\\_CHIP\\_AERNAROW](#) 140
  - #define [DAVIS128\\_CONFIG\\_CHIP\\_USEAOUT](#) 141
  - #define [DAVIS128\\_CONFIG\\_CHIP\\_GLOBAL\\_SHUTTER](#) 142
  - #define [DAVIS128\\_CONFIG\\_CHIP\\_SELECTGRAYCOUNTER](#) 143
- 
- #define [DAVIS208\\_CONFIG\\_BIAS\\_APSOVERFLOWLEVEL](#) 0
  - #define [DAVIS208\\_CONFIG\\_BIAS\\_APSCAS](#) 1
  - #define [DAVIS208\\_CONFIG\\_BIAS\\_ADCREFHIGH](#) 2
  - #define [DAVIS208\\_CONFIG\\_BIAS\\_ADCREFLOW](#) 3

- #define [DAVIS208\\_CONFIG\\_BIAS\\_RESETHIGHPASS](#) 6
  - #define [DAVIS208\\_CONFIG\\_BIAS\\_REFSS](#) 7
  - #define [DAVIS208\\_CONFIG\\_BIAS\\_LOCALBUFBN](#) 8
  - #define [DAVIS208\\_CONFIG\\_BIAS\\_PADFOLLRN](#) 9
  - #define [DAVIS208\\_CONFIG\\_BIAS\\_DIFFBN](#) 10
  - #define [DAVIS208\\_CONFIG\\_BIAS\\_ONBN](#) 11
  - #define [DAVIS208\\_CONFIG\\_BIAS\\_OFFBN](#) 12
  - #define [DAVIS208\\_CONFIG\\_BIAS\\_PIXINVRN](#) 13
  - #define [DAVIS208\\_CONFIG\\_BIAS\\_PRBP](#) 14
  - #define [DAVIS208\\_CONFIG\\_BIAS\\_PRSFBP](#) 15
  - #define [DAVIS208\\_CONFIG\\_BIAS\\_REFRBP](#) 16
  - #define [DAVIS208\\_CONFIG\\_BIAS\\_READOUTBUFBP](#) 17
  - #define [DAVIS208\\_CONFIG\\_BIAS\\_APSROSFBN](#) 18
  - #define [DAVIS208\\_CONFIG\\_BIAS\\_ADCCOMPBP](#) 19
  - #define [DAVIS208\\_CONFIG\\_BIAS\\_COLSELLOWBN](#) 20
  - #define [DAVIS208\\_CONFIG\\_BIAS\\_DACBUFBP](#) 21
  - #define [DAVIS208\\_CONFIG\\_BIAS\\_LCOLTIMEOUTBN](#) 22
  - #define [DAVIS208\\_CONFIG\\_BIAS\\_AEPDBN](#) 23
  - #define [DAVIS208\\_CONFIG\\_BIAS\\_AEPUXBP](#) 24
  - #define [DAVIS208\\_CONFIG\\_BIAS\\_AEPUYBP](#) 25
  - #define [DAVIS208\\_CONFIG\\_BIAS\\_IFREFRBN](#) 26
  - #define [DAVIS208\\_CONFIG\\_BIAS\\_IFTHRBN](#) 27
  - #define [DAVIS208\\_CONFIG\\_BIAS\\_REGBIASBP](#) 28
  - #define [DAVIS208\\_CONFIG\\_BIAS\\_REFSSBN](#) 30
  - #define [DAVIS208\\_CONFIG\\_BIAS\\_BIASBUFFER](#) 34
  - #define [DAVIS208\\_CONFIG\\_BIAS\\_SSP](#) 35
  - #define [DAVIS208\\_CONFIG\\_BIAS\\_SSN](#) 36
- 
- #define [DAVIS208\\_CONFIG\\_CHIP\\_DIGITALMUX0](#) 128
  - #define [DAVIS208\\_CONFIG\\_CHIP\\_DIGITALMUX1](#) 129
  - #define [DAVIS208\\_CONFIG\\_CHIP\\_DIGITALMUX2](#) 130
  - #define [DAVIS208\\_CONFIG\\_CHIP\\_DIGITALMUX3](#) 131
  - #define [DAVIS208\\_CONFIG\\_CHIP\\_ANALOGMUX0](#) 132
  - #define [DAVIS208\\_CONFIG\\_CHIP\\_ANALOGMUX1](#) 133
  - #define [DAVIS208\\_CONFIG\\_CHIP\\_ANALOGMUX2](#) 134
  - #define [DAVIS208\\_CONFIG\\_CHIP\\_BIASMUX0](#) 135
  - #define [DAVIS208\\_CONFIG\\_CHIP\\_RESETCALIBNEURON](#) 136
  - #define [DAVIS208\\_CONFIG\\_CHIP\\_TYPENCALIBNEURON](#) 137
  - #define [DAVIS208\\_CONFIG\\_CHIP\\_RESETTESTPIXEL](#) 138
  - #define [DAVIS208\\_CONFIG\\_CHIP\\_AERNAROW](#) 140
  - #define [DAVIS208\\_CONFIG\\_CHIP\\_USEAOUT](#) 141
  - #define [DAVIS208\\_CONFIG\\_CHIP\\_GLOBAL\\_SHUTTER](#) 142
  - #define [DAVIS208\\_CONFIG\\_CHIP\\_SELECTGRAYCOUNTER](#) 143
  - #define [DAVIS208\\_CONFIG\\_CHIP\\_SELECTPREAMPVG](#) 145
  - #define [DAVIS208\\_CONFIG\\_CHIP\\_SELECTBIASREFSS](#) 146
  - #define [DAVIS208\\_CONFIG\\_CHIP\\_SELECTSENSE](#) 147
  - #define [DAVIS208\\_CONFIG\\_CHIP\\_SELECTPOSB](#) 148
  - #define [DAVIS208\\_CONFIG\\_CHIP\\_SELECTHIGHPASS](#) 149



- #define [DAVIS240\\_CONFIG\\_BIAS\\_DIFFBN](#) 0
  - #define [DAVIS240\\_CONFIG\\_BIAS\\_ONBN](#) 1
  - #define [DAVIS240\\_CONFIG\\_BIAS\\_OFFBN](#) 2
  - #define [DAVIS240\\_CONFIG\\_BIAS\\_APSCASEPC](#) 3
  - #define [DAVIS240\\_CONFIG\\_BIAS\\_DIFFCASBNC](#) 4
  - #define [DAVIS240\\_CONFIG\\_BIAS\\_APSROSFBN](#) 5
  - #define [DAVIS240\\_CONFIG\\_BIAS\\_LOCALBUFBN](#) 6
  - #define [DAVIS240\\_CONFIG\\_BIAS\\_PIXINVBN](#) 7
  - #define [DAVIS240\\_CONFIG\\_BIAS\\_PRBP](#) 8
  - #define [DAVIS240\\_CONFIG\\_BIAS\\_PRSFBN](#) 9
  - #define [DAVIS240\\_CONFIG\\_BIAS\\_REFRBP](#) 10
  - #define [DAVIS240\\_CONFIG\\_BIAS\\_AEPDBN](#) 11
  - #define [DAVIS240\\_CONFIG\\_BIAS\\_LCOLTIMEOUTBN](#) 12
  - #define [DAVIS240\\_CONFIG\\_BIAS\\_AEPUXBP](#) 13
  - #define [DAVIS240\\_CONFIG\\_BIAS\\_AEPUYBP](#) 14
  - #define [DAVIS240\\_CONFIG\\_BIAS\\_IFTHRBN](#) 15
  - #define [DAVIS240\\_CONFIG\\_BIAS\\_IFREFRBN](#) 16
  - #define [DAVIS240\\_CONFIG\\_BIAS\\_PADFOLLBN](#) 17
  - #define [DAVIS240\\_CONFIG\\_BIAS\\_APSEVERFLOWLEVELBN](#) 18
  - #define [DAVIS240\\_CONFIG\\_BIAS\\_BIASBUFFER](#) 19
  - #define [DAVIS240\\_CONFIG\\_BIAS\\_SSP](#) 20
  - #define [DAVIS240\\_CONFIG\\_BIAS\\_SSN](#) 21
- 
- #define [DAVIS240\\_CONFIG\\_CHIP\\_DIGITALMUX0](#) 128
  - #define [DAVIS240\\_CONFIG\\_CHIP\\_DIGITALMUX1](#) 129
  - #define [DAVIS240\\_CONFIG\\_CHIP\\_DIGITALMUX2](#) 130
  - #define [DAVIS240\\_CONFIG\\_CHIP\\_DIGITALMUX3](#) 131
  - #define [DAVIS240\\_CONFIG\\_CHIP\\_ANALOGMUX0](#) 132
  - #define [DAVIS240\\_CONFIG\\_CHIP\\_ANALOGMUX1](#) 133
  - #define [DAVIS240\\_CONFIG\\_CHIP\\_ANALOGMUX2](#) 134
  - #define [DAVIS240\\_CONFIG\\_CHIP\\_BIASMUX0](#) 135
  - #define [DAVIS240\\_CONFIG\\_CHIP\\_RESETCALIBNEURON](#) 136
  - #define [DAVIS240\\_CONFIG\\_CHIP\\_TYPENCALIBNEURON](#) 137
  - #define [DAVIS240\\_CONFIG\\_CHIP\\_RESETTESTPIXEL](#) 138
  - #define [DAVIS240\\_CONFIG\\_CHIP\\_SPECIALPIXELCONTROL](#) 139
  - #define [DAVIS240\\_CONFIG\\_CHIP\\_AERNAROW](#) 140
  - #define [DAVIS240\\_CONFIG\\_CHIP\\_USEAOUT](#) 141
  - #define [DAVIS240\\_CONFIG\\_CHIP\\_GLOBAL\\_SHUTTER](#) 142
- 
- #define [DAVIS346\\_CONFIG\\_BIAS\\_APSEVERFLOWLEVEL](#) 0
  - #define [DAVIS346\\_CONFIG\\_BIAS\\_APSCAS](#) 1
  - #define [DAVIS346\\_CONFIG\\_BIAS\\_ADCREFHIGH](#) 2
  - #define [DAVIS346\\_CONFIG\\_BIAS\\_ADCREFLOW](#) 3
  - #define [DAVIS346\\_CONFIG\\_BIAS\\_ADCTESTVOLTAGE](#) 4
  - #define [DAVIS346\\_CONFIG\\_BIAS\\_LOCALBUFBN](#) 8
  - #define [DAVIS346\\_CONFIG\\_BIAS\\_PADFOLLBN](#) 9
  - #define [DAVIS346\\_CONFIG\\_BIAS\\_DIFFBN](#) 10
  - #define [DAVIS346\\_CONFIG\\_BIAS\\_ONBN](#) 11

- #define [DAVIS346\\_CONFIG\\_BIAS\\_OFFBN](#) 12
  - #define [DAVIS346\\_CONFIG\\_BIAS\\_PIXINVB](#) 13
  - #define [DAVIS346\\_CONFIG\\_BIAS\\_PRBP](#) 14
  - #define [DAVIS346\\_CONFIG\\_BIAS\\_PRFBP](#) 15
  - #define [DAVIS346\\_CONFIG\\_BIAS\\_REFRBP](#) 16
  - #define [DAVIS346\\_CONFIG\\_BIAS\\_READOUTBUFBP](#) 17
  - #define [DAVIS346\\_CONFIG\\_BIAS\\_APSROSFB](#) 18
  - #define [DAVIS346\\_CONFIG\\_BIAS\\_ADCCOMPBP](#) 19
  - #define [DAVIS346\\_CONFIG\\_BIAS\\_COLSELOWBN](#) 20
  - #define [DAVIS346\\_CONFIG\\_BIAS\\_DACBUFBP](#) 21
  - #define [DAVIS346\\_CONFIG\\_BIAS\\_LCOLTIMEOUTBN](#) 22
  - #define [DAVIS346\\_CONFIG\\_BIAS\\_AEPDBN](#) 23
  - #define [DAVIS346\\_CONFIG\\_BIAS\\_AEPUXBP](#) 24
  - #define [DAVIS346\\_CONFIG\\_BIAS\\_AEPUYBP](#) 25
  - #define [DAVIS346\\_CONFIG\\_BIAS\\_IFREFRBN](#) 26
  - #define [DAVIS346\\_CONFIG\\_BIAS\\_IFTHRBN](#) 27
  - #define [DAVIS346\\_CONFIG\\_BIAS\\_BIASBUFFER](#) 34
  - #define [DAVIS346\\_CONFIG\\_BIAS\\_SSP](#) 35
  - #define [DAVIS346\\_CONFIG\\_BIAS\\_SSN](#) 36
- 
- #define [DAVIS346\\_CONFIG\\_CHIP\\_DIGITALMUX0](#) 128
  - #define [DAVIS346\\_CONFIG\\_CHIP\\_DIGITALMUX1](#) 129
  - #define [DAVIS346\\_CONFIG\\_CHIP\\_DIGITALMUX2](#) 130
  - #define [DAVIS346\\_CONFIG\\_CHIP\\_DIGITALMUX3](#) 131
  - #define [DAVIS346\\_CONFIG\\_CHIP\\_ANALOGMUX0](#) 132
  - #define [DAVIS346\\_CONFIG\\_CHIP\\_ANALOGMUX1](#) 133
  - #define [DAVIS346\\_CONFIG\\_CHIP\\_ANALOGMUX2](#) 134
  - #define [DAVIS346\\_CONFIG\\_CHIP\\_BIASMUX0](#) 135
  - #define [DAVIS346\\_CONFIG\\_CHIP\\_RESETCALIBNEURON](#) 136
  - #define [DAVIS346\\_CONFIG\\_CHIP\\_TYPENCALIBNEURON](#) 137
  - #define [DAVIS346\\_CONFIG\\_CHIP\\_RESETTESTPIXEL](#) 138
  - #define [DAVIS346\\_CONFIG\\_CHIP\\_AERNAROW](#) 140
  - #define [DAVIS346\\_CONFIG\\_CHIP\\_USEAOUT](#) 141
  - #define [DAVIS346\\_CONFIG\\_CHIP\\_GLOBAL\\_SHUTTER](#) 142
  - #define [DAVIS346\\_CONFIG\\_CHIP\\_SELECTGRAYCOUNTER](#) 143
  - #define [DAVIS346\\_CONFIG\\_CHIP\\_TESTADC](#) 144
- 
- #define [DAVIS640\\_CONFIG\\_BIAS\\_APSEVERFLOWLEVEL](#) 0
  - #define [DAVIS640\\_CONFIG\\_BIAS\\_APSCAS](#) 1
  - #define [DAVIS640\\_CONFIG\\_BIAS\\_ADCREFHIGH](#) 2
  - #define [DAVIS640\\_CONFIG\\_BIAS\\_ADCREFLOW](#) 3
  - #define [DAVIS640\\_CONFIG\\_BIAS\\_ADCTESTVOLTAGE](#) 4
  - #define [DAVIS640\\_CONFIG\\_BIAS\\_LOCALBUFB](#) 8
  - #define [DAVIS640\\_CONFIG\\_BIAS\\_PADFOLLBN](#) 9
  - #define [DAVIS640\\_CONFIG\\_BIAS\\_DIFFBN](#) 10
  - #define [DAVIS640\\_CONFIG\\_BIAS\\_ONBN](#) 11
  - #define [DAVIS640\\_CONFIG\\_BIAS\\_OFFBN](#) 12
  - #define [DAVIS640\\_CONFIG\\_BIAS\\_PIXINVB](#) 13

- #define [DAVIS640\\_CONFIG\\_BIAS\\_PRBP](#) 14
  - #define [DAVIS640\\_CONFIG\\_BIAS\\_PRSFBP](#) 15
  - #define [DAVIS640\\_CONFIG\\_BIAS\\_REFRBP](#) 16
  - #define [DAVIS640\\_CONFIG\\_BIAS\\_READOUTBUFBP](#) 17
  - #define [DAVIS640\\_CONFIG\\_BIAS\\_APSROSFBP](#) 18
  - #define [DAVIS640\\_CONFIG\\_BIAS\\_ADCCOMPBP](#) 19
  - #define [DAVIS640\\_CONFIG\\_BIAS\\_COLSELLOWBN](#) 20
  - #define [DAVIS640\\_CONFIG\\_BIAS\\_DACBUFBP](#) 21
  - #define [DAVIS640\\_CONFIG\\_BIAS\\_LCOLTIMEOUTBN](#) 22
  - #define [DAVIS640\\_CONFIG\\_BIAS\\_AEPDBN](#) 23
  - #define [DAVIS640\\_CONFIG\\_BIAS\\_AEPUXBP](#) 24
  - #define [DAVIS640\\_CONFIG\\_BIAS\\_AEPUYBP](#) 25
  - #define [DAVIS640\\_CONFIG\\_BIAS\\_IFREFRBN](#) 26
  - #define [DAVIS640\\_CONFIG\\_BIAS\\_IFTHRBN](#) 27
  - #define [DAVIS640\\_CONFIG\\_BIAS\\_BIASBUFFER](#) 34
  - #define [DAVIS640\\_CONFIG\\_BIAS\\_SSP](#) 35
  - #define [DAVIS640\\_CONFIG\\_BIAS\\_SSN](#) 36
- 
- #define [DAVIS640\\_CONFIG\\_CHIP\\_DIGITALMUX0](#) 128
  - #define [DAVIS640\\_CONFIG\\_CHIP\\_DIGITALMUX1](#) 129
  - #define [DAVIS640\\_CONFIG\\_CHIP\\_DIGITALMUX2](#) 130
  - #define [DAVIS640\\_CONFIG\\_CHIP\\_DIGITALMUX3](#) 131
  - #define [DAVIS640\\_CONFIG\\_CHIP\\_ANALOGMUX0](#) 132
  - #define [DAVIS640\\_CONFIG\\_CHIP\\_ANALOGMUX1](#) 133
  - #define [DAVIS640\\_CONFIG\\_CHIP\\_ANALOGMUX2](#) 134
  - #define [DAVIS640\\_CONFIG\\_CHIP\\_BIASMUX0](#) 135
  - #define [DAVIS640\\_CONFIG\\_CHIP\\_RESETCALIBNEURON](#) 136
  - #define [DAVIS640\\_CONFIG\\_CHIP\\_TYPENCALIBNEURON](#) 137
  - #define [DAVIS640\\_CONFIG\\_CHIP\\_RESETTESTPIXEL](#) 138
  - #define [DAVIS640\\_CONFIG\\_CHIP\\_AERNAROW](#) 140
  - #define [DAVIS640\\_CONFIG\\_CHIP\\_USEAOUT](#) 141
  - #define [DAVIS640\\_CONFIG\\_CHIP\\_GLOBAL\\_SHUTTER](#) 142
  - #define [DAVIS640\\_CONFIG\\_CHIP\\_SELECTGRAYCOUNTER](#) 143
  - #define [DAVIS640\\_CONFIG\\_CHIP\\_TESTADC](#) 144
- 
- #define [DAVISRGB\\_CONFIG\\_BIAS\\_APSCAS](#) 0
  - #define [DAVISRGB\\_CONFIG\\_BIAS\\_OVG1LO](#) 1
  - #define [DAVISRGB\\_CONFIG\\_BIAS\\_OVG2LO](#) 2
  - #define [DAVISRGB\\_CONFIG\\_BIAS\\_TX2OVG2HI](#) 3
  - #define [DAVISRGB\\_CONFIG\\_BIAS\\_GND07](#) 4
  - #define [DAVISRGB\\_CONFIG\\_BIAS\\_ADCTESTVOLTAGE](#) 5
  - #define [DAVISRGB\\_CONFIG\\_BIAS\\_ADCREFHIGH](#) 6
  - #define [DAVISRGB\\_CONFIG\\_BIAS\\_ADCREFLOW](#) 7
  - #define [DAVISRGB\\_CONFIG\\_BIAS\\_IFREFRBN](#) 8
  - #define [DAVISRGB\\_CONFIG\\_BIAS\\_IFTHRBN](#) 9
  - #define [DAVISRGB\\_CONFIG\\_BIAS\\_LOCALBUFBP](#) 10
  - #define [DAVISRGB\\_CONFIG\\_BIAS\\_PADFOLLBN](#) 11
  - #define [DAVISRGB\\_CONFIG\\_BIAS\\_PIXINBN](#) 13

- #define [DAVISRGB\\_CONFIG\\_BIAS\\_DIFFBN](#) 14
- #define [DAVISRGB\\_CONFIG\\_BIAS\\_ONBN](#) 15
- #define [DAVISRGB\\_CONFIG\\_BIAS\\_OFFBN](#) 16
- #define [DAVISRGB\\_CONFIG\\_BIAS\\_PRBP](#) 17
- #define [DAVISRGB\\_CONFIG\\_BIAS\\_PRFBP](#) 18
- #define [DAVISRGB\\_CONFIG\\_BIAS\\_REFRBP](#) 19
- #define [DAVISRGB\\_CONFIG\\_BIAS\\_ARRAYBIASBUFFERBN](#) 20
- #define [DAVISRGB\\_CONFIG\\_BIAS\\_ARRAYLOGICBUFFERBN](#) 22
- #define [DAVISRGB\\_CONFIG\\_BIAS\\_FALLTIMEBN](#) 23
- #define [DAVISRGB\\_CONFIG\\_BIAS\\_RISETIMEBP](#) 24
- #define [DAVISRGB\\_CONFIG\\_BIAS\\_READOUTBUFBP](#) 25
- #define [DAVISRGB\\_CONFIG\\_BIAS\\_APSROSFBN](#) 26
- #define [DAVISRGB\\_CONFIG\\_BIAS\\_ADCCOMPBP](#) 27
- #define [DAVISRGB\\_CONFIG\\_BIAS\\_DACBUFBP](#) 28
- #define [DAVISRGB\\_CONFIG\\_BIAS\\_LCOLTIMEOUTBN](#) 30
- #define [DAVISRGB\\_CONFIG\\_BIAS\\_AEPDBN](#) 31
- #define [DAVISRGB\\_CONFIG\\_BIAS\\_AEPUXBP](#) 32
- #define [DAVISRGB\\_CONFIG\\_BIAS\\_AEPUYBP](#) 33
- #define [DAVISRGB\\_CONFIG\\_BIAS\\_BIASBUFFER](#) 34
- #define [DAVISRGB\\_CONFIG\\_BIAS\\_SSP](#) 35
- #define [DAVISRGB\\_CONFIG\\_BIAS\\_SSN](#) 36

- #define [DAVISRGB\\_CONFIG\\_CHIP\\_DIGITALMUX0](#) 128
- #define [DAVISRGB\\_CONFIG\\_CHIP\\_DIGITALMUX1](#) 129
- #define [DAVISRGB\\_CONFIG\\_CHIP\\_DIGITALMUX2](#) 130
- #define [DAVISRGB\\_CONFIG\\_CHIP\\_DIGITALMUX3](#) 131
- #define [DAVISRGB\\_CONFIG\\_CHIP\\_ANALOGMUX0](#) 132
- #define [DAVISRGB\\_CONFIG\\_CHIP\\_ANALOGMUX1](#) 133
- #define [DAVISRGB\\_CONFIG\\_CHIP\\_ANALOGMUX2](#) 134
- #define [DAVISRGB\\_CONFIG\\_CHIP\\_BIASMUX0](#) 135
- #define [DAVISRGB\\_CONFIG\\_CHIP\\_RESETCALIBNEURON](#) 136
- #define [DAVISRGB\\_CONFIG\\_CHIP\\_TYPCALIBNEURON](#) 137
- #define [DAVISRGB\\_CONFIG\\_CHIP\\_RESETTESTPIXEL](#) 138
- #define [DAVISRGB\\_CONFIG\\_CHIP\\_AERNAROW](#) 140
- #define [DAVISRGB\\_CONFIG\\_CHIP\\_USEAOUT](#) 141
- #define [DAVISRGB\\_CONFIG\\_CHIP\\_SELECTGRAYCOUNTER](#) 143
- #define [DAVISRGB\\_CONFIG\\_CHIP\\_TESTADC](#) 144
- #define [DAVISRGB\\_CONFIG\\_CHIP\\_ADJUSTOVG1LO](#) 145
- #define [DAVISRGB\\_CONFIG\\_CHIP\\_ADJUSTOVG2LO](#) 146
- #define [DAVISRGB\\_CONFIG\\_CHIP\\_ADJUSTTX2OVG2HI](#) 147

## Enumerations

- enum [caer\\_bias\\_shiftedsourcesource\\_operating\\_mode](#) { [SHIFTED\\_SOURCE](#) = 0, [HI\\_Z](#) = 1, [TIED\\_TO\\_RAIL](#) = 2 }
- enum [caer\\_bias\\_shiftedsourcesource\\_voltage\\_level](#) { [SPLIT\\_GATE](#) = 0, [SINGLE\\_DIODE](#) = 1, [DOUBLE\\_DIODE](#) = 2 }

## Functions

- struct [caer\\_davis\\_info](#) [caerDavisInfoGet](#) ([caerDeviceHandle](#) handle)
- uint16\_t [caerBiasVDACGenerate](#) (const struct [caer\\_bias\\_vdac](#) vdacBias)
- struct [caer\\_bias\\_vdac](#) [caerBiasVDACParse](#) (const uint16\_t vdacBias)
- uint16\_t [caerBiasCoarseFineGenerate](#) (const struct [caer\\_bias\\_coarsefine](#) coarseFineBias)
- struct [caer\\_bias\\_coarsefine](#) [caerBiasCoarseFineParse](#) (const uint16\_t coarseFineBias)
- uint16\_t [caerBiasShiftedSourceGenerate](#) (const struct [caer\\_bias\\_shiftedsourcesource](#) shiftedSourceBias)
- struct [caer\\_bias\\_shiftedsourcesource](#) [caerBiasShiftedSourceParse](#) (const uint16\_t shiftedSourceBias)

### 4.1.1 Detailed Description

DAVIS specific configuration defines and information structures.

### 4.1.2 Macro Definition Documentation

#### 4.1.2.1 CAER\_DEVICE\_DAVIS

```
#define CAER_DEVICE_DAVIS 4
```

Device type definition for iniLabs DAVIS boards, supporting both FX2 and FX3 generation devices. This is the preferred way to access cameras now.

#### 4.1.2.2 CAER\_DEVICE\_DAVIS\_FX2

```
#define CAER_DEVICE_DAVIS_FX2 1
```

Device type definition for iniLabs DAVIS FX2-based boards, like DAVIS240a/b/c. Deprecated in favor of [CAER\\_DEVICE\\_DAVIS](#).

#### 4.1.2.3 CAER\_DEVICE\_DAVIS\_FX3

```
#define CAER_DEVICE_DAVIS_FX3 2
```

Device type definition for iniLabs DAVIS FX3-based boards, like DAVIS640. Deprecated in favor of [CAER\\_DEVICE\\_DAVIS](#).

#### 4.1.2.4 DAVIS128\_CONFIG\_BIAS\_ADCCOMPBP

```
#define DAVIS128_CONFIG_BIAS_ADCCOMPBP 19
```

Parameter address for module [DAVIS128\\_CONFIG\\_BIAS](#): DAVIS128 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.5 DAVIS128\_CONFIG\_BIAS\_ADCCREFHIGH

```
#define DAVIS128_CONFIG_BIAS_ADCCREFHIGH 2
```

Parameter address for module DAVIS128\_CONFIG\_BIAS: DAVIS128 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.6 DAVIS128\_CONFIG\_BIAS\_ADCCREFLOW

```
#define DAVIS128_CONFIG_BIAS_ADCCREFLOW 3
```

Parameter address for module DAVIS128\_CONFIG\_BIAS: DAVIS128 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.7 DAVIS128\_CONFIG\_BIAS\_AEPDBN

```
#define DAVIS128_CONFIG_BIAS_AEPDBN 23
```

Parameter address for module DAVIS128\_CONFIG\_BIAS: DAVIS128 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.8 DAVIS128\_CONFIG\_BIAS\_AEPUXBP

```
#define DAVIS128_CONFIG_BIAS_AEPUXBP 24
```

Parameter address for module DAVIS128\_CONFIG\_BIAS: DAVIS128 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.9 DAVIS128\_CONFIG\_BIAS\_AEPUYBP

```
#define DAVIS128_CONFIG_BIAS_AEPUYBP 25
```

Parameter address for module DAVIS128\_CONFIG\_BIAS: DAVIS128 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.10 DAVIS128\_CONFIG\_BIAS\_APSCAS

```
#define DAVIS128_CONFIG_BIAS_APSCAS 1
```

Parameter address for module DAVIS128\_CONFIG\_BIAS: DAVIS128 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.11 DAVIS128\_CONFIG\_BIAS\_APSOEVERFLOWLEVEL

```
#define DAVIS128_CONFIG_BIAS_APSOEVERFLOWLEVEL 0
```

Parameter address for module DAVIS128\_CONFIG\_BIAS: DAVIS128 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.12 DAVIS128\_CONFIG\_BIAS\_APSROSFBN

```
#define DAVIS128_CONFIG_BIAS_APSROSFBN 18
```

Parameter address for module DAVIS128\_CONFIG\_BIAS: DAVIS128 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.13 DAVIS128\_CONFIG\_BIAS\_BIASBUFFER

```
#define DAVIS128_CONFIG_BIAS_BIASBUFFER 34
```

Parameter address for module DAVIS128\_CONFIG\_BIAS: DAVIS128 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.



#### 4.1.2.14 DAVIS128\_CONFIG\_BIAS\_COLSELLOWBN

```
#define DAVIS128_CONFIG_BIAS_COLSELLOWBN 20
```

Parameter address for module DAVIS128\_CONFIG\_BIAS: DAVIS128 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.15 DAVIS128\_CONFIG\_BIAS\_DACBUFBP

```
#define DAVIS128_CONFIG_BIAS_DACBUFBP 21
```

Parameter address for module DAVIS128\_CONFIG\_BIAS: DAVIS128 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.16 DAVIS128\_CONFIG\_BIAS\_DIFFBN

```
#define DAVIS128_CONFIG_BIAS_DIFFBN 10
```

Parameter address for module DAVIS128\_CONFIG\_BIAS: DAVIS128 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.17 DAVIS128\_CONFIG\_BIAS\_IFREFRBN

```
#define DAVIS128_CONFIG_BIAS_IFREFRBN 26
```

Parameter address for module DAVIS128\_CONFIG\_BIAS: DAVIS128 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.18 DAVIS128\_CONFIG\_BIAS\_IFTHRBN

```
#define DAVIS128_CONFIG_BIAS_IFTHRBN 27
```

Parameter address for module DAVIS128\_CONFIG\_BIAS: DAVIS128 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.19 DAVIS128\_CONFIG\_BIAS\_LCOLTIMEOUTBN

```
#define DAVIS128_CONFIG_BIAS_LCOLTIMEOUTBN 22
```

Parameter address for module DAVIS128\_CONFIG\_BIAS: DAVIS128 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.20 DAVIS128\_CONFIG\_BIAS\_LOCALBUFBN

```
#define DAVIS128_CONFIG_BIAS_LOCALBUFBN 8
```

Parameter address for module DAVIS128\_CONFIG\_BIAS: DAVIS128 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.21 DAVIS128\_CONFIG\_BIAS\_OFFBN

```
#define DAVIS128_CONFIG_BIAS_OFFBN 12
```

Parameter address for module DAVIS128\_CONFIG\_BIAS: DAVIS128 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.22 DAVIS128\_CONFIG\_BIAS\_ONBN

```
#define DAVIS128_CONFIG_BIAS_ONBN 11
```

Parameter address for module DAVIS128\_CONFIG\_BIAS: DAVIS128 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.23 DAVIS128\_CONFIG\_BIAS\_PADFOLLBN

```
#define DAVIS128_CONFIG_BIAS_PADFOLLBN 9
```

Parameter address for module DAVIS128\_CONFIG\_BIAS: DAVIS128 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.24 DAVIS128\_CONFIG\_BIAS\_PIXINBN

```
#define DAVIS128_CONFIG_BIAS_PIXINBN 13
```

Parameter address for module DAVIS128\_CONFIG\_BIAS: DAVIS128 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.25 DAVIS128\_CONFIG\_BIAS\_PRBP

```
#define DAVIS128_CONFIG_BIAS_PRBP 14
```

Parameter address for module DAVIS128\_CONFIG\_BIAS: DAVIS128 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.26 DAVIS128\_CONFIG\_BIAS\_PRSFBP

```
#define DAVIS128_CONFIG_BIAS_PRSFBP 15
```

Parameter address for module DAVIS128\_CONFIG\_BIAS: DAVIS128 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.27 DAVIS128\_CONFIG\_BIAS\_READOUTBUFBP

```
#define DAVIS128_CONFIG_BIAS_READOUTBUFBP 17
```

Parameter address for module DAVIS128\_CONFIG\_BIAS: DAVIS128 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.28 DAVIS128\_CONFIG\_BIAS\_REFRBP

```
#define DAVIS128_CONFIG_BIAS_REFRBP 16
```

Parameter address for module DAVIS128\_CONFIG\_BIAS: DAVIS128 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.29 DAVIS128\_CONFIG\_BIAS\_SSN

```
#define DAVIS128_CONFIG_BIAS_SSN 36
```

Parameter address for module DAVIS128\_CONFIG\_BIAS: DAVIS128 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.30 DAVIS128\_CONFIG\_BIAS\_SSP

```
#define DAVIS128_CONFIG_BIAS_SSP 35
```

Parameter address for module DAVIS128\_CONFIG\_BIAS: DAVIS128 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.31 DAVIS128\_CONFIG\_CHIP\_AERNAROW

```
#define DAVIS128_CONFIG_CHIP_AERNAROW 140
```

Parameter address for module DAVIS128\_CONFIG\_CHIP: DAVIS128 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.32 DAVIS128\_CONFIG\_CHIP\_ANALOGMUX0

```
#define DAVIS128_CONFIG_CHIP_ANALOGMUX0 132
```

Parameter address for module DAVIS128\_CONFIG\_CHIP: DAVIS128 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.33 DAVIS128\_CONFIG\_CHIP\_ANALOGMUX1

```
#define DAVIS128_CONFIG_CHIP_ANALOGMUX1 133
```

Parameter address for module DAVIS128\_CONFIG\_CHIP: DAVIS128 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.34 DAVIS128\_CONFIG\_CHIP\_ANALOGMUX2

```
#define DAVIS128_CONFIG_CHIP_ANALOGMUX2 134
```

Parameter address for module DAVIS128\_CONFIG\_CHIP: DAVIS128 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.35 DAVIS128\_CONFIG\_CHIP\_BIASMUX0

```
#define DAVIS128_CONFIG_CHIP_BIASMUX0 135
```

Parameter address for module DAVIS128\_CONFIG\_CHIP: DAVIS128 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.36 DAVIS128\_CONFIG\_CHIP\_DIGITALMUX0

```
#define DAVIS128_CONFIG_CHIP_DIGITALMUX0 128
```

Parameter address for module DAVIS128\_CONFIG\_CHIP: DAVIS128 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.37 DAVIS128\_CONFIG\_CHIP\_DIGITALMUX1

```
#define DAVIS128_CONFIG_CHIP_DIGITALMUX1 129
```

Parameter address for module DAVIS128\_CONFIG\_CHIP: DAVIS128 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.38 DAVIS128\_CONFIG\_CHIP\_DIGITALMUX2

```
#define DAVIS128_CONFIG_CHIP_DIGITALMUX2 130
```

Parameter address for module DAVIS128\_CONFIG\_CHIP: DAVIS128 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.39 DAVIS128\_CONFIG\_CHIP\_DIGITALMUX3

```
#define DAVIS128_CONFIG_CHIP_DIGITALMUX3 131
```

Parameter address for module DAVIS128\_CONFIG\_CHIP: DAVIS128 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.40 DAVIS128\_CONFIG\_CHIP\_GLOBAL\_SHUTTER

```
#define DAVIS128_CONFIG_CHIP_GLOBAL_SHUTTER 142
```

Parameter address for module DAVIS128\_CONFIG\_CHIP: DAVIS128 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.41 DAVIS128\_CONFIG\_CHIP\_RESETCALIBNEURON

```
#define DAVIS128_CONFIG_CHIP_RESETCALIBNEURON 136
```

Parameter address for module DAVIS128\_CONFIG\_CHIP: DAVIS128 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.42 DAVIS128\_CONFIG\_CHIP\_RESETTESTPIXEL

```
#define DAVIS128_CONFIG_CHIP_RESETTESTPIXEL 138
```

Parameter address for module DAVIS128\_CONFIG\_CHIP: DAVIS128 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.43 DAVIS128\_CONFIG\_CHIP\_SELECTGRAYCOUNTER

```
#define DAVIS128_CONFIG_CHIP_SELECTGRAYCOUNTER 143
```

Parameter address for module DAVIS128\_CONFIG\_CHIP: DAVIS128 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.44 DAVIS128\_CONFIG\_CHIP\_TYPENCALIBNEURON

```
#define DAVIS128_CONFIG_CHIP_TYPENCALIBNEURON 137
```

Parameter address for module DAVIS128\_CONFIG\_CHIP: DAVIS128 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.



#### 4.1.2.45 DAVIS128\_CONFIG\_CHIP\_USEAOUT

```
#define DAVIS128_CONFIG_CHIP_USEAOUT 141
```

Parameter address for module DAVIS128\_CONFIG\_CHIP: DAVIS128 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.46 DAVIS208\_CONFIG\_BIAS\_ADCCOMPBP

```
#define DAVIS208_CONFIG_BIAS_ADCCOMPBP 19
```

Parameter address for module DAVIS208\_CONFIG\_BIAS: DAVIS208 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.47 DAVIS208\_CONFIG\_BIAS\_ADCREFHIGH

```
#define DAVIS208_CONFIG_BIAS_ADCREFHIGH 2
```

Parameter address for module DAVIS208\_CONFIG\_BIAS: DAVIS208 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.48 DAVIS208\_CONFIG\_BIAS\_ADCREFLOW

```
#define DAVIS208_CONFIG_BIAS_ADCREFLOW 3
```

Parameter address for module DAVIS208\_CONFIG\_BIAS: DAVIS208 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.49 DAVIS208\_CONFIG\_BIAS\_AEPDBN

```
#define DAVIS208_CONFIG_BIAS_AEPDBN 23
```

Parameter address for module DAVIS208\_CONFIG\_BIAS: DAVIS208 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.50 DAVIS208\_CONFIG\_BIAS\_AEPUXBP

```
#define DAVIS208_CONFIG_BIAS_AEPUXBP 24
```

Parameter address for module DAVIS208\_CONFIG\_BIAS: DAVIS208 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.51 DAVIS208\_CONFIG\_BIAS\_AEPUYBP

```
#define DAVIS208_CONFIG_BIAS_AEPUYBP 25
```

Parameter address for module DAVIS208\_CONFIG\_BIAS: DAVIS208 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.52 DAVIS208\_CONFIG\_BIAS\_APSCAS

```
#define DAVIS208_CONFIG_BIAS_APSCAS 1
```

Parameter address for module DAVIS208\_CONFIG\_BIAS: DAVIS208 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.53 DAVIS208\_CONFIG\_BIAS\_APSEVERFLOWLEVEL

```
#define DAVIS208_CONFIG_BIAS_APSEVERFLOWLEVEL 0
```

Parameter address for module DAVIS208\_CONFIG\_BIAS: DAVIS208 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.54 DAVIS208\_CONFIG\_BIAS\_APSROSFBN

```
#define DAVIS208_CONFIG_BIAS_APSROSFBN 18
```

Parameter address for module DAVIS208\_CONFIG\_BIAS: DAVIS208 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.55 DAVIS208\_CONFIG\_BIAS\_BIASBUFFER

```
#define DAVIS208_CONFIG_BIAS_BIASBUFFER 34
```

Parameter address for module DAVIS208\_CONFIG\_BIAS: DAVIS208 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.56 DAVIS208\_CONFIG\_BIAS\_COLSELLOWBN

```
#define DAVIS208_CONFIG_BIAS_COLSELLOWBN 20
```

Parameter address for module DAVIS208\_CONFIG\_BIAS: DAVIS208 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.57 DAVIS208\_CONFIG\_BIAS\_DACBUFBP

```
#define DAVIS208_CONFIG_BIAS_DACBUFBP 21
```

Parameter address for module DAVIS208\_CONFIG\_BIAS: DAVIS208 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.58 DAVIS208\_CONFIG\_BIAS\_DIFFBN

```
#define DAVIS208_CONFIG_BIAS_DIFFBN 10
```

Parameter address for module DAVIS208\_CONFIG\_BIAS: DAVIS208 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.59 DAVIS208\_CONFIG\_BIAS\_IFREFRBN

```
#define DAVIS208_CONFIG_BIAS_IFREFRBN 26
```

Parameter address for module DAVIS208\_CONFIG\_BIAS: DAVIS208 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.60 DAVIS208\_CONFIG\_BIAS\_IFTHRBN

```
#define DAVIS208_CONFIG_BIAS_IFTHRBN 27
```

Parameter address for module DAVIS208\_CONFIG\_BIAS: DAVIS208 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.61 DAVIS208\_CONFIG\_BIAS\_LCOLTIMEOUTBN

```
#define DAVIS208_CONFIG_BIAS_LCOLTIMEOUTBN 22
```

Parameter address for module DAVIS208\_CONFIG\_BIAS: DAVIS208 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.62 DAVIS208\_CONFIG\_BIAS\_LOCALBUFBN

```
#define DAVIS208_CONFIG_BIAS_LOCALBUFBN 8
```

Parameter address for module DAVIS208\_CONFIG\_BIAS: DAVIS208 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.63 DAVIS208\_CONFIG\_BIAS\_OFFBN

```
#define DAVIS208_CONFIG_BIAS_OFFBN 12
```

Parameter address for module DAVIS208\_CONFIG\_BIAS: DAVIS208 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.64 DAVIS208\_CONFIG\_BIAS\_ONBN

```
#define DAVIS208_CONFIG_BIAS_ONBN 11
```

Parameter address for module DAVIS208\_CONFIG\_BIAS: DAVIS208 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.65 DAVIS208\_CONFIG\_BIAS\_PADFOLLBN

```
#define DAVIS208_CONFIG_BIAS_PADFOLLBN 9
```

Parameter address for module DAVIS208\_CONFIG\_BIAS: DAVIS208 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.66 DAVIS208\_CONFIG\_BIAS\_PIXINBN

```
#define DAVIS208_CONFIG_BIAS_PIXINBN 13
```

Parameter address for module DAVIS208\_CONFIG\_BIAS: DAVIS208 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.67 DAVIS208\_CONFIG\_BIAS\_PRBP

```
#define DAVIS208_CONFIG_BIAS_PRBP 14
```

Parameter address for module DAVIS208\_CONFIG\_BIAS: DAVIS208 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.68 DAVIS208\_CONFIG\_BIAS\_PRSFBP

```
#define DAVIS208_CONFIG_BIAS_PRSFBP 15
```

Parameter address for module DAVIS208\_CONFIG\_BIAS: DAVIS208 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.69 DAVIS208\_CONFIG\_BIAS\_READOUTBUFBP

```
#define DAVIS208_CONFIG_BIAS_READOUTBUFBP 17
```

Parameter address for module DAVIS208\_CONFIG\_BIAS: DAVIS208 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.



## 4.1.2.70 DAVIS208\_CONFIG\_BIAS\_REFRBP

```
#define DAVIS208_CONFIG_BIAS_REFRBP 16
```

Parameter address for module DAVIS208\_CONFIG\_BIAS: DAVIS208 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

## 4.1.2.71 DAVIS208\_CONFIG\_BIAS\_REFSS

```
#define DAVIS208_CONFIG_BIAS_REFSS 7
```

Parameter address for module DAVIS208\_CONFIG\_BIAS: DAVIS208 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

## 4.1.2.72 DAVIS208\_CONFIG\_BIAS\_REFSSBN

```
#define DAVIS208_CONFIG_BIAS_REFSSBN 30
```

Parameter address for module DAVIS208\_CONFIG\_BIAS: DAVIS208 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.73 DAVIS208\_CONFIG\_BIAS\_REGBIASBP

```
#define DAVIS208_CONFIG_BIAS_REGBIASBP 28
```

Parameter address for module DAVIS208\_CONFIG\_BIAS: DAVIS208 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.74 DAVIS208\_CONFIG\_BIAS\_RESETHIGHPASS

```
#define DAVIS208_CONFIG_BIAS_RESETHIGHPASS 6
```

Parameter address for module DAVIS208\_CONFIG\_BIAS: DAVIS208 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.75 DAVIS208\_CONFIG\_BIAS\_SSN

```
#define DAVIS208_CONFIG_BIAS_SSN 36
```

Parameter address for module DAVIS208\_CONFIG\_BIAS: DAVIS208 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.76 DAVIS208\_CONFIG\_BIAS\_SSP

```
#define DAVIS208_CONFIG_BIAS_SSP 35
```

Parameter address for module DAVIS208\_CONFIG\_BIAS: DAVIS208 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.77 DAVIS208\_CONFIG\_CHIP\_AERNAROW

```
#define DAVIS208_CONFIG_CHIP_AERNAROW 140
```

Parameter address for module DAVIS208\_CONFIG\_CHIP: DAVIS208 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.78 DAVIS208\_CONFIG\_CHIP\_ANALOGMUX0

```
#define DAVIS208_CONFIG_CHIP_ANALOGMUX0 132
```

Parameter address for module DAVIS208\_CONFIG\_CHIP: DAVIS208 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.79 DAVIS208\_CONFIG\_CHIP\_ANALOGMUX1

```
#define DAVIS208_CONFIG_CHIP_ANALOGMUX1 133
```

Parameter address for module DAVIS208\_CONFIG\_CHIP: DAVIS208 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.80 DAVIS208\_CONFIG\_CHIP\_ANALOGMUX2

```
#define DAVIS208_CONFIG_CHIP_ANALOGMUX2 134
```

Parameter address for module DAVIS208\_CONFIG\_CHIP: DAVIS208 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.81 DAVIS208\_CONFIG\_CHIP\_BIASMUX0

```
#define DAVIS208_CONFIG_CHIP_BIASMUX0 135
```

Parameter address for module DAVIS208\_CONFIG\_CHIP: DAVIS208 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.82 DAVIS208\_CONFIG\_CHIP\_DIGITALMUX0

```
#define DAVIS208_CONFIG_CHIP_DIGITALMUX0 128
```

Parameter address for module DAVIS208\_CONFIG\_CHIP: DAVIS208 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.83 DAVIS208\_CONFIG\_CHIP\_DIGITALMUX1

```
#define DAVIS208_CONFIG_CHIP_DIGITALMUX1 129
```

Parameter address for module DAVIS208\_CONFIG\_CHIP: DAVIS208 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.84 DAVIS208\_CONFIG\_CHIP\_DIGITALMUX2

```
#define DAVIS208_CONFIG_CHIP_DIGITALMUX2 130
```

Parameter address for module DAVIS208\_CONFIG\_CHIP: DAVIS208 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.85 DAVIS208\_CONFIG\_CHIP\_DIGITALMUX3

```
#define DAVIS208_CONFIG_CHIP_DIGITALMUX3 131
```

Parameter address for module DAVIS208\_CONFIG\_CHIP: DAVIS208 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.86 DAVIS208\_CONFIG\_CHIP\_GLOBAL\_SHUTTER

```
#define DAVIS208_CONFIG_CHIP_GLOBAL_SHUTTER 142
```

Parameter address for module DAVIS208\_CONFIG\_CHIP: DAVIS208 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.87 DAVIS208\_CONFIG\_CHIP\_RESETCALIBNEURON

```
#define DAVIS208_CONFIG_CHIP_RESETCALIBNEURON 136
```

Parameter address for module DAVIS208\_CONFIG\_CHIP: DAVIS208 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.88 DAVIS208\_CONFIG\_CHIP\_RESETESTPIXEL

```
#define DAVIS208_CONFIG_CHIP_RESETESTPIXEL 138
```

Parameter address for module DAVIS208\_CONFIG\_CHIP: DAVIS208 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.89 DAVIS208\_CONFIG\_CHIP\_SELECTBIASREFSS

```
#define DAVIS208_CONFIG_CHIP_SELECTBIASREFSS 146
```

Parameter address for module DAVIS208\_CONFIG\_CHIP: DAVIS208 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.90 DAVIS208\_CONFIG\_CHIP\_SELECTGRAYCOUNTER

```
#define DAVIS208_CONFIG_CHIP_SELECTGRAYCOUNTER 143
```

Parameter address for module DAVIS208\_CONFIG\_CHIP: DAVIS208 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.91 DAVIS208\_CONFIG\_CHIP\_SELECTHIGHPASS

```
#define DAVIS208_CONFIG_CHIP_SELECTHIGHPASS 149
```

Parameter address for module DAVIS208\_CONFIG\_CHIP: DAVIS208 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.92 DAVIS208\_CONFIG\_CHIP\_SELECTPOSB

```
#define DAVIS208_CONFIG_CHIP_SELECTPOSB 148
```

Parameter address for module DAVIS208\_CONFIG\_CHIP: DAVIS208 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.93 DAVIS208\_CONFIG\_CHIP\_SELECTPREAMPAVG

```
#define DAVIS208_CONFIG_CHIP_SELECTPREAMPAVG 145
```

Parameter address for module DAVIS208\_CONFIG\_CHIP: DAVIS208 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.94 DAVIS208\_CONFIG\_CHIP\_SELECTSENSE

```
#define DAVIS208_CONFIG_CHIP_SELECTSENSE 147
```

Parameter address for module DAVIS208\_CONFIG\_CHIP: DAVIS208 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.95 DAVIS208\_CONFIG\_CHIP\_TYPENCALIBNEURON

```
#define DAVIS208_CONFIG_CHIP_TYPENCALIBNEURON 137
```

Parameter address for module DAVIS208\_CONFIG\_CHIP: DAVIS208 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.96 DAVIS208\_CONFIG\_CHIP\_USEAOUT

```
#define DAVIS208_CONFIG_CHIP_USEAOUT 141
```

Parameter address for module DAVIS208\_CONFIG\_CHIP: DAVIS208 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.97 DAVIS240\_CONFIG\_BIAS\_AEPDBN

```
#define DAVIS240_CONFIG_BIAS_AEPDBN 11
```

Parameter address for module DAVIS240\_CONFIG\_BIAS: DAVIS240chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.98 DAVIS240\_CONFIG\_BIAS\_AEPUXBP

```
#define DAVIS240_CONFIG_BIAS_AEPUXBP 13
```

Parameter address for module DAVIS240\_CONFIG\_BIAS: DAVIS240chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.99 DAVIS240\_CONFIG\_BIAS\_AEPUYBP

```
#define DAVIS240_CONFIG_BIAS_AEPUYBP 14
```

Parameter address for module DAVIS240\_CONFIG\_BIAS: DAVIS240chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.100 DAVIS240\_CONFIG\_BIAS\_APSCASEPC

```
#define DAVIS240_CONFIG_BIAS_APSCASEPC 3
```

Parameter address for module DAVIS240\_CONFIG\_BIAS: DAVIS240chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.101 DAVIS240\_CONFIG\_BIAS\_APSEVERFLOWLEVELBN

```
#define DAVIS240_CONFIG_BIAS_APSEVERFLOWLEVELBN 18
```

Parameter address for module DAVIS240\_CONFIG\_BIAS: DAVIS240chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.102 DAVIS240\_CONFIG\_BIAS\_APSROSFBN

```
#define DAVIS240_CONFIG_BIAS_APSROSFBN 5
```

Parameter address for module DAVIS240\_CONFIG\_BIAS: DAVIS240chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.103 DAVIS240\_CONFIG\_BIAS\_BIASBUFFER

```
#define DAVIS240_CONFIG_BIAS_BIASBUFFER 19
```

Parameter address for module DAVIS240\_CONFIG\_BIAS: DAVIS240chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.104 DAVIS240\_CONFIG\_BIAS\_DIFFBN

```
#define DAVIS240_CONFIG_BIAS_DIFFBN 0
```

Parameter address for module DAVIS240\_CONFIG\_BIAS: DAVIS240chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.105 DAVIS240\_CONFIG\_BIAS\_DIFFCASBNC

```
#define DAVIS240_CONFIG_BIAS_DIFFCASBNC 4
```

Parameter address for module DAVIS240\_CONFIG\_BIAS: DAVIS240chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.



#### 4.1.2.106 DAVIS240\_CONFIG\_BIAS\_IFREFRBN

```
#define DAVIS240_CONFIG_BIAS_IFREFRBN 16
```

Parameter address for module DAVIS240\_CONFIG\_BIAS: DAVIS240chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.107 DAVIS240\_CONFIG\_BIAS\_IFTHRBN

```
#define DAVIS240_CONFIG_BIAS_IFTHRBN 15
```

Parameter address for module DAVIS240\_CONFIG\_BIAS: DAVIS240chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.108 DAVIS240\_CONFIG\_BIAS\_LCOLTIMEOUTBN

```
#define DAVIS240_CONFIG_BIAS_LCOLTIMEOUTBN 12
```

Parameter address for module DAVIS240\_CONFIG\_BIAS: DAVIS240chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.109 DAVIS240\_CONFIG\_BIAS\_LOCALBUFBN

```
#define DAVIS240_CONFIG_BIAS_LOCALBUFBN 6
```

Parameter address for module DAVIS240\_CONFIG\_BIAS: DAVIS240chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.110 DAVIS240\_CONFIG\_BIAS\_OFFBN

```
#define DAVIS240_CONFIG_BIAS_OFFBN 2
```

Parameter address for module DAVIS240\_CONFIG\_BIAS: DAVIS240chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.111 DAVIS240\_CONFIG\_BIAS\_ONBN

```
#define DAVIS240_CONFIG_BIAS_ONBN 1
```

Parameter address for module DAVIS240\_CONFIG\_BIAS: DAVIS240chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.112 DAVIS240\_CONFIG\_BIAS\_PADFOLLBN

```
#define DAVIS240_CONFIG_BIAS_PADFOLLBN 17
```

Parameter address for module DAVIS240\_CONFIG\_BIAS: DAVIS240chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.113 DAVIS240\_CONFIG\_BIAS\_PIXINBN

```
#define DAVIS240_CONFIG_BIAS_PIXINBN 7
```

Parameter address for module DAVIS240\_CONFIG\_BIAS: DAVIS240chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.114 DAVIS240\_CONFIG\_BIAS\_PRBP

```
#define DAVIS240_CONFIG_BIAS_PRBP 8
```

Parameter address for module DAVIS240\_CONFIG\_BIAS: DAVIS240chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.115 DAVIS240\_CONFIG\_BIAS\_PRFBP

```
#define DAVIS240_CONFIG_BIAS_PRFBP 9
```

Parameter address for module DAVIS240\_CONFIG\_BIAS: DAVIS240chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.116 DAVIS240\_CONFIG\_BIAS\_REFRBP

```
#define DAVIS240_CONFIG_BIAS_REFRBP 10
```

Parameter address for module DAVIS240\_CONFIG\_BIAS: DAVIS240chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.117 DAVIS240\_CONFIG\_BIAS\_SSN

```
#define DAVIS240_CONFIG_BIAS_SSN 21
```

Parameter address for module DAVIS240\_CONFIG\_BIAS: DAVIS240chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.118 DAVIS240\_CONFIG\_BIAS\_SSP

```
#define DAVIS240_CONFIG_BIAS_SSP 20
```

Parameter address for module DAVIS240\_CONFIG\_BIAS: DAVIS240chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.119 DAVIS240\_CONFIG\_CHIP\_AERNAROW

```
#define DAVIS240_CONFIG_CHIP_AERNAROW 140
```

Parameter address for module DAVIS240\_CONFIG\_CHIP: DAVIS240 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead. On DAVIS240B cameras, DAVIS240\_CONFIG\_CHIP\_SPECIALPIXELCONTROL can be used to enable the test pixel array.

#### 4.1.2.120 DAVIS240\_CONFIG\_CHIP\_ANALOGMUX0

```
#define DAVIS240_CONFIG_CHIP_ANALOGMUX0 132
```

Parameter address for module DAVIS240\_CONFIG\_CHIP: DAVIS240 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead. On DAVIS240B cameras, DAVIS240\_CONFIG\_CHIP\_SPECIALPIXELCONTROL can be used to enable the test pixel array.

#### 4.1.2.121 DAVIS240\_CONFIG\_CHIP\_ANALOGMUX1

```
#define DAVIS240_CONFIG_CHIP_ANALOGMUX1 133
```

Parameter address for module DAVIS240\_CONFIG\_CHIP: DAVIS240 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead. On DAVIS240B cameras, DAVIS240\_CONFIG\_CHIP\_SPECIALPIXELCONTROL can be used to enable the test pixel array.

#### 4.1.2.122 DAVIS240\_CONFIG\_CHIP\_ANALOGMUX2

```
#define DAVIS240_CONFIG_CHIP_ANALOGMUX2 134
```

Parameter address for module DAVIS240\_CONFIG\_CHIP: DAVIS240 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead. On DAVIS240B cameras, DAVIS240\_CONFIG\_CHIP\_SPECIALPIXELCONTROL can be used to enable the test pixel array.

#### 4.1.2.123 DAVIS240\_CONFIG\_CHIP\_BIASMUX0

```
#define DAVIS240_CONFIG_CHIP_BIASMUX0 135
```

Parameter address for module DAVIS240\_CONFIG\_CHIP: DAVIS240 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead. On DAVIS240B cameras, DAVIS240\_CONFIG\_CHIP\_SPECIALPIXELCONTROL can be used to enable the test pixel array.

#### 4.1.2.124 DAVIS240\_CONFIG\_CHIP\_DIGITALMUX0

```
#define DAVIS240_CONFIG_CHIP_DIGITALMUX0 128
```

Parameter address for module DAVIS240\_CONFIG\_CHIP: DAVIS240 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead. On DAVIS240B cameras, DAVIS240\_CONFIG\_CHIP\_SPECIALPIXELCONTROL can be used to enable the test pixel array.

#### 4.1.2.125 DAVIS240\_CONFIG\_CHIP\_DIGITALMUX1

```
#define DAVIS240_CONFIG_CHIP_DIGITALMUX1 129
```

Parameter address for module DAVIS240\_CONFIG\_CHIP: DAVIS240 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead. On DAVIS240B cameras, DAVIS240\_CONFIG\_CHIP\_SPECIALPIXELCONTROL can be used to enable the test pixel array.

#### 4.1.2.126 DAVIS240\_CONFIG\_CHIP\_DIGITALMUX2

```
#define DAVIS240_CONFIG_CHIP_DIGITALMUX2 130
```

Parameter address for module DAVIS240\_CONFIG\_CHIP: DAVIS240 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead. On DAVIS240B cameras, DAVIS240\_CONFIG\_CHIP\_SPECIALPIXELCONTROL can be used to enable the test pixel array.

#### 4.1.2.127 DAVIS240\_CONFIG\_CHIP\_DIGITALMUX3

```
#define DAVIS240_CONFIG_CHIP_DIGITALMUX3 131
```

Parameter address for module DAVIS240\_CONFIG\_CHIP: DAVIS240 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead. On DAVIS240B cameras, DAVIS240\_CONFIG\_CHIP\_SPECIALPIXELCONTROL can be used to enable the test pixel array.

#### 4.1.2.128 DAVIS240\_CONFIG\_CHIP\_GLOBAL\_SHUTTER

```
#define DAVIS240_CONFIG_CHIP_GLOBAL_SHUTTER 142
```

Parameter address for module DAVIS240\_CONFIG\_CHIP: DAVIS240 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead. On DAVIS240B cameras, DAVIS240\_CONFIG\_CHIP\_SPECIALPIXELCONTROL can be used to enable the test pixel array.

#### 4.1.2.129 DAVIS240\_CONFIG\_CHIP\_RESETCALIBNEURON

```
#define DAVIS240_CONFIG_CHIP_RESETCALIBNEURON 136
```

Parameter address for module DAVIS240\_CONFIG\_CHIP: DAVIS240 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead. On DAVIS240B cameras, DAVIS240\_CONFIG\_CHIP\_SPECIALPIXELCONTROL can be used to enable the test pixel array.

#### 4.1.2.130 DAVIS240\_CONFIG\_CHIP\_RESETTESTPIXEL

```
#define DAVIS240_CONFIG_CHIP_RESETTESTPIXEL 138
```

Parameter address for module DAVIS240\_CONFIG\_CHIP: DAVIS240 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead. On DAVIS240B cameras, DAVIS240\_CONFIG\_CHIP\_SPECIALPIXELCONTROL can be used to enable the test pixel array.

#### 4.1.2.131 DAVIS240\_CONFIG\_CHIP\_SPECIALPIXELCONTROL

```
#define DAVIS240_CONFIG_CHIP_SPECIALPIXELCONTROL 139
```

Parameter address for module DAVIS240\_CONFIG\_CHIP: DAVIS240 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead. On DAVIS240B cameras, DAVIS240\_CONFIG\_CHIP\_SPECIALPIXELCONTROL can be used to enable the test pixel array.

#### 4.1.2.132 DAVIS240\_CONFIG\_CHIP\_TYPENCALIBNEURON

```
#define DAVIS240_CONFIG_CHIP_TYPENCALIBNEURON 137
```

Parameter address for module DAVIS240\_CONFIG\_CHIP: DAVIS240 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead. On DAVIS240B cameras, DAVIS240\_CONFIG\_CHIP\_SPECIALPIXELCONTROL can be used to enable the test pixel array.

#### 4.1.2.133 DAVIS240\_CONFIG\_CHIP\_USEAOUT

```
#define DAVIS240_CONFIG_CHIP_USEAOUT 141
```

Parameter address for module DAVIS240\_CONFIG\_CHIP: DAVIS240 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead. On DAVIS240B cameras, DAVIS240\_CONFIG\_CHIP\_SPECIALPIXELCONTROL can be used to enable the test pixel array.

#### 4.1.2.134 DAVIS346\_CONFIG\_BIAS\_ADCCOMPBP

```
#define DAVIS346_CONFIG_BIAS_ADCCOMPBP 19
```

Parameter address for module DAVIS346\_CONFIG\_BIAS: DAVIS346 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.135 DAVIS346\_CONFIG\_BIAS\_ADCREFHIGH

```
#define DAVIS346_CONFIG_BIAS_ADCREFHIGH 2
```

Parameter address for module DAVIS346\_CONFIG\_BIAS: DAVIS346 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.136 DAVIS346\_CONFIG\_BIAS\_ADCREFLOW

```
#define DAVIS346_CONFIG_BIAS_ADCREFLOW 3
```

Parameter address for module DAVIS346\_CONFIG\_BIAS: DAVIS346 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.137 DAVIS346\_CONFIG\_BIAS\_ADCTESTVOLTAGE

```
#define DAVIS346_CONFIG_BIAS_ADCTESTVOLTAGE 4
```

Parameter address for module DAVIS346\_CONFIG\_BIAS: DAVIS346 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.138 DAVIS346\_CONFIG\_BIAS\_AEPDBN

```
#define DAVIS346_CONFIG_BIAS_AEPDBN 23
```

Parameter address for module DAVIS346\_CONFIG\_BIAS: DAVIS346 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.139 DAVIS346\_CONFIG\_BIAS\_AEPUXBP

```
#define DAVIS346_CONFIG_BIAS_AEPUXBP 24
```

Parameter address for module DAVIS346\_CONFIG\_BIAS: DAVIS346 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.



#### 4.1.2.140 DAVIS346\_CONFIG\_BIAS\_AEPUYBP

```
#define DAVIS346_CONFIG_BIAS_AEPUYBP 25
```

Parameter address for module DAVIS346\_CONFIG\_BIAS: DAVIS346 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.141 DAVIS346\_CONFIG\_BIAS\_APSCAS

```
#define DAVIS346_CONFIG_BIAS_APSCAS 1
```

Parameter address for module DAVIS346\_CONFIG\_BIAS: DAVIS346 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.142 DAVIS346\_CONFIG\_BIAS\_APSEVERFLOWLEVEL

```
#define DAVIS346_CONFIG_BIAS_APSEVERFLOWLEVEL 0
```

Parameter address for module DAVIS346\_CONFIG\_BIAS: DAVIS346 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.143 DAVIS346\_CONFIG\_BIAS\_APSROSFBN

```
#define DAVIS346_CONFIG_BIAS_APSROSFBN 18
```

Parameter address for module DAVIS346\_CONFIG\_BIAS: DAVIS346 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.144 DAVIS346\_CONFIG\_BIAS\_BIASBUFFER

```
#define DAVIS346_CONFIG_BIAS_BIASBUFFER 34
```

Parameter address for module DAVIS346\_CONFIG\_BIAS: DAVIS346 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.145 DAVIS346\_CONFIG\_BIAS\_COLSELLOWBN

```
#define DAVIS346_CONFIG_BIAS_COLSELLOWBN 20
```

Parameter address for module DAVIS346\_CONFIG\_BIAS: DAVIS346 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.146 DAVIS346\_CONFIG\_BIAS\_DACBUFBP

```
#define DAVIS346_CONFIG_BIAS_DACBUFBP 21
```

Parameter address for module DAVIS346\_CONFIG\_BIAS: DAVIS346 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.147 DAVIS346\_CONFIG\_BIAS\_DIFFBN

```
#define DAVIS346_CONFIG_BIAS_DIFFBN 10
```

Parameter address for module DAVIS346\_CONFIG\_BIAS: DAVIS346 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.148 DAVIS346\_CONFIG\_BIAS\_IFREFRBN

```
#define DAVIS346_CONFIG_BIAS_IFREFRBN 26
```

Parameter address for module DAVIS346\_CONFIG\_BIAS: DAVIS346 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.149 DAVIS346\_CONFIG\_BIAS\_IFTHRBN

```
#define DAVIS346_CONFIG_BIAS_IFTHRBN 27
```

Parameter address for module DAVIS346\_CONFIG\_BIAS: DAVIS346 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.150 DAVIS346\_CONFIG\_BIAS\_LCOLTIMEOUTBN

```
#define DAVIS346_CONFIG_BIAS_LCOLTIMEOUTBN 22
```

Parameter address for module DAVIS346\_CONFIG\_BIAS: DAVIS346 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.151 DAVIS346\_CONFIG\_BIAS\_LOCALBUFBN

```
#define DAVIS346_CONFIG_BIAS_LOCALBUFBN 8
```

Parameter address for module DAVIS346\_CONFIG\_BIAS: DAVIS346 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

## 4.1.2.152 DAVIS346\_CONFIG\_BIAS\_OFFBN

```
#define DAVIS346_CONFIG_BIAS_OFFBN 12
```

Parameter address for module DAVIS346\_CONFIG\_BIAS: DAVIS346 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

## 4.1.2.153 DAVIS346\_CONFIG\_BIAS\_ONBN

```
#define DAVIS346_CONFIG_BIAS_ONBN 11
```

Parameter address for module DAVIS346\_CONFIG\_BIAS: DAVIS346 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

## 4.1.2.154 DAVIS346\_CONFIG\_BIAS\_PADFOLLBN

```
#define DAVIS346_CONFIG_BIAS_PADFOLLBN 9
```

Parameter address for module DAVIS346\_CONFIG\_BIAS: DAVIS346 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.155 DAVIS346\_CONFIG\_BIAS\_PIXINBN

```
#define DAVIS346_CONFIG_BIAS_PIXINBN 13
```

Parameter address for module DAVIS346\_CONFIG\_BIAS: DAVIS346 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.156 DAVIS346\_CONFIG\_BIAS\_PRBP

```
#define DAVIS346_CONFIG_BIAS_PRBP 14
```

Parameter address for module DAVIS346\_CONFIG\_BIAS: DAVIS346 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.157 DAVIS346\_CONFIG\_BIAS\_PRSFBP

```
#define DAVIS346_CONFIG_BIAS_PRSFBP 15
```

Parameter address for module DAVIS346\_CONFIG\_BIAS: DAVIS346 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

**4.1.2.158 DAVIS346\_CONFIG\_BIAS\_READOUTBUFBP**

```
#define DAVIS346_CONFIG_BIAS_READOUTBUFBP 17
```

Parameter address for module DAVIS346\_CONFIG\_BIAS: DAVIS346 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

**4.1.2.159 DAVIS346\_CONFIG\_BIAS\_REFRBP**

```
#define DAVIS346_CONFIG_BIAS_REFRBP 16
```

Parameter address for module DAVIS346\_CONFIG\_BIAS: DAVIS346 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

**4.1.2.160 DAVIS346\_CONFIG\_BIAS\_SSN**

```
#define DAVIS346_CONFIG_BIAS_SSN 36
```

Parameter address for module DAVIS346\_CONFIG\_BIAS: DAVIS346 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.161 DAVIS346\_CONFIG\_BIAS\_SSP

```
#define DAVIS346_CONFIG_BIAS_SSP 35
```

Parameter address for module DAVIS346\_CONFIG\_BIAS: DAVIS346 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.162 DAVIS346\_CONFIG\_CHIP\_AERNAROW

```
#define DAVIS346_CONFIG_CHIP_AERNAROW 140
```

Parameter address for module DAVIS346\_CONFIG\_CHIP: DAVIS346 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.163 DAVIS346\_CONFIG\_CHIP\_ANALOGMUX0

```
#define DAVIS346_CONFIG_CHIP_ANALOGMUX0 132
```

Parameter address for module DAVIS346\_CONFIG\_CHIP: DAVIS346 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.164 DAVIS346\_CONFIG\_CHIP\_ANALOGMUX1

```
#define DAVIS346_CONFIG_CHIP_ANALOGMUX1 133
```

Parameter address for module DAVIS346\_CONFIG\_CHIP: DAVIS346 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.165 DAVIS346\_CONFIG\_CHIP\_ANALOGMUX2

```
#define DAVIS346_CONFIG_CHIP_ANALOGMUX2 134
```

Parameter address for module DAVIS346\_CONFIG\_CHIP: DAVIS346 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.



**4.1.2.166 DAVIS346\_CONFIG\_CHIP\_BIASMUX0**

```
#define DAVIS346_CONFIG_CHIP_BIASMUX0 135
```

Parameter address for module DAVIS346\_CONFIG\_CHIP: DAVIS346 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

**4.1.2.167 DAVIS346\_CONFIG\_CHIP\_DIGITALMUX0**

```
#define DAVIS346_CONFIG_CHIP_DIGITALMUX0 128
```

Parameter address for module DAVIS346\_CONFIG\_CHIP: DAVIS346 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

**4.1.2.168 DAVIS346\_CONFIG\_CHIP\_DIGITALMUX1**

```
#define DAVIS346_CONFIG_CHIP_DIGITALMUX1 129
```

Parameter address for module DAVIS346\_CONFIG\_CHIP: DAVIS346 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

**4.1.2.169 DAVIS346\_CONFIG\_CHIP\_DIGITALMUX2**

```
#define DAVIS346_CONFIG_CHIP_DIGITALMUX2 130
```

Parameter address for module DAVIS346\_CONFIG\_CHIP: DAVIS346 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

**4.1.2.170 DAVIS346\_CONFIG\_CHIP\_DIGITALMUX3**

```
#define DAVIS346_CONFIG_CHIP_DIGITALMUX3 131
```

Parameter address for module DAVIS346\_CONFIG\_CHIP: DAVIS346 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

**4.1.2.171 DAVIS346\_CONFIG\_CHIP\_GLOBAL\_SHUTTER**

```
#define DAVIS346_CONFIG_CHIP_GLOBAL_SHUTTER 142
```

Parameter address for module DAVIS346\_CONFIG\_CHIP: DAVIS346 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.172 DAVIS346\_CONFIG\_CHIP\_RESETCALIBNEURON

```
#define DAVIS346_CONFIG_CHIP_RESETCALIBNEURON 136
```

Parameter address for module DAVIS346\_CONFIG\_CHIP: DAVIS346 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.173 DAVIS346\_CONFIG\_CHIP\_RESETESTPIXEL

```
#define DAVIS346_CONFIG_CHIP_RESETESTPIXEL 138
```

Parameter address for module DAVIS346\_CONFIG\_CHIP: DAVIS346 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.174 DAVIS346\_CONFIG\_CHIP\_SELECTGRAYCOUNTER

```
#define DAVIS346_CONFIG_CHIP_SELECTGRAYCOUNTER 143
```

Parameter address for module DAVIS346\_CONFIG\_CHIP: DAVIS346 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.175 DAVIS346\_CONFIG\_CHIP\_TESTADC

```
#define DAVIS346_CONFIG_CHIP_TESTADC 144
```

Parameter address for module DAVIS346\_CONFIG\_CHIP: DAVIS346 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.176 DAVIS346\_CONFIG\_CHIP\_TYPENCALIBNEURON

```
#define DAVIS346_CONFIG_CHIP_TYPENCALIBNEURON 137
```

Parameter address for module DAVIS346\_CONFIG\_CHIP: DAVIS346 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.177 DAVIS346\_CONFIG\_CHIP\_USEAOUT

```
#define DAVIS346_CONFIG_CHIP_USEAOUT 141
```

Parameter address for module DAVIS346\_CONFIG\_CHIP: DAVIS346 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

**4.1.2.178 DAVIS640\_CONFIG\_BIAS\_ADCCOMPBP**

```
#define DAVIS640_CONFIG_BIAS_ADCCOMPBP 19
```

Parameter address for module DAVIS640\_CONFIG\_BIAS: DAVIS640 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

**4.1.2.179 DAVIS640\_CONFIG\_BIAS\_ADCREFHIGH**

```
#define DAVIS640_CONFIG_BIAS_ADCREFHIGH 2
```

Parameter address for module DAVIS640\_CONFIG\_BIAS: DAVIS640 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

**4.1.2.180 DAVIS640\_CONFIG\_BIAS\_ADCREFLOW**

```
#define DAVIS640_CONFIG_BIAS_ADCREFLOW 3
```

Parameter address for module DAVIS640\_CONFIG\_BIAS: DAVIS640 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.181 DAVIS640\_CONFIG\_BIAS\_ADCTESTVOLTAGE

```
#define DAVIS640_CONFIG_BIAS_ADCTESTVOLTAGE 4
```

Parameter address for module DAVIS640\_CONFIG\_BIAS: DAVIS640 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.182 DAVIS640\_CONFIG\_BIAS\_AEPDBN

```
#define DAVIS640_CONFIG_BIAS_AEPDBN 23
```

Parameter address for module DAVIS640\_CONFIG\_BIAS: DAVIS640 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.183 DAVIS640\_CONFIG\_BIAS\_AEPUXBP

```
#define DAVIS640_CONFIG_BIAS_AEPUXBP 24
```

Parameter address for module DAVIS640\_CONFIG\_BIAS: DAVIS640 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.184 DAVIS640\_CONFIG\_BIAS\_AEPUYBP

```
#define DAVIS640_CONFIG_BIAS_AEPUYBP 25
```

Parameter address for module DAVIS640\_CONFIG\_BIAS: DAVIS640 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.185 DAVIS640\_CONFIG\_BIAS\_APSCAS

```
#define DAVIS640_CONFIG_BIAS_APSCAS 1
```

Parameter address for module DAVIS640\_CONFIG\_BIAS: DAVIS640 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.186 DAVIS640\_CONFIG\_BIAS\_APSEVERFLOWLEVEL

```
#define DAVIS640_CONFIG_BIAS_APSEVERFLOWLEVEL 0
```

Parameter address for module DAVIS640\_CONFIG\_BIAS: DAVIS640 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.187 DAVIS640\_CONFIG\_BIAS\_APSROSFBN

```
#define DAVIS640_CONFIG_BIAS_APSROSFBN 18
```

Parameter address for module DAVIS640\_CONFIG\_BIAS: DAVIS640 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.188 DAVIS640\_CONFIG\_BIAS\_BIASBUFFER

```
#define DAVIS640_CONFIG_BIAS_BIASBUFFER 34
```

Parameter address for module DAVIS640\_CONFIG\_BIAS: DAVIS640 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.189 DAVIS640\_CONFIG\_BIAS\_COLSELOWBN

```
#define DAVIS640_CONFIG_BIAS_COLSELOWBN 20
```

Parameter address for module DAVIS640\_CONFIG\_BIAS: DAVIS640 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.190 DAVIS640\_CONFIG\_BIAS\_DACBUFBP

```
#define DAVIS640_CONFIG_BIAS_DACBUFBP 21
```

Parameter address for module DAVIS640\_CONFIG\_BIAS: DAVIS640 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.191 DAVIS640\_CONFIG\_BIAS\_DIFFBN

```
#define DAVIS640_CONFIG_BIAS_DIFFBN 10
```

Parameter address for module DAVIS640\_CONFIG\_BIAS: DAVIS640 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.192 DAVIS640\_CONFIG\_BIAS\_IFREFRBN

```
#define DAVIS640_CONFIG_BIAS_IFREFRBN 26
```

Parameter address for module DAVIS640\_CONFIG\_BIAS: DAVIS640 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.193 DAVIS640\_CONFIG\_BIAS\_IFTHRBN

```
#define DAVIS640_CONFIG_BIAS_IFTHRBN 27
```

Parameter address for module DAVIS640\_CONFIG\_BIAS: DAVIS640 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.194 DAVIS640\_CONFIG\_BIAS\_LCOLTIMEOUTBN

```
#define DAVIS640_CONFIG_BIAS_LCOLTIMEOUTBN 22
```

Parameter address for module DAVIS640\_CONFIG\_BIAS: DAVIS640 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.195 DAVIS640\_CONFIG\_BIAS\_LOCALBUFBN

```
#define DAVIS640_CONFIG_BIAS_LOCALBUFBN 8
```

Parameter address for module DAVIS640\_CONFIG\_BIAS: DAVIS640 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.



**4.1.2.196 DAVIS640\_CONFIG\_BIAS\_OFFBN**

```
#define DAVIS640_CONFIG_BIAS_OFFBN 12
```

Parameter address for module DAVIS640\_CONFIG\_BIAS: DAVIS640 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

**4.1.2.197 DAVIS640\_CONFIG\_BIAS\_ONBN**

```
#define DAVIS640_CONFIG_BIAS_ONBN 11
```

Parameter address for module DAVIS640\_CONFIG\_BIAS: DAVIS640 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

**4.1.2.198 DAVIS640\_CONFIG\_BIAS\_PADFOLLBN**

```
#define DAVIS640_CONFIG_BIAS_PADFOLLBN 9
```

Parameter address for module DAVIS640\_CONFIG\_BIAS: DAVIS640 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.199 DAVIS640\_CONFIG\_BIAS\_PIXINBN

```
#define DAVIS640_CONFIG_BIAS_PIXINBN 13
```

Parameter address for module DAVIS640\_CONFIG\_BIAS: DAVIS640 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.200 DAVIS640\_CONFIG\_BIAS\_PRBP

```
#define DAVIS640_CONFIG_BIAS_PRBP 14
```

Parameter address for module DAVIS640\_CONFIG\_BIAS: DAVIS640 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.201 DAVIS640\_CONFIG\_BIAS\_PRSFBP

```
#define DAVIS640_CONFIG_BIAS_PRSFBP 15
```

Parameter address for module DAVIS640\_CONFIG\_BIAS: DAVIS640 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.202 DAVIS640\_CONFIG\_BIAS\_READOUTBUFBP

```
#define DAVIS640_CONFIG_BIAS_READOUTBUFBP 17
```

Parameter address for module DAVIS640\_CONFIG\_BIAS: DAVIS640 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.203 DAVIS640\_CONFIG\_BIAS\_REFRBP

```
#define DAVIS640_CONFIG_BIAS_REFRBP 16
```

Parameter address for module DAVIS640\_CONFIG\_BIAS: DAVIS640 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.204 DAVIS640\_CONFIG\_BIAS\_SSN

```
#define DAVIS640_CONFIG_BIAS_SSN 36
```

Parameter address for module DAVIS640\_CONFIG\_BIAS: DAVIS640 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.205 DAVIS640\_CONFIG\_BIAS\_SSP

```
#define DAVIS640_CONFIG_BIAS_SSP 35
```

Parameter address for module DAVIS640\_CONFIG\_BIAS: DAVIS640 chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.206 DAVIS640\_CONFIG\_CHIP\_AERNAROW

```
#define DAVIS640_CONFIG_CHIP_AERNAROW 140
```

Parameter address for module DAVIS640\_CONFIG\_CHIP: DAVIS640 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.207 DAVIS640\_CONFIG\_CHIP\_ANALOGMUX0

```
#define DAVIS640_CONFIG_CHIP_ANALOGMUX0 132
```

Parameter address for module DAVIS640\_CONFIG\_CHIP: DAVIS640 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.208 DAVIS640\_CONFIG\_CHIP\_ANALOGMUX1

```
#define DAVIS640_CONFIG_CHIP_ANALOGMUX1 133
```

Parameter address for module DAVIS640\_CONFIG\_CHIP: DAVIS640 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.209 DAVIS640\_CONFIG\_CHIP\_ANALOGMUX2

```
#define DAVIS640_CONFIG_CHIP_ANALOGMUX2 134
```

Parameter address for module DAVIS640\_CONFIG\_CHIP: DAVIS640 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.210 DAVIS640\_CONFIG\_CHIP\_BIASMUX0

```
#define DAVIS640_CONFIG_CHIP_BIASMUX0 135
```

Parameter address for module DAVIS640\_CONFIG\_CHIP: DAVIS640 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.211 DAVIS640\_CONFIG\_CHIP\_DIGITALMUX0

```
#define DAVIS640_CONFIG_CHIP_DIGITALMUX0 128
```

Parameter address for module DAVIS640\_CONFIG\_CHIP: DAVIS640 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.212 DAVIS640\_CONFIG\_CHIP\_DIGITALMUX1

```
#define DAVIS640_CONFIG_CHIP_DIGITALMUX1 129
```

Parameter address for module DAVIS640\_CONFIG\_CHIP: DAVIS640 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.213 DAVIS640\_CONFIG\_CHIP\_DIGITALMUX2

```
#define DAVIS640_CONFIG_CHIP_DIGITALMUX2 130
```

Parameter address for module DAVIS640\_CONFIG\_CHIP: DAVIS640 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.214 DAVIS640\_CONFIG\_CHIP\_DIGITALMUX3

```
#define DAVIS640_CONFIG_CHIP_DIGITALMUX3 131
```

Parameter address for module DAVIS640\_CONFIG\_CHIP: DAVIS640 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.215 DAVIS640\_CONFIG\_CHIP\_GLOBAL\_SHUTTER

```
#define DAVIS640_CONFIG_CHIP_GLOBAL_SHUTTER 142
```

Parameter address for module DAVIS640\_CONFIG\_CHIP: DAVIS640 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

**4.1.2.216 DAVIS640\_CONFIG\_CHIP\_RESETCALIBNEURON**

```
#define DAVIS640_CONFIG_CHIP_RESETCALIBNEURON 136
```

Parameter address for module DAVIS640\_CONFIG\_CHIP: DAVIS640 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

**4.1.2.217 DAVIS640\_CONFIG\_CHIP\_RESETESTPIXEL**

```
#define DAVIS640_CONFIG_CHIP_RESETESTPIXEL 138
```

Parameter address for module DAVIS640\_CONFIG\_CHIP: DAVIS640 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

**4.1.2.218 DAVIS640\_CONFIG\_CHIP\_SELECTGRAYCOUNTER**

```
#define DAVIS640_CONFIG_CHIP_SELECTGRAYCOUNTER 143
```

Parameter address for module DAVIS640\_CONFIG\_CHIP: DAVIS640 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

**4.1.2.219 DAVIS640\_CONFIG\_CHIP\_TESTADC**

```
#define DAVIS640_CONFIG_CHIP_TESTADC 144
```

Parameter address for module DAVIS640\_CONFIG\_CHIP: DAVIS640 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

**4.1.2.220 DAVIS640\_CONFIG\_CHIP\_TYPENCALIBNEURON**

```
#define DAVIS640_CONFIG_CHIP_TYPENCALIBNEURON 137
```

Parameter address for module DAVIS640\_CONFIG\_CHIP: DAVIS640 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

**4.1.2.221 DAVIS640\_CONFIG\_CHIP\_USEAOUT**

```
#define DAVIS640_CONFIG_CHIP_USEAOUT 141
```

Parameter address for module DAVIS640\_CONFIG\_CHIP: DAVIS640 chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

**4.1.2.222 DAVIS\_CHIP\_DAVIS128**

```
#define DAVIS_CHIP_DAVIS128 3
```

DAVIS128 chip identifier. 128x128, color possible, internal ADC.

**4.1.2.223 DAVIS\_CHIP\_DAVIS208**

```
#define DAVIS_CHIP_DAVIS208 8
```

DAVIS208 chip identifier. 208x192, special sensitive test pixels, color possible, internal ADC.

**4.1.2.224 DAVIS\_CHIP\_DAVIS240A**

```
#define DAVIS_CHIP_DAVIS240A 0
```

DAVIS240A chip identifier. 240x180, no color, no global shutter.

**4.1.2.225 DAVIS\_CHIP\_DAVIS240B**

```
#define DAVIS_CHIP_DAVIS240B 1
```

DAVIS240B chip identifier. 240x180, no color, 50 test columns left-side.

**4.1.2.226 DAVIS\_CHIP\_DAVIS240C**

```
#define DAVIS_CHIP_DAVIS240C 2
```

DAVIS240C chip identifier. 240x180, no color.

**4.1.2.227 DAVIS\_CHIP\_DAVIS346A**

```
#define DAVIS_CHIP_DAVIS346A 4
```

DAVIS346A chip identifier. 346x260, color possible, internal ADC.

**4.1.2.228 DAVIS\_CHIP\_DAVIS346B**

```
#define DAVIS_CHIP_DAVIS346B 5
```

DAVIS346B chip identifier. 346x260, color possible, internal ADC.

**4.1.2.229 DAVIS\_CHIP\_DAVIS346C**

```
#define DAVIS_CHIP_DAVIS346C 9
```

DAVIS346C chip identifier. 346x260, BSI, color possible, internal ADC.

#### 4.1.2.230 DAVIS\_CHIP\_DAVIS640

```
#define DAVIS_CHIP_DAVIS640 6
```

DAVIS640 chip identifier. 640x480, color possible, internal ADC.

#### 4.1.2.231 DAVIS\_CHIP\_DAVISRGB

```
#define DAVIS_CHIP_DAVISRGB 7
```

DAVISRGB chip identifier. 640x480 APS, 320x240 DVS, color possible, internal ADC.

#### 4.1.2.232 DAVIS\_CONFIG\_APS

```
#define DAVIS_CONFIG_APS 2
```

Module address: device-side APS (Frame) configuration. The APS (Active-Pixel-Sensor) is responsible for getting the normal, synchronous frame from the camera chip. It supports various options for very precise timing control, as well as Region of Interest imaging.

#### 4.1.2.233 DAVIS\_CONFIG\_APS\_ADC\_TEST\_MODE

```
#define DAVIS_CONFIG_APS_ADC_TEST_MODE 39
```

Parameter address for module DAVIS\_CONFIG\_APS: put all APS pixels into reset, while keeping everything else running. This is only useful for testing and characterizing the internal ADC, to minimize noise.

#### 4.1.2.234 DAVIS\_CONFIG\_APS\_AUTOEXPOSURE

```
#define DAVIS_CONFIG_APS_AUTOEXPOSURE 81
```

Parameter address for module DAVIS\_CONFIG\_APS: automatic exposure control, tries to set the exposure value automatically to an appropriate value to maximize information in the scene and minimize under- and over-exposure.

#### 4.1.2.235 DAVIS\_CONFIG\_APS\_COLOR\_FILTER

```
#define DAVIS_CONFIG_APS_COLOR_FILTER 3
```

Parameter address for module DAVIS\_CONFIG\_APS: read-only parameter, contains information on the type of color filter present on the device. This is reserved for internal use and should not be used by anything other than libcaer. Please see the 'struct [caer\\_davis\\_info](#)' documentation to get proper color filter information.

#### 4.1.2.236 DAVIS\_CONFIG\_APS\_COLUMN\_SETTLE

```
#define DAVIS_CONFIG_APS_COLUMN_SETTLE 16
```

Parameter address for module DAVIS\_CONFIG\_APS: column settle time in ADCClock cycles.



**4.1.2.237 DAVIS\_CONFIG\_APS\_END\_COLUMN\_0**

```
#define DAVIS_CONFIG_APS_END_COLUMN_0 11
```

Parameter address for module DAVIS\_CONFIG\_APS: end position on the X axis for Region of Interest 0. Must be between 0 and APS\_SIZE\_X-1, and be greater or equal to DAVIS\_CONFIG\_APS\_START\_COLUMN\_0.

**4.1.2.238 DAVIS\_CONFIG\_APS\_END\_COLUMN\_1**

```
#define DAVIS_CONFIG_APS_END_COLUMN_1 22
```

Parameter address for module DAVIS\_CONFIG\_APS: end position on the X axis for Region of Interest 1. Must be between 0 and APS\_SIZE\_X-1, and be greater or equal to DAVIS\_CONFIG\_APS\_START\_COLUMN\_1.

**4.1.2.239 DAVIS\_CONFIG\_APS\_END\_COLUMN\_2**

```
#define DAVIS_CONFIG_APS_END_COLUMN_2 26
```

Parameter address for module DAVIS\_CONFIG\_APS: end position on the X axis for Region of Interest 2. Must be between 0 and APS\_SIZE\_X-1, and be greater or equal to DAVIS\_CONFIG\_APS\_START\_COLUMN\_2.

**4.1.2.240 DAVIS\_CONFIG\_APS\_END\_COLUMN\_3**

```
#define DAVIS_CONFIG_APS_END_COLUMN_3 30
```

Parameter address for module DAVIS\_CONFIG\_APS: end position on the X axis for Region of Interest 3. Must be between 0 and APS\_SIZE\_X-1, and be greater or equal to DAVIS\_CONFIG\_APS\_START\_COLUMN\_3.

**4.1.2.241 DAVIS\_CONFIG\_APS\_END\_ROW\_0**

```
#define DAVIS_CONFIG_APS_END_ROW_0 12
```

Parameter address for module DAVIS\_CONFIG\_APS: end position on the Y axis for Region of Interest 0. Must be between 0 and APS\_SIZE\_Y-1, and be greater or equal to DAVIS\_CONFIG\_APS\_START\_ROW\_0.

**4.1.2.242 DAVIS\_CONFIG\_APS\_END\_ROW\_1**

```
#define DAVIS_CONFIG_APS_END_ROW_1 23
```

Parameter address for module DAVIS\_CONFIG\_APS: end position on the Y axis for Region of Interest 1. Must be between 0 and APS\_SIZE\_Y-1, and be greater or equal to DAVIS\_CONFIG\_APS\_START\_ROW\_1.

**4.1.2.243 DAVIS\_CONFIG\_APS\_END\_ROW\_2**

```
#define DAVIS_CONFIG_APS_END_ROW_2 27
```

Parameter address for module DAVIS\_CONFIG\_APS: end position on the Y axis for Region of Interest 2. Must be between 0 and APS\_SIZE\_Y-1, and be greater or equal to DAVIS\_CONFIG\_APS\_START\_ROW\_2.

#### 4.1.2.244 DAVIS\_CONFIG\_APS\_END\_ROW\_3

```
#define DAVIS_CONFIG_APS_END_ROW_3 31
```

Parameter address for module DAVIS\_CONFIG\_APS: end position on the Y axis for Region of Interest 3. Must be between 0 and APS\_SIZE\_Y-1, and be greater or equal to DAVIS\_CONFIG\_APS\_START\_ROW\_3.

#### 4.1.2.245 DAVIS\_CONFIG\_APS\_EXPOSURE

```
#define DAVIS_CONFIG_APS_EXPOSURE 13
```

Parameter address for module DAVIS\_CONFIG\_APS: frame exposure time in microseconds, up to about one second maximum. Very precise for Global Shutter, slightly less exact for Rolling Shutter due to column-based timing constraints.

#### 4.1.2.246 DAVIS\_CONFIG\_APS\_FRAME\_DELAY

```
#define DAVIS_CONFIG_APS_FRAME_DELAY 14
```

Parameter address for module DAVIS\_CONFIG\_APS: delay between consecutive frames in microseconds, up to about one second maximum. This can be used to achieve slower frame-rates, down to about 1 Hertz.

#### 4.1.2.247 DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER

```
#define DAVIS_CONFIG_APS_GLOBAL_SHUTTER 8
```

Parameter address for module DAVIS\_CONFIG\_APS: enable Global Shutter mode instead of Rolling Shutter. The Global Shutter eliminates motion artifacts, but is noisier than the Rolling Shutter (worse quality).

#### 4.1.2.248 DAVIS\_CONFIG\_APS\_HAS\_EXTERNAL\_ADC

```
#define DAVIS_CONFIG_APS_HAS_EXTERNAL_ADC 32
```

Parameter address for module DAVIS\_CONFIG\_APS: read-only parameter, information about the presence of an external ADC to read the pixel values. This is reserved for internal use and should not be used by anything other than libcaer. Please see the 'struct [caer\\_davis\\_info](#)' documentation to get this information.

#### 4.1.2.249 DAVIS\_CONFIG\_APS\_HAS\_GLOBAL\_SHUTTER

```
#define DAVIS_CONFIG_APS_HAS_GLOBAL_SHUTTER 7
```

Parameter address for module DAVIS\_CONFIG\_APS: read-only parameter, information about the presence of the global shutter feature. This is reserved for internal use and should not be used by anything other than libcaer. Please see the 'struct [caer\\_davis\\_info](#)' documentation to get this information.

#### 4.1.2.250 DAVIS\_CONFIG\_APS\_HAS\_INTERNAL\_ADC

```
#define DAVIS_CONFIG_APS_HAS_INTERNAL_ADC 33
```

Parameter address for module DAVIS\_CONFIG\_APS: read-only parameter, information about the presence of an internal, on-chip ADC to read the pixel values. This is reserved for internal use and should not be used by anything other than libcaer. Please see the 'struct [caer\\_davis\\_info](#)' documentation to get this information.

#### 4.1.2.251 DAVIS\_CONFIG\_APS\_HAS\_QUAD\_ROI

```
#define DAVIS_CONFIG_APS_HAS_QUAD_ROI 19
```

Parameter address for module DAVIS\_CONFIG\_APS: read-only parameter, information about the presence of the Quadruple Region of Interest feature. This is reserved for internal use and should not be used by anything other than libcaer. Please see the 'struct [caer\\_davis\\_info](#)' documentation to get this information.

#### 4.1.2.252 DAVIS\_CONFIG\_APS\_NULL\_SETTLE

```
#define DAVIS_CONFIG_APS_NULL_SETTLE 18
```

Parameter address for module DAVIS\_CONFIG\_APS: null (between states) settle time in ADCClock cycles.

#### 4.1.2.253 DAVIS\_CONFIG\_APS\_ORIENTATION\_INFO

```
#define DAVIS_CONFIG_APS_ORIENTATION_INFO 2
```

Parameter address for module DAVIS\_CONFIG\_APS: read-only parameter, contains information on the orientation of the X/Y axes, whether they should be inverted or not on the host when parsing incoming pixels, as well as if the X or Y axes need to be flipped when reading the pixels. Bit 2: apsInvertXY Bit 1: apsFlipX Bit 0: apsFlipY This is reserved for internal use and should not be used by anything other than libcaer. Please see the 'struct [caer\\_davis\\_info](#)' documentation to get proper size information that already considers the rotation and orientation settings.

#### 4.1.2.254 DAVIS\_CONFIG\_APS\_RAMP\_RESET

```
#define DAVIS_CONFIG_APS_RAMP_RESET 37
```

Parameter address for module DAVIS\_CONFIG\_APS: ramp reset time in ADCClock cycles.

#### 4.1.2.255 DAVIS\_CONFIG\_APS\_RAMP\_SHORT\_RESET

```
#define DAVIS_CONFIG_APS_RAMP_SHORT_RESET 38
```

Parameter address for module DAVIS\_CONFIG\_APS: only perform a short ramp (half length) during reset reads, given that the voltage should always be close to the top of the range. This increases the frame-rate, but may have impacts on image quality, especially in very bright regions.

#### 4.1.2.256 DAVIS\_CONFIG\_APS\_RESET\_READ

```
#define DAVIS_CONFIG_APS_RESET_READ 5
```

Parameter address for module DAVIS\_CONFIG\_APS: enable the reset read phase in addition to the signal read, to allow for correlated double sampling schemes. This heavily improves image quality and should always be turned on. In special cases, especially when the camera is perfectly stationary, this can be turned off for longer periods of time to achieve a higher frame-rate and significantly faster frame capture.

#### 4.1.2.257 DAVIS\_CONFIG\_APS\_RESET\_SETTLE

```
#define DAVIS_CONFIG_APS_RESET_SETTLE 15
```

Parameter address for module DAVIS\_CONFIG\_APS: column reset settle time in ADCClock cycles.

#### 4.1.2.258 DAVIS\_CONFIG\_APS\_ROW\_SETTLE

```
#define DAVIS_CONFIG_APS_ROW_SETTLE 17
```

Parameter address for module DAVIS\_CONFIG\_APS: row settle time in ADCClock cycles.

#### 4.1.2.259 DAVIS\_CONFIG\_APS\_RUN

```
#define DAVIS_CONFIG_APS_RUN 4
```

Parameter address for module DAVIS\_CONFIG\_APS: enable the APS module and take intensity images of the scene. While this parameter is enabled, frames will be taken continuously. To slow down the frame-rate, see DAVIS\_CONFIG\_APS\_FRAME\_DELAY. To only take snapshots, see DAVIS\_CONFIG\_APS\_SNAPSHOT.

#### 4.1.2.260 DAVIS\_CONFIG\_APS\_SAMPLE\_ENABLE

```
#define DAVIS_CONFIG_APS_SAMPLE_ENABLE 35
```

Parameter address for module DAVIS\_CONFIG\_APS: enable sampling of pixel voltage by the internal ADC circuitry. Must always be enabled to get proper frame values.

#### 4.1.2.261 DAVIS\_CONFIG\_APS\_SAMPLE\_SETTLE

```
#define DAVIS_CONFIG_APS_SAMPLE_SETTLE 36
```

Parameter address for module DAVIS\_CONFIG\_APS: sample settle time in ADCClock cycles.

#### 4.1.2.262 DAVIS\_CONFIG\_APS\_SIZE\_COLUMNS

```
#define DAVIS_CONFIG_APS_SIZE_COLUMNS 0
```

Parameter address for module DAVIS\_CONFIG\_APS: read-only parameter, contains the X axis resolution of the APS frames returned by the camera. This is reserved for internal use and should not be used by anything other than libcaer. Please see the 'struct [caer\\_davis\\_info](#)' documentation to get proper size information that already considers the rotation and orientation settings.

#### 4.1.2.263 DAVIS\_CONFIG\_APS\_SIZE\_ROWS

```
#define DAVIS_CONFIG_APS_SIZE_ROWS 1
```

Parameter address for module DAVIS\_CONFIG\_APS: read-only parameter, contains the Y axis resolution of the APS frames returned by the camera. This is reserved for internal use and should not be used by anything other than libcaer. Please see the 'struct [caer\\_davis\\_info](#)' documentation to get proper size information that already considers the rotation and orientation settings.

#### 4.1.2.264 DAVIS\_CONFIG\_APS\_SNAPSHOT

```
#define DAVIS_CONFIG_APS_SNAPSHOT 80
```

Parameter address for module DAVIS\_CONFIG\_APS: takes a snapshot (one frame), like a photo-camera. More efficient implementation that just toggling the DAVIS\_CONFIG\_APS\_RUN parameter. The APS module should not be running prior to calling this, as it only makes sense if frames are not being generated at the time. Also, DAVIS\_CONFIG\_APS\_FRAME\_DELAY should be set to zero if only doing snapshots, to ensure a quicker readiness for the next one, since the delay is always observed after taking a frame.

#### 4.1.2.265 DAVIS\_CONFIG\_APS\_START\_COLUMN\_0

```
#define DAVIS_CONFIG_APS_START_COLUMN_0 9
```

Parameter address for module DAVIS\_CONFIG\_APS: start position on the X axis for Region of Interest 0. Must be between 0 and APS\_SIZE\_X-1, and be smaller or equal to DAVIS\_CONFIG\_APS\_END\_COLUMN\_0 for the ROI region to be enabled. Setting it to APS\_SIZE\_X itself deactivates this ROI region completely.

#### 4.1.2.266 DAVIS\_CONFIG\_APS\_START\_COLUMN\_1

```
#define DAVIS_CONFIG_APS_START_COLUMN_1 20
```

Parameter address for module DAVIS\_CONFIG\_APS: start position on the X axis for Region of Interest 1. Must be between 0 and APS\_SIZE\_X-1, and be smaller or equal to DAVIS\_CONFIG\_APS\_END\_COLUMN\_1 for the ROI region to be enabled. Setting it to APS\_SIZE\_X itself deactivates this ROI region completely.

#### 4.1.2.267 DAVIS\_CONFIG\_APS\_START\_COLUMN\_2

```
#define DAVIS_CONFIG_APS_START_COLUMN_2 24
```

Parameter address for module DAVIS\_CONFIG\_APS: start position on the X axis for Region of Interest 2. Must be between 0 and APS\_SIZE\_X-1, and be smaller or equal to DAVIS\_CONFIG\_APS\_END\_COLUMN\_2 for the ROI region to be enabled. Setting it to APS\_SIZE\_X itself deactivates this ROI region completely.

#### 4.1.2.268 DAVIS\_CONFIG\_APS\_START\_COLUMN\_3

```
#define DAVIS_CONFIG_APS_START_COLUMN_3 28
```

Parameter address for module DAVIS\_CONFIG\_APS: start position on the X axis for Region of Interest 3. Must be between 0 and APS\_SIZE\_X-1, and be smaller or equal to DAVIS\_CONFIG\_APS\_END\_COLUMN\_3 for the ROI region to be enabled. Setting it to APS\_SIZE\_X itself deactivates this ROI region completely.

**4.1.2.269 DAVIS\_CONFIG\_APS\_START\_ROW\_0**

```
#define DAVIS_CONFIG_APS_START_ROW_0 10
```

Parameter address for module DAVIS\_CONFIG\_APS: start position on the Y axis for Region of Interest 0. Must be between 0 and APS\_SIZE\_Y-1, and be smaller or equal to DAVIS\_CONFIG\_APS\_END\_ROW\_0.

**4.1.2.270 DAVIS\_CONFIG\_APS\_START\_ROW\_1**

```
#define DAVIS_CONFIG_APS_START_ROW_1 21
```

Parameter address for module DAVIS\_CONFIG\_APS: start position on the Y axis for Region of Interest 1. Must be between 0 and APS\_SIZE\_Y-1, and be smaller or equal to DAVIS\_CONFIG\_APS\_END\_ROW\_1.

**4.1.2.271 DAVIS\_CONFIG\_APS\_START\_ROW\_2**

```
#define DAVIS_CONFIG_APS_START_ROW_2 25
```

Parameter address for module DAVIS\_CONFIG\_APS: start position on the Y axis for Region of Interest 2. Must be between 0 and APS\_SIZE\_Y-1, and be smaller or equal to DAVIS\_CONFIG\_APS\_END\_ROW\_2.

**4.1.2.272 DAVIS\_CONFIG\_APS\_START\_ROW\_3**

```
#define DAVIS_CONFIG_APS_START_ROW_3 29
```

Parameter address for module DAVIS\_CONFIG\_APS: start position on the Y axis for Region of Interest 3. Must be between 0 and APS\_SIZE\_Y-1, and be smaller or equal to DAVIS\_CONFIG\_APS\_END\_ROW\_3.

**4.1.2.273 DAVIS\_CONFIG\_APS\_USE\_INTERNAL\_ADC**

```
#define DAVIS_CONFIG_APS_USE_INTERNAL_ADC 34
```

Parameter address for module DAVIS\_CONFIG\_APS: use the internal, on-chip ADC instead of the external one. This enables a much faster and more power-efficient readout for the frames, and should as such always be preferred.

**4.1.2.274 DAVIS\_CONFIG\_APS\_WAIT\_ON\_TRANSFER\_STALL**

```
#define DAVIS_CONFIG_APS_WAIT_ON_TRANSFER_STALL 6
```

Parameter address for module DAVIS\_CONFIG\_APS: if the output FIFO for this module is full, stall the APS state machine and wait until it's free again, instead of just dropping the pixels as they are being read out. This guarantees a complete frame readout, at the possible cost of slight timing differences between pixels. If disabled, incomplete frames may be transmitted and will then be dropped on the host, resulting in lower frame-rates, especially during high DVS traffic.

**4.1.2.275 DAVIS\_CONFIG\_BIAS**

```
#define DAVIS_CONFIG_BIAS 5
```

Module address: device-side chip bias configuration. Shared with DAVIS\_CONFIG\_CHIP. This state machine is responsible for configuring the chip's bias generator.

**4.1.2.276 DAVIS\_CONFIG\_CHIP**

```
#define DAVIS_CONFIG_CHIP 5
```

Module address: device-side chip control configuration. Shared with DAVIS\_CONFIG\_BIAS. This state machine is responsible for configuring the chip's internal control shift registers, to set special options.

**4.1.2.277 DAVIS\_CONFIG\_DVS**

```
#define DAVIS_CONFIG_DVS 1
```

Module address: device-side DVS configuration. The DVS state machine handshakes with the chip's AER bus and gets the polarity events from it. It supports various configurable delays, as well as advanced filtering capabilities on the polarity events.

**4.1.2.278 DAVIS\_CONFIG\_DVS\_ACK\_DELAY\_COLUMN**

```
#define DAVIS_CONFIG_DVS_ACK_DELAY_COLUMN 5
```

Parameter address for module DAVIS\_CONFIG\_DVS: delay capturing the data and acknowledging it on the AER bus for the column events (serial AER protocol) by this many LogicClock cycles.

**4.1.2.279 DAVIS\_CONFIG\_DVS\_ACK\_DELAY\_ROW**

```
#define DAVIS_CONFIG_DVS_ACK_DELAY_ROW 4
```

Parameter address for module DAVIS\_CONFIG\_DVS: delay capturing the data and acknowledging it on the AER bus for the row events (serial AER protocol) by this many LogicClock cycles.

**4.1.2.280 DAVIS\_CONFIG\_DVS\_ACK\_EXTENSION\_COLUMN**

```
#define DAVIS_CONFIG_DVS_ACK_EXTENSION_COLUMN 7
```

Parameter address for module DAVIS\_CONFIG\_DVS: extend the length of the acknowledge on the AER bus for the column events (serial AER protocol) by this many LogicClock cycles.

**4.1.2.281 DAVIS\_CONFIG\_DVS\_ACK\_EXTENSION\_ROW**

```
#define DAVIS_CONFIG_DVS_ACK_EXTENSION_ROW 6
```

Parameter address for module DAVIS\_CONFIG\_DVS: extend the length of the acknowledge on the AER bus for the row events (serial AER protocol) by this many LogicClock cycles.

#### 4.1.2.282 DAVIS\_CONFIG\_DVS\_EXTERNAL\_AER\_CONTROL

```
#define DAVIS_CONFIG_DVS_EXTERNAL_AER_CONTROL 10
```

Parameter address for module DAVIS\_CONFIG\_DVS: enable external AER control. This ensures the chip and the DVS pixel array are running, but doesn't do the handshake and leaves the ACK pin in high-impedance, to allow for an external system to take over the AER communication with the chip. DAVIS\_CONFIG\_DVS\_RUN has to be turned off for this to work.

#### 4.1.2.283 DAVIS\_CONFIG\_DVS\_FILTER\_BACKGROUND\_ACTIVITY

```
#define DAVIS_CONFIG_DVS_FILTER_BACKGROUND_ACTIVITY 29
```

Parameter address for module DAVIS\_CONFIG\_DVS: enable the background-activity filter, which tries to remove events caused by transistor leakage, by rejecting uncorrelated events.

#### 4.1.2.284 DAVIS\_CONFIG\_DVS\_FILTER\_BACKGROUND\_ACTIVITY\_DELTAT

```
#define DAVIS_CONFIG_DVS_FILTER_BACKGROUND_ACTIVITY_DELTAT 30
```

Parameter address for module DAVIS\_CONFIG\_DVS: specify the time difference constant for the background-activity filter in microseconds. Events that do correlated within this time-frame are let through, while others are filtered out.

#### 4.1.2.285 DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_0\_COLUMN

```
#define DAVIS_CONFIG_DVS_FILTER_PIXEL_0_COLUMN 13
```

Parameter address for module DAVIS\_CONFIG\_DVS: the pixel filter completely suppresses up to eight pixels in the DVS array, filtering out all events produced by them. This is the pixel 0, X axis setting.

#### 4.1.2.286 DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_0\_ROW

```
#define DAVIS_CONFIG_DVS_FILTER_PIXEL_0_ROW 12
```

Parameter address for module DAVIS\_CONFIG\_DVS: the pixel filter completely suppresses up to eight pixels in the DVS array, filtering out all events produced by them. This is the pixel 0, Y axis setting.

#### 4.1.2.287 DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_1\_COLUMN

```
#define DAVIS_CONFIG_DVS_FILTER_PIXEL_1_COLUMN 15
```

Parameter address for module DAVIS\_CONFIG\_DVS: the pixel filter completely suppresses up to eight pixels in the DVS array, filtering out all events produced by them. This is the pixel 1, X axis setting.



**4.1.2.288 DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_1\_ROW**

```
#define DAVIS_CONFIG_DVS_FILTER_PIXEL_1_ROW 14
```

Parameter address for module DAVIS\_CONFIG\_DVS: the pixel filter completely suppresses up to eight pixels in the DVS array, filtering out all events produced by them. This is the pixel 1, Y axis setting.

**4.1.2.289 DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_2\_COLUMN**

```
#define DAVIS_CONFIG_DVS_FILTER_PIXEL_2_COLUMN 17
```

Parameter address for module DAVIS\_CONFIG\_DVS: the pixel filter completely suppresses up to eight pixels in the DVS array, filtering out all events produced by them. This is the pixel 2, X axis setting.

**4.1.2.290 DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_2\_ROW**

```
#define DAVIS_CONFIG_DVS_FILTER_PIXEL_2_ROW 16
```

Parameter address for module DAVIS\_CONFIG\_DVS: the pixel filter completely suppresses up to eight pixels in the DVS array, filtering out all events produced by them. This is the pixel 2, Y axis setting.

**4.1.2.291 DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_3\_COLUMN**

```
#define DAVIS_CONFIG_DVS_FILTER_PIXEL_3_COLUMN 19
```

Parameter address for module DAVIS\_CONFIG\_DVS: the pixel filter completely suppresses up to eight pixels in the DVS array, filtering out all events produced by them. This is the pixel 3, X axis setting.

**4.1.2.292 DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_3\_ROW**

```
#define DAVIS_CONFIG_DVS_FILTER_PIXEL_3_ROW 18
```

Parameter address for module DAVIS\_CONFIG\_DVS: the pixel filter completely suppresses up to eight pixels in the DVS array, filtering out all events produced by them. This is the pixel 3, Y axis setting.

**4.1.2.293 DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_4\_COLUMN**

```
#define DAVIS_CONFIG_DVS_FILTER_PIXEL_4_COLUMN 21
```

Parameter address for module DAVIS\_CONFIG\_DVS: the pixel filter completely suppresses up to eight pixels in the DVS array, filtering out all events produced by them. This is the pixel 4, X axis setting.

**4.1.2.294 DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_4\_ROW**

```
#define DAVIS_CONFIG_DVS_FILTER_PIXEL_4_ROW 20
```

Parameter address for module DAVIS\_CONFIG\_DVS: the pixel filter completely suppresses up to eight pixels in the DVS array, filtering out all events produced by them. This is the pixel 4, Y axis setting.

#### 4.1.2.295 DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_5\_COLUMN

```
#define DAVIS_CONFIG_DVS_FILTER_PIXEL_5_COLUMN 23
```

Parameter address for module DAVIS\_CONFIG\_DVS: the pixel filter completely suppresses up to eight pixels in the DVS array, filtering out all events produced by them. This is the pixel 5, X axis setting.

#### 4.1.2.296 DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_5\_ROW

```
#define DAVIS_CONFIG_DVS_FILTER_PIXEL_5_ROW 22
```

Parameter address for module DAVIS\_CONFIG\_DVS: the pixel filter completely suppresses up to eight pixels in the DVS array, filtering out all events produced by them. This is the pixel 5, Y axis setting.

#### 4.1.2.297 DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_6\_COLUMN

```
#define DAVIS_CONFIG_DVS_FILTER_PIXEL_6_COLUMN 25
```

Parameter address for module DAVIS\_CONFIG\_DVS: the pixel filter completely suppresses up to eight pixels in the DVS array, filtering out all events produced by them. This is the pixel 6, X axis setting.

#### 4.1.2.298 DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_6\_ROW

```
#define DAVIS_CONFIG_DVS_FILTER_PIXEL_6_ROW 24
```

Parameter address for module DAVIS\_CONFIG\_DVS: the pixel filter completely suppresses up to eight pixels in the DVS array, filtering out all events produced by them. This is the pixel 6, Y axis setting.

#### 4.1.2.299 DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_7\_COLUMN

```
#define DAVIS_CONFIG_DVS_FILTER_PIXEL_7_COLUMN 27
```

Parameter address for module DAVIS\_CONFIG\_DVS: the pixel filter completely suppresses up to eight pixels in the DVS array, filtering out all events produced by them. This is the pixel 7, X axis setting.

#### 4.1.2.300 DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_7\_ROW

```
#define DAVIS_CONFIG_DVS_FILTER_PIXEL_7_ROW 26
```

Parameter address for module DAVIS\_CONFIG\_DVS: the pixel filter completely suppresses up to eight pixels in the DVS array, filtering out all events produced by them. This is the pixel 7, Y axis setting.

#### 4.1.2.301 DAVIS\_CONFIG\_DVS\_FILTER\_ROW\_ONLY\_EVENTS

```
#define DAVIS_CONFIG_DVS_FILTER_ROW_ONLY_EVENTS 9
```

Parameter address for module DAVIS\_CONFIG\_DVS: enable row-only event filter, to eliminate spurious row events with no following columns events. This can happen on DAVIS240 chips, or following the various pixel and background-activity filtering stages, which drop column events to achieve their effect. This should always be enabled!

#### 4.1.2.302 DAVIS\_CONFIG\_DVS\_HAS\_BACKGROUND\_ACTIVITY\_FILTER

```
#define DAVIS_CONFIG_DVS_HAS_BACKGROUND_ACTIVITY_FILTER 28
```

Parameter address for module DAVIS\_CONFIG\_DVS: read-only parameter, information about the presence of the background-activity filter feature. This is reserved for internal use and should not be used by anything other than libcaer. Please see the 'struct [caer\\_davis\\_info](#)' documentation to get this information.

#### 4.1.2.303 DAVIS\_CONFIG\_DVS\_HAS\_PIXEL\_FILTER

```
#define DAVIS_CONFIG_DVS_HAS_PIXEL_FILTER 11
```

Parameter address for module DAVIS\_CONFIG\_DVS: read-only parameter, information about the presence of the pixel filter feature. This is reserved for internal use and should not be used by anything other than libcaer. Please see the 'struct [caer\\_davis\\_info](#)' documentation to get this information.

#### 4.1.2.304 DAVIS\_CONFIG\_DVS\_HAS\_TEST\_EVENT\_GENERATOR

```
#define DAVIS_CONFIG_DVS_HAS_TEST_EVENT_GENERATOR 31
```

Parameter address for module DAVIS\_CONFIG\_DVS: read-only parameter, information about the presence of the test event generator feature. This is reserved for internal use and should not be used by anything other than libcaer. Please see the 'struct [caer\\_davis\\_info](#)' documentation to get this information.

#### 4.1.2.305 DAVIS\_CONFIG\_DVS\_ORIENTATION\_INFO

```
#define DAVIS_CONFIG_DVS_ORIENTATION_INFO 2
```

Parameter address for module DAVIS\_CONFIG\_DVS: read-only parameter, contains information on the orientation of the X/Y axes, whether they should be inverted or not on the host when parsing incoming events. Bit 2: dvsInvert↔XY Bit 1: reserved Bit 0: reserved This is reserved for internal use and should not be used by anything other than libcaer. Please see the 'struct [caer\\_davis\\_info](#)' documentation to get proper size information that already considers the rotation and orientation settings.

#### 4.1.2.306 DAVIS\_CONFIG\_DVS\_RUN

```
#define DAVIS_CONFIG_DVS_RUN 3
```

Parameter address for module DAVIS\_CONFIG\_DVS: run the DVS state machine and get polarity events from the chip by handshaking with its AER bus.

#### 4.1.2.307 DAVIS\_CONFIG\_DVS\_SIZE\_COLUMNS

```
#define DAVIS_CONFIG_DVS_SIZE_COLUMNS 0
```

Parameter address for module DAVIS\_CONFIG\_DVS: read-only parameter, contains the X axis resolution of the DVS events returned by the camera. This is reserved for internal use and should not be used by anything other than libcaer. Please see the 'struct [caer\\_davis\\_info](#)' documentation to get proper size information that already considers the rotation and orientation settings.

**4.1.2.308 DAVIS\_CONFIG\_DVS\_SIZE\_ROWS**

```
#define DAVIS_CONFIG_DVS_SIZE_ROWS 1
```

Parameter address for module DAVIS\_CONFIG\_DVS: read-only parameter, contains the Y axis resolution of the DVS events returned by the camera. This is reserved for internal use and should not be used by anything other than libcaer. Please see the 'struct [caer\\_davis\\_info](#)' documentation to get proper size information that already considers the rotation and orientation settings.

**4.1.2.309 DAVIS\_CONFIG\_DVS\_TEST\_EVENT\_GENERATOR\_ENABLE**

```
#define DAVIS_CONFIG_DVS_TEST_EVENT_GENERATOR_ENABLE 32
```

Parameter address for module DAVIS\_CONFIG\_DVS: enable the test event generator for debugging purposes. This generates fake events that appear to originate from all rows sequentially, and for each row going through all its columns, first with an ON polarity and then with an OFF polarity. Both DAVIS\_CONFIG\_DVS\_RUN and DAVIS\_CONFIG\_DVS\_EXTERNAL\_AER\_CONTROL have to be turned off for this to work.

**4.1.2.310 DAVIS\_CONFIG\_DVS\_WAIT\_ON\_TRANSFER\_STALL**

```
#define DAVIS_CONFIG_DVS_WAIT_ON_TRANSFER_STALL 8
```

Parameter address for module DAVIS\_CONFIG\_DVS: if the output FIFO for this module is full, stall the AER handshake with the chip and wait until it's free again, instead of just continuing the handshake and dropping the resulting events.

**4.1.2.311 DAVIS\_CONFIG\_EXTINPUT**

```
#define DAVIS_CONFIG_EXTINPUT 4
```

Module address: device-side External Input (signal detector/generator) configuration. The External Input module is used to detect external signals on the external input jack and inject an event into the event stream when this happens. It can detect pulses of a specific length or rising and falling edges. On some systems, a signal generator module is also present, which can generate PWM-like pulsed signals with configurable timing.

**4.1.2.312 DAVIS\_CONFIG\_EXTINPUT\_DETECT\_FALLING\_EDGES**

```
#define DAVIS_CONFIG_EXTINPUT_DETECT_FALLING_EDGES 2
```

Parameter address for module DAVIS\_CONFIG\_EXTINPUT: send a special EXTERNAL\_INPUT\_FALLING\_EDGE event when a falling edge is detected (transition from high voltage to low).

**4.1.2.313 DAVIS\_CONFIG\_EXTINPUT\_DETECT\_FALLING\_EDGES1**

```
#define DAVIS_CONFIG_EXTINPUT_DETECT_FALLING_EDGES1 17
```

Parameter address for module DAVIS\_CONFIG\_EXTINPUT: send a special EXTERNAL\_INPUT1\_FALLING\_EDGE event when a falling edge is detected (transition from high voltage to low).

#### 4.1.2.314 DAVIS\_CONFIG\_EXTINPUT\_DETECT\_FALLING\_EDGES2

```
#define DAVIS_CONFIG_EXTINPUT_DETECT_FALLING_EDGES2 23
```

Parameter address for module DAVIS\_CONFIG\_EXTINPUT: send a special EXTERNAL\_INPUT2\_FALLING\_EDGE event when a falling edge is detected (transition from high voltage to low).

#### 4.1.2.315 DAVIS\_CONFIG\_EXTINPUT\_DETECT\_PULSE\_LENGTH

```
#define DAVIS_CONFIG_EXTINPUT_DETECT_PULSE_LENGTH 5
```

Parameter address for module DAVIS\_CONFIG\_EXTINPUT: the minimal length that a pulse must have to trigger the sending of a special event. This is measured in cycles at LogicClock frequency (see 'struct [caer\\_davis\\_info](#)' for details on how to get the frequency).

#### 4.1.2.316 DAVIS\_CONFIG\_EXTINPUT\_DETECT\_PULSE\_LENGTH1

```
#define DAVIS_CONFIG_EXTINPUT_DETECT_PULSE_LENGTH1 20
```

Parameter address for module DAVIS\_CONFIG\_EXTINPUT: the minimal length that a pulse must have to trigger the sending of a special event. This is measured in cycles at LogicClock frequency (see 'struct [caer\\_davis\\_info](#)' for details on how to get the frequency).

#### 4.1.2.317 DAVIS\_CONFIG\_EXTINPUT\_DETECT\_PULSE\_LENGTH2

```
#define DAVIS_CONFIG_EXTINPUT_DETECT_PULSE_LENGTH2 26
```

Parameter address for module DAVIS\_CONFIG\_EXTINPUT: the minimal length that a pulse must have to trigger the sending of a special event. This is measured in cycles at LogicClock frequency (see 'struct [caer\\_davis\\_info](#)' for details on how to get the frequency).

#### 4.1.2.318 DAVIS\_CONFIG\_EXTINPUT\_DETECT\_PULSE\_POLARITY

```
#define DAVIS_CONFIG_EXTINPUT_DETECT_PULSE_POLARITY 4
```

Parameter address for module DAVIS\_CONFIG\_EXTINPUT: the polarity the pulse must exhibit to be detected as such. '1' means active high; a pulse will start when the signal goes from low to high and will continue to be seen as the same pulse as long as it stays high. '0' means active low; a pulse will start when the signal goes from high to low and will continue to be seen as the same pulse as long as it stays low.

#### 4.1.2.319 DAVIS\_CONFIG\_EXTINPUT\_DETECT\_PULSE\_POLARITY1

```
#define DAVIS_CONFIG_EXTINPUT_DETECT_PULSE_POLARITY1 19
```

Parameter address for module DAVIS\_CONFIG\_EXTINPUT: the polarity the pulse must exhibit to be detected as such. '1' means active high; a pulse will start when the signal goes from low to high and will continue to be seen as the same pulse as long as it stays high. '0' means active low; a pulse will start when the signal goes from high to low and will continue to be seen as the same pulse as long as it stays low.

#### 4.1.2.320 DAVIS\_CONFIG\_EXTINPUT\_DETECT\_PULSE\_POLARITY2

```
#define DAVIS_CONFIG_EXTINPUT_DETECT_PULSE_POLARITY2 25
```

Parameter address for module DAVIS\_CONFIG\_EXTINPUT: the polarity the pulse must exhibit to be detected as such. '1' means active high; a pulse will start when the signal goes from low to high and will continue to be seen as the same pulse as long as it stays high. '0' means active low; a pulse will start when the signal goes from high to low and will continue to be seen as the same pulse as long as it stays low.

#### 4.1.2.321 DAVIS\_CONFIG\_EXTINPUT\_DETECT\_PULSES

```
#define DAVIS_CONFIG_EXTINPUT_DETECT_PULSES 3
```

Parameter address for module DAVIS\_CONFIG\_EXTINPUT: send a special EXTERNAL\_INPUT\_PULSE event when a pulse, of a specified, configurable polarity and length, is detected. See DAVIS\_CONFIG\_EXTINPUT\_DETECT\_PULSE\_POLARITY and DAVIS\_CONFIG\_EXTINPUT\_DETECT\_PULSE\_LENGTH for more details.

#### 4.1.2.322 DAVIS\_CONFIG\_EXTINPUT\_DETECT\_PULSES1

```
#define DAVIS_CONFIG_EXTINPUT_DETECT_PULSES1 18
```

Parameter address for module DAVIS\_CONFIG\_EXTINPUT: send a special EXTERNAL\_INPUT1\_PULSE event when a pulse, of a specified, configurable polarity and length, is detected. See DAVIS\_CONFIG\_EXTINPUT\_DETECT\_PULSE\_POLARITY1 and DAVIS\_CONFIG\_EXTINPUT\_DETECT\_PULSE\_LENGTH1 for more details.

#### 4.1.2.323 DAVIS\_CONFIG\_EXTINPUT\_DETECT\_PULSES2

```
#define DAVIS_CONFIG_EXTINPUT_DETECT_PULSES2 24
```

Parameter address for module DAVIS\_CONFIG\_EXTINPUT: send a special EXTERNAL\_INPUT2\_PULSE event when a pulse, of a specified, configurable polarity and length, is detected. See DAVIS\_CONFIG\_EXTINPUT\_DETECT\_PULSE\_POLARITY2 and DAVIS\_CONFIG\_EXTINPUT\_DETECT\_PULSE\_LENGTH2 for more details.

#### 4.1.2.324 DAVIS\_CONFIG\_EXTINPUT\_DETECT\_RISING\_EDGES

```
#define DAVIS_CONFIG_EXTINPUT_DETECT_RISING_EDGES 1
```

Parameter address for module DAVIS\_CONFIG\_EXTINPUT: send a special EXTERNAL\_INPUT\_RISING\_EDGE event when a rising edge is detected (transition from low voltage to high).

#### 4.1.2.325 DAVIS\_CONFIG\_EXTINPUT\_DETECT\_RISING\_EDGES1

```
#define DAVIS_CONFIG_EXTINPUT_DETECT_RISING_EDGES1 16
```

Parameter address for module DAVIS\_CONFIG\_EXTINPUT: send a special EXTERNAL\_INPUT1\_RISING\_EDGE event when a rising edge is detected (transition from low voltage to high).

#### 4.1.2.326 DAVIS\_CONFIG\_EXTINPUT\_DETECT\_RISING\_EDGES2

```
#define DAVIS_CONFIG_EXTINPUT_DETECT_RISING_EDGES2 22
```

Parameter address for module DAVIS\_CONFIG\_EXTINPUT: send a special EXTERNAL\_INPUT2\_RISING\_EDGE event when a rising edge is detected (transition from low voltage to high).

#### 4.1.2.327 DAVIS\_CONFIG\_EXTINPUT\_GENERATE\_INJECT\_ON\_FALLING\_EDGE

```
#define DAVIS_CONFIG_EXTINPUT_GENERATE_INJECT_ON_FALLING_EDGE 13
```

Parameter address for module DAVIS\_CONFIG\_EXTINPUT: enables event injection when a falling edge occurs in the generated signal; a special event EXTERNAL\_GENERATOR\_FALLING\_EDGE is emitted into the event stream.

#### 4.1.2.328 DAVIS\_CONFIG\_EXTINPUT\_GENERATE\_INJECT\_ON\_RISING\_EDGE

```
#define DAVIS_CONFIG_EXTINPUT_GENERATE_INJECT_ON_RISING_EDGE 12
```

Parameter address for module DAVIS\_CONFIG\_EXTINPUT: enables event injection when a rising edge occurs in the generated signal; a special event EXTERNAL\_GENERATOR\_RISING\_EDGE is emitted into the event stream.

#### 4.1.2.329 DAVIS\_CONFIG\_EXTINPUT\_GENERATE\_PULSE\_INTERVAL

```
#define DAVIS_CONFIG_EXTINPUT_GENERATE_PULSE_INTERVAL 10
```

Parameter address for module DAVIS\_CONFIG\_EXTINPUT: the interval between the start of two consecutive pulses, expressed in cycles at LogicClock frequency (see 'struct [caer\\_davis\\_info](#)' for details on how to get the frequency). This must be bigger or equal to DAVIS\_CONFIG\_EXTINPUT\_GENERATE\_PULSE\_LENGTH. To generate a signal with 50% duty cycle, this would have to be exactly double of DAVIS\_CONFIG\_EXTINPUT\_GENERATE\_PULSE\_LENGTH.

#### 4.1.2.330 DAVIS\_CONFIG\_EXTINPUT\_GENERATE\_PULSE\_LENGTH

```
#define DAVIS_CONFIG_EXTINPUT_GENERATE_PULSE_LENGTH 11
```

Parameter address for module DAVIS\_CONFIG\_EXTINPUT: the length a pulse stays active, expressed in cycles at LogicClock frequency (see 'struct [caer\\_davis\\_info](#)' for details on how to get the frequency). This must be smaller or equal to DAVIS\_CONFIG\_EXTINPUT\_GENERATE\_PULSE\_INTERVAL. To generate a signal with 50% duty cycle, this would have to be exactly half of DAVIS\_CONFIG\_EXTINPUT\_GENERATE\_PULSE\_INTERVAL.

#### 4.1.2.331 DAVIS\_CONFIG\_EXTINPUT\_GENERATE\_PULSE\_POLARITY

```
#define DAVIS_CONFIG_EXTINPUT_GENERATE_PULSE_POLARITY 9
```

Parameter address for module DAVIS\_CONFIG\_EXTINPUT: polarity of the PWM-like signal to be generated. '1' means active high, '0' means active low.

#### 4.1.2.332 DAVIS\_CONFIG\_EXTINPUT\_GENERATE\_USE\_CUSTOM\_SIGNAL

```
#define DAVIS_CONFIG_EXTINPUT_GENERATE_USE_CUSTOM_SIGNAL 8
```

Parameter address for module DAVIS\_CONFIG\_EXTINPUT: instead of generating a PWM-like signal by using the configured parameters, use a signal on the FPGA/CPLD that's passed as an input to the External Input module. By default this is disabled and tied to ground, but it can be useful for customized logic designs.

#### 4.1.2.333 DAVIS\_CONFIG\_EXTINPUT\_HAS\_EXTRA\_DETECTORS

```
#define DAVIS_CONFIG_EXTINPUT_HAS_EXTRA_DETECTORS 14
```

Parameter address for module DAVIS\_CONFIG\_EXTINPUT: read-only parameter, information about the presence of the extra detectors feature. This is reserved for internal use and should not be used by anything other than libcaer. Please see the 'struct [caer\\_davis\\_info](#)' documentation to get this information.

#### 4.1.2.334 DAVIS\_CONFIG\_EXTINPUT\_HAS\_GENERATOR

```
#define DAVIS_CONFIG_EXTINPUT_HAS_GENERATOR 6
```

Parameter address for module DAVIS\_CONFIG\_EXTINPUT: read-only parameter, information about the presence of the signal generator feature. This is reserved for internal use and should not be used by anything other than libcaer. Please see the 'struct [caer\\_davis\\_info](#)' documentation to get this information.

#### 4.1.2.335 DAVIS\_CONFIG\_EXTINPUT\_RUN\_DETECTOR

```
#define DAVIS_CONFIG_EXTINPUT_RUN_DETECTOR 0
```

Parameter address for module DAVIS\_CONFIG\_EXTINPUT: enable the signal detector module. It generates events when it sees certain types of signals, such as edges or pulses of a defined length, on the IN JACK signal. This can be useful to inject events into the event stream in response to external stimuli or controls, such as turning on a LED lamp.

#### 4.1.2.336 DAVIS\_CONFIG\_EXTINPUT\_RUN\_DETECTOR1

```
#define DAVIS_CONFIG_EXTINPUT_RUN_DETECTOR1 15
```

Parameter address for module DAVIS\_CONFIG\_EXTINPUT: enable the signal detector module. It generates events when it sees certain types of signals, such as edges or pulses of a defined length, on the B1P20 input pin. This can be useful to inject events into the event stream in response to external stimuli or controls, such as turning on a LED lamp.

#### 4.1.2.337 DAVIS\_CONFIG\_EXTINPUT\_RUN\_DETECTOR2

```
#define DAVIS_CONFIG_EXTINPUT_RUN_DETECTOR2 21
```

Parameter address for module DAVIS\_CONFIG\_EXTINPUT: enable the signal detector module. It generates events when it sees certain types of signals, such as edges or pulses of a defined length, on the B1P21 input pin. This can be useful to inject events into the event stream in response to external stimuli or controls, such as turning on a LED lamp.



**4.1.2.338 DAVIS\_CONFIG\_EXTINPUT\_RUN\_GENERATOR**

```
#define DAVIS_CONFIG_EXTINPUT_RUN_GENERATOR 7
```

Parameter address for module DAVIS\_CONFIG\_EXTINPUT: enable the signal generator module. It generates a PWM-like signal based on configurable parameters and outputs it on the OUT JACK signal.

**4.1.2.339 DAVIS\_CONFIG\_IMU**

```
#define DAVIS_CONFIG_IMU 3
```

Module address: device-side IMU (Inertial Measurement Unit) configuration. The IMU module connects to the external IMU chip and sends data on the device's movement in space. It can configure various options on the external chip, such as accelerometer range or gyroscope refresh rate.

**4.1.2.340 DAVIS\_CONFIG\_IMU\_ACCEL\_FULL\_SCALE**

```
#define DAVIS_CONFIG_IMU_ACCEL_FULL_SCALE 8
```

Parameter address for module DAVIS\_CONFIG\_IMU: select the full scale range of the accelerometer outputs. Valid values are: 0 - +- 2 g 1 - +- 4 g 2 - +- 8 g 3 - +- 16 g

**4.1.2.341 DAVIS\_CONFIG\_IMU\_ACCEL\_STANDBY**

```
#define DAVIS_CONFIG_IMU_ACCEL_STANDBY 2
```

Parameter address for module DAVIS\_CONFIG\_IMU: put the accelerometer sensor in standby, disabling it.

**4.1.2.342 DAVIS\_CONFIG\_IMU\_DIGITAL\_LOW\_PASS\_FILTER**

```
#define DAVIS_CONFIG_IMU_DIGITAL_LOW_PASS_FILTER 7
```

Parameter address for module DAVIS\_CONFIG\_IMU: this configures the digital low-pass filter for both the accelerometer and the gyroscope. Valid values are from 0 to 7 and have the following meaning: 0 - Accel: BW=260Hz, Delay=0ms, FS=1kHz - Gyro: BW=256Hz, Delay=0.98ms, FS=8kHz 1 - Accel: BW=184Hz, Delay=2.0ms, FS=1kHz - Gyro: BW=188Hz, Delay=1.9ms, FS=1kHz 2 - Accel: BW=94Hz, Delay=3.0ms, FS=1kHz - Gyro: BW=98Hz, Delay=2.8ms, FS=1kHz 3 - Accel: BW=44Hz, Delay=4.9ms, FS=1kHz - Gyro: BW=42Hz, Delay=4.8ms, FS=1kHz 4 - Accel: BW=21Hz, Delay=8.5ms, FS=1kHz - Gyro: BW=20Hz, Delay=8.3ms, FS=1kHz 5 - Accel: BW=10Hz, Delay=13.8ms, FS=1kHz - Gyro: BW=10Hz, Delay=13.4ms, FS=1kHz 6 - Accel: BW=5Hz, Delay=19.0ms, FS=1kHz - Gyro: BW=5Hz, Delay=18.6ms, FS=1kHz 7 - Accel: RESERVED, FS=1kHz - Gyro: RESERVED, FS=8kHz

**4.1.2.343 DAVIS\_CONFIG\_IMU\_GYRO\_FULL\_SCALE**

```
#define DAVIS_CONFIG_IMU_GYRO_FULL_SCALE 9
```

Parameter address for module DAVIS\_CONFIG\_IMU: select the full scale range of the gyroscope outputs. Valid values are: 0 - +- 250 °/s 1 - +- 500 °/s 2 - +- 1000 °/s 3 - +- 2000 °/s

#### 4.1.2.344 DAVIS\_CONFIG\_IMU\_GYRO\_STANDBY

```
#define DAVIS_CONFIG_IMU_GYRO_STANDBY 3
```

Parameter address for module DAVIS\_CONFIG\_IMU: put the gyroscope sensor in standby, disabling it.

#### 4.1.2.345 DAVIS\_CONFIG\_IMU\_LP\_CYCLE

```
#define DAVIS_CONFIG_IMU_LP_CYCLE 4
```

Parameter address for module DAVIS\_CONFIG\_IMU: put the IMU into Cycle Mode. In Cycle Mode, the device cycles between sleep mode and waking up to take a single sample of data from the accelerometer at a rate determined by DAVIS\_CONFIG\_IMU\_LP\_WAKEUP.

#### 4.1.2.346 DAVIS\_CONFIG\_IMU\_LP\_WAKEUP

```
#define DAVIS_CONFIG_IMU_LP_WAKEUP 5
```

Parameter address for module DAVIS\_CONFIG\_IMU: rate at which the IMU takes an accelerometer sample while in Cycle Mode (see DAVIS\_CONFIG\_IMU\_LP\_CYCLE). Valid values are: 0 - 1.25 Hz wake-up frequency 1 - 5 Hz wake-up frequency 2 - 20 Hz wake-up frequency 3 - 40 Hz wake-up frequency

#### 4.1.2.347 DAVIS\_CONFIG\_IMU\_ORIENTATION\_INFO

```
#define DAVIS_CONFIG_IMU_ORIENTATION_INFO 10
```

Parameter address for module DAVIS\_CONFIG\_IMU: read-only parameter, contains information on the orientation of the X/Y/Z axes, whether they should be flipped or not on the host when parsing incoming IMU data samples. Bit 2: imuFlipX Bit 1: imuFlipY Bit 0: imuFlipZ This is reserved for internal use and should not be used by anything other than libcaer. Generated IMU events are already properly flipped when returned to the user.

#### 4.1.2.348 DAVIS\_CONFIG\_IMU\_RUN

```
#define DAVIS_CONFIG_IMU_RUN 0
```

Parameter address for module DAVIS\_CONFIG\_IMU: run the IMU state machine to get information about the movement and position of the device. This takes the IMU chip out of sleep.

#### 4.1.2.349 DAVIS\_CONFIG\_IMU\_SAMPLE\_RATE\_DIVIDER

```
#define DAVIS_CONFIG_IMU_SAMPLE_RATE_DIVIDER 6
```

Parameter address for module DAVIS\_CONFIG\_IMU: this specifies the divider from the Gyroscope Output Rate used to generate the Sample Rate for the IMU. Valid values are from 0 to 255. The Sample Rate is generated like this:  $\text{Sample Rate} = \text{Gyroscope Output Rate} / (1 + \text{DAVIS\_CONFIG\_IMU\_SAMPLE\_RATE\_DIVIDER})$  where Gyroscope Output Rate = 8 kHz when DAVIS\_CONFIG\_IMU\_DIGITAL\_LOW\_PASS\_FILTER is disabled (set to 0 or 7), and 1 kHz when enabled. Note: the accelerometer output rate is 1 kHz. This means that for a Sample Rate greater than 1 kHz, the same accelerometer sample may be output multiple times.

#### 4.1.2.350 DAVIS\_CONFIG\_IMU\_TEMP\_STANDBY

```
#define DAVIS_CONFIG_IMU_TEMP_STANDBY 1
```

Parameter address for module DAVIS\_CONFIG\_IMU: put the temperature sensor in standby, disabling it.

#### 4.1.2.351 DAVIS\_CONFIG\_MICROPHONE

```
#define DAVIS_CONFIG_MICROPHONE 7
```

Module address: device-side microphone configuration. The Microphone module enables the use of InvenSense stereo microphones to capture samples of sound from devices that support is, such as the miniDAVIS346.

#### 4.1.2.352 DAVIS\_CONFIG\_MICROPHONE\_RUN

```
#define DAVIS_CONFIG_MICROPHONE_RUN 0
```

Parameter address for module DAVIS\_CONFIG\_MICROPHONE: enable the Microphone module, which provides stereo samples of sound recorded by on-board InvenSense microphones.

#### 4.1.2.353 DAVIS\_CONFIG\_MICROPHONE\_SAMPLE\_FREQUENCY

```
#define DAVIS_CONFIG_MICROPHONE_SAMPLE_FREQUENCY 1
```

Parameter address for module DAVIS\_CONFIG\_MICROPHONE: allows setting the sample frequency of the stereo microphones, by specifying the length of an SCK clock cycle in LogicClock cycles. Value can be between 30 and 215 inclusive. The desired value can be calculated in the following way:  $\text{floor}(100'000'000/64/ <\text{desired freq} > \text{ in } \text{hz} >= \text{in})$  For example for 48 KHz sampling frequency, this would be 32. For 44.1 KHz it would be 35, and for 16 KHz it would be 97.

#### 4.1.2.354 DAVIS\_CONFIG\_MUX

```
#define DAVIS_CONFIG_MUX 0
```

Module address: device-side Multiplexer configuration. The Multiplexer is responsible for mixing, timestamping and outputting (via USB) the various event types generated by the device. It is also responsible for timestamp generation and synchronization.

#### 4.1.2.355 DAVIS\_CONFIG\_MUX\_DROP\_APS\_ON\_TRANSFER\_STALL

```
#define DAVIS_CONFIG_MUX_DROP_APS_ON_TRANSFER_STALL 5
```

Parameter address for module DAVIS\_CONFIG\_MUX: drop APS events if the USB output FIFO is full, instead of having them pile up at the input FIFOs. This normally should not be enabled to guarantee complete, coherent frame events, though small timing differences may cause a reduction in observed image quality.

**4.1.2.356 DAVIS\_CONFIG\_MUX\_DROP\_DVS\_ON\_TRANSFER\_STALL**

```
#define DAVIS_CONFIG_MUX_DROP_DVS_ON_TRANSFER_STALL 4
```

Parameter address for module DAVIS\_CONFIG\_MUX: drop DVS events if the USB output FIFO is full, instead of having them pile up at the input FIFOs.

**4.1.2.357 DAVIS\_CONFIG\_MUX\_DROP\_EXTINPUT\_ON\_TRANSFER\_STALL**

```
#define DAVIS_CONFIG_MUX_DROP_EXTINPUT_ON_TRANSFER_STALL 7
```

Parameter address for module DAVIS\_CONFIG\_MUX: drop External Input events if the USB output FIFO is full, instead of having them pile up at the input FIFOs.

**4.1.2.358 DAVIS\_CONFIG\_MUX\_DROP\_IMU\_ON\_TRANSFER\_STALL**

```
#define DAVIS_CONFIG_MUX_DROP_IMU_ON_TRANSFER_STALL 6
```

Parameter address for module DAVIS\_CONFIG\_MUX: drop IMU events if the USB output FIFO is full, instead of having them pile up at the input FIFOs. This normally should not be enabled to guarantee complete, coherent IMU events, and not get incomplete or wrong IMU information.

**4.1.2.359 DAVIS\_CONFIG\_MUX\_DROP\_MIC\_ON\_TRANSFER\_STALL**

```
#define DAVIS_CONFIG_MUX_DROP_MIC_ON_TRANSFER_STALL 8
```

Parameter address for module DAVIS\_CONFIG\_MUX: drop Microphone sample events if the USB output FIFO is full, instead of having them pile up at the input FIFOs.

**4.1.2.360 DAVIS\_CONFIG\_MUX\_FORCE\_CHIP\_BIAS\_ENABLE**

```
#define DAVIS_CONFIG_MUX_FORCE_CHIP_BIAS_ENABLE 3
```

Parameter address for module DAVIS\_CONFIG\_MUX: under normal circumstances, the chip's bias generator is only powered up when either the DVS or the APS state machines are running, to save power. This flag forces the bias generator to be powered up all the time, which may be useful when one wants to shut-down both APS and DVS temporarily, but still have a quick and well-defined resume behavior.

**4.1.2.361 DAVIS\_CONFIG\_MUX\_RUN**

```
#define DAVIS_CONFIG_MUX_RUN 0
```

Parameter address for module DAVIS\_CONFIG\_MUX: run the Multiplexer state machine, which is responsible for mixing the various event types at the device level, timestamping them and outputting them via USB or other connectors.

#### 4.1.2.362 DAVIS\_CONFIG\_MUX\_TIMESTAMP\_RESET

```
#define DAVIS_CONFIG_MUX_TIMESTAMP_RESET 2
```

Parameter address for module DAVIS\_CONFIG\_MUX: reset the Timestamp Generator to zero. This also sends a reset pulse to all connected slave devices, resetting their timestamp too.

#### 4.1.2.363 DAVIS\_CONFIG\_MUX\_TIMESTAMP\_RUN

```
#define DAVIS_CONFIG_MUX_TIMESTAMP_RUN 1
```

Parameter address for module DAVIS\_CONFIG\_MUX: run the Timestamp Generator inside the Multiplexer state machine, which will provide microsecond accurate timestamps to the events passing through.

#### 4.1.2.364 DAVIS\_CONFIG\_SYSINFO

```
#define DAVIS_CONFIG_SYSINFO 6
```

Module address: device-side system information. The system information module provides various details on the device, such as currently installed logic revision or clock speeds. All its parameters are read-only. This is reserved for internal use and should not be used by anything other than libcaer. Please see the 'struct [caer\\_davis\\_info](#)' documentation for more details on what information is available.

#### 4.1.2.365 DAVIS\_CONFIG\_SYSINFO\_ADC\_CLOCK

```
#define DAVIS_CONFIG_SYSINFO_ADC_CLOCK 4
```

Parameter address for module DAVIS\_CONFIG\_SYSINFO: read-only parameter, the frequency in MHz at which the FPGA/CPLD logic related to APS frame grabbing is running. This is reserved for internal use and should not be used by anything other than libcaer. Please see the 'struct [caer\\_davis\\_info](#)' documentation to get this information.

#### 4.1.2.366 DAVIS\_CONFIG\_SYSINFO\_CHIP\_IDENTIFIER

```
#define DAVIS_CONFIG_SYSINFO_CHIP_IDENTIFIER 1
```

Parameter address for module DAVIS\_CONFIG\_SYSINFO: read-only parameter, an integer used to identify the different types of sensor chips used on the device. This is reserved for internal use and should not be used by anything other than libcaer. Please see the 'struct [caer\\_davis\\_info](#)' documentation to get this information.

#### 4.1.2.367 DAVIS\_CONFIG\_SYSINFO\_DEVICE\_IS\_MASTER

```
#define DAVIS_CONFIG_SYSINFO_DEVICE_IS_MASTER 2
```

Parameter address for module DAVIS\_CONFIG\_SYSINFO: read-only parameter, whether the device is currently a timestamp master or slave when synchronizing multiple devices together. This is reserved for internal use and should not be used by anything other than libcaer. Please see the 'struct [caer\\_davis\\_info](#)' documentation to get this information.

#### 4.1.2.368 DAVIS\_CONFIG\_SYSINFO\_LOGIC\_CLOCK

```
#define DAVIS_CONFIG_SYSINFO_LOGIC_CLOCK 3
```

Parameter address for module DAVIS\_CONFIG\_SYSINFO: read-only parameter, the frequency in MHz at which the main FPGA/CPLD logic is running. This is reserved for internal use and should not be used by anything other than libcaer. Please see the 'struct [caer\\_davis\\_info](#)' documentation to get this information.

#### 4.1.2.369 DAVIS\_CONFIG\_SYSINFO\_LOGIC\_VERSION

```
#define DAVIS_CONFIG_SYSINFO_LOGIC_VERSION 0
```

Parameter address for module DAVIS\_CONFIG\_SYSINFO: read-only parameter, the version of the logic currently running on the device's FPGA/CPLD. This is reserved for internal use and should not be used by anything other than libcaer. Please see the 'struct [caer\\_davis\\_info](#)' documentation to get this information.

#### 4.1.2.370 DAVIS\_CONFIG\_USB

```
#define DAVIS_CONFIG_USB 9
```

Module address: device-side USB output configuration. The USB output module forwards the data from the device and the FPGA/CPLD to the USB chip, usually a Cypress FX2 or FX3.

#### 4.1.2.371 DAVIS\_CONFIG\_USB\_EARLY\_PACKET\_DELAY

```
#define DAVIS_CONFIG_USB_EARLY_PACKET_DELAY 1
```

Parameter address for module DAVIS\_CONFIG\_USB: the time delay after which a packet of data is committed to USB, even if it is not full yet (short USB packet). The value is in 125µs time-slices, corresponding to how USB schedules its operations (a value of 4 for example would mean waiting at most 0.5ms until sending a short USB packet to the host).

#### 4.1.2.372 DAVIS\_CONFIG\_USB\_RUN

```
#define DAVIS_CONFIG_USB_RUN 0
```

Parameter address for module DAVIS\_CONFIG\_USB: enable the USB FIFO module, which transfers the data from the FPGA/CPLD to the USB chip, to be then sent to the host. Turning this off will suppress any USB data communication!

#### 4.1.2.373 DAVISRGB\_CONFIG\_APS\_GSFDRESET

```
#define DAVISRGB_CONFIG_APS_GSFDRESET 55
```

Parameter address for module DAVIS\_CONFIG\_APS (only for DAVIS RGB chip): Global Shutter FD reset time in ADCClock cycles.

**4.1.2.374 DAVISRGB\_CONFIG\_APS\_GSPDRESET**

```
#define DAVISRGB_CONFIG_APS_GSPDRESET 52
```

Parameter address for module DAVIS\_CONFIG\_APS (only for DAVIS RGB chip): Global Shutter PD reset time in ADCClock cycles.

**4.1.2.375 DAVISRGB\_CONFIG\_APS\_GSRESETFALL**

```
#define DAVISRGB_CONFIG_APS_GSRESETFALL 53
```

Parameter address for module DAVIS\_CONFIG\_APS (only for DAVIS RGB chip): Global Shutter Reset Fall time in ADCClock cycles.

**4.1.2.376 DAVISRGB\_CONFIG\_APS\_GSTXFALL**

```
#define DAVISRGB_CONFIG_APS_GSTXFALL 54
```

Parameter address for module DAVIS\_CONFIG\_APS (only for DAVIS RGB chip): Global Shutter Transfer Fall time in ADCClock cycles.

**4.1.2.377 DAVISRGB\_CONFIG\_APS\_RSFDSETTLE**

```
#define DAVISRGB_CONFIG_APS_RSFDSETTLE 51
```

Parameter address for module DAVIS\_CONFIG\_APS (only for DAVIS RGB chip): Rolling Shutter FD settle time in ADCClock cycles.

**4.1.2.378 DAVISRGB\_CONFIG\_APS\_TRANSFER**

```
#define DAVISRGB_CONFIG_APS_TRANSFER 50
```

Parameter address for module DAVIS\_CONFIG\_APS (only for DAVIS RGB chip): charge transfer time in ADCClock cycles.

**4.1.2.379 DAVISRGB\_CONFIG\_BIAS\_ADCCOMPBP**

```
#define DAVISRGB_CONFIG_BIAS_ADCCOMPBP 27
```

Parameter address for module DAVISRGB\_CONFIG\_BIAS: DAVISRGB chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.380 DAVISRGB\_CONFIG\_BIAS\_ADCCREFHIGH

```
#define DAVISRGB_CONFIG_BIAS_ADCCREFHIGH 6
```

Parameter address for module DAVISRGB\_CONFIG\_BIAS: DAVISRGB chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.381 DAVISRGB\_CONFIG\_BIAS\_ADCCREFLOW

```
#define DAVISRGB_CONFIG_BIAS_ADCCREFLOW 7
```

Parameter address for module DAVISRGB\_CONFIG\_BIAS: DAVISRGB chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.382 DAVISRGB\_CONFIG\_BIAS\_ADCTESTVOLTAGE

```
#define DAVISRGB_CONFIG_BIAS_ADCTESTVOLTAGE 5
```

Parameter address for module DAVISRGB\_CONFIG\_BIAS: DAVISRGB chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.



**4.1.2.383 DAVISRGB\_CONFIG\_BIAS\_AEPDBN**

```
#define DAVISRGB_CONFIG_BIAS_AEPDBN 31
```

Parameter address for module DAVISRGB\_CONFIG\_BIAS: DAVISRGB chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

**4.1.2.384 DAVISRGB\_CONFIG\_BIAS\_AEPUXBP**

```
#define DAVISRGB_CONFIG_BIAS_AEPUXBP 32
```

Parameter address for module DAVISRGB\_CONFIG\_BIAS: DAVISRGB chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

**4.1.2.385 DAVISRGB\_CONFIG\_BIAS\_AEPUYBP**

```
#define DAVISRGB_CONFIG_BIAS_AEPUYBP 33
```

Parameter address for module DAVISRGB\_CONFIG\_BIAS: DAVISRGB chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.386 DAVISRGB\_CONFIG\_BIAS\_APSCAS

```
#define DAVISRGB_CONFIG_BIAS_APSCAS 0
```

Parameter address for module DAVISRGB\_CONFIG\_BIAS: DAVISRGB chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.387 DAVISRGB\_CONFIG\_BIAS\_APSROSFBN

```
#define DAVISRGB_CONFIG_BIAS_APSROSFBN 26
```

Parameter address for module DAVISRGB\_CONFIG\_BIAS: DAVISRGB chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.388 DAVISRGB\_CONFIG\_BIAS\_ARRAYBIASBUFFERBN

```
#define DAVISRGB_CONFIG_BIAS_ARRAYBIASBUFFERBN 20
```

Parameter address for module DAVISRGB\_CONFIG\_BIAS: DAVISRGB chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

**4.1.2.389 DAVISRGB\_CONFIG\_BIAS\_ARRAYLOGICBUFFERBN**

```
#define DAVISRGB_CONFIG_BIAS_ARRAYLOGICBUFFERBN 22
```

Parameter address for module DAVISRGB\_CONFIG\_BIAS: DAVISRGB chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

**4.1.2.390 DAVISRGB\_CONFIG\_BIAS\_BIASBUFFER**

```
#define DAVISRGB_CONFIG_BIAS_BIASBUFFER 34
```

Parameter address for module DAVISRGB\_CONFIG\_BIAS: DAVISRGB chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

**4.1.2.391 DAVISRGB\_CONFIG\_BIAS\_DACBUFBP**

```
#define DAVISRGB_CONFIG_BIAS_DACBUFBP 28
```

Parameter address for module DAVISRGB\_CONFIG\_BIAS: DAVISRGB chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.392 DAVISRGB\_CONFIG\_BIAS\_DIFFBN

```
#define DAVISRGB_CONFIG_BIAS_DIFFBN 14
```

Parameter address for module DAVISRGB\_CONFIG\_BIAS: DAVISRGB chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.393 DAVISRGB\_CONFIG\_BIAS\_FALLTIMEBN

```
#define DAVISRGB_CONFIG_BIAS_FALLTIMEBN 23
```

Parameter address for module DAVISRGB\_CONFIG\_BIAS: DAVISRGB chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.394 DAVISRGB\_CONFIG\_BIAS\_GND07

```
#define DAVISRGB_CONFIG_BIAS_GND07 4
```

Parameter address for module DAVISRGB\_CONFIG\_BIAS: DAVISRGB chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

**4.1.2.395 DAVISRGB\_CONFIG\_BIAS\_IFREFRBN**

```
#define DAVISRGB_CONFIG_BIAS_IFREFRBN 8
```

Parameter address for module DAVISRGB\_CONFIG\_BIAS: DAVISRGB chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

**4.1.2.396 DAVISRGB\_CONFIG\_BIAS\_IFTHRBN**

```
#define DAVISRGB_CONFIG_BIAS_IFTHRBN 9
```

Parameter address for module DAVISRGB\_CONFIG\_BIAS: DAVISRGB chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

**4.1.2.397 DAVISRGB\_CONFIG\_BIAS\_LCOLTIMEOUTBN**

```
#define DAVISRGB_CONFIG_BIAS_LCOLTIMEOUTBN 30
```

Parameter address for module DAVISRGB\_CONFIG\_BIAS: DAVISRGB chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.398 DAVISRGB\_CONFIG\_BIAS\_LOCALBUFBN

```
#define DAVISRGB_CONFIG_BIAS_LOCALBUFBN 10
```

Parameter address for module DAVISRGB\_CONFIG\_BIAS: DAVISRGB chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.399 DAVISRGB\_CONFIG\_BIAS\_OFFBN

```
#define DAVISRGB_CONFIG_BIAS_OFFBN 16
```

Parameter address for module DAVISRGB\_CONFIG\_BIAS: DAVISRGB chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.400 DAVISRGB\_CONFIG\_BIAS\_ONBN

```
#define DAVISRGB_CONFIG_BIAS_ONBN 15
```

Parameter address for module DAVISRGB\_CONFIG\_BIAS: DAVISRGB chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.401 DAVISRGB\_CONFIG\_BIAS\_OVG1LO

```
#define DAVISRGB_CONFIG_BIAS_OVG1LO 1
```

Parameter address for module DAVISRGB\_CONFIG\_BIAS: DAVISRGB chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.402 DAVISRGB\_CONFIG\_BIAS\_OVG2LO

```
#define DAVISRGB_CONFIG_BIAS_OVG2LO 2
```

Parameter address for module DAVISRGB\_CONFIG\_BIAS: DAVISRGB chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.403 DAVISRGB\_CONFIG\_BIAS\_PADFOLLBN

```
#define DAVISRGB_CONFIG_BIAS_PADFOLLBN 11
```

Parameter address for module DAVISRGB\_CONFIG\_BIAS: DAVISRGB chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.404 DAVISRGB\_CONFIG\_BIAS\_PIXINBN

```
#define DAVISRGB_CONFIG_BIAS_PIXINBN 13
```

Parameter address for module DAVISRGB\_CONFIG\_BIAS: DAVISRGB chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.405 DAVISRGB\_CONFIG\_BIAS\_PRBP

```
#define DAVISRGB_CONFIG_BIAS_PRBP 17
```

Parameter address for module DAVISRGB\_CONFIG\_BIAS: DAVISRGB chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.406 DAVISRGB\_CONFIG\_BIAS\_PRSFBP

```
#define DAVISRGB_CONFIG_BIAS_PRSFBP 18
```

Parameter address for module DAVISRGB\_CONFIG\_BIAS: DAVISRGB chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.



#### 4.1.2.407 DAVISRGB\_CONFIG\_BIAS\_READOUTBUFBP

```
#define DAVISRGB_CONFIG_BIAS_READOUTBUFBP 25
```

Parameter address for module DAVISRGB\_CONFIG\_BIAS: DAVISRGB chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.408 DAVISRGB\_CONFIG\_BIAS\_REFRBP

```
#define DAVISRGB_CONFIG_BIAS_REFRBP 19
```

Parameter address for module DAVISRGB\_CONFIG\_BIAS: DAVISRGB chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.409 DAVISRGB\_CONFIG\_BIAS\_RISETIMEBP

```
#define DAVISRGB_CONFIG_BIAS_RISETIMEBP 24
```

Parameter address for module DAVISRGB\_CONFIG\_BIAS: DAVISRGB chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.410 DAVISRGB\_CONFIG\_BIAS\_SSN

```
#define DAVISRGB_CONFIG_BIAS_SSN 36
```

Parameter address for module DAVISRGB\_CONFIG\_BIAS: DAVISRGB chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.411 DAVISRGB\_CONFIG\_BIAS\_SSP

```
#define DAVISRGB_CONFIG_BIAS_SSP 35
```

Parameter address for module DAVISRGB\_CONFIG\_BIAS: DAVISRGB chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.1.2.412 DAVISRGB\_CONFIG\_BIAS\_TX2OVG2HI

```
#define DAVISRGB_CONFIG_BIAS_TX2OVG2HI 3
```

Parameter address for module DAVISRGB\_CONFIG\_BIAS: DAVISRGB chip biases. Bias configuration values must be generated using the proper functions, which are:

- [caerBiasVDACGenerate\(\)](#) for VDAC (voltage) biases.
- [caerBiasCoarseFineGenerate\(\)](#) for coarse-fine (current) biases.
- [caerBiasShiftedSourceGenerate\(\)](#) for shifted-source biases. See '<http://inilabs.com/support/biasing/>' for more details.

**4.1.2.413 DAVISRGB\_CONFIG\_CHIP\_ADJUSTOVG1LO**

```
#define DAVISRGB_CONFIG_CHIP_ADJUSTOVG1LO 145
```

Parameter address for module DAVISRGB\_CONFIG\_CHIP: DAVISRGB chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

**4.1.2.414 DAVISRGB\_CONFIG\_CHIP\_ADJUSTOVG2LO**

```
#define DAVISRGB_CONFIG_CHIP_ADJUSTOVG2LO 146
```

Parameter address for module DAVISRGB\_CONFIG\_CHIP: DAVISRGB chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

**4.1.2.415 DAVISRGB\_CONFIG\_CHIP\_ADJUSTTX2OVG2HI**

```
#define DAVISRGB_CONFIG_CHIP_ADJUSTTX2OVG2HI 147
```

Parameter address for module DAVISRGB\_CONFIG\_CHIP: DAVISRGB chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

**4.1.2.416 DAVISRGB\_CONFIG\_CHIP\_AERNAROW**

```
#define DAVISRGB_CONFIG_CHIP_AERNAROW 140
```

Parameter address for module DAVISRGB\_CONFIG\_CHIP: DAVISRGB chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

**4.1.2.417 DAVISRGB\_CONFIG\_CHIP\_ANALOGMUX0**

```
#define DAVISRGB_CONFIG_CHIP_ANALOGMUX0 132
```

Parameter address for module DAVISRGB\_CONFIG\_CHIP: DAVISRGB chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

**4.1.2.418 DAVISRGB\_CONFIG\_CHIP\_ANALOGMUX1**

```
#define DAVISRGB_CONFIG_CHIP_ANALOGMUX1 133
```

Parameter address for module DAVISRGB\_CONFIG\_CHIP: DAVISRGB chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.419 DAVISRGB\_CONFIG\_CHIP\_ANALOGMUX2

```
#define DAVISRGB_CONFIG_CHIP_ANALOGMUX2 134
```

Parameter address for module DAVISRGB\_CONFIG\_CHIP: DAVISRGB chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.420 DAVISRGB\_CONFIG\_CHIP\_BIASMUX0

```
#define DAVISRGB_CONFIG_CHIP_BIASMUX0 135
```

Parameter address for module DAVISRGB\_CONFIG\_CHIP: DAVISRGB chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.421 DAVISRGB\_CONFIG\_CHIP\_DIGITALMUX0

```
#define DAVISRGB_CONFIG_CHIP_DIGITALMUX0 128
```

Parameter address for module DAVISRGB\_CONFIG\_CHIP: DAVISRGB chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.422 DAVISRGB\_CONFIG\_CHIP\_DIGITALMUX1

```
#define DAVISRGB_CONFIG_CHIP_DIGITALMUX1 129
```

Parameter address for module DAVISRGB\_CONFIG\_CHIP: DAVISRGB chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.423 DAVISRGB\_CONFIG\_CHIP\_DIGITALMUX2

```
#define DAVISRGB_CONFIG_CHIP_DIGITALMUX2 130
```

Parameter address for module DAVISRGB\_CONFIG\_CHIP: DAVISRGB chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.424 DAVISRGB\_CONFIG\_CHIP\_DIGITALMUX3

```
#define DAVISRGB_CONFIG_CHIP_DIGITALMUX3 131
```

Parameter address for module DAVISRGB\_CONFIG\_CHIP: DAVISRGB chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

**4.1.2.425 DAVISRGB\_CONFIG\_CHIP\_RESETCALIBNEURON**

```
#define DAVISRGB_CONFIG_CHIP_RESETCALIBNEURON 136
```

Parameter address for module DAVISRGB\_CONFIG\_CHIP: DAVISRGB chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

**4.1.2.426 DAVISRGB\_CONFIG\_CHIP\_RESETESTPIXEL**

```
#define DAVISRGB_CONFIG_CHIP_RESETESTPIXEL 138
```

Parameter address for module DAVISRGB\_CONFIG\_CHIP: DAVISRGB chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

**4.1.2.427 DAVISRGB\_CONFIG\_CHIP\_SELECTGRAYCOUNTER**

```
#define DAVISRGB_CONFIG_CHIP_SELECTGRAYCOUNTER 143
```

Parameter address for module DAVISRGB\_CONFIG\_CHIP: DAVISRGB chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

**4.1.2.428 DAVISRGB\_CONFIG\_CHIP\_TESTADC**

```
#define DAVISRGB_CONFIG_CHIP_TESTADC 144
```

Parameter address for module DAVISRGB\_CONFIG\_CHIP: DAVISRGB chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

**4.1.2.429 DAVISRGB\_CONFIG\_CHIP\_TYPENCALIBNEURON**

```
#define DAVISRGB_CONFIG_CHIP_TYPENCALIBNEURON 137
```

Parameter address for module DAVISRGB\_CONFIG\_CHIP: DAVISRGB chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

**4.1.2.430 DAVISRGB\_CONFIG\_CHIP\_USEAOUT**

```
#define DAVISRGB_CONFIG_CHIP_USEAOUT 141
```

Parameter address for module DAVISRGB\_CONFIG\_CHIP: DAVISRGB chip configuration. These are for expert control and should never be used or changed unless for advanced debugging purposes. To change the Global Shutter configuration, please use DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER instead.

#### 4.1.2.431 IS\_DAVIS128

```
#define IS_DAVIS128(  
    chipID ) ((chipID) == DAVIS_CHIP_DAVIS128)
```

Macros to check a chip identifier integer against the known chip types. Returns true if a chip identifier matches, false otherwise.

#### 4.1.2.432 IS\_DAVIS208

```
#define IS_DAVIS208(  
    chipID ) ((chipID) == DAVIS_CHIP_DAVIS208)
```

Macros to check a chip identifier integer against the known chip types. Returns true if a chip identifier matches, false otherwise.

#### 4.1.2.433 IS\_DAVIS240

```
#define IS_DAVIS240(  
    chipID ) (IS_DAVIS240A(chipID) || IS_DAVIS240B(chipID) || IS_DAVIS240C(chipID))
```

Macros to check a chip identifier integer against the known chip types. Returns true if a chip identifier matches, false otherwise.

#### 4.1.2.434 IS\_DAVIS240A

```
#define IS_DAVIS240A(  
    chipID ) ((chipID) == DAVIS_CHIP_DAVIS240A)
```

Macros to check a chip identifier integer against the known chip types. Returns true if a chip identifier matches, false otherwise.

#### 4.1.2.435 IS\_DAVIS240B

```
#define IS_DAVIS240B(  
    chipID ) ((chipID) == DAVIS_CHIP_DAVIS240B)
```

Macros to check a chip identifier integer against the known chip types. Returns true if a chip identifier matches, false otherwise.

#### 4.1.2.436 IS\_DAVIS240C

```
#define IS_DAVIS240C(  
    chipID ) ((chipID) == DAVIS_CHIP_DAVIS240C)
```

Macros to check a chip identifier integer against the known chip types. Returns true if a chip identifier matches, false otherwise.

#### 4.1.2.437 IS\_DAVIS346

```
#define IS_DAVIS346(  
    chipID ) (IS_DAVIS346A(chipID) || IS_DAVIS346B(chipID) || IS_DAVIS346C(chipID))
```

Macros to check a chip identifier integer against the known chip types. Returns true if a chip identifier matches, false otherwise.

#### 4.1.2.438 IS\_DAVIS346A

```
#define IS_DAVIS346A(  
    chipID ) ((chipID) == DAVIS_CHIP_DAVIS346A)
```

Macros to check a chip identifier integer against the known chip types. Returns true if a chip identifier matches, false otherwise.

#### 4.1.2.439 IS\_DAVIS346B

```
#define IS_DAVIS346B(  
    chipID ) ((chipID) == DAVIS_CHIP_DAVIS346B)
```

Macros to check a chip identifier integer against the known chip types. Returns true if a chip identifier matches, false otherwise.

#### 4.1.2.440 IS\_DAVIS346C

```
#define IS_DAVIS346C(  
    chipID ) ((chipID) == DAVIS_CHIP_DAVIS346C)
```

Macros to check a chip identifier integer against the known chip types. Returns true if a chip identifier matches, false otherwise.

#### 4.1.2.441 IS\_DAVIS640

```
#define IS_DAVIS640(  
    chipID ) ((chipID) == DAVIS_CHIP_DAVIS640)
```

Macros to check a chip identifier integer against the known chip types. Returns true if a chip identifier matches, false otherwise.

#### 4.1.2.442 IS\_DAVISRGB

```
#define IS_DAVISRGB(  
    chipID ) ((chipID) == DAVIS_CHIP_DAVISRGB)
```

Macros to check a chip identifier integer against the known chip types. Returns true if a chip identifier matches, false otherwise.

### 4.1.3 Enumeration Type Documentation

#### 4.1.3.1 caer\_bias\_shiftedsourcesource\_operating\_mode

```
enum caer_bias_shiftedsourcesource_operating_mode
```

Shifted-source bias operating mode.

## Enumerator

SHIFTED_SOURCE	Standard mode.
HI_Z	High impedance (driven from outside).
TIED_TO_RAIL	Tied to ground (SSN) or VDD (SSP).

## 4.1.3.2 caer\_bias\_shiftedsource\_voltage\_level

```
enum caer_bias_shiftedsource_voltage_level
```

Shifted-source bias voltage level.

## Enumerator

SPLIT_GATE	Standard mode (200-400mV).
SINGLE_DIODE	Higher shifted-source voltage (one cascode).
DOUBLE_DIODE	Even higher shifted-source voltage (two cascodes).

## 4.1.4 Function Documentation

## 4.1.4.1 caerBiasCoarseFineGenerate()

```
uint16_t caerBiasCoarseFineGenerate (
    const struct caer_bias_coarsefine coarseFineBias )
```

Transform coarse-fine bias structure into internal integer representation, suited for sending directly to the device via [caerDeviceConfigSet\(\)](#).

## Parameters

<i>coarseFineBias</i>	coarse-fine bias structure.
-----------------------	-----------------------------

## Returns

internal integer representation for device configuration.

## 4.1.4.2 caerBiasCoarseFineParse()

```
struct caer_bias_coarsefine caerBiasCoarseFineParse (
    const uint16_t coarseFineBias )
```



Transform internal integer representation, as received by calls to [caerDeviceConfigGet\(\)](#), into a coarse-fine bias structure, for easier handling and understanding of the various parameters.

**Parameters**

<i>coarseFineBias</i>	internal integer representation from device.
-----------------------	--

**Returns**

coarse-fine bias structure.

**4.1.4.3 caerBiasShiftedSourceGenerate()**

```
uint16_t caerBiasShiftedSourceGenerate (
    const struct caer_bias_shiftedsource shiftedSourceBias )
```

Transform shifted-source bias structure into internal integer representation, suited for sending directly to the device via [caerDeviceConfigSet\(\)](#).

**Parameters**

<i>shiftedSourceBias</i>	shifted-source bias structure.
--------------------------	--------------------------------

**Returns**

internal integer representation for device configuration.

**4.1.4.4 caerBiasShiftedSourceParse()**

```
struct caer_bias_shiftedsource caerBiasShiftedSourceParse (
    const uint16_t shiftedSourceBias )
```

Transform internal integer representation, as received by calls to [caerDeviceConfigGet\(\)](#), into a shifted-source bias structure, for easier handling and understanding of the various parameters.

**Parameters**

<i>shiftedSourceBias</i>	internal integer representation from device.
--------------------------	--

**Returns**

shifted-source bias structure.

#### 4.1.4.5 caerBiasVDACGenerate()

```
uint16_t caerBiasVDACGenerate (
    const struct caer_bias_vdac vdacBias )
```

Transform VDAC bias structure into internal integer representation, suited for sending directly to the device via [caerDeviceConfigSet\(\)](#).

##### Parameters

<i>vdacBias</i>	VDAC bias structure.
-----------------	----------------------

##### Returns

internal integer representation for device configuration.

#### 4.1.4.6 caerBiasVDACParse()

```
struct caer_bias_vdac caerBiasVDACParse (
    const uint16_t vdacBias )
```

Transform internal integer representation, as received by calls to [caerDeviceConfigGet\(\)](#), into a VDAC bias structure, for easier handling and understanding of the various parameters.

##### Parameters

<i>vdacBias</i>	internal integer representation from device.
-----------------	--

##### Returns

VDAC bias structure.

#### 4.1.4.7 caerDavisInfoGet()

```
struct caer_davis_info caerDavisInfoGet (
    caerDeviceHandle handle )
```

Return basic information on the device, such as its ID, its resolution, the logic version, and so on. See the 'struct [caer\\_davis\\_info](#)' documentation for more details.

##### Parameters

<i>handle</i>	a valid device handle.
---------------	------------------------

**Returns**

a copy of the device information structure if successful, an empty structure (all zeros) on failure.

**4.2 devices/device.h File Reference**

```
#include "../libcaer.h"
#include "../events/packetContainer.h"
```

**Macros**

- `#define CAER_HOST_CONFIG_DATAEXCHANGE -2`
- `#define CAER_HOST_CONFIG_PACKETS -3`
- `#define CAER_HOST_CONFIG_LOG -4`
- `#define CAER_HOST_CONFIG_DATAEXCHANGE_BUFFER_SIZE 0`
- `#define CAER_HOST_CONFIG_DATAEXCHANGE_BLOCKING 1`
- `#define CAER_HOST_CONFIG_DATAEXCHANGE_START_PRODUCERS 2`
- `#define CAER_HOST_CONFIG_DATAEXCHANGE_STOP_PRODUCERS 3`
- `#define CAER_HOST_CONFIG_PACKETS_MAX_CONTAINER_PACKET_SIZE 0`
- `#define CAER_HOST_CONFIG_PACKETS_MAX_CONTAINER_INTERVAL 1`
- `#define CAER_HOST_CONFIG_LOG_LEVEL 0`

**Typedefs**

- `typedef struct caer_device_handle * caerDeviceHandle`

**Functions**

- `bool caerDeviceClose (caerDeviceHandle *handle)`
- `bool caerDeviceSendDefaultConfig (caerDeviceHandle handle)`
- `bool caerDeviceConfigSet (caerDeviceHandle handle, int8_t modAddr, uint8_t paramAddr, uint32_t param)`
- `bool caerDeviceConfigGet (caerDeviceHandle handle, int8_t modAddr, uint8_t paramAddr, uint32_t *param)`
- `bool caerDeviceDataStart (caerDeviceHandle handle, void(*dataNotifyIncrease)(void *ptr), void(*dataNotifyDecrease)(void *ptr), void *dataNotifyUserPtr, void(*dataShutdownNotify)(void *ptr), void *dataShutdownUserPtr)`
- `bool caerDeviceDataStop (caerDeviceHandle handle)`
- `caerEventPacketContainer caerDeviceDataGet (caerDeviceHandle handle)`

**4.2.1 Detailed Description**

Common functions to access, configure and exchange data with supported devices. Also contains defines for host related configuration options.

**4.2.2 Macro Definition Documentation**

#### 4.2.2.1 CAER\_HOST\_CONFIG\_DATAEXCHANGE

```
#define CAER_HOST_CONFIG_DATAEXCHANGE -2
```

Module address: host-side data exchange (ring-buffer) configuration.

#### 4.2.2.2 CAER\_HOST\_CONFIG\_DATAEXCHANGE\_BLOCKING

```
#define CAER_HOST_CONFIG_DATAEXCHANGE_BLOCKING 1
```

Parameter address for module CAER\_HOST\_CONFIG\_DATAEXCHANGE: when calling [caerDeviceDataGet\(\)](#), the function can either be blocking, meaning it waits until it has a valid EventPacketContainer to return, or not, meaning it returns right away. This behavior can be set with this flag. Please see the [caerDeviceDataGet\(\)](#) documentation for more information on its return values.

#### 4.2.2.3 CAER\_HOST\_CONFIG\_DATAEXCHANGE\_BUFFER\_SIZE

```
#define CAER_HOST_CONFIG_DATAEXCHANGE_BUFFER_SIZE 0
```

Parameter address for module CAER\_HOST\_CONFIG\_DATAEXCHANGE: set size of elements that can be held by the thread-safe FIFO buffer between the data transfer thread and the main thread. The default values are usually fine, only change them if you're running into lots of dropped/missing packets; you can turn on the INFO log level to see when this is the case.

#### 4.2.2.4 CAER\_HOST\_CONFIG\_DATAEXCHANGE\_START\_PRODUCERS

```
#define CAER_HOST_CONFIG_DATAEXCHANGE_START_PRODUCERS 2
```

Parameter address for module CAER\_HOST\_CONFIG\_DATAEXCHANGE: whether to start all the data producer modules on the device (DVS, APS, Mux, ...) automatically when starting the data transfer thread with [caerDeviceDataStart\(\)](#) or not. If disabled, be aware you will have to start the right modules manually, which can be useful if you need precise control over which ones are running at any time.

#### 4.2.2.5 CAER\_HOST\_CONFIG\_DATAEXCHANGE\_STOP\_PRODUCERS

```
#define CAER_HOST_CONFIG_DATAEXCHANGE_STOP_PRODUCERS 3
```

Parameter address for module CAER\_HOST\_CONFIG\_DATAEXCHANGE: whether to stop all the data producer modules on the device (DVS, APS, Mux, ...) automatically when stopping the data transfer thread with [caerDeviceDataStop\(\)](#) or not. If disabled, be aware you will have to stop the right modules manually, to halt the data flow, which can be useful if you need precise control over which ones are running at any time.

#### 4.2.2.6 CAER\_HOST\_CONFIG\_LOG

```
#define CAER_HOST_CONFIG_LOG -4
```

Module address: host-side logging configuration.

#### 4.2.2.7 CAER\_HOST\_CONFIG\_LOG\_LEVEL

```
#define CAER_HOST_CONFIG_LOG_LEVEL 0
```

Parameter address for module CAER\_HOST\_CONFIG\_LOG: set the log-level for this device, to be used when logging messages. Defaults to the value of the global log-level when the device was first opened.

#### 4.2.2.8 CAER\_HOST\_CONFIG\_PACKETS

```
#define CAER_HOST_CONFIG_PACKETS -3
```

Module address: host-side event packets generation configuration.

#### 4.2.2.9 CAER\_HOST\_CONFIG\_PACKETS\_MAX\_CONTAINER\_INTERVAL

```
#define CAER_HOST_CONFIG_PACKETS_MAX_CONTAINER_INTERVAL 1
```

Parameter address for module CAER\_HOST\_CONFIG\_PACKETS: set the time interval between subsequent packet containers. Must be at least 1 microsecond. The value is in microseconds, and is checked across all types of events contained in the EventPacketContainer.

#### 4.2.2.10 CAER\_HOST\_CONFIG\_PACKETS\_MAX\_CONTAINER\_PACKET\_SIZE

```
#define CAER_HOST_CONFIG_PACKETS_MAX_CONTAINER_PACKET_SIZE 0
```

Parameter address for module CAER\_HOST\_CONFIG\_PACKETS: set the maximum number of events any of a packet container's packets may hold before it's made available to the user. Set to zero to disable. This is checked for each number of events held in each typed EventPacket that is a part of the EventPacketContainer.

### 4.2.3 Typedef Documentation

#### 4.2.3.1 caerDeviceHandle

```
typedef struct caer_device_handle* caerDeviceHandle
```

Pointer to an open device on which to operate.

### 4.2.4 Function Documentation

#### 4.2.4.1 caerDeviceClose()

```
bool caerDeviceClose (
    caerDeviceHandle * handle )
```

Close a previously opened device and invalidate its handle.

## Parameters

<i>handle</i>	pointer to a valid device handle. Will set handle to NULL if closing is successful, to prevent further usage of this handle for other operations.
---------------	---

## Returns

true if closing was successful, false on errors.

## 4.2.4.2 caerDeviceConfigGet()

```
bool caerDeviceConfigGet (
    caerDeviceHandle handle,
    int8_t modAddr,
    uint8_t paramAddr,
    uint32_t * param )
```

Get the value of a configuration parameter.

## Parameters

<i>handle</i>	a valid device handle.
<i>modAddr</i>	a module address, used to specify which configuration module one wants to query. Negative addresses are used for host-side configuration, while positive addresses (including zero) are used for device-side configuration.
<i>paramAddr</i>	a parameter address, to select a specific parameter to query from this particular configuration module. Only positive numbers (including zero) are allowed.
<i>param</i>	a pointer to an integer, in which to store the configuration parameter's current value. The integer will always be either set to zero (on failure), or to the current value (on success).

## Returns

true if getting the configuration was successful, false on errors.

## 4.2.4.3 caerDeviceConfigSet()

```
bool caerDeviceConfigSet (
    caerDeviceHandle handle,
    int8_t modAddr,
    uint8_t paramAddr,
    uint32_t param )
```

Set a configuration parameter to a given value.

## Parameters

<i>handle</i>	a valid device handle.
<i>modAddr</i>	a module address, used to specify which configuration module one wants to update. Negative addresses are used for host-side configuration, while positive addresses (including zero) are used for device-side configuration.
<i>paramAddr</i>	a parameter address, to select a specific parameter to update from this particular configuration module. Only positive numbers (including zero) are allowed.
<i>param</i>	a configuration parameter's new value.

## Returns

true if sending the configuration was successful, false on errors.

## 4.2.4.4 caerDeviceDataGet()

```
caerEventPacketContainer caerDeviceDataGet (
    caerDeviceHandle handle )
```

Get an event packet container, which contains events of various types generated by the device, for further processing. The returned data structures are allocated in memory and will need to be freed. The [caerEventPacketContainerFree\(\)](#) function can be used to correctly free the full container memory. For single caerEventPackets, just use free(). This function can be made blocking with the CAER\_HOST\_CONFIG\_DATAEXCHANGE\_BLOCKING configuration parameter. By default it is non-blocking.

## Parameters

<i>handle</i>	a valid device handle.
---------------	------------------------

## Returns

a valid event packet container. NULL will be returned on errors, such as exceptional device shutdown, or when there is no container available in non-blocking mode. Always check this return value!

## 4.2.4.5 caerDeviceDataStart()

```
bool caerDeviceDataStart (
    caerDeviceHandle handle,
    void(*) (void *ptr) dataNotifyIncrease,
    void(*) (void *ptr) dataNotifyDecrease,
    void * dataNotifyUserPtr,
    void(*) (void *ptr) dataShutdownNotify,
    void * dataShutdownUserPtr )
```

Start getting data from the device, setting up the data transfers and starting the data producers (see [CAER\\_HOST\\_CONFIG\\_DATAEXCHANGE\\_START\\_PRODUCERS](#)). Supports notification of new data and exceptional shutdown events via user-defined call-backs.



## Parameters

<i>handle</i>	a valid device handle.
<i>dataNotifyIncrease</i>	function pointer, called every time a new piece of data available and has been put in the FIFO buffer for consumption. <code>dataNotifyUserPtr</code> will be passed as parameter to the function.
<i>dataNotifyDecrease</i>	function pointer, called every time a new piece of data has been consumed from the FIFO buffer inside <code>caerDeviceDataGet()</code> . <code>dataNotifyUserPtr</code> will be passed as parameter to the function.
<i>dataNotifyUserPtr</i>	pointer that will be passed to the <code>dataNotifyIncrease</code> and <code>dataNotifyDecrease</code> functions. Can be NULL.
<i>dataShutdownNotify</i>	function pointer, called on exceptional shut-down of the data transfers. This is used to detect exceptional shut-downs that do not come from calling <code>caerDeviceDataStop()</code> , such as when the device is disconnected or all data transfers fail.
<i>dataShutdownUserPtr</i>	pointer that will be passed to the <code>dataShutdownNotify</code> function. Can be NULL.

## Returns

true if starting the data transfer was successful, false on errors.

## 4.2.4.6 caerDeviceDataStop()

```
bool caerDeviceDataStop (
    caerDeviceHandle handle )
```

Stop getting data from the device, shutting down the data transfers and stopping the data producers (see `CAER_HOST_CONFIG_DATAEXCHANGE_STOP_PRODUCERS`). This normal shut-down will not generate a notification (see `caerDeviceDataStart()`).

## Parameters

<i>handle</i>	a valid device handle.
---------------	------------------------

## Returns

true if stopping the data transfer was successful, false on errors.

## 4.2.4.7 caerDeviceSendDefaultConfig()

```
bool caerDeviceSendDefaultConfig (
    caerDeviceHandle handle )
```

Send a set of good default configuration settings to the device. This avoids users having to set every configuration option each time, especially when wanting to get going quickly or just needing to change a few settings to get to the desired operating mode.

**Parameters**

<i>handle</i>	a valid device handle.
---------------	------------------------

**Returns**

true if sending the configuration was successful, false on errors.

**4.3 devices/dvs128.h File Reference**

```
#include "usb.h"
#include "../events/polarity.h"
#include "../events/special.h"
```

**Data Structures**

- struct [caer\\_dvs128\\_info](#)

**Macros**

- #define [CAER\\_DEVICE\\_DVS128](#) 0
- #define [DVS128\\_CONFIG\\_DVS](#) 0
- #define [DVS128\\_CONFIG\\_BIAS](#) 1
- #define [DVS128\\_CONFIG\\_DVS\\_RUN](#) 0
- #define [DVS128\\_CONFIG\\_DVS\\_TIMESTAMP\\_RESET](#) 1
- #define [DVS128\\_CONFIG\\_DVS\\_ARRAY\\_RESET](#) 2
- #define [DVS128\\_CONFIG\\_DVS\\_TS\\_MASTER](#) 3
- #define [DVS128\\_CONFIG\\_BIAS\\_CAS](#) 0
- #define [DVS128\\_CONFIG\\_BIAS\\_INJGND](#) 1
- #define [DVS128\\_CONFIG\\_BIAS\\_REQPD](#) 2
- #define [DVS128\\_CONFIG\\_BIAS\\_PUX](#) 3
- #define [DVS128\\_CONFIG\\_BIAS\\_DIFFOFF](#) 4
- #define [DVS128\\_CONFIG\\_BIAS\\_REQ](#) 5
- #define [DVS128\\_CONFIG\\_BIAS\\_REFR](#) 6
- #define [DVS128\\_CONFIG\\_BIAS\\_PUY](#) 7
- #define [DVS128\\_CONFIG\\_BIAS\\_DIFFON](#) 8
- #define [DVS128\\_CONFIG\\_BIAS\\_DIFF](#) 9
- #define [DVS128\\_CONFIG\\_BIAS\\_FOLL](#) 10
- #define [DVS128\\_CONFIG\\_BIAS\\_PR](#) 11

**Functions**

- struct [caer\\_dvs128\\_info](#) [caerDVS128InfoGet](#) ([caerDeviceHandle](#) handle)

**4.3.1 Detailed Description**

DVS128 specific configuration defines and information structures.

## 4.3.2 Macro Definition Documentation

### 4.3.2.1 CAER\_DEVICE\_DVS128

```
#define CAER_DEVICE_DVS128 0
```

Device type definition for iniLabs DVS128.

### 4.3.2.2 DVS128\_CONFIG\_BIAS

```
#define DVS128_CONFIG_BIAS 1
```

Module address: device-side chip bias generator configuration.

### 4.3.2.3 DVS128\_CONFIG\_BIAS\_CAS

```
#define DVS128_CONFIG_BIAS_CAS 0
```

Parameter address for module DVS128\_CONFIG\_BIAS: First stage amplifier cascode bias. See '<http://inilabs.com/support/biasing/>' for more details.

### 4.3.2.4 DVS128\_CONFIG\_BIAS\_DIFF

```
#define DVS128_CONFIG_BIAS_DIFF 9
```

Parameter address for module DVS128\_CONFIG\_BIAS: Differential (second stage amplifier) bias. See '<http://inilabs.com/support/biasing/>' for more details.

### 4.3.2.5 DVS128\_CONFIG\_BIAS\_DIFFOFF

```
#define DVS128_CONFIG_BIAS_DIFFOFF 4
```

Parameter address for module DVS128\_CONFIG\_BIAS: Off events threshold bias. See '<http://inilabs.com/support/biasing/>' for more details.

### 4.3.2.6 DVS128\_CONFIG\_BIAS\_DIFFON

```
#define DVS128_CONFIG_BIAS_DIFFON 8
```

Parameter address for module DVS128\_CONFIG\_BIAS: On events threshold bias. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.3.2.7 DVS128\_CONFIG\_BIAS\_FOLL

```
#define DVS128_CONFIG_BIAS_FOLL 10
```

Parameter address for module DVS128\_CONFIG\_BIAS: Source follower bias. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.3.2.8 DVS128\_CONFIG\_BIAS\_INJGND

```
#define DVS128_CONFIG_BIAS_INJGND 1
```

Parameter address for module DVS128\_CONFIG\_BIAS: Injected ground bias. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.3.2.9 DVS128\_CONFIG\_BIAS\_PR

```
#define DVS128_CONFIG_BIAS_PR 11
```

Parameter address for module DVS128\_CONFIG\_BIAS: Photoreceptor bias. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.3.2.10 DVS128\_CONFIG\_BIAS\_PUX

```
#define DVS128_CONFIG_BIAS_PUX 3
```

Parameter address for module DVS128\_CONFIG\_BIAS: Pull up on request from X arbiter (AER). See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.3.2.11 DVS128\_CONFIG\_BIAS\_PUY

```
#define DVS128_CONFIG_BIAS_PUY 7
```

Parameter address for module DVS128\_CONFIG\_BIAS: Pull up on request from Y arbiter (AER). See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.3.2.12 DVS128\_CONFIG\_BIAS\_REFR

```
#define DVS128_CONFIG_BIAS_REFR 6
```

Parameter address for module DVS128\_CONFIG\_BIAS: Refractory period bias. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.3.2.13 DVS128\_CONFIG\_BIAS\_REQ

```
#define DVS128_CONFIG_BIAS_REQ 5
```

Parameter address for module DVS128\_CONFIG\_BIAS: Pull down for passive load inverters in digital AER pixel circuitry. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.3.2.14 DVS128\_CONFIG\_BIAS\_REQPD

```
#define DVS128_CONFIG_BIAS_REQPD 2
```

Parameter address for module DVS128\_CONFIG\_BIAS: Pull down on chip request (AER). See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.3.2.15 DVS128\_CONFIG\_DVS

```
#define DVS128_CONFIG_DVS 0
```

Module address: device-side DVS configuration.

#### 4.3.2.16 DVS128\_CONFIG\_DVS\_ARRAY\_RESET

```
#define DVS128_CONFIG_DVS_ARRAY_RESET 2
```

Parameter address for module DVS128\_CONFIG\_DVS: reset the whole DVS pixel array. This is a temporary configuration switch and will reset itself right away.

#### 4.3.2.17 DVS128\_CONFIG\_DVS\_RUN

```
#define DVS128_CONFIG_DVS_RUN 0
```

Parameter address for module DVS128\_CONFIG\_DVS: run the DVS chip and generate polarity event data.

#### 4.3.2.18 DVS128\_CONFIG\_DVS\_TIMESTAMP\_RESET

```
#define DVS128_CONFIG_DVS_TIMESTAMP_RESET 1
```

Parameter address for module DVS128\_CONFIG\_DVS: reset the time-stamp counter of the device. This is a temporary configuration switch and will reset itself right away.

#### 4.3.2.19 DVS128\_CONFIG\_DVS\_TS\_MASTER

```
#define DVS128_CONFIG_DVS_TS_MASTER 3
```

Parameter address for module DVS128\_CONFIG\_DVS: control if this DVS is a timestamp master device. Default is enabled.

### 4.3.3 Function Documentation

#### 4.3.3.1 caerDVS128InfoGet()

```
struct caer_dvs128_info caerDVS128InfoGet (  
    caerDeviceHandle handle )
```

Return basic information on the device, such as its ID, its resolution, the logic version, and so on. See the 'struct caer\_dvs128\_info' documentation for more details.

### Parameters

<i>handle</i>	a valid device handle.
---------------	------------------------

### Returns

a copy of the device information structure if successful, an empty structure (all zeros) on failure.

## 4.4 devices/dynapse.h File Reference

```
#include "usb.h"
#include "../events/spike.h"
#include "../events/special.h"
```

### Data Structures

- struct [caer\\_dynapse\\_info](#)
- struct [caer\\_bias\\_dynapse](#)

### Macros

- #define [CAER\\_DEVICE\\_DYNAPSE](#) 3
- #define [DYNAPSE\\_CHIP\\_DYNAPSE](#) 64
- #define [DYNAPSE\\_CONFIG\\_MUX](#) 0
- #define [DYNAPSE\\_CONFIG\\_AER](#) 1
- #define [DYNAPSE\\_CONFIG\\_CHIP](#) 5
- #define [DYNAPSE\\_CONFIG\\_SYSINFO](#) 6
- #define [DYNAPSE\\_CONFIG\\_USB](#) 9
- #define [DYNAPSE\\_CONFIG\\_CLEAR\\_CAM](#) 10
- #define [DYNAPSE\\_CONFIG\\_DEFAULT\\_SRAM](#) 11
- #define [DYNAPSE\\_CONFIG\\_MONITOR\\_NEU](#) 12
- #define [DYNAPSE\\_CONFIG\\_DEFAULT\\_SRAM\\_EMPTY](#) 13
- #define [DYNAPSE\\_CONFIG\\_SRAM](#) 14
- #define [DYNAPSE\\_CONFIG\\_SYNAPSERECONFIG](#) 15
- #define [DYNAPSE\\_CONFIG\\_SPIKEGEN](#) 16
- #define [DYNAPSE\\_CONFIG\\_POISSONSPIKEGEN](#) 18
- #define [DYNAPSE\\_CONFIG\\_POISSONSPIKEGEN\\_RUN](#) 0
- #define [DYNAPSE\\_CONFIG\\_POISSONSPIKEGEN\\_WRITEADDRESS](#) 1
- #define [DYNAPSE\\_CONFIG\\_POISSONSPIKEGEN\\_WRITEDATA](#) 2
- #define [DYNAPSE\\_CONFIG\\_POISSONSPIKEGEN\\_CHIPID](#) 3
- #define [DYNAPSE\\_CONFIG\\_SPIKEGEN\\_RUN](#) 0
- #define [DYNAPSE\\_CONFIG\\_SPIKEGEN\\_VARMODE](#) 1
- #define [DYNAPSE\\_CONFIG\\_SPIKEGEN\\_BASEADDR](#) 2
- #define [DYNAPSE\\_CONFIG\\_SPIKEGEN\\_STIMCOUNT](#) 3
- #define [DYNAPSE\\_CONFIG\\_SPIKEGEN\\_ISI](#) 4
- #define [DYNAPSE\\_CONFIG\\_SPIKEGEN\\_ISIBASE](#) 5
- #define [DYNAPSE\\_CONFIG\\_SPIKEGEN\\_REPEAT](#) 6
- #define [DYNAPSE\\_CONFIG\\_SYNAPSERECONFIG\\_RUN](#) 0
- #define [DYNAPSE\\_CONFIG\\_SYNAPSERECONFIG\\_GLOBALKERNEL](#) 1

- #define DYNAPSE\_CONFIG\_SYNAPSERECONFIG\_USESRAMKERNELS 2
- #define DYNAPSE\_CONFIG\_SYNAPSERECONFIG\_CHIPSELECT 3
- #define DYNAPSE\_CONFIG\_SYNAPSERECONFIG\_SRAMBASEADDR 4
- #define DYNAPSE\_CONFIG\_SRAM\_ADDRESS 1
- #define DYNAPSE\_CONFIG\_SRAM\_READDATA 2
- #define DYNAPSE\_CONFIG\_SRAM\_WRITEDATA 3
- #define DYNAPSE\_CONFIG\_SRAM\_RWCOMMAND 4
- #define DYNAPSE\_CONFIG\_SRAM\_READ 0
- #define DYNAPSE\_CONFIG\_SRAM\_WRITE 1
- #define DYNAPSE\_CONFIG\_SRAM\_BURSTMODE 5
- #define DYNAPSE\_CONFIG\_MUX\_RUN 0
- #define DYNAPSE\_CONFIG\_MUX\_TIMESTAMP\_RUN 1
- #define DYNAPSE\_CONFIG\_MUX\_TIMESTAMP\_RESET 2
- #define DYNAPSE\_CONFIG\_MUX\_FORCE\_CHIP\_BIAS\_ENABLE 3
- #define DYNAPSE\_CONFIG\_MUX\_DROP\_AER\_ON\_TRANSFER\_STALL 4
- #define DYNAPSE\_CONFIG\_AER\_RUN 3
- #define DYNAPSE\_CONFIG\_AER\_ACK\_DELAY 4
- #define DYNAPSE\_CONFIG\_AER\_ACK\_EXTENSION 6
- #define DYNAPSE\_CONFIG\_AER\_WAIT\_ON\_TRANSFER\_STALL 8
- #define DYNAPSE\_CONFIG\_AER\_EXTERNAL\_AER\_CONTROL 10
- #define DYNAPSE\_CONFIG\_CHIP\_RUN 0
- #define DYNAPSE\_CONFIG\_CHIP\_ID 1
- #define DYNAPSE\_CONFIG\_CHIP\_CONTENT 2
- #define DYNAPSE\_CONFIG\_CHIP\_REQ\_DELAY 3
- #define DYNAPSE\_CONFIG\_CHIP\_REQ\_EXTENSION 4
- #define DYNAPSE\_CONFIG\_SYSINFO\_LOGIC\_VERSION 0
- #define DYNAPSE\_CONFIG\_SYSINFO\_CHIP\_IDENTIFIER 1
- #define DYNAPSE\_CONFIG\_SYSINFO\_DEVICE\_IS\_MASTER 2
- #define DYNAPSE\_CONFIG\_SYSINFO\_LOGIC\_CLOCK 3
- #define DYNAPSE\_CONFIG\_USB\_RUN 0
- #define DYNAPSE\_CONFIG\_USB\_EARLY\_PACKET\_DELAY 1
- #define DYNAPSE\_CONFIG\_SRAM\_DIRECTION\_POS 0
- #define DYNAPSE\_CONFIG\_SRAM\_DIRECTION\_NEG 1
- #define DYNAPSE\_CONFIG\_SRAM\_DIRECTION\_Y\_NORTH 0
- #define DYNAPSE\_CONFIG\_SRAM\_DIRECTION\_Y\_SOUTH 1
- #define DYNAPSE\_CONFIG\_SRAM\_DIRECTION\_X\_EAST 0
- #define DYNAPSE\_CONFIG\_SRAM\_DIRECTION\_X\_WEST 1
- #define DYNAPSE\_X4BOARD\_NUMCHIPS 4
- #define DYNAPSE\_X4BOARD\_NEUX 64
- #define DYNAPSE\_X4BOARD\_NEUY 64
- #define DYNAPSE\_X4BOARD\_COREX 4
- #define DYNAPSE\_X4BOARD\_COREY 4
- #define DYNAPSE\_CONFIG\_DYNAPSE\_U0 0  
*Chip 0 ID.*
- #define DYNAPSE\_CONFIG\_DYNAPSE\_U1 1  
*Chip 1 ID.*
- #define DYNAPSE\_CONFIG\_DYNAPSE\_U2 2  
*Chip 2 ID.*
- #define DYNAPSE\_CONFIG\_DYNAPSE\_U3 3  
*Chip 3 ID.*
- #define DYNAPSE\_CONFIG\_NUMCORES 4  
*Number of cores per chip.*
- #define DYNAPSE\_CONFIG\_NUMNEURONS 1024  
*Number of neurons in single chip.*

- #define `DYNAPSE_CONFIG_NUMNEURONS_CORE` 256  
*Number of neurons per core.*
- #define `DYNAPSE_CONFIG_XCHIPSIZE` 32  
*Number of columns of neurons in a chip.*
- #define `DYNAPSE_CONFIG_YCHIPSIZE` 32  
*Number of rows of neurons in a core.*
- #define `DYNAPSE_CONFIG_NEUCOL` 16  
*Number of columns of neurons in a core.*
- #define `DYNAPSE_CONFIG_NEUROW` 16  
*Number of rows of neurons in a core.*
- #define `DYNAPSE_CONFIG_CAMCOL` 16  
*Number of columns of CAMs in a core.*
- #define `DYNAPSE_CONFIG_NUMCAM_NEU` 64  
*Number of CAMs per neuron.*
- #define `DYNAPSE_CONFIG_NUMSRAM_NEU` 4  
*Number of SRAM cells per neuron.*
- #define `DYNAPSE_CONFIG_CAMTYPE_F_EXC` 3  
*Fast excitatory synapse.*
- #define `DYNAPSE_CONFIG_CAMTYPE_S_EXC` 2  
*Slow excitatory synapse.*
- #define `DYNAPSE_CONFIG_CAMTYPE_F_INH` 1  
*Fast inhibitory synapse.*
- #define `DYNAPSE_CONFIG_CAMTYPE_S_INH` 0  
*Slow inhibitory synapse.*
- #define `DYNAPSE_CONFIG_BIAS_C0_PULSE_PWLK_P` 0
- #define `DYNAPSE_CONFIG_BIAS_C0_PS_WEIGHT_INH_S_N` 2
- #define `DYNAPSE_CONFIG_BIAS_C0_PS_WEIGHT_INH_F_N` 4
- #define `DYNAPSE_CONFIG_BIAS_C0_PS_WEIGHT_EXC_S_N` 6
- #define `DYNAPSE_CONFIG_BIAS_C0_PS_WEIGHT_EXC_F_N` 8
- #define `DYNAPSE_CONFIG_BIAS_C0_IF_RFR_N` 10
- #define `DYNAPSE_CONFIG_BIAS_C0_IF_TAU1_N` 12
- #define `DYNAPSE_CONFIG_BIAS_C0_IF_AHTAU_N` 14
- #define `DYNAPSE_CONFIG_BIAS_C0_IF_CASC_N` 16
- #define `DYNAPSE_CONFIG_BIAS_C0_IF_TAU2_N` 18
- #define `DYNAPSE_CONFIG_BIAS_C0_IF_BUF_P` 20
- #define `DYNAPSE_CONFIG_BIAS_C0_IF_AHTHR_N` 22
- #define `DYNAPSE_CONFIG_BIAS_C0_IF_THR_N` 24
- #define `DYNAPSE_CONFIG_BIAS_C0_NPDPIE_THR_S_P` 26
- #define `DYNAPSE_CONFIG_BIAS_C0_NPDPIE_THR_F_P` 28
- #define `DYNAPSE_CONFIG_BIAS_C0_NPDPII_THR_F_P` 30
- #define `DYNAPSE_CONFIG_BIAS_C0_NPDPII_THR_S_P` 32
- #define `DYNAPSE_CONFIG_BIAS_C0_IF_NMDA_N` 34
- #define `DYNAPSE_CONFIG_BIAS_C0_IF_DC_P` 36
- #define `DYNAPSE_CONFIG_BIAS_C0_IF_AHW_P` 38
- #define `DYNAPSE_CONFIG_BIAS_C0_NPDPII_TAU_S_P` 40
- #define `DYNAPSE_CONFIG_BIAS_C0_NPDPII_TAU_F_P` 42
- #define `DYNAPSE_CONFIG_BIAS_C0_NPDPIE_TAU_F_P` 44
- #define `DYNAPSE_CONFIG_BIAS_C0_NPDPIE_TAU_S_P` 46
- #define `DYNAPSE_CONFIG_BIAS_C0_R2R_P` 48
- #define `DYNAPSE_CONFIG_BIAS_C1_PULSE_PWLK_P` 1
- #define `DYNAPSE_CONFIG_BIAS_C1_PS_WEIGHT_INH_S_N` 3
- #define `DYNAPSE_CONFIG_BIAS_C1_PS_WEIGHT_INH_F_N` 5
- #define `DYNAPSE_CONFIG_BIAS_C1_PS_WEIGHT_EXC_S_N` 7



- `#define DYNAPSE_CONFIG_BIAS_C1_PS_WEIGHT_EXC_F_N 9`
- `#define DYNAPSE_CONFIG_BIAS_C1_IF_RFR_N 11`
- `#define DYNAPSE_CONFIG_BIAS_C1_IF_TAU1_N 13`
- `#define DYNAPSE_CONFIG_BIAS_C1_IF_AHTAU_N 15`
- `#define DYNAPSE_CONFIG_BIAS_C1_IF_CASC_N 17`
- `#define DYNAPSE_CONFIG_BIAS_C1_IF_TAU2_N 19`
- `#define DYNAPSE_CONFIG_BIAS_C1_IF_BUF_P 21`
- `#define DYNAPSE_CONFIG_BIAS_C1_IF_AHTHR_N 23`
- `#define DYNAPSE_CONFIG_BIAS_C1_IF_THR_N 25`
- `#define DYNAPSE_CONFIG_BIAS_C1_NPDPIE_THR_S_P 27`
- `#define DYNAPSE_CONFIG_BIAS_C1_NPDPIE_THR_F_P 29`
- `#define DYNAPSE_CONFIG_BIAS_C1_NPDPII_THR_F_P 31`
- `#define DYNAPSE_CONFIG_BIAS_C1_NPDPII_THR_S_P 33`
- `#define DYNAPSE_CONFIG_BIAS_C1_IF_NMDA_N 35`
- `#define DYNAPSE_CONFIG_BIAS_C1_IF_DC_P 37`
- `#define DYNAPSE_CONFIG_BIAS_C1_IF_AHW_P 39`
- `#define DYNAPSE_CONFIG_BIAS_C1_NPDPII_TAU_S_P 41`
- `#define DYNAPSE_CONFIG_BIAS_C1_NPDPII_TAU_F_P 43`
- `#define DYNAPSE_CONFIG_BIAS_C1_NPDPIE_TAU_F_P 45`
- `#define DYNAPSE_CONFIG_BIAS_C1_NPDPIE_TAU_S_P 47`
- `#define DYNAPSE_CONFIG_BIAS_C1_R2R_P 49`
- `#define DYNAPSE_CONFIG_BIAS_U_BUFFER 50`
- `#define DYNAPSE_CONFIG_BIAS_U_SSP 51`
- `#define DYNAPSE_CONFIG_BIAS_U_SSN 52`
- `#define DYNAPSE_CONFIG_BIAS_C2_PULSE_PWLK_P 64`
- `#define DYNAPSE_CONFIG_BIAS_C2_PS_WEIGHT_INH_S_N 66`
- `#define DYNAPSE_CONFIG_BIAS_C2_PS_WEIGHT_INH_F_N 68`
- `#define DYNAPSE_CONFIG_BIAS_C2_PS_WEIGHT_EXC_S_N 70`
- `#define DYNAPSE_CONFIG_BIAS_C2_PS_WEIGHT_EXC_F_N 72`
- `#define DYNAPSE_CONFIG_BIAS_C2_IF_RFR_N 74`
- `#define DYNAPSE_CONFIG_BIAS_C2_IF_TAU1_N 76`
- `#define DYNAPSE_CONFIG_BIAS_C2_IF_AHTAU_N 78`
- `#define DYNAPSE_CONFIG_BIAS_C2_IF_CASC_N 80`
- `#define DYNAPSE_CONFIG_BIAS_C2_IF_TAU2_N 82`
- `#define DYNAPSE_CONFIG_BIAS_C2_IF_BUF_P 84`
- `#define DYNAPSE_CONFIG_BIAS_C2_IF_AHTHR_N 86`
- `#define DYNAPSE_CONFIG_BIAS_C2_IF_THR_N 88`
- `#define DYNAPSE_CONFIG_BIAS_C2_NPDPIE_THR_S_P 90`
- `#define DYNAPSE_CONFIG_BIAS_C2_NPDPIE_THR_F_P 92`
- `#define DYNAPSE_CONFIG_BIAS_C2_NPDPII_THR_F_P 94`
- `#define DYNAPSE_CONFIG_BIAS_C2_NPDPII_THR_S_P 96`
- `#define DYNAPSE_CONFIG_BIAS_C2_IF_NMDA_N 98`
- `#define DYNAPSE_CONFIG_BIAS_C2_IF_DC_P 100`
- `#define DYNAPSE_CONFIG_BIAS_C2_IF_AHW_P 102`
- `#define DYNAPSE_CONFIG_BIAS_C2_NPDPII_TAU_S_P 104`
- `#define DYNAPSE_CONFIG_BIAS_C2_NPDPII_TAU_F_P 106`
- `#define DYNAPSE_CONFIG_BIAS_C2_NPDPIE_TAU_F_P 108`
- `#define DYNAPSE_CONFIG_BIAS_C2_NPDPIE_TAU_S_P 110`
- `#define DYNAPSE_CONFIG_BIAS_C2_R2R_P 112`
- `#define DYNAPSE_CONFIG_BIAS_C3_PULSE_PWLK_P 65`
- `#define DYNAPSE_CONFIG_BIAS_C3_PS_WEIGHT_INH_S_N 67`
- `#define DYNAPSE_CONFIG_BIAS_C3_PS_WEIGHT_INH_F_N 69`
- `#define DYNAPSE_CONFIG_BIAS_C3_PS_WEIGHT_EXC_S_N 71`
- `#define DYNAPSE_CONFIG_BIAS_C3_PS_WEIGHT_EXC_F_N 73`
- `#define DYNAPSE_CONFIG_BIAS_C3_IF_RFR_N 75`

- `#define DYNAPSE_CONFIG_BIAS_C3_IF_TAU1_N` 77
- `#define DYNAPSE_CONFIG_BIAS_C3_IF_AHTAU_N` 79
- `#define DYNAPSE_CONFIG_BIAS_C3_IF_CASC_N` 81
- `#define DYNAPSE_CONFIG_BIAS_C3_IF_TAU2_N` 83
- `#define DYNAPSE_CONFIG_BIAS_C3_IF_BUF_P` 85
- `#define DYNAPSE_CONFIG_BIAS_C3_IF_AHTHR_N` 87
- `#define DYNAPSE_CONFIG_BIAS_C3_IF_THR_N` 89
- `#define DYNAPSE_CONFIG_BIAS_C3_NPDPIE_THR_S_P` 91
- `#define DYNAPSE_CONFIG_BIAS_C3_NPDPIE_THR_F_P` 93
- `#define DYNAPSE_CONFIG_BIAS_C3_NPDPII_THR_F_P` 95
- `#define DYNAPSE_CONFIG_BIAS_C3_NPDPII_THR_S_P` 97
- `#define DYNAPSE_CONFIG_BIAS_C3_IF_NMDA_N` 99
- `#define DYNAPSE_CONFIG_BIAS_C3_IF_DC_P` 101
- `#define DYNAPSE_CONFIG_BIAS_C3_IF_AHW_P` 103
- `#define DYNAPSE_CONFIG_BIAS_C3_NPDPII_TAU_S_P` 105
- `#define DYNAPSE_CONFIG_BIAS_C3_NPDPII_TAU_F_P` 107
- `#define DYNAPSE_CONFIG_BIAS_C3_NPDPIE_TAU_F_P` 109
- `#define DYNAPSE_CONFIG_BIAS_C3_NPDPIE_TAU_S_P` 111
- `#define DYNAPSE_CONFIG_BIAS_C3_R2R_P` 113
- `#define DYNAPSE_CONFIG_BIAS_D_BUFFER` 114
- `#define DYNAPSE_CONFIG_BIAS_D_SSP` 115
- `#define DYNAPSE_CONFIG_BIAS_D_SSN` 116

## Functions

- struct `caer_dynapse_info` `caerDynapseInfoGet` (`caerDeviceHandle` handle)
- `uint32_t` `caerBiasDynapseGenerate` (const struct `caer_bias_dynapse` dynapseBias)
- struct `caer_bias_dynapse` `caerBiasDynapseParse` (const `uint32_t` dynapseBias)
- bool `caerDynapseWriteSramWords` (`caerDeviceHandle` handle, const `uint16_t` \*data, `uint32_t` baseAddr, `size_t` numWords)
- bool `caerDynapseWritePoissonSpikeRate` (`caerDeviceHandle` handle, `uint16_t` neuronAddr, float rateHz)
- bool `caerDynapseWriteSram` (`caerDeviceHandle` handle, `uint8_t` coreId, `uint8_t` neuronAddrCore, `uint8_t` virtualCoreId, bool sx, `uint8_t` dx, bool sy, `uint8_t` dy, `uint8_t` sramId, `uint8_t` destinationCore) `__attribute__((deprecated))`
- bool `caerDynapseWriteSramN` (`caerDeviceHandle` handle, `uint16_t` neuronAddr, `uint8_t` sramId, `uint8_t` virtualCoreId, bool sx, `uint8_t` dx, bool sy, `uint8_t` dy, `uint8_t` destinationCore)
- bool `caerDynapseWriteCam` (`caerDeviceHandle` handle, `uint16_t` inputNeuronAddr, `uint16_t` neuronAddr, `uint8_t` camId, `uint8_t` synapseType)
- bool `caerDynapseSendDataToUSB` (`caerDeviceHandle` handle, const `uint32_t` \*data, `size_t` numConfig)
- `uint32_t` `caerDynapseGenerateCamBits` (`uint16_t` inputNeuronAddr, `uint16_t` neuronAddr, `uint8_t` camId, `uint8_t` synapseType)
- `uint32_t` `caerDynapseGenerateSramBits` (`uint16_t` neuronAddr, `uint8_t` sramId, `uint8_t` virtualCoreId, bool sx, `uint8_t` dx, bool sy, `uint8_t` dy, `uint8_t` destinationCore)
- `uint16_t` `caerDynapseCoreXYToNeuronId` (`uint8_t` coreId, `uint8_t` columnX, `uint8_t` rowY)
- `uint16_t` `caerDynapseCoreAddrToNeuronId` (`uint8_t` coreId, `uint8_t` neuronAddrCore)
- `uint16_t` `caerDynapseSpikeEventGetX` (`caerSpikeEventConst` event)
- `uint16_t` `caerDynapseSpikeEventGetY` (`caerSpikeEventConst` event)
- struct `caer_spike_event` `caerDynapseSpikeEventFromXY` (`uint16_t` x, `uint16_t` y)

### 4.4.1 Detailed Description

Dynap-se specific configuration defines and information structures.

## 4.4.2 Macro Definition Documentation

### 4.4.2.1 CAER\_DEVICE\_DYNAPSE

```
#define CAER_DEVICE_DYNAPSE 3
```

Device type definition for iniLabs Dynap-se FX2-based boards.

### 4.4.2.2 DYNAPSE\_CHIP\_DYNAPSE

```
#define DYNAPSE_CHIP_DYNAPSE 64
```

Dynap-se chip identifier.

### 4.4.2.3 DYNAPSE\_CONFIG\_AER

```
#define DYNAPSE_CONFIG_AER 1
```

Module address: device-side AER configuration (from chip). The AER state machine handshakes with the chip's AER bus and gets the spike events from it. It supports various configurable delays.

### 4.4.2.4 DYNAPSE\_CONFIG\_AER\_ACK\_DELAY

```
#define DYNAPSE_CONFIG_AER_ACK_DELAY 4
```

Parameter address for module DYNAPSE\_CONFIG\_AER: delay capturing the data and acknowledging it on the AER bus for the events by this many LogicClock cycles.

### 4.4.2.5 DYNAPSE\_CONFIG\_AER\_ACK\_EXTENSION

```
#define DYNAPSE_CONFIG_AER_ACK_EXTENSION 6
```

Parameter address for module DYNAPSE\_CONFIG\_AER: extend the length of the acknowledge on the AER bus for the events by this many LogicClock cycles.

### 4.4.2.6 DYNAPSE\_CONFIG\_AER\_EXTERNAL\_AER\_CONTROL

```
#define DYNAPSE_CONFIG_AER_EXTERNAL_AER_CONTROL 10
```

Parameter address for module DYNAPSE\_CONFIG\_AER: enable external AER control. This ensures the chip and the neuron array are running, but doesn't do the handshake and leaves the ACK pin in high-impedance, to allow for an external system to take over the AER communication with the chip. DYNAPSE\_CONFIG\_AER\_RUN has to be turned off for this to work.

#### 4.4.2.7 DYNAPSE\_CONFIG\_AER\_RUN

```
#define DYNAPSE_CONFIG_AER_RUN 3
```

Parameter address for module DYNAPSE\_CONFIG\_AER: run the AER state machine and get spike events from the chip by handshaking with its AER bus.

#### 4.4.2.8 DYNAPSE\_CONFIG\_AER\_WAIT\_ON\_TRANSFER\_STALL

```
#define DYNAPSE_CONFIG_AER_WAIT_ON_TRANSFER_STALL 8
```

Parameter address for module DYNAPSE\_CONFIG\_AER: if the output FIFO for this module is full, stall the AER handshake with the chip and wait until it's free again, instead of just continuing the handshake and dropping the resulting events.

#### 4.4.2.9 DYNAPSE\_CONFIG\_BIAS\_C0\_PULSE\_PWLK\_P

```
#define DYNAPSE_CONFIG_BIAS_C0_PULSE_PWLK_P 0
```

Parameter address for module DYNAPSE\_CONFIG\_BIAS: DYNAPSE chip biases. Bias configuration values must be generated using the proper functions, which are:

- `caerBiasDynapseGenerate()` for Dynap-se coarse-fine (current) biases. See '<https://inilabs.com/support/hardware/user-guide-dynap-se/>' section 'Neuron's behaviors and parameters tuning'.

#### 4.4.2.10 DYNAPSE\_CONFIG\_CHIP

```
#define DYNAPSE_CONFIG_CHIP 5
```

Module address: device-side chip control configuration. This state machine is responsible for configuring the chip's internal control registers, to set special options and biases.

#### 4.4.2.11 DYNAPSE\_CONFIG\_CHIP\_CONTENT

```
#define DYNAPSE_CONFIG_CHIP_CONTENT 2
```

Parameter address for module DYNAPSE\_CONFIG\_CHIP: set the configuration content to send to the chip. Every time this changes, the chip ID is appended and the configuration is sent out to the chip.

#### 4.4.2.12 DYNAPSE\_CONFIG\_CHIP\_ID

```
#define DYNAPSE_CONFIG_CHIP_ID 1
```

Parameter address for module DYNAPSE\_CONFIG\_CHIP: set the chip ID to which configuration content is being sent.

#### 4.4.2.13 DYNAPSE\_CONFIG\_CHIP\_REQ\_DELAY

```
#define DYNAPSE_CONFIG_CHIP_REQ_DELAY 3
```

Parameter address for module DYNAPSE\_CONFIG\_CHIP: delay doing the request after putting out the data by this many LogicClock cycles.

#### 4.4.2.14 DYNAPSE\_CONFIG\_CHIP\_REQ\_EXTENSION

```
#define DYNAPSE_CONFIG_CHIP_REQ_EXTENSION 4
```

Parameter address for module DYNAPSE\_CONFIG\_CHIP: extend the request after receiving the ACK by this many LogicClock cycles.

#### 4.4.2.15 DYNAPSE\_CONFIG\_CHIP\_RUN

```
#define DYNAPSE_CONFIG_CHIP_RUN 0
```

Parameter address for module DYNAPSE\_CONFIG\_CHIP: enable the configuration AER state machine to send bias and control configuration to the chip.

#### 4.4.2.16 DYNAPSE\_CONFIG\_CLEAR\_CAM

```
#define DYNAPSE_CONFIG_CLEAR_CAM 10
```

Clear CAM content, on all cores of a chip. No arguments are used. Remember to select the chip you want to configure before this!

#### 4.4.2.17 DYNAPSE\_CONFIG\_DEFAULT\_SRAM

```
#define DYNAPSE_CONFIG_DEFAULT_SRAM 11
```

Clear SRAM content, use one SRAM cell (cell 0, out of the four available) to monitor neurons via USB. 'modAddr' is the chip ID on which to operate, other arguments are unused. Remember to also select the chip you want to configure before this!

#### 4.4.2.18 DYNAPSE\_CONFIG\_DEFAULT\_SRAM\_EMPTY

```
#define DYNAPSE_CONFIG_DEFAULT_SRAM_EMPTY 13
```

Clear SRAM content, route nothing outside (all four SRAM cells zero). No arguments are used. Remember to select the chip you want to configure before this!

#### 4.4.2.19 DYNAPSE\_CONFIG\_MONITOR\_NEU

```
#define DYNAPSE_CONFIG_MONITOR_NEU 12
```

Setup analog neuron monitoring via SMA connectors. 'modAddr' takes the core ID to be monitored, 'paramAddr' the neuron ID. Remember to select the chip you want to configure before this!

#### 4.4.2.20 DYNAPSE\_CONFIG\_MUX

```
#define DYNAPSE_CONFIG_MUX 0
```

Module address: device-side Multiplexer configuration. The Multiplexer is responsible for mixing, timestamping and outputting (via USB) the various event types generated by the device. It is also responsible for timestamp generation.

#### 4.4.2.21 DYNAPSE\_CONFIG\_MUX\_DROP\_AER\_ON\_TRANSFER\_STALL

```
#define DYNAPSE_CONFIG_MUX_DROP_AER_ON_TRANSFER_STALL 4
```

Parameter address for module DYNAPSE\_CONFIG\_MUX: drop AER events if the USB output FIFO is full, instead of having them pile up at the input FIFOs.

#### 4.4.2.22 DYNAPSE\_CONFIG\_MUX\_FORCE\_CHIP\_BIAS\_ENABLE

```
#define DYNAPSE_CONFIG_MUX_FORCE_CHIP_BIAS_ENABLE 3
```

Parameter address for module DYNAPSE\_CONFIG\_MUX: under normal circumstances, the chip's bias generator is only powered up when either the AER or the configuration state machines are running, to save power. This flag forces the bias generator to be powered up all the time.

#### 4.4.2.23 DYNAPSE\_CONFIG\_MUX\_RUN

```
#define DYNAPSE_CONFIG_MUX_RUN 0
```

Parameter address for module DYNAPSE\_CONFIG\_MUX: run the Multiplexer state machine, which is responsible for mixing the various event types at the device level, timestamping them and outputting them via USB or other connectors.

#### 4.4.2.24 DYNAPSE\_CONFIG\_MUX\_TIMESTAMP\_RESET

```
#define DYNAPSE_CONFIG_MUX_TIMESTAMP_RESET 2
```

Parameter address for module DYNAPSE\_CONFIG\_MUX: reset the Timestamp Generator to zero. This also sends a reset pulse to all connected slave devices, resetting their timestamp too.

#### 4.4.2.25 DYNAPSE\_CONFIG\_MUX\_TIMESTAMP\_RUN

```
#define DYNAPSE_CONFIG_MUX_TIMESTAMP_RUN 1
```

Parameter address for module DYNAPSE\_CONFIG\_MUX: run the Timestamp Generator inside the Multiplexer state machine, which will provide microsecond accurate timestamps to the events passing through.

#### 4.4.2.26 DYNAPSE\_CONFIG\_POISSONSPIKEGEN

```
#define DYNAPSE_CONFIG_POISSONSPIKEGEN 18
```

Module address: Device side poisson generator configuration Provides run/stop control of poisson spike generation and rate setting for 1024 sources.

#### 4.4.2.27 DYNAPSE\_CONFIG\_POISSONSPIKEGEN\_CHIPID

```
#define DYNAPSE_CONFIG_POISSONSPIKEGEN_CHIPID 3
```

Parameter address for module DYNAPSE\_CONFIG\_POISSONSPIKEGEN: Chip ID of the chip that will receive events generated by the poisson spike generator.

#### 4.4.2.28 DYNAPSE\_CONFIG\_POISSONSPIKEGEN\_RUN

```
#define DYNAPSE_CONFIG_POISSONSPIKEGEN_RUN 0
```

Parameter address for module DYNAPSE\_CONFIG\_POISSONSPIKEGEN: Enables or disables generation of poisson spike trains.

#### 4.4.2.29 DYNAPSE\_CONFIG\_POISSONSPIKEGEN\_WRITEADDRESS

```
#define DYNAPSE_CONFIG_POISSONSPIKEGEN_WRITEADDRESS 1
```

Parameter address for module DYNAPSE\_CONFIG\_POISSONSPIKEGEN: Selects the address of a poisson spike train source. Writing to this parameter will apply the rate previously written to the WRITEDATA field.

#### 4.4.2.30 DYNAPSE\_CONFIG\_POISSONSPIKEGEN\_WRITEDATA

```
#define DYNAPSE_CONFIG_POISSONSPIKEGEN_WRITEDATA 2
```

Parameter address for module DYNAPSE\_CONFIG\_POISSONSPIKEGEN: Holds data that will be written to the address specified by WRITEADDRESS.

#### 4.4.2.31 DYNAPSE\_CONFIG\_SPIKEGEN

```
#define DYNAPSE_CONFIG_SPIKEGEN 16
```

Module address: Device side spike generator module configuration. Provides start/stop control of spike train application and selection of fixed/variable inter-spike intervals and their location in memory.

#### 4.4.2.32 DYNAPSE\_CONFIG\_SPIKEGEN\_BASEADDR

```
#define DYNAPSE_CONFIG_SPIKEGEN_BASEADDR 2
```

Parameter address for module DYNAPSE\_CONFIG\_SPIKEGEN: Sets the start address of a spike train in memory.

**4.4.2.33 DYNAPSE\_CONFIG\_SPIKEGEN\_ISI**

```
#define DYNAPSE_CONFIG_SPIKEGEN_ISI 4
```

Parameter address for module DYNAPSE\_CONFIG\_SPIKEGEN: Sets the inter-spike interval that will be used in fixed ISI mode (VARMODE false).

**4.4.2.34 DYNAPSE\_CONFIG\_SPIKEGEN\_ISIBASE**

```
#define DYNAPSE_CONFIG_SPIKEGEN_ISIBASE 5
```

Parameter address for module DYNAPSE\_CONFIG\_SPIKEGEN: Sets the time base resolution for inter-spike intervals as the number of FPGA clock cycles.

**4.4.2.35 DYNAPSE\_CONFIG\_SPIKEGEN\_REPEAT**

```
#define DYNAPSE_CONFIG_SPIKEGEN_REPEAT 6
```

Parameter address for module DYNAPSE\_CONFIG\_SPIKEGEN: Sets repeat mode to true or false.

**4.4.2.36 DYNAPSE\_CONFIG\_SPIKEGEN\_RUN**

```
#define DYNAPSE_CONFIG_SPIKEGEN_RUN 0
```

Parameter address for module DYNAPSE\_CONFIG\_SPIKEGEN: Instructs the spike generator to start applying the configured spike train when the parameter changes from false to true.

**4.4.2.37 DYNAPSE\_CONFIG\_SPIKEGEN\_STIMCOUNT**

```
#define DYNAPSE_CONFIG_SPIKEGEN_STIMCOUNT 3
```

Parameter address for module DYNAPSE\_CONFIG\_SPIKEGEN: Sets the number of events to read from memory for a single application of a spike train.

**4.4.2.38 DYNAPSE\_CONFIG\_SPIKEGEN\_VARMODE**

```
#define DYNAPSE_CONFIG_SPIKEGEN_VARMODE 1
```

Parameter address for module DYNAPSE\_CONFIG\_SPIKEGEN: Selects variable inter-spike interval mode (true) or fixed inter-spike interval mode (false).

**4.4.2.39 DYNAPSE\_CONFIG\_SRAM**

```
#define DYNAPSE_CONFIG_SRAM 14
```

Module address: device side SRAM controller configuration. The module holds an address, a word to be written to SRAM, the most recent word read using a read command, and a read/write command. Reads/writes are triggered when the address field is changed. Example: `caerDynapseWriteSramWords(devHandle, SRAMData, baseAddr, numWords);` Writes numWords words from array SRAMData to the SRAM, starting at baseAddr. This define is for internal use of [caerDynapseWriteSramWords\(\)](#); it can be used on its own, but we recommend using the above function that hides all the internal details of writing to the FPGA SRAM.



#### 4.4.2.40 DYNAPSE\_CONFIG\_SRAM\_ADDRESS

```
#define DYNAPSE_CONFIG_SRAM_ADDRESS 1
```

Parameter address for module DYNAPSE\_CONFIG\_SRAM: Holds the address that will be used for the next read/write. Writing or reading this field will trigger the command contained in the command register to be executed on the FPGA.

#### 4.4.2.41 DYNAPSE\_CONFIG\_SRAM\_BURSTMODE

```
#define DYNAPSE_CONFIG_SRAM_BURSTMODE 5
```

Parameter address for module DYNAPSE\_CONFIG\_SRAM: Burst mode enable for fast writing. Disables updates on address change and instead updates on data change, while automatically incrementing the writing address. Two 16-bit words are written per 32-bit word sent to the SPI controller starting with the least significant half word.

#### 4.4.2.42 DYNAPSE\_CONFIG\_SRAM\_DIRECTION\_POS

```
#define DYNAPSE_CONFIG_SRAM_DIRECTION_POS 0
```

On-chip SRAM for spike routing.

#### 4.4.2.43 DYNAPSE\_CONFIG\_SRAM\_READ

```
#define DYNAPSE_CONFIG_SRAM_READ 0
```

Command for module DYNAPSE\_CONFIG\_SRAM: Read command for the RWCOMMAND field. Example: caer↔DeviceConfigSet(devHandle, DYNAPSE\_CONFIG\_SRAM, DYNAPSE\_CONFIG\_SRAM\_RWCOMMAND, DYNAPSE\_CONFIG\_SRAM\_READ); Sets the SRAM controller up for doing reads.

#### 4.4.2.44 DYNAPSE\_CONFIG\_SRAM\_READDATA

```
#define DYNAPSE_CONFIG_SRAM_READDATA 2
```

Parameter address for module DYNAPSE\_CONFIG\_SRAM: Holds the most recently read data from the SRAM. Read-only parameter.

#### 4.4.2.45 DYNAPSE\_CONFIG\_SRAM\_RWCOMMAND

```
#define DYNAPSE_CONFIG_SRAM_RWCOMMAND 4
```

Parameter address for module DYNAPSE\_CONFIG\_SRAM: Holds the command that will be executed when the address field is written to. Example: caer↔DeviceConfigSet(devHandle, DYNAPSE\_CONFIG\_SRAM, DYNAPSE\_CONFIG\_SRAM\_RWCOMMAND, DYNAPSE\_CONFIG\_SRAM\_WRITE); Sets the SRAM controller up for doing writes. DYNAPSE\_CONFIG\_SRAM\_READ and DYNAPSE\_CONFIG\_SRAM\_WRITE are supported.

#### 4.4.2.46 DYNAPSE\_CONFIG\_SRAM\_WRITE

```
#define DYNAPSE_CONFIG_SRAM_WRITE 1
```

Command for module DYNAPSE\_CONFIG\_SRAM: Write command for the RWCOMMAND field. Example: caerDeviceConfigSet(devHandle, DYNAPSE\_CONFIG\_SRAM, DYNAPSE\_CONFIG\_SRAM\_RWCOMMAND, DYNAPSE\_CONFIG\_SRAM\_WRITE); Sets the SRAM controller up for doing writes.

#### 4.4.2.47 DYNAPSE\_CONFIG\_SRAM\_WRITEDATA

```
#define DYNAPSE_CONFIG_SRAM_WRITEDATA 3
```

Parameter address for module DYNAPSE\_CONFIG\_SRAM: Holds the data that will be written on the next write. Example: caerDeviceConfigSet(devHandle, DYNAPSE\_CONFIG\_SRAM, DYNAPSE\_CONFIG\_SRAM\_WRITE\_DATA, wData); caerDeviceConfigSet(devHandle, DYNAPSE\_CONFIG\_SRAM, DYNAPSE\_CONFIG\_SRAM\_RWCOMMAND, DYNAPSE\_CONFIG\_SRAM\_WRITE); caerDeviceConfigSet(devHandle, DYNAPSE\_CONFIG\_SRAM, DYNAPSE\_CONFIG\_SRAM\_ADDRESS, wAddr); Writes wData to the address specified by wAddr.

#### 4.4.2.48 DYNAPSE\_CONFIG\_SYNAPSERECONFIG

```
#define DYNAPSE_CONFIG_SYNAPSERECONFIG 15
```

Module address: Device side Synapse Reconfiguration module configuration. Provides run control, selection between using a single kernel for all neurons and reading per-neuron kernels from SRAM, programming of the global kernel, as well as target output chip ID selection and SRAM kernel table base address.

#### 4.4.2.49 DYNAPSE\_CONFIG\_SYNAPSERECONFIG\_CHIPSELECT

```
#define DYNAPSE_CONFIG_SYNAPSERECONFIG_CHIPSELECT 3
```

Parameter address for module DYNAPSE\_CONFIG\_SYNAPSERECONFIG: Select which chip outputs should go to.

#### 4.4.2.50 DYNAPSE\_CONFIG\_SYNAPSERECONFIG\_GLOBAKERNEL

```
#define DYNAPSE_CONFIG_SYNAPSERECONFIG_GLOBAKERNEL 1
```

Parameter address for module DYNAPSE\_CONFIG\_SYNAPSERECONFIG: Bits 16 down to 12 select the address in the global kernel table and bits 11 down to 0 specify the data. The 12 data bits are split into 4\*3 synaptic weight bits which map onto positive/negative polarity events from 2 DVS pixels.

#### 4.4.2.51 DYNAPSE\_CONFIG\_SYNAPSERECONFIG\_RUN

```
#define DYNAPSE_CONFIG_SYNAPSERECONFIG_RUN 0
```

Parameter address for module DYNAPSE\_CONFIG\_SYNAPSERECONFIG: Run control. Starts and stops handshaking with DVS.

#### 4.4.2.52 DYNAPSE\_CONFIG\_SYNAPSERECONFIG\_SRAMBASEADDR

```
#define DYNAPSE_CONFIG_SYNAPSERECONFIG_SRAMBASEADDR 4
```

Parameter address for module DYNAPSE\_CONFIG\_SYNAPSERECONFIG: SRAM base address configuration in increments of 32 Kib. Setting this to N will place the SRAM kernel LUT in the range  $[N \cdot 2^{15}, ((N+1) \cdot 2^{15}) - 1]$

#### 4.4.2.53 DYNAPSE\_CONFIG\_SYNAPSERECONFIG\_USESRAMKERNELS

```
#define DYNAPSE_CONFIG_SYNAPSERECONFIG_USESRAMKERNELS 2
```

Parameter address for module DYNAPSE\_CONFIG\_SYNAPSERECONFIG: Boolean parameter for selecting between using kernels stored in SRAM or the global kernel table. 1 for SRAM, 0 for global kernel table.

#### 4.4.2.54 DYNAPSE\_CONFIG\_SYSINFO

```
#define DYNAPSE_CONFIG_SYSINFO 6
```

Module address: device-side system information. The system information module provides various details on the device, such as currently installed logic revision or clock speeds. All its parameters are read-only. This is reserved for internal use and should not be used by anything other than libcaer. Please see the 'struct [caer\\_dynapse\\_info](#)' documentation for more details on what information is available.

#### 4.4.2.55 DYNAPSE\_CONFIG\_SYSINFO\_CHIP\_IDENTIFIER

```
#define DYNAPSE_CONFIG_SYSINFO_CHIP_IDENTIFIER 1
```

Parameter address for module DYNAPSE\_CONFIG\_SYSINFO: read-only parameter, an integer used to identify the different types of sensor chips used on the device. This is reserved for internal use and should not be used by anything other than libcaer. Please see the 'struct [caer\\_dynapse\\_info](#)' documentation to get this information.

#### 4.4.2.56 DYNAPSE\_CONFIG\_SYSINFO\_DEVICE\_IS\_MASTER

```
#define DYNAPSE_CONFIG_SYSINFO_DEVICE_IS_MASTER 2
```

Parameter address for module DYNAPSE\_CONFIG\_SYSINFO: read-only parameter, whether the device is currently a timestamp master or slave when synchronizing multiple devices together. This is reserved for internal use and should not be used by anything other than libcaer. Please see the 'struct [caer\\_dynapse\\_info](#)' documentation to get this information.

#### 4.4.2.57 DYNAPSE\_CONFIG\_SYSINFO\_LOGIC\_CLOCK

```
#define DYNAPSE_CONFIG_SYSINFO_LOGIC_CLOCK 3
```

Parameter address for module DYNAPSE\_CONFIG\_SYSINFO: read-only parameter, the frequency in MHz at which the main FPGA/CPLD logic is running. This is reserved for internal use and should not be used by anything other than libcaer. Please see the 'struct [caer\\_dynapse\\_info](#)' documentation to get this information.

#### 4.4.2.58 DYNAPSE\_CONFIG\_SYSINFO\_LOGIC\_VERSION

```
#define DYNAPSE_CONFIG_SYSINFO_LOGIC_VERSION 0
```

Parameter address for module DYNAPSE\_CONFIG\_SYSINFO: read-only parameter, the version of the logic currently running on the device's FPGA/CPLD. This is reserved for internal use and should not be used by anything other than libcaer. Please see the 'struct [caer\\_dynapse\\_info](#)' documentation to get this information.

#### 4.4.2.59 DYNAPSE\_CONFIG\_USB

```
#define DYNAPSE_CONFIG_USB 9
```

Module address: device-side USB output configuration. The USB output module forwards the data from the device and the FPGA/CPLD to the USB chip, usually a Cypress FX2 or FX3.

#### 4.4.2.60 DYNAPSE\_CONFIG\_USB\_EARLY\_PACKET\_DELAY

```
#define DYNAPSE_CONFIG_USB_EARLY_PACKET_DELAY 1
```

Parameter address for module DYNAPSE\_CONFIG\_USB: the time delay after which a packet of data is committed to USB, even if it is not full yet (short USB packet). The value is in 125 $\mu$ s time-slices, corresponding to how USB schedules its operations (a value of 4 for example would mean waiting at most 0.5ms until sending a short USB packet to the host).

#### 4.4.2.61 DYNAPSE\_CONFIG\_USB\_RUN

```
#define DYNAPSE_CONFIG_USB_RUN 0
```

Parameter address for module DYNAPSE\_CONFIG\_USB: enable the USB FIFO module, which transfers the data from the FPGA/CPLD to the USB chip, to be then sent to the host. Turning this off will suppress any USB data communication!

#### 4.4.2.62 DYNAPSE\_X4BOARD\_COREX

```
#define DYNAPSE_X4BOARD_COREX 4
```

Number of cores in the x direction of the board.

#### 4.4.2.63 DYNAPSE\_X4BOARD\_COREY

```
#define DYNAPSE_X4BOARD_COREY 4
```

Number of cores in the y direction of the board.

#### 4.4.2.64 DYNAPSE\_X4BOARD\_NEUX

```
#define DYNAPSE_X4BOARD_NEUX 64
```

Number of neurons in the x direction of the board.

#### 4.4.2.65 DYNAPSE\_X4BOARD\_NEUY

```
#define DYNAPSE_X4BOARD_NEUY 64
```

Number of neurons in the y direction of the board.

#### 4.4.2.66 DYNAPSE\_X4BOARD\_NUMCHIPS

```
#define DYNAPSE_X4BOARD_NUMCHIPS 4
```

Number of chips on the board.

### 4.4.3 Function Documentation

#### 4.4.3.1 caerBiasDynapseGenerate()

```
uint32_t caerBiasDynapseGenerate (
    const struct caer_bias_dynapse dynapseBias )
```

Transform coarse-fine bias structure into internal integer representation, suited for sending directly to the device via [caerDeviceConfigSet\(\)](#).

##### Parameters

<i>dynapseBias</i>	coarse-fine bias structure.
--------------------	-----------------------------

##### Returns

internal integer representation for device configuration.

#### 4.4.3.2 caerBiasDynapseParse()

```
struct caer_bias_dynapse caerBiasDynapseParse (
    const uint32_t dynapseBias )
```

Transform internal integer representation, as received by calls to [caerDeviceConfigGet\(\)](#), into a coarse-fine bias structure, for easier handling and understanding of the various parameters.

##### Parameters

<i>dynapseBias</i>	internal integer representation from device.
--------------------	--

**Returns**

coarse-fine bias structure.

**4.4.3.3 caerDynapseCoreAddrToNeuronId()**

```
uint16_t caerDynapseCoreAddrToNeuronId (
    uint8_t coreId,
    uint8_t neuronAddrCore )
```

Map core ID and per-core neuron address to the correct chip global neuron address.

**Parameters**

<i>coreId</i>	the chip's core ID, range [0,3].
<i>neuronAddrCore</i>	the neuron's address within this core, range [0,255].

**Returns**

chip global neuron address.

**4.4.3.4 caerDynapseCoreXYToNeuronId()**

```
uint16_t caerDynapseCoreXYToNeuronId (
    uint8_t coreId,
    uint8_t columnX,
    uint8_t rowY )
```

Map core ID and column/row address to the correct chip global neuron address.

**Parameters**

<i>coreId</i>	the chip's core ID, range [0,3].
<i>columnX</i>	the neuron's column address, range [0,15].
<i>rowY</i>	the neuron's row address, range [0,15].

**Returns**

chip global neuron address.

**4.4.3.5 caerDynapseGenerateCamBits()**

```
uint32_t caerDynapseGenerateCamBits (
    uint16_t inputNeuronAddr,
```

```
uint16_t neuronAddr,
uint8_t camId,
uint8_t synapseType )
```

Generate bits to write a single CAM, to specify which spikes are allowed as input into a neuron.

#### Parameters

<i>inputNeuronAddr</i>	the neuron address that should be let in as input to this neuron, range [0,1023] (use <a href="#">caerDynapseCoreXYToNeuronId()</a> for a 2D mapping).
<i>neuronAddr</i>	the neuron to program, range [0,1023] (use <a href="#">caerDynapseCoreXYToNeuronId()</a> for a 2D mapping).
<i>camId</i>	CAM address (synapse), each neuron has 64, range [0,63].
<i>synapseType</i>	one of the four possible synaptic weights: [DYNAPSE_CONFIG_CAMTYPE_F_EXC,DYNAPSE_CONFIG_CAMTYPE_S_EXC,DYNAPSE_CONFIG_CAMTYPE_F_INH,DYNAPSE_CONFIG_CAMTYPE_S_INH].

#### Returns

bits to send to device.

#### 4.4.3.6 caerDynapseGenerateSramBits()

```
uint32_t caerDynapseGenerateSramBits (
    uint16_t neuronAddr,
    uint8_t sramId,
    uint8_t virtualCoreId,
    bool sx,
    uint8_t dx,
    bool sy,
    uint8_t dy,
    uint8_t destinationCore )
```

Generate bits to write one of the 4 SRAMs of a single neuron. Writing the SRAM means writing the destination address of where the spikes will be routed to. This works on the on-chip SRAM!

#### Parameters

<i>neuronAddr</i>	the neuron to program, range [0,1023] (use <a href="#">caerDynapseCoreXYToNeuronId()</a> for a 2D mapping).
<i>sramId</i>	SRAM address (one of four cells), range [0,3].
<i>virtualCoreId</i>	fake source core ID, set it to this value instead of the actual source core ID, range [0,3].
<i>sx</i>	X direction, can be one of: [DYNAPSE_CONFIG_SRAM_DIRECTION_X_EAST,DYNAPSE_CONFIG_SRAM_DIRECTION_X_WEST].
<i>dx</i>	X delta, number of chips to jumps before reaching destination, range is [0,3].
<i>sy</i>	Y direction, can be one of: [DYNAPSE_CONFIG_SRAM_DIRECTION_Y_NORTH,DYNAPSE_CONFIG_SRAM_DIRECTION_Y_SOUTH].
<i>dy</i>	Y delta, number of chips to jumps before reaching destination, range is [0,3].
<i>destinationCore</i>	spike destination core, uses one-hot coding for the 4 cores: [C3,C2,C1,C0] -> [0,0,0,0] (0 decimal) no core, [1,1,1,1] (15 decimal) all cores

**Returns**

bits to send to device.

**4.4.3.7 caerDynapseInfoGet()**

```
struct caer_dynapse_info caerDynapseInfoGet (
    caerDeviceHandle handle )
```

Return basic information on the device, such as its ID, the logic version, and so on. See the 'struct [caer\\_dynapse\\_info](#)' documentation for more details.

**Parameters**

<i>handle</i>	a valid device handle.
---------------	------------------------

**Returns**

a copy of the device information structure if successful, an empty structure (all zeros) on failure.

**4.4.3.8 caerDynapseSendDataToUSB()**

```
bool caerDynapseSendDataToUSB (
    caerDeviceHandle handle,
    const uint32_t * data,
    size_t numConfig )
```

Send array of configuration parameters to the device via USB.

Remember to select the chip you want to configure before calling this function!

**Parameters**

<i>handle</i>	a valid device handle.
<i>data</i>	an array of integers holding configuration data.
<i>numConfig</i>	number of configuration parameters to send.

**Returns**

true on success, false otherwise.



#### 4.4.3.9 caerDynapseSpikeEventFromXY()

```
struct caer_spike_event caerDynapseSpikeEventFromXY (
    uint16_t x,
    uint16_t y )
```

Get the chip ID, core ID and neuron ID from the X and Y coordinates. This is the reverse transform to: [caerDynapseSpikeEventGetX\(\)](#) / [caerDynapseSpikeEventGetY\(\)](#). The return value is a 'struct caer\_spike\_event' because it already has functions to get/set all the needed values.

##### Parameters

<i>x</i>	a X coordinate as returned by <a href="#">caerDynapseSpikeEventGetX()</a> .
<i>y</i>	a Y coordinate as returned by <a href="#">caerDynapseSpikeEventGetY()</a> .

##### Returns

a SpikeEvent struct holding chip ID, core ID and neuron ID.

#### 4.4.3.10 caerDynapseSpikeEventGetX()

```
uint16_t caerDynapseSpikeEventGetX (
    caerSpikeEventConst event )
```

Get the X (column) address for a spike event, in pixels. The (0, 0) address is in the upper left corner.

##### Parameters

<i>event</i>	a valid SpikeEvent pointer. Cannot be NULL.
--------------	---

##### Returns

the event X address in pixels.

#### 4.4.3.11 caerDynapseSpikeEventGetY()

```
uint16_t caerDynapseSpikeEventGetY (
    caerSpikeEventConst event )
```

Get the Y (row) address for a spike event, in pixels. The (0, 0) address is in the upper left corner.

##### Parameters

<i>event</i>	a valid SpikeEvent pointer. Cannot be NULL.
--------------	---

**Returns**

the event Y address in pixels.

**4.4.3.12 caerDynapseWriteCam()**

```
bool caerDynapseWriteCam (
    caerDeviceHandle handle,
    uint16_t inputNeuronAddr,
    uint16_t neuronAddr,
    uint8_t camId,
    uint8_t synapseType )
```

Write a single CAM, to specify which spikes are allowed as input into a neuron.

Remember to select the chip you want to configure before calling this function!

**Parameters**

<i>handle</i>	a valid device handle.
<i>inputNeuronAddr</i>	the neuron address that should be let in as input to this neuron, range [0,1023].
<i>neuronAddr</i>	the neuron address whose CAM should be programmed, range [0,1023].
<i>camId</i>	CAM address (synapse), each neuron has 64, range [0,63].
<i>synapseType</i>	one of the four possible synaptic weights: [DYNAPSE_CONFIG_CAMTYPE_F_EXC,DYNAPSE_CONFIG_CAMTYPE_S_EXC,DY↔ NAPSE_CONFIG_CAMTYPE_F_INH,DYNAPSE_CONFIG_CAMTYPE_S_INH].

**Returns**

true on success, false otherwise.

**4.4.3.13 caerDynapseWritePoissonSpikeRate()**

```
bool caerDynapseWritePoissonSpikeRate (
    caerDeviceHandle handle,
    uint16_t neuronAddr,
    float rateHz )
```

Specifies the poisson spike generator's spike rate.

**Parameters**

<i>handle</i>	a valid device handle.
<i>neuronAddr</i>	The target neuron of the poisson spike train, range [0,1023].
<i>rateHz</i>	The rate in Hz of the spike train, this will be quantized to the nearest supported level, range [0,4300].

**Returns**

true on success, false otherwise.

**4.4.3.14 caerDynapseWriteSram()**

```
bool caerDynapseWriteSram (
    caerDeviceHandle handle,
    uint8_t coreId,
    uint8_t neuronAddrCore,
    uint8_t virtualCoreId,
    bool sx,
    uint8_t dx,
    bool sy,
    uint8_t dy,
    uint8_t sramId,
    uint8_t destinationCore )
```

THIS FUNCTION IS DEPRECATED. USE [caerDynapseWriteSramN\(\)](#) INSTEAD! The new function uses the global neuron ID (range [0,1023]) like all others, instead of the separate core ID/neuron ID syntax. Also the arguments are in the same order as [caerDynapseGenerateSramBits\(\)](#), in particular the 'sramId' comes right after 'neuronId'.

Write one of the 4 SRAMs of a single neuron. Writing the SRAM means writing the destination address of where the spikes will be routed to. This works on the on-chip SRAM!

Remember to select the chip you want to configure before calling this function!

**Parameters**

<i>handle</i>	a valid device handle.
<i>coreId</i>	the chip's core ID, range [0,3].
<i>neuronAddrCore</i>	the neuron's address within this core, range [0,255].
<i>virtualCoreId</i>	fake source core ID, set it to this value instead of the actual source core ID, range [0,3].
<i>sx</i>	X direction, can be one of: [DYNAPSE_CONFIG_SRAM_DIRECTION_X_EAST,DYNAPSE_CONFIG_SRAM_DIRECTION_X_WEST].
<i>dx</i>	X delta, number of chips to jumps before reaching destination, range is [0,3].
<i>sy</i>	Y direction, can be one of: [DYNAPSE_CONFIG_SRAM_DIRECTION_Y_NORTH,DYNAPSE_CONFIG_SRAM_DIRECTION_Y_SOUTH].
<i>dy</i>	Y delta, number of chips to jumps before reaching destination, range is [0,3].
<i>sramId</i>	SRAM address (one of four cells), range [0,3].
<i>destinationCore</i>	spike destination core, uses one-hot coding for the 4 cores: [C3,C2,C1,C0] -> [0,0,0,0] (0 decimal) no core, [1,1,1,1] (15 decimal) all cores

**Returns**

true on success, false otherwise.

#### 4.4.3.15 caerDynapseWriteSramN()

```
bool caerDynapseWriteSramN (
    caerDeviceHandle handle,
    uint16_t neuronAddr,
    uint8_t sramId,
    uint8_t virtualCoreId,
    bool sx,
    uint8_t dx,
    bool sy,
    uint8_t dy,
    uint8_t destinationCore )
```

Write one of the 4 SRAMs of a single neuron. Writing the SRAM means writing the destination address of where the spikes will be routed to. This works on the on-chip SRAM!

Remember to select the chip you want to configure before calling this function!

##### Parameters

<i>handle</i>	a valid device handle.
<i>neuronAddr</i>	the neuron to program, range [0,1023] (use <a href="#">caerDynapseCoreXYToNeuronId()</a> for a 2D mapping).
<i>sramId</i>	SRAM address (one of four cells), range [0,3].
<i>virtualCoreId</i>	fake source core ID, set it to this value instead of the actual source core ID, range [0,3].
<i>sx</i>	X direction, can be one of: [DYNAPSE_CONFIG_SRAM_DIRECTION_X_EAST,DYNAPSE_CONFIG_SRAM_DIRECTION_X_WEST].
<i>dx</i>	X delta, number of chips to jumps before reaching destination, range is [0,3].
<i>sy</i>	Y direction, can be one of: [DYNAPSE_CONFIG_SRAM_DIRECTION_Y_NORTH,DYNAPSE_CONFIG_SRAM_DIRECTION_Y_SOUTH].
<i>dy</i>	Y delta, number of chips to jumps before reaching destination, range is [0,3].
<i>destinationCore</i>	spike destination core, uses one-hot coding for the 4 cores: [C3,C2,C1,C0] -> [0,0,0,0] (0 decimal) no core, [1,1,1,1] (15 decimal) all cores

##### Returns

true on success, false otherwise.

#### 4.4.3.16 caerDynapseWriteSramWords()

```
bool caerDynapseWriteSramWords (
    caerDeviceHandle handle,
    const uint16_t * data,
    uint32_t baseAddr,
    size_t numWords )
```

Transfer 16bit words from memory to device SRAM, with configurable starting address and number of words. This works on the FPGA SRAM!

#### Parameters

<i>handle</i>	a valid device handle.
<i>data</i>	array from which to read data to send to SRAM.
<i>baseAddr</i>	SRAM start address where to put the data.
<i>numWords</i>	number of 16bit words to transfer.

#### Returns

true on success, false otherwise.

## 4.5 devices/edvs.h File Reference

```
#include "serial.h"
#include "../events/polarity.h"
#include "../events/special.h"
```

#### Data Structures

- struct [caer\\_edvs\\_info](#)

#### Macros

- #define [CAER\\_DEVICE\\_EDVS](#) 5
- #define [EDVS\\_CONFIG\\_DVS](#) 0
- #define [EDVS\\_CONFIG\\_BIAS](#) 1
- #define [EDVS\\_CONFIG\\_DVS\\_RUN](#) 0
- #define [EDVS\\_CONFIG\\_DVS\\_TIMESTAMP\\_RESET](#) 1
- #define [EDVS\\_CONFIG\\_BIAS\\_CAS](#) 0
- #define [EDVS\\_CONFIG\\_BIAS\\_INJGND](#) 1
- #define [EDVS\\_CONFIG\\_BIAS\\_REQPD](#) 2
- #define [EDVS\\_CONFIG\\_BIAS\\_PUX](#) 3
- #define [EDVS\\_CONFIG\\_BIAS\\_DIFFOFF](#) 4
- #define [EDVS\\_CONFIG\\_BIAS\\_REQ](#) 5
- #define [EDVS\\_CONFIG\\_BIAS\\_REFR](#) 6
- #define [EDVS\\_CONFIG\\_BIAS\\_PUY](#) 7
- #define [EDVS\\_CONFIG\\_BIAS\\_DIFFON](#) 8
- #define [EDVS\\_CONFIG\\_BIAS\\_DIFF](#) 9
- #define [EDVS\\_CONFIG\\_BIAS\\_FOLL](#) 10
- #define [EDVS\\_CONFIG\\_BIAS\\_PR](#) 11

#### Functions

- struct [caer\\_edvs\\_info](#) [caerEDVSInfoGet](#) ([caerDeviceHandle](#) handle)

### 4.5.1 Detailed Description

eDVS4337 specific configuration defines and information structures.

### 4.5.2 Macro Definition Documentation

#### 4.5.2.1 CAER\_DEVICE\_EDVS

```
#define CAER_DEVICE_EDVS 5
```

Device type definition for iniLabs eDVS4337.

#### 4.5.2.2 EDVS\_CONFIG\_BIAS

```
#define EDVS_CONFIG_BIAS 1
```

Module address: device-side chip bias generator configuration.

#### 4.5.2.3 EDVS\_CONFIG\_BIAS\_CAS

```
#define EDVS_CONFIG_BIAS_CAS 0
```

Parameter address for module EDVS\_CONFIG\_BIAS: First stage amplifier cascode bias. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.5.2.4 EDVS\_CONFIG\_BIAS\_DIFF

```
#define EDVS_CONFIG_BIAS_DIFF 9
```

Parameter address for module EDVS\_CONFIG\_BIAS: Differential (second stage amplifier) bias. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.5.2.5 EDVS\_CONFIG\_BIAS\_DIFFOFF

```
#define EDVS_CONFIG_BIAS_DIFFOFF 4
```

Parameter address for module EDVS\_CONFIG\_BIAS: Off events threshold bias. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.5.2.6 EDVS\_CONFIG\_BIAS\_DIFFON

```
#define EDVS_CONFIG_BIAS_DIFFON 8
```

Parameter address for module EDVS\_CONFIG\_BIAS: On events threshold bias. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.5.2.7 EDVS\_CONFIG\_BIAS\_FOLL

```
#define EDVS_CONFIG_BIAS_FOLL 10
```

Parameter address for module EDVS\_CONFIG\_BIAS: Source follower bias. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.5.2.8 EDVS\_CONFIG\_BIAS\_INJGND

```
#define EDVS_CONFIG_BIAS_INJGND 1
```

Parameter address for module EDVS\_CONFIG\_BIAS: Injected ground bias. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.5.2.9 EDVS\_CONFIG\_BIAS\_PR

```
#define EDVS_CONFIG_BIAS_PR 11
```

Parameter address for module EDVS\_CONFIG\_BIAS: Photoreceptor bias. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.5.2.10 EDVS\_CONFIG\_BIAS\_PUX

```
#define EDVS_CONFIG_BIAS_PUX 3
```

Parameter address for module EDVS\_CONFIG\_BIAS: Pull up on request from X arbiter (AER). See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.5.2.11 EDVS\_CONFIG\_BIAS\_PUY

```
#define EDVS_CONFIG_BIAS_PUY 7
```

Parameter address for module EDVS\_CONFIG\_BIAS: Pull up on request from Y arbiter (AER). See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.5.2.12 EDVS\_CONFIG\_BIAS\_REFR

```
#define EDVS_CONFIG_BIAS_REFR 6
```

Parameter address for module EDVS\_CONFIG\_BIAS: Refractory period bias. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.5.2.13 EDVS\_CONFIG\_BIAS\_REQ

```
#define EDVS_CONFIG_BIAS_REQ 5
```

Parameter address for module EDVS\_CONFIG\_BIAS: Pull down for passive load inverters in digital AER pixel circuitry. See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.5.2.14 EDVS\_CONFIG\_BIAS\_REQPD

```
#define EDVS_CONFIG_BIAS_REQPD 2
```

Parameter address for module EDVS\_CONFIG\_BIAS: Pull down on chip request (AER). See '<http://inilabs.com/support/biasing/>' for more details.

#### 4.5.2.15 EDVS\_CONFIG\_DVS

```
#define EDVS_CONFIG_DVS 0
```

Module address: device-side DVS configuration.

#### 4.5.2.16 EDVS\_CONFIG\_DVS\_RUN

```
#define EDVS_CONFIG_DVS_RUN 0
```

Parameter address for module EDVS\_CONFIG\_DVS: run the DVS chip and generate polarity event data.

#### 4.5.2.17 EDVS\_CONFIG\_DVS\_TIMESTAMP\_RESET

```
#define EDVS_CONFIG_DVS_TIMESTAMP_RESET 1
```

Parameter address for module EDVS\_CONFIG\_DVS: reset the time-stamp counter of the device. This is a temporary configuration switch and will reset itself right away.

### 4.5.3 Function Documentation

#### 4.5.3.1 caerEDVSInfoGet()

```
struct caer_edvs_info caerEDVSInfoGet (
    caerDeviceHandle handle )
```

Return basic information on the device, such as its ID, its resolution, the logic version, and so on. See the 'struct [caer\\_edvs\\_info](#)' documentation for more details.

##### Parameters

<i>handle</i>	a valid device handle.
---------------	------------------------

##### Returns

a copy of the device information structure if successful, an empty structure (all zeros) on failure.



## 4.6 devices/serial.h File Reference

```
#include "device.h"
```

### Macros

- `#define CAER_HOST_CONFIG_SERIAL -1`
- `#define CAER_HOST_CONFIG_SERIAL_READ_SIZE 0`
- `#define CAER_HOST_CONFIG_SERIAL_BAUD_RATE_2M 2000000`
- `#define CAER_HOST_CONFIG_SERIAL_BAUD_RATE_4M 4000000`
- `#define CAER_HOST_CONFIG_SERIAL_BAUD_RATE_8M 8000000`
- `#define CAER_HOST_CONFIG_SERIAL_BAUD_RATE_12M 12000000`

### Functions

- `caerDeviceHandle caerDeviceOpenSerial` (uint16\_t deviceId, uint16\_t deviceType, const char \*serialPort↔  
Name, uint32\_t serialBaudRate)

### 4.6.1 Detailed Description

Common functions to access, configure and exchange data with supported serial port devices. Also contains defines for serial port specific configuration options.

### 4.6.2 Macro Definition Documentation

#### 4.6.2.1 CAER\_HOST\_CONFIG\_SERIAL

```
#define CAER_HOST_CONFIG_SERIAL -1
```

Module address: host-side serial port configuration.

#### 4.6.2.2 CAER\_HOST\_CONFIG\_SERIAL\_BAUD\_RATE\_12M

```
#define CAER_HOST_CONFIG_SERIAL_BAUD_RATE_12M 12000000
```

Parameter values for module CAER\_HOST\_CONFIG\_SERIAL: possible baud-rates for serial port communication.

#### 4.6.2.3 CAER\_HOST\_CONFIG\_SERIAL\_BAUD\_RATE\_2M

```
#define CAER_HOST_CONFIG_SERIAL_BAUD_RATE_2M 2000000
```

Parameter values for module CAER\_HOST\_CONFIG\_SERIAL: possible baud-rates for serial port communication.

#### 4.6.2.4 CAER\_HOST\_CONFIG\_SERIAL\_BAUD\_RATE\_4M

```
#define CAER_HOST_CONFIG_SERIAL_BAUD_RATE_4M 4000000
```

Parameter values for module CAER\_HOST\_CONFIG\_SERIAL: possible baud-rates for serial port communication.

#### 4.6.2.5 CAER\_HOST\_CONFIG\_SERIAL\_BAUD\_RATE\_8M

```
#define CAER_HOST_CONFIG_SERIAL_BAUD_RATE_8M 8000000
```

Parameter values for module CAER\_HOST\_CONFIG\_SERIAL: possible baud-rates for serial port communication.

#### 4.6.2.6 CAER\_HOST\_CONFIG\_SERIAL\_READ\_SIZE

```
#define CAER_HOST_CONFIG_SERIAL_READ_SIZE 0
```

Parameter address for module CAER\_HOST\_CONFIG\_SERIAL: read size for serial port communication.

### 4.6.3 Function Documentation

#### 4.6.3.1 caerDeviceOpenSerial()

```
caerDeviceHandle caerDeviceOpenSerial (
    uint16_t deviceID,
    uint16_t deviceType,
    const char * serialPortName,
    uint32_t serialBaudRate )
```

Open a specified serial port device, assign an ID to it and return a handle for further usage. Various means can be employed to limit the selection of the device.

##### Parameters

<i>deviceID</i>	a unique ID to identify the device from others. Will be used as the source for EventPackets being generate from its data.
<i>deviceType</i>	type of the device to open. Currently supported are: CAER_DEVICE_EDVS
<i>serialPortName</i>	name of the serial port device to open.
<i>serialBaudRate</i>	baud-rate for serial port communication.

### Returns

a valid device handle that can be used with the other libcaer functions, or NULL on error. Always check for this!

## 4.7 devices/usb.h File Reference

```
#include "device.h"
```

### Macros

- `#define CAER_HOST_CONFIG_USB -1`
- `#define CAER_HOST_CONFIG_USB_BUFFER_NUMBER 0`
- `#define CAER_HOST_CONFIG_USB_BUFFER_SIZE 1`

### Functions

- `caerDeviceHandle caerDeviceOpen` (uint16\_t deviceId, uint16\_t deviceType, uint8\_t busNumberRestrict, uint8\_t devAddressRestrict, const char \*serialNumberRestrict)

#### 4.7.1 Detailed Description

Common functions to access, configure and exchange data with supported USB devices. Also contains defines for USB specific configuration options.

#### 4.7.2 Macro Definition Documentation

##### 4.7.2.1 CAER\_HOST\_CONFIG\_USB

```
#define CAER_HOST_CONFIG_USB -1
```

Module address: host-side USB configuration.

##### 4.7.2.2 CAER\_HOST\_CONFIG\_USB\_BUFFER\_NUMBER

```
#define CAER_HOST_CONFIG_USB_BUFFER_NUMBER 0
```

Parameter address for module CAER\_HOST\_CONFIG\_USB: set number of buffers used by libusb for asynchronous data transfers with the USB device. The default values are usually fine, only change them if you're running into I/O limits.

#### 4.7.2.3 CAER\_HOST\_CONFIG\_USB\_BUFFER\_SIZE

```
#define CAER_HOST_CONFIG_USB_BUFFER_SIZE 1
```

Parameter address for module CAER\_HOST\_CONFIG\_USB: set size of each buffer used by libusb for asynchronous data transfers with the USB device. The default values are usually fine, only change them if you're running into I/O limits.

### 4.7.3 Function Documentation

#### 4.7.3.1 caerDeviceOpen()

```
caerDeviceHandle caerDeviceOpen (
    uint16_t deviceID,
    uint16_t deviceType,
    uint8_t busNumberRestrict,
    uint8_t devAddressRestrict,
    const char * serialNumberRestrict )
```

Open a specified USB device, assign an ID to it and return a handle for further usage. Various means can be employed to limit the selection of the device.

##### Parameters

<i>deviceID</i>	a unique ID to identify the device from others. Will be used as the source for EventPackets being generate from its data.
<i>deviceType</i>	type of the device to open. Currently supported are: CAER_DEVICE_DVS128, CAER_DEVICE_DAVIS, CAER_DEVICE_DYNAPSE
<i>busNumberRestrict</i>	restrict the search for viable devices to only this USB bus number.
<i>devAddressRestrict</i>	restrict the search for viable devices to only this USB device address.
<i>serialNumberRestrict</i>	restrict the search for viable devices to only devices which do possess the given Serial Number in their USB SerialNumber descriptor.

##### Returns

a valid device handle that can be used with the other libcaer functions, or NULL on error. Always check for this!

## 4.8 events/common.h File Reference

```
#include "../libcaer.h"
```

##### Macros

- #define TS\_OVERFLOW\_SHIFT 31

- `#define CAER_DEFAULT_EVENT_TYPES_COUNT 13`
  - `#define CAER_EVENT_PACKET_HEADER_SIZE 28`
  - `#define CAER_ITERATOR_ALL_START(PACKET_HEADER, EVENT_TYPE)`
  - `#define CAER_ITERATOR_ALL_END }`
  - `#define CAER_ITERATOR_VALID_START(PACKET_HEADER, EVENT_TYPE)`
  - `#define CAER_ITERATOR_VALID_END }`
- 
- `#define VALID_MARK_SHIFT 0`
  - `#define VALID_MARK_MASK 0x00000001`

## Typedefs

- `typedef struct caer_event_packet_header * caerEventPacketHeader`
- `typedef const struct caer_event_packet_header * caerEventPacketHeaderConst`

## Enumerations

- `enum caer_default_event_types {`  
`SPECIAL_EVENT = 0, POLARITY_EVENT = 1, FRAME_EVENT = 2, IMU6_EVENT = 3,`  
`IMU9_EVENT = 4, SAMPLE_EVENT = 5, EAR_EVENT = 6, CONFIG_EVENT = 7,`  
`POINT1D_EVENT = 8, POINT2D_EVENT = 9, POINT3D_EVENT = 10, POINT4D_EVENT = 11,`  
`SPIKE_EVENT = 12 }`

## Functions

- `PACKED_STRUCT` (struct caer\_event\_packet\_header { int16\_t eventType;int16\_t eventSource;int32\_t eventSize;int32\_t eventTSOffset;int32\_t eventTSOverflow;int32\_t eventCapacity;int32\_t eventNumber;int32\_t eventValid;})
- `static int16_t caerEventPacketHeaderGetEventType (caerEventPacketHeaderConst header)`
- `static void caerEventPacketHeaderSetEventType (caerEventPacketHeader header, int16_t eventType)`
- `static int16_t caerEventPacketHeaderGetEventSource (caerEventPacketHeaderConst header)`
- `static void caerEventPacketHeaderSetEventSource (caerEventPacketHeader header, int16_t eventSource)`
- `static int32_t caerEventPacketHeaderGetEventSize (caerEventPacketHeaderConst header)`
- `static void caerEventPacketHeaderSetEventSize (caerEventPacketHeader header, int32_t eventSize)`
- `static int32_t caerEventPacketHeaderGetEventTSOffset (caerEventPacketHeaderConst header)`
- `static void caerEventPacketHeaderSetEventTSOffset (caerEventPacketHeader header, int32_t eventTSOffset)`
- `static int32_t caerEventPacketHeaderGetEventTSOverflow (caerEventPacketHeaderConst header)`
- `static void caerEventPacketHeaderSetEventTSOverflow (caerEventPacketHeader header, int32_t eventTSOverflow)`
- `static int32_t caerEventPacketHeaderGetEventCapacity (caerEventPacketHeaderConst header)`
- `static void caerEventPacketHeaderSetEventCapacity (caerEventPacketHeader header, int32_t eventsCapacity)`
- `static int32_t caerEventPacketHeaderGetEventNumber (caerEventPacketHeaderConst header)`
- `static void caerEventPacketHeaderSetEventNumber (caerEventPacketHeader header, int32_t eventsNumber)`
- `static int32_t caerEventPacketHeaderGetEventValid (caerEventPacketHeaderConst header)`
- `static void caerEventPacketHeaderSetEventValid (caerEventPacketHeader header, int32_t eventsValid)`
- `static const void * caerGenericEventGetEvent (caerEventPacketHeaderConst headerPtr, int32_t n)`

- static int32\_t [caerGenericEventGetTimestamp](#) (const void \*eventPtr, caerEventPacketHeaderConst headerPtr)
- static int64\_t [caerGenericEventGetTimestamp64](#) (const void \*eventPtr, caerEventPacketHeaderConst headerPtr)
- static bool [caerGenericEventsValid](#) (const void \*eventPtr)
- static bool [caerGenericEventCopy](#) (void \*eventPtrDestination, const void \*eventPtrSource, caerEventPacketHeaderConst headerPtrDestination, caerEventPacketHeaderConst headerPtrSource)
- static int64\_t [caerEventPacketGetDataSize](#) (caerEventPacketHeaderConst header)
- static int64\_t [caerEventPacketGetSize](#) (caerEventPacketHeaderConst header)
- static bool [caerEventPacketEquals](#) (caerEventPacketHeaderConst firstPacket, caerEventPacketHeaderConst secondPacket)
- static void [caerEventPacketClear](#) (caerEventPacketHeader packet)
- static void [caerEventPacketClean](#) (caerEventPacketHeader packet)
- **memset** (((uint8\_t \*) packet)+offset, 0, (size\_t)((eventCapacity - eventValid) \* eventSize))
- **caerEventPacketHeaderSetEventNumber** (packet, eventValid)
- static caerEventPacketHeader [caerEventPacketResize](#) (caerEventPacketHeader packet, int32\_t newEventCapacity)
- static caerEventPacketHeader [caerEventPacketGrow](#) (caerEventPacketHeader packet, int32\_t newEventCapacity)
- static caerEventPacketHeader [caerEventPacketAppend](#) (caerEventPacketHeader packet, caerEventPacketHeader appendPacket)
- static caerEventPacketHeader [caerEventPacketCopy](#) (caerEventPacketHeaderConst packet)
- static caerEventPacketHeader [caerEventPacketCopyOnlyEvents](#) (caerEventPacketHeaderConst packet)
- static caerEventPacketHeader [caerEventPacketCopyOnlyValidEvents](#) (caerEventPacketHeaderConst packet)
- **caerEventPacketHeaderSetEventCapacity** (packetCopy, eventValid)
- **caerEventPacketHeaderSetEventNumber** (packetCopy, eventValid)
- **return** (packetCopy)

#### 4.8.1 Detailed Description

Common EventPacket header format definition and handling functions. Every EventPacket, of any type, has as a first member a common header, which describes various properties of the contained events. This allows easy parsing of events. See the 'struct caer\_event\_packet\_header' documentation for more details.

#### 4.8.2 Macro Definition Documentation

##### 4.8.2.1 CAER\_DEFAULT\_EVENT\_TYPES\_COUNT

```
#define CAER_DEFAULT_EVENT_TYPES_COUNT 13
```

Number of default event types that are part of libcaer. Corresponds to the count of definitions inside the 'enum caer\_default\_event\_types' enumeration.

##### 4.8.2.2 CAER\_EVENT\_PACKET\_HEADER\_SIZE

```
#define CAER_EVENT_PACKET_HEADER_SIZE 28
```

Size of the EventPacket header. This is constant across all supported systems.

## 4.8.2.3 CAER\_ITERATOR\_ALL\_END

```
#define CAER_ITERATOR_ALL_END }
```

Generic iterator close statement.

## 4.8.2.4 CAER\_ITERATOR\_ALL\_START

```
#define CAER_ITERATOR_ALL_START(  
    PACKET_HEADER,  
    EVENT_TYPE )
```

**Value:**

```
for (int32_t caerIteratorCounter = 0; \  
    caerIteratorCounter < caerEventPacketHeaderGetEventNumber(  
    PACKET_HEADER); \  
    caerIteratorCounter++) { \  
    EVENT_TYPE caerIteratorElement = (EVENT_TYPE) caerGenericEventGetEvent(  
    PACKET_HEADER, caerIteratorCounter);
```

Generic iterator over all events in a packet. Returns the current index in the 'caerIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerIteratorElement' variable of type EVENT\_TYPE.

PACKET\_HEADER: a valid EventPacket header pointer. Cannot be NULL. EVENT\_TYPE: the event pointer type for this EventPacket (ie. caerPolarityEvent or caerFrameEvent).

## 4.8.2.5 CAER\_ITERATOR\_VALID\_END

```
#define CAER_ITERATOR_VALID_END }
```

Generic iterator close statement.

## 4.8.2.6 CAER\_ITERATOR\_VALID\_START

```
#define CAER_ITERATOR_VALID_START(  
    PACKET_HEADER,  
    EVENT_TYPE )
```

**Value:**

```
for (int32_t caerIteratorCounter = 0; \  
    caerIteratorCounter < caerEventPacketHeaderGetEventNumber(  
    PACKET_HEADER); \  
    caerIteratorCounter++) { \  
    EVENT_TYPE caerIteratorElement = (EVENT_TYPE) caerGenericEventGetEvent(  
    PACKET_HEADER, caerIteratorCounter); \  
    if (!caerGenericEventIsValid(caerIteratorElement)) { continue; }
```

Generic iterator over only the valid events in a packet. Returns the current index in the 'caerIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerIteratorElement' variable of type EVENT\_TYPE.

PACKET\_HEADER: a valid EventPacket header pointer. Cannot be NULL. EVENT\_TYPE: the event pointer type for this EventPacket (ie. caerPolarityEvent or caerFrameEvent).

#### 4.8.2.7 TS\_OVERFLOW\_SHIFT

```
#define TS_OVERFLOW_SHIFT 31
```

64bit timestamp support: since timestamps wrap around after some time, being only 31 bit (32 bit signed int), another timestamp at the packet level provides another 31 bit (32 bit signed int), to enable the generation of a 62 bit (64 bit signed int) microsecond timestamp which is guaranteed to never wrap around (in the next 146'138 years at least). The TSOvflow needs to be shifted by 31 thus when constructing such a timestamp.

#### 4.8.2.8 VALID\_MARK\_MASK

```
#define VALID_MARK_MASK 0x00000001
```

Generic validity mark: this bit is used to mark whether an event is still valid or not, and can be used to efficiently filter out events from a packet. The caerXXXEventValidate() and caerXXXEventInvalidate() functions should be used to toggle this! 0 in the 0th bit of the first byte means invalid, 1 means valid. This way zeroing-out an event packet sets all its events to invalid. Care must be taken to put the field containing the validity mark always as the first member of an event.

#### 4.8.2.9 VALID\_MARK\_SHIFT

```
#define VALID_MARK_SHIFT 0
```

Generic validity mark: this bit is used to mark whether an event is still valid or not, and can be used to efficiently filter out events from a packet. The caerXXXEventValidate() and caerXXXEventInvalidate() functions should be used to toggle this! 0 in the 0th bit of the first byte means invalid, 1 means valid. This way zeroing-out an event packet sets all its events to invalid. Care must be taken to put the field containing the validity mark always as the first member of an event.

### 4.8.3 Typedef Documentation

#### 4.8.3.1 caerEventPacketHeader

```
typedef struct caer_event_packet_header* caerEventPacketHeader
```

Type for pointer to EventPacket header data structure.

### 4.8.4 Enumeration Type Documentation

#### 4.8.4.1 caer\_default\_event\_types

```
enum caer_default_event_types
```

List of supported event types. Each event type has its own integer representation. All event types below 100 are reserved for use by libcaer and cAER. DO NOT USE THEM FOR YOUR OWN EVENT TYPES!



## Enumerator

SPECIAL_EVENT	Special events.
POLARITY_EVENT	Polarity (change, DVS) events.
FRAME_EVENT	Frame (intensity, APS) events.
IMU6_EVENT	6 axes IMU events.
IMU9_EVENT	9 axes IMU events.
SAMPLE_EVENT	ADC sample events.
EAR_EVENT	Ear (cochlea) events.
CONFIG_EVENT	Device configuration events.
POINT1D_EVENT	1D measurement events.
POINT2D_EVENT	2D measurement events.
POINT3D_EVENT	3D measurement events.
POINT4D_EVENT	4D measurement events.
SPIKE_EVENT	Spike events.

## 4.8.5 Function Documentation

## 4.8.5.1 caerEventPacketAppend()

```
static caerEventPacketHeader caerEventPacketAppend (
    caerEventPacketHeader packet,
    caerEventPacketHeader appendPacket ) [inline], [static]
```

Appends an event packet to another. This is a simple append operation, no timestamp reordering is done. Please ensure time is monotonically increasing over the two packets! Use free() to reclaim this memory afterwards.

## Parameters

<i>packet</i>	the main events packet.
<i>appendPacket</i>	the events packet to append on the main one.

## Returns

a valid event packet handle or NULL on error. On success, the old packet handle is to be considered invalid and not to be used anymore. On failure, the old packet handle is not touched in any way. The appendPacket handle is never touched in any way.

## 4.8.5.2 caerEventPacketClean()

```
static void caerEventPacketClean (
    caerEventPacketHeader packet ) [inline], [static]
```

Clean a packet by removing all invalid events, so that the total number of events is the number of valid events. The packet's capacity doesn't change.

**Parameters**

<i>packet</i>	an event packet to clean.
---------------	---------------------------

**4.8.5.3 caerEventPacketClear()**

```
static void caerEventPacketClear (
    caerEventPacketHeader packet ) [inline], [static]
```

Clear a packet by zeroing out all events. Capacity doesn't change, event number is set to zero.

**Parameters**

<i>packet</i>	an event packet to clear out.
---------------	-------------------------------

**4.8.5.4 caerEventPacketCopy()**

```
static caerEventPacketHeader caerEventPacketCopy (
    caerEventPacketHeaderConst packet ) [inline], [static]
```

Make a full copy of an event packet (up to eventCapacity).

**Parameters**

<i>packet</i>	an event packet to copy.
---------------	--------------------------

**Returns**

a full copy of an event packet.

**4.8.5.5 caerEventPacketCopyOnlyEvents()**

```
static caerEventPacketHeader caerEventPacketCopyOnlyEvents (
    caerEventPacketHeaderConst packet ) [inline], [static]
```

Make a copy of an event packet, sized down to only include the currently present events (eventNumber, valid+invalid), and not including the possible extra unused events (up to eventCapacity).

**Parameters**

<i>packet</i>	an event packet to copy.
---------------	--------------------------

**Returns**

a sized down copy of an event packet.

**4.8.5.6 caerEventPacketCopyOnlyValidEvents()**

```
static caerEventPacketHeader caerEventPacketCopyOnlyValidEvents (  
    caerEventPacketHeaderConst packet ) [inline], [static]
```

Make a copy of an event packet, sized down to only include the currently valid events (eventValid), and discarding everything else.

**Parameters**

<i>packet</i>	an event packet to copy.
---------------	--------------------------

**Returns**

a copy of an event packet, containing only valid events.

**4.8.5.7 caerEventPacketEquals()**

```
static bool caerEventPacketEquals (  
    caerEventPacketHeaderConst firstPacket,  
    caerEventPacketHeaderConst secondPacket ) [inline], [static]
```

Verify if two event packets are equal. This means that the header and all events are equal.

**Parameters**

<i>firstPacket</i>	an event packet to be compared.
<i>secondPacket</i>	the other event packet to compare against.

**Returns**

true if both are the same, false otherwise.

**4.8.5.8 caerEventPacketGetDataSize()**

```
static int64_t caerEventPacketGetDataSize (  
    caerEventPacketHeaderConst header ) [inline], [static]
```

Get the data size of an event packet, in bytes. This is only the size of the data portion, excluding the header.

**Parameters**

<i>header</i>	a valid EventPacket header pointer. Cannot be NULL.
---------------	---

**Returns**

the event packet data size in bytes.

**4.8.5.9 caerEventPacketGetSize()**

```
static int64_t caerEventPacketGetSize (
    caerEventPacketHeaderConst header ) [inline], [static]
```

Get the full size of an event packet, in bytes. This includes both the header and the data portion.

**Parameters**

<i>header</i>	a valid EventPacket header pointer. Cannot be NULL.
---------------	---

**Returns**

the event packet size in bytes.

**4.8.5.10 caerEventPacketGrow()**

```
static caerEventPacketHeader caerEventPacketGrow (
    caerEventPacketHeader packet,
    int32_t newEventCapacity ) [inline], [static]
```

Grows an event packet. This only supports strictly increasing the size of a packet. For a more flexible resize operation, see [caerEventPacketResize\(\)](#). Use `free()` to reclaim this memory afterwards.

**Parameters**

<i>packet</i>	the current event packet.
<i>newEventCapacity</i>	the new maximum number of events this packet can hold. Cannot be zero.

**Returns**

a valid event packet handle or NULL on error. On success, the old packet handle is to be considered invalid and not to be used anymore. On failure, the old packet handle is not touched in any way.

#### 4.8.5.11 caerEventPacketHeaderGetEventCapacity()

```
static int32_t caerEventPacketHeaderGetEventCapacity (  
    caerEventPacketHeaderConst header ) [inline], [static]
```

Get the maximum number of events this packet can store.

##### Parameters

<i>header</i>	a valid EventPacket header pointer. Cannot be NULL.
---------------	---

##### Returns

the number of events this packet can hold.

#### 4.8.5.12 caerEventPacketHeaderGetEventNumber()

```
static int32_t caerEventPacketHeaderGetEventNumber (  
    caerEventPacketHeaderConst header ) [inline], [static]
```

Get the number of events currently stored in this packet, considering both valid and invalid events.

##### Parameters

<i>header</i>	a valid EventPacket header pointer. Cannot be NULL.
---------------	---

##### Returns

the number of events in this packet.

#### 4.8.5.13 caerEventPacketHeaderGetEventSize()

```
static int32_t caerEventPacketHeaderGetEventSize (  
    caerEventPacketHeaderConst header ) [inline], [static]
```

Get the size of a single event, in bytes. All events inside an event packet always have the same size.

##### Parameters

<i>header</i>	a valid EventPacket header pointer. Cannot be NULL.
---------------	---

##### Returns

the event size in bytes.

#### 4.8.5.14 caerEventPacketHeaderGetEventSource()

```
static int16_t caerEventPacketHeaderGetEventSource (  
    caerEventPacketHeaderConst header ) [inline], [static]
```

Get the numerical event source ID, representing the event source that generated all the events present in this packet.

##### Parameters

<i>header</i>	a valid EventPacket header pointer. Cannot be NULL.
---------------	---

##### Returns

the numerical event source ID.

#### 4.8.5.15 caerEventPacketHeaderGetEventTSOffset()

```
static int32_t caerEventPacketHeaderGetEventTSOffset (  
    caerEventPacketHeaderConst header ) [inline], [static]
```

Get the offset, in bytes, to where the field with the main 32 bit timestamp is stored. This is useful for generic access to the timestamp field, given that different event types might have it at different offsets or might even have multiple timestamps, in which case this offset references the 'main' timestamp, the most representative one.

##### Parameters

<i>header</i>	a valid EventPacket header pointer. Cannot be NULL.
---------------	---

##### Returns

the event timestamp offset in bytes.

#### 4.8.5.16 caerEventPacketHeaderGetEventTSOverflow()

```
static int32_t caerEventPacketHeaderGetEventTSOverflow (  
    caerEventPacketHeaderConst header ) [inline], [static]
```

Get the 32 bit timestamp overflow counter (in microseconds). This is per-packet and is used to generate a 64 bit timestamp that never wraps around. Since timestamps wrap around after some time, being only 31 bit (32 bit signed int), another timestamp at the packet level provides another 31 bit (32 bit signed int), to enable the generation of a 62 bit (64 bit signed int) microsecond timestamp which is guaranteed to never wrap around (in the next 146'138 years at least).

## Parameters

<i>header</i>	a valid EventPacket header pointer. Cannot be NULL.
---------------	---

## Returns

the packet-level timestamp overflow counter, in microseconds.

**4.8.5.17 caerEventPacketHeaderGetEventType()**

```
static int16_t caerEventPacketHeaderGetEventType (  
    caerEventPacketHeaderConst header ) [inline], [static]
```

Return the numerical event type ID, representing the event type this EventPacket is containing.

## Parameters

<i>header</i>	a valid EventPacket header pointer. Cannot be NULL.
---------------	---

## Returns

the numerical event type (see 'enum caer\_default\_event\_types').

**4.8.5.18 caerEventPacketHeaderGetEventValid()**

```
static int32_t caerEventPacketHeaderGetEventValid (  
    caerEventPacketHeaderConst header ) [inline], [static]
```

Get the number of valid events in this packet, disregarding invalid ones (where the invalid mark is set).

## Parameters

<i>header</i>	a valid EventPacket header pointer. Cannot be NULL.
---------------	---

## Returns

the number of valid events in this packet.

**4.8.5.19 caerEventPacketHeaderSetEventCapacity()**

```
static void caerEventPacketHeaderSetEventCapacity (  
    caerEventPacketHeader header,  
    int32_t eventsCapacity ) [inline], [static]
```

Set the maximum number of events this packet can store. This is determined at packet allocation time and should not be changed during the life-time of the packet.

#### Parameters

<i>header</i>	a valid EventPacket header pointer. Cannot be NULL.
<i>eventsCapacity</i>	the number of events this packet can hold.

#### 4.8.5.20 caerEventPacketHeaderSetEventNumber()

```
static void caerEventPacketHeaderSetEventNumber (
    caerEventPacketHeader header,
    int32_t eventsNumber ) [inline], [static]
```

Set the number of events currently stored in this packet, considering both valid and invalid events.

#### Parameters

<i>header</i>	a valid EventPacket header pointer. Cannot be NULL.
<i>eventsNumber</i>	the number of events in this packet.

#### 4.8.5.21 caerEventPacketHeaderSetEventSize()

```
static void caerEventPacketHeaderSetEventSize (
    caerEventPacketHeader header,
    int32_t eventSize ) [inline], [static]
```

Set the size of a single event, in bytes. All events inside an event packet always have the same size.

#### Parameters

<i>header</i>	a valid EventPacket header pointer. Cannot be NULL.
<i>eventSize</i>	the event size in bytes.

#### 4.8.5.22 caerEventPacketHeaderSetEventSource()

```
static void caerEventPacketHeaderSetEventSource (
    caerEventPacketHeader header,
    int16_t eventSource ) [inline], [static]
```

Set the numerical event source ID, representing the event source that generated all the events present in this packet. This ID should be unique at least within a process, if not within the whole system, to guarantee correct identification of who generated an event later on.



## Parameters

<i>header</i>	a valid EventPacket header pointer. Cannot be NULL.
<i>eventSource</i>	the numerical event source ID.

## 4.8.5.23 caerEventPacketHeaderSetEventTSOffset()

```
static void caerEventPacketHeaderSetEventTSOffset (
    caerEventPacketHeader header,
    int32_t eventTSOffset ) [inline], [static]
```

Set the offset, in bytes, to where the field with the main 32 bit timestamp is stored. This is useful for generic access to the timestamp field, given that different event types might have it at different offsets or might even have multiple timestamps, in which case this offset references the 'main' timestamp, the most representative one.

## Parameters

<i>header</i>	a valid EventPacket header pointer. Cannot be NULL.
<i>eventTSOffset</i>	the event timestamp offset in bytes.

## 4.8.5.24 caerEventPacketHeaderSetEventTSOverflow()

```
static void caerEventPacketHeaderSetEventTSOverflow (
    caerEventPacketHeader header,
    int32_t eventTSOverflow ) [inline], [static]
```

Set the 32 bit timestamp overflow counter (in microseconds). This is per-packet and is used to generate a 64 bit timestamp that never wraps around. Since timestamps wrap around after some time, being only 31 bit (32 bit signed int), another timestamp at the packet level provides another 31 bit (32 bit signed int), to enable the generation of a 62 bit (64 bit signed int) microsecond timestamp which is guaranteed to never wrap around (in the next 146'138 years at least).

## Parameters

<i>header</i>	a valid EventPacket header pointer. Cannot be NULL.
<i>eventTSOverflow</i>	the packet-level timestamp overflow counter, in microseconds.

## 4.8.5.25 caerEventPacketHeaderSetEventType()

```
static void caerEventPacketHeaderSetEventType (
    caerEventPacketHeader header,
    int16_t eventType ) [inline], [static]
```

Set the numerical event type ID, representing the event type this EventPacket will contain. All event types below 100 are reserved for use by libcaer and cAER. DO NOT USE THEM FOR YOUR OWN EVENT TYPES!

#### Parameters

<i>header</i>	a valid EventPacket header pointer. Cannot be NULL.
<i>eventType</i>	the numerical event type (see 'enum caer_default_event_types').

#### 4.8.5.26 caerEventPacketHeaderSetEventValid()

```
static void caerEventPacketHeaderSetEventValid (
    caerEventPacketHeader header,
    int32_t eventsValid ) [inline], [static]
```

Set the number of valid events in this packet, disregarding invalid ones (where the invalid mark is set).

#### Parameters

<i>header</i>	a valid EventPacket header pointer. Cannot be NULL.
<i>eventsValid</i>	the number of valid events in this packet.

#### 4.8.5.27 caerEventPacketResize()

```
static caerEventPacketHeader caerEventPacketResize (
    caerEventPacketHeader packet,
    int32_t newEventCapacity ) [inline], [static]
```

Resize an event packet. First, the packet is cleaned (all invalid events removed), then:

- If the old and new event capacity are equal, nothing else changes.
- If the new capacity is bigger, the packet is enlarged and the new events are initialized to all zeros (invalid).
- If the new capacity is smaller, the packet is truncated at the given point. Use free() to reclaim this memory afterwards.

#### Parameters

<i>packet</i>	the current event packet.
<i>newEventCapacity</i>	the new maximum number of events this packet can hold. Cannot be zero.

#### Returns

a valid event packet handle or NULL on error. On success, the old packet handle is to be considered invalid and not to be used anymore. On failure, the old packet handle is still valid, but will have been cleaned of all invalid events!

## 4.8.5.28 caerGenericEventCopy()

```
static bool caerGenericEventCopy (
    void * eventPtrDestination,
    const void * eventPtrSource,
    caerEventPacketHeaderConst headerPtrDestination,
    caerEventPacketHeaderConst headerPtrSource ) [inline], [static]
```

Copy a given event's content to another location in memory.

## Parameters

<i>eventPtrDestination</i>	a generic pointer to an event to copy to. Cannot be NULL.
<i>eventPtrSource</i>	a generic pointer to an event to copy from. Cannot be NULL.
<i>headerPtrDestination</i>	a valid EventPacket header pointer from the destination packet. Cannot be NULL.
<i>headerPtrSource</i>	a valid EventPacket header pointer from the source packet. Cannot be NULL.

## Returns

true on successful copy, false otherwise.

## 4.8.5.29 caerGenericEventGetEvent()

```
static const void* caerGenericEventGetEvent (
    caerEventPacketHeaderConst headerPtr,
    int32_t n ) [inline], [static]
```

Get a generic pointer to an event, without having to know what event type the packet is containing.

## Parameters

<i>headerPtr</i>	a valid EventPacket header pointer. Cannot be NULL.
<i>n</i>	the index of the returned event. Must be within [0,eventNumber[ bounds.

## Returns

a generic pointer to the requested event. NULL on error. This points to unmodifiable memory, as it should never be used for anything other than read operations, such as [caerGenericEventGetTimestamp\(\)](#). Don't modify the memory, you have no idea what it is! If you do know, just use the proper typed packet functions.

## 4.8.5.30 caerGenericEventGetTimestamp()

```
static int32_t caerGenericEventGetTimestamp (
    const void * eventPtr,
    caerEventPacketHeaderConst headerPtr ) [inline], [static]
```

Get the main 32 bit timestamp for a generic event, without having to know what event type the packet is containing.

#### Parameters

<i>eventPtr</i>	a generic pointer to an event. Cannot be NULL.
<i>headerPtr</i>	a valid EventPacket header pointer. Cannot be NULL.

#### Returns

the main 32 bit timestamp of this event.

#### 4.8.5.31 caerGenericEventGetTimestamp64()

```
static int64_t caerGenericEventGetTimestamp64 (  
    const void * eventPtr,  
    caerEventPacketHeaderConst headerPtr ) [inline], [static]
```

Get the main 64 bit timestamp for a generic event, without having to know what event type the packet is containing. This takes the per-packet timestamp into account too, generating a timestamp that doesn't suffer from overflow problems.

#### Parameters

<i>eventPtr</i>	a generic pointer to an event. Cannot be NULL.
<i>headerPtr</i>	a valid EventPacket header pointer. Cannot be NULL.

#### Returns

the main 64 bit timestamp of this event.

#### 4.8.5.32 caerGenericEventIsValid()

```
static bool caerGenericEventIsValid (  
    const void * eventPtr ) [inline], [static]
```

Check if the given generic event is valid or not.

#### Parameters

<i>eventPtr</i>	a generic pointer to an event. Cannot be NULL.
-----------------	--

#### Returns

true if the event is valid, false otherwise.

## 4.8.5.33 PACKED\_STRUCT()

```
PACKED_STRUCT (
    struct caer_event_packet_header { int16_t eventType;int16_t eventSource;int32_t
eventSize;int32_t eventTSOffset;int32_t eventTSOverflow;int32_t eventCapacity;int32_t event←
Number;int32_t eventValid;} )
```

EventPacket header data structure definition. The size, also defined in CAER\_EVENT\_PACKET\_HEADER\_SIZE, must always be constant. The header is common to all types of event packets and is always the very first member of an event packet data structure. Signed integers are used for compatibility with languages that do not have unsigned ones, such as Java.

## 4.9 events/config.h File Reference

```
#include "common.h"
```

## Macros

- `#define CAER_CONFIGURATION_ITERATOR_ALL_START(CONFIGURATION_PACKET)`
  - `#define CAER_CONFIGURATION_CONST_ITERATOR_ALL_START(CONFIGURATION_PACKET)`
  - `#define CAER_CONFIGURATION_ITERATOR_ALL_END }`
  - `#define CAER_CONFIGURATION_ITERATOR_VALID_START(CONFIGURATION_PACKET)`
  - `#define CAER_CONFIGURATION_CONST_ITERATOR_VALID_START(CONFIGURATION_PACKET)`
  - `#define CAER_CONFIGURATION_ITERATOR_VALID_END }`
  - `#define CAER_CONFIGURATION_REVERSE_ITERATOR_ALL_START(CONFIGURATION_PACKET)`
  - `#define CAER_CONFIGURATION_CONST_REVERSE_ITERATOR_ALL_START(CONFIGURATION_P←`  
`ACKET)`
  - `#define CAER_CONFIGURATION_REVERSE_ITERATOR_ALL_END }`
  - `#define CAER_CONFIGURATION_REVERSE_ITERATOR_VALID_START(CONFIGURATION_PACKET)`
  - `#define CAER_CONFIGURATION_CONST_REVERSE_ITERATOR_VALID_START(CONFIGURATION_←`  
`PACKET)`
  - `#define CAER_CONFIGURATION_REVERSE_ITERATOR_VALID_END }`
- 
- `#define CONFIG_MODULE_ADDR_SHIFT 1`
  - `#define CONFIG_MODULE_ADDR_MASK 0x0000007F`

## Typedefs

- `typedef struct caer_configuration_event * caerConfigurationEvent`
- `typedef const struct caer_configuration_event * caerConfigurationEventConst`
- `typedef struct caer_configuration_event_packet * caerConfigurationEventPacket`
- `typedef const struct caer_configuration_event_packet * caerConfigurationEventPacketConst`

## Functions

- [PACKED\\_STRUCT](#) (struct caer\_configuration\_event { uint8\_t moduleAddress;uint8\_t parameter↵ Address;uint32\_t parameter;int32\_t timestamp;})
- [PACKED\\_STRUCT](#) (struct caer\_configuration\_event\_packet { struct caer\_event\_packet\_header packet↵ Header;struct caer\_configuration\_event events[;];})
- [caerConfigurationEventPacket caerConfigurationEventPacketAllocate](#) (int32\_t eventCapacity, int16\_t event↵ Source, int32\_t tsOverflow)
- static [caerConfigurationEvent caerConfigurationEventPacketGetEvent](#) ([caerConfigurationEventPacket](#) packet, int32\_t n)
- static caerConfigurationEventConst [caerConfigurationEventPacketGetEventConst](#) (caerConfiguration↵ EventPacketConst packet, int32\_t n)
- static int32\_t [caerConfigurationEventGetTimestamp](#) (caerConfigurationEventConst event)
- static int64\_t [caerConfigurationEventGetTimestamp64](#) (caerConfigurationEventConst event, caer↵ ConfigurationEventPacketConst packet)
- static void [caerConfigurationEventSetTimestamp](#) (caerConfigurationEvent event, int32\_t timestamp)
- static bool [caerConfigurationEventsIsValid](#) (caerConfigurationEventConst event)
- static void [caerConfigurationEventValidate](#) (caerConfigurationEvent event, [caerConfigurationEventPacket](#) packet)
- static void [caerConfigurationEventInvalidate](#) (caerConfigurationEvent event, [caerConfigurationEventPacket](#) packet)
- static uint8\_t [caerConfigurationEventGetModuleAddress](#) (caerConfigurationEventConst event)
- static void [caerConfigurationEventSetModuleAddress](#) (caerConfigurationEvent event, uint8\_t module↵ Address)
- static uint8\_t [caerConfigurationEventGetParameterAddress](#) (caerConfigurationEventConst event)
- static void [caerConfigurationEventSetParameterAddress](#) (caerConfigurationEvent event, uint8\_t parameter↵ Address)
- static uint32\_t [caerConfigurationEventGetParameter](#) (caerConfigurationEventConst event)
- static void [caerConfigurationEventSetParameter](#) (caerConfigurationEvent event, uint32\_t parameter)

### 4.9.1 Detailed Description

Configuration Events format definition and handling functions. This event contains information about the current configuration of the device. By having configuration as a standardized event format, it becomes host-software agnostic, and it also becomes part of the event stream, enabling easy tracking of changes through time, by putting them into the event stream at the moment they happen. While the resolution of the timestamps for these events is in microseconds for compatibility with all other event types, the precision is in the order of ~1-20 milliseconds, given that these events are generated and injected on the host-side.

### 4.9.2 Macro Definition Documentation

#### 4.9.2.1 CAER\_CONFIGURATION\_CONST\_ITERATOR\_ALL\_START

```
#define CAER_CONFIGURATION_CONST_ITERATOR_ALL_START (
    CONFIGURATION_PACKET )
```

**Value:**

```
for (int32_t caerConfigurationIteratorCounter = 0; \
    caerConfigurationIteratorCounter < caerEventPacketHeaderGetEventNumber
    (&(CONFIGURATION_PACKET)->packetHeader); \
    caerConfigurationIteratorCounter++) { \
    caerConfigurationEventConst caerConfigurationIteratorElement =
    caerConfigurationEventPacketGetEventConst (CONFIGURATION_PACKET,
    caerConfigurationIteratorCounter);
```

Const-Iterator over all configuration events in a packet. Returns the current index in the 'caerConfigurationIteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerConfigurationIteratorElement' variable of type caerConfigurationEventConst.

CONFIGURATION\_PACKET: a valid ConfigurationEventPacket pointer. Cannot be NULL.

#### 4.9.2.2 CAER\_CONFIGURATION\_CONST\_ITERATOR\_VALID\_START

```
#define CAER_CONFIGURATION_CONST_ITERATOR_VALID_START(
    CONFIGURATION_PACKET )
```

**Value:**

```
for (int32_t caerConfigurationIteratorCounter = 0; \
    caerConfigurationIteratorCounter < caerEventPacketHeaderGetEventNumber
    (&(CONFIGURATION_PACKET)->packetHeader); \
    caerConfigurationIteratorCounter++) { \
    caerConfigurationEventConst caerConfigurationIteratorElement =
    caerConfigurationEventPacketGetEventConst (CONFIGURATION_PACKET,
    caerConfigurationIteratorCounter); \
    if (!caerConfigurationEventIsValid(caerConfigurationIteratorElement))
    { continue; }
```

Const-Iterator over only the valid configuration events in a packet. Returns the current index in the 'caerConfigurationIteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerConfigurationIteratorElement' variable of type caerConfigurationEventConst.

CONFIGURATION\_PACKET: a valid ConfigurationEventPacket pointer. Cannot be NULL.

#### 4.9.2.3 CAER\_CONFIGURATION\_CONST\_REVERSE\_ITERATOR\_ALL\_START

```
#define CAER_CONFIGURATION_CONST_REVERSE_ITERATOR_ALL_START(
    CONFIGURATION_PACKET )
```

**Value:**

```
for (int32_t caerConfigurationIteratorCounter =
    caerEventPacketHeaderGetEventNumber (&(CONFIGURATION_PACKET)->
    packetHeader) - 1; \
    caerConfigurationIteratorCounter >= 0; \
    caerConfigurationIteratorCounter--) { \
    caerConfigurationEventConst caerConfigurationIteratorElement =
    caerConfigurationEventPacketGetEventConst (CONFIGURATION_PACKET,
    caerConfigurationIteratorCounter);
```

Const-Reverse iterator over all configuration events in a packet. Returns the current index in the 'caerConfigurationIteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerConfigurationIteratorElement' variable of type caerConfigurationEventConst.

CONFIGURATION\_PACKET: a valid ConfigurationEventPacket pointer. Cannot be NULL.

#### 4.9.2.4 CAER\_CONFIGURATION\_CONST\_REVERSE\_ITERATOR\_VALID\_START

```
#define CAER_CONFIGURATION_CONST_REVERSE_ITERATOR_VALID_START(
    CONFIGURATION_PACKET )
```

**Value:**

```
for (int32_t caerConfigurationIteratorCounter =
    caerEventPacketHeaderGetEventNumber(&(CONFIGURATION_PACKET)->
    packetHeader) - 1; \
    caerConfigurationIteratorCounter >= 0; \
    caerConfigurationIteratorCounter--) { \
    caerConfigurationEventConst caerConfigurationIteratorElement =
    caerConfigurationEventPacketGetEventConst(CONFIGURATION_PACKET,
    caerConfigurationIteratorCounter); \
    if (!caerConfigurationEventIsValid(caerConfigurationIteratorElement))
    { continue; }
```

Const-Reverse iterator over only the valid configuration events in a packet. Returns the current index in the 'caerConfigurationIteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerConfigurationIteratorElement' variable of type caerConfigurationEventConst.

CONFIGURATION\_PACKET: a valid ConfigurationEventPacket pointer. Cannot be NULL.

#### 4.9.2.5 CAER\_CONFIGURATION\_ITERATOR\_ALL\_END

```
#define CAER_CONFIGURATION_ITERATOR_ALL_END }
```

Iterator close statement.

#### 4.9.2.6 CAER\_CONFIGURATION\_ITERATOR\_ALL\_START

```
#define CAER_CONFIGURATION_ITERATOR_ALL_START(
    CONFIGURATION_PACKET )
```

**Value:**

```
for (int32_t caerConfigurationIteratorCounter = 0; \
    caerConfigurationIteratorCounter < caerEventPacketHeaderGetEventNumber
    (&(CONFIGURATION_PACKET)->packetHeader); \
    caerConfigurationIteratorCounter++) { \
    caerConfigurationEvent caerConfigurationIteratorElement =
    caerConfigurationEventPacketGetEvent(CONFIGURATION_PACKET,
    caerConfigurationIteratorCounter);
```

Iterator over all configuration events in a packet. Returns the current index in the 'caerConfigurationIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerConfigurationIteratorElement' variable of type caerConfigurationEvent.

CONFIGURATION\_PACKET: a valid ConfigurationEventPacket pointer. Cannot be NULL.

#### 4.9.2.7 CAER\_CONFIGURATION\_ITERATOR\_VALID\_END

```
#define CAER_CONFIGURATION_ITERATOR_VALID_END }
```

Iterator close statement.



## 4.9.2.8 CAER\_CONFIGURATION\_ITERATOR\_VALID\_START

```
#define CAER_CONFIGURATION_ITERATOR_VALID_START (
    CONFIGURATION_PACKET )
```

**Value:**

```
for (int32_t caerConfigurationIteratorCounter = 0; \
    caerConfigurationIteratorCounter < caerEventPacketHeaderGetEventNumber
    (&(CONFIGURATION_PACKET)->packetHeader); \
    caerConfigurationIteratorCounter++) { \
    caerConfigurationEvent caerConfigurationIteratorElement =
    caerConfigurationEventPacketGetEvent(CONFIGURATION_PACKET,
    caerConfigurationIteratorCounter); \
    if (!caerConfigurationEventIsValid(caerConfigurationIteratorElement))
    { continue; }
```

Iterator over only the valid configuration events in a packet. Returns the current index in the 'caerConfigurationIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerConfigurationIteratorElement' variable of type caerConfigurationEvent.

CONFIGURATION\_PACKET: a valid ConfigurationEventPacket pointer. Cannot be NULL.

## 4.9.2.9 CAER\_CONFIGURATION\_REVERSE\_ITERATOR\_ALL\_END

```
#define CAER_CONFIGURATION_REVERSE_ITERATOR_ALL_END }
```

Reverse iterator close statement.

## 4.9.2.10 CAER\_CONFIGURATION\_REVERSE\_ITERATOR\_ALL\_START

```
#define CAER_CONFIGURATION_REVERSE_ITERATOR_ALL_START (
    CONFIGURATION_PACKET )
```

**Value:**

```
for (int32_t caerConfigurationIteratorCounter =
    caerEventPacketHeaderGetEventNumber(&(CONFIGURATION_PACKET)->
    packetHeader) - 1; \
    caerConfigurationIteratorCounter >= 0; \
    caerConfigurationIteratorCounter--) { \
    caerConfigurationEvent caerConfigurationIteratorElement =
    caerConfigurationEventPacketGetEvent(CONFIGURATION_PACKET,
    caerConfigurationIteratorCounter);
```

Reverse iterator over all configuration events in a packet. Returns the current index in the 'caerConfigurationIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerConfigurationIteratorElement' variable of type caerConfigurationEvent.

CONFIGURATION\_PACKET: a valid ConfigurationEventPacket pointer. Cannot be NULL.

## 4.9.2.11 CAER\_CONFIGURATION\_REVERSE\_ITERATOR\_VALID\_END

```
#define CAER_CONFIGURATION_REVERSE_ITERATOR_VALID_END }
```

Reverse iterator close statement.

#### 4.9.2.12 CAER\_CONFIGURATION\_REVERSE\_ITERATOR\_VALID\_START

```
#define CAER_CONFIGURATION_REVERSE_ITERATOR_VALID_START (
    CONFIGURATION_PACKET )
```

**Value:**

```
for (int32_t caerConfigurationIteratorCounter =
    caerEventPacketHeaderGetEventNumber (& (CONFIGURATION_PACKET) ->
    packetHeader) - 1; \
    caerConfigurationIteratorCounter >= 0; \
    caerConfigurationIteratorCounter--) { \
    caerConfigurationEvent caerConfigurationIteratorElement =
    caerConfigurationEventPacketGetEvent (CONFIGURATION_PACKET,
    caerConfigurationIteratorCounter); \
    if (!caerConfigurationEventIsValid (caerConfigurationIteratorElement))
    { continue; }
```

Reverse iterator over only the valid configuration events in a packet. Returns the current index in the 'caerConfigurationIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerConfigurationIteratorElement' variable of type caerConfigurationEvent.

CONFIGURATION\_PACKET: a valid ConfigurationEventPacket pointer. Cannot be NULL.

#### 4.9.2.13 CONFIG\_MODULE\_ADDR\_MASK

```
#define CONFIG_MODULE_ADDR_MASK 0x0000007F
```

Shift and mask values for the module address. Module address is only 7 bits, since the eighth bit is used device-side to differentiate reads from writes. Here we can just re-use it for the validity mark.

#### 4.9.2.14 CONFIG\_MODULE\_ADDR\_SHIFT

```
#define CONFIG_MODULE_ADDR_SHIFT 1
```

Shift and mask values for the module address. Module address is only 7 bits, since the eighth bit is used device-side to differentiate reads from writes. Here we can just re-use it for the validity mark.

### 4.9.3 Typedef Documentation

#### 4.9.3.1 caerConfigurationEvent

```
typedef struct caer_configuration_event* caerConfigurationEvent
```

Type for pointer to configuration event data structure.

#### 4.9.3.2 caerConfigurationEventPacket

```
typedef struct caer_configuration_event_packet* caerConfigurationEventPacket
```

Type for pointer to configuration event packet data structure.

## 4.9.4 Function Documentation

### 4.9.4.1 caerConfigurationEventGetModuleAddress()

```
static uint8_t caerConfigurationEventGetModuleAddress (  
    caerConfigurationEventConst event ) [inline], [static]
```

Get the configuration event's module address.

#### Parameters

<i>event</i>	a valid ConfigurationEvent pointer. Cannot be NULL.
--------------	---

#### Returns

configuration module address.

### 4.9.4.2 caerConfigurationEventGetParameter()

```
static uint32_t caerConfigurationEventGetParameter (  
    caerConfigurationEventConst event ) [inline], [static]
```

Get the configuration event's parameter.

#### Parameters

<i>event</i>	a valid ConfigurationEvent pointer. Cannot be NULL.
--------------	---

#### Returns

configuration parameter.

### 4.9.4.3 caerConfigurationEventGetParameterAddress()

```
static uint8_t caerConfigurationEventGetParameterAddress (  
    caerConfigurationEventConst event ) [inline], [static]
```

Get the configuration event's parameter address.

#### Parameters

<i>event</i>	a valid ConfigurationEvent pointer. Cannot be NULL.
--------------	---

**Returns**

configuration parameter address.

**4.9.4.4 caerConfigurationEventGetTimestamp()**

```
static int32_t caerConfigurationEventGetTimestamp (
    caerConfigurationEventConst event ) [inline], [static]
```

Get the 32bit event timestamp, in microseconds. Be aware that this wraps around! You can either ignore this fact, or handle the special 'TIMESTAMP\_WRAP' event that is generated when this happens, or use the 64bit timestamp which never wraps around. See '[caerEventPacketHeaderGetEventTSOverflow\(\)](#)' documentation for more details on the 64bit timestamp.

**Parameters**

<i>event</i>	a valid ConfigurationEvent pointer. Cannot be NULL.
--------------	---

**Returns**

this event's 32bit microsecond timestamp.

**4.9.4.5 caerConfigurationEventGetTimestamp64()**

```
static int64_t caerConfigurationEventGetTimestamp64 (
    caerConfigurationEventConst event,
    caerConfigurationEventPacketConst packet ) [inline], [static]
```

Get the 64bit event timestamp, in microseconds. See '[caerEventPacketHeaderGetEventTSOverflow\(\)](#)' documentation for more details on the 64bit timestamp.

**Parameters**

<i>event</i>	a valid ConfigurationEvent pointer. Cannot be NULL.
<i>packet</i>	the ConfigurationEventPacket pointer for the packet containing this event. Cannot be NULL.

**Returns**

this event's 64bit microsecond timestamp.

**4.9.4.6 caerConfigurationEventInvalidate()**

```
static void caerConfigurationEventInvalidate (
    caerConfigurationEvent event,
    caerConfigurationEventPacket packet ) [inline], [static]
```

Invalidate the current event by setting its valid bit to false and decreasing the number of valid events held in the packet. Only works with events that are already valid!

**Parameters**

<i>event</i>	a valid ConfigurationEvent pointer. Cannot be NULL.
<i>packet</i>	the ConfigurationEventPacket pointer for the packet containing this event. Cannot be NULL.

**4.9.4.7 caerConfigurationEventIsValid()**

```
static bool caerConfigurationEventIsValid (
    caerConfigurationEventConst event ) [inline], [static]
```

Check if this configuration event is valid.

**Parameters**

<i>event</i>	a valid ConfigurationEvent pointer. Cannot be NULL.
--------------	---

**Returns**

true if valid, false if not.

**4.9.4.8 caerConfigurationEventPacketAllocate()**

```
caerConfigurationEventPacket caerConfigurationEventPacketAllocate (
    int32_t eventCapacity,
    int16_t eventSource,
    int32_t tsOverflow )
```

Allocate a new configuration events packet. Use free() to reclaim this memory.

**Parameters**

<i>eventCapacity</i>	the maximum number of events this packet will hold.
<i>eventSource</i>	the unique ID representing the source/generator of this packet.
<i>tsOverflow</i>	the current timestamp overflow counter value for this packet.

**Returns**

a valid ConfigurationEventPacket handle or NULL on error.

#### 4.9.4.9 caerConfigurationEventPacketGetEvent()

```
static caerConfigurationEvent caerConfigurationEventPacketGetEvent (
    caerConfigurationEventPacket packet,
    int32_t n ) [inline], [static]
```

Get the configuration event at the given index from the event packet.

##### Parameters

<i>packet</i>	a valid ConfigurationEventPacket pointer. Cannot be NULL.
<i>n</i>	the index of the returned event. Must be within [0,eventCapacity[ bounds.

##### Returns

the requested configuration event. NULL on error.

#### 4.9.4.10 caerConfigurationEventPacketGetEventConst()

```
static caerConfigurationEventConst caerConfigurationEventPacketGetEventConst (
    caerConfigurationEventPacketConst packet,
    int32_t n ) [inline], [static]
```

Get the configuration event at the given index from the event packet. This is a read-only event, do not change its contents in any way!

##### Parameters

<i>packet</i>	a valid ConfigurationEventPacket pointer. Cannot be NULL.
<i>n</i>	the index of the returned event. Must be within [0,eventCapacity[ bounds.

##### Returns

the requested read-only configuration event. NULL on error.

#### 4.9.4.11 caerConfigurationEventSetModuleAddress()

```
static void caerConfigurationEventSetModuleAddress (
    caerConfigurationEvent event,
    uint8_t moduleAddress ) [inline], [static]
```

Set the configuration event's module address.

## Parameters

<i>event</i>	a valid ConfigurationEvent pointer. Cannot be NULL.
<i>moduleAddress</i>	configuration module address.

## 4.9.4.12 caerConfigurationEventSetParameter()

```
static void caerConfigurationEventSetParameter (
    caerConfigurationEvent event,
    uint32_t parameter ) [inline], [static]
```

Set the configuration event's parameter.

## Parameters

<i>event</i>	a valid ConfigurationEvent pointer. Cannot be NULL.
<i>parameter</i>	configuration parameter.

## 4.9.4.13 caerConfigurationEventSetParameterAddress()

```
static void caerConfigurationEventSetParameterAddress (
    caerConfigurationEvent event,
    uint8_t parameterAddress ) [inline], [static]
```

Set the configuration event's parameter address.

## Parameters

<i>event</i>	a valid ConfigurationEvent pointer. Cannot be NULL.
<i>parameterAddress</i>	configuration parameter address.

## 4.9.4.14 caerConfigurationEventSetTimestamp()

```
static void caerConfigurationEventSetTimestamp (
    caerConfigurationEvent event,
    int32_t timestamp ) [inline], [static]
```

Set the 32bit event timestamp, the value has to be in microseconds.

## Parameters

<i>event</i>	a valid ConfigurationEvent pointer. Cannot be NULL.
<i>timestamp</i>	a positive 32bit microsecond timestamp.

#### 4.9.4.15 caerConfigurationEventValidate()

```
static void caerConfigurationEventValidate (
    caerConfigurationEvent event,
    caerConfigurationEventPacket packet ) [inline], [static]
```

Validate the current event by setting its valid bit to true and increasing the event packet's event count and valid event count. Only works on events that are invalid. DO NOT CALL THIS AFTER HAVING PREVIOUSLY ALREADY INVALIDATED THIS EVENT, the total count will be incorrect.

##### Parameters

<i>event</i>	a valid ConfigurationEvent pointer. Cannot be NULL.
<i>packet</i>	the ConfigurationEventPacket pointer for the packet containing this event. Cannot be NULL.

#### 4.9.4.16 PACKED\_STRUCT() [1/2]

```
PACKED_STRUCT (
    struct caer_configuration_event { uint8_t moduleAddress;uint8_t parameterAddress;uint32_t parameter;int32_t timestamp;} )
```

Configuration event data structure definition. This contains the actual configuration module address, the parameter address and the actual parameter content, as well as the 32 bit event timestamp. Signed integers are used for fields that are to be interpreted directly, for compatibility with languages that do not have unsigned integer types, such as Java.

#### 4.9.4.17 PACKED\_STRUCT() [2/2]

```
PACKED_STRUCT (
    struct caer_configuration_event_packet { struct caer_event_packet_header packetHeader;struct caer_configuration_event events[];} )
```

Configuration event packet data structure definition. EventPackets are always made up of the common packet header, followed by 'eventCapacity' events. Everything has to be in one contiguous memory block.

## 4.10 events/ear.h File Reference

```
#include "common.h"
```



## Macros

- #define [CAER\\_EAR\\_ITERATOR\\_ALL\\_START](#)(EAR\_PACKET)
  - #define [CAER\\_EAR\\_CONST\\_ITERATOR\\_ALL\\_START](#)(EAR\_PACKET)
  - #define [CAER\\_EAR\\_ITERATOR\\_ALL\\_END](#) }
  - #define [CAER\\_EAR\\_ITERATOR\\_VALID\\_START](#)(EAR\_PACKET)
  - #define [CAER\\_EAR\\_CONST\\_ITERATOR\\_VALID\\_START](#)(EAR\_PACKET)
  - #define [CAER\\_EAR\\_ITERATOR\\_VALID\\_END](#) }
  - #define [CAER\\_EAR\\_REVERSE\\_ITERATOR\\_ALL\\_START](#)(EAR\_PACKET)
  - #define [CAER\\_EAR\\_CONST\\_REVERSE\\_ITERATOR\\_ALL\\_START](#)(EAR\_PACKET)
  - #define [CAER\\_EAR\\_REVERSE\\_ITERATOR\\_ALL\\_END](#) }
  - #define [CAER\\_EAR\\_REVERSE\\_ITERATOR\\_VALID\\_START](#)(EAR\_PACKET)
  - #define [CAER\\_EAR\\_CONST\\_REVERSE\\_ITERATOR\\_VALID\\_START](#)(EAR\_PACKET)
  - #define [CAER\\_EAR\\_REVERSE\\_ITERATOR\\_VALID\\_END](#) }
- 
- #define [EAR\\_SHIFT](#) 1
  - #define [EAR\\_MASK](#) 0x0000000F
  - #define [EAR\\_CHANNEL\\_SHIFT](#) 5
  - #define [EAR\\_CHANNEL\\_MASK](#) 0x000007FF
  - #define [EAR\\_NEURON\\_SHIFT](#) 16
  - #define [EAR\\_NEURON\\_MASK](#) 0x000000FF
  - #define [EAR\\_FILTER\\_SHIFT](#) 24
  - #define [EAR\\_FILTER\\_MASK](#) 0x000000FF

## Typedefs

- typedef struct caer\_ear\_event \* [caerEarEvent](#)
- typedef const struct caer\_ear\_event \* **caerEarEventConst**
- typedef struct caer\_ear\_event\_packet \* [caerEarEventPacket](#)
- typedef const struct caer\_ear\_event\_packet \* **caerEarEventPacketConst**

## Functions

- [PACKED\\_STRUCT](#) (struct caer\_ear\_event { uint32\_t data;int32\_t timestamp;})
- [PACKED\\_STRUCT](#) (struct caer\_ear\_event\_packet { struct caer\_event\_packet\_header packetHeader;struct caer\_ear\_event events[ ];})
- [caerEarEventPacket caerEarEventPacketAllocate](#) (int32\_t eventCapacity, int16\_t eventSource, int32\_t ts↵ Overflow)
- static [caerEarEvent caerEarEventPacketGetEvent](#) ([caerEarEventPacket](#) packet, int32\_t n)
- static caerEarEventConst [caerEarEventPacketGetEventConst](#) (caerEarEventPacketConst packet, int32\_t n)
- static int32\_t [caerEarEventGetTimestamp](#) (caerEarEventConst event)
- static int64\_t [caerEarEventGetTimestamp64](#) (caerEarEventConst event, caerEarEventPacketConst packet)
- static void [caerEarEventSetTimestamp](#) ([caerEarEvent](#) event, int32\_t timestamp)
- static bool [caerEarEventsValid](#) (caerEarEventConst event)
- static void [caerEarEventValidate](#) ([caerEarEvent](#) event, [caerEarEventPacket](#) packet)
- static void [caerEarEventInvalidate](#) ([caerEarEvent](#) event, [caerEarEventPacket](#) packet)
- static uint8\_t [caerEarEventGetEar](#) (caerEarEventConst event)
- static void [caerEarEventSetEar](#) ([caerEarEvent](#) event, uint8\_t ear)
- static uint16\_t [caerEarEventGetChannel](#) (caerEarEventConst event)
- static void [caerEarEventSetChannel](#) ([caerEarEvent](#) event, uint16\_t channel)
- static uint8\_t [caerEarEventGetNeuron](#) (caerEarEventConst event)
- static void [caerEarEventSetNeuron](#) ([caerEarEvent](#) event, uint8\_t neuron)
- static uint8\_t [caerEarEventGetFilter](#) (caerEarEventConst event)
- static void [caerEarEventSetFilter](#) ([caerEarEvent](#) event, uint8\_t filter)

### 4.10.1 Detailed Description

Ear (Cochlea) Events format definition and handling functions. This encodes events from a silicon cochlea chip, containing information about which ear (microphone) generated the event, as well as which channel was involved and additional information on filters and neurons.

### 4.10.2 Macro Definition Documentation

#### 4.10.2.1 CAER\_EAR\_CONST\_ITERATOR\_ALL\_START

```
#define CAER_EAR_CONST_ITERATOR_ALL_START(  
    EAR_PACKET )
```

**Value:**

```
for (int32_t caerEarIteratorCounter = 0; \  
    caerEarIteratorCounter < caerEventPacketHeaderGetEventNumber(&(\  
    EAR_PACKET)->packetHeader); \  
    caerEarIteratorCounter++) { \  
    caerEarEventConst caerEarIteratorElement = \  
    caerEarEventPacketGetEventConst(EAR_PACKET, caerEarIteratorCounter);
```

Const-Iterator over all ear events in a packet. Returns the current index in the 'caerEarIteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerEarIteratorElement' variable of type caerEarEventConst.

EAR\_PACKET: a valid EarEventPacket pointer. Cannot be NULL.

#### 4.10.2.2 CAER\_EAR\_CONST\_ITERATOR\_VALID\_START

```
#define CAER_EAR_CONST_ITERATOR_VALID_START(  
    EAR_PACKET )
```

**Value:**

```
for (int32_t caerEarIteratorCounter = 0; \  
    caerEarIteratorCounter < caerEventPacketHeaderGetEventNumber(&(\  
    EAR_PACKET)->packetHeader); \  
    caerEarIteratorCounter++) { \  
    caerEarEventConst caerEarIteratorElement = \  
    caerEarEventPacketGetEventConst(EAR_PACKET, caerEarIteratorCounter); \  
    if (!caerEarEventIsValid(caerEarIteratorElement)) { continue; }
```

Const-Iterator over only the valid ear events in a packet. Returns the current index in the 'caerEarIteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerEarIteratorElement' variable of type caerEarEventConst.

EAR\_PACKET: a valid EarEventPacket pointer. Cannot be NULL.

## 4.10.2.3 CAER\_EAR\_CONST\_REVERSE\_ITERATOR\_ALL\_START

```
#define CAER_EAR_CONST_REVERSE_ITERATOR_ALL_START(
    EAR_PACKET )
```

**Value:**

```
for (int32_t caerEarIteratorCounter = caerEventPacketHeaderGetEventNumber
    (&(EAR_PACKET)->packetHeader) - 1; \
    caerEarIteratorCounter >= 0; \
    caerEarIteratorCounter--) { \
    caerEarEventConst caerEarIteratorElement =
    caerEarEventPacketGetEventConst(EAR_PACKET, caerEarIteratorCounter);
```

Const-Reverse iterator over all ear events in a packet. Returns the current index in the 'caerEarIteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerEarIteratorElement' variable of type caerEarEventConst.

EAR\_PACKET: a valid EarEventPacket pointer. Cannot be NULL.

## 4.10.2.4 CAER\_EAR\_CONST\_REVERSE\_ITERATOR\_VALID\_START

```
#define CAER_EAR_CONST_REVERSE_ITERATOR_VALID_START(
    EAR_PACKET )
```

**Value:**

```
for (int32_t caerEarIteratorCounter = caerEventPacketHeaderGetEventNumber
    (&(EAR_PACKET)->packetHeader) - 1; \
    caerEarIteratorCounter >= 0; \
    caerEarIteratorCounter--) { \
    caerEarEventConst caerEarIteratorElement =
    caerEarEventPacketGetEventConst(EAR_PACKET, caerEarIteratorCounter); \
    if (!caerEarEventIsValid(caerEarIteratorElement)) { continue; }
```

Const-Reverse iterator over only the valid ear events in a packet. Returns the current index in the 'caerEarIteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerEarIteratorElement' variable of type caerEarEventConst.

EAR\_PACKET: a valid EarEventPacket pointer. Cannot be NULL.

## 4.10.2.5 CAER\_EAR\_ITERATOR\_ALL\_END

```
#define CAER_EAR_ITERATOR_ALL_END }
```

Iterator close statement.

#### 4.10.2.6 CAER\_EAR\_ITERATOR\_ALL\_START

```
#define CAER_EAR_ITERATOR_ALL_START(
    EAR_PACKET )
```

**Value:**

```
for (int32_t caerEarIteratorCounter = 0; \
    caerEarIteratorCounter < caerEventPacketHeaderGetEventNumber(&(
    EAR_PACKET)->packetHeader); \
    caerEarIteratorCounter++) { \
    caerEarEvent caerEarIteratorElement = caerEarEventPacketGetEvent(
    EAR_PACKET, caerEarIteratorCounter);
```

Iterator over all ear events in a packet. Returns the current index in the 'caerEarIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerEarIteratorElement' variable of type caerEarEvent.

EAR\_PACKET: a valid EarEventPacket pointer. Cannot be NULL.

#### 4.10.2.7 CAER\_EAR\_ITERATOR\_VALID\_END

```
#define CAER_EAR_ITERATOR_VALID_END }
```

Iterator close statement.

#### 4.10.2.8 CAER\_EAR\_ITERATOR\_VALID\_START

```
#define CAER_EAR_ITERATOR_VALID_START(
    EAR_PACKET )
```

**Value:**

```
for (int32_t caerEarIteratorCounter = 0; \
    caerEarIteratorCounter < caerEventPacketHeaderGetEventNumber(&(
    EAR_PACKET)->packetHeader); \
    caerEarIteratorCounter++) { \
    caerEarEvent caerEarIteratorElement = caerEarEventPacketGetEvent(
    EAR_PACKET, caerEarIteratorCounter); \
    if (!caerEarEventIsValid(caerEarIteratorElement)) { continue; }
```

Iterator over only the valid ear events in a packet. Returns the current index in the 'caerEarIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerEarIteratorElement' variable of type caerEarEvent.

EAR\_PACKET: a valid EarEventPacket pointer. Cannot be NULL.

#### 4.10.2.9 CAER\_EAR\_REVERSE\_ITERATOR\_ALL\_END

```
#define CAER_EAR_REVERSE_ITERATOR_ALL_END }
```

Reverse iterator close statement.

## 4.10.2.10 CAER\_EAR\_REVERSE\_ITERATOR\_ALL\_START

```
#define CAER_EAR_REVERSE_ITERATOR_ALL_START (
    EAR_PACKET )
```

**Value:**

```
for (int32_t caerEarIteratorCounter = caerEventPacketHeaderGetEventNumber
    (&(EAR_PACKET)->packetHeader) - 1; \
    caerEarIteratorCounter >= 0; \
    caerEarIteratorCounter--) { \
    caerEarEvent caerEarIteratorElement = caerEarEventPacketGetEvent (
    EAR_PACKET, caerEarIteratorCounter);
```

Reverse iterator over all ear events in a packet. Returns the current index in the 'caerEarIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerEarIteratorElement' variable of type caerEarEvent.

EAR\_PACKET: a valid EarEventPacket pointer. Cannot be NULL.

## 4.10.2.11 CAER\_EAR\_REVERSE\_ITERATOR\_VALID\_END

```
#define CAER_EAR_REVERSE_ITERATOR_VALID_END }
```

Reverse iterator close statement.

## 4.10.2.12 CAER\_EAR\_REVERSE\_ITERATOR\_VALID\_START

```
#define CAER_EAR_REVERSE_ITERATOR_VALID_START (
    EAR_PACKET )
```

**Value:**

```
for (int32_t caerEarIteratorCounter = caerEventPacketHeaderGetEventNumber
    (&(EAR_PACKET)->packetHeader) - 1; \
    caerEarIteratorCounter >= 0; \
    caerEarIteratorCounter--) { \
    caerEarEvent caerEarIteratorElement = caerEarEventPacketGetEvent (
    EAR_PACKET, caerEarIteratorCounter); \
    if (!caerEarEventIsValid(caerEarIteratorElement)) { continue; }
```

Reverse iterator over only the valid ear events in a packet. Returns the current index in the 'caerEarIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerEarIteratorElement' variable of type caerEarEvent.

EAR\_PACKET: a valid EarEventPacket pointer. Cannot be NULL.

## 4.10.2.13 EAR\_CHANNEL\_MASK

```
#define EAR_CHANNEL_MASK 0x000007FF
```

Shift and mask values for the ear event values coming from a cochlea: the ear position (up to 16), the channel number (up to 2048), the ganglion (up to 256) and the filter (up to 256). Bit 0 is the valid mark, see 'common.h' for more details.

#### 4.10.2.14 EAR\_CHANNEL\_SHIFT

```
#define EAR_CHANNEL_SHIFT 5
```

Shift and mask values for the ear event values coming from a cochlea: the ear position (up to 16), the channel number (up to 2048), the ganglion (up to 256) and the filter (up to 256). Bit 0 is the valid mark, see '[common.h](#)' for more details.

#### 4.10.2.15 EAR\_FILTER\_MASK

```
#define EAR_FILTER_MASK 0x000000FF
```

Shift and mask values for the ear event values coming from a cochlea: the ear position (up to 16), the channel number (up to 2048), the ganglion (up to 256) and the filter (up to 256). Bit 0 is the valid mark, see '[common.h](#)' for more details.

#### 4.10.2.16 EAR\_FILTER\_SHIFT

```
#define EAR_FILTER_SHIFT 24
```

Shift and mask values for the ear event values coming from a cochlea: the ear position (up to 16), the channel number (up to 2048), the ganglion (up to 256) and the filter (up to 256). Bit 0 is the valid mark, see '[common.h](#)' for more details.

#### 4.10.2.17 EAR\_MASK

```
#define EAR_MASK 0x0000000F
```

Shift and mask values for the ear event values coming from a cochlea: the ear position (up to 16), the channel number (up to 2048), the ganglion (up to 256) and the filter (up to 256). Bit 0 is the valid mark, see '[common.h](#)' for more details.

#### 4.10.2.18 EAR\_NEURON\_MASK

```
#define EAR_NEURON_MASK 0x000000FF
```

Shift and mask values for the ear event values coming from a cochlea: the ear position (up to 16), the channel number (up to 2048), the ganglion (up to 256) and the filter (up to 256). Bit 0 is the valid mark, see '[common.h](#)' for more details.

#### 4.10.2.19 EAR\_NEURON\_SHIFT

```
#define EAR_NEURON_SHIFT 16
```

Shift and mask values for the ear event values coming from a cochlea: the ear position (up to 16), the channel number (up to 2048), the ganglion (up to 256) and the filter (up to 256). Bit 0 is the valid mark, see '[common.h](#)' for more details.

#### 4.10.2.20 EAR\_SHIFT

```
#define EAR_SHIFT 1
```

Shift and mask values for the ear event values coming from a cochlea: the ear position (up to 16), the channel number (up to 2048), the ganglion (up to 256) and the filter (up to 256). Bit 0 is the valid mark, see '[common.h](#)' for more details.

### 4.10.3 Typedef Documentation

#### 4.10.3.1 caerEarEvent

```
typedef struct caer_ear_event* caerEarEvent
```

Type for pointer to ear (cochlea) event data structure.

#### 4.10.3.2 caerEarEventPacket

```
typedef struct caer_ear_event_packet* caerEarEventPacket
```

Type for pointer to ear (cochlea) event packet data structure.

### 4.10.4 Function Documentation

#### 4.10.4.1 caerEarEventGetChannel()

```
static uint16_t caerEarEventGetChannel (  
    caerEarEventConst event ) [inline], [static]
```

Get the channel (frequency band) ID. The channels count from 0 upward, where 0 is the highest frequency channel, while higher numbers are progressively lower frequency channels. This is derived from how the actual human ear works.

##### Parameters

<i>event</i>	a valid EarEvent pointer. Cannot be NULL.
--------------	---

##### Returns

the channel (frequency band) ID.

#### 4.10.4.2 caerEarEventGetEar()

```
static uint8_t caerEarEventGetEar (
    caerEarEventConst event ) [inline], [static]
```

Get the numerical ID of the ear (microphone). Usually, 0 is left, 1 is right for 2 ear cochleas. For 4 ear cochleas, 0 is front left, 1 is front right, 2 is back left and 3 is back right.

##### Parameters

<i>event</i>	a valid EarEvent pointer. Cannot be NULL.
--------------	---

##### Returns

the ear (microphone) ID.

#### 4.10.4.3 caerEarEventGetTimestamp()

```
static int32_t caerEarEventGetTimestamp (
    caerEarEventConst event ) [inline], [static]
```

Get the 32bit event timestamp, in microseconds. Be aware that this wraps around! You can either ignore this fact, or handle the special 'TIMESTAMP\_WRAP' event that is generated when this happens, or use the 64bit timestamp which never wraps around. See '[caerEventPacketHeaderGetEventTSOverflow\(\)](#)' documentation for more details on the 64bit timestamp.

##### Parameters

<i>event</i>	a valid EarEvent pointer. Cannot be NULL.
--------------	---

##### Returns

this event's 32bit microsecond timestamp.

#### 4.10.4.4 caerEarEventGetTimestamp64()

```
static int64_t caerEarEventGetTimestamp64 (
    caerEarEventConst event,
    caerEarEventPacketConst packet ) [inline], [static]
```

Get the 64bit event timestamp, in microseconds. See '[caerEventPacketHeaderGetEventTSOverflow\(\)](#)' documentation for more details on the 64bit timestamp.

##### Parameters

<i>event</i>	a valid EarEvent pointer. Cannot be NULL.
<i>packet</i>	the EarEventPacket pointer for the packet containing this event. Cannot be NULL.



**Returns**

this event's 64bit microsecond timestamp.

**4.10.4.5 caerEarEventInvalidate()**

```
static void caerEarEventInvalidate (
    caerEarEvent event,
    caerEarEventPacket packet ) [inline], [static]
```

Invalidate the current event by setting its valid bit to false and decreasing the number of valid events held in the packet. Only works with events that are already valid!

**Parameters**

<i>event</i>	a valid EarEvent pointer. Cannot be NULL.
<i>packet</i>	the EarEventPacket pointer for the packet containing this event. Cannot be NULL.

**4.10.4.6 caerEarEventIsValid()**

```
static bool caerEarEventIsValid (
    caerEarEventConst event ) [inline], [static]
```

Check if this ear (cochlea) event is valid.

**Parameters**

<i>event</i>	a valid EarEvent pointer. Cannot be NULL.
--------------	---

**Returns**

true if valid, false if not.

**4.10.4.7 caerEarEventPacketAllocate()**

```
caerEarEventPacket caerEarEventPacketAllocate (
    int32_t eventCapacity,
    int16_t eventSource,
    int32_t tsOverflow )
```

Allocate a new ear (cochlea) events packet. Use free() to reclaim this memory.

**Parameters**

<i>eventCapacity</i>	the maximum number of events this packet will hold.
<i>eventSource</i>	the unique ID representing the source/generator of this packet.
<i>tsOverflow</i>	the current timestamp overflow counter value for this packet.

**Returns**

a valid EarEventPacket handle or NULL on error.

**4.10.4.8 caerEarEventPacketGetEvent()**

```
static caerEarEvent caerEarEventPacketGetEvent (
    caerEarEventPacket packet,
    int32_t n ) [inline], [static]
```

Get the ear (cochlea) event at the given index from the event packet.

**Parameters**

<i>packet</i>	a valid EarEventPacket pointer. Cannot be NULL.
<i>n</i>	the index of the returned event. Must be within [0,eventCapacity[ bounds.

**Returns**

the requested ear (cochlea) event. NULL on error.

**4.10.4.9 caerEarEventPacketGetEventConst()**

```
static caerEarEventConst caerEarEventPacketGetEventConst (
    caerEarEventPacketConst packet,
    int32_t n ) [inline], [static]
```

Get the ear (cochlea) event at the given index from the event packet. This is a read-only event, do not change its contents in any way!

**Parameters**

<i>packet</i>	a valid EarEventPacket pointer. Cannot be NULL.
<i>n</i>	the index of the returned event. Must be within [0,eventCapacity[ bounds.

**Returns**

the requested read-only ear (cochlea) event. NULL on error.

**4.10.4.10 caerEarEventSetChannel()**

```
static void caerEarEventSetChannel (
    caerEarEvent event,
    uint16_t channel ) [inline], [static]
```

Set the channel (frequency band) ID. The channels count from 0 upward, where 0 is the highest frequency channel, while higher numbers are progressively lower frequency channels. This is derived from how the actual human ear works.

**Parameters**

<i>event</i>	a valid EarEvent pointer. Cannot be NULL.
<i>channel</i>	the channel (frequency band) ID.

**4.10.4.11 caerEarEventSetEar()**

```
static void caerEarEventSetEar (
    caerEarEvent event,
    uint8_t ear ) [inline], [static]
```

Set the numerical ID of the ear (microphone). Usually, 0 is left, 1 is right for 2 ear cochleas. For 4 ear cochleas, 0 is front left, 1 is front right, 2 is back left and 3 is back right.

**Parameters**

<i>event</i>	a valid EarEvent pointer. Cannot be NULL.
<i>ear</i>	the ear (microphone) ID.

**4.10.4.12 caerEarEventSetTimestamp()**

```
static void caerEarEventSetTimestamp (
    caerEarEvent event,
    int32_t timestamp ) [inline], [static]
```

Set the 32bit event timestamp, the value has to be in microseconds.

**Parameters**

<i>event</i>	a valid EarEvent pointer. Cannot be NULL.
<i>timestamp</i>	a positive 32bit microsecond timestamp.

#### 4.10.4.13 caerEarEventValidate()

```
static void caerEarEventValidate (
    caerEarEvent event,
    caerEarEventPacket packet ) [inline], [static]
```

Validate the current event by setting its valid bit to true and increasing the event packet's event count and valid event count. Only works on events that are invalid. DO NOT CALL THIS AFTER HAVING PREVIOUSLY ALREADY INVALIDATED THIS EVENT, the total count will be incorrect.

##### Parameters

<i>event</i>	a valid EarEvent pointer. Cannot be NULL.
<i>packet</i>	the EarEventPacket pointer for the packet containing this event. Cannot be NULL.

#### 4.10.4.14 PACKED\_STRUCT() [1/2]

```
PACKED_STRUCT (
    struct caer_ear_event { uint32_t data;int32_t timestamp;} )
```

Ear (cochlea) event data structure definition. Contains information on events gotten from a cochlea chip: ears, channels, neurons and filters are stored. Signed integers are used for fields that are to be interpreted directly, for compatibility with languages that do not have unsigned integer types, such as Java.

#### 4.10.4.15 PACKED\_STRUCT() [2/2]

```
PACKED_STRUCT (
    struct caer_ear_event_packet { struct caer_event_packet_header packetHeader;struct
    caer_ear_event events[];} )
```

Ear (cochlea) event packet data structure definition. EventPackets are always made up of the common packet header, followed by 'eventCapacity' events. Everything has to be in one contiguous memory block.

## 4.11 events/frame.h File Reference

```
#include "common.h"
```

## Macros

- `#define CAER_FRAME_ITERATOR_ALL_START(FRAME_PACKET)`
  - `#define CAER_FRAME_CONST_ITERATOR_ALL_START(FRAME_PACKET)`
  - `#define CAER_FRAME_ITERATOR_ALL_END }`
  - `#define CAER_FRAME_ITERATOR_VALID_START(FRAME_PACKET)`
  - `#define CAER_FRAME_CONST_ITERATOR_VALID_START(FRAME_PACKET)`
  - `#define CAER_FRAME_ITERATOR_VALID_END }`
  - `#define CAER_FRAME_REVERSE_ITERATOR_ALL_START(FRAME_PACKET)`
  - `#define CAER_FRAME_CONST_REVERSE_ITERATOR_ALL_START(FRAME_PACKET)`
  - `#define CAER_FRAME_REVERSE_ITERATOR_ALL_END }`
  - `#define CAER_FRAME_REVERSE_ITERATOR_VALID_START(FRAME_PACKET)`
  - `#define CAER_FRAME_CONST_REVERSE_ITERATOR_VALID_START(FRAME_PACKET)`
  - `#define CAER_FRAME_REVERSE_ITERATOR_VALID_END }`
- 
- `#define FRAME_COLOR_CHANNELS_SHIFT 1`
  - `#define FRAME_COLOR_CHANNELS_MASK 0x00000007`
  - `#define FRAME_COLOR_FILTER_SHIFT 4`
  - `#define FRAME_COLOR_FILTER_MASK 0x0000000F`
  - `#define FRAME_ROI_IDENTIFIER_SHIFT 8`
  - `#define FRAME_ROI_IDENTIFIER_MASK 0x0000007F`

## Typedefs

- `typedef struct caer_frame_event * caerFrameEvent`
- `typedef const struct caer_frame_event * caerFrameEventConst`
- `typedef struct caer_frame_event_packet * caerFrameEventPacket`
- `typedef const struct caer_frame_event_packet * caerFrameEventPacketConst`

## Enumerations

- `enum caer_frame_event_color_channels { GRAYSCALE = 1, RGB = 3, RGBA = 4 }`
- `enum caer_frame_event_color_filter {  
MONO = 0, RGBG = 1, GRGB = 2, GBGR = 3,  
BGRG = 4, RGBW = 5, GRWB = 6, WBGR = 7,  
BWRG = 8 }`

## Functions

- `PACKED_STRUCT` (struct caer\_frame\_event { uint32\_t info;int32\_t ts\_startframe;int32\_t ts\_endframe;int32\_t ts\_startexposure;int32\_t ts\_endexposure;int32\_t lengthX;int32\_t lengthY;int32\_t positionX;int32\_t positionY;uint16\_t pixels[1];})
- `PACKED_STRUCT` (struct caer\_frame\_event\_packet { struct caer\_event\_packet\_header packetHeader;})
- `caerFrameEventPacket caerFrameEventPacketAllocate` (int32\_t eventCapacity, int16\_t eventSource, int32\_t tsOverflow, int32\_t maxLengthX, int32\_t maxLengthY, int16\_t maxChannelNumber)
- `static caerFrameEvent caerFrameEventPacketGetEvent` (caerFrameEventPacket packet, int32\_t n)
- `static caerFrameEventConst caerFrameEventPacketGetEventConst` (caerFrameEventPacketConst packet, int32\_t n)
- `static int32_t caerFrameEventGetTSSStartOfFrame` (caerFrameEventConst event)

- static int64\_t [caerFrameEventGetTSStartOfFrame64](#) (caerFrameEventConst event, caerFrameEventPacketConst packet)
- static void [caerFrameEventSetTSStartOfFrame](#) (caerFrameEvent event, int32\_t startFrame)
- static int32\_t [caerFrameEventGetTSEndOfFrame](#) (caerFrameEventConst event)
- static int64\_t [caerFrameEventGetTSEndOfFrame64](#) (caerFrameEventConst event, caerFrameEventPacketConst packet)
- static void [caerFrameEventSetTSEndOfFrame](#) (caerFrameEvent event, int32\_t endFrame)
- static int32\_t [caerFrameEventGetTSStartOfExposure](#) (caerFrameEventConst event)
- static int64\_t [caerFrameEventGetTSStartOfExposure64](#) (caerFrameEventConst event, caerFrameEventPacketConst packet)
- static void [caerFrameEventSetTSStartOfExposure](#) (caerFrameEvent event, int32\_t startExposure)
- static int32\_t [caerFrameEventGetTSEndOfExposure](#) (caerFrameEventConst event)
- static int64\_t [caerFrameEventGetTSEndOfExposure64](#) (caerFrameEventConst event, caerFrameEventPacketConst packet)
- static void [caerFrameEventSetTSEndOfExposure](#) (caerFrameEvent event, int32\_t endExposure)
- static int32\_t [caerFrameEventGetExposureLength](#) (caerFrameEventConst event)
- static int32\_t [caerFrameEventGetTimestamp](#) (caerFrameEventConst event)
- static int64\_t [caerFrameEventGetTimestamp64](#) (caerFrameEventConst event, caerFrameEventPacketConst packet)
- static bool [caerFrameEventsValid](#) (caerFrameEventConst event)
- static void [caerFrameEventValidate](#) (caerFrameEvent event, caerFrameEventPacket packet)
- static void [caerFrameEventInvalidate](#) (caerFrameEvent event, caerFrameEventPacket packet)
- static size\_t [caerFrameEventPacketGetPixelsSize](#) (caerFrameEventPacketConst packet)
- static size\_t [caerFrameEventPacketGetPixelsMaxIndex](#) (caerFrameEventPacketConst packet)
- static uint8\_t [caerFrameEventGetROIIdentifier](#) (caerFrameEventConst event)
- static void [caerFrameEventSetROIIdentifier](#) (caerFrameEvent event, uint8\_t roiIdentifier)
- static enum [caer\\_frame\\_event\\_color\\_filter](#) [caerFrameEventGetColorFilter](#) (caerFrameEventConst event)
- static void [caerFrameEventSetColorFilter](#) (caerFrameEvent event, enum [caer\\_frame\\_event\\_color\\_filter](#) colorFilter)
- static int32\_t [caerFrameEventGetLengthX](#) (caerFrameEventConst event)
- static int32\_t [caerFrameEventGetLengthY](#) (caerFrameEventConst event)
- static enum [caer\\_frame\\_event\\_color\\_channels](#) [caerFrameEventGetChannelNumber](#) (caerFrameEventConst event)
- static void [caerFrameEventSetLengthXLengthYChannelNumber](#) (caerFrameEvent event, int32\_t lengthX, int32\_t lengthY, enum [caer\\_frame\\_event\\_color\\_channels](#) channelNumber, caerFrameEventPacketConst packet)
- static size\_t [caerFrameEventGetPixelsMaxIndex](#) (caerFrameEventConst event)
- static size\_t [caerFrameEventGetPixelsSize](#) (caerFrameEventConst event)
- static int32\_t [caerFrameEventGetPositionX](#) (caerFrameEventConst event)
- static void [caerFrameEventSetPositionX](#) (caerFrameEvent event, int32\_t positionX)
- static int32\_t [caerFrameEventGetPositionY](#) (caerFrameEventConst event)
- static void [caerFrameEventSetPositionY](#) (caerFrameEvent event, int32\_t positionY)
- static uint16\_t [caerFrameEventGetPixel](#) (caerFrameEventConst event, int32\_t xAddress, int32\_t yAddress)
- static void [caerFrameEventSetPixel](#) (caerFrameEvent event, int32\_t xAddress, int32\_t yAddress, uint16\_t pixelValue)
- static uint16\_t [caerFrameEventGetPixelForChannel](#) (caerFrameEventConst event, int32\_t xAddress, int32\_t yAddress, uint8\_t channel)
- static void [caerFrameEventSetPixelForChannel](#) (caerFrameEvent event, int32\_t xAddress, int32\_t yAddress, uint8\_t channel, uint16\_t pixelValue)
- static uint16\_t [caerFrameEventGetPixelUnsafe](#) (caerFrameEventConst event, int32\_t xAddress, int32\_t yAddress)
- static void [caerFrameEventSetPixelUnsafe](#) (caerFrameEvent event, int32\_t xAddress, int32\_t yAddress, uint16\_t pixelValue)
- static uint16\_t [caerFrameEventGetPixelForChannelUnsafe](#) (caerFrameEventConst event, int32\_t xAddress, int32\_t yAddress, uint8\_t channel)

- static void [caerFrameEventSetPixelForChannelUnsafe](#) ([caerFrameEvent](#) event, int32\_t xAddress, int32\_t yAddress, uint8\_t channel, uint16\_t pixelValue)
- static uint16\_t \* [caerFrameEventGetPixelArrayUnsafe](#) ([caerFrameEvent](#) event)
- static const uint16\_t \* [caerFrameEventGetPixelArrayUnsafeConst](#) ([caerFrameEventConst](#) event)

### 4.11.1 Detailed Description

Frame Events format definition and handling functions. This event type encodes intensity frames, like you would get from a normal APS camera. It supports multiple channels for color, color filter information, as well as multiple Regions of Interest (ROI). The (0, 0) pixel is in the upper left corner of the screen, like in OpenCV/computer graphics. The pixel array is laid out row by row (increasing X axis), going from top to bottom (increasing Y axis). To copy a frame event, the usual assignment operator = cannot be used. Please use [caerGenericEventCopy\(\)](#) to copy frame events!

### 4.11.2 Macro Definition Documentation

#### 4.11.2.1 CAER\_FRAME\_CONST\_ITERATOR\_ALL\_START

```
#define CAER_FRAME_CONST_ITERATOR_ALL_START(  
    FRAME_PACKET )
```

**Value:**

```
for (int32_t caerFrameIteratorCounter = 0; \  
    caerFrameIteratorCounter < caerEventPacketHeaderGetEventNumber(&  
    (FRAME_PACKET)->packetHeader); \  
    caerFrameIteratorCounter++) { \  
    caerFrameEventConst caerFrameIteratorElement =  
    caerFrameEventPacketGetEventConst (FRAME_PACKET, caerFrameIteratorCounter);
```

Const-Iterator over all frame events in a packet. Returns the current index in the 'caerFrameIteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerFrameIteratorElement' variable of type [caerFrameEventConst](#).

FRAME\_PACKET: a valid [FrameEventPacket](#) pointer. Cannot be NULL.

#### 4.11.2.2 CAER\_FRAME\_CONST\_ITERATOR\_VALID\_START

```
#define CAER_FRAME_CONST_ITERATOR_VALID_START(  
    FRAME_PACKET )
```

**Value:**

```
for (int32_t caerFrameIteratorCounter = 0; \  
    caerFrameIteratorCounter < caerEventPacketHeaderGetEventNumber(&  
    (FRAME_PACKET)->packetHeader); \  
    caerFrameIteratorCounter++) { \  
    caerFrameEventConst caerFrameIteratorElement =  
    caerFrameEventPacketGetEventConst (FRAME_PACKET, caerFrameIteratorCounter);  
    if (!caerFrameEventIsValid(caerFrameIteratorElement)) { continue; }
```

Const-Iterator over only the valid frame events in a packet. Returns the current index in the 'caerFrameIteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerFrameIteratorElement' variable of type [caerFrameEventConst](#).

FRAME\_PACKET: a valid [FrameEventPacket](#) pointer. Cannot be NULL.

#### 4.11.2.3 CAER\_FRAME\_CONST\_REVERSE\_ITERATOR\_ALL\_START

```
#define CAER_FRAME_CONST_REVERSE_ITERATOR_ALL_START (
    FRAME_PACKET )
```

**Value:**

```
for (int32_t caerFrameIteratorCounter = caerEventPacketHeaderGetEventNumber
    (&(FRAME_PACKET)->packetHeader) - 1; \
    caerFrameIteratorCounter >= 0; \
    caerFrameIteratorCounter--) { \
    caerFrameEventConst caerFrameIteratorElement =
    caerFrameEventPacketGetEventConst (FRAME_PACKET, caerFrameIteratorCounter);
```

Const-Reverse iterator over all frame events in a packet. Returns the current index in the 'caerFrameIteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerFrameIteratorElement' variable of type caerFrameEventConst.

FRAME\_PACKET: a valid FrameEventPacket pointer. Cannot be NULL.

#### 4.11.2.4 CAER\_FRAME\_CONST\_REVERSE\_ITERATOR\_VALID\_START

```
#define CAER_FRAME_CONST_REVERSE_ITERATOR_VALID_START (
    FRAME_PACKET )
```

**Value:**

```
for (int32_t caerFrameIteratorCounter = caerEventPacketHeaderGetEventNumber
    (&(FRAME_PACKET)->packetHeader) - 1; \
    caerFrameIteratorCounter >= 0; \
    caerFrameIteratorCounter--) { \
    caerFrameEventConst caerFrameIteratorElement =
    caerFrameEventPacketGetEventConst (FRAME_PACKET, caerFrameIteratorCounter);
    \
    if (!caerFrameEventIsValid(caerFrameIteratorElement)) { continue; }
```

Const-Reverse iterator over only the valid frame events in a packet. Returns the current index in the 'caerFrameIteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerFrameIteratorElement' variable of type caerFrameEventConst.

FRAME\_PACKET: a valid FrameEventPacket pointer. Cannot be NULL.

#### 4.11.2.5 CAER\_FRAME\_ITERATOR\_ALL\_END

```
#define CAER_FRAME_ITERATOR_ALL_END }
```

Iterator close statement.



## 4.11.2.6 CAER\_FRAME\_ITERATOR\_ALL\_START

```
#define CAER_FRAME_ITERATOR_ALL_START(
    FRAME_PACKET )
```

**Value:**

```
for (int32_t caerFrameIteratorCounter = 0; \
     caerFrameIteratorCounter < caerEventPacketHeaderGetEventNumber(&
(FRAME_PACKET)->packetHeader); \
     caerFrameIteratorCounter++) { \
    caerFrameEvent caerFrameIteratorElement = caerFrameEventPacketGetEvent(
FRAME_PACKET, caerFrameIteratorCounter);
```

Iterator over all frame events in a packet. Returns the current index in the 'caerFrameIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerFrameIteratorElement' variable of type caerFrameEvent.

FRAME\_PACKET: a valid FrameEventPacket pointer. Cannot be NULL.

## 4.11.2.7 CAER\_FRAME\_ITERATOR\_VALID\_END

```
#define CAER_FRAME_ITERATOR_VALID_END }
```

Iterator close statement.

## 4.11.2.8 CAER\_FRAME\_ITERATOR\_VALID\_START

```
#define CAER_FRAME_ITERATOR_VALID_START(
    FRAME_PACKET )
```

**Value:**

```
for (int32_t caerFrameIteratorCounter = 0; \
     caerFrameIteratorCounter < caerEventPacketHeaderGetEventNumber(&
(FRAME_PACKET)->packetHeader); \
     caerFrameIteratorCounter++) { \
    caerFrameEvent caerFrameIteratorElement = caerFrameEventPacketGetEvent(
FRAME_PACKET, caerFrameIteratorCounter); \
    if (!caerFrameEventIsValid(caerFrameIteratorElement)) { continue; }
```

Iterator over only the valid frame events in a packet. Returns the current index in the 'caerFrameIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerFrameIteratorElement' variable of type caerFrameEvent.

FRAME\_PACKET: a valid FrameEventPacket pointer. Cannot be NULL.

## 4.11.2.9 CAER\_FRAME\_REVERSE\_ITERATOR\_ALL\_END

```
#define CAER_FRAME_REVERSE_ITERATOR_ALL_END }
```

Reverse iterator close statement.

#### 4.11.2.10 CAER\_FRAME\_REVERSE\_ITERATOR\_ALL\_START

```
#define CAER_FRAME_REVERSE_ITERATOR_ALL_START(
    FRAME_PACKET )
```

**Value:**

```
for (int32_t caerFrameIteratorCounter = caerEventPacketHeaderGetEventNumber
    (&(FRAME_PACKET)->packetHeader) - 1; \
    caerFrameIteratorCounter >= 0; \
    caerFrameIteratorCounter--) { \
    caerFrameEvent caerFrameIteratorElement = caerFrameEventPacketGetEvent(
    FRAME_PACKET, caerFrameIteratorCounter);
```

Reverse iterator over all frame events in a packet. Returns the current index in the 'caerFrameIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerFrameIteratorElement' variable of type caerFrameEvent.

FRAME\_PACKET: a valid FrameEventPacket pointer. Cannot be NULL.

#### 4.11.2.11 CAER\_FRAME\_REVERSE\_ITERATOR\_VALID\_END

```
#define CAER_FRAME_REVERSE_ITERATOR_VALID_END }
```

Reverse iterator close statement.

#### 4.11.2.12 CAER\_FRAME\_REVERSE\_ITERATOR\_VALID\_START

```
#define CAER_FRAME_REVERSE_ITERATOR_VALID_START(
    FRAME_PACKET )
```

**Value:**

```
for (int32_t caerFrameIteratorCounter = caerEventPacketHeaderGetEventNumber
    (&(FRAME_PACKET)->packetHeader) - 1; \
    caerFrameIteratorCounter >= 0; \
    caerFrameIteratorCounter--) { \
    caerFrameEvent caerFrameIteratorElement = caerFrameEventPacketGetEvent(
    FRAME_PACKET, caerFrameIteratorCounter); \
    if (!caerFrameEventIsValid(caerFrameIteratorElement)) { continue; }
```

Reverse iterator over only the valid frame events in a packet. Returns the current index in the 'caerFrameIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerFrameIteratorElement' variable of type caerFrameEvent.

FRAME\_PACKET: a valid FrameEventPacket pointer. Cannot be NULL.

#### 4.11.2.13 FRAME\_COLOR\_CHANNELS\_MASK

```
#define FRAME_COLOR_CHANNELS_MASK 0x00000007
```

Shift and mask values for the color channels number, the color filter arrangement and the ROI identifier contained in the 'info' field of the frame event. Multiple channels (RGB for example) are possible, see the 'enum caer\_frame\_event\_color\_channels'. To understand the original color filter arrangement to interpolate color images, see the 'enum caer\_frame\_event\_color\_filter'. Also, up to 128 different Regions of Interest (ROI) can be tracked. Bit 0 is the valid mark, see [common.h](#) for more details.

#### 4.11.2.14 FRAME\_COLOR\_CHANNELS\_SHIFT

```
#define FRAME_COLOR_CHANNELS_SHIFT 1
```

Shift and mask values for the color channels number, the color filter arrangement and the ROI identifier contained in the 'info' field of the frame event. Multiple channels (RGB for example) are possible, see the 'enum caer\_frame\_event\_color\_channels'. To understand the original color filter arrangement to interpolate color images, see the 'enum caer\_frame\_event\_color\_filter'. Also, up to 128 different Regions of Interest (ROI) can be tracked. Bit 0 is the valid mark, see '[common.h](#)' for more details.

#### 4.11.2.15 FRAME\_COLOR\_FILTER\_MASK

```
#define FRAME_COLOR_FILTER_MASK 0x0000000F
```

Shift and mask values for the color channels number, the color filter arrangement and the ROI identifier contained in the 'info' field of the frame event. Multiple channels (RGB for example) are possible, see the 'enum caer\_frame\_event\_color\_channels'. To understand the original color filter arrangement to interpolate color images, see the 'enum caer\_frame\_event\_color\_filter'. Also, up to 128 different Regions of Interest (ROI) can be tracked. Bit 0 is the valid mark, see '[common.h](#)' for more details.

#### 4.11.2.16 FRAME\_COLOR\_FILTER\_SHIFT

```
#define FRAME_COLOR_FILTER_SHIFT 4
```

Shift and mask values for the color channels number, the color filter arrangement and the ROI identifier contained in the 'info' field of the frame event. Multiple channels (RGB for example) are possible, see the 'enum caer\_frame\_event\_color\_channels'. To understand the original color filter arrangement to interpolate color images, see the 'enum caer\_frame\_event\_color\_filter'. Also, up to 128 different Regions of Interest (ROI) can be tracked. Bit 0 is the valid mark, see '[common.h](#)' for more details.

#### 4.11.2.17 FRAME\_ROI\_IDENTIFIER\_MASK

```
#define FRAME_ROI_IDENTIFIER_MASK 0x0000007F
```

Shift and mask values for the color channels number, the color filter arrangement and the ROI identifier contained in the 'info' field of the frame event. Multiple channels (RGB for example) are possible, see the 'enum caer\_frame\_event\_color\_channels'. To understand the original color filter arrangement to interpolate color images, see the 'enum caer\_frame\_event\_color\_filter'. Also, up to 128 different Regions of Interest (ROI) can be tracked. Bit 0 is the valid mark, see '[common.h](#)' for more details.

#### 4.11.2.18 FRAME\_ROI\_IDENTIFIER\_SHIFT

```
#define FRAME_ROI_IDENTIFIER_SHIFT 8
```

Shift and mask values for the color channels number, the color filter arrangement and the ROI identifier contained in the 'info' field of the frame event. Multiple channels (RGB for example) are possible, see the 'enum caer\_frame\_event\_color\_channels'. To understand the original color filter arrangement to interpolate color images, see the 'enum caer\_frame\_event\_color\_filter'. Also, up to 128 different Regions of Interest (ROI) can be tracked. Bit 0 is the valid mark, see '[common.h](#)' for more details.

### 4.11.3 Typedef Documentation

#### 4.11.3.1 caerFrameEvent

```
typedef struct caer_frame_event* caerFrameEvent
```

Type for pointer to frame event data structure.

#### 4.11.3.2 caerFrameEventPacket

```
typedef struct caer_frame_event_packet* caerFrameEventPacket
```

Type for pointer to frame event packet data structure.

### 4.11.4 Enumeration Type Documentation

#### 4.11.4.1 caer\_frame\_event\_color\_channels

```
enum caer_frame_event_color_channels
```

List of all frame event color channel identifiers. Used to interpret the frame event color channel field.

##### Enumerator

GRAYSCALE	Grayscale, one channel only.
RGB	Red Green Blue, 3 color channels.
RGBA	Red Green Blue Alpha, 3 color channels plus transparency.

#### 4.11.4.2 caer\_frame\_event\_color\_filter

```
enum caer_frame_event_color_filter
```

List of all frame event color filter identifiers. Used to interpret the frame event color filter field.

##### Enumerator

MONO	No color filter present, all light passes.
RGBG	Standard Bayer color filter, 1 red 2 green 1 blue. Variation 1.
GRGB	Standard Bayer color filter, 1 red 2 green 1 blue. Variation 2.

## Enumerator

GBGR	Standard Bayer color filter, 1 red 2 green 1 blue. Variation 3.
BGRG	Standard Bayer color filter, 1 red 2 green 1 blue. Variation 4.
RGBW	Modified Bayer color filter, with white (pass all light) instead of extra green. Variation 1.
GRWB	Modified Bayer color filter, with white (pass all light) instead of extra green. Variation 2.
WBGR	Modified Bayer color filter, with white (pass all light) instead of extra green. Variation 3.
BWRG	Modified Bayer color filter, with white (pass all light) instead of extra green. Variation 4.

## 4.11.5 Function Documentation

## 4.11.5.1 caerFrameEventGetChannelNumber()

```
static enum caer_frame_event_color_channels caerFrameEventGetChannelNumber (
    caerFrameEventConst event ) [inline], [static]
```

Get the actual color channels number for the current frame. This can be used to store RGB frames for example.

## Parameters

<i>event</i>	a valid FrameEvent pointer. Cannot be NULL.
--------------	---

## Returns

frame color channels number.

## 4.11.5.2 caerFrameEventGetColorFilter()

```
static enum caer_frame_event_color_filter caerFrameEventGetColorFilter (
    caerFrameEventConst event ) [inline], [static]
```

Get the identifier for the color filter used by the sensor. Useful for interpolating color images.

## Parameters

<i>event</i>	a valid FrameEvent pointer. Cannot be NULL.
--------------	---

## Returns

color filter identifier.

#### 4.11.5.3 caerFrameEventGetExposureLength()

```
static int32_t caerFrameEventGetExposureLength (  
    caerFrameEventConst event ) [inline], [static]
```

The total length, in microseconds, of the frame exposure time.

##### Parameters

<i>event</i>	a valid FrameEvent pointer. Cannot be NULL.
--------------	---

##### Returns

the exposure time in microseconds.

#### 4.11.5.4 caerFrameEventGetLengthX()

```
static int32_t caerFrameEventGetLengthX (  
    caerFrameEventConst event ) [inline], [static]
```

Get the actual X axis length for the current frame.

##### Parameters

<i>event</i>	a valid FrameEvent pointer. Cannot be NULL.
--------------	---

##### Returns

frame X axis length.

#### 4.11.5.5 caerFrameEventGetLengthY()

```
static int32_t caerFrameEventGetLengthY (  
    caerFrameEventConst event ) [inline], [static]
```

Get the actual Y axis length for the current frame.

##### Parameters

<i>event</i>	a valid FrameEvent pointer. Cannot be NULL.
--------------	---

##### Returns

frame Y axis length.

#### 4.11.5.6 caerFrameEventGetPixel()

```
static uint16_t caerFrameEventGetPixel (
    caerFrameEventConst event,
    int32_t xAddress,
    int32_t yAddress ) [inline], [static]
```

Get the pixel value at the specified (X, Y) address. (X, Y) are checked against the actual possible values for this frame. Different channels are not taken into account! The (0, 0) pixel is in the upper left corner, like in OpenCV/computer graphics.

##### Parameters

<i>event</i>	a valid FrameEvent pointer. Cannot be NULL.
<i>xAddress</i>	X address value (checked).
<i>yAddress</i>	Y address value (checked).

##### Returns

pixel value (normalized to 16 bit depth).

#### 4.11.5.7 caerFrameEventGetPixelArrayUnsafe()

```
static uint16_t* caerFrameEventGetPixelArrayUnsafe (
    caerFrameEvent event ) [inline], [static]
```

Get a direct pointer to the underlying pixels array. This can be used to both get and set values. No checks at all are performed at any point, nor any conversions, use this at your own risk! Remember that the 16 bit pixel values are in little-endian! The pixel array is laid out row by row (increasing X axis), going from top to bottom (increasing Y axis).

##### Parameters

<i>event</i>	a valid FrameEvent pointer. Cannot be NULL.
--------------	---

##### Returns

the pixels array (16 bit integers are little-endian).

#### 4.11.5.8 caerFrameEventGetPixelArrayUnsafeConst()

```
static const uint16_t* caerFrameEventGetPixelArrayUnsafeConst (
    caerFrameEventConst event ) [inline], [static]
```

Get a direct read-only pointer to the underlying pixels array. This can be used to only get values. No checks at all are performed at any point, nor any conversions, use this at your own risk! Remember that the 16 bit pixel values are in little-endian! The pixel array is laid out row by row (increasing X axis), going from top to bottom (increasing Y axis).

#### Parameters

<i>event</i>	a valid FrameEvent pointer. Cannot be NULL.
--------------	---

#### Returns

the read-only pixels array (16 bit integers are little-endian).

#### 4.11.5.9 caerFrameEventGetPixelForChannel()

```
static uint16_t caerFrameEventGetPixelForChannel (
    caerFrameEventConst event,
    int32_t xAddress,
    int32_t yAddress,
    uint8_t channel ) [inline], [static]
```

Get the pixel value at the specified (X, Y) address, taking into account the specified channel. (X, Y) and the channel number are checked against the actual possible values for this frame. The (0, 0) pixel is in the upper left corner, like in OpenCV/computer graphics.

#### Parameters

<i>event</i>	a valid FrameEvent pointer. Cannot be NULL.
<i>xAddress</i>	X address value (checked).
<i>yAddress</i>	Y address value (checked).
<i>channel</i>	the channel number (checked).

#### Returns

pixel value (normalized to 16 bit depth).

#### 4.11.5.10 caerFrameEventGetPixelForChannelUnsafe()

```
static uint16_t caerFrameEventGetPixelForChannelUnsafe (
    caerFrameEventConst event,
    int32_t xAddress,
    int32_t yAddress,
    uint8_t channel ) [inline], [static]
```

Get the pixel value at the specified (X, Y) address, taking into account the specified channel. No checks on (X, Y) and the channel number are performed! The (0, 0) pixel is in the upper left corner, like in OpenCV/computer graphics.



## Parameters

<i>event</i>	a valid FrameEvent pointer. Cannot be NULL.
<i>xAddress</i>	X address value (unchecked).
<i>yAddress</i>	Y address value (unchecked).
<i>channel</i>	the channel number (unchecked).

## Returns

pixel value (normalized to 16 bit depth).

## 4.11.5.11 caerFrameEventGetPixelsMaxIndex()

```
static size_t caerFrameEventGetPixelsMaxIndex (  
    caerFrameEventConst event ) [inline], [static]
```

Get the maximum valid index into the pixel array, at which you can still get valid pixels.

## Parameters

<i>event</i>	a valid FrameEvent pointer. Cannot be NULL.
--------------	---

## Returns

maximum valid pixels array index.

## 4.11.5.12 caerFrameEventGetPixelsSize()

```
static size_t caerFrameEventGetPixelsSize (  
    caerFrameEventConst event ) [inline], [static]
```

Get the maximum size of the pixels array in bytes, in which you can still get valid pixels.

## Parameters

<i>event</i>	a valid FrameEvent pointer. Cannot be NULL.
--------------	---

## Returns

maximum valid pixels array size in bytes.

#### 4.11.5.13 caerFrameEventGetPixelUnsafe()

```
static uint16_t caerFrameEventGetPixelUnsafe (
    caerFrameEventConst event,
    int32_t xAddress,
    int32_t yAddress ) [inline], [static]
```

Get the pixel value at the specified (X, Y) address. No checks on (X, Y) are performed! The (0, 0) pixel is in the upper left corner, like in OpenCV/computer graphics.

##### Parameters

<i>event</i>	a valid FrameEvent pointer. Cannot be NULL.
<i>xAddress</i>	X address value (unchecked).
<i>yAddress</i>	Y address value (unchecked).

##### Returns

pixel value (normalized to 16 bit depth).

#### 4.11.5.14 caerFrameEventGetPositionX()

```
static int32_t caerFrameEventGetPositionX (
    caerFrameEventConst event ) [inline], [static]
```

Get the X axis position offset. This is used to place partial frames, like the ones gotten from ROI readouts, in the visual space.

##### Parameters

<i>event</i>	a valid FrameEvent pointer. Cannot be NULL.
--------------	---

##### Returns

X axis position offset.

#### 4.11.5.15 caerFrameEventGetPositionY()

```
static int32_t caerFrameEventGetPositionY (
    caerFrameEventConst event ) [inline], [static]
```

Get the Y axis position offset. This is used to place partial frames, like the ones gotten from ROI readouts, in the visual space.

**Parameters**

<i>event</i>	a valid FrameEvent pointer. Cannot be NULL.
--------------	---

**Returns**

Y axis position offset.

**4.11.5.16 caerFrameEventGetROIIdentifier()**

```
static uint8_t caerFrameEventGetROIIdentifier (  
    caerFrameEventConst event ) [inline], [static]
```

Get the numerical identifier for the Region of Interest (ROI) region, to distinguish between multiple of them.

**Parameters**

<i>event</i>	a valid FrameEvent pointer. Cannot be NULL.
--------------	---

**Returns**

numerical ROI identifier.

**4.11.5.17 caerFrameEventGetTimestamp()**

```
static int32_t caerFrameEventGetTimestamp (  
    caerFrameEventConst event ) [inline], [static]
```

Get the 32bit event timestamp, in microseconds. This is a median of the exposure timestamps. Be aware that this wraps around! You can either ignore this fact, or handle the special 'TIMESTAMP\_WRAP' event that is generated when this happens, or use the 64bit timestamp which never wraps around. See '[caerEventPacketHeaderGetEventTSOverflow\(\)](#)' documentation for more details on the 64bit timestamp.

**Parameters**

<i>event</i>	a valid FrameEvent pointer. Cannot be NULL.
--------------	---

**Returns**

this event's 32bit microsecond timestamp.

#### 4.11.5.18 caerFrameEventGetTimestamp64()

```
static int64_t caerFrameEventGetTimestamp64 (
    caerFrameEventConst event,
    caerFrameEventPacketConst packet ) [inline], [static]
```

Get the 64bit event timestamp, in microseconds. This is a median of the exposure timestamps. See '[caerEventPacketHeaderGetEventTSOverflow\(\)](#)' documentation for more details on the 64bit timestamp.

##### Parameters

<i>event</i>	a valid FrameEvent pointer. Cannot be NULL.
<i>packet</i>	the FrameEventPacket pointer for the packet containing this event. Cannot be NULL.

##### Returns

this event's 64bit microsecond timestamp.

#### 4.11.5.19 caerFrameEventGetTSEndOfExposure()

```
static int32_t caerFrameEventGetTSEndOfExposure (
    caerFrameEventConst event ) [inline], [static]
```

Get the 32bit end of exposure timestamp, in microseconds. Be aware that this wraps around! You can either ignore this fact, or handle the special 'TIMESTAMP\_WRAP' event that is generated when this happens, or use the 64bit timestamp which never wraps around. See '[caerEventPacketHeaderGetEventTSOverflow\(\)](#)' documentation for more details on the 64bit timestamp.

##### Parameters

<i>event</i>	a valid FrameEvent pointer. Cannot be NULL.
--------------	---

##### Returns

this event's 32bit microsecond end of exposure timestamp.

#### 4.11.5.20 caerFrameEventGetTSEndOfExposure64()

```
static int64_t caerFrameEventGetTSEndOfExposure64 (
    caerFrameEventConst event,
    caerFrameEventPacketConst packet ) [inline], [static]
```

Get the 64bit end of exposure timestamp, in microseconds. See '[caerEventPacketHeaderGetEventTSOverflow\(\)](#)' documentation for more details on the 64bit timestamp.

## Parameters

<i>event</i>	a valid FrameEvent pointer. Cannot be NULL.
<i>packet</i>	the FrameEventPacket pointer for the packet containing this event. Cannot be NULL.

## Returns

this event's 64bit microsecond end of exposure timestamp.

## 4.11.5.21 caerFrameEventGetTSEndOfFrame()

```
static int32_t caerFrameEventGetTSEndOfFrame (  
    caerFrameEventConst event ) [inline], [static]
```

Get the 32bit end of frame capture timestamp, in microseconds. Be aware that this wraps around! You can either ignore this fact, or handle the special 'TIMESTAMP\_WRAP' event that is generated when this happens, or use the 64bit timestamp which never wraps around. See '[caerEventPacketHeaderGetEventTSOverflow\(\)](#)' documentation for more details on the 64bit timestamp.

## Parameters

<i>event</i>	a valid FrameEvent pointer. Cannot be NULL.
--------------	---

## Returns

this event's 32bit microsecond end of frame timestamp.

## 4.11.5.22 caerFrameEventGetTSEndOfFrame64()

```
static int64_t caerFrameEventGetTSEndOfFrame64 (  
    caerFrameEventConst event,  
    caerFrameEventPacketConst packet ) [inline], [static]
```

Get the 64bit end of frame capture timestamp, in microseconds. See '[caerEventPacketHeaderGetEventTSOverflow\(\)](#)' documentation for more details on the 64bit timestamp.

## Parameters

<i>event</i>	a valid FrameEvent pointer. Cannot be NULL.
<i>packet</i>	the FrameEventPacket pointer for the packet containing this event. Cannot be NULL.

## Returns

this event's 64bit microsecond end of frame timestamp.

#### 4.11.5.23 caerFrameEventGetTSStartOfExposure()

```
static int32_t caerFrameEventGetTSStartOfExposure (
    caerFrameEventConst event ) [inline], [static]
```

Get the 32bit start of exposure timestamp, in microseconds. Be aware that this wraps around! You can either ignore this fact, or handle the special 'TIMESTAMP\_WRAP' event that is generated when this happens, or use the 64bit timestamp which never wraps around. See '[caerEventPacketHeaderGetEventTSOverflow\(\)](#)' documentation for more details on the 64bit timestamp.

##### Parameters

<i>event</i>	a valid FrameEvent pointer. Cannot be NULL.
--------------	---

##### Returns

this event's 32bit microsecond start of exposure timestamp.

#### 4.11.5.24 caerFrameEventGetTSStartOfExposure64()

```
static int64_t caerFrameEventGetTSStartOfExposure64 (
    caerFrameEventConst event,
    caerFrameEventPacketConst packet ) [inline], [static]
```

Get the 64bit start of exposure timestamp, in microseconds. See '[caerEventPacketHeaderGetEventTSOverflow\(\)](#)' documentation for more details on the 64bit timestamp.

##### Parameters

<i>event</i>	a valid FrameEvent pointer. Cannot be NULL.
<i>packet</i>	the FrameEventPacket pointer for the packet containing this event. Cannot be NULL.

##### Returns

this event's 64bit microsecond start of exposure timestamp.

#### 4.11.5.25 caerFrameEventGetTSStartOfFrame()

```
static int32_t caerFrameEventGetTSStartOfFrame (
    caerFrameEventConst event ) [inline], [static]
```

Get the 32bit start of frame capture timestamp, in microseconds. Be aware that this wraps around! You can either ignore this fact, or handle the special 'TIMESTAMP\_WRAP' event that is generated when this happens, or use the 64bit timestamp which never wraps around. See '[caerEventPacketHeaderGetEventTSOverflow\(\)](#)' documentation for more details on the 64bit timestamp.

## Parameters

<i>event</i>	a valid FrameEvent pointer. Cannot be NULL.
--------------	---

## Returns

this event's 32bit microsecond start of frame timestamp.

## 4.11.5.26 caerFrameEventGetTSStartOfFrame64()

```
static int64_t caerFrameEventGetTSStartOfFrame64 (  
    caerFrameEventConst event,  
    caerFrameEventPacketConst packet ) [inline], [static]
```

Get the 64bit start of frame capture timestamp, in microseconds. See '[caerEventPacketHeaderGetEventTSOverflow\(\)](#)' documentation for more details on the 64bit timestamp.

## Parameters

<i>event</i>	a valid FrameEvent pointer. Cannot be NULL.
<i>packet</i>	the FrameEventPacket pointer for the packet containing this event. Cannot be NULL.

## Returns

this event's 64bit microsecond start of frame timestamp.

## 4.11.5.27 caerFrameEventInvalidate()

```
static void caerFrameEventInvalidate (  
    caerFrameEvent event,  
    caerFrameEventPacket packet ) [inline], [static]
```

Invalidate the current event by setting its valid bit to false and decreasing the number of valid events held in the packet. Only works with events that are already valid!

## Parameters

<i>event</i>	a valid FrameEvent pointer. Cannot be NULL.
<i>packet</i>	the FrameEventPacket pointer for the packet containing this event. Cannot be NULL.

## 4.11.5.28 caerFrameEventIsValid()

```
static bool caerFrameEventIsValid (
    caerFrameEventConst event ) [inline], [static]
```

Check if this frame event is valid.

## Parameters

<i>event</i>	a valid FrameEvent pointer. Cannot be NULL.
--------------	---

## Returns

true if valid, false if not.

## 4.11.5.29 caerFrameEventPacketAllocate()

```
caerFrameEventPacket caerFrameEventPacketAllocate (
    int32_t eventCapacity,
    int16_t eventSource,
    int32_t tsOverflow,
    int32_t maxLengthX,
    int32_t maxLengthY,
    int16_t maxChannelNumber )
```

Allocate a new frame events packet. Use free() to reclaim this memory. The frame events allocate memory for a maximum sized pixels array, depending on the parameters passed to this function, so that every event occupies the same amount of memory (constant size). The actual frames inside of it might be smaller than that, for example when using ROI, and their actual size is stored inside the frame event and should always be queried from there. The unused part of a pixels array is guaranteed to be zeros.

## Parameters

<i>eventCapacity</i>	the maximum number of events this packet will hold.
<i>eventSource</i>	the unique ID representing the source/generator of this packet.
<i>tsOverflow</i>	the current timestamp overflow counter value for this packet.
<i>maxLengthX</i>	the maximum expected X axis size for frames in this packet.
<i>maxLengthY</i>	the maximum expected Y axis size for frames in this packet.
<i>maxChannelNumber</i>	the maximum expected number of channels for frames in this packet.

## Returns

a valid FrameEventPacket handle or NULL on error.



## 4.11.5.30 caerFrameEventPacketGetEvent()

```
static caerFrameEvent caerFrameEventPacketGetEvent (
    caerFrameEventPacket packet,
    int32_t n ) [inline], [static]
```

Get the frame event at the given index from the event packet. To copy a frame event, the usual assignment operator = cannot be used. Please use [caerGenericEventCopy\(\)](#) to copy frame events!

## Parameters

<i>packet</i>	a valid FrameEventPacket pointer. Cannot be NULL.
<i>n</i>	the index of the returned event. Must be within [0,eventCapacity[ bounds.

## Returns

the requested frame event. NULL on error.

## 4.11.5.31 caerFrameEventPacketGetEventConst()

```
static caerFrameEventConst caerFrameEventPacketGetEventConst (
    caerFrameEventPacketConst packet,
    int32_t n ) [inline], [static]
```

Get the frame event at the given index from the event packet. This is a read-only event, do not change its contents in any way! To copy a frame event, the usual assignment operator = cannot be used. Please use [caerGenericEventCopy\(\)](#) to copy frame events!

## Parameters

<i>packet</i>	a valid FrameEventPacket pointer. Cannot be NULL.
<i>n</i>	the index of the returned event. Must be within [0,eventCapacity[ bounds.

## Returns

the requested read-only frame event. NULL on error.

## 4.11.5.32 caerFrameEventPacketGetPixelsMaxIndex()

```
static size_t caerFrameEventPacketGetPixelsMaxIndex (
    caerFrameEventPacketConst packet ) [inline], [static]
```

Get the maximum index into the pixels array, based upon how much memory was allocated to it by '[caerFrameEventPacketAllocate\(\)](#)'.

## Parameters

<i>packet</i>	a valid FrameEventPacket pointer. Cannot be NULL.
---------------	---

## Returns

maximum pixels array index.

## 4.11.5.33 caerFrameEventPacketGetPixelsSize()

```
static size_t caerFrameEventPacketGetPixelsSize (
    caerFrameEventPacketConst packet ) [inline], [static]
```

Get the maximum size of the pixels array in bytes, based upon how much memory was allocated to it by '[caerFrameEventPacketAllocate\(\)](#)'.

## Parameters

<i>packet</i>	a valid FrameEventPacket pointer. Cannot be NULL.
---------------	---

## Returns

maximum pixels array size in bytes.

## 4.11.5.34 caerFrameEventSetColorFilter()

```
static void caerFrameEventSetColorFilter (
    caerFrameEvent event,
    enum caer_frame_event_color_filter colorFilter ) [inline], [static]
```

Set the identifier for the color filter used by the sensor. Useful for interpolating color images.

## Parameters

<i>event</i>	a valid FrameEvent pointer. Cannot be NULL.
<i>colorFilter</i>	color filter identifier.

## 4.11.5.35 caerFrameEventSetLengthXLengthYChannelNumber()

```
static void caerFrameEventSetLengthXLengthYChannelNumber (
    caerFrameEvent event,
```

```

int32_t lengthX,
int32_t lengthY,
enum caer_frame_event_color_channels channelNumber,
caerFrameEventPacketConst packet ) [inline], [static]

```

Set the X and Y axes length and the color channels number for a frame, while taking into account the maximum amount of memory available for the pixel array, as allocated in 'caerFrameEventPacketAllocate()'.

#### Parameters

<i>event</i>	a valid FrameEvent pointer. Cannot be NULL.
<i>lengthX</i>	the frame's X axis length.
<i>lengthY</i>	the frame's Y axis length.
<i>channelNumber</i>	the number of color channels for this frame.
<i>packet</i>	the FrameEventPacket pointer for the packet containing this event. Cannot be NULL.

#### 4.11.5.36 caerFrameEventSetPixel()

```

static void caerFrameEventSetPixel (
    caerFrameEvent event,
    int32_t xAddress,
    int32_t yAddress,
    uint16_t pixelValue ) [inline], [static]

```

Set the pixel value at the specified (X, Y) address. (X, Y) are checked against the actual possible values for this frame. Different channels are not taken into account! The (0, 0) pixel is in the upper left corner, like in OpenCV/computer graphics.

#### Parameters

<i>event</i>	a valid FrameEvent pointer. Cannot be NULL.
<i>xAddress</i>	X address value (checked).
<i>yAddress</i>	Y address value (checked).
<i>pixelValue</i>	pixel value (normalized to 16 bit depth).

#### 4.11.5.37 caerFrameEventSetPixelForChannel()

```

static void caerFrameEventSetPixelForChannel (
    caerFrameEvent event,
    int32_t xAddress,
    int32_t yAddress,
    uint8_t channel,
    uint16_t pixelValue ) [inline], [static]

```

Set the pixel value at the specified (X, Y) address, taking into account the specified channel. (X, Y) and the channel number are checked against the actual possible values for this frame. The (0, 0) pixel is in the upper left corner, like in OpenCV/computer graphics.

## Parameters

<i>event</i>	a valid FrameEvent pointer. Cannot be NULL.
<i>xAddress</i>	X address value (checked).
<i>yAddress</i>	Y address value (checked).
<i>channel</i>	the channel number (checked).
<i>pixelValue</i>	pixel value (normalized to 16 bit depth).

## 4.11.5.38 caerFrameEventSetPixelForChannelUnsafe()

```
static void caerFrameEventSetPixelForChannelUnsafe (
    caerFrameEvent event,
    int32_t xAddress,
    int32_t yAddress,
    uint8_t channel,
    uint16_t pixelValue ) [inline], [static]
```

Set the pixel value at the specified (X, Y) address, taking into account the specified channel. No checks on (X, Y) and the channel number are performed! The (0, 0) pixel is in the upper left corner, like in OpenCV/computer graphics.

## Parameters

<i>event</i>	a valid FrameEvent pointer. Cannot be NULL.
<i>xAddress</i>	X address value (unchecked).
<i>yAddress</i>	Y address value (unchecked).
<i>channel</i>	the channel number (unchecked).
<i>pixelValue</i>	pixel value (normalized to 16 bit depth).

## 4.11.5.39 caerFrameEventSetPixelUnsafe()

```
static void caerFrameEventSetPixelUnsafe (
    caerFrameEvent event,
    int32_t xAddress,
    int32_t yAddress,
    uint16_t pixelValue ) [inline], [static]
```

Set the pixel value at the specified (X, Y) address. No checks on (X, Y) are performed! The (0, 0) pixel is in the upper left corner, like in OpenCV/computer graphics.

## Parameters

<i>event</i>	a valid FrameEvent pointer. Cannot be NULL.
<i>xAddress</i>	X address value (unchecked).
<i>yAddress</i>	Y address value (unchecked).
<i>pixelValue</i>	pixel value (normalized to 16 bit depth).

#### 4.11.5.40 caerFrameEventSetPositionX()

```
static void caerFrameEventSetPositionX (
    caerFrameEvent event,
    int32_t positionX ) [inline], [static]
```

Set the X axis position offset. This is used to place partial frames, like the ones gotten from ROI readouts, in the visual space.

##### Parameters

<i>event</i>	a valid FrameEvent pointer. Cannot be NULL.
<i>positionX</i>	X axis position offset.

#### 4.11.5.41 caerFrameEventSetPositionY()

```
static void caerFrameEventSetPositionY (
    caerFrameEvent event,
    int32_t positionY ) [inline], [static]
```

Set the Y axis position offset. This is used to place partial frames, like the ones gotten from ROI readouts, in the visual space.

##### Parameters

<i>event</i>	a valid FrameEvent pointer. Cannot be NULL.
<i>positionY</i>	Y axis position offset.

#### 4.11.5.42 caerFrameEventSetROIIdentifier()

```
static void caerFrameEventSetROIIdentifier (
    caerFrameEvent event,
    uint8_t roiIdentifier ) [inline], [static]
```

Set the numerical identifier for the Region of Interest (ROI) region, to distinguish between multiple of them.

##### Parameters

<i>event</i>	a valid FrameEvent pointer. Cannot be NULL.
<i>roiIdentifier</i>	numerical ROI identifier.

#### 4.11.5.43 caerFrameEventSetTSEndOfExposure()

```
static void caerFrameEventSetTSEndOfExposure (
    caerFrameEvent event,
    int32_t endExposure ) [inline], [static]
```

Set the 32bit end of exposure timestamp, the value has to be in microseconds.

##### Parameters

<i>event</i>	a valid FrameEvent pointer. Cannot be NULL.
<i>endExposure</i>	a positive 32bit microsecond timestamp.

#### 4.11.5.44 caerFrameEventSetTSEndOfFrame()

```
static void caerFrameEventSetTSEndOfFrame (
    caerFrameEvent event,
    int32_t endFrame ) [inline], [static]
```

Set the 32bit end of frame capture timestamp, the value has to be in microseconds.

##### Parameters

<i>event</i>	a valid FrameEvent pointer. Cannot be NULL.
<i>endFrame</i>	a positive 32bit microsecond timestamp.

#### 4.11.5.45 caerFrameEventSetTSStartOfExposure()

```
static void caerFrameEventSetTSStartOfExposure (
    caerFrameEvent event,
    int32_t startExposure ) [inline], [static]
```

Set the 32bit start of exposure timestamp, the value has to be in microseconds.

##### Parameters

<i>event</i>	a valid FrameEvent pointer. Cannot be NULL.
<i>startExposure</i>	a positive 32bit microsecond timestamp.

#### 4.11.5.46 caerFrameEventSetTSStartOfFrame()

```
static void caerFrameEventSetTSStartOfFrame (
```

```

    caerFrameEvent event,
    int32_t startFrame ) [inline], [static]

```

Set the 32bit start of frame capture timestamp, the value has to be in microseconds.

#### Parameters

<i>event</i>	a valid FrameEvent pointer. Cannot be NULL.
<i>startFrame</i>	a positive 32bit microsecond timestamp.

#### 4.11.5.47 caerFrameEventValidate()

```

static void caerFrameEventValidate (
    caerFrameEvent event,
    caerFrameEventPacket packet ) [inline], [static]

```

Validate the current event by setting its valid bit to true and increasing the event packet's event count and valid event count. Only works on events that are invalid. DO NOT CALL THIS AFTER HAVING PREVIOUSLY ALREADY INVALIDATED THIS EVENT, the total count will be incorrect.

#### Parameters

<i>event</i>	a valid FrameEvent pointer. Cannot be NULL.
<i>packet</i>	the FrameEventPacket pointer for the packet containing this event. Cannot be NULL.

#### 4.11.5.48 PACKED\_STRUCT() [1/2]

```

PACKED_STRUCT (
    struct caer_frame_event { uint32_t info;int32_t ts_startframe;int32_t ts_endframe;int32↵
    _t ts_startexposure;int32_t ts_endexposure;int32_t lengthX;int32_t lengthY;int32_t positionX;int32↵
    _t positionY:uint16_t pixels[1];} )

```

Frame event data structure definition. This contains the actual information on the frame (ROI, color channels, color filter), several timestamps to signal start and end of capture and of exposure, as well as the actual pixels, in a 16 bit normalized format. The (0, 0) address is in the upper left corner, like in OpenCV/computer graphics. The pixel array is laid out row by row (increasing X axis), going from top to bottom (increasing Y axis). Signed integers are used for fields that are to be interpreted directly, for compatibility with languages that do not have unsigned integer types, such as Java. To copy a frame event, the usual assignment operator = cannot be used. Please use [caerGenericEventCopy\(\)](#) to copy frame events!

#### 4.11.5.49 PACKED\_STRUCT() [2/2]

```

PACKED_STRUCT (
    struct caer_frame_event_packet { struct caer_event_packet_header packetHeader;}
)

```

Frame event packet data structure definition. EventPackets are always made up of the common packet header, followed by 'eventCapacity' events. Everything has to be in one contiguous memory block. Direct access to the events array is not possible for Frame events. To calculate position offsets, use the 'eventSize' field in the packet header.

## 4.12 events/imu6.h File Reference

```
#include "common.h"
```

### Macros

- `#define CAER_IMU6_ITERATOR_ALL_START(IMU6_PACKET)`
- `#define CAER_IMU6_CONST_ITERATOR_ALL_START(IMU6_PACKET)`
- `#define CAER_IMU6_ITERATOR_ALL_END }`
- `#define CAER_IMU6_ITERATOR_VALID_START(IMU6_PACKET)`
- `#define CAER_IMU6_CONST_ITERATOR_VALID_START(IMU6_PACKET)`
- `#define CAER_IMU6_ITERATOR_VALID_END }`
- `#define CAER_IMU6_REVERSE_ITERATOR_ALL_START(IMU6_PACKET)`
- `#define CAER_IMU6_CONST_REVERSE_ITERATOR_ALL_START(IMU6_PACKET)`
- `#define CAER_IMU6_REVERSE_ITERATOR_ALL_END }`
- `#define CAER_IMU6_REVERSE_ITERATOR_VALID_START(IMU6_PACKET)`
- `#define CAER_IMU6_CONST_REVERSE_ITERATOR_VALID_START(IMU6_PACKET)`
- `#define CAER_IMU6_REVERSE_ITERATOR_VALID_END }`

### Typedefs

- `typedef struct caer_imu6_event * caerIMU6Event`
- `typedef const struct caer_imu6_event * caerIMU6EventConst`
- `typedef struct caer_imu6_event_packet * caerIMU6EventPacket`
- `typedef const struct caer_imu6_event_packet * caerIMU6EventPacketConst`

### Functions

- `PACKED_STRUCT` (struct caer\_imu6\_event { uint32\_t info;int32\_t timestamp;float accel\_x;float accel\_y;float accel\_z;float gyro\_x;float gyro\_y;float gyro\_z;float temp;})
- `PACKED_STRUCT` (struct caer\_imu6\_event\_packet { struct caer\_event\_packet\_header packetHeader;struct caer\_imu6\_event events[ ];})
- `caerIMU6EventPacket caerIMU6EventPacketAllocate` (int32\_t eventCapacity, int16\_t eventSource, int32\_t tsOverflow)
- `static caerIMU6Event caerIMU6EventPacketGetEvent` (caerIMU6EventPacket packet, int32\_t n)
- `static caerIMU6EventConst caerIMU6EventPacketGetEventConst` (caerIMU6EventPacketConst packet, int32\_t n)
- `static int32_t caerIMU6EventGetTimestamp` (caerIMU6EventConst event)
- `static int64_t caerIMU6EventGetTimestamp64` (caerIMU6EventConst event, caerIMU6EventPacketConst packet)
- `static void caerIMU6EventSetTimestamp` (caerIMU6Event event, int32\_t timestamp)
- `static bool caerIMU6EventIsValid` (caerIMU6EventConst event)
- `static void caerIMU6EventValidate` (caerIMU6Event event, caerIMU6EventPacket packet)
- `static void caerIMU6EventInvalidate` (caerIMU6Event event, caerIMU6EventPacket packet)
- `static float caerIMU6EventGetAccelX` (caerIMU6EventConst event)
- `static void caerIMU6EventSetAccelX` (caerIMU6Event event, float accelX)
- `static float caerIMU6EventGetAccelY` (caerIMU6EventConst event)
- `static void caerIMU6EventSetAccelY` (caerIMU6Event event, float accelY)
- `static float caerIMU6EventGetAccelZ` (caerIMU6EventConst event)
- `static void caerIMU6EventSetAccelZ` (caerIMU6Event event, float accelZ)



- static float [caerIMU6EventGetGyroX](#) ([caerIMU6EventConst](#) event)
- static void [caerIMU6EventSetGyroX](#) ([caerIMU6Event](#) event, float gyroX)
- static float [caerIMU6EventGetGyroY](#) ([caerIMU6EventConst](#) event)
- static void [caerIMU6EventSetGyroY](#) ([caerIMU6Event](#) event, float gyroY)
- static float [caerIMU6EventGetGyroZ](#) ([caerIMU6EventConst](#) event)
- static void [caerIMU6EventSetGyroZ](#) ([caerIMU6Event](#) event, float gyroZ)
- static float [caerIMU6EventGetTemp](#) ([caerIMU6EventConst](#) event)
- static void [caerIMU6EventSetTemp](#) ([caerIMU6Event](#) event, float temp)

### 4.12.1 Detailed Description

IMU6 (6 axes) Events format definition and handling functions. This contains data coming from the Inertial Measurement Unit chip, with the 3-axes accelerometer and 3-axes gyroscope. Temperature is also included.

### 4.12.2 Macro Definition Documentation

#### 4.12.2.1 CAER\_IMU6\_CONST\_ITERATOR\_ALL\_START

```
#define CAER_IMU6_CONST_ITERATOR_ALL_START(  
    IMU6_PACKET )
```

**Value:**

```
for (int32_t caerIMU6IteratorCounter = 0; \
     caerIMU6IteratorCounter < caerEventPacketHeaderGetEventNumber(&(
IMU6_PACKET)->packetHeader); \
     caerIMU6IteratorCounter++) { \
     caerIMU6EventConst caerIMU6IteratorElement =
caerIMU6EventPacketGetEventConst(IMU6_PACKET, caerIMU6IteratorCounter);
```

Const-Iterator over all IMU6 events in a packet. Returns the current index in the 'caerIMU6IteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerIMU6IteratorElement' variable of type caerIMU6EventConst.

IMU6\_PACKET: a valid IMU6EventPacket pointer. Cannot be NULL.

#### 4.12.2.2 CAER\_IMU6\_CONST\_ITERATOR\_VALID\_START

```
#define CAER_IMU6_CONST_ITERATOR_VALID_START(  
    IMU6_PACKET )
```

**Value:**

```
for (int32_t caerIMU6IteratorCounter = 0; \
     caerIMU6IteratorCounter < caerEventPacketHeaderGetEventNumber(&(
IMU6_PACKET)->packetHeader); \
     caerIMU6IteratorCounter++) { \
     caerIMU6EventConst caerIMU6IteratorElement =
caerIMU6EventPacketGetEventConst(IMU6_PACKET, caerIMU6IteratorCounter); \
     if (!caerIMU6EventIsValid(caerIMU6IteratorElement)) { continue; }
```

Const-Iterator over only the valid IMU6 events in a packet. Returns the current index in the 'caerIMU6IteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerIMU6IteratorElement' variable of type caerIMU6EventConst.

IMU6\_PACKET: a valid IMU6EventPacket pointer. Cannot be NULL.

#### 4.12.2.3 CAER\_IMU6\_CONST\_REVERSE\_ITERATOR\_ALL\_START

```
#define CAER_IMU6_CONST_REVERSE_ITERATOR_ALL_START(
    IMU6_PACKET )
```

**Value:**

```
for (int32_t caerIMU6IteratorCounter = caerEventPacketHeaderGetEventNumber
    (&(IMU6_PACKET)->packetHeader) - 1; \
    caerIMU6IteratorCounter >= 0; \
    caerIMU6IteratorCounter--) { \
    caerIMU6EventConst caerIMU6IteratorElement =
    caerIMU6EventPacketGetEventConst(IMU6_PACKET, caerIMU6IteratorCounter);
```

Const-Reverse iterator over all IMU6 events in a packet. Returns the current index in the 'caerIMU6IteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerIMU6IteratorElement' variable of type caerIMU6EventConst.

IMU6\_PACKET: a valid IMU6EventPacket pointer. Cannot be NULL.

#### 4.12.2.4 CAER\_IMU6\_CONST\_REVERSE\_ITERATOR\_VALID\_START

```
#define CAER_IMU6_CONST_REVERSE_ITERATOR_VALID_START(
    IMU6_PACKET )
```

**Value:**

```
for (int32_t caerIMU6IteratorCounter = caerEventPacketHeaderGetEventNumber
    (&(IMU6_PACKET)->packetHeader) - 1; \
    caerIMU6IteratorCounter >= 0; \
    caerIMU6IteratorCounter--) { \
    caerIMU6EventConst caerIMU6IteratorElement =
    caerIMU6EventPacketGetEventConst(IMU6_PACKET, caerIMU6IteratorCounter); \
    if (!caerIMU6EventIsValid(caerIMU6IteratorElement)) { continue; }
```

Const-Reverse iterator over only the valid IMU6 events in a packet. Returns the current index in the 'caerIMU6IteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerIMU6IteratorElement' variable of type caerIMU6EventConst.

IMU6\_PACKET: a valid IMU6EventPacket pointer. Cannot be NULL.

#### 4.12.2.5 CAER\_IMU6\_ITERATOR\_ALL\_END

```
#define CAER_IMU6_ITERATOR_ALL_END }
```

Iterator close statement.

## 4.12.2.6 CAER\_IMU6\_ITERATOR\_ALL\_START

```
#define CAER_IMU6_ITERATOR_ALL_START(  
    IMU6_PACKET )
```

**Value:**

```
for (int32_t caerIMU6IteratorCounter = 0; \  
    caerIMU6IteratorCounter < caerEventPacketHeaderGetEventNumber(&(\  
    IMU6_PACKET)->packetHeader); \  
    caerIMU6IteratorCounter++) { \  
    caerIMU6Event caerIMU6IteratorElement = caerIMU6EventPacketGetEvent(  
    IMU6_PACKET, caerIMU6IteratorCounter);
```

Iterator over all IMU6 events in a packet. Returns the current index in the 'caerIMU6IteratorCounter' variable of type 'int32\_t' and the current event in the 'caerIMU6IteratorElement' variable of type caerIMU6Event.

IMU6\_PACKET: a valid IMU6EventPacket pointer. Cannot be NULL.

## 4.12.2.7 CAER\_IMU6\_ITERATOR\_VALID\_END

```
#define CAER_IMU6_ITERATOR_VALID_END }
```

Iterator close statement.

## 4.12.2.8 CAER\_IMU6\_ITERATOR\_VALID\_START

```
#define CAER_IMU6_ITERATOR_VALID_START(  
    IMU6_PACKET )
```

**Value:**

```
for (int32_t caerIMU6IteratorCounter = 0; \  
    caerIMU6IteratorCounter < caerEventPacketHeaderGetEventNumber(&(\  
    IMU6_PACKET)->packetHeader); \  
    caerIMU6IteratorCounter++) { \  
    caerIMU6Event caerIMU6IteratorElement = caerIMU6EventPacketGetEvent(  
    IMU6_PACKET, caerIMU6IteratorCounter); \  
    if (!caerIMU6EventIsValid(caerIMU6IteratorElement)) { continue; }
```

Iterator over only the valid IMU6 events in a packet. Returns the current index in the 'caerIMU6IteratorCounter' variable of type 'int32\_t' and the current event in the 'caerIMU6IteratorElement' variable of type caerIMU6Event.

IMU6\_PACKET: a valid IMU6EventPacket pointer. Cannot be NULL.

## 4.12.2.9 CAER\_IMU6\_REVERSE\_ITERATOR\_ALL\_END

```
#define CAER_IMU6_REVERSE_ITERATOR_ALL_END }
```

Reverse iterator close statement.

#### 4.12.2.10 CAER\_IMU6\_REVERSE\_ITERATOR\_ALL\_START

```
#define CAER_IMU6_REVERSE_ITERATOR_ALL_START(
    IMU6_PACKET )
```

##### Value:

```
for (int32_t caerIMU6IteratorCounter = caerEventPacketHeaderGetEventNumber
    (&(IMU6_PACKET)->packetHeader) - 1; \
    caerIMU6IteratorCounter >= 0; \
    caerIMU6IteratorCounter--) { \
    caerIMU6Event caerIMU6IteratorElement = caerIMU6EventPacketGetEvent(
        IMU6_PACKET, caerIMU6IteratorCounter);
```

Reverse iterator over all IMU6 events in a packet. Returns the current index in the 'caerIMU6IteratorCounter' variable of type 'int32\_t' and the current event in the 'caerIMU6IteratorElement' variable of type caerIMU6Event.

IMU6\_PACKET: a valid IMU6EventPacket pointer. Cannot be NULL.

#### 4.12.2.11 CAER\_IMU6\_REVERSE\_ITERATOR\_VALID\_END

```
#define CAER_IMU6_REVERSE_ITERATOR_VALID_END }
```

Reverse iterator close statement.

#### 4.12.2.12 CAER\_IMU6\_REVERSE\_ITERATOR\_VALID\_START

```
#define CAER_IMU6_REVERSE_ITERATOR_VALID_START(
    IMU6_PACKET )
```

##### Value:

```
for (int32_t caerIMU6IteratorCounter = caerEventPacketHeaderGetEventNumber
    (&(IMU6_PACKET)->packetHeader) - 1; \
    caerIMU6IteratorCounter >= 0; \
    caerIMU6IteratorCounter--) { \
    caerIMU6Event caerIMU6IteratorElement = caerIMU6EventPacketGetEvent(
        IMU6_PACKET, caerIMU6IteratorCounter); \
    if (!caerIMU6EventIsValid(caerIMU6IteratorElement)) { continue; }
```

Reverse iterator over only the valid IMU6 events in a packet. Returns the current index in the 'caerIMU6IteratorCounter' variable of type 'int32\_t' and the current event in the 'caerIMU6IteratorElement' variable of type caerIMU6Event.

IMU6\_PACKET: a valid IMU6EventPacket pointer. Cannot be NULL.

### 4.12.3 Typedef Documentation

#### 4.12.3.1 caerIMU6Event

```
typedef struct caer_imu6_event* caerIMU6Event
```

Type for pointer to IMU 6-axes event data structure.

#### 4.12.3.2 caerIMU6EventPacket

```
typedef struct caer_imu6_event_packet* caerIMU6EventPacket
```

Type for pointer to IMU 6-axes event packet data structure.

### 4.12.4 Function Documentation

#### 4.12.4.1 caerIMU6EventGetAccelX()

```
static float caerIMU6EventGetAccelX (  
    caerIMU6EventConst event ) [inline], [static]
```

Get the X axis acceleration reading (from accelerometer). This is in g (1 g = 9.81 m/s<sup>2</sup>).

##### Parameters

<i>event</i>	a valid IMU6Event pointer. Cannot be NULL.
--------------	--

##### Returns

acceleration on the X axis.

#### 4.12.4.2 caerIMU6EventGetAccelY()

```
static float caerIMU6EventGetAccelY (  
    caerIMU6EventConst event ) [inline], [static]
```

Get the Y axis acceleration reading (from accelerometer). This is in g (1 g = 9.81 m/s<sup>2</sup>).

##### Parameters

<i>event</i>	a valid IMU6Event pointer. Cannot be NULL.
--------------	--

**Returns**

acceleration on the Y axis.

**4.12.4.3 caerIMU6EventGetAccelZ()**

```
static float caerIMU6EventGetAccelZ (
    caerIMU6EventConst event ) [inline], [static]
```

Get the Z axis acceleration reading (from accelerometer). This is in g (1 g = 9.81 m/s<sup>2</sup>).

**Parameters**

<i>event</i>	a valid IMU6Event pointer. Cannot be NULL.
--------------	--

**Returns**

acceleration on the Z axis.

**4.12.4.4 caerIMU6EventGetGyroX()**

```
static float caerIMU6EventGetGyroX (
    caerIMU6EventConst event ) [inline], [static]
```

Get the X axis (roll) angular velocity reading (from gyroscope). This is in °/s (deg/sec).

**Parameters**

<i>event</i>	a valid IMU6Event pointer. Cannot be NULL.
--------------	--

**Returns**

angular velocity on the X axis (roll).

**4.12.4.5 caerIMU6EventGetGyroY()**

```
static float caerIMU6EventGetGyroY (
    caerIMU6EventConst event ) [inline], [static]
```

Get the Y axis (pitch) angular velocity reading (from gyroscope). This is in °/s (deg/sec).

**Parameters**

<i>event</i>	a valid IMU6Event pointer. Cannot be NULL.
--------------	--

**Returns**

angular velocity on the Y axis (pitch).

**4.12.4.6 caerIMU6EventGetGyroZ()**

```
static float caerIMU6EventGetGyroZ (  
    caerIMU6EventConst event ) [inline], [static]
```

Get the Z axis (yaw) angular velocity reading (from gyroscope). This is in %s (deg/sec).

**Parameters**

<i>event</i>	a valid IMU6Event pointer. Cannot be NULL.
--------------	--

**Returns**

angular velocity on the Z axis (yaw).

**4.12.4.7 caerIMU6EventGetTemp()**

```
static float caerIMU6EventGetTemp (  
    caerIMU6EventConst event ) [inline], [static]
```

Get the temperature reading. This is in °C.

**Parameters**

<i>event</i>	a valid IMU6Event pointer. Cannot be NULL.
--------------	--

**Returns**

temperature in °C.

**4.12.4.8 caerIMU6EventGetTimestamp()**

```
static int32_t caerIMU6EventGetTimestamp (  
    caerIMU6EventConst event ) [inline], [static]
```

Get the 32bit event timestamp, in microseconds. Be aware that this wraps around! You can either ignore this fact, or handle the special 'TIMESTAMP\_WRAP' event that is generated when this happens, or use the 64bit timestamp which never wraps around. See '[caerEventPacketHeaderGetEventTSOverflow\(\)](#)' documentation for more details on the 64bit timestamp.

#### Parameters

<i>event</i>	a valid IMU6Event pointer. Cannot be NULL.
--------------	--

#### Returns

this event's 32bit microsecond timestamp.

#### 4.12.4.9 caerIMU6EventGetTimestamp64()

```
static int64_t caerIMU6EventGetTimestamp64 (
    caerIMU6EventConst event,
    caerIMU6EventPacketConst packet ) [inline], [static]
```

Get the 64bit event timestamp, in microseconds. See '[caerEventPacketHeaderGetEventTSOverflow\(\)](#)' documentation for more details on the 64bit timestamp.

#### Parameters

<i>event</i>	a valid IMU6Event pointer. Cannot be NULL.
<i>packet</i>	the IMU6EventPacket pointer for the packet containing this event. Cannot be NULL.

#### Returns

this event's 64bit microsecond timestamp.

#### 4.12.4.10 caerIMU6EventInvalidate()

```
static void caerIMU6EventInvalidate (
    caerIMU6Event event,
    caerIMU6EventPacket packet ) [inline], [static]
```

Invalidate the current event by setting its valid bit to false and decreasing the number of valid events held in the packet. Only works with events that are already valid!

#### Parameters

<i>event</i>	a valid IMU6Event pointer. Cannot be NULL.
<i>packet</i>	the IMU6EventPacket pointer for the packet containing this event. Cannot be NULL.



#### 4.12.4.11 caerIMU6EventsValid()

```
static bool caerIMU6EventIsValid (
    caerIMU6EventConst event ) [inline], [static]
```

Check if this IMU 6-axes event is valid.

##### Parameters

<i>event</i>	a valid IMU6Event pointer. Cannot be NULL.
--------------	--

##### Returns

true if valid, false if not.

#### 4.12.4.12 caerIMU6EventPacketAllocate()

```
caerIMU6EventPacket caerIMU6EventPacketAllocate (
    int32_t eventCapacity,
    int16_t eventSource,
    int32_t tsOverflow )
```

Allocate a new IMU 6-axes events packet. Use free() to reclaim this memory.

##### Parameters

<i>eventCapacity</i>	the maximum number of events this packet will hold.
<i>eventSource</i>	the unique ID representing the source/generator of this packet.
<i>tsOverflow</i>	the current timestamp overflow counter value for this packet.

##### Returns

a valid IMU6EventPacket handle or NULL on error.

#### 4.12.4.13 caerIMU6EventPacketGetEvent()

```
static caerIMU6Event caerIMU6EventPacketGetEvent (
    caerIMU6EventPacket packet,
    int32_t n ) [inline], [static]
```

Get the IMU 6-axes event at the given index from the event packet.

**Parameters**

<i>packet</i>	a valid IMU6EventPacket pointer. Cannot be NULL.
<i>n</i>	the index of the returned event. Must be within [0,eventCapacity[ bounds.

**Returns**

the requested IMU 6-axes event. NULL on error.

**4.12.4.14 caerIMU6EventPacketGetEventConst()**

```
static caerIMU6EventConst caerIMU6EventPacketGetEventConst (  
    caerIMU6EventPacketConst packet,  
    int32_t n ) [inline], [static]
```

Get the IMU 6-axes event at the given index from the event packet. This is a read-only event, do not change its contents in any way!

**Parameters**

<i>packet</i>	a valid IMU6EventPacket pointer. Cannot be NULL.
<i>n</i>	the index of the returned event. Must be within [0,eventCapacity[ bounds.

**Returns**

the requested read-only IMU 6-axes event. NULL on error.

**4.12.4.15 caerIMU6EventSetAccelX()**

```
static void caerIMU6EventSetAccelX (  
    caerIMU6Event event,  
    float accelX ) [inline], [static]
```

Set the X axis acceleration reading (from accelerometer). This is in g (1 g = 9.81 m/s<sup>2</sup>).

**Parameters**

<i>event</i>	a valid IMU6Event pointer. Cannot be NULL.
<i>accelX</i>	acceleration on the X axis.

#### 4.12.4.16 caerIMU6EventSetAccelY()

```
static void caerIMU6EventSetAccelY (
    caerIMU6Event event,
    float accelY ) [inline], [static]
```

Set the Y axis acceleration reading (from accelerometer). This is in g (1 g = 9.81 m/s<sup>2</sup>).

##### Parameters

<i>event</i>	a valid IMU6Event pointer. Cannot be NULL.
<i>accelY</i>	acceleration on the Y axis.

#### 4.12.4.17 caerIMU6EventSetAccelZ()

```
static void caerIMU6EventSetAccelZ (
    caerIMU6Event event,
    float accelZ ) [inline], [static]
```

Set the Z axis acceleration reading (from accelerometer). This is in g (1 g = 9.81 m/s<sup>2</sup>).

##### Parameters

<i>event</i>	a valid IMU6Event pointer. Cannot be NULL.
<i>accelZ</i>	acceleration on the Z axis.

#### 4.12.4.18 caerIMU6EventSetGyroX()

```
static void caerIMU6EventSetGyroX (
    caerIMU6Event event,
    float gyroX ) [inline], [static]
```

Set the X axis (roll) angular velocity reading (from gyroscope). This is in °/s (deg/sec).

##### Parameters

<i>event</i>	a valid IMU6Event pointer. Cannot be NULL.
<i>gyroX</i>	angular velocity on the X axis (roll).

#### 4.12.4.19 caerIMU6EventSetGyroY()

```
static void caerIMU6EventSetGyroY (
```

```

    caerIMU6Event event,
    float gyroY ) [inline], [static]

```

Set the Y axis (pitch) angular velocity reading (from gyroscope). This is in °/s (deg/sec).

#### Parameters

<i>event</i>	a valid IMU6Event pointer. Cannot be NULL.
<i>gyroY</i>	angular velocity on the Y axis (pitch).

#### 4.12.4.20 caerIMU6EventSetGyroZ()

```

static void caerIMU6EventSetGyroZ (
    caerIMU6Event event,
    float gyroZ ) [inline], [static]

```

Set the Z axis (yaw) angular velocity reading (from gyroscope). This is in °/s (deg/sec).

#### Parameters

<i>event</i>	a valid IMU6Event pointer. Cannot be NULL.
<i>gyroZ</i>	angular velocity on the Z axis (yaw).

#### 4.12.4.21 caerIMU6EventSetTemp()

```

static void caerIMU6EventSetTemp (
    caerIMU6Event event,
    float temp ) [inline], [static]

```

Set the temperature reading. This is in °C.

#### Parameters

<i>event</i>	a valid IMU6Event pointer. Cannot be NULL.
<i>temp</i>	temperature in °C.

#### 4.12.4.22 caerIMU6EventSetTimestamp()

```

static void caerIMU6EventSetTimestamp (
    caerIMU6Event event,
    int32_t timestamp ) [inline], [static]

```

Set the 32bit event timestamp, the value has to be in microseconds.

## Parameters

<i>event</i>	a valid IMU6Event pointer. Cannot be NULL.
<i>timestamp</i>	a positive 32bit microsecond timestamp.

## 4.12.4.23 caerIMU6EventValidate()

```
static void caerIMU6EventValidate (
    caerIMU6Event event,
    caerIMU6EventPacket packet ) [inline], [static]
```

Validate the current event by setting its valid bit to true and increasing the event packet's event count and valid event count. Only works on events that are invalid. DO NOT CALL THIS AFTER HAVING PREVIOUSLY ALREADY INVALIDATED THIS EVENT, the total count will be incorrect.

## Parameters

<i>event</i>	a valid IMU6Event pointer. Cannot be NULL.
<i>packet</i>	the IMU6EventPacket pointer for the packet containing this event. Cannot be NULL.

## 4.12.4.24 PACKED\_STRUCT() [1/2]

```
PACKED_STRUCT (
    struct caer_imu6_event { uint32_t info;int32_t timestamp;float accel_x;float
    accel_y;float accel_z;float gyro_x;float gyro_y;float gyro_z;float temp;} )
```

IMU 6-axes event data structure definition. This contains accelerometer and gyroscope headings, plus temperature. The X, Y and Z axes are referred to the camera plane. X increases to the right, Y going up and Z towards where the lens is pointing. Rotation for the gyroscope is counter-clockwise along the increasing axis, for all three axes. Floats are in IEEE 754-2008 binary32 format. Signed integers are used for fields that are to be interpreted directly, for compatibility with languages that do not have unsigned integer types, such as Java.

## 4.12.4.25 PACKED\_STRUCT() [2/2]

```
PACKED_STRUCT (
    struct caer_imu6_event_packet { struct caer_event_packet_header packetHeader;struct
    caer_imu6_event events[];} )
```

IMU 6-axes event packet data structure definition. EventPackets are always made up of the common packet header, followed by 'eventCapacity' events. Everything has to be in one contiguous memory block.

## 4.13 events/imu9.h File Reference

```
#include "common.h"
```

## Macros

- `#define CAER_IMU9_ITERATOR_ALL_START(IMU9_PACKET)`
- `#define CAER_IMU9_CONST_ITERATOR_ALL_START(IMU9_PACKET)`
- `#define CAER_IMU9_ITERATOR_ALL_END }`
- `#define CAER_IMU9_ITERATOR_VALID_START(IMU9_PACKET)`
- `#define CAER_IMU9_CONST_ITERATOR_VALID_START(IMU9_PACKET)`
- `#define CAER_IMU9_ITERATOR_VALID_END }`
- `#define CAER_IMU9_REVERSE_ITERATOR_ALL_START(IMU9_PACKET)`
- `#define CAER_IMU9_CONST_REVERSE_ITERATOR_ALL_START(IMU9_PACKET)`
- `#define CAER_IMU9_REVERSE_ITERATOR_ALL_END }`
- `#define CAER_IMU9_REVERSE_ITERATOR_VALID_START(IMU9_PACKET)`
- `#define CAER_IMU9_CONST_REVERSE_ITERATOR_VALID_START(IMU9_PACKET)`
- `#define CAER_IMU9_REVERSE_ITERATOR_VALID_END }`

## Typedefs

- `typedef struct caer_imu9_event * caerIMU9Event`
- `typedef const struct caer_imu9_event * caerIMU9EventConst`
- `typedef struct caer_imu9_event_packet * caerIMU9EventPacket`
- `typedef const struct caer_imu9_event_packet * caerIMU9EventPacketConst`

## Functions

- `PACKED_STRUCT` (struct caer\_imu9\_event { uint32\_t info;int32\_t timestamp;float accel\_x;float accel\_y;float accel\_z;float gyro\_x;float gyro\_y;float gyro\_z;float temp;float comp\_x;float comp\_y;float comp\_z;})
- `PACKED_STRUCT` (struct caer\_imu9\_event\_packet { struct caer\_event\_packet\_header packetHeader;struct caer\_imu9\_event events[ ];})
- `caerIMU9EventPacket caerIMU9EventPacketAllocate` (int32\_t eventCapacity, int16\_t eventSource, int32\_t tsOverflow)
- static `caerIMU9Event caerIMU9EventPacketGetEvent` (caerIMU9EventPacket packet, int32\_t n)
- static `caerIMU9EventConst caerIMU9EventPacketGetEventConst` (caerIMU9EventPacketConst packet, int32\_t n)
- static `int32_t caerIMU9EventGetTimestamp` (caerIMU9EventConst event)
- static `int64_t caerIMU9EventGetTimestamp64` (caerIMU9EventConst event, caerIMU9EventPacketConst packet)
- static void `caerIMU9EventSetTimestamp` (caerIMU9Event event, int32\_t timestamp)
- static bool `caerIMU9EventIsValid` (caerIMU9EventConst event)
- static void `caerIMU9EventValidate` (caerIMU9Event event, caerIMU9EventPacket packet)
- static void `caerIMU9EventInvalidate` (caerIMU9Event event, caerIMU9EventPacket packet)
- static float `caerIMU9EventGetAccelX` (caerIMU9EventConst event)
- static void `caerIMU9EventSetAccelX` (caerIMU9Event event, float accelX)
- static float `caerIMU9EventGetAccelY` (caerIMU9EventConst event)
- static void `caerIMU9EventSetAccelY` (caerIMU9Event event, float accelY)
- static float `caerIMU9EventGetAccelZ` (caerIMU9EventConst event)
- static void `caerIMU9EventSetAccelZ` (caerIMU9Event event, float accelZ)
- static float `caerIMU9EventGetGyroX` (caerIMU9EventConst event)
- static void `caerIMU9EventSetGyroX` (caerIMU9Event event, float gyroX)
- static float `caerIMU9EventGetGyroY` (caerIMU9EventConst event)
- static void `caerIMU9EventSetGyroY` (caerIMU9Event event, float gyroY)
- static float `caerIMU9EventGetGyroZ` (caerIMU9EventConst event)
- static void `caerIMU9EventSetGyroZ` (caerIMU9Event event, float gyroZ)
- static float `caerIMU9EventGetTemp` (caerIMU9EventConst event)

- static void [caerIMU9EventSetTemp](#) ([caerIMU9Event](#) event, float temp)
- static float [caerIMU9EventGetCompX](#) ([caerIMU9EventConst](#) event)
- static void [caerIMU9EventSetCompX](#) ([caerIMU9Event](#) event, float compX)
- static float [caerIMU9EventGetCompY](#) ([caerIMU9EventConst](#) event)
- static void [caerIMU9EventSetCompY](#) ([caerIMU9Event](#) event, float compY)
- static float [caerIMU9EventGetCompZ](#) ([caerIMU9EventConst](#) event)
- static void [caerIMU9EventSetCompZ](#) ([caerIMU9Event](#) event, float compZ)

#### 4.13.1 Detailed Description

IMU9 (9 axes) Events format definition and handling functions. This contains data coming from the Inertial Measurement Unit chip, with the 3-axes accelerometer and 3-axes gyroscope. Temperature is also included. Further, 3-axes from the magnetometer are included, which can be used to get a compass-like heading.

#### 4.13.2 Macro Definition Documentation

##### 4.13.2.1 CAER\_IMU9\_CONST\_ITERATOR\_ALL\_START

```
#define CAER_IMU9_CONST_ITERATOR_ALL_START(  
    IMU9_PACKET )
```

**Value:**

```
for (int32_t caerIMU9IteratorCounter = 0; \
     caerIMU9IteratorCounter < caerEventPacketHeaderGetEventNumber(&(
IMU9_PACKET)->packetHeader); \
     caerIMU9IteratorCounter++) { \
     caerIMU9EventConst caerIMU9IteratorElement =
caerIMU9EventPacketGetEventConst(IMU9_PACKET, caerIMU9IteratorCounter);
```

Const-Iterator over all IMU9 events in a packet. Returns the current index in the 'caerIMU9IteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerIMU9IteratorElement' variable of type caerIMU9EventConst.

IMU9\_PACKET: a valid IMU9EventPacket pointer. Cannot be NULL.

##### 4.13.2.2 CAER\_IMU9\_CONST\_ITERATOR\_VALID\_START

```
#define CAER_IMU9_CONST_ITERATOR_VALID_START(  
    IMU9_PACKET )
```

**Value:**

```
for (int32_t caerIMU9IteratorCounter = 0; \
     caerIMU9IteratorCounter < caerEventPacketHeaderGetEventNumber(&(
IMU9_PACKET)->packetHeader); \
     caerIMU9IteratorCounter++) { \
     caerIMU9EventConst caerIMU9IteratorElement =
caerIMU9EventPacketGetEventConst(IMU9_PACKET, caerIMU9IteratorCounter); \
     if (!caerIMU9EventIsValid(caerIMU9IteratorElement)) { continue; }
```

Const-Iterator over only the valid IMU9 events in a packet. Returns the current index in the 'caerIMU9IteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerIMU9IteratorElement' variable of type caerIMU9EventConst.

IMU9\_PACKET: a valid IMU9EventPacket pointer. Cannot be NULL.

#### 4.13.2.3 CAER\_IMU9\_CONST\_REVERSE\_ITERATOR\_ALL\_START

```
#define CAER_IMU9_CONST_REVERSE_ITERATOR_ALL_START(
    IMU9_PACKET )
```

**Value:**

```
for (int32_t caerIMU9IteratorCounter = caerEventPacketHeaderGetEventNumber
    (&(IMU9_PACKET)->packetHeader) - 1; \
    caerIMU9IteratorCounter >= 0; \
    caerIMU9IteratorCounter--) { \
    caerIMU9EventConst caerIMU9IteratorElement =
    caerIMU9EventPacketGetEventConst(IMU9_PACKET, caerIMU9IteratorCounter);
```

Const-Reverse iterator over all IMU9 events in a packet. Returns the current index in the 'caerIMU9IteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerIMU9IteratorElement' variable of type caerIMU9EventConst.

IMU9\_PACKET: a valid IMU9EventPacket pointer. Cannot be NULL.

#### 4.13.2.4 CAER\_IMU9\_CONST\_REVERSE\_ITERATOR\_VALID\_START

```
#define CAER_IMU9_CONST_REVERSE_ITERATOR_VALID_START(
    IMU9_PACKET )
```

**Value:**

```
for (int32_t caerIMU9IteratorCounter = caerEventPacketHeaderGetEventNumber
    (&(IMU9_PACKET)->packetHeader) - 1; \
    caerIMU9IteratorCounter >= 0; \
    caerIMU9IteratorCounter--) { \
    caerIMU9EventConst caerIMU9IteratorElement =
    caerIMU9EventPacketGetEventConst(IMU9_PACKET, caerIMU9IteratorCounter); \
    if (!caerIMU9EventIsValid(caerIMU9IteratorElement)) { continue; }
```

Const-Reverse iterator over only the valid IMU9 events in a packet. Returns the current index in the 'caerIMU9IteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerIMU9IteratorElement' variable of type caerIMU9EventConst.

IMU9\_PACKET: a valid IMU9EventPacket pointer. Cannot be NULL.

#### 4.13.2.5 CAER\_IMU9\_ITERATOR\_ALL\_END

```
#define CAER_IMU9_ITERATOR_ALL_END }
```

Iterator close statement.



## 4.13.2.6 CAER\_IMU9\_ITERATOR\_ALL\_START

```
#define CAER_IMU9_ITERATOR_ALL_START(  
    IMU9_PACKET )
```

**Value:**

```
for (int32_t caerIMU9IteratorCounter = 0; \  
    caerIMU9IteratorCounter < caerEventPacketHeaderGetEventNumber(&(\  
    IMU9_PACKET)->packetHeader); \  
    caerIMU9IteratorCounter++) { \  
    caerIMU9Event caerIMU9IteratorElement = caerIMU9EventPacketGetEvent(  
    IMU9_PACKET, caerIMU9IteratorCounter);
```

Iterator over all IMU9 events in a packet. Returns the current index in the 'caerIMU9IteratorCounter' variable of type 'int32\_t' and the current event in the 'caerIMU9IteratorElement' variable of type caerIMU9Event.

IMU9\_PACKET: a valid IMU9EventPacket pointer. Cannot be NULL.

## 4.13.2.7 CAER\_IMU9\_ITERATOR\_VALID\_END

```
#define CAER_IMU9_ITERATOR_VALID_END }
```

Iterator close statement.

## 4.13.2.8 CAER\_IMU9\_ITERATOR\_VALID\_START

```
#define CAER_IMU9_ITERATOR_VALID_START(  
    IMU9_PACKET )
```

**Value:**

```
for (int32_t caerIMU9IteratorCounter = 0; \  
    caerIMU9IteratorCounter < caerEventPacketHeaderGetEventNumber(&(\  
    IMU9_PACKET)->packetHeader); \  
    caerIMU9IteratorCounter++) { \  
    caerIMU9Event caerIMU9IteratorElement = caerIMU9EventPacketGetEvent(  
    IMU9_PACKET, caerIMU9IteratorCounter); \  
    if (!caerIMU9EventIsValid(caerIMU9IteratorElement)) { continue; }
```

Iterator over only the valid IMU9 events in a packet. Returns the current index in the 'caerIMU9IteratorCounter' variable of type 'int32\_t' and the current event in the 'caerIMU9IteratorElement' variable of type caerIMU9Event.

IMU9\_PACKET: a valid IMU9EventPacket pointer. Cannot be NULL.

## 4.13.2.9 CAER\_IMU9\_REVERSE\_ITERATOR\_ALL\_END

```
#define CAER_IMU9_REVERSE_ITERATOR_ALL_END }
```

Reverse iterator close statement.

#### 4.13.2.10 CAER\_IMU9\_REVERSE\_ITERATOR\_ALL\_START

```
#define CAER_IMU9_REVERSE_ITERATOR_ALL_START(
    IMU9_PACKET )
```

**Value:**

```
for (int32_t caerIMU9IteratorCounter = caerEventPacketHeaderGetEventNumber
    (&(IMU9_PACKET)->packetHeader) - 1; \
    caerIMU9IteratorCounter >= 0; \
    caerIMU9IteratorCounter--) { \
    caerIMU9Event caerIMU9IteratorElement = caerIMU9EventPacketGetEvent(
        IMU9_PACKET, caerIMU9IteratorCounter);
```

Reverse iterator over all IMU9 events in a packet. Returns the current index in the 'caerIMU9IteratorCounter' variable of type 'int32\_t' and the current event in the 'caerIMU9IteratorElement' variable of type caerIMU9Event.

IMU9\_PACKET: a valid IMU9EventPacket pointer. Cannot be NULL.

#### 4.13.2.11 CAER\_IMU9\_REVERSE\_ITERATOR\_VALID\_END

```
#define CAER_IMU9_REVERSE_ITERATOR_VALID_END }
```

Reverse iterator close statement.

#### 4.13.2.12 CAER\_IMU9\_REVERSE\_ITERATOR\_VALID\_START

```
#define CAER_IMU9_REVERSE_ITERATOR_VALID_START(
    IMU9_PACKET )
```

**Value:**

```
for (int32_t caerIMU9IteratorCounter = caerEventPacketHeaderGetEventNumber
    (&(IMU9_PACKET)->packetHeader) - 1; \
    caerIMU9IteratorCounter >= 0; \
    caerIMU9IteratorCounter--) { \
    caerIMU9Event caerIMU9IteratorElement = caerIMU9EventPacketGetEvent(
        IMU9_PACKET, caerIMU9IteratorCounter); \
    if (!caerIMU9EventIsValid(caerIMU9IteratorElement)) { continue; }
```

Reverse iterator over only the valid IMU9 events in a packet. Returns the current index in the 'caerIMU9IteratorCounter' variable of type 'int32\_t' and the current event in the 'caerIMU9IteratorElement' variable of type caerIMU9Event.

IMU9\_PACKET: a valid IMU9EventPacket pointer. Cannot be NULL.

### 4.13.3 Typedef Documentation

#### 4.13.3.1 caerIMU9Event

```
typedef struct caer_imu9_event* caerIMU9Event
```

Type for pointer to IMU 9-axes event data structure.

#### 4.13.3.2 caerIMU9EventPacket

```
typedef struct caer_imu9_event_packet* caerIMU9EventPacket
```

Type for pointer to IMU 9-axes event packet data structure.

### 4.13.4 Function Documentation

#### 4.13.4.1 caerIMU9EventGetAccelX()

```
static float caerIMU9EventGetAccelX (  
    caerIMU9EventConst event ) [inline], [static]
```

Get the X axis acceleration reading (from accelerometer). This is in g (1 g = 9.81 m/s<sup>2</sup>).

##### Parameters

<i>event</i>	a valid IMU9Event pointer. Cannot be NULL.
--------------	--

##### Returns

acceleration on the X axis.

#### 4.13.4.2 caerIMU9EventGetAccelY()

```
static float caerIMU9EventGetAccelY (  
    caerIMU9EventConst event ) [inline], [static]
```

Get the Y axis acceleration reading (from accelerometer). This is in g (1 g = 9.81 m/s<sup>2</sup>).

##### Parameters

<i>event</i>	a valid IMU9Event pointer. Cannot be NULL.
--------------	--

**Returns**

acceleration on the Y axis.

**4.13.4.3 caerIMU9EventGetAccelZ()**

```
static float caerIMU9EventGetAccelZ (
    caerIMU9EventConst event ) [inline], [static]
```

Get the Z axis acceleration reading (from accelerometer). This is in g (1 g = 9.81 m/s<sup>2</sup>).

**Parameters**

<i>event</i>	a valid IMU9Event pointer. Cannot be NULL.
--------------	--

**Returns**

acceleration on the Z axis.

**4.13.4.4 caerIMU9EventGetCompX()**

```
static float caerIMU9EventGetCompX (
    caerIMU9EventConst event ) [inline], [static]
```

Get the X axis compass heading (from magnetometer). This is in  $\mu$ T.

**Parameters**

<i>event</i>	a valid IMU9Event pointer. Cannot be NULL.
--------------	--

**Returns**

X axis compass heading.

**4.13.4.5 caerIMU9EventGetCompY()**

```
static float caerIMU9EventGetCompY (
    caerIMU9EventConst event ) [inline], [static]
```

Get the Y axis compass heading (from magnetometer). This is in  $\mu$ T.

**Parameters**

<i>event</i>	a valid IMU9Event pointer. Cannot be NULL.
--------------	--

**Returns**

Y axis compass heading.

**4.13.4.6 caerIMU9EventGetCompZ()**

```
static float caerIMU9EventGetCompZ (  
    caerIMU9EventConst event ) [inline], [static]
```

Get the Z axis compass heading (from magnetometer). This is in  $\mu\text{T}$ .

**Parameters**

<i>event</i>	a valid IMU9Event pointer. Cannot be NULL.
--------------	--

**Returns**

Z axis compass heading.

**4.13.4.7 caerIMU9EventGetGyroX()**

```
static float caerIMU9EventGetGyroX (  
    caerIMU9EventConst event ) [inline], [static]
```

Get the X axis (roll) angular velocity reading (from gyroscope). This is in  $^{\circ}/\text{s}$  (deg/sec).

**Parameters**

<i>event</i>	a valid IMU9Event pointer. Cannot be NULL.
--------------	--

**Returns**

angular velocity on the X axis (roll).

**4.13.4.8 caerIMU9EventGetGyroY()**

```
static float caerIMU9EventGetGyroY (  
    caerIMU9EventConst event ) [inline], [static]
```

Get the Y axis (pitch) angular velocity reading (from gyroscope). This is in °/s (deg/sec).

**Parameters**

<i>event</i>	a valid IMU9Event pointer. Cannot be NULL.
--------------	--

**Returns**

angular velocity on the Y axis (pitch).

**4.13.4.9 caerIMU9EventGetGyroZ()**

```
static float caerIMU9EventGetGyroZ (  
    caerIMU9EventConst event ) [inline], [static]
```

Get the Z axis (yaw) angular velocity reading (from gyroscope). This is in %s (deg/sec).

**Parameters**

<i>event</i>	a valid IMU9Event pointer. Cannot be NULL.
--------------	--

**Returns**

angular velocity on the Z axis (yaw).

**4.13.4.10 caerIMU9EventGetTemp()**

```
static float caerIMU9EventGetTemp (  
    caerIMU9EventConst event ) [inline], [static]
```

Get the temperature reading. This is in °C.

**Parameters**

<i>event</i>	a valid IMU9Event pointer. Cannot be NULL.
--------------	--

**Returns**

temperature in °C.

**4.13.4.11 caerIMU9EventGetTimestamp()**

```
static int32_t caerIMU9EventGetTimestamp (  
    caerIMU9EventConst event ) [inline], [static]
```

Get the 32bit event timestamp, in microseconds. Be aware that this wraps around! You can either ignore this fact, or handle the special 'TIMESTAMP\_WRAP' event that is generated when this happens, or use the 64bit timestamp which never wraps around. See '[caerEventPacketHeaderGetEventTSOverflow\(\)](#)' documentation for more details on the 64bit timestamp.

#### Parameters

<i>event</i>	a valid IMU9Event pointer. Cannot be NULL.
--------------	--

#### Returns

this event's 32bit microsecond timestamp.

#### 4.13.4.12 caerIMU9EventGetTimestamp64()

```
static int64_t caerIMU9EventGetTimestamp64 (
    caerIMU9EventConst event,
    caerIMU9EventPacketConst packet ) [inline], [static]
```

Get the 64bit event timestamp, in microseconds. See '[caerEventPacketHeaderGetEventTSOverflow\(\)](#)' documentation for more details on the 64bit timestamp.

#### Parameters

<i>event</i>	a valid IMU9Event pointer. Cannot be NULL.
<i>packet</i>	the IMU9EventPacket pointer for the packet containing this event. Cannot be NULL.

#### Returns

this event's 64bit microsecond timestamp.

#### 4.13.4.13 caerIMU9EventInvalidate()

```
static void caerIMU9EventInvalidate (
    caerIMU9Event event,
    caerIMU9EventPacket packet ) [inline], [static]
```

Invalidate the current event by setting its valid bit to false and decreasing the number of valid events held in the packet. Only works with events that are already valid!

#### Parameters

<i>event</i>	a valid IMU9Event pointer. Cannot be NULL.
<i>packet</i>	the IMU9EventPacket pointer for the packet containing this event. Cannot be NULL.



#### 4.13.4.14 caerIMU9EventsValid()

```
static bool caerIMU9EventIsValid (
    caerIMU9EventConst event ) [inline], [static]
```

Check if this IMU 9-axes event is valid.

##### Parameters

<i>event</i>	a valid IMU9Event pointer. Cannot be NULL.
--------------	--

##### Returns

true if valid, false if not.

#### 4.13.4.15 caerIMU9EventPacketAllocate()

```
caerIMU9EventPacket caerIMU9EventPacketAllocate (
    int32_t eventCapacity,
    int16_t eventSource,
    int32_t tsOverflow )
```

Allocate a new IMU 9-axes events packet. Use free() to reclaim this memory.

##### Parameters

<i>eventCapacity</i>	the maximum number of events this packet will hold.
<i>eventSource</i>	the unique ID representing the source/generator of this packet.
<i>tsOverflow</i>	the current timestamp overflow counter value for this packet.

##### Returns

a valid IMU9EventPacket handle or NULL on error.

#### 4.13.4.16 caerIMU9EventPacketGetEvent()

```
static caerIMU9Event caerIMU9EventPacketGetEvent (
    caerIMU9EventPacket packet,
    int32_t n ) [inline], [static]
```

Get the IMU 9-axes event at the given index from the event packet.

## Parameters

<i>packet</i>	a valid IMU9EventPacket pointer. Cannot be NULL.
<i>n</i>	the index of the returned event. Must be within [0,eventCapacity[ bounds.

## Returns

the requested IMU 9-axes event. NULL on error.

## 4.13.4.17 caerIMU9EventPacketGetEventConst()

```
static caerIMU9EventConst caerIMU9EventPacketGetEventConst (
    caerIMU9EventPacketConst packet,
    int32_t n ) [inline], [static]
```

Get the IMU 9-axes event at the given index from the event packet. This is a read-only event, do not change its contents in any way!

## Parameters

<i>packet</i>	a valid IMU9EventPacket pointer. Cannot be NULL.
<i>n</i>	the index of the returned event. Must be within [0,eventCapacity[ bounds.

## Returns

the requested read-only IMU 9-axes event. NULL on error.

## 4.13.4.18 caerIMU9EventSetAccelX()

```
static void caerIMU9EventSetAccelX (
    caerIMU9Event event,
    float accelX ) [inline], [static]
```

Set the X axis acceleration reading (from accelerometer). This is in g (1 g = 9.81 m/s<sup>2</sup>).

## Parameters

<i>event</i>	a valid IMU9Event pointer. Cannot be NULL.
<i>accelX</i>	acceleration on the X axis.

#### 4.13.4.19 caerIMU9EventSetAccelY()

```
static void caerIMU9EventSetAccelY (
    caerIMU9Event event,
    float accelY ) [inline], [static]
```

Set the Y axis acceleration reading (from accelerometer). This is in g (1 g = 9.81 m/s<sup>2</sup>).

##### Parameters

<i>event</i>	a valid IMU9Event pointer. Cannot be NULL.
<i>accelY</i>	acceleration on the Y axis.

#### 4.13.4.20 caerIMU9EventSetAccelZ()

```
static void caerIMU9EventSetAccelZ (
    caerIMU9Event event,
    float accelZ ) [inline], [static]
```

Set the Z axis acceleration reading (from accelerometer). This is in g (1 g = 9.81 m/s<sup>2</sup>).

##### Parameters

<i>event</i>	a valid IMU9Event pointer. Cannot be NULL.
<i>accelZ</i>	acceleration on the Z axis.

#### 4.13.4.21 caerIMU9EventSetCompX()

```
static void caerIMU9EventSetCompX (
    caerIMU9Event event,
    float compX ) [inline], [static]
```

Set the X axis compass heading (from magnetometer). This is in  $\mu$ T.

##### Parameters

<i>event</i>	a valid IMU9Event pointer. Cannot be NULL.
<i>compX</i>	X axis compass heading.

#### 4.13.4.22 caerIMU9EventSetCompY()

```
static void caerIMU9EventSetCompY (
```

```

    caerIMU9Event event,
    float compY ) [inline], [static]

```

Set the Y axis compass heading (from magnetometer). This is in  $\mu\text{T}$ .

#### Parameters

<i>event</i>	a valid IMU9Event pointer. Cannot be NULL.
<i>compY</i>	Y axis compass heading.

#### 4.13.4.23 caerIMU9EventSetCompZ()

```

static void caerIMU9EventSetCompZ (
    caerIMU9Event event,
    float compZ ) [inline], [static]

```

Set the Z axis compass heading (from magnetometer). This is in  $\mu\text{T}$ .

#### Parameters

<i>event</i>	a valid IMU9Event pointer. Cannot be NULL.
<i>compZ</i>	Z axis compass heading.

#### 4.13.4.24 caerIMU9EventSetGyroX()

```

static void caerIMU9EventSetGyroX (
    caerIMU9Event event,
    float gyroX ) [inline], [static]

```

Set the X axis (roll) angular velocity reading (from gyroscope). This is in  $\text{°/s}$  (deg/sec).

#### Parameters

<i>event</i>	a valid IMU9Event pointer. Cannot be NULL.
<i>gyroX</i>	angular velocity on the X axis (roll).

#### 4.13.4.25 caerIMU9EventSetGyroY()

```

static void caerIMU9EventSetGyroY (
    caerIMU9Event event,
    float gyroY ) [inline], [static]

```

Set the Y axis (pitch) angular velocity reading (from gyroscope). This is in  $\text{°/s}$  (deg/sec).

## Parameters

<i>event</i>	a valid IMU9Event pointer. Cannot be NULL.
<i>gyroY</i>	angular velocity on the Y axis (pitch).

## 4.13.4.26 caerIMU9EventSetGyroZ()

```
static void caerIMU9EventSetGyroZ (  
    caerIMU9Event event,  
    float gyroZ ) [inline], [static]
```

Set the Z axis (yaw) angular velocity reading (from gyroscope). This is in %s (deg/sec).

## Parameters

<i>event</i>	a valid IMU9Event pointer. Cannot be NULL.
<i>gyroZ</i>	angular velocity on the Z axis (yaw).

## 4.13.4.27 caerIMU9EventSetTemp()

```
static void caerIMU9EventSetTemp (  
    caerIMU9Event event,  
    float temp ) [inline], [static]
```

Set the temperature reading. This is in °C.

## Parameters

<i>event</i>	a valid IMU9Event pointer. Cannot be NULL.
<i>temp</i>	temperature in °C.

## 4.13.4.28 caerIMU9EventSetTimestamp()

```
static void caerIMU9EventSetTimestamp (  
    caerIMU9Event event,  
    int32_t timestamp ) [inline], [static]
```

Set the 32bit event timestamp, the value has to be in microseconds.

## Parameters

<i>event</i>	a valid IMU9Event pointer. Cannot be NULL.
<i>timestamp</i>	a positive 32bit microsecond timestamp.

#### 4.13.4.29 caerIMU9EventValidate()

```
static void caerIMU9EventValidate (
    caerIMU9Event event,
    caerIMU9EventPacket packet ) [inline], [static]
```

Validate the current event by setting its valid bit to true and increasing the event packet's event count and valid event count. Only works on events that are invalid. DO NOT CALL THIS AFTER HAVING PREVIOUSLY ALREADY INVALIDATED THIS EVENT, the total count will be incorrect.

##### Parameters

<i>event</i>	a valid IMU9Event pointer. Cannot be NULL.
<i>packet</i>	the IMU9EventPacket pointer for the packet containing this event. Cannot be NULL.

#### 4.13.4.30 PACKED\_STRUCT() [1/2]

```
PACKED_STRUCT (
    struct caer_imu9_event { uint32_t info;int32_t timestamp;float accel_x;float
    accel_y;float accel_z;float gyro_x;float gyro_y;float gyro_z;float temp;float comp_x;float
    comp_y;float comp_z;} )
```

IMU 9-axes event data structure definition. This contains accelerometer and gyroscope headings, plus temperature, and magnetometer readings. The X, Y and Z axes are referred to the camera plane. X increases to the right, Y going up and Z towards where the lens is pointing. Rotation for the gyroscope is counter-clockwise along the increasing axis, for all three axes. Floats are in IEEE 754-2008 binary32 format. Signed integers are used for fields that are to be interpreted directly, for compatibility with languages that do not have unsigned integer types, such as Java.

#### 4.13.4.31 PACKED\_STRUCT() [2/2]

```
PACKED_STRUCT (
    struct caer_imu9_event_packet { struct caer_event_packet_header packetHeader;struct
    caer_imu9_event events[];}
```

IMU 9-axes event packet data structure definition. EventPackets are always made up of the common packet header, followed by 'eventCapacity' events. Everything has to be in one contiguous memory block.

## 4.14 events/packetContainer.h File Reference

```
#include "common.h"
```

## Macros

- `#define CAER_EVENT_PACKET_CONTAINER_ITERATOR_START(PACKET_CONTAINER)`
- `#define CAER_EVENT_PACKET_CONTAINER_CONST_ITERATOR_START(PACKET_CONTAINER)`
- `#define CAER_EVENT_PACKET_CONTAINER_ITERATOR_END }`

## Typedefs

- `typedef struct caer_event_packet_container * caerEventPacketContainer`
- `typedef const struct caer_event_packet_container * caerEventPacketContainerConst`

## Functions

- `PACKED_STRUCT` (struct caer\_event\_packet\_container { int64\_t lowestEventTimestamp;int64\_t highestEventTimestamp;int32\_t eventsNumber;int32\_t eventsValidNumber;int32\_t eventPacketsNumber;caerEventPacketHeader eventPackets[ ];})
- `caerEventPacketContainer caerEventPacketContainerAllocate` (int32\_t eventPacketsNumber)
- `void caerEventPacketContainerFree` (caerEventPacketContainer container)
- `static void caerEventPacketContainerUpdateStatistics` (caerEventPacketContainer container)
- `static int32_t caerEventPacketContainerGetEventPacketsNumber` (caerEventPacketContainerConst container)
- `static void caerEventPacketContainerSetEventPacketsNumber` (caerEventPacketContainer container, int32\_t eventPacketsNumber)
- `static caerEventPacketHeader caerEventPacketContainerGetEventPacket` (caerEventPacketContainerConst container, int32\_t n)
- `static caerEventPacketHeaderConst caerEventPacketContainerGetEventPacketConst` (caerEventPacketContainerConst container, int32\_t n)
- `static void caerEventPacketContainerSetEventPacket` (caerEventPacketContainer container, int32\_t n, caerEventPacketHeader packetHeader)
- `static int64_t caerEventPacketContainerGetLowestEventTimestamp` (caerEventPacketContainerConst container)
- `static int64_t caerEventPacketContainerGetHighestEventTimestamp` (caerEventPacketContainerConst container)
- `static int32_t caerEventPacketContainerGetEventsNumber` (caerEventPacketContainerConst container)
- `static int32_t caerEventPacketContainerGetEventsValidNumber` (caerEventPacketContainerConst container)
- `static caerEventPacketHeader caerEventPacketContainerFindEventPacketByType` (caerEventPacketContainerConst container, int16\_t typeId)
- `static caerEventPacketHeaderConst caerEventPacketContainerFindEventPacketByTypeConst` (caerEventPacketContainerConst container, int16\_t typeId)
- `static caerEventPacketContainer caerEventPacketContainerCopyAllEvents` (caerEventPacketContainerConst container)
- `static caerEventPacketContainer caerEventPacketContainerCopyValidEvents` (caerEventPacketContainerConst container)

### 4.14.1 Detailed Description

EventPacketContainer format definition and handling functions. An EventPacketContainer is a logical construct that contains packets of events (EventPackets) of different event types, with the aim of keeping related events of differing types, such as DVS and IMU data, together. Such a relation is usually based on time intervals, trying to keep groups of event happening in a certain time-slice together. This time-order is based on the *main* time-stamp of an event, the one whose offset is referenced in the event packet header and that is used by the caerGenericEvent\*() functions. It's guaranteed that all conforming input modules keep to this rule, generating containers that include all events from all types within the given time-slice. The smallest and largest timestamps are tracked at the packet container level as a convenience, to avoid having to examine all packets for this often useful piece of information. All integers are in their native host format, as this is a purely internal, in-memory data structure, never meant for exchange between different systems (and different endianness).

== Packet Containers and Input Modules == The "packeting system" works in this way: events are accumulated by type in a packet, and that packet is part of a packet container, by an input module. The packet container is then sent out for processing when either the configured time limit or the size limit are hit. The time limit is always active, in microseconds, and basically tells you the time-span an event packet covers. This enables regular, constant delivery of packets, that cover a period of time. The size limit is an add-on to prevent packets to grow to immense sizes (like if the time limit is high and there is lots of activity). As soon as a packet hits the number of events in the size limit, it is sent out. The regular time limit is not reset in this case. This size limit can be disabled by setting it to 0. The cAER DVS128/DAVIS/File/Network input modules call these two configuration variables "PacketContainerInterval" and "PacketContainerMaxPacketSize". Too small packet sizes or intervals simply mean more packets, which may negatively affect performance. It's usually a good idea to set the size to something around 4-8K, and the time to a good value based on the application you're building, so if you need ms-reaction-time, you probably want to set it to 1000µs, so that you do get new data every ms. If on the other hand you're looking at a static scene and just want to detect that something is passing by once every while, a higher number like 100ms might also be perfectly appropriate.

### 4.14.2 Macro Definition Documentation

#### 4.14.2.1 CAER\_EVENT\_PACKET\_CONTAINER\_CONST\_ITERATOR\_START

```
#define CAER_EVENT_PACKET_CONTAINER_CONST_ITERATOR_START(
    PACKET_CONTAINER )
```

**Value:**

```
if ((PACKET_CONTAINER) != NULL) { \
    for (int32_t caerEventPacketContainerIteratorCounter = 0; \
         caerEventPacketContainerIteratorCounter < \
         caerEventPacketContainerGetEventPacketsNumber(PACKET_CONTAINER); \
         caerEventPacketContainerIteratorCounter++) { \
        caerEventPacketHeaderConst caerEventPacketContainerIteratorElement = \
        caerEventPacketContainerGetEventPacketConst(PACKET_CONTAINER, \
        caerEventPacketContainerIteratorCounter); \
        if (caerEventPacketContainerIteratorElement == NULL) { continue; }
```

Const-Iterator over all event packets in an event packet container. Returns the current index in the 'caerEventPacketContainerIteratorCounter' variable of type 'int32\_t' and the current read-only event packet in the 'caerEventPacketContainerIteratorElement' variable of type caerEventPacketHeaderConst. The current packet may be NULL, in which case it is skipped during iteration.

PACKET\_CONTAINER: a valid EventPacketContainer handle. If NULL, no iteration is performed.



#### 4.14.2.2 CAER\_EVENT\_PACKET\_CONTAINER\_ITERATOR\_END

```
#define CAER_EVENT_PACKET_CONTAINER_ITERATOR_END } }
```

Iterator close statement.

#### 4.14.2.3 CAER\_EVENT\_PACKET\_CONTAINER\_ITERATOR\_START

```
#define CAER_EVENT_PACKET_CONTAINER_ITERATOR_START(  
    PACKET_CONTAINER )
```

**Value:**

```
if ((PACKET_CONTAINER) != NULL) { \
    for (int32_t caerEventPacketContainerIteratorCounter = 0; \
        caerEventPacketContainerIteratorCounter < \
        caerEventPacketContainerGetEventPacketsNumber(PACKET_CONTAINER); \
        caerEventPacketContainerIteratorCounter++) { \
        caerEventPacketHeader caerEventPacketContainerIteratorElement = \
        caerEventPacketContainerGetEventPacket(PACKET_CONTAINER, \
        caerEventPacketContainerIteratorCounter); \
        if (caerEventPacketContainerIteratorElement == NULL) { continue; }
```

Iterator over all event packets in an event packet container. Returns the current index in the 'caerEventPacketContainerIteratorCounter' variable of type 'int32\_t' and the current event packet in the 'caerEventPacketContainerIteratorElement' variable of type caerEventPacketHeader. The current packet may be NULL, in which case it is skipped during iteration.

PACKET\_CONTAINER: a valid EventPacketContainer handle. If NULL, no iteration is performed.

### 4.14.3 Typedef Documentation

#### 4.14.3.1 caerEventPacketContainer

```
typedef struct caer_event_packet_container* caerEventPacketContainer
```

Type for pointer to EventPacketContainer data structure.

### 4.14.4 Function Documentation

#### 4.14.4.1 caerEventPacketContainerAllocate()

```
caerEventPacketContainer caerEventPacketContainerAllocate (  
    int32_t eventPacketsNumber )
```

Allocate a new EventPacketContainer with enough space to store up to the given number of EventPacket pointers. All packet pointers will be NULL initially.

**Parameters**

<i>eventPacketsNumber</i>	the maximum number of EventPacket pointers that can be stored in this container.
---------------------------	--

**Returns**

a valid EventPacketContainer handle or NULL on error.

**4.14.4.2 caerEventPacketContainerCopyAllEvents()**

```
static caerEventPacketContainer caerEventPacketContainerCopyAllEvents (  
    caerEventPacketContainerConst container ) [inline], [static]
```

Make a deep copy of an event packet container and all of its event packets and their current events.

**Parameters**

<i>container</i>	an event packet container to copy.
------------------	------------------------------------

**Returns**

a deep copy of an event packet container, containing all events.

**4.14.4.3 caerEventPacketContainerCopyValidEvents()**

```
static caerEventPacketContainer caerEventPacketContainerCopyValidEvents (  
    caerEventPacketContainerConst container ) [inline], [static]
```

Make a deep copy of an event packet container, with its event packets sized down to only include the currently valid events (eventValid), and discarding everything else.

**Parameters**

<i>container</i>	an event packet container to copy.
------------------	------------------------------------

**Returns**

a deep copy of an event packet container, containing only valid events.

#### 4.14.4.4 caerEventPacketContainerFindEventPacketByType()

```
static caerEventPacketHeader caerEventPacketContainerFindEventPacketByType (
    caerEventPacketContainerConst container,
    int16_t typeID ) [inline], [static]
```

Get the pointer to an EventPacket stored in this container with the given event type. This returns the first found event packet with that type ID, or NULL if we get to the end without finding any such event packet.

##### Parameters

<i>container</i>	a valid EventPacketContainer handle. If NULL, returns NULL too.
<i>typeID</i>	the event type to search for.

##### Returns

a pointer to an EventPacket with a certain type or NULL if none found.

#### 4.14.4.5 caerEventPacketContainerFindEventPacketByTypeConst()

```
static caerEventPacketHeaderConst caerEventPacketContainerFindEventPacketByTypeConst (
    caerEventPacketContainerConst container,
    int16_t typeID ) [inline], [static]
```

Get the pointer to a read-only EventPacket stored in this container with the given event type. This returns the first found event packet with that type ID, or NULL if we get to the end without finding any such event packet.

##### Parameters

<i>container</i>	a valid EventPacketContainer handle. If NULL, returns NULL too.
<i>typeID</i>	the event type to search for.

##### Returns

a pointer to a read-only EventPacket with a certain type or NULL if none found.

#### 4.14.4.6 caerEventPacketContainerFree()

```
void caerEventPacketContainerFree (
    caerEventPacketContainer container )
```

Free the memory occupied by an EventPacketContainer, as well as freeing all of its contained EventPackets and their memory. If you don't want the contained EventPackets to be freed, make sure that you set their pointers to NULL before calling this.

## Parameters

<i>container</i>	the container to be freed.
------------------	----------------------------

4.14.4.7 `caerEventPacketContainerGetEventPacket()`

```
static caerEventPacketHeader caerEventPacketContainerGetEventPacket (
    caerEventPacketContainerConst container,
    int32_t n ) [inline], [static]
```

Get the pointer to the EventPacket stored in this container at the given index.

## Parameters

<i>container</i>	a valid EventPacketContainer handle. If NULL, returns NULL too.
<i>n</i>	the index of the EventPacket to get.

## Returns

a pointer to an EventPacket or NULL on error.

4.14.4.8 `caerEventPacketContainerGetEventPacketConst()`

```
static caerEventPacketHeaderConst caerEventPacketContainerGetEventPacketConst (
    caerEventPacketContainerConst container,
    int32_t n ) [inline], [static]
```

Get the pointer to the EventPacket stored in this container at the given index. This is a read-only EventPacket, do not change its contents in any way!

## Parameters

<i>container</i>	a valid EventPacketContainer handle. If NULL, returns NULL too.
<i>n</i>	the index of the EventPacket to get.

## Returns

a pointer to a read-only EventPacket or NULL on error.

4.14.4.9 `caerEventPacketContainerGetEventPacketsNumber()`

```
static int32_t caerEventPacketContainerGetEventPacketsNumber (
    caerEventPacketContainerConst container ) [inline], [static]
```

Get the maximum number of EventPacket pointers that can be stored in this particular EventPacketContainer.

#### Parameters

<i>container</i>	a valid EventPacketContainer handle. If NULL, zero is returned.
------------------	---

#### Returns

the number of EventPacket pointers that can be contained.

#### 4.14.4.10 caerEventPacketContainerGetEventsNumber()

```
static int32_t caerEventPacketContainerGetEventsNumber (  
    caerEventPacketContainerConst container ) [inline], [static]
```

Get the number of events contained in this event packet container.

#### Parameters

<i>container</i>	a valid EventPacketContainer handle. If NULL, 0 is returned.
------------------	--

#### Returns

the number of events in this container.

#### 4.14.4.11 caerEventPacketContainerGetEventsValidNumber()

```
static int32_t caerEventPacketContainerGetEventsValidNumber (  
    caerEventPacketContainerConst container ) [inline], [static]
```

Get the number of valid events contained in this event packet container.

#### Parameters

<i>container</i>	a valid EventPacketContainer handle. If NULL, 0 is returned.
------------------	--

#### Returns

the number of valid events in this container.

#### 4.14.4.12 caerEventPacketContainerGetHighestEventTimestamp()

```
static int64_t caerEventPacketContainerGetHighestEventTimestamp (
    caerEventPacketContainerConst container ) [inline], [static]
```

Get the highest timestamp contained in this event packet container.

##### Parameters

<i>container</i>	a valid EventPacketContainer handle. If NULL, -1 is returned.
------------------	---

##### Returns

the highest timestamp (in  $\mu$ s) or -1 if not initialized.

#### 4.14.4.13 caerEventPacketContainerGetLowestEventTimestamp()

```
static int64_t caerEventPacketContainerGetLowestEventTimestamp (
    caerEventPacketContainerConst container ) [inline], [static]
```

Get the lowest timestamp contained in this event packet container.

##### Parameters

<i>container</i>	a valid EventPacketContainer handle. If NULL, -1 is returned.
------------------	---

##### Returns

the lowest timestamp (in  $\mu$ s) or -1 if not initialized.

#### 4.14.4.14 caerEventPacketContainerSetEventPacket()

```
static void caerEventPacketContainerSetEventPacket (
    caerEventPacketContainer container,
    int32_t n,
    caerEventPacketHeader packetHeader ) [inline], [static]
```

Set the pointer to the EventPacket stored in this container at the given index.

##### Parameters

<i>container</i>	a valid EventPacketContainer handle. If NULL, nothing happens.
<i>n</i>	the index of the EventPacket to set.
<i>packetHeader</i>	a pointer to an EventPacket's header. Can be NULL.

## 4.14.4.15 caerEventPacketContainerSetEventPacketsNumber()

```
static void caerEventPacketContainerSetEventPacketsNumber (
    caerEventPacketContainer container,
    int32_t eventPacketsNumber ) [inline], [static]
```

Set the maximum number of EventPacket pointers that can be stored in this particular EventPacketContainer. This should never be used directly, [caerEventPacketContainerAllocate\(\)](#) sets this for you.

## Parameters

<i>container</i>	a valid EventPacketContainer handle. If NULL, nothing happens.
<i>eventPacketsNumber</i>	the number of EventPacket pointers that can be contained.

## 4.14.4.16 caerEventPacketContainerUpdateStatistics()

```
static void caerEventPacketContainerUpdateStatistics (
    caerEventPacketContainer container ) [inline], [static]
```

Recalculates and updates all the packet-container level statistics (event counts and timestamps).

## Parameters

<i>container</i>	a valid EventPacketContainer handle. If NULL, nothing happens.
------------------	--

## 4.14.4.17 PACKED\_STRUCT()

```
PACKED_STRUCT (
    struct caer_event_packet_container { int64_t lowestEventTimestamp;int64_t highest↔
EventTimestamp;int32_t eventsNumber;int32_t eventsValidNumber;int32_t eventPacketsNumber;caer↔
EventPacketHeader eventPackets[]; } )
```

EventPacketContainer data structure definition. Signed integers are used for compatibility with languages that do not have unsigned ones, such as Java.

## 4.15 events/point1d.h File Reference

```
#include "common.h"
```

## Macros

- `#define CAER_POINT1D_ITERATOR_ALL_START(PPOINT1D_PACKET)`
  - `#define CAER_POINT1D_CONST_ITERATOR_ALL_START(PPOINT1D_PACKET)`
  - `#define CAER_POINT1D_ITERATOR_ALL_END }`
  - `#define CAER_POINT1D_ITERATOR_VALID_START(PPOINT1D_PACKET)`
  - `#define CAER_POINT1D_CONST_ITERATOR_VALID_START(PPOINT1D_PACKET)`
  - `#define CAER_POINT1D_ITERATOR_VALID_END }`
  - `#define CAER_POINT1D_REVERSE_ITERATOR_ALL_START(PPOINT1D_PACKET)`
  - `#define CAER_POINT1D_CONST_REVERSE_ITERATOR_ALL_START(PPOINT1D_PACKET)`
  - `#define CAER_POINT1D_REVERSE_ITERATOR_ALL_END }`
  - `#define CAER_POINT1D_REVERSE_ITERATOR_VALID_START(PPOINT1D_PACKET)`
  - `#define CAER_POINT1D_CONST_REVERSE_ITERATOR_VALID_START(PPOINT1D_PACKET)`
  - `#define CAER_POINT1D_REVERSE_ITERATOR_VALID_END }`
- 
- `#define POINT1D_TYPE_SHIFT 1`
  - `#define POINT1D_TYPE_MASK 0x0000007F`
  - `#define POINT1D_SCALE_SHIFT 8`
  - `#define POINT1D_SCALE_MASK 0x000000FF`

## Typedefs

- `typedef struct caer_point1d_event * caerPoint1DEvent`
- `typedef const struct caer_point1d_event * caerPoint1DEventConst`
- `typedef struct caer_point1d_event_packet * caerPoint1DEventPacket`
- `typedef const struct caer_point1d_event_packet * caerPoint1DEventPacketConst`

## Functions

- `PACKED_STRUCT` (struct caer\_point1d\_event { uint32\_t info;float x;int32\_t timestamp;})
- `PACKED_STRUCT` (struct caer\_point1d\_event\_packet { struct caer\_event\_packet\_header packet←Header;struct caer\_point1d\_event events[;];})
- `caerPoint1DEventPacket caerPoint1DEventPacketAllocate` (int32\_t eventCapacity, int16\_t eventSource, int32\_t tsOverflow)
- `static caerPoint1DEvent caerPoint1DEventPacketGetEvent` (caerPoint1DEventPacket packet, int32\_t n)
- `static caerPoint1DEventConst caerPoint1DEventPacketGetEventConst` (caerPoint1DEventPacketConst packet, int32\_t n)
- `static int32_t caerPoint1DEventGetTimestamp` (caerPoint1DEventConst event)
- `static int64_t caerPoint1DEventGetTimestamp64` (caerPoint1DEventConst event, caerPoint1DEventPacket←Const packet)
- `static void caerPoint1DEventSetTimestamp` (caerPoint1DEvent event, int32\_t timestamp)
- `static bool caerPoint1DEventIsValid` (caerPoint1DEventConst event)
- `static void caerPoint1DEventValidate` (caerPoint1DEvent event, caerPoint1DEventPacket packet)
- `static void caerPoint1DEventInvalidate` (caerPoint1DEvent event, caerPoint1DEventPacket packet)
- `static uint8_t caerPoint1DEventGetType` (caerPoint1DEventConst event)
- `static void caerPoint1DEventSetType` (caerPoint1DEvent event, uint8\_t type)
- `static int8_t caerPoint1DEventGetScale` (caerPoint1DEventConst event)
- `static void caerPoint1DEventSetScale` (caerPoint1DEvent event, int8\_t scale)
- `static float caerPoint1DEventGetX` (caerPoint1DEventConst event)
- `static void caerPoint1DEventSetX` (caerPoint1DEvent event, float x)



### 4.15.1 Detailed Description

THIS EVENT DEFINITIONS IS STILL TO BE CONSIDERED EXPERIMENTAL AND IS SUBJECT TO FUTURE CHANGES AND REVISIONS!

Point1D Events format definition and handling functions. This contains one dimensional data points as floats, together with support for distinguishing type and scale.

### 4.15.2 Macro Definition Documentation

#### 4.15.2.1 CAER\_POINT1D\_CONST\_ITERATOR\_ALL\_START

```
#define CAER_POINT1D_CONST_ITERATOR_ALL_START(  
    POINT1D_PACKET )
```

**Value:**

```
for (int32_t caerPoint1DIteratorCounter = 0; \  
    caerPoint1DIteratorCounter < caerEventPacketHeaderGetEventNumber  
    (&(POINT1D_PACKET)->packetHeader); \  
    caerPoint1DIteratorCounter++) { \  
    caerPoint1DEventConst caerPoint1DIteratorElement =  
    caerPoint1DEventPacketGetEventConst (POINT1D_PACKET,  
    caerPoint1DIteratorCounter);
```

Const-Iterator over all Point1D events in a packet. Returns the current index in the 'caerPoint1DIteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerPoint1DIteratorElement' variable of type caer↵ Point1DEventConst.

POINT1D\_PACKET: a valid Point1DEventPacket pointer. Cannot be NULL.

#### 4.15.2.2 CAER\_POINT1D\_CONST\_ITERATOR\_VALID\_START

```
#define CAER_POINT1D_CONST_ITERATOR_VALID_START(  
    POINT1D_PACKET )
```

**Value:**

```
for (int32_t caerPoint1DIteratorCounter = 0; \  
    caerPoint1DIteratorCounter < caerEventPacketHeaderGetEventNumber  
    (&(POINT1D_PACKET)->packetHeader); \  
    caerPoint1DIteratorCounter++) { \  
    caerPoint1DEventConst caerPoint1DIteratorElement =  
    caerPoint1DEventPacketGetEventConst (POINT1D_PACKET,  
    caerPoint1DIteratorCounter); \  
    if (!caerPoint1DEventIsValid(caerPoint1DIteratorElement)) { continue; }
```

Const-Iterator over only the valid Point1D events in a packet. Returns the current index in the 'caerPoint1DIterator↵ Counter' variable of type 'int32\_t' and the current read-only event in the 'caerPoint1DIteratorElement' variable of type caerPoint1DEventConst.

POINT1D\_PACKET: a valid Point1DEventPacket pointer. Cannot be NULL.

#### 4.15.2.3 CAER\_POINT1D\_CONST\_REVERSE\_ITERATOR\_ALL\_START

```
#define CAER_POINT1D_CONST_REVERSE_ITERATOR_ALL_START (
    POINT1D_PACKET )
```

**Value:**

```
for (int32_t caerPoint1DIteratorCounter = caerEventPacketHeaderGetEventNumber
    (&(POINT1D_PACKET)->packetHeader) - 1; \
    caerPoint1DIteratorCounter >= 0; \
    caerPoint1DIteratorCounter--) { \
    caerPoint1DEventConst caerPoint1DIteratorElement =
    caerPoint1DEventPacketGetEventConst (POINT1D_PACKET,
    caerPoint1DIteratorCounter);
```

Const-Reverse iterator over all Point1D events in a packet. Returns the current index in the 'caerPoint1DIteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerPoint1DIteratorElement' variable of type caerPoint1DEventConst.

POINT1D\_PACKET: a valid Point1DEventPacket pointer. Cannot be NULL.

#### 4.15.2.4 CAER\_POINT1D\_CONST\_REVERSE\_ITERATOR\_VALID\_START

```
#define CAER_POINT1D_CONST_REVERSE_ITERATOR_VALID_START (
    POINT1D_PACKET )
```

**Value:**

```
for (int32_t caerPoint1DIteratorCounter = caerEventPacketHeaderGetEventNumber
    (&(POINT1D_PACKET)->packetHeader) - 1; \
    caerPoint1DIteratorCounter >= 0; \
    caerPoint1DIteratorCounter--) { \
    caerPoint1DEventConst caerPoint1DIteratorElement =
    caerPoint1DEventPacketGetEventConst (POINT1D_PACKET,
    caerPoint1DIteratorCounter); \
    if (!caerPoint1DEventIsValid(caerPoint1DIteratorElement)) { continue; }
```

Const-Reverse iterator over only the valid Point1D events in a packet. Returns the current index in the 'caerPoint1DIteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerPoint1DIteratorElement' variable of type caerPoint1DEventConst.

POINT1D\_PACKET: a valid Point1DEventPacket pointer. Cannot be NULL.

#### 4.15.2.5 CAER\_POINT1D\_ITERATOR\_ALL\_END

```
#define CAER_POINT1D_ITERATOR_ALL_END }
```

Iterator close statement.

## 4.15.2.6 CAER\_POINT1D\_ITERATOR\_ALL\_START

```
#define CAER_POINT1D_ITERATOR_ALL_START(
    POINT1D_PACKET )
```

**Value:**

```
for (int32_t caerPoint1DIteratorCounter = 0; \
    caerPoint1DIteratorCounter < caerEventPacketHeaderGetEventNumber
    (&(POINT1D_PACKET)->packetHeader); \
    caerPoint1DIteratorCounter++) { \
    caerPoint1DEvent caerPoint1DIteratorElement =
    caerPoint1DEventPacketGetEvent(POINT1D_PACKET, caerPoint1DIteratorCounter);
```

Iterator over all Point1D events in a packet. Returns the current index in the 'caerPoint1DIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerPoint1DIteratorElement' variable of type caerPoint1DEvent.

POINT1D\_PACKET: a valid Point1DEventPacket pointer. Cannot be NULL.

## 4.15.2.7 CAER\_POINT1D\_ITERATOR\_VALID\_END

```
#define CAER_POINT1D_ITERATOR_VALID_END }
```

Iterator close statement.

## 4.15.2.8 CAER\_POINT1D\_ITERATOR\_VALID\_START

```
#define CAER_POINT1D_ITERATOR_VALID_START(
    POINT1D_PACKET )
```

**Value:**

```
for (int32_t caerPoint1DIteratorCounter = 0; \
    caerPoint1DIteratorCounter < caerEventPacketHeaderGetEventNumber
    (&(POINT1D_PACKET)->packetHeader); \
    caerPoint1DIteratorCounter++) { \
    caerPoint1DEvent caerPoint1DIteratorElement =
    caerPoint1DEventPacketGetEvent(POINT1D_PACKET, caerPoint1DIteratorCounter); \
    if (!caerPoint1DEventIsValid(caerPoint1DIteratorElement)) { continue; }
```

Iterator over only the valid Point1D events in a packet. Returns the current index in the 'caerPoint1DIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerPoint1DIteratorElement' variable of type caerPoint1DEvent.

POINT1D\_PACKET: a valid Point1DEventPacket pointer. Cannot be NULL.

## 4.15.2.9 CAER\_POINT1D\_REVERSE\_ITERATOR\_ALL\_END

```
#define CAER_POINT1D_REVERSE_ITERATOR_ALL_END }
```

Reverse iterator close statement.

#### 4.15.2.10 CAER\_POINT1D\_REVERSE\_ITERATOR\_ALL\_START

```
#define CAER_POINT1D_REVERSE_ITERATOR_ALL_START(
    POINT1D_PACKET )
```

**Value:**

```
for (int32_t caerPoint1DIteratorCounter = caerEventPacketHeaderGetEventNumber
    (&(POINT1D_PACKET)->packetHeader) - 1; \
    caerPoint1DIteratorCounter >= 0; \
    caerPoint1DIteratorCounter--) { \
    caerPoint1DEvent caerPoint1DIteratorElement =
    caerPoint1DEventPacketGetEvent(POINT1D_PACKET, caerPoint1DIteratorCounter);
```

Reverse iterator over all Point1D events in a packet. Returns the current index in the 'caerPoint1DIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerPoint1DIteratorElement' variable of type caerPoint1DEvent.

POINT1D\_PACKET: a valid Point1DEventPacket pointer. Cannot be NULL.

#### 4.15.2.11 CAER\_POINT1D\_REVERSE\_ITERATOR\_VALID\_END

```
#define CAER_POINT1D_REVERSE_ITERATOR_VALID_END }
```

Reverse iterator close statement.

#### 4.15.2.12 CAER\_POINT1D\_REVERSE\_ITERATOR\_VALID\_START

```
#define CAER_POINT1D_REVERSE_ITERATOR_VALID_START(
    POINT1D_PACKET )
```

**Value:**

```
for (int32_t caerPoint1DIteratorCounter = caerEventPacketHeaderGetEventNumber
    (&(POINT1D_PACKET)->packetHeader) - 1; \
    caerPoint1DIteratorCounter >= 0; \
    caerPoint1DIteratorCounter--) { \
    caerPoint1DEvent caerPoint1DIteratorElement =
    caerPoint1DEventPacketGetEvent(POINT1D_PACKET, caerPoint1DIteratorCounter); \
    if (!caerPoint1DEventIsValid(caerPoint1DIteratorElement)) { continue; }
```

Reverse iterator over only the valid Point1D events in a packet. Returns the current index in the 'caerPoint1DIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerPoint1DIteratorElement' variable of type caerPoint1DEvent.

POINT1D\_PACKET: a valid Point1DEventPacket pointer. Cannot be NULL.

#### 4.15.2.13 POINT1D\_SCALE\_MASK

```
#define POINT1D_SCALE_MASK 0x000000FF
```

Shift and mask values for type and scale information associated with a Point1D event. Up to 128 types are supported. The scale is given as orders of magnitude, from  $10^{-128}$  to  $10^{127}$ . Bit 0 is the valid mark, see 'common.h' for more details.

#### 4.15.2.14 POINT1D\_SCALE\_SHIFT

```
#define POINT1D_SCALE_SHIFT 8
```

Shift and mask values for type and scale information associated with a Point1D event. Up to 128 types are supported. The scale is given as orders of magnitude, from  $10^{-128}$  to  $10^{127}$ . Bit 0 is the valid mark, see 'common.h' for more details.

#### 4.15.2.15 POINT1D\_TYPE\_MASK

```
#define POINT1D_TYPE_MASK 0x0000007F
```

Shift and mask values for type and scale information associated with a Point1D event. Up to 128 types are supported. The scale is given as orders of magnitude, from  $10^{-128}$  to  $10^{127}$ . Bit 0 is the valid mark, see 'common.h' for more details.

#### 4.15.2.16 POINT1D\_TYPE\_SHIFT

```
#define POINT1D_TYPE_SHIFT 1
```

Shift and mask values for type and scale information associated with a Point1D event. Up to 128 types are supported. The scale is given as orders of magnitude, from  $10^{-128}$  to  $10^{127}$ . Bit 0 is the valid mark, see 'common.h' for more details.

### 4.15.3 Typedef Documentation

#### 4.15.3.1 caerPoint1DEvent

```
typedef struct caer_point1d_event* caerPoint1DEvent
```

Type for pointer to Point1D event data structure.

#### 4.15.3.2 caerPoint1DEventPacket

```
typedef struct caer_point1d_event_packet* caerPoint1DEventPacket
```

Type for pointer to Point1D event packet data structure.

### 4.15.4 Function Documentation

#### 4.15.4.1 caerPoint1DEventGetScale()

```
static int8_t caerPoint1DEventGetScale (  
    caerPoint1DEventConst event ) [inline], [static]
```

Get the measurement scale. This allows order of magnitude shifts on the measured value to be applied automatically, such as having measurements of type Distance (meters) and storing the values as centimeters ( $10^{-2}$ ) for higher precision, but keeping that information around to allow easy changes of unit.

**Parameters**

<i>event</i>	a valid Point1DEvent pointer. Cannot be NULL.
--------------	---

**Returns**

the Point1D measurement scale.

**4.15.4.2 caerPoint1DEventGetTimestamp()**

```
static int32_t caerPoint1DEventGetTimestamp (
    caerPoint1DEventConst event ) [inline], [static]
```

Get the 32bit event timestamp, in microseconds. Be aware that this wraps around! You can either ignore this fact, or handle the special 'TIMESTAMP\_WRAP' event that is generated when this happens, or use the 64bit timestamp which never wraps around. See '[caerEventPacketHeaderGetEventTSOverflow\(\)](#)' documentation for more details on the 64bit timestamp.

**Parameters**

<i>event</i>	a valid Point1DEvent pointer. Cannot be NULL.
--------------	---

**Returns**

this event's 32bit microsecond timestamp.

**4.15.4.3 caerPoint1DEventGetTimestamp64()**

```
static int64_t caerPoint1DEventGetTimestamp64 (
    caerPoint1DEventConst event,
    caerPoint1DEventPacketConst packet ) [inline], [static]
```

Get the 64bit event timestamp, in microseconds. See '[caerEventPacketHeaderGetEventTSOverflow\(\)](#)' documentation for more details on the 64bit timestamp.

**Parameters**

<i>event</i>	a valid Point1DEvent pointer. Cannot be NULL.
<i>packet</i>	the Point1DEventPacket pointer for the packet containing this event. Cannot be NULL.

**Returns**

this event's 64bit microsecond timestamp.

#### 4.15.4.4 caerPoint1DEventGetType()

```
static uint8_t caerPoint1DEventGetType (
    caerPoint1DEventConst event ) [inline], [static]
```

Get the measurement event type. This is useful to distinguish between different measurements, for example distance or weight.

##### Parameters

<i>event</i>	a valid Point1DEvent pointer. Cannot be NULL.
--------------	---

##### Returns

the Point1D measurement type.

#### 4.15.4.5 caerPoint1DEventGetX()

```
static float caerPoint1DEventGetX (
    caerPoint1DEventConst event ) [inline], [static]
```

Get the X axis measurement.

##### Parameters

<i>event</i>	a valid Point1DEvent pointer. Cannot be NULL.
--------------	---

##### Returns

X axis measurement.

#### 4.15.4.6 caerPoint1DEventInvalidate()

```
static void caerPoint1DEventInvalidate (
    caerPoint1DEvent event,
    caerPoint1DEventPacket packet ) [inline], [static]
```

Invalidate the current event by setting its valid bit to false and decreasing the number of valid events held in the packet. Only works with events that are already valid!

##### Parameters

<i>event</i>	a valid Point1DEvent pointer. Cannot be NULL.
<i>packet</i>	the Point1DEventPacket pointer for the packet containing this event. Cannot be NULL.

#### 4.15.4.7 caerPoint1DEventIsValid()

```
static bool caerPoint1DEventIsValid (
    caerPoint1DEventConst event ) [inline], [static]
```

Check if this Point1D event is valid.

##### Parameters

<i>event</i>	a valid Point1DEvent pointer. Cannot be NULL.
--------------	---

##### Returns

true if valid, false if not.

#### 4.15.4.8 caerPoint1DEventPacketAllocate()

```
caerPoint1DEventPacket caerPoint1DEventPacketAllocate (
    int32_t eventCapacity,
    int16_t eventSource,
    int32_t tsOverflow )
```

Allocate a new Point1D events packet. Use free() to reclaim this memory.

##### Parameters

<i>eventCapacity</i>	the maximum number of events this packet will hold.
<i>eventSource</i>	the unique ID representing the source/generator of this packet.
<i>tsOverflow</i>	the current timestamp overflow counter value for this packet.

##### Returns

a valid Point1DEventPacket handle or NULL on error.

#### 4.15.4.9 caerPoint1DEventPacketGetEvent()

```
static caerPoint1DEvent caerPoint1DEventPacketGetEvent (
    caerPoint1DEventPacket packet,
    int32_t n ) [inline], [static]
```

Get the Point1D event at the given index from the event packet.



## Parameters

<i>packet</i>	a valid Point1DEventPacket pointer. Cannot be NULL.
<i>n</i>	the index of the returned event. Must be within [0,eventCapacity[ bounds.

## Returns

the requested Point1D event. NULL on error.

## 4.15.4.10 caerPoint1DEventPacketGetEventConst()

```
static caerPoint1DEventConst caerPoint1DEventPacketGetEventConst (
    caerPoint1DEventPacketConst packet,
    int32_t n ) [inline], [static]
```

Get the Point1D event at the given index from the event packet. This is a read-only event, do not change its contents in any way!

## Parameters

<i>packet</i>	a valid Point1DEventPacket pointer. Cannot be NULL.
<i>n</i>	the index of the returned event. Must be within [0,eventCapacity[ bounds.

## Returns

the requested read-only Point1D event. NULL on error.

## 4.15.4.11 caerPoint1DEventSetScale()

```
static void caerPoint1DEventSetScale (
    caerPoint1DEvent event,
    int8_t scale ) [inline], [static]
```

Set the measurement scale. This allows order of magnitude shifts on the measured value to be applied automatically, such as having measurements of type Distance (meters) and storing the values as centimeters ( $10^{-2}$ ) for higher precision, but keeping that information around to allow easy changes of unit.

## Parameters

<i>event</i>	a valid Point1DEvent pointer. Cannot be NULL.
<i>scale</i>	the Point1D measurement scale.

#### 4.15.4.12 caerPoint1DEventSetTimestamp()

```
static void caerPoint1DEventSetTimestamp (
    caerPoint1DEvent event,
    int32_t timestamp ) [inline], [static]
```

Set the 32bit event timestamp, the value has to be in microseconds.

##### Parameters

<i>event</i>	a valid Point1DEvent pointer. Cannot be NULL.
<i>timestamp</i>	a positive 32bit microsecond timestamp.

#### 4.15.4.13 caerPoint1DEventSetType()

```
static void caerPoint1DEventSetType (
    caerPoint1DEvent event,
    uint8_t type ) [inline], [static]
```

Set the measurement event type. This is useful to distinguish between different measurements, for example distance or weight.

##### Parameters

<i>event</i>	a valid Point1DEvent pointer. Cannot be NULL.
<i>type</i>	the Point1D measurement type.

#### 4.15.4.14 caerPoint1DEventSetX()

```
static void caerPoint1DEventSetX (
    caerPoint1DEvent event,
    float x ) [inline], [static]
```

Set the X axis measurement.

##### Parameters

<i>event</i>	a valid Point1DEvent pointer. Cannot be NULL.
<i>x</i>	X axis measurement.

#### 4.15.4.15 caerPoint1DEventValidate()

```
static void caerPoint1DEventValidate (
```

```
caerPoint1DEvent event,
caerPoint1DEventPacket packet ) [inline], [static]
```

Validate the current event by setting its valid bit to true and increasing the event packet's event count and valid event count. Only works on events that are invalid. DO NOT CALL THIS AFTER HAVING PREVIOUSLY ALREADY INVALIDATED THIS EVENT, the total count will be incorrect.

#### Parameters

<i>event</i>	a valid Point1DEvent pointer. Cannot be NULL.
<i>packet</i>	the Point1DEventPacket pointer for the packet containing this event. Cannot be NULL.

#### 4.15.4.16 PACKED\_STRUCT() [1/2]

```
PACKED_STRUCT (
    struct caer_point1d_event { uint32_t info;float x;int32_t timestamp;} )
```

Point1D event data structure definition. This contains information about the measurement, such as a type and a scale field, together with the usual validity mark. The one measurement (x) is stored as a float. Floats are in IEEE 754-2008 binary32 format. Signed integers are used for fields that are to be interpreted directly, for compatibility with languages that do not have unsigned integer types, such as Java.

#### 4.15.4.17 PACKED\_STRUCT() [2/2]

```
PACKED_STRUCT (
    struct caer_point1d_event_packet { struct caer_event_packet_header packetHeader;struct
caer_point1d_event events[];} )
```

Point1D event packet data structure definition. EventPackets are always made up of the common packet header, followed by 'eventCapacity' events. Everything has to be in one contiguous memory block.

## 4.16 events/point2d.h File Reference

```
#include "common.h"
```

### Macros

- `#define CAER_POINT2D_ITERATOR_ALL_START(POINT2D_PACKET)`
- `#define CAER_POINT2D_CONST_ITERATOR_ALL_START(POINT2D_PACKET)`
- `#define CAER_POINT2D_ITERATOR_ALL_END }`
- `#define CAER_POINT2D_ITERATOR_VALID_START(POINT2D_PACKET)`
- `#define CAER_POINT2D_CONST_ITERATOR_VALID_START(POINT2D_PACKET)`
- `#define CAER_POINT2D_ITERATOR_VALID_END }`
- `#define CAER_POINT2D_REVERSE_ITERATOR_ALL_START(POINT2D_PACKET)`
- `#define CAER_POINT2D_CONST_REVERSE_ITERATOR_ALL_START(POINT2D_PACKET)`
- `#define CAER_POINT2D_REVERSE_ITERATOR_ALL_END }`

- `#define CAER_POINT2D_REVERSE_ITERATOR_VALID_START(PPOINT2D_PACKET)`
- `#define CAER_POINT2D_CONST_REVERSE_ITERATOR_VALID_START(PPOINT2D_PACKET)`
- `#define CAER_POINT2D_REVERSE_ITERATOR_VALID_END }`

- `#define POINT2D_TYPE_SHIFT 1`
- `#define POINT2D_TYPE_MASK 0x0000007F`
- `#define POINT2D_SCALE_SHIFT 8`
- `#define POINT2D_SCALE_MASK 0x000000FF`

## Typedefs

- `typedef struct caer_point2d_event * caerPoint2DEvent`
- `typedef const struct caer_point2d_event * caerPoint2DEventConst`
- `typedef struct caer_point2d_event_packet * caerPoint2DEventPacket`
- `typedef const struct caer_point2d_event_packet * caerPoint2DEventPacketConst`

## Functions

- `PACKED_STRUCT` (struct caer\_point2d\_event { uint32\_t info;float x;float y;int32\_t timestamp;})
- `PACKED_STRUCT` (struct caer\_point2d\_event\_packet { struct caer\_event\_packet\_header packet; Header;struct caer\_point2d\_event events[ ];})
- `caerPoint2DEventPacket caerPoint2DEventPacketAllocate` (int32\_t eventCapacity, int16\_t eventSource, int32\_t tsOverflow)
- `static caerPoint2DEvent caerPoint2DEventPacketGetEvent` (caerPoint2DEventPacket packet, int32\_t n)
- `static caerPoint2DEventConst caerPoint2DEventPacketGetEventConst` (caerPoint2DEventPacketConst packet, int32\_t n)
- `static int32_t caerPoint2DEventGetTimestamp` (caerPoint2DEventConst event)
- `static int64_t caerPoint2DEventGetTimestamp64` (caerPoint2DEventConst event, caerPoint2DEventPacketConst packet)
- `static void caerPoint2DEventSetTimestamp` (caerPoint2DEvent event, int32\_t timestamp)
- `static bool caerPoint2DEventIsValid` (caerPoint2DEventConst event)
- `static void caerPoint2DEventValidate` (caerPoint2DEvent event, caerPoint2DEventPacket packet)
- `static void caerPoint2DEventInvalidate` (caerPoint2DEvent event, caerPoint2DEventPacket packet)
- `static uint8_t caerPoint2DEventGetType` (caerPoint2DEventConst event)
- `static void caerPoint2DEventSetType` (caerPoint2DEvent event, uint8\_t type)
- `static int8_t caerPoint2DEventGetScale` (caerPoint2DEventConst event)
- `static void caerPoint2DEventSetScale` (caerPoint2DEvent event, int8\_t scale)
- `static float caerPoint2DEventGetX` (caerPoint2DEventConst event)
- `static void caerPoint2DEventSetX` (caerPoint2DEvent event, float x)
- `static float caerPoint2DEventGetY` (caerPoint2DEventConst event)
- `static void caerPoint2DEventSetY` (caerPoint2DEvent event, float y)

### 4.16.1 Detailed Description

THIS EVENT DEFINITIONS IS STILL TO BE CONSIDERED EXPERIMENTAL AND IS SUBJECT TO FUTURE CHANGES AND REVISIONS!

Point2D Events format definition and handling functions. This contains two dimensional data points as floats, together with support for distinguishing type and scale.

## 4.16.2 Macro Definition Documentation

### 4.16.2.1 CAER\_POINT2D\_CONST\_ITERATOR\_ALL\_START

```
#define CAER_POINT2D_CONST_ITERATOR_ALL_START(  
    POINT2D_PACKET )
```

#### Value:

```
for (int32_t caerPoint2DIteratorCounter = 0; \  
    caerPoint2DIteratorCounter < caerEventPacketHeaderGetEventNumber  
    (&(POINT2D_PACKET)->packetHeader); \  
    caerPoint2DIteratorCounter++) { \  
    caerPoint2DEventConst caerPoint2DIteratorElement =  
    caerPoint2DEventPacketGetEventConst (POINT2D_PACKET,  
    caerPoint2DIteratorCounter);
```

Const-Iterator over all Point2D events in a packet. Returns the current index in the 'caerPoint2DIteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerPoint2DIteratorElement' variable of type caer↵Point2DEventConst.

POINT2D\_PACKET: a valid Point2DEventPacket pointer. Cannot be NULL.

### 4.16.2.2 CAER\_POINT2D\_CONST\_ITERATOR\_VALID\_START

```
#define CAER_POINT2D_CONST_ITERATOR_VALID_START(  
    POINT2D_PACKET )
```

#### Value:

```
for (int32_t caerPoint2DIteratorCounter = 0; \  
    caerPoint2DIteratorCounter < caerEventPacketHeaderGetEventNumber  
    (&(POINT2D_PACKET)->packetHeader); \  
    caerPoint2DIteratorCounter++) { \  
    caerPoint2DEventConst caerPoint2DIteratorElement =  
    caerPoint2DEventPacketGetEventConst (POINT2D_PACKET,  
    caerPoint2DIteratorCounter); \  
    if (!caerPoint2DEventIsValid(caerPoint2DIteratorElement)) { continue; }
```

Const-Iterator over only the valid Point2D events in a packet. Returns the current index in the 'caerPoint2DIterator↵Counter' variable of type 'int32\_t' and the current read-only event in the 'caerPoint2DIteratorElement' variable of type caerPoint2DEventConst.

POINT2D\_PACKET: a valid Point2DEventPacket pointer. Cannot be NULL.

#### 4.16.2.3 CAER\_POINT2D\_CONST\_REVERSE\_ITERATOR\_ALL\_START

```
#define CAER_POINT2D_CONST_REVERSE_ITERATOR_ALL_START (
    POINT2D_PACKET )
```

**Value:**

```
for (int32_t caerPoint2DIteratorCounter = caerEventPacketHeaderGetEventNumber
    (&(POINT2D_PACKET)->packetHeader) - 1; \
    caerPoint2DIteratorCounter >= 0; \
    caerPoint2DIteratorCounter--) { \
    caerPoint2DEventConst caerPoint2DIteratorElement =
    caerPoint2DEventPacketGetEventConst (POINT2D_PACKET,
    caerPoint2DIteratorCounter);
```

Const-Reverse iterator over all Point2D events in a packet. Returns the current index in the 'caerPoint2DIteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerPoint2DIteratorElement' variable of type caerPoint2DEventConst.

POINT2D\_PACKET: a valid Point2DEventPacket pointer. Cannot be NULL.

#### 4.16.2.4 CAER\_POINT2D\_CONST\_REVERSE\_ITERATOR\_VALID\_START

```
#define CAER_POINT2D_CONST_REVERSE_ITERATOR_VALID_START (
    POINT2D_PACKET )
```

**Value:**

```
for (int32_t caerPoint2DIteratorCounter = caerEventPacketHeaderGetEventNumber
    (&(POINT2D_PACKET)->packetHeader) - 1; \
    caerPoint2DIteratorCounter >= 0; \
    caerPoint2DIteratorCounter--) { \
    caerPoint2DEventConst caerPoint2DIteratorElement =
    caerPoint2DEventPacketGetEventConst (POINT2D_PACKET,
    caerPoint2DIteratorCounter); \
    if (!caerPoint2DEventIsValid(caerPoint2DIteratorElement)) { continue; }
```

Const-Reverse iterator over only the valid Point2D events in a packet. Returns the current index in the 'caerPoint2DIteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerPoint2DIteratorElement' variable of type caerPoint2DEventConst.

POINT2D\_PACKET: a valid Point2DEventPacket pointer. Cannot be NULL.

#### 4.16.2.5 CAER\_POINT2D\_ITERATOR\_ALL\_END

```
#define CAER_POINT2D_ITERATOR_ALL_END }
```

Iterator close statement.

## 4.16.2.6 CAER\_POINT2D\_ITERATOR\_ALL\_START

```
#define CAER_POINT2D_ITERATOR_ALL_START(
    POINT2D_PACKET )
```

**Value:**

```
for (int32_t caerPoint2DIteratorCounter = 0; \
    caerPoint2DIteratorCounter < caerEventPacketHeaderGetEventNumber
    (&(POINT2D_PACKET)->packetHeader); \
    caerPoint2DIteratorCounter++) { \
    caerPoint2DEvent caerPoint2DIteratorElement =
    caerPoint2DEventPacketGetEvent(POINT2D_PACKET, caerPoint2DIteratorCounter);
```

Iterator over all Point2D events in a packet. Returns the current index in the 'caerPoint2DIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerPoint2DIteratorElement' variable of type caerPoint2DEvent.

POINT2D\_PACKET: a valid Point2DEventPacket pointer. Cannot be NULL.

## 4.16.2.7 CAER\_POINT2D\_ITERATOR\_VALID\_END

```
#define CAER_POINT2D_ITERATOR_VALID_END }
```

Iterator close statement.

## 4.16.2.8 CAER\_POINT2D\_ITERATOR\_VALID\_START

```
#define CAER_POINT2D_ITERATOR_VALID_START(
    POINT2D_PACKET )
```

**Value:**

```
for (int32_t caerPoint2DIteratorCounter = 0; \
    caerPoint2DIteratorCounter < caerEventPacketHeaderGetEventNumber
    (&(POINT2D_PACKET)->packetHeader); \
    caerPoint2DIteratorCounter++) { \
    caerPoint2DEvent caerPoint2DIteratorElement =
    caerPoint2DEventPacketGetEvent(POINT2D_PACKET, caerPoint2DIteratorCounter); \
    if (!caerPoint2DEventIsValid(caerPoint2DIteratorElement)) { continue; }
```

Iterator over only the valid Point2D events in a packet. Returns the current index in the 'caerPoint2DIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerPoint2DIteratorElement' variable of type caerPoint2DEvent.

POINT2D\_PACKET: a valid Point2DEventPacket pointer. Cannot be NULL.

## 4.16.2.9 CAER\_POINT2D\_REVERSE\_ITERATOR\_ALL\_END

```
#define CAER_POINT2D_REVERSE_ITERATOR_ALL_END }
```

Reverse iterator close statement.

#### 4.16.2.10 CAER\_POINT2D\_REVERSE\_ITERATOR\_ALL\_START

```
#define CAER_POINT2D_REVERSE_ITERATOR_ALL_START(
    POINT2D_PACKET )
```

**Value:**

```
for (int32_t caerPoint2DIteratorCounter = caerEventPacketHeaderGetEventNumber
    (&(POINT2D_PACKET)->packetHeader) - 1; \
    caerPoint2DIteratorCounter >= 0; \
    caerPoint2DIteratorCounter--) { \
    caerPoint2DEvent caerPoint2DIteratorElement =
    caerPoint2DEventPacketGetEvent(POINT2D_PACKET, caerPoint2DIteratorCounter);
```

Reverse iterator over all Point2D events in a packet. Returns the current index in the 'caerPoint2DIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerPoint2DIteratorElement' variable of type caerPoint2DEvent.

POINT2D\_PACKET: a valid Point2DEventPacket pointer. Cannot be NULL.

#### 4.16.2.11 CAER\_POINT2D\_REVERSE\_ITERATOR\_VALID\_END

```
#define CAER_POINT2D_REVERSE_ITERATOR_VALID_END }
```

Reverse iterator close statement.

#### 4.16.2.12 CAER\_POINT2D\_REVERSE\_ITERATOR\_VALID\_START

```
#define CAER_POINT2D_REVERSE_ITERATOR_VALID_START(
    POINT2D_PACKET )
```

**Value:**

```
for (int32_t caerPoint2DIteratorCounter = caerEventPacketHeaderGetEventNumber
    (&(POINT2D_PACKET)->packetHeader) - 1; \
    caerPoint2DIteratorCounter >= 0; \
    caerPoint2DIteratorCounter--) { \
    caerPoint2DEvent caerPoint2DIteratorElement =
    caerPoint2DEventPacketGetEvent(POINT2D_PACKET, caerPoint2DIteratorCounter); \
    if (!caerPoint2DEventIsValid(caerPoint2DIteratorElement)) { continue; }
```

Reverse iterator over only the valid Point2D events in a packet. Returns the current index in the 'caerPoint2DIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerPoint2DIteratorElement' variable of type caerPoint2DEvent.

POINT2D\_PACKET: a valid Point2DEventPacket pointer. Cannot be NULL.

#### 4.16.2.13 POINT2D\_SCALE\_MASK

```
#define POINT2D_SCALE_MASK 0x000000FF
```

Shift and mask values for type and scale information associated with a Point2D event. Up to 128 types are supported. The scale is given as orders of magnitude, from  $10^{-128}$  to  $10^{127}$ . Bit 0 is the valid mark, see 'common.h' for more details.



#### 4.16.2.14 POINT2D\_SCALE\_SHIFT

```
#define POINT2D_SCALE_SHIFT 8
```

Shift and mask values for type and scale information associated with a Point2D event. Up to 128 types are supported. The scale is given as orders of magnitude, from  $10^{-128}$  to  $10^{127}$ . Bit 0 is the valid mark, see '[common.h](#)' for more details.

#### 4.16.2.15 POINT2D\_TYPE\_MASK

```
#define POINT2D_TYPE_MASK 0x0000007F
```

Shift and mask values for type and scale information associated with a Point2D event. Up to 128 types are supported. The scale is given as orders of magnitude, from  $10^{-128}$  to  $10^{127}$ . Bit 0 is the valid mark, see '[common.h](#)' for more details.

#### 4.16.2.16 POINT2D\_TYPE\_SHIFT

```
#define POINT2D_TYPE_SHIFT 1
```

Shift and mask values for type and scale information associated with a Point2D event. Up to 128 types are supported. The scale is given as orders of magnitude, from  $10^{-128}$  to  $10^{127}$ . Bit 0 is the valid mark, see '[common.h](#)' for more details.

### 4.16.3 Typedef Documentation

#### 4.16.3.1 caerPoint2DEvent

```
typedef struct caer_point2d_event* caerPoint2DEvent
```

Type for pointer to Point2D event data structure.

#### 4.16.3.2 caerPoint2DEventPacket

```
typedef struct caer_point2d_event_packet* caerPoint2DEventPacket
```

Type for pointer to Point2D event packet data structure.

### 4.16.4 Function Documentation

#### 4.16.4.1 caerPoint2DEventGetScale()

```
static int8_t caerPoint2DEventGetScale (  
    caerPoint2DEventConst event ) [inline], [static]
```

Get the measurement scale. This allows order of magnitude shifts on the measured value to be applied automatically, such as having measurements of type Distance (meters) and storing the values as centimeters ( $10^{-2}$ ) for higher precision, but keeping that information around to allow easy changes of unit.

**Parameters**

<i>event</i>	a valid Point2DEvent pointer. Cannot be NULL.
--------------	---

**Returns**

the Point2D measurement scale.

**4.16.4.2 caerPoint2DEventGetTimestamp()**

```
static int32_t caerPoint2DEventGetTimestamp (
    caerPoint2DEventConst event ) [inline], [static]
```

Get the 32bit event timestamp, in microseconds. Be aware that this wraps around! You can either ignore this fact, or handle the special 'TIMESTAMP\_WRAP' event that is generated when this happens, or use the 64bit timestamp which never wraps around. See '[caerEventPacketHeaderGetEventTSOverflow\(\)](#)' documentation for more details on the 64bit timestamp.

**Parameters**

<i>event</i>	a valid Point2DEvent pointer. Cannot be NULL.
--------------	---

**Returns**

this event's 32bit microsecond timestamp.

**4.16.4.3 caerPoint2DEventGetTimestamp64()**

```
static int64_t caerPoint2DEventGetTimestamp64 (
    caerPoint2DEventConst event,
    caerPoint2DEventPacketConst packet ) [inline], [static]
```

Get the 64bit event timestamp, in microseconds. See '[caerEventPacketHeaderGetEventTSOverflow\(\)](#)' documentation for more details on the 64bit timestamp.

**Parameters**

<i>event</i>	a valid Point2DEvent pointer. Cannot be NULL.
<i>packet</i>	the Point2DEventPacket pointer for the packet containing this event. Cannot be NULL.

**Returns**

this event's 64bit microsecond timestamp.

#### 4.16.4.4 caerPoint2DEventGetType()

```
static uint8_t caerPoint2DEventGetType (
    caerPoint2DEventConst event ) [inline], [static]
```

Get the measurement event type. This is useful to distinguish between different measurements, for example distance or weight.

##### Parameters

<i>event</i>	a valid Point2DEvent pointer. Cannot be NULL.
--------------	---

##### Returns

the Point2D measurement type.

#### 4.16.4.5 caerPoint2DEventGetX()

```
static float caerPoint2DEventGetX (
    caerPoint2DEventConst event ) [inline], [static]
```

Get the X axis measurement.

##### Parameters

<i>event</i>	a valid Point2DEvent pointer. Cannot be NULL.
--------------	---

##### Returns

X axis measurement.

#### 4.16.4.6 caerPoint2DEventGetY()

```
static float caerPoint2DEventGetY (
    caerPoint2DEventConst event ) [inline], [static]
```

Get the Y axis measurement.

##### Parameters

<i>event</i>	a valid Point2DEvent pointer. Cannot be NULL.
--------------	---

**Returns**

Y axis measurement.

**4.16.4.7 caerPoint2DEventInvalidate()**

```
static void caerPoint2DEventInvalidate (
    caerPoint2DEvent event,
    caerPoint2DEventPacket packet ) [inline], [static]
```

Invalidate the current event by setting its valid bit to false and decreasing the number of valid events held in the packet. Only works with events that are already valid!

**Parameters**

<i>event</i>	a valid Point2DEvent pointer. Cannot be NULL.
<i>packet</i>	the Point2DEventPacket pointer for the packet containing this event. Cannot be NULL.

**4.16.4.8 caerPoint2DEventIsValid()**

```
static bool caerPoint2DEventIsValid (
    caerPoint2DEventConst event ) [inline], [static]
```

Check if this Point2D event is valid.

**Parameters**

<i>event</i>	a valid Point2DEvent pointer. Cannot be NULL.
--------------	---

**Returns**

true if valid, false if not.

**4.16.4.9 caerPoint2DEventPacketAllocate()**

```
caerPoint2DEventPacket caerPoint2DEventPacketAllocate (
    int32_t eventCapacity,
    int16_t eventSource,
    int32_t tsOverflow )
```

Allocate a new Point2D events packet. Use free() to reclaim this memory.

## Parameters

<i>eventCapacity</i>	the maximum number of events this packet will hold.
<i>eventSource</i>	the unique ID representing the source/generator of this packet.
<i>tsOverflow</i>	the current timestamp overflow counter value for this packet.

## Returns

a valid Point2DEventPacket handle or NULL on error.

## 4.16.4.10 caerPoint2DEventPacketGetEvent()

```
static caerPoint2DEvent caerPoint2DEventPacketGetEvent (
    caerPoint2DEventPacket packet,
    int32_t n ) [inline], [static]
```

Get the Point2D event at the given index from the event packet.

## Parameters

<i>packet</i>	a valid Point2DEventPacket pointer. Cannot be NULL.
<i>n</i>	the index of the returned event. Must be within [0,eventCapacity[ bounds.

## Returns

the requested Point2D event. NULL on error.

## 4.16.4.11 caerPoint2DEventPacketGetEventConst()

```
static caerPoint2DEventConst caerPoint2DEventPacketGetEventConst (
    caerPoint2DEventPacketConst packet,
    int32_t n ) [inline], [static]
```

Get the Point2D event at the given index from the event packet. This is a read-only event, do not change its contents in any way!

## Parameters

<i>packet</i>	a valid Point2DEventPacket pointer. Cannot be NULL.
<i>n</i>	the index of the returned event. Must be within [0,eventCapacity[ bounds.

**Returns**

the requested read-only Point2D event. NULL on error.

**4.16.4.12 caerPoint2DEventSetScale()**

```
static void caerPoint2DEventSetScale (
    caerPoint2DEvent event,
    int8_t scale ) [inline], [static]
```

Set the measurement scale. This allows order of magnitude shifts on the measured value to be applied automatically, such as having measurements of type Distance (meters) and storing the values as centimeters ( $10^{-2}$ ) for higher precision, but keeping that information around to allow easy changes of unit.

**Parameters**

<i>event</i>	a valid Point2DEvent pointer. Cannot be NULL.
<i>scale</i>	the Point2D measurement scale.

**4.16.4.13 caerPoint2DEventSetTimestamp()**

```
static void caerPoint2DEventSetTimestamp (
    caerPoint2DEvent event,
    int32_t timestamp ) [inline], [static]
```

Set the 32bit event timestamp, the value has to be in microseconds.

**Parameters**

<i>event</i>	a valid Point2DEvent pointer. Cannot be NULL.
<i>timestamp</i>	a positive 32bit microsecond timestamp.

**4.16.4.14 caerPoint2DEventSetType()**

```
static void caerPoint2DEventSetType (
    caerPoint2DEvent event,
    uint8_t type ) [inline], [static]
```

Set the measurement event type. This is useful to distinguish between different measurements, for example distance or weight.

**Parameters**

<i>event</i>	a valid Point2DEvent pointer. Cannot be NULL.
<i>type</i>	the Point2D measurement type.

#### 4.16.4.15 caerPoint2DEventSetX()

```
static void caerPoint2DEventSetX (
    caerPoint2DEvent event,
    float x ) [inline], [static]
```

Set the X axis measurement.

##### Parameters

<i>event</i>	a valid Point2DEvent pointer. Cannot be NULL.
<i>x</i>	X axis measurement.

#### 4.16.4.16 caerPoint2DEventSetY()

```
static void caerPoint2DEventSetY (
    caerPoint2DEvent event,
    float y ) [inline], [static]
```

Set the Y axis measurement.

##### Parameters

<i>event</i>	a valid Point2DEvent pointer. Cannot be NULL.
<i>y</i>	Y axis measurement.

#### 4.16.4.17 caerPoint2DEventValidate()

```
static void caerPoint2DEventValidate (
    caerPoint2DEvent event,
    caerPoint2DEventPacket packet ) [inline], [static]
```

Validate the current event by setting its valid bit to true and increasing the event packet's event count and valid event count. Only works on events that are invalid. DO NOT CALL THIS AFTER HAVING PREVIOUSLY ALREADY INVALIDATED THIS EVENT, the total count will be incorrect.

##### Parameters

<i>event</i>	a valid Point2DEvent pointer. Cannot be NULL.
<i>packet</i>	the Point2DEventPacket pointer for the packet containing this event. Cannot be NULL.

#### 4.16.4.18 PACKED\_STRUCT() [1/2]

```
PACKED_STRUCT (
    struct caer_point2d_event { uint32_t info;float x;float y;int32_t timestamp;} )
```

Point2D event data structure definition. This contains information about the measurement, such as a type and a scale field, together with the usual validity mark. The two measurements (x, y) are stored as floats. Floats are in IEEE 754-2008 binary32 format. Signed integers are used for fields that are to be interpreted directly, for compatibility with languages that do not have unsigned integer types, such as Java.

#### 4.16.4.19 PACKED\_STRUCT() [2/2]

```
PACKED_STRUCT (
    struct caer_point2d_event_packet { struct caer_event_packet_header packetHeader;struct
    caer_point2d_event events[];} )
```

Point2D event packet data structure definition. EventPackets are always made up of the common packet header, followed by 'eventCapacity' events. Everything has to be in one contiguous memory block.

## 4.17 events/point3d.h File Reference

```
#include "common.h"
```

### Macros

- `#define CAER_POINT3D_ITERATOR_ALL_START(POINT3D_PACKET)`
  - `#define CAER_POINT3D_CONST_ITERATOR_ALL_START(POINT3D_PACKET)`
  - `#define CAER_POINT3D_ITERATOR_ALL_END }`
  - `#define CAER_POINT3D_ITERATOR_VALID_START(POINT3D_PACKET)`
  - `#define CAER_POINT3D_CONST_ITERATOR_VALID_START(POINT3D_PACKET)`
  - `#define CAER_POINT3D_ITERATOR_VALID_END }`
  - `#define CAER_POINT3D_REVERSE_ITERATOR_ALL_START(POINT3D_PACKET)`
  - `#define CAER_POINT3D_CONST_REVERSE_ITERATOR_ALL_START(POINT3D_PACKET)`
  - `#define CAER_POINT3D_REVERSE_ITERATOR_ALL_END }`
  - `#define CAER_POINT3D_REVERSE_ITERATOR_VALID_START(POINT3D_PACKET)`
  - `#define CAER_POINT3D_CONST_REVERSE_ITERATOR_VALID_START(POINT3D_PACKET)`
  - `#define CAER_POINT3D_REVERSE_ITERATOR_VALID_END }`
- 
- `#define POINT3D_TYPE_SHIFT 1`
  - `#define POINT3D_TYPE_MASK 0x0000007F`
  - `#define POINT3D_SCALE_SHIFT 8`
  - `#define POINT3D_SCALE_MASK 0x000000FF`



## Typedefs

- typedef struct caer\_point3d\_event \* [caerPoint3DEvent](#)
- typedef const struct caer\_point3d\_event \* **caerPoint3DEventConst**
- typedef struct caer\_point3d\_event\_packet \* [caerPoint3DEventPacket](#)
- typedef const struct caer\_point3d\_event\_packet \* **caerPoint3DEventPacketConst**

## Functions

- [PACKED\\_STRUCT](#) (struct caer\_point3d\_event { uint32\_t info;float x;float y;float z;int32\_t timestamp;})
- [PACKED\\_STRUCT](#) (struct caer\_point3d\_event\_packet { struct caer\_event\_packet\_header packet;↵ Header;struct caer\_point3d\_event events[ ];})
- [caerPoint3DEventPacket](#) [caerPoint3DEventPacketAllocate](#) (int32\_t eventCapacity, int16\_t eventSource, int32\_t tsOverflow)
- static [caerPoint3DEvent](#) [caerPoint3DEventPacketGetEvent](#) ([caerPoint3DEventPacket](#) packet, int32\_t n)
- static [caerPoint3DEventConst](#) [caerPoint3DEventPacketGetEventConst](#) ([caerPoint3DEventPacketConst](#) packet, int32\_t n)
- static int32\_t [caerPoint3DEventGetTimestamp](#) ([caerPoint3DEventConst](#) event)
- static int64\_t [caerPoint3DEventGetTimestamp64](#) ([caerPoint3DEventConst](#) event, [caerPoint3DEventPacket](#)↵ Const packet)
- static void [caerPoint3DEventSetTimestamp](#) ([caerPoint3DEvent](#) event, int32\_t timestamp)
- static bool [caerPoint3DEventIsValid](#) ([caerPoint3DEventConst](#) event)
- static void [caerPoint3DEventValidate](#) ([caerPoint3DEvent](#) event, [caerPoint3DEventPacket](#) packet)
- static void [caerPoint3DEventInvalidate](#) ([caerPoint3DEvent](#) event, [caerPoint3DEventPacket](#) packet)
- static uint8\_t [caerPoint3DEventGetType](#) ([caerPoint3DEventConst](#) event)
- static void [caerPoint3DEventSetType](#) ([caerPoint3DEvent](#) event, uint8\_t type)
- static int8\_t [caerPoint3DEventGetScale](#) ([caerPoint3DEventConst](#) event)
- static void [caerPoint3DEventSetScale](#) ([caerPoint3DEvent](#) event, int8\_t scale)
- static float [caerPoint3DEventGetX](#) ([caerPoint3DEventConst](#) event)
- static void [caerPoint3DEventSetX](#) ([caerPoint3DEvent](#) event, float x)
- static float [caerPoint3DEventGetY](#) ([caerPoint3DEventConst](#) event)
- static void [caerPoint3DEventSetY](#) ([caerPoint3DEvent](#) event, float y)
- static float [caerPoint3DEventGetZ](#) ([caerPoint3DEventConst](#) event)
- static void [caerPoint3DEventSetZ](#) ([caerPoint3DEvent](#) event, float z)

### 4.17.1 Detailed Description

THIS EVENT DEFINITIONS IS STILL TO BE CONSIDERED EXPERIMENTAL AND IS SUBJECT TO FUTURE CHANGES AND REVISIONS!

Point3D Events format definition and handling functions. This contains three dimensional data points as floats, together with support for distinguishing type and scale.

### 4.17.2 Macro Definition Documentation

#### 4.17.2.1 CAER\_POINT3D\_CONST\_ITERATOR\_ALL\_START

```
#define CAER_POINT3D_CONST_ITERATOR_ALL_START(
    POINT3D_PACKET )
```

##### Value:

```
for (int32_t caerPoint3DIteratorCounter = 0; \
    caerPoint3DIteratorCounter < caerEventPacketHeaderGetEventNumber
    (&(POINT3D_PACKET)->packetHeader); \
    caerPoint3DIteratorCounter++) { \
    caerPoint3DEventConst caerPoint3DIteratorElement =
    caerPoint3DEventPacketGetEventConst (POINT3D_PACKET,
    caerPoint3DIteratorCounter);
```

Const-Iterator over all Point3D events in a packet. Returns the current index in the 'caerPoint3DIteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerPoint3DIteratorElement' variable of type caerPoint3DEventConst.

POINT3D\_PACKET: a valid Point3DEventPacket pointer. Cannot be NULL.

#### 4.17.2.2 CAER\_POINT3D\_CONST\_ITERATOR\_VALID\_START

```
#define CAER_POINT3D_CONST_ITERATOR_VALID_START(
    POINT3D_PACKET )
```

##### Value:

```
for (int32_t caerPoint3DIteratorCounter = 0; \
    caerPoint3DIteratorCounter < caerEventPacketHeaderGetEventNumber
    (&(POINT3D_PACKET)->packetHeader); \
    caerPoint3DIteratorCounter++) { \
    caerPoint3DEventConst caerPoint3DIteratorElement =
    caerPoint3DEventPacketGetEventConst (POINT3D_PACKET,
    caerPoint3DIteratorCounter); \
    if (!caerPoint3DEventIsValid(caerPoint3DIteratorElement)) { continue; }
```

Const-Iterator over only the valid Point3D events in a packet. Returns the current index in the 'caerPoint3DIteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerPoint3DIteratorElement' variable of type caerPoint3DEventConst.

POINT3D\_PACKET: a valid Point3DEventPacket pointer. Cannot be NULL.

#### 4.17.2.3 CAER\_POINT3D\_CONST\_REVERSE\_ITERATOR\_ALL\_START

```
#define CAER_POINT3D_CONST_REVERSE_ITERATOR_ALL_START(
    POINT3D_PACKET )
```

##### Value:

```
for (int32_t caerPoint3DIteratorCounter = caerEventPacketHeaderGetEventNumber
    (&(POINT3D_PACKET)->packetHeader) - 1; \
    caerPoint3DIteratorCounter >= 0; \
    caerPoint3DIteratorCounter--) { \
    caerPoint3DEventConst caerPoint3DIteratorElement =
    caerPoint3DEventPacketGetEventConst (POINT3D_PACKET,
    caerPoint3DIteratorCounter);
```

Const-Reverse iterator over all Point3D events in a packet. Returns the current index in the 'caerPoint3DIteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerPoint3DIteratorElement' variable of type caerPoint3DEventConst.

POINT3D\_PACKET: a valid Point3DEventPacket pointer. Cannot be NULL.

## 4.17.2.4 CAER\_POINT3D\_CONST\_REVERSE\_ITERATOR\_VALID\_START

```
#define CAER_POINT3D_CONST_REVERSE_ITERATOR_VALID_START(
    POINT3D_PACKET )
```

**Value:**

```
for (int32_t caerPoint3DIteratorCounter = caerEventPacketHeaderGetEventNumber
    (&(POINT3D_PACKET)->packetHeader) - 1; \
    caerPoint3DIteratorCounter >= 0; \
    caerPoint3DIteratorCounter--) { \
    caerPoint3DEventConst caerPoint3DIteratorElement =
    caerPoint3DEventPacketGetEventConst(POINT3D_PACKET,
    caerPoint3DIteratorCounter); \
    if (!caerPoint3DEventIsValid(caerPoint3DIteratorElement)) { continue; }
```

Const-Reverse iterator over only the valid Point3D events in a packet. Returns the current index in the 'caerPoint3DIteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerPoint3DIteratorElement' variable of type caerPoint3DEventConst.

POINT3D\_PACKET: a valid Point3DEventPacket pointer. Cannot be NULL.

## 4.17.2.5 CAER\_POINT3D\_ITERATOR\_ALL\_END

```
#define CAER_POINT3D_ITERATOR_ALL_END }
```

Iterator close statement.

## 4.17.2.6 CAER\_POINT3D\_ITERATOR\_ALL\_START

```
#define CAER_POINT3D_ITERATOR_ALL_START(
    POINT3D_PACKET )
```

**Value:**

```
for (int32_t caerPoint3DIteratorCounter = 0; \
    caerPoint3DIteratorCounter < caerEventPacketHeaderGetEventNumber
    (&(POINT3D_PACKET)->packetHeader); \
    caerPoint3DIteratorCounter++) { \
    caerPoint3DEvent caerPoint3DIteratorElement =
    caerPoint3DEventPacketGetEvent(POINT3D_PACKET, caerPoint3DIteratorCounter);
```

Iterator over all Point3D events in a packet. Returns the current index in the 'caerPoint3DIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerPoint3DIteratorElement' variable of type caerPoint3DEvent.

POINT3D\_PACKET: a valid Point3DEventPacket pointer. Cannot be NULL.

## 4.17.2.7 CAER\_POINT3D\_ITERATOR\_VALID\_END

```
#define CAER_POINT3D_ITERATOR_VALID_END }
```

Iterator close statement.

#### 4.17.2.8 CAER\_POINT3D\_ITERATOR\_VALID\_START

```
#define CAER_POINT3D_ITERATOR_VALID_START(
    POINT3D_PACKET )
```

**Value:**

```
for (int32_t caerPoint3DIteratorCounter = 0; \
    caerPoint3DIteratorCounter < caerEventPacketHeaderGetEventNumber
    (&(POINT3D_PACKET)->packetHeader); \
    caerPoint3DIteratorCounter++) { \
    caerPoint3DEvent caerPoint3DIteratorElement =
    caerPoint3DEventPacketGetEvent(POINT3D_PACKET, caerPoint3DIteratorCounter); \
    if (!caerPoint3DEventIsValid(caerPoint3DIteratorElement)) { continue; }
```

Iterator over only the valid Point3D events in a packet. Returns the current index in the 'caerPoint3DIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerPoint3DIteratorElement' variable of type caerPoint3DEvent.

POINT3D\_PACKET: a valid Point3DEventPacket pointer. Cannot be NULL.

#### 4.17.2.9 CAER\_POINT3D\_REVERSE\_ITERATOR\_ALL\_END

```
#define CAER_POINT3D_REVERSE_ITERATOR_ALL_END }
```

Reverse iterator close statement.

#### 4.17.2.10 CAER\_POINT3D\_REVERSE\_ITERATOR\_ALL\_START

```
#define CAER_POINT3D_REVERSE_ITERATOR_ALL_START(
    POINT3D_PACKET )
```

**Value:**

```
for (int32_t caerPoint3DIteratorCounter = caerEventPacketHeaderGetEventNumber
    (&(POINT3D_PACKET)->packetHeader) - 1; \
    caerPoint3DIteratorCounter >= 0; \
    caerPoint3DIteratorCounter--) { \
    caerPoint3DEvent caerPoint3DIteratorElement =
    caerPoint3DEventPacketGetEvent(POINT3D_PACKET, caerPoint3DIteratorCounter);
```

Reverse iterator over all Point3D events in a packet. Returns the current index in the 'caerPoint3DIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerPoint3DIteratorElement' variable of type caerPoint3DEvent.

POINT3D\_PACKET: a valid Point3DEventPacket pointer. Cannot be NULL.

#### 4.17.2.11 CAER\_POINT3D\_REVERSE\_ITERATOR\_VALID\_END

```
#define CAER_POINT3D_REVERSE_ITERATOR_VALID_END }
```

Reverse iterator close statement.

## 4.17.2.12 CAER\_POINT3D\_REVERSE\_ITERATOR\_VALID\_START

```
#define CAER_POINT3D_REVERSE_ITERATOR_VALID_START(
    POINT3D_PACKET )
```

**Value:**

```
for (int32_t caerPoint3DIteratorCounter = caerEventPacketHeaderGetEventNumber
    (&(POINT3D_PACKET)->packetHeader) - 1; \
    caerPoint3DIteratorCounter >= 0; \
    caerPoint3DIteratorCounter--) { \
    caerPoint3DEvent caerPoint3DIteratorElement =
    caerPoint3DEventPacketGetEvent(POINT3D_PACKET, caerPoint3DIteratorCounter); \
    if (!caerPoint3DEventIsValid(caerPoint3DIteratorElement)) { continue; }
```

Reverse iterator over only the valid Point3D events in a packet. Returns the current index in the 'caerPoint3DIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerPoint3DIteratorElement' variable of type caerPoint3DEvent.

POINT3D\_PACKET: a valid Point3DEventPacket pointer. Cannot be NULL.

## 4.17.2.13 POINT3D\_SCALE\_MASK

```
#define POINT3D_SCALE_MASK 0x000000FF
```

Shift and mask values for type and scale information associated with a Point3D event. Up to 128 types are supported. The scale is given as orders of magnitude, from  $10^{-128}$  to  $10^{127}$ . Bit 0 is the valid mark, see 'common.h' for more details.

## 4.17.2.14 POINT3D\_SCALE\_SHIFT

```
#define POINT3D_SCALE_SHIFT 8
```

Shift and mask values for type and scale information associated with a Point3D event. Up to 128 types are supported. The scale is given as orders of magnitude, from  $10^{-128}$  to  $10^{127}$ . Bit 0 is the valid mark, see 'common.h' for more details.

## 4.17.2.15 POINT3D\_TYPE\_MASK

```
#define POINT3D_TYPE_MASK 0x0000007F
```

Shift and mask values for type and scale information associated with a Point3D event. Up to 128 types are supported. The scale is given as orders of magnitude, from  $10^{-128}$  to  $10^{127}$ . Bit 0 is the valid mark, see 'common.h' for more details.

## 4.17.2.16 POINT3D\_TYPE\_SHIFT

```
#define POINT3D_TYPE_SHIFT 1
```

Shift and mask values for type and scale information associated with a Point3D event. Up to 128 types are supported. The scale is given as orders of magnitude, from  $10^{-128}$  to  $10^{127}$ . Bit 0 is the valid mark, see 'common.h' for more details.

### 4.17.3 Typedef Documentation

#### 4.17.3.1 caerPoint3DEvent

```
typedef struct caer_point3d_event* caerPoint3DEvent
```

Type for pointer to Point3D event data structure.

#### 4.17.3.2 caerPoint3DEventPacket

```
typedef struct caer_point3d_event_packet* caerPoint3DEventPacket
```

Type for pointer to Point3D event packet data structure.

### 4.17.4 Function Documentation

#### 4.17.4.1 caerPoint3DEventGetScale()

```
static int8_t caerPoint3DEventGetScale (
    caerPoint3DEventConst event ) [inline], [static]
```

Get the measurement scale. This allows order of magnitude shifts on the measured value to be applied automatically, such as having measurements of type Distance (meters) and storing the values as centimeters ( $10^{-2}$ ) for higher precision, but keeping that information around to allow easy changes of unit.

##### Parameters

<i>event</i>	a valid Point3DEvent pointer. Cannot be NULL.
--------------	---

##### Returns

the Point3D measurement scale.

#### 4.17.4.2 caerPoint3DEventGetTimestamp()

```
static int32_t caerPoint3DEventGetTimestamp (
    caerPoint3DEventConst event ) [inline], [static]
```

Get the 32bit event timestamp, in microseconds. Be aware that this wraps around! You can either ignore this fact, or handle the special 'TIMESTAMP\_WRAP' event that is generated when this happens, or use the 64bit timestamp which never wraps around. See '[caerEventPacketHeaderGetEventTSOverflow\(\)](#)' documentation for more details on the 64bit timestamp.

## Parameters

<i>event</i>	a valid Point3DEvent pointer. Cannot be NULL.
--------------	---

## Returns

this event's 32bit microsecond timestamp.

## 4.17.4.3 caerPoint3DEventGetTimestamp64()

```
static int64_t caerPoint3DEventGetTimestamp64 (  
    caerPoint3DEventConst event,  
    caerPoint3DEventPacketConst packet ) [inline], [static]
```

Get the 64bit event timestamp, in microseconds. See '[caerEventPacketHeaderGetEventTSOverflow\(\)](#)' documentation for more details on the 64bit timestamp.

## Parameters

<i>event</i>	a valid Point3DEvent pointer. Cannot be NULL.
<i>packet</i>	the Point3DEventPacket pointer for the packet containing this event. Cannot be NULL.

## Returns

this event's 64bit microsecond timestamp.

## 4.17.4.4 caerPoint3DEventGetType()

```
static uint8_t caerPoint3DEventGetType (  
    caerPoint3DEventConst event ) [inline], [static]
```

Get the measurement event type. This is useful to distinguish between different measurements, for example distance or weight.

## Parameters

<i>event</i>	a valid Point3DEvent pointer. Cannot be NULL.
--------------	---

## Returns

the Point3D measurement type.

#### 4.17.4.5 caerPoint3DEventGetX()

```
static float caerPoint3DEventGetX (  
    caerPoint3DEventConst event ) [inline], [static]
```

Get the X axis measurement.

##### Parameters

<i>event</i>	a valid Point3DEvent pointer. Cannot be NULL.
--------------	---

##### Returns

X axis measurement.

#### 4.17.4.6 caerPoint3DEventGetY()

```
static float caerPoint3DEventGetY (  
    caerPoint3DEventConst event ) [inline], [static]
```

Get the Y axis measurement.

##### Parameters

<i>event</i>	a valid Point3DEvent pointer. Cannot be NULL.
--------------	---

##### Returns

Y axis measurement.

#### 4.17.4.7 caerPoint3DEventGetZ()

```
static float caerPoint3DEventGetZ (  
    caerPoint3DEventConst event ) [inline], [static]
```

Get the Z axis measurement.

##### Parameters

<i>event</i>	a valid Point3DEvent pointer. Cannot be NULL.
--------------	---

##### Returns

Z axis measurement.



#### 4.17.4.8 caerPoint3DEventInvalidate()

```
static void caerPoint3DEventInvalidate (
    caerPoint3DEvent event,
    caerPoint3DEventPacket packet ) [inline], [static]
```

Invalidate the current event by setting its valid bit to false and decreasing the number of valid events held in the packet. Only works with events that are already valid!

##### Parameters

<i>event</i>	a valid Point3DEvent pointer. Cannot be NULL.
<i>packet</i>	the Point3DEventPacket pointer for the packet containing this event. Cannot be NULL.

#### 4.17.4.9 caerPoint3DEventIsValid()

```
static bool caerPoint3DEventIsValid (
    caerPoint3DEventConst event ) [inline], [static]
```

Check if this Point3D event is valid.

##### Parameters

<i>event</i>	a valid Point3DEvent pointer. Cannot be NULL.
--------------	---

##### Returns

true if valid, false if not.

#### 4.17.4.10 caerPoint3DEventPacketAllocate()

```
caerPoint3DEventPacket caerPoint3DEventPacketAllocate (
    int32_t eventCapacity,
    int16_t eventSource,
    int32_t tsOverflow )
```

Allocate a new Point3D events packet. Use free() to reclaim this memory.

##### Parameters

<i>eventCapacity</i>	the maximum number of events this packet will hold.
<i>eventSource</i>	the unique ID representing the source/generator of this packet.
<i>tsOverflow</i>	the current timestamp overflow counter value for this packet.

**Returns**

a valid Point3DEventPacket handle or NULL on error.

**4.17.4.11 caerPoint3DEventPacketGetEvent()**

```
static caerPoint3DEvent caerPoint3DEventPacketGetEvent (
    caerPoint3DEventPacket packet,
    int32_t n ) [inline], [static]
```

Get the Point3D event at the given index from the event packet.

**Parameters**

<i>packet</i>	a valid Point3DEventPacket pointer. Cannot be NULL.
<i>n</i>	the index of the returned event. Must be within [0,eventCapacity[ bounds.

**Returns**

the requested Point3D event. NULL on error.

**4.17.4.12 caerPoint3DEventPacketGetEventConst()**

```
static caerPoint3DEventConst caerPoint3DEventPacketGetEventConst (
    caerPoint3DEventPacketConst packet,
    int32_t n ) [inline], [static]
```

Get the Point3D event at the given index from the event packet. This is a read-only event, do not change its contents in any way!

**Parameters**

<i>packet</i>	a valid Point3DEventPacket pointer. Cannot be NULL.
<i>n</i>	the index of the returned event. Must be within [0,eventCapacity[ bounds.

**Returns**

the requested read-only Point3D event. NULL on error.

**4.17.4.13 caerPoint3DEventSetScale()**

```
static void caerPoint3DEventSetScale (
    caerPoint3DEvent event,
    int8_t scale ) [inline], [static]
```

Set the measurement scale. This allows order of magnitude shifts on the measured value to be applied automatically, such as having measurements of type Distance (meters) and storing the values as centimeters ( $10^{-2}$ ) for higher precision, but keeping that information around to allow easy changes of unit.

**Parameters**

<i>event</i>	a valid Point3DEvent pointer. Cannot be NULL.
<i>scale</i>	the Point3D measurement scale.

**4.17.4.14 caerPoint3DEventSetTimestamp()**

```
static void caerPoint3DEventSetTimestamp (
    caerPoint3DEvent event,
    int32_t timestamp ) [inline], [static]
```

Set the 32bit event timestamp, the value has to be in microseconds.

**Parameters**

<i>event</i>	a valid Point3DEvent pointer. Cannot be NULL.
<i>timestamp</i>	a positive 32bit microsecond timestamp.

**4.17.4.15 caerPoint3DEventSetType()**

```
static void caerPoint3DEventSetType (
    caerPoint3DEvent event,
    uint8_t type ) [inline], [static]
```

Set the measurement event type. This is useful to distinguish between different measurements, for example distance or weight.

**Parameters**

<i>event</i>	a valid Point3DEvent pointer. Cannot be NULL.
<i>type</i>	the Point3D measurement type.

**4.17.4.16 caerPoint3DEventSetX()**

```
static void caerPoint3DEventSetX (
    caerPoint3DEvent event,
    float x ) [inline], [static]
```

Set the X axis measurement.

## Parameters

<i>event</i>	a valid Point3DEvent pointer. Cannot be NULL.
<i>x</i>	X axis measurement.

## 4.17.4.17 caerPoint3DEventSetY()

```
static void caerPoint3DEventSetY (  
    caerPoint3DEvent event,  
    float y ) [inline], [static]
```

Set the Y axis measurement.

## Parameters

<i>event</i>	a valid Point3DEvent pointer. Cannot be NULL.
<i>y</i>	Y axis measurement.

## 4.17.4.18 caerPoint3DEventSetZ()

```
static void caerPoint3DEventSetZ (  
    caerPoint3DEvent event,  
    float z ) [inline], [static]
```

Set the Z axis measurement.

## Parameters

<i>event</i>	a valid Point3DEvent pointer. Cannot be NULL.
<i>z</i>	Z axis measurement.

## 4.17.4.19 caerPoint3DEventValidate()

```
static void caerPoint3DEventValidate (  
    caerPoint3DEvent event,  
    caerPoint3DEventPacket packet ) [inline], [static]
```

Validate the current event by setting its valid bit to true and increasing the event packet's event count and valid event count. Only works on events that are invalid. DO NOT CALL THIS AFTER HAVING PREVIOUSLY ALREADY INVALIDATED THIS EVENT, the total count will be incorrect.

## Parameters

<i>event</i>	a valid Point3DEvent pointer. Cannot be NULL.
<i>packet</i>	the Point3DEventPacket pointer for the packet containing this event. Cannot be NULL.

## 4.17.4.20 PACKED\_STRUCT() [1/2]

```
PACKED_STRUCT (
    struct caer_point3d_event { uint32_t info;float x;float y;float z;int32_t timestamp;}
)
```

Point3D event data structure definition. This contains information about the measurement, such as a type and a scale field, together with the usual validity mark. The three measurements (x, y, z) are stored as floats. Floats are in IEEE 754-2008 binary32 format. Signed integers are used for fields that are to be interpreted directly, for compatibility with languages that do not have unsigned integer types, such as Java.

## 4.17.4.21 PACKED\_STRUCT() [2/2]

```
PACKED_STRUCT (
    struct caer_point3d_event_packet { struct caer_event_packet_header packetHeader;struct
    caer_point3d_event events[];} )
```

Point3D event packet data structure definition. EventPackets are always made up of the common packet header, followed by 'eventCapacity' events. Everything has to be in one contiguous memory block.

## 4.18 events/point4d.h File Reference

```
#include "common.h"
```

## Macros

- `#define CAER_POINT4D_ITERATOR_ALL_START(POINT4D_PACKET)`
  - `#define CAER_POINT4D_CONST_ITERATOR_ALL_START(POINT4D_PACKET)`
  - `#define CAER_POINT4D_ITERATOR_ALL_END }`
  - `#define CAER_POINT4D_ITERATOR_VALID_START(POINT4D_PACKET)`
  - `#define CAER_POINT4D_CONST_ITERATOR_VALID_START(POINT4D_PACKET)`
  - `#define CAER_POINT4D_ITERATOR_VALID_END }`
  - `#define CAER_POINT4D_REVERSE_ITERATOR_ALL_START(POINT4D_PACKET)`
  - `#define CAER_POINT4D_CONST_REVERSE_ITERATOR_ALL_START(POINT4D_PACKET)`
  - `#define CAER_POINT4D_REVERSE_ITERATOR_ALL_END }`
  - `#define CAER_POINT4D_REVERSE_ITERATOR_VALID_START(POINT4D_PACKET)`
  - `#define CAER_POINT4D_CONST_REVERSE_ITERATOR_VALID_START(POINT4D_PACKET)`
  - `#define CAER_POINT4D_REVERSE_ITERATOR_VALID_END }`
- 
- `#define POINT4D_TYPE_SHIFT 1`
  - `#define POINT4D_TYPE_MASK 0x0000007F`
  - `#define POINT4D_SCALE_SHIFT 8`
  - `#define POINT4D_SCALE_MASK 0x000000FF`

## Typedefs

- typedef struct caer\_point4d\_event \* [caerPoint4DEvent](#)
- typedef const struct caer\_point4d\_event \* **caerPoint4DEventConst**
- typedef struct caer\_point4d\_event\_packet \* [caerPoint4DEventPacket](#)
- typedef const struct caer\_point4d\_event\_packet \* **caerPoint4DEventPacketConst**

## Functions

- [PACKED\\_STRUCT](#) (struct caer\_point4d\_event { uint32\_t info;float x;float y;float z;float w;int32\_t timestamp;})
- [PACKED\\_STRUCT](#) (struct caer\_point4d\_event\_packet { struct caer\_event\_packet\_header packet; Header;struct caer\_point4d\_event events[;];})
- [caerPoint4DEventPacket](#) [caerPoint4DEventPacketAllocate](#) (int32\_t eventCapacity, int16\_t eventSource, int32\_t tsOverflow)
- static [caerPoint4DEvent](#) [caerPoint4DEventPacketGetEvent](#) ([caerPoint4DEventPacket](#) packet, int32\_t n)
- static [caerPoint4DEventConst](#) [caerPoint4DEventPacketGetEventConst](#) ([caerPoint4DEventPacketConst](#) packet, int32\_t n)
- static int32\_t [caerPoint4DEventGetTimestamp](#) ([caerPoint4DEventConst](#) event)
- static int64\_t [caerPoint4DEventGetTimestamp64](#) ([caerPoint4DEventConst](#) event, [caerPoint4DEventPacket](#)↵ Const packet)
- static void [caerPoint4DEventSetTimestamp](#) ([caerPoint4DEvent](#) event, int32\_t timestamp)
- static bool [caerPoint4DEventIsValid](#) ([caerPoint4DEventConst](#) event)
- static void [caerPoint4DEventValidate](#) ([caerPoint4DEvent](#) event, [caerPoint4DEventPacket](#) packet)
- static void [caerPoint4DEventInvalidate](#) ([caerPoint4DEvent](#) event, [caerPoint4DEventPacket](#) packet)
- static uint8\_t [caerPoint4DEventGetType](#) ([caerPoint4DEventConst](#) event)
- static void [caerPoint4DEventSetType](#) ([caerPoint4DEvent](#) event, uint8\_t type)
- static int8\_t [caerPoint4DEventGetScale](#) ([caerPoint4DEventConst](#) event)
- static void [caerPoint4DEventSetScale](#) ([caerPoint4DEvent](#) event, int8\_t scale)
- static float [caerPoint4DEventGetX](#) ([caerPoint4DEventConst](#) event)
- static void [caerPoint4DEventSetX](#) ([caerPoint4DEvent](#) event, float x)
- static float [caerPoint4DEventGetY](#) ([caerPoint4DEventConst](#) event)
- static void [caerPoint4DEventSetY](#) ([caerPoint4DEvent](#) event, float y)
- static float [caerPoint4DEventGetZ](#) ([caerPoint4DEventConst](#) event)
- static void [caerPoint4DEventSetZ](#) ([caerPoint4DEvent](#) event, float z)
- static float [caerPoint4DEventGetW](#) ([caerPoint4DEventConst](#) event)
- static void [caerPoint4DEventSetW](#) ([caerPoint4DEvent](#) event, float w)

### 4.18.1 Detailed Description

THIS EVENT DEFINITION IS STILL TO BE CONSIDERED EXPERIMENTAL AND IS SUBJECT TO FUTURE C↵ANGES AND REVISIONS!

Point4D Events format definition and handling functions. This contains four dimensional data points as floats, together with support for distinguishing type and scale. Useful for homogeneous coordinates for example.

### 4.18.2 Macro Definition Documentation

## 4.18.2.1 CAER\_POINT4D\_CONST\_ITERATOR\_ALL\_START

```
#define CAER_POINT4D_CONST_ITERATOR_ALL_START(
    POINT4D_PACKET )
```

**Value:**

```
for (int32_t caerPoint4DIteratorCounter = 0; \
    caerPoint4DIteratorCounter < caerEventPacketHeaderGetEventNumber
    (&(POINT4D_PACKET)->packetHeader); \
    caerPoint4DIteratorCounter++) { \
    caerPoint4DEventConst caerPoint4DIteratorElement =
    caerPoint4DEventPacketGetEventConst (POINT4D_PACKET,
    caerPoint4DIteratorCounter);
```

Const-Iterator over all Point4D events in a packet. Returns the current index in the 'caerPoint4DIteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerPoint4DIteratorElement' variable of type caerPoint4DEventConst.

POINT4D\_PACKET: a valid Point4DEventPacket pointer. Cannot be NULL.

## 4.18.2.2 CAER\_POINT4D\_CONST\_ITERATOR\_VALID\_START

```
#define CAER_POINT4D_CONST_ITERATOR_VALID_START(
    POINT4D_PACKET )
```

**Value:**

```
for (int32_t caerPoint4DIteratorCounter = 0; \
    caerPoint4DIteratorCounter < caerEventPacketHeaderGetEventNumber
    (&(POINT4D_PACKET)->packetHeader); \
    caerPoint4DIteratorCounter++) { \
    caerPoint4DEventConst caerPoint4DIteratorElement =
    caerPoint4DEventPacketGetEventConst (POINT4D_PACKET,
    caerPoint4DIteratorCounter); \
    if (!caerPoint4DEventIsValid(caerPoint4DIteratorElement)) { continue; }
```

Const-Iterator over only the valid Point4D events in a packet. Returns the current index in the 'caerPoint4DIteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerPoint4DIteratorElement' variable of type caerPoint4DEventConst.

POINT4D\_PACKET: a valid Point4DEventPacket pointer. Cannot be NULL.

## 4.18.2.3 CAER\_POINT4D\_CONST\_REVERSE\_ITERATOR\_ALL\_START

```
#define CAER_POINT4D_CONST_REVERSE_ITERATOR_ALL_START(
    POINT4D_PACKET )
```

**Value:**

```
for (int32_t caerPoint4DIteratorCounter = caerEventPacketHeaderGetEventNumber
    (&(POINT4D_PACKET)->packetHeader) - 1; \
    caerPoint4DIteratorCounter >= 0; \
    caerPoint4DIteratorCounter--) { \
    caerPoint4DEventConst caerPoint4DIteratorElement =
    caerPoint4DEventPacketGetEventConst (POINT4D_PACKET,
    caerPoint4DIteratorCounter);
```

Const-Reverse iterator over all Point4D events in a packet. Returns the current index in the 'caerPoint4DIteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerPoint4DIteratorElement' variable of type caerPoint4DEventConst.

POINT4D\_PACKET: a valid Point4DEventPacket pointer. Cannot be NULL.

#### 4.18.2.4 CAER\_POINT4D\_CONST\_REVERSE\_ITERATOR\_VALID\_START

```
#define CAER_POINT4D_CONST_REVERSE_ITERATOR_VALID_START(
    POINT4D_PACKET )
```

**Value:**

```
for (int32_t caerPoint4DIteratorCounter = caerEventPacketHeaderGetEventNumber
    (&(POINT4D_PACKET)->packetHeader) - 1; \
    caerPoint4DIteratorCounter >= 0; \
    caerPoint4DIteratorCounter--) { \
    caerPoint4DEventConst caerPoint4DIteratorElement =
    caerPoint4DEventPacketGetEventConst(POINT4D_PACKET,
    caerPoint4DIteratorCounter); \
    if (!caerPoint4DEventIsValid(caerPoint4DIteratorElement)) { continue; }
```

Const-Reverse iterator over only the valid Point4D events in a packet. Returns the current index in the 'caerPoint4DIteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerPoint4DIteratorElement' variable of type caerPoint4DEventConst.

POINT4D\_PACKET: a valid Point4DEventPacket pointer. Cannot be NULL.

#### 4.18.2.5 CAER\_POINT4D\_ITERATOR\_ALL\_END

```
#define CAER_POINT4D_ITERATOR_ALL_END }
```

Iterator close statement.

#### 4.18.2.6 CAER\_POINT4D\_ITERATOR\_ALL\_START

```
#define CAER_POINT4D_ITERATOR_ALL_START(
    POINT4D_PACKET )
```

**Value:**

```
for (int32_t caerPoint4DIteratorCounter = 0; \
    caerPoint4DIteratorCounter < caerEventPacketHeaderGetEventNumber
    (&(POINT4D_PACKET)->packetHeader); \
    caerPoint4DIteratorCounter++) { \
    caerPoint4DEvent caerPoint4DIteratorElement =
    caerPoint4DEventPacketGetEvent(POINT4D_PACKET, caerPoint4DIteratorCounter);
```

Iterator over all Point4D events in a packet. Returns the current index in the 'caerPoint4DIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerPoint4DIteratorElement' variable of type caerPoint4DEvent.

POINT4D\_PACKET: a valid Point4DEventPacket pointer. Cannot be NULL.

#### 4.18.2.7 CAER\_POINT4D\_ITERATOR\_VALID\_END

```
#define CAER_POINT4D_ITERATOR_VALID_END }
```

Iterator close statement.



## 4.18.2.8 CAER\_POINT4D\_ITERATOR\_VALID\_START

```
#define CAER_POINT4D_ITERATOR_VALID_START(
    POINT4D_PACKET )
```

**Value:**

```
for (int32_t caerPoint4DIteratorCounter = 0; \
    caerPoint4DIteratorCounter < caerEventPacketHeaderGetEventNumber
    (&(POINT4D_PACKET)->packetHeader); \
    caerPoint4DIteratorCounter++) { \
    caerPoint4DEvent caerPoint4DIteratorElement =
    caerPoint4DEventPacketGetEvent(POINT4D_PACKET, caerPoint4DIteratorCounter); \
    if (!caerPoint4DEventIsValid(caerPoint4DIteratorElement)) { continue; }
```

Iterator over only the valid Point4D events in a packet. Returns the current index in the 'caerPoint4DIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerPoint4DIteratorElement' variable of type caerPoint4DEvent.

POINT4D\_PACKET: a valid Point4DEventPacket pointer. Cannot be NULL.

## 4.18.2.9 CAER\_POINT4D\_REVERSE\_ITERATOR\_ALL\_END

```
#define CAER_POINT4D_REVERSE_ITERATOR_ALL_END }
```

Reverse iterator close statement.

## 4.18.2.10 CAER\_POINT4D\_REVERSE\_ITERATOR\_ALL\_START

```
#define CAER_POINT4D_REVERSE_ITERATOR_ALL_START(
    POINT4D_PACKET )
```

**Value:**

```
for (int32_t caerPoint4DIteratorCounter = caerEventPacketHeaderGetEventNumber
    (&(POINT4D_PACKET)->packetHeader) - 1; \
    caerPoint4DIteratorCounter >= 0; \
    caerPoint4DIteratorCounter--) { \
    caerPoint4DEvent caerPoint4DIteratorElement =
    caerPoint4DEventPacketGetEvent(POINT4D_PACKET, caerPoint4DIteratorCounter);
```

Reverse iterator over all Point4D events in a packet. Returns the current index in the 'caerPoint4DIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerPoint4DIteratorElement' variable of type caerPoint4DEvent.

POINT4D\_PACKET: a valid Point4DEventPacket pointer. Cannot be NULL.

## 4.18.2.11 CAER\_POINT4D\_REVERSE\_ITERATOR\_VALID\_END

```
#define CAER_POINT4D_REVERSE_ITERATOR_VALID_END }
```

Reverse iterator close statement.

#### 4.18.2.12 CAER\_POINT4D\_REVERSE\_ITERATOR\_VALID\_START

```
#define CAER_POINT4D_REVERSE_ITERATOR_VALID_START(
    POINT4D_PACKET )
```

**Value:**

```
for (int32_t caerPoint4DIteratorCounter = caerEventPacketHeaderGetEventNumber
    (&(POINT4D_PACKET)->packetHeader) - 1; \
    caerPoint4DIteratorCounter >= 0; \
    caerPoint4DIteratorCounter--) { \
    caerPoint4DEvent caerPoint4DIteratorElement =
    caerPoint4DEventPacketGetEvent(POINT4D_PACKET, caerPoint4DIteratorCounter); \
    if (!caerPoint4DEventIsValid(caerPoint4DIteratorElement)) { continue; }
```

Reverse iterator over only the valid Point4D events in a packet. Returns the current index in the 'caerPoint4DIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerPoint4DIteratorElement' variable of type caerPoint4DEvent.

POINT4D\_PACKET: a valid Point4DEventPacket pointer. Cannot be NULL.

#### 4.18.2.13 POINT4D\_SCALE\_MASK

```
#define POINT4D_SCALE_MASK 0x000000FF
```

Shift and mask values for type and scale information associated with a Point4D event. Up to 128 types are supported. The scale is given as orders of magnitude, from  $10^{-128}$  to  $10^{127}$ . Bit 0 is the valid mark, see '[common.h](#)' for more details.

#### 4.18.2.14 POINT4D\_SCALE\_SHIFT

```
#define POINT4D_SCALE_SHIFT 8
```

Shift and mask values for type and scale information associated with a Point4D event. Up to 128 types are supported. The scale is given as orders of magnitude, from  $10^{-128}$  to  $10^{127}$ . Bit 0 is the valid mark, see '[common.h](#)' for more details.

#### 4.18.2.15 POINT4D\_TYPE\_MASK

```
#define POINT4D_TYPE_MASK 0x0000007F
```

Shift and mask values for type and scale information associated with a Point4D event. Up to 128 types are supported. The scale is given as orders of magnitude, from  $10^{-128}$  to  $10^{127}$ . Bit 0 is the valid mark, see '[common.h](#)' for more details.

#### 4.18.2.16 POINT4D\_TYPE\_SHIFT

```
#define POINT4D_TYPE_SHIFT 1
```

Shift and mask values for type and scale information associated with a Point4D event. Up to 128 types are supported. The scale is given as orders of magnitude, from  $10^{-128}$  to  $10^{127}$ . Bit 0 is the valid mark, see '[common.h](#)' for more details.

### 4.18.3 Typedef Documentation

#### 4.18.3.1 caerPoint4DEvent

```
typedef struct caer_point4d_event* caerPoint4DEvent
```

Type for pointer to Point4D event data structure.

#### 4.18.3.2 caerPoint4DEventPacket

```
typedef struct caer_point4d_event_packet* caerPoint4DEventPacket
```

Type for pointer to Point4D event packet data structure.

### 4.18.4 Function Documentation

#### 4.18.4.1 caerPoint4DEventGetScale()

```
static int8_t caerPoint4DEventGetScale (
    caerPoint4DEventConst event ) [inline], [static]
```

Get the measurement scale. This allows order of magnitude shifts on the measured value to be applied automatically, such as having measurements of type Distance (meters) and storing the values as centimeters ( $10^{-2}$ ) for higher precision, but keeping that information around to allow easy changes of unit.

##### Parameters

<i>event</i>	a valid Point4DEvent pointer. Cannot be NULL.
--------------	---

##### Returns

the Point4D measurement scale.

#### 4.18.4.2 caerPoint4DEventGetTimestamp()

```
static int32_t caerPoint4DEventGetTimestamp (
    caerPoint4DEventConst event ) [inline], [static]
```

Get the 32bit event timestamp, in microseconds. Be aware that this wraps around! You can either ignore this fact, or handle the special 'TIMESTAMP\_WRAP' event that is generated when this happens, or use the 64bit timestamp which never wraps around. See '[caerEventPacketHeaderGetEventTSOverflow\(\)](#)' documentation for more details on the 64bit timestamp.

**Parameters**

<i>event</i>	a valid Point4DEvent pointer. Cannot be NULL.
--------------	---

**Returns**

this event's 32bit microsecond timestamp.

**4.18.4.3 caerPoint4DEventGetTimestamp64()**

```
static int64_t caerPoint4DEventGetTimestamp64 (  
    caerPoint4DEventConst event,  
    caerPoint4DEventPacketConst packet ) [inline], [static]
```

Get the 64bit event timestamp, in microseconds. See '[caerEventPacketHeaderGetEventTSOverflow\(\)](#)' documentation for more details on the 64bit timestamp.

**Parameters**

<i>event</i>	a valid Point4DEvent pointer. Cannot be NULL.
<i>packet</i>	the Point4DEventPacket pointer for the packet containing this event. Cannot be NULL.

**Returns**

this event's 64bit microsecond timestamp.

**4.18.4.4 caerPoint4DEventGetType()**

```
static uint8_t caerPoint4DEventGetType (  
    caerPoint4DEventConst event ) [inline], [static]
```

Get the measurement event type. This is useful to distinguish between different measurements, for example distance or weight.

**Parameters**

<i>event</i>	a valid Point4DEvent pointer. Cannot be NULL.
--------------	---

**Returns**

the Point4D measurement type.

#### 4.18.4.5 caerPoint4DEventGetW()

```
static float caerPoint4DEventGetW (  
    caerPoint4DEventConst event ) [inline], [static]
```

Get the W axis measurement.

##### Parameters

<i>event</i>	a valid Point4DEvent pointer. Cannot be NULL.
--------------	---

##### Returns

W axis measurement.

#### 4.18.4.6 caerPoint4DEventGetX()

```
static float caerPoint4DEventGetX (  
    caerPoint4DEventConst event ) [inline], [static]
```

Get the X axis measurement.

##### Parameters

<i>event</i>	a valid Point4DEvent pointer. Cannot be NULL.
--------------	---

##### Returns

X axis measurement.

#### 4.18.4.7 caerPoint4DEventGetY()

```
static float caerPoint4DEventGetY (  
    caerPoint4DEventConst event ) [inline], [static]
```

Get the Y axis measurement.

##### Parameters

<i>event</i>	a valid Point4DEvent pointer. Cannot be NULL.
--------------	---

##### Returns

Y axis measurement.

#### 4.18.4.8 caerPoint4DEventGetZ()

```
static float caerPoint4DEventGetZ (
    caerPoint4DEventConst event ) [inline], [static]
```

Get the Z axis measurement.

##### Parameters

<i>event</i>	a valid Point4DEvent pointer. Cannot be NULL.
--------------	---

##### Returns

Z axis measurement.

#### 4.18.4.9 caerPoint4DEventInvalidate()

```
static void caerPoint4DEventInvalidate (
    caerPoint4DEvent event,
    caerPoint4DEventPacket packet ) [inline], [static]
```

Invalidate the current event by setting its valid bit to false and decreasing the number of valid events held in the packet. Only works with events that are already valid!

##### Parameters

<i>event</i>	a valid Point4DEvent pointer. Cannot be NULL.
<i>packet</i>	the Point4DEventPacket pointer for the packet containing this event. Cannot be NULL.

#### 4.18.4.10 caerPoint4DEventIsValid()

```
static bool caerPoint4DEventIsValid (
    caerPoint4DEventConst event ) [inline], [static]
```

Check if this Point4D event is valid.

##### Parameters

<i>event</i>	a valid Point4DEvent pointer. Cannot be NULL.
--------------	---

**Returns**

true if valid, false if not.

**4.18.4.11 caerPoint4DEventPacketAllocate()**

```
caerPoint4DEventPacket caerPoint4DEventPacketAllocate (
    int32_t eventCapacity,
    int16_t eventSource,
    int32_t tsOverflow )
```

Allocate a new Point4D events packet. Use free() to reclaim this memory.

**Parameters**

<i>eventCapacity</i>	the maximum number of events this packet will hold.
<i>eventSource</i>	the unique ID representing the source/generator of this packet.
<i>tsOverflow</i>	the current timestamp overflow counter value for this packet.

**Returns**

a valid Point4DEventPacket handle or NULL on error.

**4.18.4.12 caerPoint4DEventPacketGetEvent()**

```
static caerPoint4DEvent caerPoint4DEventPacketGetEvent (
    caerPoint4DEventPacket packet,
    int32_t n ) [inline], [static]
```

Get the Point4D event at the given index from the event packet.

**Parameters**

<i>packet</i>	a valid Point4DEventPacket pointer. Cannot be NULL.
<i>n</i>	the index of the returned event. Must be within [0,eventCapacity[ bounds.

**Returns**

the requested Point4D event. NULL on error.

**4.18.4.13 caerPoint4DEventPacketGetEventConst()**

```
static caerPoint4DEventConst caerPoint4DEventPacketGetEventConst (
    caerPoint4DEventPacketConst packet,
    int32_t n ) [inline], [static]
```

Get the Point4D event at the given index from the event packet. This is a read-only event, do not change its contents in any way!

#### Parameters

<i>packet</i>	a valid Point4DEventPacket pointer. Cannot be NULL.
<i>n</i>	the index of the returned event. Must be within [0,eventCapacity[ bounds.

#### Returns

the requested read-only Point4D event. NULL on error.

#### 4.18.4.14 caerPoint4DEventSetScale()

```
static void caerPoint4DEventSetScale (
    caerPoint4DEvent event,
    int8_t scale ) [inline], [static]
```

Set the measurement scale. This allows order of magnitude shifts on the measured value to be applied automatically, such as having measurements of type Distance (meters) and storing the values as centimeters ( $10^{-2}$ ) for higher precision, but keeping that information around to allow easy changes of unit.

#### Parameters

<i>event</i>	a valid Point4DEvent pointer. Cannot be NULL.
<i>scale</i>	the Point4D measurement scale.

#### 4.18.4.15 caerPoint4DEventSetTimestamp()

```
static void caerPoint4DEventSetTimestamp (
    caerPoint4DEvent event,
    int32_t timestamp ) [inline], [static]
```

Set the 32bit event timestamp, the value has to be in microseconds.

#### Parameters

<i>event</i>	a valid Point4DEvent pointer. Cannot be NULL.
<i>timestamp</i>	a positive 32bit microsecond timestamp.

#### 4.18.4.16 caerPoint4DEventSetType()

```
static void caerPoint4DEventSetType (
```



```
caerPoint4DEvent event,  
uint8_t type ) [inline], [static]
```

Set the measurement event type. This is useful to distinguish between different measurements, for example distance or weight.

#### Parameters

<i>event</i>	a valid Point4DEvent pointer. Cannot be NULL.
<i>type</i>	the Point4D measurement type.

#### 4.18.4.17 caerPoint4DEventSetW()

```
static void caerPoint4DEventSetW (  
    caerPoint4DEvent event,  
    float w ) [inline], [static]
```

Set the W axis measurement.

#### Parameters

<i>event</i>	a valid Point4DEvent pointer. Cannot be NULL.
<i>w</i>	W axis measurement.

#### 4.18.4.18 caerPoint4DEventSetX()

```
static void caerPoint4DEventSetX (  
    caerPoint4DEvent event,  
    float x ) [inline], [static]
```

Set the X axis measurement.

#### Parameters

<i>event</i>	a valid Point4DEvent pointer. Cannot be NULL.
<i>x</i>	X axis measurement.

#### 4.18.4.19 caerPoint4DEventSetY()

```
static void caerPoint4DEventSetY (  
    caerPoint4DEvent event,  
    float y ) [inline], [static]
```

Set the Y axis measurement.

## Parameters

<i>event</i>	a valid Point4DEvent pointer. Cannot be NULL.
<i>y</i>	Y axis measurement.

## 4.18.4.20 caerPoint4DEventSetZ()

```
static void caerPoint4DEventSetZ (
    caerPoint4DEvent event,
    float z ) [inline], [static]
```

Set the Z axis measurement.

## Parameters

<i>event</i>	a valid Point4DEvent pointer. Cannot be NULL.
<i>z</i>	Z axis measurement.

## 4.18.4.21 caerPoint4DEventValidate()

```
static void caerPoint4DEventValidate (
    caerPoint4DEvent event,
    caerPoint4DEventPacket packet ) [inline], [static]
```

Validate the current event by setting its valid bit to true and increasing the event packet's event count and valid event count. Only works on events that are invalid. DO NOT CALL THIS AFTER HAVING PREVIOUSLY ALREADY INVALIDATED THIS EVENT, the total count will be incorrect.

## Parameters

<i>event</i>	a valid Point4DEvent pointer. Cannot be NULL.
<i>packet</i>	the Point4DEventPacket pointer for the packet containing this event. Cannot be NULL.

## 4.18.4.22 PACKED\_STRUCT() [1/2]

```
PACKED_STRUCT (
    struct caer_point4d_event { uint32_t info;float x;float y;float z;float w;int32_t
timestamp;} )
```

Point4D event data structure definition. This contains information about the measurement, such as a type and a scale field, together with the usual validity mark. The four measurements (x, y, z, w) are stored as floats. Floats are in IEEE 754-2008 binary32 format. Signed integers are used for fields that are to be interpreted directly, for compatibility with languages that do not have unsigned integer types, such as Java.

## 4.18.4.23 PACKED\_STRUCT() [2/2]

```
PACKED_STRUCT (
    struct caer_point4d_event_packet { struct caer_event_packet_header packetHeader; struct
    caer_point4d_event events[]; } )
```

Point4D event packet data structure definition. EventPackets are always made up of the common packet header, followed by 'eventCapacity' events. Everything has to be in one contiguous memory block.

## 4.19 events/polarity.h File Reference

```
#include "common.h"
```

## Macros

- #define [CAER\\_POLARITY\\_ITERATOR\\_ALL\\_START](#)(POLARITY\_PACKET)
- #define [CAER\\_POLARITY\\_CONST\\_ITERATOR\\_ALL\\_START](#)(POLARITY\_PACKET)
- #define [CAER\\_POLARITY\\_ITERATOR\\_ALL\\_END](#) }
- #define [CAER\\_POLARITY\\_ITERATOR\\_VALID\\_START](#)(POLARITY\_PACKET)
- #define [CAER\\_POLARITY\\_CONST\\_ITERATOR\\_VALID\\_START](#)(POLARITY\_PACKET)
- #define [CAER\\_POLARITY\\_ITERATOR\\_VALID\\_END](#) }
- #define [CAER\\_POLARITY\\_REVERSE\\_ITERATOR\\_ALL\\_START](#)(POLARITY\_PACKET)
- #define [CAER\\_POLARITY\\_CONST\\_REVERSE\\_ITERATOR\\_ALL\\_START](#)(POLARITY\_PACKET)
- #define [CAER\\_POLARITY\\_REVERSE\\_ITERATOR\\_ALL\\_END](#) }
- #define [CAER\\_POLARITY\\_REVERSE\\_ITERATOR\\_VALID\\_START](#)(POLARITY\_PACKET)
- #define [CAER\\_POLARITY\\_CONST\\_REVERSE\\_ITERATOR\\_VALID\\_START](#)(POLARITY\_PACKET)
- #define [CAER\\_POLARITY\\_REVERSE\\_ITERATOR\\_VALID\\_END](#) }

- #define [POLARITY\\_SHIFT](#) 1
- #define [POLARITY\\_MASK](#) 0x00000001
- #define [POLARITY\\_Y\\_ADDR\\_SHIFT](#) 2
- #define [POLARITY\\_Y\\_ADDR\\_MASK](#) 0x00007FFF
- #define [POLARITY\\_X\\_ADDR\\_SHIFT](#) 17
- #define [POLARITY\\_X\\_ADDR\\_MASK](#) 0x00007FFF

## Typedefs

- typedef struct caer\_polarity\_event \* [caerPolarityEvent](#)
- typedef const struct caer\_polarity\_event \* [caerPolarityEventConst](#)
- typedef struct caer\_polarity\_event\_packet \* [caerPolarityEventPacket](#)
- typedef const struct caer\_polarity\_event\_packet \* [caerPolarityEventPacketConst](#)

## Functions

- [PACKED\\_STRUCT](#) (struct caer\_polarity\_event { uint32\_t data;int32\_t timestamp;})
- [PACKED\\_STRUCT](#) (struct caer\_polarity\_event\_packet { struct caer\_event\_packet\_header packet↵ Header;struct caer\_polarity\_event events[;];})
- [caerPolarityEventPacket](#) [caerPolarityEventPacketAllocate](#) (int32\_t eventCapacity, int16\_t eventSource, int32\_t tsOverflow)
- static [caerPolarityEvent](#) [caerPolarityEventPacketGetEvent](#) ([caerPolarityEventPacket](#) packet, int32\_t n)
- static caerPolarityEventConst [caerPolarityEventPacketGetEventConst](#) ([caerPolarityEventPacketConst](#) packet, int32\_t n)
- static int32\_t [caerPolarityEventGetTimestamp](#) ([caerPolarityEventConst](#) event)
- static int64\_t [caerPolarityEventGetTimestamp64](#) ([caerPolarityEventConst](#) event, [caerPolarityEventPacketConst](#)↵ packet)
- static void [caerPolarityEventSetTimestamp](#) ([caerPolarityEvent](#) event, int32\_t timestamp)
- static bool [caerPolarityEventsIsValid](#) ([caerPolarityEventConst](#) event)
- static void [caerPolarityEventValidate](#) ([caerPolarityEvent](#) event, [caerPolarityEventPacket](#) packet)
- static void [caerPolarityEventInvalidate](#) ([caerPolarityEvent](#) event, [caerPolarityEventPacket](#) packet)
- static bool [caerPolarityEventGetPolarity](#) ([caerPolarityEventConst](#) event)
- static void [caerPolarityEventSetPolarity](#) ([caerPolarityEvent](#) event, bool polarity)
- static uint16\_t [caerPolarityEventGetY](#) ([caerPolarityEventConst](#) event)
- static void [caerPolarityEventSetY](#) ([caerPolarityEvent](#) event, uint16\_t yAddress)
- static uint16\_t [caerPolarityEventGetX](#) ([caerPolarityEventConst](#) event)
- static void [caerPolarityEventSetX](#) ([caerPolarityEvent](#) event, uint16\_t xAddress)

### 4.19.1 Detailed Description

Polarity Events format definition and handling functions. This event contains change information, with an X/Y address and an ON/OFF polarity. The (0, 0) address is in the upper left corner of the screen, like in OpenCV/computer graphics.

### 4.19.2 Macro Definition Documentation

#### 4.19.2.1 CAER\_POLARITY\_CONST\_ITERATOR\_ALL\_START

```
#define CAER_POLARITY_CONST_ITERATOR_ALL_START(  
    POLARITY_PACKET )
```

**Value:**

```
for (int32_t caerPolarityIteratorCounter = 0; \  
    caerPolarityIteratorCounter < caerEventPacketHeaderGetEventNumber  
    (&(POLARITY_PACKET)->packetHeader); \  
    caerPolarityIteratorCounter++) { \  
    caerPolarityEventConst caerPolarityIteratorElement =  
    caerPolarityEventPacketGetEventConst(POLARITY_PACKET,  
    caerPolarityIteratorCounter);
```

Const-Iterator over all polarity events in a packet. Returns the current index in the 'caerPolarityIteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerPolarityIteratorElement' variable of type caer↵PolarityEventConst.

POLARITY\_PACKET: a valid PolarityEventPacket pointer. Cannot be NULL.

## 4.19.2.2 CAER\_POLARITY\_CONST\_ITERATOR\_VALID\_START

```
#define CAER_POLARITY_CONST_ITERATOR_VALID_START(
    POLARITY_PACKET )
```

**Value:**

```
for (int32_t caerPolarityIteratorCounter = 0; \
    caerPolarityIteratorCounter < caerEventPacketHeaderGetEventNumber
    (&(POLARITY_PACKET)->packetHeader); \
    caerPolarityIteratorCounter++) { \
    caerPolarityEventConst caerPolarityIteratorElement =
    caerPolarityEventPacketGetEventConst(POLARITY_PACKET,
    caerPolarityIteratorCounter); \
    if (!caerPolarityEventIsValid(caerPolarityIteratorElement)) { continue; }
```

Const-Iterator over only the valid polarity events in a packet. Returns the current index in the 'caerPolarityIteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerPolarityIteratorElement' variable of type caerPolarityEventConst.

POLARITY\_PACKET: a valid PolarityEventPacket pointer. Cannot be NULL.

## 4.19.2.3 CAER\_POLARITY\_CONST\_REVERSE\_ITERATOR\_ALL\_START

```
#define CAER_POLARITY_CONST_REVERSE_ITERATOR_ALL_START(
    POLARITY_PACKET )
```

**Value:**

```
for (int32_t caerPolarityIteratorCounter = caerEventPacketHeaderGetEventNumber
    (&(POLARITY_PACKET)->packetHeader) - 1; \
    caerPolarityIteratorCounter >= 0; \
    caerPolarityIteratorCounter--) { \
    caerPolarityEventConst caerPolarityIteratorElement =
    caerPolarityEventPacketGetEventConst(POLARITY_PACKET,
    caerPolarityIteratorCounter);
```

Const-Reverse iterator over all polarity events in a packet. Returns the current index in the 'caerPolarityIteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerPolarityIteratorElement' variable of type caerPolarityEventConst.

POLARITY\_PACKET: a valid PolarityEventPacket pointer. Cannot be NULL.

## 4.19.2.4 CAER\_POLARITY\_CONST\_REVERSE\_ITERATOR\_VALID\_START

```
#define CAER_POLARITY_CONST_REVERSE_ITERATOR_VALID_START(
    POLARITY_PACKET )
```

**Value:**

```
for (int32_t caerPolarityIteratorCounter = caerEventPacketHeaderGetEventNumber
    (&(POLARITY_PACKET)->packetHeader) - 1; \
    caerPolarityIteratorCounter >= 0; \
    caerPolarityIteratorCounter--) { \
    caerPolarityEventConst caerPolarityIteratorElement =
    caerPolarityEventPacketGetEventConst(POLARITY_PACKET,
    caerPolarityIteratorCounter); \
    if (!caerPolarityEventIsValid(caerPolarityIteratorElement)) { continue; }
```

Const-Reverse iterator over only the valid polarity events in a packet. Returns the current index in the 'caerPolarityIteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerPolarityIteratorElement' variable of type caerPolarityEventConst.

POLARITY\_PACKET: a valid PolarityEventPacket pointer. Cannot be NULL.

#### 4.19.2.5 CAER\_POLARITY\_ITERATOR\_ALL\_END

```
#define CAER_POLARITY_ITERATOR_ALL_END }
```

Iterator close statement.

#### 4.19.2.6 CAER\_POLARITY\_ITERATOR\_ALL\_START

```
#define CAER_POLARITY_ITERATOR_ALL_START(  
    POLARITY_PACKET )
```

**Value:**

```
for (int32_t caerPolarityIteratorCounter = 0; \  
    caerPolarityIteratorCounter < caerEventPacketHeaderGetEventNumber  
    (&(POLARITY_PACKET)->packetHeader); \  
    caerPolarityIteratorCounter++) { \  
    caerPolarityEvent caerPolarityIteratorElement =  
    caerPolarityEventPacketGetEvent(POLARITY_PACKET, caerPolarityIteratorCounter  
    );
```

Iterator over all polarity events in a packet. Returns the current index in the 'caerPolarityIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerPolarityIteratorElement' variable of type caerPolarityEvent.

POLARITY\_PACKET: a valid PolarityEventPacket pointer. Cannot be NULL.

#### 4.19.2.7 CAER\_POLARITY\_ITERATOR\_VALID\_END

```
#define CAER_POLARITY_ITERATOR_VALID_END }
```

Iterator close statement.

#### 4.19.2.8 CAER\_POLARITY\_ITERATOR\_VALID\_START

```
#define CAER_POLARITY_ITERATOR_VALID_START(  
    POLARITY_PACKET )
```

**Value:**

```
for (int32_t caerPolarityIteratorCounter = 0; \  
    caerPolarityIteratorCounter < caerEventPacketHeaderGetEventNumber  
    (&(POLARITY_PACKET)->packetHeader); \  
    caerPolarityIteratorCounter++) { \  
    caerPolarityEvent caerPolarityIteratorElement =  
    caerPolarityEventPacketGetEvent(POLARITY_PACKET, caerPolarityIteratorCounter  
    ); \  
    if (!caerPolarityEventIsValid(caerPolarityIteratorElement)) { continue; }
```

Iterator over only the valid polarity events in a packet. Returns the current index in the 'caerPolarityIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerPolarityIteratorElement' variable of type caerPolarityEvent.

POLARITY\_PACKET: a valid PolarityEventPacket pointer. Cannot be NULL.

## 4.19.2.9 CAER\_POLARITY\_REVERSE\_ITERATOR\_ALL\_END

```
#define CAER_POLARITY_REVERSE_ITERATOR_ALL_END }
```

Reverse iterator close statement.

## 4.19.2.10 CAER\_POLARITY\_REVERSE\_ITERATOR\_ALL\_START

```
#define CAER_POLARITY_REVERSE_ITERATOR_ALL_START (
    POLARITY_PACKET )
```

**Value:**

```
for (int32_t caerPolarityIteratorCounter = caerEventPacketHeaderGetEventNumber
    (&(POLARITY_PACKET)->packetHeader) - 1; \
    caerPolarityIteratorCounter >= 0; \
    caerPolarityIteratorCounter--) { \
    caerPolarityEvent caerPolarityIteratorElement =
    caerPolarityEventPacketGetEvent(POLARITY_PACKET, caerPolarityIteratorCounter
    );
```

Reverse iterator over all polarity events in a packet. Returns the current index in the 'caerPolarityIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerPolarityIteratorElement' variable of type caerPolarityEvent.

POLARITY\_PACKET: a valid PolarityEventPacket pointer. Cannot be NULL.

## 4.19.2.11 CAER\_POLARITY\_REVERSE\_ITERATOR\_VALID\_END

```
#define CAER_POLARITY_REVERSE_ITERATOR_VALID_END }
```

Reverse iterator close statement.

## 4.19.2.12 CAER\_POLARITY\_REVERSE\_ITERATOR\_VALID\_START

```
#define CAER_POLARITY_REVERSE_ITERATOR_VALID_START (
    POLARITY_PACKET )
```

**Value:**

```
for (int32_t caerPolarityIteratorCounter = caerEventPacketHeaderGetEventNumber
    (&(POLARITY_PACKET)->packetHeader) - 1; \
    caerPolarityIteratorCounter >= 0; \
    caerPolarityIteratorCounter--) { \
    caerPolarityEvent caerPolarityIteratorElement =
    caerPolarityEventPacketGetEvent(POLARITY_PACKET, caerPolarityIteratorCounter
    ); \
    if (!caerPolarityEventIsValid(caerPolarityIteratorElement)) { continue; }
```

Reverse iterator over only the valid polarity events in a packet. Returns the current index in the 'caerPolarityIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerPolarityIteratorElement' variable of type caerPolarityEvent.

POLARITY\_PACKET: a valid PolarityEventPacket pointer. Cannot be NULL.

#### 4.19.2.13 POLARITY\_MASK

```
#define POLARITY_MASK 0x00000001
```

Shift and mask values for the polarity, X and Y addresses of a polarity event. Addresses up to 15 bit are supported. Polarity is ON(=1) or OFF(=0). Bit 0 is the valid mark, see '[common.h](#)' for more details.

#### 4.19.2.14 POLARITY\_SHIFT

```
#define POLARITY_SHIFT 1
```

Shift and mask values for the polarity, X and Y addresses of a polarity event. Addresses up to 15 bit are supported. Polarity is ON(=1) or OFF(=0). Bit 0 is the valid mark, see '[common.h](#)' for more details.

#### 4.19.2.15 POLARITY\_X\_ADDR\_MASK

```
#define POLARITY_X_ADDR_MASK 0x00007FFF
```

Shift and mask values for the polarity, X and Y addresses of a polarity event. Addresses up to 15 bit are supported. Polarity is ON(=1) or OFF(=0). Bit 0 is the valid mark, see '[common.h](#)' for more details.

#### 4.19.2.16 POLARITY\_X\_ADDR\_SHIFT

```
#define POLARITY_X_ADDR_SHIFT 17
```

Shift and mask values for the polarity, X and Y addresses of a polarity event. Addresses up to 15 bit are supported. Polarity is ON(=1) or OFF(=0). Bit 0 is the valid mark, see '[common.h](#)' for more details.

#### 4.19.2.17 POLARITY\_Y\_ADDR\_MASK

```
#define POLARITY_Y_ADDR_MASK 0x00007FFF
```

Shift and mask values for the polarity, X and Y addresses of a polarity event. Addresses up to 15 bit are supported. Polarity is ON(=1) or OFF(=0). Bit 0 is the valid mark, see '[common.h](#)' for more details.

#### 4.19.2.18 POLARITY\_Y\_ADDR\_SHIFT

```
#define POLARITY_Y_ADDR_SHIFT 2
```

Shift and mask values for the polarity, X and Y addresses of a polarity event. Addresses up to 15 bit are supported. Polarity is ON(=1) or OFF(=0). Bit 0 is the valid mark, see '[common.h](#)' for more details.

### 4.19.3 Typedef Documentation



#### 4.19.3.1 caerPolarityEvent

```
typedef struct caer_polarity_event* caerPolarityEvent
```

Type for pointer to polarity event data structure.

#### 4.19.3.2 caerPolarityEventPacket

```
typedef struct caer_polarity_event_packet* caerPolarityEventPacket
```

Type for pointer to polarity event packet data structure.

### 4.19.4 Function Documentation

#### 4.19.4.1 caerPolarityEventGetPolarity()

```
static bool caerPolarityEventGetPolarity (  
    caerPolarityEventConst event ) [inline], [static]
```

Get the change event polarity. 1 is ON, 0 is OFF.

##### Parameters

<i>event</i>	a valid PolarityEvent pointer. Cannot be NULL.
--------------	--

##### Returns

event polarity value.

#### 4.19.4.2 caerPolarityEventGetTimestamp()

```
static int32_t caerPolarityEventGetTimestamp (  
    caerPolarityEventConst event ) [inline], [static]
```

Get the 32bit event timestamp, in microseconds. Be aware that this wraps around! You can either ignore this fact, or handle the special 'TIMESTAMP\_WRAP' event that is generated when this happens, or use the 64bit timestamp which never wraps around. See '[caerEventPacketHeaderGetEventTSOverflow\(\)](#)' documentation for more details on the 64bit timestamp.

##### Parameters

<i>event</i>	a valid PolarityEvent pointer. Cannot be NULL.
--------------	--

**Returns**

this event's 32bit microsecond timestamp.

**4.19.4.3 caerPolarityEventGetTimestamp64()**

```
static int64_t caerPolarityEventGetTimestamp64 (
    caerPolarityEventConst event,
    caerPolarityEventPacketConst packet ) [inline], [static]
```

Get the 64bit event timestamp, in microseconds. See '[caerEventPacketHeaderGetEventTSOverflow\(\)](#)' documentation for more details on the 64bit timestamp.

**Parameters**

<i>event</i>	a valid PolarityEvent pointer. Cannot be NULL.
<i>packet</i>	the PolarityEventPacket pointer for the packet containing this event. Cannot be NULL.

**Returns**

this event's 64bit microsecond timestamp.

**4.19.4.4 caerPolarityEventGetX()**

```
static uint16_t caerPolarityEventGetX (
    caerPolarityEventConst event ) [inline], [static]
```

Get the X (column) address for a change event, in pixels. The (0, 0) address is in the upper left corner, like in OpenCV/computer graphics.

**Parameters**

<i>event</i>	a valid PolarityEvent pointer. Cannot be NULL.
--------------	--

**Returns**

the event X address.

**4.19.4.5 caerPolarityEventGetY()**

```
static uint16_t caerPolarityEventGetY (
    caerPolarityEventConst event ) [inline], [static]
```

Get the Y (row) address for a change event, in pixels. The (0, 0) address is in the upper left corner, like in OpenCV/computer graphics.

**Parameters**

<i>event</i>	a valid PolarityEvent pointer. Cannot be NULL.
--------------	--

**Returns**

the event Y address.

**4.19.4.6 caerPolarityEventInvalidate()**

```
static void caerPolarityEventInvalidate (
    caerPolarityEvent event,
    caerPolarityEventPacket packet ) [inline], [static]
```

Invalidate the current event by setting its valid bit to false and decreasing the number of valid events held in the packet. Only works with events that are already valid!

**Parameters**

<i>event</i>	a valid PolarityEvent pointer. Cannot be NULL.
<i>packet</i>	the PolarityEventPacket pointer for the packet containing this event. Cannot be NULL.

**4.19.4.7 caerPolarityEventIsValid()**

```
static bool caerPolarityEventIsValid (
    caerPolarityEventConst event ) [inline], [static]
```

Check if this polarity event is valid.

**Parameters**

<i>event</i>	a valid PolarityEvent pointer. Cannot be NULL.
--------------	--

**Returns**

true if valid, false if not.

**4.19.4.8 caerPolarityEventPacketAllocate()**

```
caerPolarityEventPacket caerPolarityEventPacketAllocate (
    int32_t eventCapacity,
```

```

    int16_t eventSource,
    int32_t tsOverflow )

```

Allocate a new polarity events packet. Use free() to reclaim this memory.

#### Parameters

<i>eventCapacity</i>	the maximum number of events this packet will hold.
<i>eventSource</i>	the unique ID representing the source/generator of this packet.
<i>tsOverflow</i>	the current timestamp overflow counter value for this packet.

#### Returns

a valid PolarityEventPacket handle or NULL on error.

#### 4.19.4.9 caerPolarityEventPacketGetEvent()

```

static caerPolarityEvent caerPolarityEventPacketGetEvent (
    caerPolarityEventPacket packet,
    int32_t n ) [inline], [static]

```

Get the polarity event at the given index from the event packet.

#### Parameters

<i>packet</i>	a valid PolarityEventPacket pointer. Cannot be NULL.
<i>n</i>	the index of the returned event. Must be within [0,eventCapacity[ bounds.

#### Returns

the requested polarity event. NULL on error.

#### 4.19.4.10 caerPolarityEventPacketGetEventConst()

```

static caerPolarityEventConst caerPolarityEventPacketGetEventConst (
    caerPolarityEventPacketConst packet,
    int32_t n ) [inline], [static]

```

Get the polarity event at the given index from the event packet. This is a read-only event, do not change its contents in any way!

#### Parameters

<i>packet</i>	a valid PolarityEventPacket pointer. Cannot be NULL.
<i>n</i>	the index of the returned event. Must be within [0,eventCapacity[ bounds.

**Returns**

the requested read-only polarity event. NULL on error.

**4.19.4.11 caerPolarityEventSetPolarity()**

```
static void caerPolarityEventSetPolarity (
    caerPolarityEvent event,
    bool polarity ) [inline], [static]
```

Set the change event polarity. 1 is ON, 0 is OFF.

**Parameters**

<i>event</i>	a valid PolarityEvent pointer. Cannot be NULL.
<i>polarity</i>	event polarity value.

**4.19.4.12 caerPolarityEventSetTimestamp()**

```
static void caerPolarityEventSetTimestamp (
    caerPolarityEvent event,
    int32_t timestamp ) [inline], [static]
```

Set the 32bit event timestamp, the value has to be in microseconds.

**Parameters**

<i>event</i>	a valid PolarityEvent pointer. Cannot be NULL.
<i>timestamp</i>	a positive 32bit microsecond timestamp.

**4.19.4.13 caerPolarityEventSetX()**

```
static void caerPolarityEventSetX (
    caerPolarityEvent event,
    uint16_t xAddress ) [inline], [static]
```

Set the X (column) address for a change event, in pixels. The (0, 0) address is in the upper left corner, like in OpenCV/computer graphics.

**Parameters**

<i>event</i>	a valid PolarityEvent pointer. Cannot be NULL.
<i>xAddress</i>	the event X address.

#### 4.19.4.14 caerPolarityEventSetY()

```
static void caerPolarityEventSetY (
    caerPolarityEvent event,
    uint16_t yAddress ) [inline], [static]
```

Set the Y (row) address for a change event, in pixels. The (0, 0) address is in the upper left corner, like in OpenCV/computer graphics.

##### Parameters

<i>event</i>	a valid PolarityEvent pointer. Cannot be NULL.
<i>yAddress</i>	the event Y address.

#### 4.19.4.15 caerPolarityEventValidate()

```
static void caerPolarityEventValidate (
    caerPolarityEvent event,
    caerPolarityEventPacket packet ) [inline], [static]
```

Validate the current event by setting its valid bit to true and increasing the event packet's event count and valid event count. Only works on events that are invalid. DO NOT CALL THIS AFTER HAVING PREVIOUSLY ALREADY INVALIDATED THIS EVENT, the total count will be incorrect.

##### Parameters

<i>event</i>	a valid PolarityEvent pointer. Cannot be NULL.
<i>packet</i>	the PolarityEventPacket pointer for the packet containing this event. Cannot be NULL.

#### 4.19.4.16 PACKED\_STRUCT() [1/2]

```
PACKED_STRUCT (
    struct caer_polarity_event { uint32_t data;int32_t timestamp;} )
```

Polarity event data structure definition. This contains the actual X/Y addresses, the polarity, as well as the 32 bit event timestamp. The (0, 0) address is in the upper left corner of the screen, like in OpenCV/computer graphics. Signed integers are used for fields that are to be interpreted directly, for compatibility with languages that do not have unsigned integer types, such as Java.

## 4.19.4.17 PACKED\_STRUCT() [2/2]

```
PACKED_STRUCT (
    struct caer_polarity_event_packet { struct caer_event_packet_header packet↔
Header; struct caer_polarity_event events[]; } )
```

Polarity event packet data structure definition. EventPackets are always made up of the common packet header, followed by 'eventCapacity' events. Everything has to be in one contiguous memory block.

## 4.20 events/sample.h File Reference

```
#include "common.h"
```

## Macros

- #define [CAER\\_SAMPLE\\_ITERATOR\\_ALL\\_START](#)(SAMPLE\_PACKET)
  - #define [CAER\\_SAMPLE\\_CONST\\_ITERATOR\\_ALL\\_START](#)(SAMPLE\_PACKET)
  - #define [CAER\\_SAMPLE\\_ITERATOR\\_ALL\\_END](#) }
  - #define [CAER\\_SAMPLE\\_ITERATOR\\_VALID\\_START](#)(SAMPLE\_PACKET)
  - #define [CAER\\_SAMPLE\\_CONST\\_ITERATOR\\_VALID\\_START](#)(SAMPLE\_PACKET)
  - #define [CAER\\_SAMPLE\\_ITERATOR\\_VALID\\_END](#) }
  - #define [CAER\\_SAMPLE\\_REVERSE\\_ITERATOR\\_ALL\\_START](#)(SAMPLE\_PACKET)
  - #define [CAER\\_SAMPLE\\_CONST\\_REVERSE\\_ITERATOR\\_ALL\\_START](#)(SAMPLE\_PACKET)
  - #define [CAER\\_SAMPLE\\_REVERSE\\_ITERATOR\\_ALL\\_END](#) }
  - #define [CAER\\_SAMPLE\\_REVERSE\\_ITERATOR\\_VALID\\_START](#)(SAMPLE\_PACKET)
  - #define [CAER\\_SAMPLE\\_CONST\\_REVERSE\\_ITERATOR\\_VALID\\_START](#)(SAMPLE\_PACKET)
  - #define [CAER\\_SAMPLE\\_REVERSE\\_ITERATOR\\_VALID\\_END](#) }
- 
- #define [SAMPLE\\_TYPE\\_SHIFT](#) 1
  - #define [SAMPLE\\_TYPE\\_MASK](#) 0x0000007F
  - #define [SAMPLE\\_SHIFT](#) 8
  - #define [SAMPLE\\_MASK](#) 0x00FFFFFF

## Typedefs

- typedef struct caer\_sample\_event \* [caerSampleEvent](#)
- typedef const struct caer\_sample\_event \* **caerSampleEventConst**
- typedef struct caer\_sample\_event\_packet \* [caerSampleEventPacket](#)
- typedef const struct caer\_sample\_event\_packet \* **caerSampleEventPacketConst**

## Functions

- [PACKED\\_STRUCT](#) (struct caer\_sample\_event { uint32\_t data;int32\_t timestamp;})
- [PACKED\\_STRUCT](#) (struct caer\_sample\_event\_packet { struct caer\_event\_packet\_header packet←Header;struct caer\_sample\_event events[ ];})
- [caerSampleEventPacket](#) [caerSampleEventPacketAllocate](#) (int32\_t eventCapacity, int16\_t eventSource, int32\_t tsOverflow)
- static [caerSampleEvent](#) [caerSampleEventPacketGetEvent](#) ([caerSampleEventPacket](#) packet, int32\_t n)
- static caerSampleEventConst [caerSampleEventPacketGetEventConst](#) ([caerSampleEventPacketConst](#) packet, int32\_t n)
- static int32\_t [caerSampleEventGetTimestamp](#) ([caerSampleEventConst](#) event)
- static int64\_t [caerSampleEventGetTimestamp64](#) ([caerSampleEventConst](#) event, [caerSampleEventPacket](#)←Const packet)
- static void [caerSampleEventSetTimestamp](#) ([caerSampleEvent](#) event, int32\_t timestamp)
- static bool [caerSampleEventIsValid](#) ([caerSampleEventConst](#) event)
- static void [caerSampleEventValidate](#) ([caerSampleEvent](#) event, [caerSampleEventPacket](#) packet)
- static void [caerSampleEventInvalidate](#) ([caerSampleEvent](#) event, [caerSampleEventPacket](#) packet)
- static uint8\_t [caerSampleEventGetType](#) ([caerSampleEventConst](#) event)
- static void [caerSampleEventSetType](#) ([caerSampleEvent](#) event, uint8\_t type)
- static uint32\_t [caerSampleEventGetSample](#) ([caerSampleEventConst](#) event)
- static void [caerSampleEventSetSample](#) ([caerSampleEvent](#) event, uint32\_t sample)

### 4.20.1 Detailed Description

Sample (ADC) Events format definition and handling functions. Represents different types of ADC readings, up to 24 bits of resolution.

### 4.20.2 Macro Definition Documentation

#### 4.20.2.1 CAER\_SAMPLE\_CONST\_ITERATOR\_ALL\_START

```
#define CAER_SAMPLE_CONST_ITERATOR_ALL_START(  
    SAMPLE_PACKET )
```

**Value:**

```
for (int32_t caerSampleIteratorCounter = 0; \  
    caerSampleIteratorCounter < caerEventPacketHeaderGetEventNumber(  
    &(SAMPLE_PACKET)->packetHeader); \  
    caerSampleIteratorCounter++) { \  
    caerSampleEventConst caerSampleIteratorElement =  
    caerSampleEventPacketGetEventConst (SAMPLE_PACKET,  
    caerSampleIteratorCounter);
```

Const-Iterator over all sample events in a packet. Returns the current index in the 'caerSampleIteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerSampleIteratorElement' variable of type caer←SampleEventConst.

SAMPLE\_PACKET: a valid SampleEventPacket pointer. Cannot be NULL.



## 4.20.2.2 CAER\_SAMPLE\_CONST\_ITERATOR\_VALID\_START

```
#define CAER_SAMPLE_CONST_ITERATOR_VALID_START(
    SAMPLE_PACKET )
```

**Value:**

```
for (int32_t caerSampleIteratorCounter = 0; \
    caerSampleIteratorCounter < caerEventPacketHeaderGetEventNumber(
    &(SAMPLE_PACKET)->packetHeader); \
    caerSampleIteratorCounter++) { \
    caerSampleEventConst caerSampleIteratorElement =
    caerSampleEventPacketGetEventConst(SAMPLE_PACKET,
    caerSampleIteratorCounter); \
    if (!caerSampleEventIsValid(caerSampleIteratorElement)) { continue; }
```

Const-Iterator over only the valid sample events in a packet. Returns the current index in the 'caerSampleIteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerSampleIteratorElement' variable of type caerSampleEventConst.

SAMPLE\_PACKET: a valid SampleEventPacket pointer. Cannot be NULL.

## 4.20.2.3 CAER\_SAMPLE\_CONST\_REVERSE\_ITERATOR\_ALL\_START

```
#define CAER_SAMPLE_CONST_REVERSE_ITERATOR_ALL_START(
    SAMPLE_PACKET )
```

**Value:**

```
for (int32_t caerSampleIteratorCounter = caerEventPacketHeaderGetEventNumber
    (&(SAMPLE_PACKET)->packetHeader) - 1; \
    caerSampleIteratorCounter >= 0; \
    caerSampleIteratorCounter--) { \
    caerSampleEventConst caerSampleIteratorElement =
    caerSampleEventPacketGetEventConst(SAMPLE_PACKET,
    caerSampleIteratorCounter);
```

Const-Reverse iterator over all sample events in a packet. Returns the current index in the 'caerSampleIteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerSampleIteratorElement' variable of type caerSampleEventConst.

SAMPLE\_PACKET: a valid SampleEventPacket pointer. Cannot be NULL.

## 4.20.2.4 CAER\_SAMPLE\_CONST\_REVERSE\_ITERATOR\_VALID\_START

```
#define CAER_SAMPLE_CONST_REVERSE_ITERATOR_VALID_START(
    SAMPLE_PACKET )
```

**Value:**

```
for (int32_t caerSampleIteratorCounter = caerEventPacketHeaderGetEventNumber
    (&(SAMPLE_PACKET)->packetHeader) - 1; \
    caerSampleIteratorCounter >= 0; \
    caerSampleIteratorCounter--) { \
    caerSampleEventConst caerSampleIteratorElement =
    caerSampleEventPacketGetEventConst(SAMPLE_PACKET,
    caerSampleIteratorCounter); \
    if (!caerSampleEventIsValid(caerSampleIteratorElement)) { continue; }
```

Const-Reverse iterator over only the valid sample events in a packet. Returns the current index in the 'caerSampleIteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerSampleIteratorElement' variable of type caerSampleEventConst.

SAMPLE\_PACKET: a valid SampleEventPacket pointer. Cannot be NULL.

#### 4.20.2.5 CAER\_SAMPLE\_ITERATOR\_ALL\_END

```
#define CAER_SAMPLE_ITERATOR_ALL_END }
```

Iterator close statement.

#### 4.20.2.6 CAER\_SAMPLE\_ITERATOR\_ALL\_START

```
#define CAER_SAMPLE_ITERATOR_ALL_START(  
    SAMPLE_PACKET )
```

**Value:**

```
for (int32_t caerSampleIteratorCounter = 0; \  
    caerSampleIteratorCounter < caerEventPacketHeaderGetEventNumber(  
        &(SAMPLE_PACKET)->packetHeader); \  
    caerSampleIteratorCounter++) { \  
    caerSampleEvent caerSampleIteratorElement =  
        caerSampleEventPacketGetEvent(SAMPLE_PACKET, caerSampleIteratorCounter);
```

Iterator over all sample events in a packet. Returns the current index in the 'caerSampleIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerSampleIteratorElement' variable of type caerSampleEvent.

SAMPLE\_PACKET: a valid SampleEventPacket pointer. Cannot be NULL.

#### 4.20.2.7 CAER\_SAMPLE\_ITERATOR\_VALID\_END

```
#define CAER_SAMPLE_ITERATOR_VALID_END }
```

Iterator close statement.

#### 4.20.2.8 CAER\_SAMPLE\_ITERATOR\_VALID\_START

```
#define CAER_SAMPLE_ITERATOR_VALID_START(  
    SAMPLE_PACKET )
```

**Value:**

```
for (int32_t caerSampleIteratorCounter = 0; \  
    caerSampleIteratorCounter < caerEventPacketHeaderGetEventNumber(  
        &(SAMPLE_PACKET)->packetHeader); \  
    caerSampleIteratorCounter++) { \  
    caerSampleEvent caerSampleIteratorElement =  
        caerSampleEventPacketGetEvent(SAMPLE_PACKET, caerSampleIteratorCounter); \  
    if (!caerSampleEventIsValid(caerSampleIteratorElement)) { continue; }
```

Iterator over only the valid sample events in a packet. Returns the current index in the 'caerSampleIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerSampleIteratorElement' variable of type caerSampleEvent.

SAMPLE\_PACKET: a valid SampleEventPacket pointer. Cannot be NULL.

## 4.20.2.9 CAER\_SAMPLE\_REVERSE\_ITERATOR\_ALL\_END

```
#define CAER_SAMPLE_REVERSE_ITERATOR_ALL_END }
```

Reverse iterator close statement.

## 4.20.2.10 CAER\_SAMPLE\_REVERSE\_ITERATOR\_ALL\_START

```
#define CAER_SAMPLE_REVERSE_ITERATOR_ALL_START(  
    SAMPLE_PACKET )
```

**Value:**

```
for (int32_t caerSampleIteratorCounter = caerEventPacketHeaderGetEventNumber  
    (&(SAMPLE_PACKET)->packetHeader) - 1; \  
    caerSampleIteratorCounter >= 0; \  
    caerSampleIteratorCounter--) { \  
    caerSampleEvent caerSampleIteratorElement =  
    caerSampleEventPacketGetEvent(SAMPLE_PACKET, caerSampleIteratorCounter);
```

Reverse iterator over all sample events in a packet. Returns the current index in the 'caerSampleIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerSampleIteratorElement' variable of type caerSampleEvent.

SAMPLE\_PACKET: a valid SampleEventPacket pointer. Cannot be NULL.

## 4.20.2.11 CAER\_SAMPLE\_REVERSE\_ITERATOR\_VALID\_END

```
#define CAER_SAMPLE_REVERSE_ITERATOR_VALID_END }
```

Reverse iterator close statement.

## 4.20.2.12 CAER\_SAMPLE\_REVERSE\_ITERATOR\_VALID\_START

```
#define CAER_SAMPLE_REVERSE_ITERATOR_VALID_START(  
    SAMPLE_PACKET )
```

**Value:**

```
for (int32_t caerSampleIteratorCounter = caerEventPacketHeaderGetEventNumber  
    (&(SAMPLE_PACKET)->packetHeader) - 1; \  
    caerSampleIteratorCounter >= 0; \  
    caerSampleIteratorCounter--) { \  
    caerSampleEvent caerSampleIteratorElement =  
    caerSampleEventPacketGetEvent(SAMPLE_PACKET, caerSampleIteratorCounter); \  
    if (!caerSampleEventIsValid(caerSampleIteratorElement)) { continue; }
```

Reverse iterator over only the valid sample events in a packet. Returns the current index in the 'caerSampleIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerSampleIteratorElement' variable of type caerSampleEvent.

SAMPLE\_PACKET: a valid SampleEventPacket pointer. Cannot be NULL.

#### 4.20.2.13 SAMPLE\_MASK

```
#define SAMPLE_MASK 0x00FFFFFF
```

Shift and mask values for the sample type and the actual sample value of an ADC sample. Up to 128 sample types are supported, with 24 bits of data per sample. Higher values mean a higher voltage, 0 is ground. Bit 0 is the valid mark, see '[common.h](#)' for more details.

#### 4.20.2.14 SAMPLE\_SHIFT

```
#define SAMPLE_SHIFT 8
```

Shift and mask values for the sample type and the actual sample value of an ADC sample. Up to 128 sample types are supported, with 24 bits of data per sample. Higher values mean a higher voltage, 0 is ground. Bit 0 is the valid mark, see '[common.h](#)' for more details.

#### 4.20.2.15 SAMPLE\_TYPE\_MASK

```
#define SAMPLE_TYPE_MASK 0x0000007F
```

Shift and mask values for the sample type and the actual sample value of an ADC sample. Up to 128 sample types are supported, with 24 bits of data per sample. Higher values mean a higher voltage, 0 is ground. Bit 0 is the valid mark, see '[common.h](#)' for more details.

#### 4.20.2.16 SAMPLE\_TYPE\_SHIFT

```
#define SAMPLE_TYPE_SHIFT 1
```

Shift and mask values for the sample type and the actual sample value of an ADC sample. Up to 128 sample types are supported, with 24 bits of data per sample. Higher values mean a higher voltage, 0 is ground. Bit 0 is the valid mark, see '[common.h](#)' for more details.

### 4.20.3 Typedef Documentation

#### 4.20.3.1 caerSampleEvent

```
typedef struct caer_sample_event* caerSampleEvent
```

Type for pointer to ADC sample event data structure.

#### 4.20.3.2 caerSampleEventPacket

```
typedef struct caer_sample_event_packet* caerSampleEventPacket
```

Type for pointer to ADC sample event packet data structure.

## 4.20.4 Function Documentation

### 4.20.4.1 caerSampleEventGetSample()

```
static uint32_t caerSampleEventGetSample (
    caerSampleEventConst event ) [inline], [static]
```

Get the ADC sample value. Up to 24 bits of resolution are possible. Higher values mean a higher voltage, 0 is ground.

#### Parameters

<i>event</i>	a valid SampleEvent pointer. Cannot be NULL.
--------------	--

#### Returns

the ADC sample value.

### 4.20.4.2 caerSampleEventGetTimestamp()

```
static int32_t caerSampleEventGetTimestamp (
    caerSampleEventConst event ) [inline], [static]
```

Get the 32bit event timestamp, in microseconds. Be aware that this wraps around! You can either ignore this fact, or handle the special 'TIMESTAMP\_WRAP' event that is generated when this happens, or use the 64bit timestamp which never wraps around. See '[caerEventPacketHeaderGetEventTSOverflow\(\)](#)' documentation for more details on the 64bit timestamp.

#### Parameters

<i>event</i>	a valid SampleEvent pointer. Cannot be NULL.
--------------	--

#### Returns

this event's 32bit microsecond timestamp.

### 4.20.4.3 caerSampleEventGetTimestamp64()

```
static int64_t caerSampleEventGetTimestamp64 (
    caerSampleEventConst event,
    caerSampleEventPacketConst packet ) [inline], [static]
```

Get the 64bit event timestamp, in microseconds. See '[caerEventPacketHeaderGetEventTSOverflow\(\)](#)' documentation for more details on the 64bit timestamp.

**Parameters**

<i>event</i>	a valid SampleEvent pointer. Cannot be NULL.
<i>packet</i>	the SampleEventPacket pointer for the packet containing this event. Cannot be NULL.

**Returns**

this event's 64bit microsecond timestamp.

**4.20.4.4 caerSampleEventGetType()**

```
static uint8_t caerSampleEventGetType (  
    caerSampleEventConst event ) [inline], [static]
```

Get the ADC sample event type. This is useful to distinguish between different measurements, for example from two separate microphones on a device.

**Parameters**

<i>event</i>	a valid SampleEvent pointer. Cannot be NULL.
--------------	--

**Returns**

the ADC sample type.

**4.20.4.5 caerSampleEventInvalidate()**

```
static void caerSampleEventInvalidate (  
    caerSampleEvent event,  
    caerSampleEventPacket packet ) [inline], [static]
```

Invalidate the current event by setting its valid bit to false and decreasing the number of valid events held in the packet. Only works with events that are already valid!

**Parameters**

<i>event</i>	a valid SampleEvent pointer. Cannot be NULL.
<i>packet</i>	the SampleEventPacket pointer for the packet containing this event. Cannot be NULL.

**4.20.4.6 caerSampleEventIsValid()**

```
static bool caerSampleEventIsValid (  

```

```
caerSampleEventConst event ) [inline], [static]
```

Check if this ADC sample event is valid.

#### Parameters

<i>event</i>	a valid SampleEvent pointer. Cannot be NULL.
--------------	--

#### Returns

true if valid, false if not.

#### 4.20.4.7 caerSampleEventPacketAllocate()

```
caerSampleEventPacket caerSampleEventPacketAllocate (
    int32_t eventCapacity,
    int16_t eventSource,
    int32_t tsOverflow )
```

Allocate a new ADC sample events packet. Use free() to reclaim this memory.

#### Parameters

<i>eventCapacity</i>	the maximum number of events this packet will hold.
<i>eventSource</i>	the unique ID representing the source/generator of this packet.
<i>tsOverflow</i>	the current timestamp overflow counter value for this packet.

#### Returns

a valid SampleEventPacket handle or NULL on error.

#### 4.20.4.8 caerSampleEventPacketGetEvent()

```
static caerSampleEvent caerSampleEventPacketGetEvent (
    caerSampleEventPacket packet,
    int32_t n ) [inline], [static]
```

Get the ADC sample event at the given index from the event packet.

#### Parameters

<i>packet</i>	a valid SampleEventPacket pointer. Cannot be NULL.
<i>n</i>	the index of the returned event. Must be within [0,eventCapacity[ bounds.

**Returns**

the requested ADC sample event. NULL on error.

**4.20.4.9 caerSampleEventPacketGetEventConst()**

```
static caerSampleEventConst caerSampleEventPacketGetEventConst (
    caerSampleEventPacketConst packet,
    int32_t n ) [inline], [static]
```

Get the ADC sample event at the given index from the event packet. This is a read-only event, do not change its contents in any way!

**Parameters**

<i>packet</i>	a valid SampleEventPacket pointer. Cannot be NULL.
<i>n</i>	the index of the returned event. Must be within [0,eventCapacity[ bounds.

**Returns**

the requested read-only ADC sample event. NULL on error.

**4.20.4.10 caerSampleEventSetSample()**

```
static void caerSampleEventSetSample (
    caerSampleEvent event,
    uint32_t sample ) [inline], [static]
```

Set the ADC sample value. Up to 24 bits of resolution are possible. Higher values mean a higher voltage, 0 is ground.

**Parameters**

<i>event</i>	a valid SampleEvent pointer. Cannot be NULL.
<i>sample</i>	the ADC sample value.

**4.20.4.11 caerSampleEventSetTimestamp()**

```
static void caerSampleEventSetTimestamp (
    caerSampleEvent event,
    int32_t timestamp ) [inline], [static]
```

Set the 32bit event timestamp, the value has to be in microseconds.



## Parameters

<i>event</i>	a valid SampleEvent pointer. Cannot be NULL.
<i>timestamp</i>	a positive 32bit microsecond timestamp.

## 4.20.4.12 caerSampleEventSetType()

```
static void caerSampleEventSetType (
    caerSampleEvent event,
    uint8_t type ) [inline], [static]
```

Set the ADC sample event type. This is useful to distinguish between different measurements, for example from two separate microphones on a device.

## Parameters

<i>event</i>	a valid SampleEvent pointer. Cannot be NULL.
<i>type</i>	the ADC sample type.

## 4.20.4.13 caerSampleEventValidate()

```
static void caerSampleEventValidate (
    caerSampleEvent event,
    caerSampleEventPacket packet ) [inline], [static]
```

Validate the current event by setting its valid bit to true and increasing the event packet's event count and valid event count. Only works on events that are invalid. DO NOT CALL THIS AFTER HAVING PREVIOUSLY ALREADY INVALIDATED THIS EVENT, the total count will be incorrect.

## Parameters

<i>event</i>	a valid SampleEvent pointer. Cannot be NULL.
<i>packet</i>	the SampleEventPacket pointer for the packet containing this event. Cannot be NULL.

## 4.20.4.14 PACKED\_STRUCT() [1/2]

```
PACKED_STRUCT (
    struct caer_sample_event { uint32_t data; int32_t timestamp; } )
```

ADC sample event data structure definition. Contains a type indication to separate different ADC readouts, as well as a value for that readout, up to 24 bits resolution. Signed integers are used for fields that are to be interpreted directly, for compatibility with languages that do not have unsigned integer types, such as Java.

## 4.20.4.15 PACKED\_STRUCT() [2/2]

```
PACKED_STRUCT (
    struct caer_sample_event_packet { struct caer_event_packet_header packetHeader; struct
    caer_sample_event events[]; } )
```

ADC sample event packet data structure definition. EventPackets are always made up of the common packet header, followed by 'eventCapacity' events. Everything has to be in one contiguous memory block.

## 4.21 events/special.h File Reference

```
#include "common.h"
```

## Macros

- #define [CAER\\_SPECIAL\\_ITERATOR\\_ALL\\_START](#)(SPECIAL\_PACKET)
  - #define [CAER\\_SPECIAL\\_CONST\\_ITERATOR\\_ALL\\_START](#)(SPECIAL\_PACKET)
  - #define [CAER\\_SPECIAL\\_ITERATOR\\_ALL\\_END](#) }
  - #define [CAER\\_SPECIAL\\_ITERATOR\\_VALID\\_START](#)(SPECIAL\_PACKET)
  - #define [CAER\\_SPECIAL\\_CONST\\_ITERATOR\\_VALID\\_START](#)(SPECIAL\_PACKET)
  - #define [CAER\\_SPECIAL\\_ITERATOR\\_VALID\\_END](#) }
  - #define [CAER\\_SPECIAL\\_REVERSE\\_ITERATOR\\_ALL\\_START](#)(SPECIAL\_PACKET)
  - #define [CAER\\_SPECIAL\\_CONST\\_REVERSE\\_ITERATOR\\_ALL\\_START](#)(SPECIAL\_PACKET)
  - #define [CAER\\_SPECIAL\\_REVERSE\\_ITERATOR\\_ALL\\_END](#) }
  - #define [CAER\\_SPECIAL\\_REVERSE\\_ITERATOR\\_VALID\\_START](#)(SPECIAL\_PACKET)
  - #define [CAER\\_SPECIAL\\_CONST\\_REVERSE\\_ITERATOR\\_VALID\\_START](#)(SPECIAL\_PACKET)
  - #define [CAER\\_SPECIAL\\_REVERSE\\_ITERATOR\\_VALID\\_END](#) }
- 
- #define [SPECIAL\\_TYPE\\_SHIFT](#) 1
  - #define [SPECIAL\\_TYPE\\_MASK](#) 0x0000007F
  - #define [SPECIAL\\_DATA\\_SHIFT](#) 8
  - #define [SPECIAL\\_DATA\\_MASK](#) 0x00FFFFFF

## Typedefs

- typedef struct caer\_special\_event \* [caerSpecialEvent](#)
- typedef const struct caer\_special\_event \* **caerSpecialEventConst**
- typedef struct caer\_special\_event\_packet \* [caerSpecialEventPacket](#)
- typedef const struct caer\_special\_event\_packet \* **caerSpecialEventPacketConst**

## Enumerations

- enum `caer_special_event_types` {  
`TIMESTAMP_WRAP` = 0, `TIMESTAMP_RESET` = 1, `EXTERNAL_INPUT_RISING_EDGE` = 2, `EXTERNAL_INPUT_FALLING_EDGE` = 3,  
`EXTERNAL_INPUT_PULSE` = 4, `DVS_ROW_ONLY` = 5, `EXTERNAL_INPUT1_RISING_EDGE` = 6, `EXTERNAL_INPUT1_FALLING_EDGE` = 7,  
`EXTERNAL_INPUT1_PULSE` = 8, `EXTERNAL_INPUT2_RISING_EDGE` = 9, `EXTERNAL_INPUT2_FALLING_EDGE` = 10, `EXTERNAL_INPUT2_PULSE` = 11,  
`EXTERNAL_GENERATOR_RISING_EDGE` = 12, `EXTERNAL_GENERATOR_FALLING_EDGE` = 13, `APS_FRAME_START` = 14, `APS_FRAME_END` = 15,  
`APS_EXPOSURE_START` = 16, `APS_EXPOSURE_END` = 17 }

## Functions

- `PACKED_STRUCT` (struct `caer_special_event` { `uint32_t` data; `int32_t` timestamp; })
- `PACKED_STRUCT` (struct `caer_special_event_packet` { struct `caer_event_packet_header` packet\_header; struct `caer_special_event` events[ ]; })
- `caerSpecialEventPacket` `caerSpecialEventPacketAllocate` (`int32_t` eventCapacity, `int16_t` eventSource, `int32_t` tsOverflow)
- static `caerSpecialEvent` `caerSpecialEventPacketGetEvent` (`caerSpecialEventPacket` packet, `int32_t` n)
- static `caerSpecialEventConst` `caerSpecialEventPacketGetEventConst` (`caerSpecialEventPacketConst` packet, `int32_t` n)
- static `int32_t` `caerSpecialEventGetTimestamp` (`caerSpecialEventConst` event)
- static `int64_t` `caerSpecialEventGetTimestamp64` (`caerSpecialEventConst` event, `caerSpecialEventPacketConst` packet)
- static void `caerSpecialEventSetTimestamp` (`caerSpecialEvent` event, `int32_t` timestamp)
- static bool `caerSpecialEventsIsValid` (`caerSpecialEventConst` event)
- static void `caerSpecialEventValidate` (`caerSpecialEvent` event, `caerSpecialEventPacket` packet)
- static void `caerSpecialEventInvalidate` (`caerSpecialEvent` event, `caerSpecialEventPacket` packet)
- static `uint8_t` `caerSpecialEventGetType` (`caerSpecialEventConst` event)
- static void `caerSpecialEventSetType` (`caerSpecialEvent` event, `uint8_t` type)
- static `uint32_t` `caerSpecialEventGetData` (`caerSpecialEventConst` event)
- static void `caerSpecialEventSetData` (`caerSpecialEvent` event, `uint32_t` data)
- static `caerSpecialEvent` `caerSpecialEventPacketFindEventByType` (`caerSpecialEventPacket` packet, `uint8_t` type)
- static `caerSpecialEventConst` `caerSpecialEventPacketFindEventByTypeConst` (`caerSpecialEventPacketConst` packet, `uint8_t` type)
- static `caerSpecialEvent` `caerSpecialEventPacketFindValidEventByType` (`caerSpecialEventPacket` packet, `uint8_t` type)
- static `caerSpecialEventConst` `caerSpecialEventPacketFindValidEventByTypeConst` (`caerSpecialEventPacketConst` packet, `uint8_t` type)

### 4.21.1 Detailed Description

Special Events format definition and handling functions. This event type encodes special occurrences, such as timestamp related notifications or external input events.

### 4.21.2 Macro Definition Documentation

#### 4.21.2.1 CAER\_SPECIAL\_CONST\_ITERATOR\_ALL\_START

```
#define CAER_SPECIAL_CONST_ITERATOR_ALL_START(
    SPECIAL_PACKET )
```

##### Value:

```
for (int32_t caerSpecialIteratorCounter = 0; \
    caerSpecialIteratorCounter < caerEventPacketHeaderGetEventNumber
    (&(SPECIAL_PACKET)->packetHeader); \
    caerSpecialIteratorCounter++) { \
    caerSpecialEventConst caerSpecialIteratorElement =
    caerSpecialEventPacketGetEventConst (SPECIAL_PACKET,
    caerSpecialIteratorCounter);
```

Const-Iterator over all special events in a packet. Returns the current index in the 'caerSpecialIteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerSpecialIteratorElement' variable of type caerSpecialEventConst.

SPECIAL\_PACKET: a valid SpecialEventPacket pointer. Cannot be NULL.

#### 4.21.2.2 CAER\_SPECIAL\_CONST\_ITERATOR\_VALID\_START

```
#define CAER_SPECIAL_CONST_ITERATOR_VALID_START(
    SPECIAL_PACKET )
```

##### Value:

```
for (int32_t caerSpecialIteratorCounter = 0; \
    caerSpecialIteratorCounter < caerEventPacketHeaderGetEventNumber
    (&(SPECIAL_PACKET)->packetHeader); \
    caerSpecialIteratorCounter++) { \
    caerSpecialEventConst caerSpecialIteratorElement =
    caerSpecialEventPacketGetEventConst (SPECIAL_PACKET,
    caerSpecialIteratorCounter); \
    if (!caerSpecialEventIsValid(caerSpecialIteratorElement)) { continue; }
```

Const-Iterator over only the valid special events in a packet. Returns the current index in the 'caerSpecialIteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerSpecialIteratorElement' variable of type caerSpecialEventConst.

SPECIAL\_PACKET: a valid SpecialEventPacket pointer. Cannot be NULL.

#### 4.21.2.3 CAER\_SPECIAL\_CONST\_REVERSE\_ITERATOR\_ALL\_START

```
#define CAER_SPECIAL_CONST_REVERSE_ITERATOR_ALL_START(
    SPECIAL_PACKET )
```

##### Value:

```
for (int32_t caerSpecialIteratorCounter = caerEventPacketHeaderGetEventNumber
    (&(SPECIAL_PACKET)->packetHeader) - 1; \
    caerSpecialIteratorCounter >= 0; \
    caerSpecialIteratorCounter--) { \
    caerSpecialEventConst caerSpecialIteratorElement =
    caerSpecialEventPacketGetEventConst (SPECIAL_PACKET,
    caerSpecialIteratorCounter);
```

Const-Reverse iterator over all special events in a packet. Returns the current index in the 'caerSpecialIteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerSpecialIteratorElement' variable of type caerSpecialEventConst.

SPECIAL\_PACKET: a valid SpecialEventPacket pointer. Cannot be NULL.

## 4.21.2.4 CAER\_SPECIAL\_CONST\_REVERSE\_ITERATOR\_VALID\_START

```
#define CAER_SPECIAL_CONST_REVERSE_ITERATOR_VALID_START(
    SPECIAL_PACKET )
```

**Value:**

```
for (int32_t caerSpecialIteratorCounter = caerEventPacketHeaderGetEventNumber
    (&(SPECIAL_PACKET)->packetHeader) - 1; \
    caerSpecialIteratorCounter >= 0; \
    caerSpecialIteratorCounter--) { \
    caerSpecialEventConst caerSpecialIteratorElement =
    caerSpecialEventPacketGetEventConst(SPECIAL_PACKET,
    caerSpecialIteratorCounter); \
    if (!caerSpecialEventIsValid(caerSpecialIteratorElement)) { continue; }
```

Const-Reverse iterator over only the valid special events in a packet. Returns the current index in the 'caerSpecialIteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerSpecialIteratorElement' variable of type caerSpecialEventConst.

SPECIAL\_PACKET: a valid SpecialEventPacket pointer. Cannot be NULL.

## 4.21.2.5 CAER\_SPECIAL\_ITERATOR\_ALL\_END

```
#define CAER_SPECIAL_ITERATOR_ALL_END }
```

Iterator close statement.

## 4.21.2.6 CAER\_SPECIAL\_ITERATOR\_ALL\_START

```
#define CAER_SPECIAL_ITERATOR_ALL_START(
    SPECIAL_PACKET )
```

**Value:**

```
for (int32_t caerSpecialIteratorCounter = 0; \
    caerSpecialIteratorCounter < caerEventPacketHeaderGetEventNumber
    (&(SPECIAL_PACKET)->packetHeader); \
    caerSpecialIteratorCounter++) { \
    caerSpecialEvent caerSpecialIteratorElement =
    caerSpecialEventPacketGetEvent(SPECIAL_PACKET, caerSpecialIteratorCounter);
```

Iterator over all special events in a packet. Returns the current index in the 'caerSpecialIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerSpecialIteratorElement' variable of type caerSpecialEvent.

SPECIAL\_PACKET: a valid SpecialEventPacket pointer. Cannot be NULL.

## 4.21.2.7 CAER\_SPECIAL\_ITERATOR\_VALID\_END

```
#define CAER_SPECIAL_ITERATOR_VALID_END }
```

Iterator close statement.

#### 4.21.2.8 CAER\_SPECIAL\_ITERATOR\_VALID\_START

```
#define CAER_SPECIAL_ITERATOR_VALID_START(  
    SPECIAL_PACKET )
```

**Value:**

```
for (int32_t caerSpecialIteratorCounter = 0; \  
    caerSpecialIteratorCounter < caerEventPacketHeaderGetEventNumber  
    (&(SPECIAL_PACKET)->packetHeader); \  
    caerSpecialIteratorCounter++) { \  
    caerSpecialEvent caerSpecialIteratorElement =  
    caerSpecialEventPacketGetEvent(SPECIAL_PACKET, caerSpecialIteratorCounter); \  
    if (!caerSpecialEventIsValid(caerSpecialIteratorElement)) { continue; }
```

Iterator over only the valid special events in a packet. Returns the current index in the 'caerSpecialIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerSpecialIteratorElement' variable of type caerSpecialEvent.

SPECIAL\_PACKET: a valid SpecialEventPacket pointer. Cannot be NULL.

#### 4.21.2.9 CAER\_SPECIAL\_REVERSE\_ITERATOR\_ALL\_END

```
#define CAER_SPECIAL_REVERSE_ITERATOR_ALL_END }
```

Reverse iterator close statement.

#### 4.21.2.10 CAER\_SPECIAL\_REVERSE\_ITERATOR\_ALL\_START

```
#define CAER_SPECIAL_REVERSE_ITERATOR_ALL_START(  
    SPECIAL_PACKET )
```

**Value:**

```
for (int32_t caerSpecialIteratorCounter = caerEventPacketHeaderGetEventNumber  
    (&(SPECIAL_PACKET)->packetHeader) - 1; \  
    caerSpecialIteratorCounter >= 0; \  
    caerSpecialIteratorCounter--) { \  
    caerSpecialEvent caerSpecialIteratorElement =  
    caerSpecialEventPacketGetEvent(SPECIAL_PACKET, caerSpecialIteratorCounter);
```

Reverse iterator over all special events in a packet. Returns the current index in the 'caerSpecialIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerSpecialIteratorElement' variable of type caerSpecialEvent.

SPECIAL\_PACKET: a valid SpecialEventPacket pointer. Cannot be NULL.

#### 4.21.2.11 CAER\_SPECIAL\_REVERSE\_ITERATOR\_VALID\_END

```
#define CAER_SPECIAL_REVERSE_ITERATOR_VALID_END }
```

Reverse iterator close statement.

## 4.21.2.12 CAER\_SPECIAL\_REVERSE\_ITERATOR\_VALID\_START

```
#define CAER_SPECIAL_REVERSE_ITERATOR_VALID_START(
    SPECIAL_PACKET )
```

**Value:**

```
for (int32_t caerSpecialIteratorCounter = caerEventPacketHeaderGetEventNumber
    (&(SPECIAL_PACKET)->packetHeader) - 1; \
    caerSpecialIteratorCounter >= 0; \
    caerSpecialIteratorCounter--) { \
    caerSpecialEvent caerSpecialIteratorElement =
    caerSpecialEventPacketGetEvent(SPECIAL_PACKET, caerSpecialIteratorCounter); \
    if (!caerSpecialEventIsValid(caerSpecialIteratorElement)) { continue; }
```

Reverse iterator over only the valid special events in a packet. Returns the current index in the 'caerSpecialIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerSpecialIteratorElement' variable of type caerSpecialEvent.

SPECIAL\_PACKET: a valid SpecialEventPacket pointer. Cannot be NULL.

## 4.21.2.13 SPECIAL\_DATA\_MASK

```
#define SPECIAL_DATA_MASK 0x00FFFFFF
```

Shift and mask values for the type and data portions of a special event. Up to 128 types, with 24 bits of data each, are possible. Bit 0 is the valid mark, see 'common.h' for more details.

## 4.21.2.14 SPECIAL\_DATA\_SHIFT

```
#define SPECIAL_DATA_SHIFT 8
```

Shift and mask values for the type and data portions of a special event. Up to 128 types, with 24 bits of data each, are possible. Bit 0 is the valid mark, see 'common.h' for more details.

## 4.21.2.15 SPECIAL\_TYPE\_MASK

```
#define SPECIAL_TYPE_MASK 0x0000007F
```

Shift and mask values for the type and data portions of a special event. Up to 128 types, with 24 bits of data each, are possible. Bit 0 is the valid mark, see 'common.h' for more details.

## 4.21.2.16 SPECIAL\_TYPE\_SHIFT

```
#define SPECIAL_TYPE_SHIFT 1
```

Shift and mask values for the type and data portions of a special event. Up to 128 types, with 24 bits of data each, are possible. Bit 0 is the valid mark, see 'common.h' for more details.

### 4.21.3 Typedef Documentation

#### 4.21.3.1 caerSpecialEvent

```
typedef struct caer_special_event* caerSpecialEvent
```

Type for pointer to special event data structure.

#### 4.21.3.2 caerSpecialEventPacket

```
typedef struct caer_special_event_packet* caerSpecialEventPacket
```

Type for pointer to special event packet data structure.

### 4.21.4 Enumeration Type Documentation

#### 4.21.4.1 caer\_special\_event\_types

```
enum caer_special_event_types
```

List of all special event type identifiers. Used to interpret the special event type field.

##### Enumerator

TIMESTAMP_WRAP	A 32 bit timestamp wrap occurred.
TIMESTAMP_RESET	A timestamp reset occurred.
EXTERNAL_INPUT_RISING_EDGE	A rising edge was detected (External Input module on device).
EXTERNAL_INPUT_FALLING_EDGE	A falling edge was detected (External Input module on device).
EXTERNAL_INPUT_PULSE	A pulse was detected (External Input module on device).
DVS_ROW_ONLY	A DVS row-only event was detected (a row address without any following column addresses).
EXTERNAL_INPUT1_RISING_EDGE	A rising edge was detected (External Input 1 module on device).
EXTERNAL_INPUT1_FALLING_EDGE	A falling edge was detected (External Input 1 module on device).
EXTERNAL_INPUT1_PULSE	A pulse was detected (External Input 1 module on device).
EXTERNAL_INPUT2_RISING_EDGE	A rising edge was detected (External Input 2 module on device).
EXTERNAL_INPUT2_FALLING_EDGE	A falling edge was detected (External Input 2 module on device).
EXTERNAL_INPUT2_PULSE	A pulse was detected (External Input 2 module on device).



## Enumerator

EXTERNAL_GENERATOR_RISING_EDGE	A rising edge was generated (External Input Generator module on device).
EXTERNAL_GENERATOR_FALLING_EDGE	A falling edge was generated (External Input Generator module on device).
APS_FRAME_START	An APS frame capture has started (Frame Event will follow).
APS_FRAME_END	An APS frame capture has completed (Frame Event is alongside).
APS_EXPOSURE_START	An APS frame exposure has started (Frame Event will follow).
APS_EXPOSURE_END	An APS frame exposure has completed (Frame Event will follow).

## 4.21.5 Function Documentation

## 4.21.5.1 caerSpecialEventGetData()

```
static uint32_t caerSpecialEventGetData (
    caerSpecialEventConst event ) [inline], [static]
```

Get the special event data. Its meaning depends on the type. Current types that make use of it are (see 'enum caer\_special\_event\_types'):

- DVS\_ROW\_ONLY: encodes the address of the row from the row-only event.

## Parameters

<i>event</i>	a valid SpecialEvent pointer. Cannot be NULL.
--------------	---

## Returns

the special event data.

## 4.21.5.2 caerSpecialEventGetTimestamp()

```
static int32_t caerSpecialEventGetTimestamp (
    caerSpecialEventConst event ) [inline], [static]
```

Get the 32bit event timestamp, in microseconds. Be aware that this wraps around! You can either ignore this fact, or handle the special 'TIMESTAMP\_WRAP' event that is generated when this happens, or use the 64bit timestamp which never wraps around. See '[caerEventPacketHeaderGetEventTSOverflow\(\)](#)' documentation for more details on the 64bit timestamp.

**Parameters**

<i>event</i>	a valid SpecialEvent pointer. Cannot be NULL.
--------------	---

**Returns**

this event's 32bit microsecond timestamp.

**4.21.5.3 caerSpecialEventGetTimestamp64()**

```
static int64_t caerSpecialEventGetTimestamp64 (  
    caerSpecialEventConst event,  
    caerSpecialEventPacketConst packet ) [inline], [static]
```

Get the 64bit event timestamp, in microseconds. See '[caerEventPacketHeaderGetEventTSOverflow\(\)](#)' documentation for more details on the 64bit timestamp.

**Parameters**

<i>event</i>	a valid SpecialEvent pointer. Cannot be NULL.
<i>packet</i>	the SpecialEventPacket pointer for the packet containing this event. Cannot be NULL.

**Returns**

this event's 64bit microsecond timestamp.

**4.21.5.4 caerSpecialEventGetType()**

```
static uint8_t caerSpecialEventGetType (  
    caerSpecialEventConst event ) [inline], [static]
```

Get the numerical special event type.

**Parameters**

<i>event</i>	a valid SpecialEvent pointer. Cannot be NULL.
--------------	---

**Returns**

the special event type (see 'enum caer\_special\_event\_types').

## 4.21.5.5 caerSpecialEventInvalidate()

```
static void caerSpecialEventInvalidate (
    caerSpecialEvent event,
    caerSpecialEventPacket packet ) [inline], [static]
```

Invalidate the current event by setting its valid bit to false and decreasing the number of valid events held in the packet. Only works with events that are already valid!

## Parameters

<i>event</i>	a valid SpecialEvent pointer. Cannot be NULL.
<i>packet</i>	the SpecialEventPacket pointer for the packet containing this event. Cannot be NULL.

## 4.21.5.6 caerSpecialEventIsValid()

```
static bool caerSpecialEventIsValid (
    caerSpecialEventConst event ) [inline], [static]
```

Check if this special event is valid.

## Parameters

<i>event</i>	a valid SpecialEvent pointer. Cannot be NULL.
--------------	---

## Returns

true if valid, false if not.

## 4.21.5.7 caerSpecialEventPacketAllocate()

```
caerSpecialEventPacket caerSpecialEventPacketAllocate (
    int32_t eventCapacity,
    int16_t eventSource,
    int32_t tsOverflow )
```

Allocate a new special events packet. Use free() to reclaim this memory.

## Parameters

<i>eventCapacity</i>	the maximum number of events this packet will hold.
<i>eventSource</i>	the unique ID representing the source/generator of this packet.
<i>tsOverflow</i>	the current timestamp overflow counter value for this packet.

**Returns**

a valid `SpecialEventPacket` handle or `NULL` on error.

**4.21.5.8 caerSpecialEventPacketFindEventByType()**

```
static caerSpecialEvent caerSpecialEventPacketFindEventByType (
    caerSpecialEventPacket packet,
    uint8_t type ) [inline], [static]
```

Get the first special event with the given event type in this event packet. This returns the first found event with that type ID, or `NULL` if we get to the end without finding any such event.

**Parameters**

<i>packet</i>	a valid <code>SpecialEventPacket</code> pointer. Cannot be <code>NULL</code> .
<i>type</i>	the special event type to search for.

**Returns**

the requested special event or `NULL` on error/not found.

**4.21.5.9 caerSpecialEventPacketFindEventByTypeConst()**

```
static caerSpecialEventConst caerSpecialEventPacketFindEventByTypeConst (
    caerSpecialEventPacketConst packet,
    uint8_t type ) [inline], [static]
```

Get the first special event with the given event type in this event packet. This returns the first found event with that type ID, or `NULL` if we get to the end without finding any such event. The returned event is read-only!

**Parameters**

<i>packet</i>	a valid <code>SpecialEventPacket</code> pointer. Cannot be <code>NULL</code> .
<i>type</i>	the special event type to search for.

**Returns**

the requested read-only special event or `NULL` on error/not found.

**4.21.5.10 caerSpecialEventPacketFindValidEventByType()**

```
static caerSpecialEvent caerSpecialEventPacketFindValidEventByType (
    caerSpecialEventPacket packet,
    uint8_t type ) [inline], [static]
```

Get the first valid special event with the given event type in this event packet. This returns the first found valid event with that type ID, or NULL if we get to the end without finding any such event.

#### Parameters

<i>packet</i>	a valid SpecialEventPacket pointer. Cannot be NULL.
<i>type</i>	the special event type to search for.

#### Returns

the requested valid special event or NULL on error/not found.

#### 4.21.5.11 caerSpecialEventPacketFindValidEventByTypeConst()

```
static caerSpecialEventConst caerSpecialEventPacketFindValidEventByTypeConst (
    caerSpecialEventPacketConst packet,
    uint8_t type ) [inline], [static]
```

Get the first valid special event with the given event type in this event packet. This returns the first found valid event with that type ID, or NULL if we get to the end without finding any such event. The returned event is read-only!

#### Parameters

<i>packet</i>	a valid SpecialEventPacket pointer. Cannot be NULL.
<i>type</i>	the special event type to search for.

#### Returns

the requested read-only valid special event or NULL on error/not found.

#### 4.21.5.12 caerSpecialEventPacketGetEvent()

```
static caerSpecialEvent caerSpecialEventPacketGetEvent (
    caerSpecialEventPacket packet,
    int32_t n ) [inline], [static]
```

Get the special event at the given index from the event packet.

#### Parameters

<i>packet</i>	a valid SpecialEventPacket pointer. Cannot be NULL.
<i>n</i>	the index of the returned event. Must be within [0,eventCapacity[ bounds.

**Returns**

the requested special event. NULL on error.

**4.21.5.13 caerSpecialEventPacketGetEventConst()**

```
static caerSpecialEventConst caerSpecialEventPacketGetEventConst (
    caerSpecialEventPacketConst packet,
    int32_t n ) [inline], [static]
```

Get the special event at the given index from the event packet. This is a read-only event, do not change its contents in any way!

**Parameters**

<i>packet</i>	a valid SpecialEventPacket pointer. Cannot be NULL.
<i>n</i>	the index of the returned event. Must be within [0,eventCapacity[ bounds.

**Returns**

the requested read-only special event. NULL on error.

**4.21.5.14 caerSpecialEventSetData()**

```
static void caerSpecialEventSetData (
    caerSpecialEvent event,
    uint32_t data ) [inline], [static]
```

Set the special event data. Its meaning depends on the type. Current types that make use of it are (see 'enum caer\_special\_event\_types'):

- DVS\_ROW\_ONLY: encodes the address of the row from the row-only event.

**Parameters**

<i>event</i>	a valid SpecialEvent pointer. Cannot be NULL.
<i>data</i>	the special event data.

**4.21.5.15 caerSpecialEventSetTimestamp()**

```
static void caerSpecialEventSetTimestamp (
    caerSpecialEvent event,
    int32_t timestamp ) [inline], [static]
```

Set the 32bit event timestamp, the value has to be in microseconds.

#### Parameters

<i>event</i>	a valid SpecialEvent pointer. Cannot be NULL.
<i>timestamp</i>	a positive 32bit microsecond timestamp.

#### 4.21.5.16 caerSpecialEventSetType()

```
static void caerSpecialEventSetType (
    caerSpecialEvent event,
    uint8_t type ) [inline], [static]
```

Set the numerical special event type.

#### Parameters

<i>event</i>	a valid SpecialEvent pointer. Cannot be NULL.
<i>type</i>	the special event type (see 'enum caer_special_event_types').

#### 4.21.5.17 caerSpecialEventValidate()

```
static void caerSpecialEventValidate (
    caerSpecialEvent event,
    caerSpecialEventPacket packet ) [inline], [static]
```

Validate the current event by setting its valid bit to true and increasing the event packet's event count and valid event count. Only works on events that are invalid. DO NOT CALL THIS AFTER HAVING PREVIOUSLY ALREADY INVALIDATED THIS EVENT, the total count will be incorrect.

#### Parameters

<i>event</i>	a valid SpecialEvent pointer. Cannot be NULL.
<i>packet</i>	the SpecialEventPacket pointer for the packet containing this event. Cannot be NULL.

#### 4.21.5.18 PACKED\_STRUCT() [1/2]

```
PACKED_STRUCT (
    struct caer_special_event { uint32_t data;int32_t timestamp;} )
```

Special event data structure definition. This contains the actual data, as well as the 32 bit event timestamp. Signed integers are used for fields that are to be interpreted directly, for compatibility with languages that do not have unsigned integer types, such as Java.

#### 4.21.5.19 PACKED\_STRUCT() [2/2]

```
PACKED_STRUCT (
    struct caer_special_event_packet { struct caer_event_packet_header packetHeader; struct
    caer_special_event events[]; } )
```

Special event packet data structure definition. EventPackets are always made up of the common packet header, followed by 'eventCapacity' events. Everything has to be in one contiguous memory block.

## 4.22 events/spike.h File Reference

```
#include "common.h"
```

### Macros

- `#define CAER_SPIKE_ITERATOR_ALL_START(SPIKE_PACKET)`
- `#define CAER_SPIKE_CONST_ITERATOR_ALL_START(SPIKE_PACKET)`
- `#define CAER_SPIKE_ITERATOR_ALL_END }`
- `#define CAER_SPIKE_ITERATOR_VALID_START(SPIKE_PACKET)`
- `#define CAER_SPIKE_CONST_ITERATOR_VALID_START(SPIKE_PACKET)`
- `#define CAER_SPIKE_ITERATOR_VALID_END }`
- `#define CAER_SPIKE_REVERSE_ITERATOR_ALL_START(SPIKE_PACKET)`
- `#define CAER_SPIKE_CONST_REVERSE_ITERATOR_ALL_START(SPIKE_PACKET)`
- `#define CAER_SPIKE_REVERSE_ITERATOR_ALL_END }`
- `#define CAER_SPIKE_REVERSE_ITERATOR_VALID_START(SPIKE_PACKET)`
- `#define CAER_SPIKE_CONST_REVERSE_ITERATOR_VALID_START(SPIKE_PACKET)`
- `#define CAER_SPIKE_REVERSE_ITERATOR_VALID_END }`

- `#define SPIKE_SOURCE_CORE_ID_SHIFT 1`
- `#define SPIKE_SOURCE_CORE_ID_MASK 0x0000001F`
- `#define SPIKE_CHIP_ID_SHIFT 6`
- `#define SPIKE_CHIP_ID_MASK 0x0000003F`
- `#define SPIKE_NEURON_ID_SHIFT 12`
- `#define SPIKE_NEURON_ID_MASK 0x000FFFFF`

### Typedefs

- `typedef struct caer_spike_event * caerSpikeEvent`
- `typedef const struct caer_spike_event * caerSpikeEventConst`
- `typedef struct caer_spike_event_packet * caerSpikeEventPacket`
- `typedef const struct caer_spike_event_packet * caerSpikeEventPacketConst`



## Functions

- [PACKED\\_STRUCT](#) (struct caer\_spike\_event { uint32\_t data;int32\_t timestamp;})
- [PACKED\\_STRUCT](#) (struct caer\_spike\_event\_packet { struct caer\_event\_packet\_header packetHeader;struct caer\_spike\_event events[ ];})
- [caerSpikeEventPacket](#) [caerSpikeEventPacketAllocate](#) (int32\_t eventCapacity, int16\_t eventSource, int32\_t tsOverflow)
- static [caerSpikeEvent](#) [caerSpikeEventPacketGetEvent](#) ([caerSpikeEventPacket](#) packet, int32\_t n)
- static caerSpikeEventConst [caerSpikeEventPacketGetEventConst](#) ([caerSpikeEventPacketConst](#) packet, int32\_t n)
- static int32\_t [caerSpikeEventGetTimestamp](#) ([caerSpikeEventConst](#) event)
- static int64\_t [caerSpikeEventGetTimestamp64](#) ([caerSpikeEventConst](#) event, [caerSpikeEventPacketConst](#) packet)
- static void [caerSpikeEventSetTimestamp](#) ([caerSpikeEvent](#) event, int32\_t timestamp)
- static bool [caerSpikeEventsIsValid](#) ([caerSpikeEventConst](#) event)
- static void [caerSpikeEventValidate](#) ([caerSpikeEvent](#) event, [caerSpikeEventPacket](#) packet)
- static void [caerSpikeEventInvalidate](#) ([caerSpikeEvent](#) event, [caerSpikeEventPacket](#) packet)
- static uint8\_t [caerSpikeEventGetSourceCoreID](#) ([caerSpikeEventConst](#) event)
- static void [caerSpikeEventSetSourceCoreID](#) ([caerSpikeEvent](#) event, uint8\_t sourceCoreID)
- static uint8\_t [caerSpikeEventGetChipID](#) ([caerSpikeEventConst](#) event)
- static void [caerSpikeEventSetChipID](#) ([caerSpikeEvent](#) event, uint8\_t chipID)
- static uint32\_t [caerSpikeEventGetNeuronID](#) ([caerSpikeEventConst](#) event)
- static void [caerSpikeEventSetNeuronID](#) ([caerSpikeEvent](#) event, uint32\_t neuronID)

### 4.22.1 Detailed Description

THIS EVENT DEFINITIONS IS STILL TO BE CONSIDERED EXPERIMENTAL AND IS SUBJECT TO FUTURE CHANGES AND REVISIONS!

Spike Events format definition and handling functions. This contains spikes generated by a neuron-array chip.

### 4.22.2 Macro Definition Documentation

#### 4.22.2.1 CAER\_SPIKE\_CONST\_ITERATOR\_ALL\_START

```
#define CAER_SPIKE_CONST_ITERATOR_ALL_START(  
    SPIKE_PACKET )
```

**Value:**

```
for (int32_t caerSpikeIteratorCounter = 0; \  
    caerSpikeIteratorCounter < caerEventPacketHeaderGetEventNumber (&  
    (SPIKE_PACKET)->packetHeader); \  
    caerSpikeIteratorCounter++) { \  
    caerSpikeEventConst caerSpikeIteratorElement =  
    caerSpikeEventPacketGetEventConst (SPIKE_PACKET, caerSpikeIteratorCounter);
```

Const-Iterator over all Spike events in a packet. Returns the current index in the 'caerSpikeIteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerSpikeIteratorElement' variable of type caerSpikeEventConst.

SPIKE\_PACKET: a valid SpikeEventPacket pointer. Cannot be NULL.

#### 4.22.2.2 CAER\_SPIKE\_CONST\_ITERATOR\_VALID\_START

```
#define CAER_SPIKE_CONST_ITERATOR_VALID_START(  
    SPIKE_PACKET )
```

##### Value:

```
for (int32_t caerSpikeIteratorCounter = 0; \  
    caerSpikeIteratorCounter < caerEventPacketHeaderGetEventNumber(&  
    (SPIKE_PACKET)->packetHeader); \  
    caerSpikeIteratorCounter++) { \  
    caerSpikeEventConst caerSpikeIteratorElement =  
    caerSpikeEventPacketGetEventConst(SPIKE_PACKET, caerSpikeIteratorCounter);  
    if (!caerSpikeEventIsValid(caerSpikeIteratorElement)) { continue; }
```

Const-Iterator over only the valid Spike events in a packet. Returns the current index in the 'caerSpikeIteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerSpikeIteratorElement' variable of type caerSpikeEventConst.

SPIKE\_PACKET: a valid SpikeEventPacket pointer. Cannot be NULL.

#### 4.22.2.3 CAER\_SPIKE\_CONST\_REVERSE\_ITERATOR\_ALL\_START

```
#define CAER_SPIKE_CONST_REVERSE_ITERATOR_ALL_START(  
    SPIKE_PACKET )
```

##### Value:

```
for (int32_t caerSpikeIteratorCounter = caerEventPacketHeaderGetEventNumber  
    (&(SPIKE_PACKET)->packetHeader) - 1; \  
    caerSpikeIteratorCounter >= 0; \  
    caerSpikeIteratorCounter--) { \  
    caerSpikeEventConst caerSpikeIteratorElement =  
    caerSpikeEventPacketGetEventConst(SPIKE_PACKET, caerSpikeIteratorCounter);
```

Const-Reverse iterator over all spike events in a packet. Returns the current index in the 'caerSpikeIteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerSpikeIteratorElement' variable of type caerSpikeEventConst.

SPIKE\_PACKET: a valid SpikeEventPacket pointer. Cannot be NULL.

#### 4.22.2.4 CAER\_SPIKE\_CONST\_REVERSE\_ITERATOR\_VALID\_START

```
#define CAER_SPIKE_CONST_REVERSE_ITERATOR_VALID_START(  
    SPIKE_PACKET )
```

##### Value:

```
for (int32_t caerSpikeIteratorCounter = caerEventPacketHeaderGetEventNumber  
    (&(SPIKE_PACKET)->packetHeader) - 1; \  
    caerSpikeIteratorCounter >= 0; \  
    caerSpikeIteratorCounter--) { \  
    caerSpikeEventConst caerSpikeIteratorElement =  
    caerSpikeEventPacketGetEventConst(SPIKE_PACKET, caerSpikeIteratorCounter);  
    if (!caerSpikeEventIsValid(caerSpikeIteratorElement)) { continue; }
```

Const-Reverse iterator over only the valid spike events in a packet. Returns the current index in the 'caerSpikeIteratorCounter' variable of type 'int32\_t' and the current read-only event in the 'caerSpikeIteratorElement' variable of type caerSpikeEventConst.

SPIKE\_PACKET: a valid SpikeEventPacket pointer. Cannot be NULL.

## 4.22.2.5 CAER\_SPIKE\_ITERATOR\_ALL\_END

```
#define CAER_SPIKE_ITERATOR_ALL_END }
```

Iterator close statement.

## 4.22.2.6 CAER\_SPIKE\_ITERATOR\_ALL\_START

```
#define CAER_SPIKE_ITERATOR_ALL_START (
    SPIKE_PACKET )
```

**Value:**

```
for (int32_t caerSpikeIteratorCounter = 0; \
    caerSpikeIteratorCounter < caerEventPacketHeaderGetEventNumber (&
    (SPIKE_PACKET)->packetHeader); \
    caerSpikeIteratorCounter++) { \
    caerSpikeEvent caerSpikeIteratorElement = caerSpikeEventPacketGetEvent (
    SPIKE_PACKET, caerSpikeIteratorCounter);
```

Iterator over all Spike events in a packet. Returns the current index in the 'caerSpikeIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerSpikeIteratorElement' variable of type caerSpikeEvent.

SPIKE\_PACKET: a valid SpikeEventPacket pointer. Cannot be NULL.

## 4.22.2.7 CAER\_SPIKE\_ITERATOR\_VALID\_END

```
#define CAER_SPIKE_ITERATOR_VALID_END }
```

Iterator close statement.

## 4.22.2.8 CAER\_SPIKE\_ITERATOR\_VALID\_START

```
#define CAER_SPIKE_ITERATOR_VALID_START (
    SPIKE_PACKET )
```

**Value:**

```
for (int32_t caerSpikeIteratorCounter = 0; \
    caerSpikeIteratorCounter < caerEventPacketHeaderGetEventNumber (&
    (SPIKE_PACKET)->packetHeader); \
    caerSpikeIteratorCounter++) { \
    caerSpikeEvent caerSpikeIteratorElement = caerSpikeEventPacketGetEvent (
    SPIKE_PACKET, caerSpikeIteratorCounter); \
    if (!caerSpikeEventIsValid(caerSpikeIteratorElement)) { continue; }
```

Iterator over only the valid Spike events in a packet. Returns the current index in the 'caerSpikeIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerSpikeIteratorElement' variable of type caerSpikeEvent.

SPIKE\_PACKET: a valid SpikeEventPacket pointer. Cannot be NULL.

#### 4.22.2.9 CAER\_SPIKE\_REVERSE\_ITERATOR\_ALL\_END

```
#define CAER_SPIKE_REVERSE_ITERATOR_ALL_END }
```

Reverse iterator close statement.

#### 4.22.2.10 CAER\_SPIKE\_REVERSE\_ITERATOR\_ALL\_START

```
#define CAER_SPIKE_REVERSE_ITERATOR_ALL_START(  
    SPIKE_PACKET )
```

**Value:**

```
for (int32_t caerSpikeIteratorCounter = caerEventPacketHeaderGetEventNumber  
    (&(SPIKE_PACKET)->packetHeader) - 1; \  
    caerSpikeIteratorCounter >= 0; \  
    caerSpikeIteratorCounter--) { \  
    caerSpikeEvent caerSpikeIteratorElement = caerSpikeEventPacketGetEvent(  
        SPIKE_PACKET, caerSpikeIteratorCounter);
```

Reverse iterator over all spike events in a packet. Returns the current index in the 'caerSpikeIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerSpikeIteratorElement' variable of type caerSpikeEvent.

SPIKE\_PACKET: a valid SpikeEventPacket pointer. Cannot be NULL.

#### 4.22.2.11 CAER\_SPIKE\_REVERSE\_ITERATOR\_VALID\_END

```
#define CAER_SPIKE_REVERSE_ITERATOR_VALID_END }
```

Reverse iterator close statement.

#### 4.22.2.12 CAER\_SPIKE\_REVERSE\_ITERATOR\_VALID\_START

```
#define CAER_SPIKE_REVERSE_ITERATOR_VALID_START(  
    SPIKE_PACKET )
```

**Value:**

```
for (int32_t caerSpikeIteratorCounter = caerEventPacketHeaderGetEventNumber  
    (&(SPIKE_PACKET)->packetHeader) - 1; \  
    caerSpikeIteratorCounter >= 0; \  
    caerSpikeIteratorCounter--) { \  
    caerSpikeEvent caerSpikeIteratorElement = caerSpikeEventPacketGetEvent(  
        SPIKE_PACKET, caerSpikeIteratorCounter); \  
    if (!caerSpikeEventIsValid(caerSpikeIteratorElement)) { continue; }
```

Reverse iterator over only the valid spike events in a packet. Returns the current index in the 'caerSpikeIteratorCounter' variable of type 'int32\_t' and the current event in the 'caerSpikeIteratorElement' variable of type caerSpikeEvent.

SPIKE\_PACKET: a valid SpikeEventPacket pointer. Cannot be NULL.

#### 4.22.2.13 SPIKE\_CHIP\_ID\_MASK

```
#define SPIKE_CHIP_ID_MASK 0x0000003F
```

Shift and mask values for spike information associated with a Spike event. 32 core IDs, 64 chip IDs and up to a million neuron IDs are supported. Bit 0 is the valid mark, see '[common.h](#)' for more details.

#### 4.22.2.14 SPIKE\_CHIP\_ID\_SHIFT

```
#define SPIKE_CHIP_ID_SHIFT 6
```

Shift and mask values for spike information associated with a Spike event. 32 core IDs, 64 chip IDs and up to a million neuron IDs are supported. Bit 0 is the valid mark, see '[common.h](#)' for more details.

#### 4.22.2.15 SPIKE\_NEURON\_ID\_MASK

```
#define SPIKE_NEURON_ID_MASK 0x000FFFFF
```

Shift and mask values for spike information associated with a Spike event. 32 core IDs, 64 chip IDs and up to a million neuron IDs are supported. Bit 0 is the valid mark, see '[common.h](#)' for more details.

#### 4.22.2.16 SPIKE\_NEURON\_ID\_SHIFT

```
#define SPIKE_NEURON_ID_SHIFT 12
```

Shift and mask values for spike information associated with a Spike event. 32 core IDs, 64 chip IDs and up to a million neuron IDs are supported. Bit 0 is the valid mark, see '[common.h](#)' for more details.

#### 4.22.2.17 SPIKE\_SOURCE\_CORE\_ID\_MASK

```
#define SPIKE_SOURCE_CORE_ID_MASK 0x0000001F
```

Shift and mask values for spike information associated with a Spike event. 32 core IDs, 64 chip IDs and up to a million neuron IDs are supported. Bit 0 is the valid mark, see '[common.h](#)' for more details.

#### 4.22.2.18 SPIKE\_SOURCE\_CORE\_ID\_SHIFT

```
#define SPIKE_SOURCE_CORE_ID_SHIFT 1
```

Shift and mask values for spike information associated with a Spike event. 32 core IDs, 64 chip IDs and up to a million neuron IDs are supported. Bit 0 is the valid mark, see '[common.h](#)' for more details.

### 4.22.3 Typedef Documentation

#### 4.22.3.1 caerSpikeEvent

```
typedef struct caer_spike_event* caerSpikeEvent
```

Type for pointer to Spike event data structure.

#### 4.22.3.2 caerSpikeEventPacket

```
typedef struct caer_spike_event_packet* caerSpikeEventPacket
```

Type for pointer to Spike event packet data structure.

### 4.22.4 Function Documentation

#### 4.22.4.1 caerSpikeEventGetChipID()

```
static uint8_t caerSpikeEventGetChipID (  
    caerSpikeEventConst event ) [inline], [static]
```

Get the chip ID.

##### Parameters

<i>event</i>	a valid SpikeEvent pointer. Cannot be NULL.
--------------	---

##### Returns

the Spike's chip ID.

#### 4.22.4.2 caerSpikeEventGetNeuronID()

```
static uint32_t caerSpikeEventGetNeuronID (  
    caerSpikeEventConst event ) [inline], [static]
```

Get the neuron ID.

##### Parameters

<i>event</i>	a valid SpikeEvent pointer. Cannot be NULL.
--------------	---

**Returns**

the Spike's neuron ID.

**4.22.4.3 caerSpikeEventGetSourceCoreID()**

```
static uint8_t caerSpikeEventGetSourceCoreID (  
    caerSpikeEventConst event ) [inline], [static]
```

Get the source core ID.

**Parameters**

<i>event</i>	a valid SpikeEvent pointer. Cannot be NULL.
--------------	---

**Returns**

the Spike's source core ID.

**4.22.4.4 caerSpikeEventGetTimestamp()**

```
static int32_t caerSpikeEventGetTimestamp (  
    caerSpikeEventConst event ) [inline], [static]
```

Get the 32bit event timestamp, in microseconds. Be aware that this wraps around! You can either ignore this fact, or handle the special 'TIMESTAMP\_WRAP' event that is generated when this happens, or use the 64bit timestamp which never wraps around. See '[caerEventPacketHeaderGetEventTSOverflow\(\)](#)' documentation for more details on the 64bit timestamp.

**Parameters**

<i>event</i>	a valid SpikeEvent pointer. Cannot be NULL.
--------------	---

**Returns**

this event's 32bit microsecond timestamp.

**4.22.4.5 caerSpikeEventGetTimestamp64()**

```
static int64_t caerSpikeEventGetTimestamp64 (  
    caerSpikeEventConst event,  
    caerSpikeEventPacketConst packet ) [inline], [static]
```

Get the 64bit event timestamp, in microseconds. See '[caerEventPacketHeaderGetEventTSOverflow\(\)](#)' documentation for more details on the 64bit timestamp.

## Parameters

<i>event</i>	a valid SpikeEvent pointer. Cannot be NULL.
<i>packet</i>	the SpikeEventPacket pointer for the packet containing this event. Cannot be NULL.

## Returns

this event's 64bit microsecond timestamp.

## 4.22.4.6 caerSpikeEventInvalidate()

```
static void caerSpikeEventInvalidate (
    caerSpikeEvent event,
    caerSpikeEventPacket packet ) [inline], [static]
```

Invalidate the current event by setting its valid bit to false and decreasing the number of valid events held in the packet. Only works with events that are already valid!

## Parameters

<i>event</i>	a valid SpikeEvent pointer. Cannot be NULL.
<i>packet</i>	the SpikeEventPacket pointer for the packet containing this event. Cannot be NULL.

## 4.22.4.7 caerSpikeEventsIsValid()

```
static bool caerSpikeEventIsValid (
    caerSpikeEventConst event ) [inline], [static]
```

Check if this Spike event is valid.

## Parameters

<i>event</i>	a valid SpikeEvent pointer. Cannot be NULL.
--------------	---

## Returns

true if valid, false if not.

## 4.22.4.8 caerSpikeEventPacketAllocate()

```
caerSpikeEventPacket caerSpikeEventPacketAllocate (
    int32_t eventCapacity,
```



```
int16_t eventSource,
int32_t tsOverflow )
```

Allocate a new Spike events packet. Use free() to reclaim this memory.

#### Parameters

<i>eventCapacity</i>	the maximum number of events this packet will hold.
<i>eventSource</i>	the unique ID representing the source/generator of this packet.
<i>tsOverflow</i>	the current timestamp overflow counter value for this packet.

#### Returns

a valid SpikeEventPacket handle or NULL on error.

#### 4.22.4.9 caerSpikeEventPacketGetEvent()

```
static caerSpikeEvent caerSpikeEventPacketGetEvent (
    caerSpikeEventPacket packet,
    int32_t n ) [inline], [static]
```

Get the Spike event at the given index from the event packet.

#### Parameters

<i>packet</i>	a valid SpikeEventPacket pointer. Cannot be NULL.
<i>n</i>	the index of the returned event. Must be within [0,eventCapacity[ bounds.

#### Returns

the requested Spike event. NULL on error.

#### 4.22.4.10 caerSpikeEventPacketGetEventConst()

```
static caerSpikeEventConst caerSpikeEventPacketGetEventConst (
    caerSpikeEventPacketConst packet,
    int32_t n ) [inline], [static]
```

Get the Spike event at the given index from the event packet. This is a read-only event, do not change its contents in any way!

#### Parameters

<i>packet</i>	a valid SpikeEventPacket pointer. Cannot be NULL.
<i>n</i>	the index of the returned event. Must be within [0,eventCapacity[ bounds.

**Returns**

the requested read-only Spike event. NULL on error.

**4.22.4.11 caerSpikeEventSetChipID()**

```
static void caerSpikeEventSetChipID (
    caerSpikeEvent event,
    uint8_t chipID ) [inline], [static]
```

Set the chip ID.

**Parameters**

<i>event</i>	a valid SpikeEvent pointer. Cannot be NULL.
<i>chipID</i>	the Spike's chip ID.

**4.22.4.12 caerSpikeEventSetNeuronID()**

```
static void caerSpikeEventSetNeuronID (
    caerSpikeEvent event,
    uint32_t neuronID ) [inline], [static]
```

Set the neuron ID.

**Parameters**

<i>event</i>	a valid SpikeEvent pointer. Cannot be NULL.
<i>neuronID</i>	the Spike's neuron ID.

**4.22.4.13 caerSpikeEventSetSourceCoreID()**

```
static void caerSpikeEventSetSourceCoreID (
    caerSpikeEvent event,
    uint8_t sourceCoreID ) [inline], [static]
```

Set the source core ID.

**Parameters**

<i>event</i>	a valid SpikeEvent pointer. Cannot be NULL.
<i>sourceCoreID</i>	the Spike's source core ID.

## 4.22.4.14 caerSpikeEventSetTimestamp()

```
static void caerSpikeEventSetTimestamp (
    caerSpikeEvent event,
    int32_t timestamp ) [inline], [static]
```

Set the 32bit event timestamp, the value has to be in microseconds.

## Parameters

<i>event</i>	a valid SpikeEvent pointer. Cannot be NULL.
<i>timestamp</i>	a positive 32bit microsecond timestamp.

## 4.22.4.15 caerSpikeEventValidate()

```
static void caerSpikeEventValidate (
    caerSpikeEvent event,
    caerSpikeEventPacket packet ) [inline], [static]
```

Validate the current event by setting its valid bit to true and increasing the event packet's event count and valid event count. Only works on events that are invalid. DO NOT CALL THIS AFTER HAVING PREVIOUSLY ALREADY INVALIDATED THIS EVENT, the total count will be incorrect.

## Parameters

<i>event</i>	a valid SpikeEvent pointer. Cannot be NULL.
<i>packet</i>	the SpikeEventPacket pointer for the packet containing this event. Cannot be NULL.

## 4.22.4.16 PACKED\_STRUCT() [1/2]

```
PACKED_STRUCT (
    struct caer_spike_event { uint32_t data;int32_t timestamp;} )
```

Spike event data structure definition. This contains the core ID, the neuron ID and the timestamp of the received spike, together with the usual validity mark. Signed integers are used for fields that are to be interpreted directly, for compatibility with languages that do not have unsigned integer types, such as Java.

## 4.22.4.17 PACKED\_STRUCT() [2/2]

```
PACKED_STRUCT (
    struct caer_spike_event_packet { struct caer_event_packet_header packetHeader;struct
    caer_spike_event events[];} )
```

Spike event packet data structure definition. EventPackets are always made up of the common packet header, followed by 'eventCapacity' events. Everything has to be in one contiguous memory block.

## 4.23 frame\_utils.h File Reference

```
#include "events/frame.h"
```

### Enumerations

- enum **caer\_frame\_utils\_demosaic\_types** { DEMOSAIC\_STANDARD = 0, DEMOSAIC\_OPENCV\_NORMAL = 1, DEMOSAIC\_OPENCV\_EDGE\_AWARE = 2 }
- enum **caer\_frame\_utils\_contrast\_types** { CONTRAST\_STANDARD = 0, CONTRAST\_OPENCV\_NORMALIZATION = 1, CONTRAST\_OPENCV\_HISTOGRAM\_EQUALIZATION = 2, CONTRAST\_OPENCV\_CLAHE = 3 }

### Functions

- [caerFrameEventPacket](#) **caerFrameUtilsDemosaic** (caerFrameEventPacketConst framePacket, enum caer\_frame\_utils\_demosaic\_types demosaicType)
- void **caerFrameUtilsContrast** ([caerFrameEventPacket](#) framePacket, enum caer\_frame\_utils\_contrast\_types contrastType)

#### 4.23.1 Detailed Description

Functions for frame enhancement and demosaicing. Basic variants that don't require any external dependencies, such as OpenCV. Use of the OpenCV variants is recommended for quality and performance, and can optionally be enabled at build-time.

## 4.24 libcaer.h File Reference

```
#include <stddef.h>
#include <stdlib.h>
#include <stdbool.h>
#include <stdio.h>
#include <stdint.h>
#include <inttypes.h>
#include <string.h>
#include <errno.h>
#include "portable_endian.h"
#include "log.h"
```

## Macros

- `#define LIBCAER_VERSION ((2 * 10000) + (4 * 100) + 0)`
  - `#define LIBCAER_NAME_STRING "libcaer"`
  - `#define LIBCAER_VERSION_STRING "2.4.0"`
  - `#define LIBCAER_HAVE_SERIALDEV 1`
  - `#define LIBCAER_HAVE_OPENCV 1`
  - `#define U8T(X) ((uint8_t) (X))`
  - `#define U16T(X) ((uint16_t) (X))`
  - `#define U32T(X) ((uint32_t) (X))`
  - `#define U64T(X) ((uint64_t) (X))`
  - `#define I8T(X) ((int8_t) (X))`
  - `#define I16T(X) ((int16_t) (X))`
  - `#define I32T(X) ((int32_t) (X))`
  - `#define I64T(X) ((int64_t) (X))`
  - `#define MASK_NUMBITS32(X) U32T(U32T(U32T(1) << X) - 1)`
  - `#define MASK_NUMBITS64(X) U64T(U64T(U64T(1) << X) - 1)`
  - `#define SWAP_VAR(type, x, y) { type tmpv; tmpv = (x); (x) = (y); (y) = tmpv; }`
- 
- `#define CLEAR_NUMBITS32(VAR, SHIFT, MASK) (VAR) &= htogle32(~(U32T(U32T(MASK) << (SHIFT))))`
  - `#define CLEAR_NUMBITS16(VAR, SHIFT, MASK) (VAR) &= htogle16(~(U16T(U16T(MASK) << (SHIFT))))`
  - `#define CLEAR_NUMBITS8(VAR, SHIFT, MASK) (VAR) &= U8T(~(U8T(U8T(MASK) << (SHIFT))))`
- 
- `#define SET_NUMBITS32(VAR, SHIFT, MASK, VALUE) (VAR) |= htogle32(U32T((U32T(VALUE) & (MASK)) << (SHIFT)))`
  - `#define SET_NUMBITS16(VAR, SHIFT, MASK, VALUE) (VAR) |= htogle16(U16T((U16T(VALUE) & (MASK)) << (SHIFT)))`
  - `#define SET_NUMBITS8(VAR, SHIFT, MASK, VALUE) (VAR) |= U8T((U8T(VALUE) & (MASK)) << (SHIFT)))`
- 
- `#define GET_NUMBITS32(VAR, SHIFT, MASK) ((le32toh(VAR) >> (SHIFT)) & (MASK))`
  - `#define GET_NUMBITS16(VAR, SHIFT, MASK) ((le16toh(VAR) >> (SHIFT)) & (MASK))`
  - `#define GET_NUMBITS8(VAR, SHIFT, MASK) ((U8T(VAR) >> (SHIFT)) & (MASK))`

## Functions

- static bool `caerStrEquals` (const char \*s1, const char \*s2)
- static bool `caerStrEqualsUpTo` (const char \*s1, const char \*s2, size\_t len)
- static void `caerIntegerToByteArray` (const uint32\_t integer, uint8\_t \*byteArray, const uint8\_t byteArrayLength)
- static uint32\_t `caerByteArrayToInteger` (const uint8\_t \*byteArray, const uint8\_t byteArrayLength)

### 4.24.1 Detailed Description

Main libcaer header; provides inclusions for common system functions and definitions for useful macros used often in the code. Also includes the logging functions and definitions and several useful static inline functions for string comparison and byte array manipulation. When including libcaer, please make sure to always use the full path, ie. `#include <libcaer/libcaer.h>` and not just `#include <libcaer.h>`.

## 4.24.2 Macro Definition Documentation

### 4.24.2.1 CLEAR\_NUMBITS16

```
#define CLEAR_NUMBITS16(  
    VAR,  
    SHIFT,  
    MASK ) (VAR) &= htole16(~(U16T(U16T(MASK) << (SHIFT))))
```

Clear bits given by mask (amount) and shift (position).

### 4.24.2.2 CLEAR\_NUMBITS32

```
#define CLEAR_NUMBITS32(  
    VAR,  
    SHIFT,  
    MASK ) (VAR) &= htole32(~(U32T(U32T(MASK) << (SHIFT))))
```

Clear bits given by mask (amount) and shift (position).

### 4.24.2.3 CLEAR\_NUMBITS8

```
#define CLEAR_NUMBITS8(  
    VAR,  
    SHIFT,  
    MASK ) (VAR) &= U8T(~(U8T(U8T(MASK) << (SHIFT))))
```

Clear bits given by mask (amount) and shift (position).

### 4.24.2.4 GET\_NUMBITS16

```
#define GET_NUMBITS16(  
    VAR,  
    SHIFT,  
    MASK ) ((le16toh(VAR) >> (SHIFT)) & (MASK))
```

Get value of bits given by mask (amount) and shift (position).

### 4.24.2.5 GET\_NUMBITS32

```
#define GET_NUMBITS32(  
    VAR,  
    SHIFT,  
    MASK ) ((le32toh(VAR) >> (SHIFT)) & (MASK))
```

Get value of bits given by mask (amount) and shift (position).

#### 4.24.2.6 GET\_NUMBITS8

```
#define GET_NUMBITS8(  
    VAR,  
    SHIFT,  
    MASK ) ((U8T)(VAR) >> (SHIFT)) & (MASK)
```

Get value of bits given by mask (amount) and shift (position).

#### 4.24.2.7 I16T

```
#define I16T(  
    X ) ((int16_t) (X))
```

Cast argument to int16\_t (16bit signed integer).

#### 4.24.2.8 I32T

```
#define I32T(  
    X ) ((int32_t) (X))
```

Cast argument to int32\_t (32bit signed integer).

#### 4.24.2.9 I64T

```
#define I64T(  
    X ) ((int64_t) (X))
```

Cast argument to int64\_t (64bit signed integer).

#### 4.24.2.10 I8T

```
#define I8T(  
    X ) ((int8_t) (X))
```

Cast argument to int8\_t (8bit signed integer).

#### 4.24.2.11 LIBCAER\_HAVE\_OPENCV

```
#define LIBCAER_HAVE_OPENCV 1
```

libcaer OpenCV support.

#### 4.24.2.12 LIBCAER\_HAVE\_SERIALDEV

```
#define LIBCAER_HAVE_SERIALDEV 1
```

libcaer serial devices support.

#### 4.24.2.13 LIBCAER\_NAME\_STRING

```
#define LIBCAER_NAME_STRING "libcaer"
```

libcaer name string.

#### 4.24.2.14 LIBCAER\_VERSION

```
#define LIBCAER_VERSION ((2 * 10000) + (4 * 100) + 0)
```

libcaer version (MAJOR \* 10000 + MINOR \* 100 + PATCH).

#### 4.24.2.15 LIBCAER\_VERSION\_STRING

```
#define LIBCAER_VERSION_STRING "2.4.0"
```

libcaer version string.

#### 4.24.2.16 MASK\_NUMBITS32

```
#define MASK_NUMBITS32(  
    X ) U32T(U32T(U32T(1) << X) - 1)
```

Mask and keep only the lower X bits of a 32bit (unsigned) integer.

#### 4.24.2.17 MASK\_NUMBITS64

```
#define MASK_NUMBITS64(  
    X ) U64T(U64T(U64T(1) << X) - 1)
```

Mask and keep only the lower X bits of a 64bit (unsigned) integer.

#### 4.24.2.18 SET\_NUMBITS16

```
#define SET_NUMBITS16(  
    VAR,  
    SHIFT,  
    MASK,  
    VALUE ) (VAR) |= htole16(U16T((U16T(VALUE) & (MASK)) << (SHIFT)))
```

Set bits given by mask (amount) and shift (position) to a value.

#### 4.24.2.19 SET\_NUMBITS32

```
#define SET_NUMBITS32(  
    VAR,  
    SHIFT,  
    MASK,  
    VALUE ) (VAR) |= htole32(U32T((U32T(VALUE) & (MASK)) << (SHIFT)))
```

Set bits given by mask (amount) and shift (position) to a value.



#### 4.24.2.20 SET\_NUMBITS8

```
#define SET_NUMBITS8(  
    VAR,  
    SHIFT,  
    MASK,  
    VALUE ) (VAR) |= U8T((U8T(VALUE) & (MASK)) << (SHIFT))
```

Set bits given by mask (amount) and shift (position) to a value.

#### 4.24.2.21 SWAP\_VAR

```
#define SWAP_VAR(  
    type,  
    x,  
    y ) { type tmpv; tmpv = (x); (x) = (y); (y) = tmpv; }
```

Swap the two values of the two variables X and Y, of a common type TYPE.

#### 4.24.2.22 U16T

```
#define U16T(  
    X ) ((uint16_t) (X))
```

Cast argument to uint16\_t (16bit unsigned integer).

#### 4.24.2.23 U32T

```
#define U32T(  
    X ) ((uint32_t) (X))
```

Cast argument to uint32\_t (32bit unsigned integer).

#### 4.24.2.24 U64T

```
#define U64T(  
    X ) ((uint64_t) (X))
```

Cast argument to uint64\_t (64bit unsigned integer).

#### 4.24.2.25 U8T

```
#define U8T(  
    X ) ((uint8_t) (X))
```

Cast argument to uint8\_t (8bit unsigned integer).

### 4.24.3 Function Documentation

#### 4.24.3.1 caerByteArrayToInteger()

```
static uint32_t caerByteArrayToInteger (
    const uint8_t * byteArray,
    const uint8_t byteArrayLength ) [inline], [static]
```

Convert an unsigned byte array of up to four bytes into a 32bit unsigned integer. The byte array length decides how many resulting bits in the integer are set, and the single bytes are placed in the integer following big-endian ordering.

##### Parameters

<i>byteArray</i>	pointer to the byte array with parts of the value stored.
<i>byteArrayLength</i>	length of the array from which to convert.

##### Returns

integer representing the value stored in the byte array.

#### 4.24.3.2 caerIntegerToByteArray()

```
static void caerIntegerToByteArray (
    const uint32_t integer,
    uint8_t * byteArray,
    const uint8_t byteArrayLength ) [inline], [static]
```

Convert a 32bit unsigned integer into an unsigned byte array of up to four bytes. The integer will be stored in big-endian order, and the length will specify how many bits to convert, starting from the lowest bit.

##### Parameters

<i>integer</i>	the integer to convert.
<i>byteArray</i>	pointer to the byte array in which to store the converted values.
<i>byteArrayLength</i>	length of the byte array to convert to.

#### 4.24.3.3 caerStrEquals()

```
static bool caerStrEquals (
    const char * s1,
    const char * s2 ) [inline], [static]
```

Compare two strings for equality.

**Parameters**

<i>s1</i>	the first string, cannot be NULL.
<i>s2</i>	the second string, cannot be NULL.

**Returns**

true if equal, false otherwise.

**4.24.3.4 caerStrEqualsUpTo()**

```
static bool caerStrEqualsUpTo (
    const char * s1,
    const char * s2,
    size_t len ) [inline], [static]
```

Compare two strings for equality, up to a specified maximum length.

**Parameters**

<i>s1</i>	the first string, cannot be NULL.
<i>s2</i>	the second string, cannot be NULL.
<i>len</i>	maximum comparison length, cannot be zero.

**Returns**

true if equal, false otherwise.

**4.25 log.h File Reference**

```
#include <stdint.h>
#include <stdarg.h>
```

**Macros**

- #define **ATTRIBUTE\_FORMAT(N)**
- #define **ATTRIBUTE\_FORMAT\_VA(N)**

**Enumerations**

- enum [caer\\_log\\_level](#) {  
**CAER\_LOG\_EMERGENCY** = 0, **CAER\_LOG\_ALERT** = 1, **CAER\_LOG\_CRITICAL** = 2, **CAER\_LOG\_ERROR** = 3,  
**CAER\_LOG\_WARNING** = 4, **CAER\_LOG\_NOTICE** = 5, **CAER\_LOG\_INFO** = 6, **CAER\_LOG\_DEBUG** = 7  
}

## Functions

- void [caerLogLevelSet](#) (enum [caer\\_log\\_level](#) logLevel)
- enum [caer\\_log\\_level](#) [caerLogLevelGet](#) (void)
- void [caerLogFileDescriptorsSet](#) (int fd1, int fd2)
- int [caerLogFileDescriptorsGetFirst](#) (void)
- int [caerLogFileDescriptorsGetSecond](#) (void)
- void [caerLog](#) (enum [caer\\_log\\_level](#) logLevel, const char \*subSystem, const char \*format,...) [ATTRIBUTE\\_FORMAT\(3\)](#)
- void [caerLogVA](#) (enum [caer\\_log\\_level](#) logLevel, const char \*subSystem, const char \*format, va\_list args) [ATTRIBUTE\\_FORMAT\\_VA\(3\)](#)
- void [caerLogVAFull](#) (int logFileDescriptor1, int logFileDescriptor2, uint8\_t systemLogLevel, enum [caer\\_log\\_level](#) logLevel, const char \*subSystem, const char \*format, va\_list args) [ATTRIBUTE\\_FORMAT\\_VA\(6\)](#)

### 4.25.1 Detailed Description

Logging functions to print useful messages for the user.

### 4.25.2 Enumeration Type Documentation

#### 4.25.2.1 [caer\\_log\\_level](#)

```
enum caer\_log\_level
```

Log levels for [caerLog\(\)](#) logging function. Log messages only get printed if their log level is equal or above the global system log level, which can be set with [caerLogLevelSet\(\)](#). The default log level is CAER\_LOG\_ERROR. CAER\_LOG\_EMERGENCY is the most urgent log level and will always be printed, while CAER\_LOG\_DEBUG is the least urgent log level and will only be delivered if configured by the user.

### 4.25.3 Function Documentation

#### 4.25.3.1 [caerLog\(\)](#)

```
void caerLog (
    enum caer\_log\_level logLevel,
    const char * subSystem,
    const char * format,
    ... )
```

Main logging function. This function takes messages, formats them and sends them out to a file descriptor, respecting the system-wide log level setting and prepending the current time, the log level and a user-specified common string to the actual formatted output. The format is specified exactly as with the printf() family of functions. Please see their manual-page for more information.

**Parameters**

<i>logLevel</i>	the message-specific log level.
<i>subSystem</i>	a common, user-specified string to prepend before the message.
<i>format</i>	the message format string (see printf()).
...	the parameters to be formatted according to the format string (see printf()).

**4.25.3.2 caerLogFileDescriptorsGetFirst()**

```
int caerLogFileDescriptorsGetFirst (
    void )
```

Get the current output file descriptor 1.

**Returns**

the current output file descriptor 1.

**4.25.3.3 caerLogFileDescriptorsGetSecond()**

```
int caerLogFileDescriptorsGetSecond (
    void )
```

Get the current output file descriptor 2.

**Returns**

the current output file descriptor 2.

**4.25.3.4 caerLogFileDescriptorsSet()**

```
void caerLogFileDescriptorsSet (
    int fd1,
    int fd2 )
```

Set to which file descriptors log messages are sent. Up to two different file descriptors can be configured here. By default logging to STDERR only is enabled. If both file descriptors are identical, logging to it will only happen once, as if the second one was disabled.

**Parameters**

<i>fd1</i>	first file descriptor to log to. A negative value will disable it.
<i>fd2</i>	second file descriptor to log to. A negative value will disable it.

#### 4.25.3.5 caerLogLevelGet()

```
enum caer_log_level caerLogLevelGet (  
    void )
```

Get the current system-wide log level. Log messages are only printed if their level is equal or above this level.

##### Returns

the current system-wide log level.

#### 4.25.3.6 caerLogLevelSet()

```
void caerLogLevelSet (  
    enum caer_log_level logLevel )
```

Set the system-wide log level. Log messages will only be printed if their level is equal or above this level.

##### Parameters

<i>logLevel</i>	the system-wide log level.
-----------------	----------------------------

#### 4.25.3.7 caerLogVA()

```
void caerLogVA (  
    enum caer_log_level logLevel,  
    const char * subSystem,  
    const char * format,  
    va_list args )
```

Secondary logging function. This function takes messages, formats them and sends them out to a file descriptor, respecting the system-wide log level setting and prepending the current time, the log level and a user-specified common string to the actual formatted output. The format is specified exactly as with the printf() family of functions. The argument list is a va\_list as returned by va\_start(), following the vprintf() family of functions in its functionality. Please see their manual-page for more information.

##### Parameters

<i>logLevel</i>	the message-specific log level.
<i>subSystem</i>	a common, user-specified string to prepend before the message.
<i>format</i>	the message format string (see printf()).
<i>args</i>	the parameters to be formatted according to the format string (see printf()). This is an argument list as returned by va_start().

#### 4.25.3.8 caerLogVAFull()

```
void caerLogVAFull (
    int logFileDescriptor1,
    int logFileDescriptor2,
    uint8_t systemLogLevel,
    enum caer_log_level logLevel,
    const char * subSystem,
    const char * format,
    va_list args )
```

Tertiary logging function. This function takes messages, formats them and sends them out to up to two file descriptors, fully specified by the user; allows a user-given system log level setting to also be specified, and then prepends the current time, the message log level and a user-specified common string to the actual formatted output. The format is specified exactly as with the printf() family of functions. The argument list is a va\_list as returned by va\_start(), following the vprintf() family of functions in its functionality. Please see their manual-page for more information.

##### Parameters

<i>logFileDescriptor1</i>	first output file descriptor.
<i>logFileDescriptor2</i>	second output file descriptor.
<i>systemLogLevel</i>	the system-wide log level.
<i>logLevel</i>	the message-specific log level.
<i>subSystem</i>	a common, user-specified string to prepend before the message.
<i>format</i>	the message format string (see printf()).
<i>args</i>	the parameters to be formatted according to the format string (see printf()). This is an argument list as returned by va_start().

## 4.26 network.h File Reference

```
#include "libcaer.h"
```

### Macros

- `#define AEDAT3_NETWORK_HEADER_LENGTH 20`
- `#define AEDAT3_NETWORK_MAGIC_NUMBER 0x1D378BC90B9A6658`
- `#define AEDAT3_NETWORK_VERSION 0x01`
- `#define AEDAT3_FILE_VERSION "3.1"`
- `#define AEDAT3_MAX_UDP_SIZE (1472 - AEDAT3_NETWORK_HEADER_LENGTH)`

### Functions

- **PACKED\_STRUCT** (struct aedat3\_network\_header { int64\_t magicNumber;int64\_t sequenceNumber;int8\_t versionNumber;int8\_t formatNumber;int16\_t sourceID;})
- static struct aedat3\_network\_header **caerParseNetworkHeader** (const uint8\_t \*dataBuffer)



#### 4.26.1 Detailed Description

Useful functions for AEDAT 3.X network streams.

## 4.27 portable\_endian.h File Reference

### Functions

- static float **htobeflt** (float val)
- static float **htoleflt** (float val)
- static float **beflttoh** (float val)
- static float **leflttoh** (float val)

#### 4.27.1 Detailed Description

Endianness conversion functions for a wide variety of systems, including Linux, FreeBSD, MacOS X and Windows.



# Index

CAER\_CONFIGURATION\_CONST\_ITERATOR\_ALL↔  
\_START  
config.h, [182](#)

CAER\_CONFIGURATION\_CONST\_ITERATOR\_VA↔  
LID\_START  
config.h, [183](#)

CAER\_CONFIGURATION\_CONST\_REVERSE\_ITE↔  
RATOR\_ALL\_START  
config.h, [183](#)

CAER\_CONFIGURATION\_CONST\_REVERSE\_ITE↔  
RATOR\_VALID\_START  
config.h, [183](#)

CAER\_CONFIGURATION\_ITERATOR\_ALL\_END  
config.h, [184](#)

CAER\_CONFIGURATION\_ITERATOR\_ALL\_START  
config.h, [184](#)

CAER\_CONFIGURATION\_ITERATOR\_VALID\_END  
config.h, [184](#)

CAER\_CONFIGURATION\_ITERATOR\_VALID\_START  
config.h, [184](#)

CAER\_CONFIGURATION\_REVERSE\_ITERATOR↔  
ALL\_END  
config.h, [185](#)

CAER\_CONFIGURATION\_REVERSE\_ITERATOR↔  
ALL\_START  
config.h, [185](#)

CAER\_CONFIGURATION\_REVERSE\_ITERATOR↔  
VALID\_END  
config.h, [185](#)

CAER\_CONFIGURATION\_REVERSE\_ITERATOR↔  
VALID\_START  
config.h, [185](#)

CAER\_DEFAULT\_EVENT\_TYPES\_COUNT  
common.h, [166](#)

CAER\_DEVICE\_DAVIS\_FX2  
davis.h, [21](#)

CAER\_DEVICE\_DAVIS\_FX3  
davis.h, [21](#)

CAER\_DEVICE\_DAVIS  
davis.h, [21](#)

CAER\_DEVICE\_DVS128  
dvs128.h, [131](#)

CAER\_DEVICE\_DYNAPSE  
dynapse.h, [139](#)

CAER\_DEVICE\_EDVS  
edvs.h, [158](#)

CAER\_EAR\_CONST\_ITERATOR\_ALL\_START  
ear.h, [194](#)

CAER\_EAR\_CONST\_ITERATOR\_VALID\_START  
ear.h, [194](#)

CAER\_EAR\_CONST\_REVERSE\_ITERATOR\_ALL↔  
START  
ear.h, [194](#)

CAER\_EAR\_CONST\_REVERSE\_ITERATOR\_VALI↔  
D\_START  
ear.h, [195](#)

CAER\_EAR\_ITERATOR\_ALL\_END  
ear.h, [195](#)

CAER\_EAR\_ITERATOR\_ALL\_START  
ear.h, [195](#)

CAER\_EAR\_ITERATOR\_VALID\_END  
ear.h, [196](#)

CAER\_EAR\_ITERATOR\_VALID\_START  
ear.h, [196](#)

CAER\_EAR\_REVERSE\_ITERATOR\_ALL\_END  
ear.h, [196](#)

CAER\_EAR\_REVERSE\_ITERATOR\_ALL\_START  
ear.h, [196](#)

CAER\_EAR\_REVERSE\_ITERATOR\_VALID\_END  
ear.h, [197](#)

CAER\_EAR\_REVERSE\_ITERATOR\_VALID\_START  
ear.h, [197](#)

CAER\_EVENT\_PACKET\_CONTAINER\_CONST\_IT↔  
ERATOR\_START  
packetContainer.h, [264](#)

CAER\_EVENT\_PACKET\_CONTAINER\_ITERATOR↔  
\_END  
packetContainer.h, [264](#)

CAER\_EVENT\_PACKET\_CONTAINER\_ITERATOR↔  
\_START  
packetContainer.h, [265](#)

CAER\_EVENT\_PACKET\_HEADER\_SIZE  
common.h, [166](#)

CAER\_FRAME\_CONST\_ITERATOR\_ALL\_START  
frame.h, [207](#)

CAER\_FRAME\_CONST\_ITERATOR\_VALID\_START  
frame.h, [207](#)

CAER\_FRAME\_CONST\_REVERSE\_ITERATOR\_AL↔  
L\_START  
frame.h, [207](#)

CAER\_FRAME\_CONST\_REVERSE\_ITERATOR\_VA↔  
LID\_START  
frame.h, [208](#)

CAER\_FRAME\_ITERATOR\_ALL\_END  
frame.h, [208](#)

CAER\_FRAME\_ITERATOR\_ALL\_START  
frame.h, [208](#)

CAER\_FRAME\_ITERATOR\_VALID\_END

- frame.h, [209](#)
- CAER\_FRAME\_ITERATOR\_VALID\_START
  - frame.h, [209](#)
- CAER\_FRAME\_REVERSE\_ITERATOR\_ALL\_END
  - frame.h, [209](#)
- CAER\_FRAME\_REVERSE\_ITERATOR\_ALL\_START
  - frame.h, [209](#)
- CAER\_FRAME\_REVERSE\_ITERATOR\_VALID\_END
  - frame.h, [210](#)
- CAER\_FRAME\_REVERSE\_ITERATOR\_VALID\_START
  - frame.h, [210](#)
- CAER\_HOST\_CONFIG\_DATAEXCHANGE\_BLOCKING
  - device.h, [125](#)
- CAER\_HOST\_CONFIG\_DATAEXCHANGE\_BUFFER\_SIZE
  - device.h, [125](#)
- CAER\_HOST\_CONFIG\_DATAEXCHANGE\_START\_PRODUCERS
  - device.h, [125](#)
- CAER\_HOST\_CONFIG\_DATAEXCHANGE\_STOP\_PRODUCERS
  - device.h, [125](#)
- CAER\_HOST\_CONFIG\_DATAEXCHANGE
  - device.h, [124](#)
- CAER\_HOST\_CONFIG\_LOG\_LEVEL
  - device.h, [125](#)
- CAER\_HOST\_CONFIG\_LOG
  - device.h, [125](#)
- CAER\_HOST\_CONFIG\_PACKETS\_MAX\_CONTAINERS\_INTERVAL
  - device.h, [126](#)
- CAER\_HOST\_CONFIG\_PACKETS\_MAX\_CONTAINERS\_PACKET\_SIZE
  - device.h, [126](#)
- CAER\_HOST\_CONFIG\_PACKETS
  - device.h, [126](#)
- CAER\_HOST\_CONFIG\_SERIAL\_BAUD\_RATE\_12M
  - serial.h, [161](#)
- CAER\_HOST\_CONFIG\_SERIAL\_BAUD\_RATE\_2M
  - serial.h, [161](#)
- CAER\_HOST\_CONFIG\_SERIAL\_BAUD\_RATE\_4M
  - serial.h, [162](#)
- CAER\_HOST\_CONFIG\_SERIAL\_BAUD\_RATE\_8M
  - serial.h, [162](#)
- CAER\_HOST\_CONFIG\_SERIAL\_READ\_SIZE
  - serial.h, [162](#)
- CAER\_HOST\_CONFIG\_SERIAL
  - serial.h, [161](#)
- CAER\_HOST\_CONFIG\_USB\_BUFFER\_NUMBER
  - usb.h, [163](#)
- CAER\_HOST\_CONFIG\_USB\_BUFFER\_SIZE
  - usb.h, [163](#)
- CAER\_HOST\_CONFIG\_USB
  - usb.h, [163](#)
- CAER\_IMU6\_CONST\_ITERATOR\_ALL\_START
  - imu6.h, [233](#)
- CAER\_IMU6\_CONST\_ITERATOR\_VALID\_START
  - imu6.h, [233](#)
- CAER\_IMU6\_CONST\_REVERSE\_ITERATOR\_ALL\_START
  - imu6.h, [233](#)
- CAER\_IMU6\_CONST\_REVERSE\_ITERATOR\_VALID\_START
  - imu6.h, [234](#)
- CAER\_IMU6\_ITERATOR\_ALL\_END
  - imu6.h, [234](#)
- CAER\_IMU6\_ITERATOR\_ALL\_START
  - imu6.h, [234](#)
- CAER\_IMU6\_ITERATOR\_VALID\_END
  - imu6.h, [235](#)
- CAER\_IMU6\_ITERATOR\_VALID\_START
  - imu6.h, [235](#)
- CAER\_IMU6\_REVERSE\_ITERATOR\_ALL\_END
  - imu6.h, [235](#)
- CAER\_IMU6\_REVERSE\_ITERATOR\_ALL\_START
  - imu6.h, [235](#)
- CAER\_IMU6\_REVERSE\_ITERATOR\_VALID\_END
  - imu6.h, [236](#)
- CAER\_IMU6\_REVERSE\_ITERATOR\_VALID\_START
  - imu6.h, [236](#)
- CAER\_IMU9\_CONST\_ITERATOR\_ALL\_START
  - imu9.h, [247](#)
- CAER\_IMU9\_CONST\_ITERATOR\_VALID\_START
  - imu9.h, [247](#)
- CAER\_IMU9\_CONST\_REVERSE\_ITERATOR\_ALL\_START
  - imu9.h, [247](#)
- CAER\_IMU9\_CONST\_REVERSE\_ITERATOR\_VALID\_START
  - imu9.h, [248](#)
- CAER\_IMU9\_ITERATOR\_ALL\_END
  - imu9.h, [248](#)
- CAER\_IMU9\_ITERATOR\_ALL\_START
  - imu9.h, [248](#)
- CAER\_IMU9\_ITERATOR\_VALID\_END
  - imu9.h, [249](#)
- CAER\_IMU9\_ITERATOR\_VALID\_START
  - imu9.h, [249](#)
- CAER\_IMU9\_REVERSE\_ITERATOR\_ALL\_END
  - imu9.h, [249](#)
- CAER\_IMU9\_REVERSE\_ITERATOR\_ALL\_START
  - imu9.h, [249](#)
- CAER\_IMU9\_REVERSE\_ITERATOR\_VALID\_END
  - imu9.h, [250](#)
- CAER\_IMU9\_REVERSE\_ITERATOR\_VALID\_START
  - imu9.h, [250](#)
- CAER\_ITERATOR\_ALL\_END
  - common.h, [166](#)
- CAER\_ITERATOR\_ALL\_START
  - common.h, [167](#)
- CAER\_ITERATOR\_VALID\_END
  - common.h, [167](#)
- CAER\_ITERATOR\_VALID\_START
  - common.h, [167](#)

- CAER\_POINT1D\_CONST\_ITERATOR\_ALL\_START  
point1d.h, [273](#)
- CAER\_POINT1D\_CONST\_ITERATOR\_VALID\_START  
point1d.h, [273](#)
- CAER\_POINT1D\_CONST\_REVERSE\_ITERATOR\_↔  
ALL\_START  
point1d.h, [273](#)
- CAER\_POINT1D\_CONST\_REVERSE\_ITERATOR\_↔  
VALID\_START  
point1d.h, [274](#)
- CAER\_POINT1D\_ITERATOR\_ALL\_END  
point1d.h, [274](#)
- CAER\_POINT1D\_ITERATOR\_ALL\_START  
point1d.h, [274](#)
- CAER\_POINT1D\_ITERATOR\_VALID\_END  
point1d.h, [275](#)
- CAER\_POINT1D\_ITERATOR\_VALID\_START  
point1d.h, [275](#)
- CAER\_POINT1D\_REVERSE\_ITERATOR\_ALL\_END  
point1d.h, [275](#)
- CAER\_POINT1D\_REVERSE\_ITERATOR\_ALL\_STA↔  
RT  
point1d.h, [275](#)
- CAER\_POINT1D\_REVERSE\_ITERATOR\_VALID\_END  
point1d.h, [276](#)
- CAER\_POINT1D\_REVERSE\_ITERATOR\_VALID\_S↔  
TART  
point1d.h, [276](#)
- CAER\_POINT2D\_CONST\_ITERATOR\_ALL\_START  
point2d.h, [285](#)
- CAER\_POINT2D\_CONST\_ITERATOR\_VALID\_START  
point2d.h, [285](#)
- CAER\_POINT2D\_CONST\_REVERSE\_ITERATOR\_↔  
ALL\_START  
point2d.h, [285](#)
- CAER\_POINT2D\_CONST\_REVERSE\_ITERATOR\_↔  
VALID\_START  
point2d.h, [286](#)
- CAER\_POINT2D\_ITERATOR\_ALL\_END  
point2d.h, [286](#)
- CAER\_POINT2D\_ITERATOR\_ALL\_START  
point2d.h, [286](#)
- CAER\_POINT2D\_ITERATOR\_VALID\_END  
point2d.h, [287](#)
- CAER\_POINT2D\_ITERATOR\_VALID\_START  
point2d.h, [287](#)
- CAER\_POINT2D\_REVERSE\_ITERATOR\_ALL\_END  
point2d.h, [287](#)
- CAER\_POINT2D\_REVERSE\_ITERATOR\_ALL\_STA↔  
RT  
point2d.h, [287](#)
- CAER\_POINT2D\_REVERSE\_ITERATOR\_VALID\_END  
point2d.h, [288](#)
- CAER\_POINT2D\_REVERSE\_ITERATOR\_VALID\_S↔  
TART  
point2d.h, [288](#)
- CAER\_POINT3D\_CONST\_ITERATOR\_ALL\_START  
point3d.h, [297](#)
- CAER\_POINT3D\_CONST\_ITERATOR\_VALID\_START  
point3d.h, [298](#)
- CAER\_POINT3D\_CONST\_REVERSE\_ITERATOR\_↔  
ALL\_START  
point3d.h, [298](#)
- CAER\_POINT3D\_CONST\_REVERSE\_ITERATOR\_↔  
VALID\_START  
point3d.h, [298](#)
- CAER\_POINT3D\_ITERATOR\_ALL\_END  
point3d.h, [299](#)
- CAER\_POINT3D\_ITERATOR\_ALL\_START  
point3d.h, [299](#)
- CAER\_POINT3D\_ITERATOR\_VALID\_END  
point3d.h, [299](#)
- CAER\_POINT3D\_ITERATOR\_VALID\_START  
point3d.h, [299](#)
- CAER\_POINT3D\_REVERSE\_ITERATOR\_ALL\_END  
point3d.h, [300](#)
- CAER\_POINT3D\_REVERSE\_ITERATOR\_ALL\_STA↔  
RT  
point3d.h, [300](#)
- CAER\_POINT3D\_REVERSE\_ITERATOR\_VALID\_END  
point3d.h, [300](#)
- CAER\_POINT3D\_REVERSE\_ITERATOR\_VALID\_S↔  
TART  
point3d.h, [300](#)
- CAER\_POINT4D\_CONST\_ITERATOR\_ALL\_START  
point4d.h, [310](#)
- CAER\_POINT4D\_CONST\_ITERATOR\_VALID\_START  
point4d.h, [311](#)
- CAER\_POINT4D\_CONST\_REVERSE\_ITERATOR\_↔  
ALL\_START  
point4d.h, [311](#)
- CAER\_POINT4D\_CONST\_REVERSE\_ITERATOR\_↔  
VALID\_START  
point4d.h, [311](#)
- CAER\_POINT4D\_ITERATOR\_ALL\_END  
point4d.h, [312](#)
- CAER\_POINT4D\_ITERATOR\_ALL\_START  
point4d.h, [312](#)
- CAER\_POINT4D\_ITERATOR\_VALID\_END  
point4d.h, [312](#)
- CAER\_POINT4D\_ITERATOR\_VALID\_START  
point4d.h, [312](#)
- CAER\_POINT4D\_REVERSE\_ITERATOR\_ALL\_END  
point4d.h, [313](#)
- CAER\_POINT4D\_REVERSE\_ITERATOR\_ALL\_STA↔  
RT  
point4d.h, [313](#)
- CAER\_POINT4D\_REVERSE\_ITERATOR\_VALID\_END  
point4d.h, [313](#)
- CAER\_POINT4D\_REVERSE\_ITERATOR\_VALID\_S↔  
TART  
point4d.h, [313](#)
- CAER\_POLARITY\_CONST\_ITERATOR\_ALL\_START  
polarity.h, [324](#)
- CAER\_POLARITY\_CONST\_ITERATOR\_VALID\_ST↔  
ART

- polarity.h, [324](#)
- CAER\_POLARITY\_CONST\_REVERSE\_ITERATOR↔
  - \_ALL\_START
    - polarity.h, [325](#)
- CAER\_POLARITY\_CONST\_REVERSE\_ITERATOR↔
  - \_VALID\_START
    - polarity.h, [325](#)
- CAER\_POLARITY\_ITERATOR\_ALL\_END
  - polarity.h, [325](#)
- CAER\_POLARITY\_ITERATOR\_ALL\_START
  - polarity.h, [326](#)
- CAER\_POLARITY\_ITERATOR\_VALID\_END
  - polarity.h, [326](#)
- CAER\_POLARITY\_ITERATOR\_VALID\_START
  - polarity.h, [326](#)
- CAER\_POLARITY\_REVERSE\_ITERATOR\_ALL\_END
  - polarity.h, [326](#)
- CAER\_POLARITY\_REVERSE\_ITERATOR\_ALL\_ST↔
  - ART
    - polarity.h, [327](#)
- CAER\_POLARITY\_REVERSE\_ITERATOR\_VALID↔
  - END
    - polarity.h, [327](#)
- CAER\_POLARITY\_REVERSE\_ITERATOR\_VALID↔
  - START
    - polarity.h, [327](#)
- CAER\_SAMPLE\_CONST\_ITERATOR\_ALL\_START
  - sample.h, [336](#)
- CAER\_SAMPLE\_CONST\_ITERATOR\_VALID\_START
  - sample.h, [336](#)
- CAER\_SAMPLE\_CONST\_REVERSE\_ITERATOR\_A↔
  - LL\_START
    - sample.h, [337](#)
- CAER\_SAMPLE\_CONST\_REVERSE\_ITERATOR\_V↔
  - ALID\_START
    - sample.h, [337](#)
- CAER\_SAMPLE\_ITERATOR\_ALL\_END
  - sample.h, [337](#)
- CAER\_SAMPLE\_ITERATOR\_ALL\_START
  - sample.h, [338](#)
- CAER\_SAMPLE\_ITERATOR\_VALID\_END
  - sample.h, [338](#)
- CAER\_SAMPLE\_ITERATOR\_VALID\_START
  - sample.h, [338](#)
- CAER\_SAMPLE\_REVERSE\_ITERATOR\_ALL\_END
  - sample.h, [338](#)
- CAER\_SAMPLE\_REVERSE\_ITERATOR\_ALL\_START
  - sample.h, [339](#)
- CAER\_SAMPLE\_REVERSE\_ITERATOR\_VALID\_END
  - sample.h, [339](#)
- CAER\_SAMPLE\_REVERSE\_ITERATOR\_VALID\_ST↔
  - ART
    - sample.h, [339](#)
- CAER\_SPECIAL\_CONST\_ITERATOR\_ALL\_START
  - special.h, [347](#)
- CAER\_SPECIAL\_CONST\_ITERATOR\_VALID\_START
  - special.h, [348](#)
- CAER\_SPECIAL\_CONST\_REVERSE\_ITERATOR↔
  - ALL\_START
    - special.h, [348](#)
- CAER\_SPECIAL\_CONST\_REVERSE\_ITERATOR↔
  - VALID\_START
    - special.h, [348](#)
- CAER\_SPECIAL\_ITERATOR\_ALL\_END
  - special.h, [349](#)
- CAER\_SPECIAL\_ITERATOR\_ALL\_START
  - special.h, [349](#)
- CAER\_SPECIAL\_ITERATOR\_VALID\_END
  - special.h, [349](#)
- CAER\_SPECIAL\_ITERATOR\_VALID\_START
  - special.h, [349](#)
- CAER\_SPECIAL\_REVERSE\_ITERATOR\_ALL\_END
  - special.h, [350](#)
- CAER\_SPECIAL\_REVERSE\_ITERATOR\_ALL\_START
  - special.h, [350](#)
- CAER\_SPECIAL\_REVERSE\_ITERATOR\_VALID\_END
  - special.h, [350](#)
- CAER\_SPECIAL\_REVERSE\_ITERATOR\_VALID\_S↔
  - TART
    - special.h, [350](#)
- CAER\_SPIKE\_CONST\_ITERATOR\_ALL\_START
  - spike.h, [361](#)
- CAER\_SPIKE\_CONST\_ITERATOR\_VALID\_START
  - spike.h, [361](#)
- CAER\_SPIKE\_CONST\_REVERSE\_ITERATOR\_ALL↔
  - \_START
    - spike.h, [362](#)
- CAER\_SPIKE\_CONST\_REVERSE\_ITERATOR\_VAL↔
  - ID\_START
    - spike.h, [362](#)
- CAER\_SPIKE\_ITERATOR\_ALL\_END
  - spike.h, [362](#)
- CAER\_SPIKE\_ITERATOR\_ALL\_START
  - spike.h, [363](#)
- CAER\_SPIKE\_ITERATOR\_VALID\_END
  - spike.h, [363](#)
- CAER\_SPIKE\_ITERATOR\_VALID\_START
  - spike.h, [363](#)
- CAER\_SPIKE\_REVERSE\_ITERATOR\_ALL\_END
  - spike.h, [363](#)
- CAER\_SPIKE\_REVERSE\_ITERATOR\_ALL\_START
  - spike.h, [364](#)
- CAER\_SPIKE\_REVERSE\_ITERATOR\_VALID\_END
  - spike.h, [364](#)
- CAER\_SPIKE\_REVERSE\_ITERATOR\_VALID\_START
  - spike.h, [364](#)
- CLEAR\_NUMBITS16
  - libcaer.h, [374](#)
- CLEAR\_NUMBITS32
  - libcaer.h, [374](#)
- CLEAR\_NUMBITS8
  - libcaer.h, [374](#)
- CONFIG\_MODULE\_ADDR\_MASK
  - config.h, [186](#)
- CONFIG\_MODULE\_ADDR\_SHIFT

- config.h, [186](#)
- caer\_bias\_coarsefine, [5](#)
- caer\_bias\_dynapse, [5](#)
- caer\_bias\_shiftedsources, [6](#)
- caer\_bias\_shiftedsources\_operating\_mode
  - davis.h, [119](#)
- caer\_bias\_shiftedsources\_voltage\_level
  - davis.h, [120](#)
- caer\_bias\_vdac, [7](#)
- caer\_davis\_info, [7](#)
- caer\_default\_event\_types
  - common.h, [168](#)
- caer\_dvs128\_info, [8](#)
- caer\_dynapse\_info, [9](#)
- caer\_edvs\_info, [10](#)
- caer\_frame\_event\_color\_channels
  - frame.h, [212](#)
- caer\_frame\_event\_color\_filter
  - frame.h, [212](#)
- caer\_log\_level
  - log.h, [381](#)
- caer\_special\_event\_types
  - special.h, [352](#)
- caerBiasCoarseFineGenerate
  - davis.h, [120](#)
- caerBiasCoarseFineParse
  - davis.h, [120](#)
- caerBiasDynapseGenerate
  - dynapse.h, [149](#)
- caerBiasDynapseParse
  - dynapse.h, [149](#)
- caerBiasShiftedSourceGenerate
  - davis.h, [122](#)
- caerBiasShiftedSourceParse
  - davis.h, [122](#)
- caerBiasVDACGenerate
  - davis.h, [122](#)
- caerBiasVDACParse
  - davis.h, [123](#)
- caerByteArrayToInteger
  - libcaer.h, [378](#)
- caerConfigurationEvent
  - config.h, [186](#)
- caerConfigurationEventGetModuleAddress
  - config.h, [187](#)
- caerConfigurationEventGetParameter
  - config.h, [187](#)
- caerConfigurationEventGetParameterAddress
  - config.h, [187](#)
- caerConfigurationEventGetTimestamp
  - config.h, [188](#)
- caerConfigurationEventGetTimestamp64
  - config.h, [188](#)
- caerConfigurationEventInvalidate
  - config.h, [188](#)
- caerConfigurationEventIsValid
  - config.h, [189](#)
- caerConfigurationEventPacket
  - config.h, [186](#)
- caerConfigurationEventPacketAllocate
  - config.h, [189](#)
- caerConfigurationEventPacketGetEvent
  - config.h, [189](#)
- caerConfigurationEventPacketGetEventConst
  - config.h, [190](#)
- caerConfigurationEventSetModuleAddress
  - config.h, [190](#)
- caerConfigurationEventSetParameter
  - config.h, [191](#)
- caerConfigurationEventSetParameterAddress
  - config.h, [191](#)
- caerConfigurationEventSetTimestamp
  - config.h, [191](#)
- caerConfigurationEventValidate
  - config.h, [192](#)
- caerDVS128InfoGet
  - dvs128.h, [133](#)
- caerDavisInfoGet
  - davis.h, [123](#)
- caerDeviceClose
  - device.h, [126](#)
- caerDeviceConfigGet
  - device.h, [127](#)
- caerDeviceConfigSet
  - device.h, [127](#)
- caerDeviceDataGet
  - device.h, [128](#)
- caerDeviceDataStart
  - device.h, [128](#)
- caerDeviceDataStop
  - device.h, [129](#)
- caerDeviceHandle
  - device.h, [126](#)
- caerDeviceOpen
  - usb.h, [164](#)
- caerDeviceOpenSerial
  - serial.h, [162](#)
- caerDeviceSendDefaultConfig
  - device.h, [129](#)
- caerDynapseCoreAddrToNeuronId
  - dynapse.h, [150](#)
- caerDynapseCoreXYToNeuronId
  - dynapse.h, [150](#)
- caerDynapseGenerateCamBits
  - dynapse.h, [150](#)
- caerDynapseGenerateSramBits
  - dynapse.h, [151](#)
- caerDynapseInfoGet
  - dynapse.h, [152](#)
- caerDynapseSendDataToUSB
  - dynapse.h, [152](#)
- caerDynapseSpikeEventFromXY
  - dynapse.h, [152](#)
- caerDynapseSpikeEventGetX
  - dynapse.h, [153](#)
- caerDynapseSpikeEventGetY

- dynapse.h, 153
- caerDynapseWriteCam
  - dynapse.h, 154
- caerDynapseWritePoissonSpikeRate
  - dynapse.h, 154
- caerDynapseWriteSram
  - dynapse.h, 155
- caerDynapseWriteSramWords
  - dynapse.h, 156
- caerDynapseWriteSramN
  - dynapse.h, 155
- caerEDVSInfoGet
  - edvs.h, 160
- caerEarEvent
  - ear.h, 199
- caerEarEventGetChannel
  - ear.h, 199
- caerEarEventGetEar
  - ear.h, 199
- caerEarEventGetTimestamp
  - ear.h, 200
- caerEarEventGetTimestamp64
  - ear.h, 200
- caerEarEventInvalidate
  - ear.h, 201
- caerEarEventsIsValid
  - ear.h, 201
- caerEarEventPacket
  - ear.h, 199
- caerEarEventPacketAllocate
  - ear.h, 201
- caerEarEventPacketGetEvent
  - ear.h, 202
- caerEarEventPacketGetEventConst
  - ear.h, 202
- caerEarEventSetChannel
  - ear.h, 203
- caerEarEventSetEar
  - ear.h, 203
- caerEarEventSetTimestamp
  - ear.h, 203
- caerEarEventValidate
  - ear.h, 204
- caerEventPacketAppend
  - common.h, 169
- caerEventPacketClean
  - common.h, 169
- caerEventPacketClear
  - common.h, 170
- caerEventPacketContainer
  - packetContainer.h, 265
- caerEventPacketContainerAllocate
  - packetContainer.h, 265
- caerEventPacketContainerCopyAllEvents
  - packetContainer.h, 266
- caerEventPacketContainerCopyValidEvents
  - packetContainer.h, 266
- caerEventPacketContainerFindEventPacketByType
  - packetContainer.h, 266
- caerEventPacketContainerFindEventPacketByType↔
  - Const
    - packetContainer.h, 267
- caerEventPacketContainerFree
  - packetContainer.h, 267
- caerEventPacketContainerGetEventPacket
  - packetContainer.h, 268
- caerEventPacketContainerGetEventPacketConst
  - packetContainer.h, 268
- caerEventPacketContainerGetEventPacketsNumber
  - packetContainer.h, 268
- caerEventPacketContainerGetEventsNumber
  - packetContainer.h, 269
- caerEventPacketContainerGetEventsValidNumber
  - packetContainer.h, 269
- caerEventPacketContainerGetHighestEventTimestamp
  - packetContainer.h, 269
- caerEventPacketContainerGetLowestEventTimestamp
  - packetContainer.h, 270
- caerEventPacketContainerSetEventPacket
  - packetContainer.h, 270
- caerEventPacketContainerSetEventPacketsNumber
  - packetContainer.h, 271
- caerEventPacketContainerUpdateStatistics
  - packetContainer.h, 271
- caerEventPacketCopy
  - common.h, 170
- caerEventPacketCopyOnlyEvents
  - common.h, 170
- caerEventPacketCopyOnlyValidEvents
  - common.h, 171
- caerEventPacketEquals
  - common.h, 171
- caerEventPacketGetDataSize
  - common.h, 171
- caerEventPacketGetSize
  - common.h, 172
- caerEventPacketGrow
  - common.h, 172
- caerEventPacketHeader
  - common.h, 168
- caerEventPacketHeaderGetEventCapacity
  - common.h, 172
- caerEventPacketHeaderGetEventNumber
  - common.h, 173
- caerEventPacketHeaderGetEventSize
  - common.h, 173
- caerEventPacketHeaderGetEventSource
  - common.h, 174
- caerEventPacketHeaderGetEventTSOffset
  - common.h, 174
- caerEventPacketHeaderGetEventTSOverflow
  - common.h, 174
- caerEventPacketHeaderGetEventType
  - common.h, 175
- caerEventPacketHeaderGetEventValid
  - common.h, 175



caerEventPacketHeaderSetEventCapacity  
common.h, [175](#)

caerEventPacketHeaderSetEventNumber  
common.h, [176](#)

caerEventPacketHeaderSetEventSize  
common.h, [176](#)

caerEventPacketHeaderSetEventSource  
common.h, [176](#)

caerEventPacketHeaderSetEventTSOffset  
common.h, [177](#)

caerEventPacketHeaderSetEventTSOverflow  
common.h, [177](#)

caerEventPacketHeaderSetEventType  
common.h, [177](#)

caerEventPacketHeaderSetEventValid  
common.h, [178](#)

caerEventPacketResize  
common.h, [178](#)

caerFrameEvent  
frame.h, [212](#)

caerFrameEventGetChannelNumber  
frame.h, [213](#)

caerFrameEventGetColorFilter  
frame.h, [213](#)

caerFrameEventGetExposureLength  
frame.h, [213](#)

caerFrameEventGetLengthX  
frame.h, [214](#)

caerFrameEventGetLengthY  
frame.h, [214](#)

caerFrameEventGetPixel  
frame.h, [215](#)

caerFrameEventGetPixelArrayUnsafe  
frame.h, [215](#)

caerFrameEventGetPixelArrayUnsafeConst  
frame.h, [215](#)

caerFrameEventGetPixelForChannel  
frame.h, [216](#)

caerFrameEventGetPixelForChannelUnsafe  
frame.h, [216](#)

caerFrameEventGetPixelUnsafe  
frame.h, [217](#)

caerFrameEventGetPixelsMaxIndex  
frame.h, [217](#)

caerFrameEventGetPixelsSize  
frame.h, [217](#)

caerFrameEventGetPositionX  
frame.h, [218](#)

caerFrameEventGetPositionY  
frame.h, [218](#)

caerFrameEventGetROIIdentifier  
frame.h, [219](#)

caerFrameEventGetTSEndOfExposure  
frame.h, [220](#)

caerFrameEventGetTSEndOfExposure64  
frame.h, [220](#)

caerFrameEventGetTSEndOfFrame  
frame.h, [221](#)

caerFrameEventGetTSEndOfFrame64  
frame.h, [221](#)

caerFrameEventGetTSStartOfExposure  
frame.h, [222](#)

caerFrameEventGetTSStartOfExposure64  
frame.h, [222](#)

caerFrameEventGetTSStartOfFrame  
frame.h, [222](#)

caerFrameEventGetTSStartOfFrame64  
frame.h, [223](#)

caerFrameEventGetTimestamp  
frame.h, [219](#)

caerFrameEventGetTimestamp64  
frame.h, [219](#)

caerFrameEventInvalidate  
frame.h, [223](#)

caerFrameEventIsValid  
frame.h, [223](#)

caerFrameEventPacket  
frame.h, [212](#)

caerFrameEventPacketAllocate  
frame.h, [224](#)

caerFrameEventPacketGetEvent  
frame.h, [224](#)

caerFrameEventPacketGetEventConst  
frame.h, [225](#)

caerFrameEventPacketGetPixelsMaxIndex  
frame.h, [225](#)

caerFrameEventPacketGetPixelsSize  
frame.h, [226](#)

caerFrameEventSetColorFilter  
frame.h, [226](#)

caerFrameEventSetLengthXLengthYChannelNumber  
frame.h, [226](#)

caerFrameEventSetPixel  
frame.h, [227](#)

caerFrameEventSetPixelForChannel  
frame.h, [227](#)

caerFrameEventSetPixelForChannelUnsafe  
frame.h, [228](#)

caerFrameEventSetPixelUnsafe  
frame.h, [228](#)

caerFrameEventSetPositionX  
frame.h, [229](#)

caerFrameEventSetPositionY  
frame.h, [229](#)

caerFrameEventSetROIIdentifier  
frame.h, [229](#)

caerFrameEventSetTSEndOfExposure  
frame.h, [229](#)

caerFrameEventSetTSEndOfFrame  
frame.h, [230](#)

caerFrameEventSetTSStartOfExposure  
frame.h, [230](#)

caerFrameEventSetTSStartOfFrame  
frame.h, [230](#)

caerFrameEventValidate  
frame.h, [231](#)

caerGenericEventCopy  
common.h, 179

caerGenericEventGetEvent  
common.h, 179

caerGenericEventGetTimestamp  
common.h, 179

caerGenericEventGetTimestamp64  
common.h, 180

caerGenericEventsIsValid  
common.h, 180

caerIMU6Event  
imu6.h, 236

caerIMU6EventGetAccelX  
imu6.h, 237

caerIMU6EventGetAccelY  
imu6.h, 237

caerIMU6EventGetAccelZ  
imu6.h, 238

caerIMU6EventGetGyroX  
imu6.h, 238

caerIMU6EventGetGyroY  
imu6.h, 238

caerIMU6EventGetGyroZ  
imu6.h, 239

caerIMU6EventGetTemp  
imu6.h, 239

caerIMU6EventGetTimestamp  
imu6.h, 239

caerIMU6EventGetTimestamp64  
imu6.h, 240

caerIMU6EventInvalidate  
imu6.h, 240

caerIMU6EventsIsValid  
imu6.h, 241

caerIMU6EventPacket  
imu6.h, 237

caerIMU6EventPacketAllocate  
imu6.h, 241

caerIMU6EventPacketGetEvent  
imu6.h, 241

caerIMU6EventPacketGetEventConst  
imu6.h, 242

caerIMU6EventSetAccelX  
imu6.h, 242

caerIMU6EventSetAccelY  
imu6.h, 242

caerIMU6EventSetAccelZ  
imu6.h, 243

caerIMU6EventSetGyroX  
imu6.h, 243

caerIMU6EventSetGyroY  
imu6.h, 243

caerIMU6EventSetGyroZ  
imu6.h, 244

caerIMU6EventSetTemp  
imu6.h, 244

caerIMU6EventSetTimestamp  
imu6.h, 244

caerIMU6EventValidate  
imu6.h, 245

caerIMU9Event  
imu9.h, 250

caerIMU9EventGetAccelX  
imu9.h, 251

caerIMU9EventGetAccelY  
imu9.h, 251

caerIMU9EventGetAccelZ  
imu9.h, 252

caerIMU9EventGetCompX  
imu9.h, 252

caerIMU9EventGetCompY  
imu9.h, 252

caerIMU9EventGetCompZ  
imu9.h, 253

caerIMU9EventGetGyroX  
imu9.h, 253

caerIMU9EventGetGyroY  
imu9.h, 253

caerIMU9EventGetGyroZ  
imu9.h, 255

caerIMU9EventGetTemp  
imu9.h, 255

caerIMU9EventGetTimestamp  
imu9.h, 255

caerIMU9EventGetTimestamp64  
imu9.h, 256

caerIMU9EventInvalidate  
imu9.h, 256

caerIMU9EventsIsValid  
imu9.h, 257

caerIMU9EventPacket  
imu9.h, 251

caerIMU9EventPacketAllocate  
imu9.h, 257

caerIMU9EventPacketGetEvent  
imu9.h, 257

caerIMU9EventPacketGetEventConst  
imu9.h, 258

caerIMU9EventSetAccelX  
imu9.h, 258

caerIMU9EventSetAccelY  
imu9.h, 258

caerIMU9EventSetAccelZ  
imu9.h, 259

caerIMU9EventSetCompX  
imu9.h, 259

caerIMU9EventSetCompY  
imu9.h, 259

caerIMU9EventSetCompZ  
imu9.h, 260

caerIMU9EventSetGyroX  
imu9.h, 260

caerIMU9EventSetGyroY  
imu9.h, 260

caerIMU9EventSetGyroZ  
imu9.h, 261

caerIMU9EventSetTemp  
imu9.h, [261](#)

caerIMU9EventSetTimestamp  
imu9.h, [261](#)

caerIMU9EventValidate  
imu9.h, [262](#)

caerIntegerToByteArray  
libcaer.h, [378](#)

caerLog  
log.h, [381](#)

caerLogFileDescriptorsGetFirst  
log.h, [382](#)

caerLogFileDescriptorsGetSecond  
log.h, [382](#)

caerLogFileDescriptorsSet  
log.h, [382](#)

caerLogLevelGet  
log.h, [383](#)

caerLogLevelSet  
log.h, [383](#)

caerLogVAFull  
log.h, [384](#)

caerLogVA  
log.h, [383](#)

caerPoint1DEvent  
point1d.h, [277](#)

caerPoint1DEventGetScale  
point1d.h, [277](#)

caerPoint1DEventGetTimestamp  
point1d.h, [278](#)

caerPoint1DEventGetTimestamp64  
point1d.h, [278](#)

caerPoint1DEventGetType  
point1d.h, [278](#)

caerPoint1DEventGetX  
point1d.h, [279](#)

caerPoint1DEventInvalidate  
point1d.h, [279](#)

caerPoint1DEventIsValid  
point1d.h, [280](#)

caerPoint1DEventPacket  
point1d.h, [277](#)

caerPoint1DEventPacketAllocate  
point1d.h, [280](#)

caerPoint1DEventPacketGetEvent  
point1d.h, [280](#)

caerPoint1DEventPacketGetEventConst  
point1d.h, [281](#)

caerPoint1DEventSetScale  
point1d.h, [281](#)

caerPoint1DEventSetTimestamp  
point1d.h, [281](#)

caerPoint1DEventSetType  
point1d.h, [282](#)

caerPoint1DEventSetX  
point1d.h, [282](#)

caerPoint1DEventValidate  
point1d.h, [282](#)

caerPoint2DEvent  
point2d.h, [289](#)

caerPoint2DEventGetScale  
point2d.h, [289](#)

caerPoint2DEventGetTimestamp  
point2d.h, [290](#)

caerPoint2DEventGetTimestamp64  
point2d.h, [290](#)

caerPoint2DEventGetType  
point2d.h, [290](#)

caerPoint2DEventGetX  
point2d.h, [291](#)

caerPoint2DEventGetY  
point2d.h, [291](#)

caerPoint2DEventInvalidate  
point2d.h, [292](#)

caerPoint2DEventIsValid  
point2d.h, [292](#)

caerPoint2DEventPacket  
point2d.h, [289](#)

caerPoint2DEventPacketAllocate  
point2d.h, [292](#)

caerPoint2DEventPacketGetEvent  
point2d.h, [293](#)

caerPoint2DEventPacketGetEventConst  
point2d.h, [293](#)

caerPoint2DEventSetScale  
point2d.h, [294](#)

caerPoint2DEventSetTimestamp  
point2d.h, [294](#)

caerPoint2DEventSetType  
point2d.h, [294](#)

caerPoint2DEventSetX  
point2d.h, [295](#)

caerPoint2DEventSetY  
point2d.h, [295](#)

caerPoint2DEventValidate  
point2d.h, [295](#)

caerPoint3DEvent  
point3d.h, [302](#)

caerPoint3DEventGetScale  
point3d.h, [302](#)

caerPoint3DEventGetTimestamp  
point3d.h, [302](#)

caerPoint3DEventGetTimestamp64  
point3d.h, [303](#)

caerPoint3DEventGetType  
point3d.h, [303](#)

caerPoint3DEventGetX  
point3d.h, [303](#)

caerPoint3DEventGetY  
point3d.h, [304](#)

caerPoint3DEventGetZ  
point3d.h, [304](#)

caerPoint3DEventInvalidate  
point3d.h, [305](#)

caerPoint3DEventIsValid  
point3d.h, [305](#)

caerPoint3DEventPacket  
point3d.h, [302](#)

caerPoint3DEventPacketAllocate  
point3d.h, [305](#)

caerPoint3DEventPacketGetEvent  
point3d.h, [306](#)

caerPoint3DEventPacketGetEventConst  
point3d.h, [306](#)

caerPoint3DEventSetScale  
point3d.h, [306](#)

caerPoint3DEventSetTimestamp  
point3d.h, [307](#)

caerPoint3DEventSetType  
point3d.h, [307](#)

caerPoint3DEventSetX  
point3d.h, [307](#)

caerPoint3DEventSetY  
point3d.h, [308](#)

caerPoint3DEventSetZ  
point3d.h, [308](#)

caerPoint3DEventValidate  
point3d.h, [308](#)

caerPoint4DEvent  
point4d.h, [315](#)

caerPoint4DEventGetScale  
point4d.h, [315](#)

caerPoint4DEventGetTimestamp  
point4d.h, [315](#)

caerPoint4DEventGetTimestamp64  
point4d.h, [316](#)

caerPoint4DEventGetType  
point4d.h, [316](#)

caerPoint4DEventGetW  
point4d.h, [316](#)

caerPoint4DEventGetX  
point4d.h, [317](#)

caerPoint4DEventGetY  
point4d.h, [317](#)

caerPoint4DEventGetZ  
point4d.h, [318](#)

caerPoint4DEventInvalidate  
point4d.h, [318](#)

caerPoint4DEventIsValid  
point4d.h, [318](#)

caerPoint4DEventPacket  
point4d.h, [315](#)

caerPoint4DEventPacketAllocate  
point4d.h, [319](#)

caerPoint4DEventPacketGetEvent  
point4d.h, [319](#)

caerPoint4DEventPacketGetEventConst  
point4d.h, [319](#)

caerPoint4DEventSetScale  
point4d.h, [320](#)

caerPoint4DEventSetTimestamp  
point4d.h, [320](#)

caerPoint4DEventSetType  
point4d.h, [320](#)

caerPoint4DEventSetW  
point4d.h, [321](#)

caerPoint4DEventSetX  
point4d.h, [321](#)

caerPoint4DEventSetY  
point4d.h, [321](#)

caerPoint4DEventSetZ  
point4d.h, [322](#)

caerPoint4DEventValidate  
point4d.h, [322](#)

caerPolarityEvent  
polarity.h, [328](#)

caerPolarityEventGetPolarity  
polarity.h, [329](#)

caerPolarityEventGetTimestamp  
polarity.h, [329](#)

caerPolarityEventGetTimestamp64  
polarity.h, [330](#)

caerPolarityEventGetX  
polarity.h, [330](#)

caerPolarityEventGetY  
polarity.h, [330](#)

caerPolarityEventInvalidate  
polarity.h, [331](#)

caerPolarityEventIsValid  
polarity.h, [331](#)

caerPolarityEventPacket  
polarity.h, [329](#)

caerPolarityEventPacketAllocate  
polarity.h, [331](#)

caerPolarityEventPacketGetEvent  
polarity.h, [332](#)

caerPolarityEventPacketGetEventConst  
polarity.h, [332](#)

caerPolarityEventSetPolarity  
polarity.h, [333](#)

caerPolarityEventSetTimestamp  
polarity.h, [333](#)

caerPolarityEventSetX  
polarity.h, [333](#)

caerPolarityEventSetY  
polarity.h, [334](#)

caerPolarityEventValidate  
polarity.h, [334](#)

caerSampleEvent  
sample.h, [340](#)

caerSampleEventGetSample  
sample.h, [341](#)

caerSampleEventGetTimestamp  
sample.h, [341](#)

caerSampleEventGetTimestamp64  
sample.h, [341](#)

caerSampleEventGetType  
sample.h, [342](#)

caerSampleEventInvalidate  
sample.h, [342](#)

caerSampleEventIsValid  
sample.h, [342](#)

- caerSampleEventPacket
  - sample.h, [340](#)
- caerSampleEventPacketAllocate
  - sample.h, [343](#)
- caerSampleEventPacketGetEvent
  - sample.h, [343](#)
- caerSampleEventPacketGetEventConst
  - sample.h, [344](#)
- caerSampleEventSetSample
  - sample.h, [344](#)
- caerSampleEventSetTimestamp
  - sample.h, [344](#)
- caerSampleEventSetType
  - sample.h, [345](#)
- caerSampleEventValidate
  - sample.h, [345](#)
- caerSpecialEvent
  - special.h, [352](#)
- caerSpecialEventGetData
  - special.h, [353](#)
- caerSpecialEventGetTimestamp
  - special.h, [353](#)
- caerSpecialEventGetTimestamp64
  - special.h, [354](#)
- caerSpecialEventGetType
  - special.h, [354](#)
- caerSpecialEventInvalidate
  - special.h, [354](#)
- caerSpecialEventIsValid
  - special.h, [355](#)
- caerSpecialEventPacket
  - special.h, [352](#)
- caerSpecialEventPacketAllocate
  - special.h, [355](#)
- caerSpecialEventPacketFindEventByType
  - special.h, [356](#)
- caerSpecialEventPacketFindEventByTypeConst
  - special.h, [356](#)
- caerSpecialEventPacketFindValidEventByType
  - special.h, [356](#)
- caerSpecialEventPacketFindValidEventByTypeConst
  - special.h, [357](#)
- caerSpecialEventPacketGetEvent
  - special.h, [357](#)
- caerSpecialEventPacketGetEventConst
  - special.h, [358](#)
- caerSpecialEventSetData
  - special.h, [358](#)
- caerSpecialEventSetTimestamp
  - special.h, [358](#)
- caerSpecialEventSetType
  - special.h, [359](#)
- caerSpecialEventValidate
  - special.h, [359](#)
- caerSpikeEvent
  - spike.h, [365](#)
- caerSpikeEventGetChipID
  - spike.h, [366](#)
- caerSpikeEventGetNeuronID
  - spike.h, [366](#)
- caerSpikeEventGetSourceCoreID
  - spike.h, [367](#)
- caerSpikeEventGetTimestamp
  - spike.h, [367](#)
- caerSpikeEventGetTimestamp64
  - spike.h, [367](#)
- caerSpikeEventInvalidate
  - spike.h, [368](#)
- caerSpikeEventIsValid
  - spike.h, [368](#)
- caerSpikeEventPacket
  - spike.h, [366](#)
- caerSpikeEventPacketAllocate
  - spike.h, [368](#)
- caerSpikeEventPacketGetEvent
  - spike.h, [369](#)
- caerSpikeEventPacketGetEventConst
  - spike.h, [369](#)
- caerSpikeEventSetChipID
  - spike.h, [370](#)
- caerSpikeEventSetNeuronID
  - spike.h, [370](#)
- caerSpikeEventSetSourceCoreID
  - spike.h, [370](#)
- caerSpikeEventSetTimestamp
  - spike.h, [371](#)
- caerSpikeEventValidate
  - spike.h, [371](#)
- caerStrEquals
  - libcaer.h, [378](#)
- caerStrEqualsUpTo
  - libcaer.h, [380](#)
- common.h
  - CAER\_DEFAULT\_EVENT\_TYPES\_COUNT, [166](#)
  - CAER\_EVENT\_PACKET\_HEADER\_SIZE, [166](#)
  - CAER\_ITERATOR\_ALL\_END, [166](#)
  - CAER\_ITERATOR\_ALL\_START, [167](#)
  - CAER\_ITERATOR\_VALID\_END, [167](#)
  - CAER\_ITERATOR\_VALID\_START, [167](#)
  - caer\_default\_event\_types, [168](#)
  - caerEventPacketAppend, [169](#)
  - caerEventPacketClean, [169](#)
  - caerEventPacketClear, [170](#)
  - caerEventPacketCopy, [170](#)
  - caerEventPacketCopyOnlyEvents, [170](#)
  - caerEventPacketCopyOnlyValidEvents, [171](#)
  - caerEventPacketEquals, [171](#)
  - caerEventPacketGetDataSize, [171](#)
  - caerEventPacketGetSize, [172](#)
  - caerEventPacketGrow, [172](#)
  - caerEventPacketHeader, [168](#)
  - caerEventPacketHeaderGetEventCapacity, [172](#)
  - caerEventPacketHeaderGetEventNumber, [173](#)
  - caerEventPacketHeaderGetEventSize, [173](#)
  - caerEventPacketHeaderGetEventSource, [174](#)
  - caerEventPacketHeaderGetEventTSOffset, [174](#)

- caerEventPacketHeaderGetEventTSOverflow, 174
- caerEventPacketHeaderGetEventType, 175
- caerEventPacketHeaderGetEventValid, 175
- caerEventPacketHeaderSetEventCapacity, 175
- caerEventPacketHeaderSetEventNumber, 176
- caerEventPacketHeaderSetEventSize, 176
- caerEventPacketHeaderSetEventSource, 176
- caerEventPacketHeaderSetEventTSOffset, 177
- caerEventPacketHeaderSetEventTSOverflow, 177
- caerEventPacketHeaderSetEventType, 177
- caerEventPacketHeaderSetEventValid, 178
- caerEventPacketResize, 178
- caerGenericEventCopy, 179
- caerGenericEventGetEvent, 179
- caerGenericEventGetTimestamp, 179
- caerGenericEventGetTimestamp64, 180
- caerGenericEventsValid, 180
- PACKED\_STRUCT, 181
- TS\_OVERFLOW\_SHIFT, 167
- VALID\_MARK\_MASK, 168
- VALID\_MARK\_SHIFT, 168
- config.h
  - CAER\_CONFIGURATION\_CONST\_ITERATOR↔  
\_ALL\_START, 182
  - CAER\_CONFIGURATION\_CONST\_ITERATOR↔  
\_VALID\_START, 183
  - CAER\_CONFIGURATION\_CONST\_REVERSE↔  
\_ITERATOR\_ALL\_START, 183
  - CAER\_CONFIGURATION\_CONST\_REVERSE↔  
\_ITERATOR\_VALID\_START, 183
  - CAER\_CONFIGURATION\_ITERATOR\_ALL\_E↔  
ND, 184
  - CAER\_CONFIGURATION\_ITERATOR\_ALL\_ST↔  
ART, 184
  - CAER\_CONFIGURATION\_ITERATOR\_VALID↔  
END, 184
  - CAER\_CONFIGURATION\_ITERATOR\_VALID↔  
START, 184
  - CAER\_CONFIGURATION\_REVERSE\_ITERAT↔  
OR\_ALL\_END, 185
  - CAER\_CONFIGURATION\_REVERSE\_ITERAT↔  
OR\_ALL\_START, 185
  - CAER\_CONFIGURATION\_REVERSE\_ITERAT↔  
OR\_VALID\_END, 185
  - CAER\_CONFIGURATION\_REVERSE\_ITERAT↔  
OR\_VALID\_START, 185
  - CONFIG\_MODULE\_ADDR\_MASK, 186
  - CONFIG\_MODULE\_ADDR\_SHIFT, 186
  - caerConfigurationEvent, 186
  - caerConfigurationEventGetModuleAddress, 187
  - caerConfigurationEventGetParameter, 187
  - caerConfigurationEventGetParameterAddress, 187
  - caerConfigurationEventGetTimestamp, 188
  - caerConfigurationEventGetTimestamp64, 188
  - caerConfigurationEventInvalidate, 188
  - caerConfigurationEventsValid, 189
  - caerConfigurationEventPacket, 186
  - caerConfigurationEventPacketAllocate, 189
  - caerConfigurationEventPacketGetEvent, 189
  - caerConfigurationEventPacketGetEventConst, 190
  - caerConfigurationEventSetModuleAddress, 190
  - caerConfigurationEventSetParameter, 191
  - caerConfigurationEventSetParameterAddress, 191
  - caerConfigurationEventSetTimestamp, 191
  - caerConfigurationEventValidate, 192
  - PACKED\_STRUCT, 192
  - DAVIS128\_CONFIG\_BIAS\_ADCCOMPBP  
davis.h, 21
  - DAVIS128\_CONFIG\_BIAS\_ADCREFHIGH  
davis.h, 21
  - DAVIS128\_CONFIG\_BIAS\_ADCREFLOW  
davis.h, 22
  - DAVIS128\_CONFIG\_BIAS\_AEPDBN  
davis.h, 22
  - DAVIS128\_CONFIG\_BIAS\_AEPUXBP  
davis.h, 22
  - DAVIS128\_CONFIG\_BIAS\_AEPUYBP  
davis.h, 23
  - DAVIS128\_CONFIG\_BIAS\_APSCAS  
davis.h, 23
  - DAVIS128\_CONFIG\_BIAS\_APSEVERFLOWLEVEL  
davis.h, 23
  - DAVIS128\_CONFIG\_BIAS\_APSROSFBN  
davis.h, 24
  - DAVIS128\_CONFIG\_BIAS\_BIASBUFFER  
davis.h, 24
  - DAVIS128\_CONFIG\_BIAS\_COLSELLOWBN  
davis.h, 24
  - DAVIS128\_CONFIG\_BIAS\_DACBUFBP  
davis.h, 25
  - DAVIS128\_CONFIG\_BIAS\_DIFFBN  
davis.h, 25
  - DAVIS128\_CONFIG\_BIAS\_IFREFRBN  
davis.h, 25
  - DAVIS128\_CONFIG\_BIAS\_IFTHRBN  
davis.h, 26
  - DAVIS128\_CONFIG\_BIAS\_LCOLTIMEOUTBN  
davis.h, 26
  - DAVIS128\_CONFIG\_BIAS\_LOCALBUFBN  
davis.h, 26
  - DAVIS128\_CONFIG\_BIAS\_OFFBN  
davis.h, 27
  - DAVIS128\_CONFIG\_BIAS\_ONBN  
davis.h, 27
  - DAVIS128\_CONFIG\_BIAS\_PADFOLLBN  
davis.h, 27
  - DAVIS128\_CONFIG\_BIAS\_PIXINVBN  
davis.h, 28
  - DAVIS128\_CONFIG\_BIAS\_PRBP  
davis.h, 28
  - DAVIS128\_CONFIG\_BIAS\_PRSFBP  
davis.h, 28
  - DAVIS128\_CONFIG\_BIAS\_READOUTBUFBP  
davis.h, 29
  - DAVIS128\_CONFIG\_BIAS\_REFRBP  
davis.h, 29

- DAVIS128\_CONFIG\_BIAS\_SSN  
davis.h, [29](#)
- DAVIS128\_CONFIG\_BIAS\_SSP  
davis.h, [30](#)
- DAVIS128\_CONFIG\_CHIP\_AERNAROW  
davis.h, [30](#)
- DAVIS128\_CONFIG\_CHIP\_ANALOGMUX0  
davis.h, [30](#)
- DAVIS128\_CONFIG\_CHIP\_ANALOGMUX1  
davis.h, [30](#)
- DAVIS128\_CONFIG\_CHIP\_ANALOGMUX2  
davis.h, [31](#)
- DAVIS128\_CONFIG\_CHIP\_BIASMUX0  
davis.h, [31](#)
- DAVIS128\_CONFIG\_CHIP\_DIGITALMUX0  
davis.h, [31](#)
- DAVIS128\_CONFIG\_CHIP\_DIGITALMUX1  
davis.h, [31](#)
- DAVIS128\_CONFIG\_CHIP\_DIGITALMUX2  
davis.h, [31](#)
- DAVIS128\_CONFIG\_CHIP\_DIGITALMUX3  
davis.h, [31](#)
- DAVIS128\_CONFIG\_CHIP\_GLOBAL\_SHUTTER  
davis.h, [32](#)
- DAVIS128\_CONFIG\_CHIP\_RESETCALIBNEURON  
davis.h, [32](#)
- DAVIS128\_CONFIG\_CHIP\_RESETTESTPIXEL  
davis.h, [32](#)
- DAVIS128\_CONFIG\_CHIP\_SELECTGRAYCOUNTER  
davis.h, [32](#)
- DAVIS128\_CONFIG\_CHIP\_TYPCALIBNEURON  
davis.h, [32](#)
- DAVIS128\_CONFIG\_CHIP\_USEAOUT  
davis.h, [32](#)
- DAVIS208\_CONFIG\_BIAS\_ADCCOMPBP  
davis.h, [33](#)
- DAVIS208\_CONFIG\_BIAS\_ADCREFHIGH  
davis.h, [33](#)
- DAVIS208\_CONFIG\_BIAS\_ADCREFLOW  
davis.h, [33](#)
- DAVIS208\_CONFIG\_BIAS\_AEPDBN  
davis.h, [33](#)
- DAVIS208\_CONFIG\_BIAS\_AEPUXBP  
davis.h, [34](#)
- DAVIS208\_CONFIG\_BIAS\_AEPUYBP  
davis.h, [34](#)
- DAVIS208\_CONFIG\_BIAS\_APSCAS  
davis.h, [34](#)
- DAVIS208\_CONFIG\_BIAS\_APSOEVERFLOWLEVEL  
davis.h, [35](#)
- DAVIS208\_CONFIG\_BIAS\_APSROSFBN  
davis.h, [35](#)
- DAVIS208\_CONFIG\_BIAS\_BIASBUFFER  
davis.h, [35](#)
- DAVIS208\_CONFIG\_BIAS\_COLSELLOWBN  
davis.h, [36](#)
- DAVIS208\_CONFIG\_BIAS\_DACBUFBP  
davis.h, [36](#)
- DAVIS208\_CONFIG\_BIAS\_DIFFBN  
davis.h, [36](#)
- DAVIS208\_CONFIG\_BIAS\_IFREFRBN  
davis.h, [37](#)
- DAVIS208\_CONFIG\_BIAS\_IFTHRBN  
davis.h, [37](#)
- DAVIS208\_CONFIG\_BIAS\_LCOLTIMEOUTBN  
davis.h, [37](#)
- DAVIS208\_CONFIG\_BIAS\_LOCALBUFBN  
davis.h, [38](#)
- DAVIS208\_CONFIG\_BIAS\_OFFBN  
davis.h, [38](#)
- DAVIS208\_CONFIG\_BIAS\_ONBN  
davis.h, [38](#)
- DAVIS208\_CONFIG\_BIAS\_PADFOLLBN  
davis.h, [39](#)
- DAVIS208\_CONFIG\_BIAS\_PIXINVBN  
davis.h, [39](#)
- DAVIS208\_CONFIG\_BIAS\_PRBP  
davis.h, [39](#)
- DAVIS208\_CONFIG\_BIAS\_PRSFBP  
davis.h, [40](#)
- DAVIS208\_CONFIG\_BIAS\_READOUTBUFBP  
davis.h, [40](#)
- DAVIS208\_CONFIG\_BIAS\_REFRBP  
davis.h, [40](#)
- DAVIS208\_CONFIG\_BIAS\_REFSSBN  
davis.h, [41](#)
- DAVIS208\_CONFIG\_BIAS\_REFSS  
davis.h, [41](#)
- DAVIS208\_CONFIG\_BIAS\_REGBIASBP  
davis.h, [41](#)
- DAVIS208\_CONFIG\_BIAS\_RESETHIGHPASS  
davis.h, [42](#)
- DAVIS208\_CONFIG\_BIAS\_SSN  
davis.h, [42](#)
- DAVIS208\_CONFIG\_BIAS\_SSP  
davis.h, [42](#)
- DAVIS208\_CONFIG\_CHIP\_AERNAROW  
davis.h, [43](#)
- DAVIS208\_CONFIG\_CHIP\_ANALOGMUX0  
davis.h, [43](#)
- DAVIS208\_CONFIG\_CHIP\_ANALOGMUX1  
davis.h, [43](#)
- DAVIS208\_CONFIG\_CHIP\_ANALOGMUX2  
davis.h, [43](#)
- DAVIS208\_CONFIG\_CHIP\_BIASMUX0  
davis.h, [43](#)
- DAVIS208\_CONFIG\_CHIP\_DIGITALMUX0  
davis.h, [44](#)
- DAVIS208\_CONFIG\_CHIP\_DIGITALMUX1  
davis.h, [44](#)
- DAVIS208\_CONFIG\_CHIP\_DIGITALMUX2  
davis.h, [44](#)
- DAVIS208\_CONFIG\_CHIP\_DIGITALMUX3  
davis.h, [44](#)
- DAVIS208\_CONFIG\_CHIP\_GLOBAL\_SHUTTER  
davis.h, [44](#)



- DAVIS208\_CONFIG\_CHIP\_RESETCALIBNEURON  
davis.h, [44](#)
- DAVIS208\_CONFIG\_CHIP\_RESETTESTPIXEL  
davis.h, [45](#)
- DAVIS208\_CONFIG\_CHIP\_SELECTBIASREFSS  
davis.h, [45](#)
- DAVIS208\_CONFIG\_CHIP\_SELECTGRAYCOUNTER  
davis.h, [45](#)
- DAVIS208\_CONFIG\_CHIP\_SELECTHIGHPASS  
davis.h, [45](#)
- DAVIS208\_CONFIG\_CHIP\_SELECTPOSFB  
davis.h, [45](#)
- DAVIS208\_CONFIG\_CHIP\_SELECTPREAMPAVG  
davis.h, [45](#)
- DAVIS208\_CONFIG\_CHIP\_SELECTSENSE  
davis.h, [46](#)
- DAVIS208\_CONFIG\_CHIP\_TYPENCALIBNEURON  
davis.h, [46](#)
- DAVIS208\_CONFIG\_CHIP\_USEAOUT  
davis.h, [46](#)
- DAVIS240\_CONFIG\_BIAS\_AEPDBN  
davis.h, [46](#)
- DAVIS240\_CONFIG\_BIAS\_AEPUXBP  
davis.h, [46](#)
- DAVIS240\_CONFIG\_BIAS\_AEPUYBP  
davis.h, [47](#)
- DAVIS240\_CONFIG\_BIAS\_APSCASEPC  
davis.h, [47](#)
- DAVIS240\_CONFIG\_BIAS\_APSOEVERFLOWLEVELBN  
davis.h, [47](#)
- DAVIS240\_CONFIG\_BIAS\_APSROSFBN  
davis.h, [47](#)
- DAVIS240\_CONFIG\_BIAS\_BIASBUFFER  
davis.h, [48](#)
- DAVIS240\_CONFIG\_BIAS\_DIFFBN  
davis.h, [48](#)
- DAVIS240\_CONFIG\_BIAS\_DIFFCASBNC  
davis.h, [48](#)
- DAVIS240\_CONFIG\_BIAS\_IFREFRBN  
davis.h, [48](#)
- DAVIS240\_CONFIG\_BIAS\_IFTHRBN  
davis.h, [49](#)
- DAVIS240\_CONFIG\_BIAS\_LCOLTIMEOUTBN  
davis.h, [49](#)
- DAVIS240\_CONFIG\_BIAS\_LOCALBUFBN  
davis.h, [49](#)
- DAVIS240\_CONFIG\_BIAS\_OFFBN  
davis.h, [49](#)
- DAVIS240\_CONFIG\_BIAS\_ONBN  
davis.h, [50](#)
- DAVIS240\_CONFIG\_BIAS\_PADFOLLBN  
davis.h, [50](#)
- DAVIS240\_CONFIG\_BIAS\_PIXINVBN  
davis.h, [50](#)
- DAVIS240\_CONFIG\_BIAS\_PRBP  
davis.h, [50](#)
- DAVIS240\_CONFIG\_BIAS\_PRSFBN  
davis.h, [51](#)
- DAVIS240\_CONFIG\_BIAS\_REFRBP  
davis.h, [51](#)
- DAVIS240\_CONFIG\_BIAS\_SSN  
davis.h, [51](#)
- DAVIS240\_CONFIG\_BIAS\_SSP  
davis.h, [51](#)
- DAVIS240\_CONFIG\_CHIP\_AERNAROW  
davis.h, [52](#)
- DAVIS240\_CONFIG\_CHIP\_ANALOGMUX0  
davis.h, [52](#)
- DAVIS240\_CONFIG\_CHIP\_ANALOGMUX1  
davis.h, [52](#)
- DAVIS240\_CONFIG\_CHIP\_ANALOGMUX2  
davis.h, [52](#)
- DAVIS240\_CONFIG\_CHIP\_BIASMUX0  
davis.h, [52](#)
- DAVIS240\_CONFIG\_CHIP\_DIGITALMUX0  
davis.h, [53](#)
- DAVIS240\_CONFIG\_CHIP\_DIGITALMUX1  
davis.h, [53](#)
- DAVIS240\_CONFIG\_CHIP\_DIGITALMUX2  
davis.h, [53](#)
- DAVIS240\_CONFIG\_CHIP\_DIGITALMUX3  
davis.h, [53](#)
- DAVIS240\_CONFIG\_CHIP\_GLOBAL\_SHUTTER  
davis.h, [53](#)
- DAVIS240\_CONFIG\_CHIP\_RESETCALIBNEURON  
davis.h, [53](#)
- DAVIS240\_CONFIG\_CHIP\_RESETTESTPIXEL  
davis.h, [54](#)
- DAVIS240\_CONFIG\_CHIP\_SPECIALPIXELCONTROL  
davis.h, [54](#)
- DAVIS240\_CONFIG\_CHIP\_TYPENCALIBNEURON  
davis.h, [54](#)
- DAVIS240\_CONFIG\_CHIP\_USEAOUT  
davis.h, [54](#)
- DAVIS346\_CONFIG\_BIAS\_ADCCOMPBP  
davis.h, [54](#)
- DAVIS346\_CONFIG\_BIAS\_ADCREFHIGH  
davis.h, [55](#)
- DAVIS346\_CONFIG\_BIAS\_ADCREFLOW  
davis.h, [55](#)
- DAVIS346\_CONFIG\_BIAS\_ADCTESTVOLTAGE  
davis.h, [55](#)
- DAVIS346\_CONFIG\_BIAS\_AEPDBN  
davis.h, [56](#)
- DAVIS346\_CONFIG\_BIAS\_AEPUXBP  
davis.h, [56](#)
- DAVIS346\_CONFIG\_BIAS\_AEPUYBP  
davis.h, [56](#)
- DAVIS346\_CONFIG\_BIAS\_APSCAS  
davis.h, [57](#)
- DAVIS346\_CONFIG\_BIAS\_APSOEVERFLOWLEVEL  
davis.h, [57](#)
- DAVIS346\_CONFIG\_BIAS\_APSROSFBN  
davis.h, [57](#)
- DAVIS346\_CONFIG\_BIAS\_BIASBUFFER  
davis.h, [58](#)



- DAVIS346\_CONFIG\_BIAS\_COLSELLOWBN  
davis.h, [58](#)
- DAVIS346\_CONFIG\_BIAS\_DACBUFBP  
davis.h, [58](#)
- DAVIS346\_CONFIG\_BIAS\_DIFFBN  
davis.h, [59](#)
- DAVIS346\_CONFIG\_BIAS\_IFREFRBN  
davis.h, [59](#)
- DAVIS346\_CONFIG\_BIAS\_IFTHRBN  
davis.h, [59](#)
- DAVIS346\_CONFIG\_BIAS\_LCOLTIMEOUTBN  
davis.h, [60](#)
- DAVIS346\_CONFIG\_BIAS\_LOCALBUFBN  
davis.h, [60](#)
- DAVIS346\_CONFIG\_BIAS\_OFFBN  
davis.h, [60](#)
- DAVIS346\_CONFIG\_BIAS\_ONBN  
davis.h, [61](#)
- DAVIS346\_CONFIG\_BIAS\_PADFOLLBN  
davis.h, [61](#)
- DAVIS346\_CONFIG\_BIAS\_PIXINBN  
davis.h, [61](#)
- DAVIS346\_CONFIG\_BIAS\_PRBP  
davis.h, [62](#)
- DAVIS346\_CONFIG\_BIAS\_PRSFBP  
davis.h, [62](#)
- DAVIS346\_CONFIG\_BIAS\_READOUTBUFBP  
davis.h, [62](#)
- DAVIS346\_CONFIG\_BIAS\_REFRBP  
davis.h, [63](#)
- DAVIS346\_CONFIG\_BIAS\_SSN  
davis.h, [63](#)
- DAVIS346\_CONFIG\_BIAS\_SSP  
davis.h, [63](#)
- DAVIS346\_CONFIG\_CHIP\_AERNAROW  
davis.h, [64](#)
- DAVIS346\_CONFIG\_CHIP\_ANALOGMUX0  
davis.h, [64](#)
- DAVIS346\_CONFIG\_CHIP\_ANALOGMUX1  
davis.h, [64](#)
- DAVIS346\_CONFIG\_CHIP\_ANALOGMUX2  
davis.h, [64](#)
- DAVIS346\_CONFIG\_CHIP\_BIASMUX0  
davis.h, [64](#)
- DAVIS346\_CONFIG\_CHIP\_DIGITALMUX0  
davis.h, [65](#)
- DAVIS346\_CONFIG\_CHIP\_DIGITALMUX1  
davis.h, [65](#)
- DAVIS346\_CONFIG\_CHIP\_DIGITALMUX2  
davis.h, [65](#)
- DAVIS346\_CONFIG\_CHIP\_DIGITALMUX3  
davis.h, [65](#)
- DAVIS346\_CONFIG\_CHIP\_GLOBAL\_SHUTTER  
davis.h, [65](#)
- DAVIS346\_CONFIG\_CHIP\_RESETCALIBNEURON  
davis.h, [65](#)
- DAVIS346\_CONFIG\_CHIP\_RESETTESTPIXEL  
davis.h, [66](#)
- DAVIS346\_CONFIG\_CHIP\_SELECTGRAYCOUNTER  
davis.h, [66](#)
- DAVIS346\_CONFIG\_CHIP\_TESTADC  
davis.h, [66](#)
- DAVIS346\_CONFIG\_CHIP\_TYPENCALIBNEURON  
davis.h, [66](#)
- DAVIS346\_CONFIG\_CHIP\_USEAOUT  
davis.h, [66](#)
- DAVIS640\_CONFIG\_BIAS\_ADCCOMPBP  
davis.h, [66](#)
- DAVIS640\_CONFIG\_BIAS\_ADCREFHIGH  
davis.h, [67](#)
- DAVIS640\_CONFIG\_BIAS\_ADCREFLOW  
davis.h, [67](#)
- DAVIS640\_CONFIG\_BIAS\_ADCTESTVOLTAGE  
davis.h, [67](#)
- DAVIS640\_CONFIG\_BIAS\_AEPDBN  
davis.h, [68](#)
- DAVIS640\_CONFIG\_BIAS\_AEPUXBP  
davis.h, [68](#)
- DAVIS640\_CONFIG\_BIAS\_AEPUYBP  
davis.h, [68](#)
- DAVIS640\_CONFIG\_BIAS\_APSCAS  
davis.h, [69](#)
- DAVIS640\_CONFIG\_BIAS\_APSEVERFLOWLEVEL  
davis.h, [69](#)
- DAVIS640\_CONFIG\_BIAS\_APSROSFBN  
davis.h, [69](#)
- DAVIS640\_CONFIG\_BIAS\_BIASBUFFER  
davis.h, [70](#)
- DAVIS640\_CONFIG\_BIAS\_COLSELLOWBN  
davis.h, [70](#)
- DAVIS640\_CONFIG\_BIAS\_DACBUFBP  
davis.h, [70](#)
- DAVIS640\_CONFIG\_BIAS\_DIFFBN  
davis.h, [71](#)
- DAVIS640\_CONFIG\_BIAS\_IFREFRBN  
davis.h, [71](#)
- DAVIS640\_CONFIG\_BIAS\_IFTHRBN  
davis.h, [71](#)
- DAVIS640\_CONFIG\_BIAS\_LCOLTIMEOUTBN  
davis.h, [72](#)
- DAVIS640\_CONFIG\_BIAS\_LOCALBUFBN  
davis.h, [72](#)
- DAVIS640\_CONFIG\_BIAS\_OFFBN  
davis.h, [72](#)
- DAVIS640\_CONFIG\_BIAS\_ONBN  
davis.h, [73](#)
- DAVIS640\_CONFIG\_BIAS\_PADFOLLBN  
davis.h, [73](#)
- DAVIS640\_CONFIG\_BIAS\_PIXINBN  
davis.h, [73](#)
- DAVIS640\_CONFIG\_BIAS\_PRBP  
davis.h, [74](#)
- DAVIS640\_CONFIG\_BIAS\_PRSFBP  
davis.h, [74](#)
- DAVIS640\_CONFIG\_BIAS\_READOUTBUFBP  
davis.h, [74](#)

- DAVIS640\_CONFIG\_BIAS\_REFRBP  
davis.h, [75](#)
- DAVIS640\_CONFIG\_BIAS\_SSN  
davis.h, [75](#)
- DAVIS640\_CONFIG\_BIAS\_SSP  
davis.h, [75](#)
- DAVIS640\_CONFIG\_CHIP\_AERNAROW  
davis.h, [76](#)
- DAVIS640\_CONFIG\_CHIP\_ANALOGMUX0  
davis.h, [76](#)
- DAVIS640\_CONFIG\_CHIP\_ANALOGMUX1  
davis.h, [76](#)
- DAVIS640\_CONFIG\_CHIP\_ANALOGMUX2  
davis.h, [76](#)
- DAVIS640\_CONFIG\_CHIP\_BIASMUX0  
davis.h, [76](#)
- DAVIS640\_CONFIG\_CHIP\_DIGITALMUX0  
davis.h, [77](#)
- DAVIS640\_CONFIG\_CHIP\_DIGITALMUX1  
davis.h, [77](#)
- DAVIS640\_CONFIG\_CHIP\_DIGITALMUX2  
davis.h, [77](#)
- DAVIS640\_CONFIG\_CHIP\_DIGITALMUX3  
davis.h, [77](#)
- DAVIS640\_CONFIG\_CHIP\_GLOBAL\_SHUTTER  
davis.h, [77](#)
- DAVIS640\_CONFIG\_CHIP\_RESETCALIBNEURON  
davis.h, [77](#)
- DAVIS640\_CONFIG\_CHIP\_RESETTESTPIXEL  
davis.h, [78](#)
- DAVIS640\_CONFIG\_CHIP\_SELECTGRAYCOUNTER  
davis.h, [78](#)
- DAVIS640\_CONFIG\_CHIP\_TESTADC  
davis.h, [78](#)
- DAVIS640\_CONFIG\_CHIP\_TYPCALIBNEURON  
davis.h, [78](#)
- DAVIS640\_CONFIG\_CHIP\_USEAOUT  
davis.h, [78](#)
- DAVIS\_CHIP\_DAVIS128  
davis.h, [78](#)
- DAVIS\_CHIP\_DAVIS208  
davis.h, [79](#)
- DAVIS\_CHIP\_DAVIS240A  
davis.h, [79](#)
- DAVIS\_CHIP\_DAVIS240B  
davis.h, [79](#)
- DAVIS\_CHIP\_DAVIS240C  
davis.h, [79](#)
- DAVIS\_CHIP\_DAVIS346A  
davis.h, [79](#)
- DAVIS\_CHIP\_DAVIS346B  
davis.h, [79](#)
- DAVIS\_CHIP\_DAVIS346C  
davis.h, [79](#)
- DAVIS\_CHIP\_DAVIS640  
davis.h, [79](#)
- DAVIS\_CHIP\_DAVISRGB  
davis.h, [80](#)
- DAVIS\_CONFIG\_APS\_ADC\_TEST\_MODE  
davis.h, [80](#)
- DAVIS\_CONFIG\_APS\_AUTOEXPOSURE  
davis.h, [80](#)
- DAVIS\_CONFIG\_APS\_COLOR\_FILTER  
davis.h, [80](#)
- DAVIS\_CONFIG\_APS\_COLUMN\_SETTLE  
davis.h, [80](#)
- DAVIS\_CONFIG\_APS\_END\_COLUMN\_0  
davis.h, [80](#)
- DAVIS\_CONFIG\_APS\_END\_COLUMN\_1  
davis.h, [81](#)
- DAVIS\_CONFIG\_APS\_END\_COLUMN\_2  
davis.h, [81](#)
- DAVIS\_CONFIG\_APS\_END\_COLUMN\_3  
davis.h, [81](#)
- DAVIS\_CONFIG\_APS\_END\_ROW\_0  
davis.h, [81](#)
- DAVIS\_CONFIG\_APS\_END\_ROW\_1  
davis.h, [81](#)
- DAVIS\_CONFIG\_APS\_END\_ROW\_2  
davis.h, [81](#)
- DAVIS\_CONFIG\_APS\_END\_ROW\_3  
davis.h, [81](#)
- DAVIS\_CONFIG\_APS\_EXPOSURE  
davis.h, [82](#)
- DAVIS\_CONFIG\_APS\_FRAME\_DELAY  
davis.h, [82](#)
- DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER  
davis.h, [82](#)
- DAVIS\_CONFIG\_APS\_HAS\_EXTERNAL\_ADC  
davis.h, [82](#)
- DAVIS\_CONFIG\_APS\_HAS\_GLOBAL\_SHUTTER  
davis.h, [82](#)
- DAVIS\_CONFIG\_APS\_HAS\_INTERNAL\_ADC  
davis.h, [82](#)
- DAVIS\_CONFIG\_APS\_HAS\_QUAD\_ROI  
davis.h, [83](#)
- DAVIS\_CONFIG\_APS\_NULL\_SETTLE  
davis.h, [83](#)
- DAVIS\_CONFIG\_APS\_ORIENTATION\_INFO  
davis.h, [83](#)
- DAVIS\_CONFIG\_APS\_RAMP\_RESET  
davis.h, [83](#)
- DAVIS\_CONFIG\_APS\_RAMP\_SHORT\_RESET  
davis.h, [83](#)
- DAVIS\_CONFIG\_APS\_RESET\_READ  
davis.h, [83](#)
- DAVIS\_CONFIG\_APS\_RESET\_SETTLE  
davis.h, [84](#)
- DAVIS\_CONFIG\_APS\_ROW\_SETTLE  
davis.h, [84](#)
- DAVIS\_CONFIG\_APS\_RUN  
davis.h, [84](#)
- DAVIS\_CONFIG\_APS\_SAMPLE\_ENABLE  
davis.h, [84](#)
- DAVIS\_CONFIG\_APS\_SAMPLE\_SETTLE  
davis.h, [84](#)

- DAVIS\_CONFIG\_APS\_SIZE\_COLUMNS  
davis.h, [84](#)
- DAVIS\_CONFIG\_APS\_SIZE\_ROWS  
davis.h, [84](#)
- DAVIS\_CONFIG\_APS\_SNAPSHOT  
davis.h, [85](#)
- DAVIS\_CONFIG\_APS\_START\_COLUMN\_0  
davis.h, [85](#)
- DAVIS\_CONFIG\_APS\_START\_COLUMN\_1  
davis.h, [85](#)
- DAVIS\_CONFIG\_APS\_START\_COLUMN\_2  
davis.h, [85](#)
- DAVIS\_CONFIG\_APS\_START\_COLUMN\_3  
davis.h, [85](#)
- DAVIS\_CONFIG\_APS\_START\_ROW\_0  
davis.h, [85](#)
- DAVIS\_CONFIG\_APS\_START\_ROW\_1  
davis.h, [86](#)
- DAVIS\_CONFIG\_APS\_START\_ROW\_2  
davis.h, [86](#)
- DAVIS\_CONFIG\_APS\_START\_ROW\_3  
davis.h, [86](#)
- DAVIS\_CONFIG\_APS\_USE\_INTERNAL\_ADC  
davis.h, [86](#)
- DAVIS\_CONFIG\_APS\_WAIT\_ON\_TRANSFER\_STALL  
davis.h, [86](#)
- DAVIS\_CONFIG\_APS  
davis.h, [80](#)
- DAVIS\_CONFIG\_BIAS  
davis.h, [86](#)
- DAVIS\_CONFIG\_CHIP  
davis.h, [87](#)
- DAVIS\_CONFIG\_DVS\_ACK\_DELAY\_COLUMN  
davis.h, [87](#)
- DAVIS\_CONFIG\_DVS\_ACK\_DELAY\_ROW  
davis.h, [87](#)
- DAVIS\_CONFIG\_DVS\_ACK\_EXTENSION\_COLUMN  
davis.h, [87](#)
- DAVIS\_CONFIG\_DVS\_ACK\_EXTENSION\_ROW  
davis.h, [87](#)
- DAVIS\_CONFIG\_DVS\_EXTERNAL\_AER\_CONTROL  
davis.h, [87](#)
- DAVIS\_CONFIG\_DVS\_FILTER\_BACKGROUND\_ACTIVITY\_DELTA  
davis.h, [88](#)
- DAVIS\_CONFIG\_DVS\_FILTER\_BACKGROUND\_ACTIVITY  
davis.h, [88](#)
- DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_0\_COLUMN  
davis.h, [88](#)
- DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_0\_ROW  
davis.h, [88](#)
- DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_1\_COLUMN  
davis.h, [88](#)
- DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_1\_ROW  
davis.h, [88](#)
- DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_2\_COLUMN  
davis.h, [89](#)
- DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_2\_ROW  
davis.h, [89](#)
- DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_3\_COLUMN  
davis.h, [89](#)
- DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_3\_ROW  
davis.h, [89](#)
- DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_4\_COLUMN  
davis.h, [89](#)
- DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_4\_ROW  
davis.h, [89](#)
- DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_5\_COLUMN  
davis.h, [89](#)
- DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_5\_ROW  
davis.h, [90](#)
- DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_6\_COLUMN  
davis.h, [90](#)
- DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_6\_ROW  
davis.h, [90](#)
- DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_7\_COLUMN  
davis.h, [90](#)
- DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_7\_ROW  
davis.h, [90](#)
- DAVIS\_CONFIG\_DVS\_FILTER\_ROW\_ONLY\_EVENT\_GENERATOR  
davis.h, [90](#)
- DAVIS\_CONFIG\_DVS\_HAS\_BACKGROUND\_ACTIVITY\_FILTER  
davis.h, [90](#)
- DAVIS\_CONFIG\_DVS\_HAS\_PIXEL\_FILTER  
davis.h, [91](#)
- DAVIS\_CONFIG\_DVS\_HAS\_TEST\_EVENT\_GENERATOR  
davis.h, [91](#)
- DAVIS\_CONFIG\_DVS\_ORIENTATION\_INFO  
davis.h, [91](#)
- DAVIS\_CONFIG\_DVS\_RUN  
davis.h, [91](#)
- DAVIS\_CONFIG\_DVS\_SIZE\_COLUMNS  
davis.h, [91](#)
- DAVIS\_CONFIG\_DVS\_SIZE\_ROWS  
davis.h, [91](#)
- DAVIS\_CONFIG\_DVS\_TEST\_EVENT\_GENERATOR\_ENABLE  
davis.h, [92](#)
- DAVIS\_CONFIG\_DVS\_WAIT\_ON\_TRANSFER\_STALL  
davis.h, [92](#)
- DAVIS\_CONFIG\_DVS  
davis.h, [87](#)
- DAVIS\_CONFIG\_EXTINPUT\_DETECT\_FALLING\_EDGE1  
davis.h, [92](#)
- DAVIS\_CONFIG\_EXTINPUT\_DETECT\_FALLING\_EDGE2  
davis.h, [92](#)
- DAVIS\_CONFIG\_EXTINPUT\_DETECT\_FALLING\_EDGE  
davis.h, [92](#)

- DAVIS\_CONFIG\_EXTINPUT\_DETECT\_PULSE\_LENGTH1  
davis.h, 93
- DAVIS\_CONFIG\_EXTINPUT\_DETECT\_PULSE\_LENGTH2  
davis.h, 93
- DAVIS\_CONFIG\_EXTINPUT\_DETECT\_PULSE\_LENGTH  
davis.h, 93
- DAVIS\_CONFIG\_EXTINPUT\_DETECT\_PULSE\_POLARITY1  
davis.h, 93
- DAVIS\_CONFIG\_EXTINPUT\_DETECT\_PULSE\_POLARITY2  
davis.h, 93
- DAVIS\_CONFIG\_EXTINPUT\_DETECT\_PULSE\_POLARITY  
davis.h, 93
- DAVIS\_CONFIG\_EXTINPUT\_DETECT\_PULSES1  
davis.h, 94
- DAVIS\_CONFIG\_EXTINPUT\_DETECT\_PULSES2  
davis.h, 94
- DAVIS\_CONFIG\_EXTINPUT\_DETECT\_PULSES  
davis.h, 94
- DAVIS\_CONFIG\_EXTINPUT\_DETECT\_RISING\_EDGE1  
davis.h, 94
- DAVIS\_CONFIG\_EXTINPUT\_DETECT\_RISING\_EDGE2  
davis.h, 94
- DAVIS\_CONFIG\_EXTINPUT\_DETECT\_RISING\_EDGE  
davis.h, 94
- DAVIS\_CONFIG\_EXTINPUT\_GENERATE\_INJECT\_ON\_FALLING\_EDGE  
davis.h, 95
- DAVIS\_CONFIG\_EXTINPUT\_GENERATE\_INJECT\_ON\_RISING\_EDGE  
davis.h, 95
- DAVIS\_CONFIG\_EXTINPUT\_GENERATE\_PULSE\_INTERVAL  
davis.h, 95
- DAVIS\_CONFIG\_EXTINPUT\_GENERATE\_PULSE\_LENGTH  
davis.h, 95
- DAVIS\_CONFIG\_EXTINPUT\_GENERATE\_PULSE\_POLARITY  
davis.h, 95
- DAVIS\_CONFIG\_EXTINPUT\_GENERATE\_USE\_CUSTOM\_SIGNAL  
davis.h, 95
- DAVIS\_CONFIG\_EXTINPUT\_HAS\_EXTRA\_DETECTORS  
davis.h, 96
- DAVIS\_CONFIG\_EXTINPUT\_HAS\_GENERATOR  
davis.h, 96
- DAVIS\_CONFIG\_EXTINPUT\_RUN\_DETECTOR1  
davis.h, 96
- DAVIS\_CONFIG\_EXTINPUT\_RUN\_DETECTOR2  
davis.h, 96
- DAVIS\_CONFIG\_EXTINPUT\_RUN\_DETECTOR  
davis.h, 96
- DAVIS\_CONFIG\_EXTINPUT\_RUN\_GENERATOR  
davis.h, 96
- DAVIS\_CONFIG\_EXTINPUT  
davis.h, 92
- DAVIS\_CONFIG\_IMU\_ACCEL\_FULL\_SCALE  
davis.h, 97
- DAVIS\_CONFIG\_IMU\_ACCEL\_STANDBY  
davis.h, 97
- DAVIS\_CONFIG\_IMU\_DIGITAL\_LOW\_PASS\_FILTER  
davis.h, 97
- DAVIS\_CONFIG\_IMU\_GYRO\_FULL\_SCALE  
davis.h, 97
- DAVIS\_CONFIG\_IMU\_GYRO\_STANDBY  
davis.h, 97
- DAVIS\_CONFIG\_IMU\_LP\_CYCLE  
davis.h, 98
- DAVIS\_CONFIG\_IMU\_LP\_WAKEUP  
davis.h, 98
- DAVIS\_CONFIG\_IMU\_ORIENTATION\_INFO  
davis.h, 98
- DAVIS\_CONFIG\_IMU\_RUN  
davis.h, 98
- DAVIS\_CONFIG\_IMU\_SAMPLE\_RATE\_DIVIDER  
davis.h, 98
- DAVIS\_CONFIG\_IMU\_TEMP\_STANDBY  
davis.h, 98
- DAVIS\_CONFIG\_IMU  
davis.h, 97
- DAVIS\_CONFIG\_MICROPHONE\_RUN  
davis.h, 99
- DAVIS\_CONFIG\_MICROPHONE\_SAMPLE\_FREQUENCY  
davis.h, 99
- DAVIS\_CONFIG\_MICROPHONE  
davis.h, 99
- DAVIS\_CONFIG\_MUX\_DROP\_APS\_ON\_TRANSFER\_STALL  
davis.h, 99
- DAVIS\_CONFIG\_MUX\_DROP\_DVS\_ON\_TRANSFER\_STALL  
davis.h, 99
- DAVIS\_CONFIG\_MUX\_DROP\_EXTINPUT\_ON\_TRANSFER\_STALL  
davis.h, 100
- DAVIS\_CONFIG\_MUX\_DROP\_IMU\_ON\_TRANSFER\_STALL  
davis.h, 100
- DAVIS\_CONFIG\_MUX\_DROP\_MIC\_ON\_TRANSFER\_STALL  
davis.h, 100
- DAVIS\_CONFIG\_MUX\_FORCE\_CHIP\_BIAS\_ENABLE  
davis.h, 100
- DAVIS\_CONFIG\_MUX\_RUN  
davis.h, 100

DAVIS\_CONFIG\_MUX\_TIMESTAMP\_RESET  
     davis.h, [100](#)  
 DAVIS\_CONFIG\_MUX\_TIMESTAMP\_RUN  
     davis.h, [101](#)  
 DAVIS\_CONFIG\_MUX  
     davis.h, [99](#)  
 DAVIS\_CONFIG\_SYSINFO\_ADC\_CLOCK  
     davis.h, [101](#)  
 DAVIS\_CONFIG\_SYSINFO\_CHIP\_IDENTIFIER  
     davis.h, [101](#)  
 DAVIS\_CONFIG\_SYSINFO\_DEVICE\_IS\_MASTER  
     davis.h, [101](#)  
 DAVIS\_CONFIG\_SYSINFO\_LOGIC\_CLOCK  
     davis.h, [101](#)  
 DAVIS\_CONFIG\_SYSINFO\_LOGIC\_VERSION  
     davis.h, [102](#)  
 DAVIS\_CONFIG\_SYSINFO  
     davis.h, [101](#)  
 DAVIS\_CONFIG\_USB\_EARLY\_PACKET\_DELAY  
     davis.h, [102](#)  
 DAVIS\_CONFIG\_USB\_RUN  
     davis.h, [102](#)  
 DAVIS\_CONFIG\_USB  
     davis.h, [102](#)  
 DAVISRGB\_CONFIG\_APS\_GSFDRESET  
     davis.h, [102](#)  
 DAVISRGB\_CONFIG\_APS\_GSPDRESET  
     davis.h, [102](#)  
 DAVISRGB\_CONFIG\_APS\_GSRESETFALL  
     davis.h, [103](#)  
 DAVISRGB\_CONFIG\_APS\_GSTXFALL  
     davis.h, [103](#)  
 DAVISRGB\_CONFIG\_APS\_RSFDSETTLE  
     davis.h, [103](#)  
 DAVISRGB\_CONFIG\_APS\_TRANSFER  
     davis.h, [103](#)  
 DAVISRGB\_CONFIG\_BIAS\_ADCCOMPBP  
     davis.h, [103](#)  
 DAVISRGB\_CONFIG\_BIAS\_ADCREFHIGH  
     davis.h, [103](#)  
 DAVISRGB\_CONFIG\_BIAS\_ADCREFLOW  
     davis.h, [104](#)  
 DAVISRGB\_CONFIG\_BIAS\_ADCTESTVOLTAGE  
     davis.h, [104](#)  
 DAVISRGB\_CONFIG\_BIAS\_AEPDBN  
     davis.h, [104](#)  
 DAVISRGB\_CONFIG\_BIAS\_AEPUXBP  
     davis.h, [105](#)  
 DAVISRGB\_CONFIG\_BIAS\_AEPUYBP  
     davis.h, [105](#)  
 DAVISRGB\_CONFIG\_BIAS\_APSCAS  
     davis.h, [105](#)  
 DAVISRGB\_CONFIG\_BIAS\_APSROSFBN  
     davis.h, [106](#)  
 DAVISRGB\_CONFIG\_BIAS\_ARRAYBIASBUFFERBN  
     davis.h, [106](#)  
 DAVISRGB\_CONFIG\_BIAS\_ARRAYLOGICBUFFERBN  
     davis.h, [106](#)  
 DAVISRGB\_CONFIG\_BIAS\_BIASBUFFER  
     davis.h, [107](#)  
 DAVISRGB\_CONFIG\_BIAS\_DACBUFBP  
     davis.h, [107](#)  
 DAVISRGB\_CONFIG\_BIAS\_DIFFBN  
     davis.h, [107](#)  
 DAVISRGB\_CONFIG\_BIAS\_FALLTIMEBN  
     davis.h, [108](#)  
 DAVISRGB\_CONFIG\_BIAS\_GND07  
     davis.h, [108](#)  
 DAVISRGB\_CONFIG\_BIAS\_IFREFRBN  
     davis.h, [108](#)  
 DAVISRGB\_CONFIG\_BIAS\_IFTHRBN  
     davis.h, [109](#)  
 DAVISRGB\_CONFIG\_BIAS\_LCOLTIMEOUTBN  
     davis.h, [109](#)  
 DAVISRGB\_CONFIG\_BIAS\_LOCALBUFBN  
     davis.h, [109](#)  
 DAVISRGB\_CONFIG\_BIAS\_OFFBN  
     davis.h, [110](#)  
 DAVISRGB\_CONFIG\_BIAS\_ONBN  
     davis.h, [110](#)  
 DAVISRGB\_CONFIG\_BIAS\_OVG1LO  
     davis.h, [110](#)  
 DAVISRGB\_CONFIG\_BIAS\_OVG2LO  
     davis.h, [111](#)  
 DAVISRGB\_CONFIG\_BIAS\_PADFOLLBN  
     davis.h, [111](#)  
 DAVISRGB\_CONFIG\_BIAS\_PIXINVBN  
     davis.h, [111](#)  
 DAVISRGB\_CONFIG\_BIAS\_PRBP  
     davis.h, [112](#)  
 DAVISRGB\_CONFIG\_BIAS\_PRSFBP  
     davis.h, [112](#)  
 DAVISRGB\_CONFIG\_BIAS\_READOUTBUFBP  
     davis.h, [112](#)  
 DAVISRGB\_CONFIG\_BIAS\_REFRBP  
     davis.h, [113](#)  
 DAVISRGB\_CONFIG\_BIAS\_RISETIMEBP  
     davis.h, [113](#)  
 DAVISRGB\_CONFIG\_BIAS\_SSN  
     davis.h, [113](#)  
 DAVISRGB\_CONFIG\_BIAS\_SSP  
     davis.h, [114](#)  
 DAVISRGB\_CONFIG\_BIAS\_TX2OVG2HI  
     davis.h, [114](#)  
 DAVISRGB\_CONFIG\_CHIP\_ADJUSTOVG1LO  
     davis.h, [114](#)  
 DAVISRGB\_CONFIG\_CHIP\_ADJUSTOVG2LO  
     davis.h, [115](#)  
 DAVISRGB\_CONFIG\_CHIP\_ADJUSTTX2OVG2HI  
     davis.h, [115](#)  
 DAVISRGB\_CONFIG\_CHIP\_AERNAROW  
     davis.h, [115](#)  
 DAVISRGB\_CONFIG\_CHIP\_ANALOGMUX0  
     davis.h, [115](#)  
 DAVISRGB\_CONFIG\_CHIP\_ANALOGMUX1

- davis.h, [115](#)
- DAVISRGB\_CONFIG\_CHIP\_ANALOGMUX2
  - davis.h, [115](#)
- DAVISRGB\_CONFIG\_CHIP\_BIASMUX0
  - davis.h, [116](#)
- DAVISRGB\_CONFIG\_CHIP\_DIGITALMUX0
  - davis.h, [116](#)
- DAVISRGB\_CONFIG\_CHIP\_DIGITALMUX1
  - davis.h, [116](#)
- DAVISRGB\_CONFIG\_CHIP\_DIGITALMUX2
  - davis.h, [116](#)
- DAVISRGB\_CONFIG\_CHIP\_DIGITALMUX3
  - davis.h, [116](#)
- DAVISRGB\_CONFIG\_CHIP\_RESETCALIBNEURON
  - davis.h, [116](#)
- DAVISRGB\_CONFIG\_CHIP\_RESETTESTPIXEL
  - davis.h, [117](#)
- DAVISRGB\_CONFIG\_CHIP\_SELECTGRAYCOUNT↔
  - ER
    - davis.h, [117](#)
- DAVISRGB\_CONFIG\_CHIP\_TESTADC
  - davis.h, [117](#)
- DAVISRGB\_CONFIG\_CHIP\_TYPENCALIBNEURON
  - davis.h, [117](#)
- DAVISRGB\_CONFIG\_CHIP\_USEAOUT
  - davis.h, [117](#)
- DVS128\_CONFIG\_BIAS\_CAS
  - dvs128.h, [131](#)
- DVS128\_CONFIG\_BIAS\_DIFFOFF
  - dvs128.h, [131](#)
- DVS128\_CONFIG\_BIAS\_DIFFON
  - dvs128.h, [131](#)
- DVS128\_CONFIG\_BIAS\_DIFF
  - dvs128.h, [131](#)
- DVS128\_CONFIG\_BIAS\_FOLL
  - dvs128.h, [131](#)
- DVS128\_CONFIG\_BIAS\_INJGND
  - dvs128.h, [132](#)
- DVS128\_CONFIG\_BIAS\_PUX
  - dvs128.h, [132](#)
- DVS128\_CONFIG\_BIAS\_PUY
  - dvs128.h, [132](#)
- DVS128\_CONFIG\_BIAS\_PR
  - dvs128.h, [132](#)
- DVS128\_CONFIG\_BIAS\_REFR
  - dvs128.h, [132](#)
- DVS128\_CONFIG\_BIAS\_REQPD
  - dvs128.h, [132](#)
- DVS128\_CONFIG\_BIAS\_REQ
  - dvs128.h, [132](#)
- DVS128\_CONFIG\_BIAS
  - dvs128.h, [131](#)
- DVS128\_CONFIG\_DVS\_ARRAY\_RESET
  - dvs128.h, [133](#)
- DVS128\_CONFIG\_DVS\_RUN
  - dvs128.h, [133](#)
- DVS128\_CONFIG\_DVS\_TIMESTAMP\_RESET
  - dvs128.h, [133](#)
- DVS128\_CONFIG\_DVS\_TS\_MASTER
  - dvs128.h, [133](#)
- DVS128\_CONFIG\_DVS
  - dvs128.h, [133](#)
- DYNAPSE\_CHIP\_DYNAPSE
  - dynapse.h, [139](#)
- DYNAPSE\_CONFIG\_AER\_ACK\_DELAY
  - dynapse.h, [139](#)
- DYNAPSE\_CONFIG\_AER\_ACK\_EXTENSION
  - dynapse.h, [139](#)
- DYNAPSE\_CONFIG\_AER\_EXTERNAL\_AER\_CONT↔
  - ROL
    - dynapse.h, [139](#)
- DYNAPSE\_CONFIG\_AER\_RUN
  - dynapse.h, [139](#)
- DYNAPSE\_CONFIG\_AER\_WAIT\_ON\_TRANSFER↔
  - STALL
    - dynapse.h, [140](#)
- DYNAPSE\_CONFIG\_AER
  - dynapse.h, [139](#)
- DYNAPSE\_CONFIG\_BIAS\_C0\_PULSE\_PWLK\_P
  - dynapse.h, [140](#)
- DYNAPSE\_CONFIG\_CHIP\_CONTENT
  - dynapse.h, [140](#)
- DYNAPSE\_CONFIG\_CHIP\_ID
  - dynapse.h, [140](#)
- DYNAPSE\_CONFIG\_CHIP\_REQ\_DELAY
  - dynapse.h, [140](#)
- DYNAPSE\_CONFIG\_CHIP\_REQ\_EXTENSION
  - dynapse.h, [141](#)
- DYNAPSE\_CONFIG\_CHIP\_RUN
  - dynapse.h, [141](#)
- DYNAPSE\_CONFIG\_CHIP
  - dynapse.h, [140](#)
- DYNAPSE\_CONFIG\_CLEAR\_CAM
  - dynapse.h, [141](#)
- DYNAPSE\_CONFIG\_DEFAULT\_SRAM\_EMPTY
  - dynapse.h, [141](#)
- DYNAPSE\_CONFIG\_DEFAULT\_SRAM
  - dynapse.h, [141](#)
- DYNAPSE\_CONFIG\_MONITOR\_NEU
  - dynapse.h, [141](#)
- DYNAPSE\_CONFIG\_MUX\_DROP\_AER\_ON\_TRAN↔
  - SFER\_STALL
    - dynapse.h, [142](#)
- DYNAPSE\_CONFIG\_MUX\_FORCE\_CHIP\_BIAS\_E↔
  - NABLE
    - dynapse.h, [142](#)
- DYNAPSE\_CONFIG\_MUX\_RUN
  - dynapse.h, [142](#)
- DYNAPSE\_CONFIG\_MUX\_TIMESTAMP\_RESET
  - dynapse.h, [142](#)
- DYNAPSE\_CONFIG\_MUX\_TIMESTAMP\_RUN
  - dynapse.h, [142](#)
- DYNAPSE\_CONFIG\_MUX
  - dynapse.h, [141](#)
- DYNAPSE\_CONFIG\_POISSONSPIKEGEN\_CHIPID
  - dynapse.h, [143](#)



- DYNAPSE\_CONFIG\_POISSONSPIKEGEN\_RUN
  - dynapse.h, [143](#)
- DYNAPSE\_CONFIG\_POISSONSPIKEGEN\_WRITE↔
  - ADDRESS
    - dynapse.h, [143](#)
  - DATA
    - dynapse.h, [143](#)
- DYNAPSE\_CONFIG\_POISSONSPIKEGEN
  - dynapse.h, [142](#)
- DYNAPSE\_CONFIG\_SPIKEGEN\_BASEADDR
  - dynapse.h, [143](#)
- DYNAPSE\_CONFIG\_SPIKEGEN\_ISIBASE
  - dynapse.h, [144](#)
- DYNAPSE\_CONFIG\_SPIKEGEN\_ISI
  - dynapse.h, [143](#)
- DYNAPSE\_CONFIG\_SPIKEGEN\_REPEAT
  - dynapse.h, [144](#)
- DYNAPSE\_CONFIG\_SPIKEGEN\_RUN
  - dynapse.h, [144](#)
- DYNAPSE\_CONFIG\_SPIKEGEN\_STIMCOUNT
  - dynapse.h, [144](#)
- DYNAPSE\_CONFIG\_SPIKEGEN\_VARMODE
  - dynapse.h, [144](#)
- DYNAPSE\_CONFIG\_SPIKEGEN
  - dynapse.h, [143](#)
- DYNAPSE\_CONFIG\_SRAM\_ADDRESS
  - dynapse.h, [144](#)
- DYNAPSE\_CONFIG\_SRAM\_BURSTMODE
  - dynapse.h, [145](#)
- DYNAPSE\_CONFIG\_SRAM\_DIRECTION\_POS
  - dynapse.h, [145](#)
- DYNAPSE\_CONFIG\_SRAM\_READDATA
  - dynapse.h, [145](#)
- DYNAPSE\_CONFIG\_SRAM\_READ
  - dynapse.h, [145](#)
- DYNAPSE\_CONFIG\_SRAM\_RWCOMMAND
  - dynapse.h, [145](#)
- DYNAPSE\_CONFIG\_SRAM\_WRITEDATA
  - dynapse.h, [146](#)
- DYNAPSE\_CONFIG\_SRAM\_WRITE
  - dynapse.h, [145](#)
- DYNAPSE\_CONFIG\_SRAM
  - dynapse.h, [144](#)
- DYNAPSE\_CONFIG\_SYNAPSERECONFIG\_CHIPS↔
  - ELECT
    - dynapse.h, [146](#)
- DYNAPSE\_CONFIG\_SYNAPSERECONFIG\_GLOB↔
  - ALKERNEL
    - dynapse.h, [146](#)
- DYNAPSE\_CONFIG\_SYNAPSERECONFIG\_RUN
  - dynapse.h, [146](#)
- DYNAPSE\_CONFIG\_SYNAPSERECONFIG\_SRAM↔
  - BASEADDR
    - dynapse.h, [146](#)
- DYNAPSE\_CONFIG\_SYNAPSERECONFIG\_USES↔
  - RAMKERNELS
    - dynapse.h, [147](#)
- DYNAPSE\_CONFIG\_SYNAPSERECONFIG
  - dynapse.h, [146](#)
- DYNAPSE\_CONFIG\_SYSINFO\_CHIP\_IDENTIFIER
  - dynapse.h, [147](#)
- DYNAPSE\_CONFIG\_SYSINFO\_DEVICE\_IS\_MASTER
  - dynapse.h, [147](#)
- DYNAPSE\_CONFIG\_SYSINFO\_LOGIC\_CLOCK
  - dynapse.h, [147](#)
- DYNAPSE\_CONFIG\_SYSINFO\_LOGIC\_VERSION
  - dynapse.h, [147](#)
- DYNAPSE\_CONFIG\_SYSINFO
  - dynapse.h, [147](#)
- DYNAPSE\_CONFIG\_USB\_EARLY\_PACKET\_DELAY
  - dynapse.h, [148](#)
- DYNAPSE\_CONFIG\_USB\_RUN
  - dynapse.h, [148](#)
- DYNAPSE\_CONFIG\_USB
  - dynapse.h, [148](#)
- DYNAPSE\_X4BOARD\_COREX
  - dynapse.h, [148](#)
- DYNAPSE\_X4BOARD\_COREY
  - dynapse.h, [148](#)
- DYNAPSE\_X4BOARD\_NEUX
  - dynapse.h, [148](#)
- DYNAPSE\_X4BOARD\_NEUY
  - dynapse.h, [148](#)
- DYNAPSE\_X4BOARD\_NUMCHIPS
  - dynapse.h, [149](#)
- davis.h
  - CAER\_DEVICE\_DAVIS\_FX2, [21](#)
  - CAER\_DEVICE\_DAVIS\_FX3, [21](#)
  - CAER\_DEVICE\_DAVIS, [21](#)
  - caer\_bias\_shiftedsourcesource\_operating\_mode, [119](#)
  - caer\_bias\_shiftedsourcesource\_voltage\_level, [120](#)
  - caerBiasCoarseFineGenerate, [120](#)
  - caerBiasCoarseFineParse, [120](#)
  - caerBiasShiftedSourceGenerate, [122](#)
  - caerBiasShiftedSourceParse, [122](#)
  - caerBiasVDACGenerate, [122](#)
  - caerBiasVDACParse, [123](#)
  - caerDavisInfoGet, [123](#)
  - DAVIS128\_CONFIG\_BIAS\_ADCCOMPBP, [21](#)
  - DAVIS128\_CONFIG\_BIAS\_ADCREFHIGH, [21](#)
  - DAVIS128\_CONFIG\_BIAS\_ADCREFLOW, [22](#)
  - DAVIS128\_CONFIG\_BIAS\_AEPDBN, [22](#)
  - DAVIS128\_CONFIG\_BIAS\_AEPUXBP, [22](#)
  - DAVIS128\_CONFIG\_BIAS\_AEPUYBP, [23](#)
  - DAVIS128\_CONFIG\_BIAS\_APSCAS, [23](#)
  - DAVIS128\_CONFIG\_BIAS\_APSEOVERFLOWLE↔
    - VEL, [23](#)
  - DAVIS128\_CONFIG\_BIAS\_APSROSFBP, [24](#)
  - DAVIS128\_CONFIG\_BIAS\_BIASBUFFER, [24](#)
  - DAVIS128\_CONFIG\_BIAS\_COLSELOWBN, [24](#)
  - DAVIS128\_CONFIG\_BIAS\_DACBUFBP, [25](#)
  - DAVIS128\_CONFIG\_BIAS\_DIFFBN, [25](#)
  - DAVIS128\_CONFIG\_BIAS\_IFREFRBN, [25](#)
  - DAVIS128\_CONFIG\_BIAS\_IFTHRBN, [26](#)

- DAVIS128\_CONFIG\_BIAS\_LCOLTIMEOUTBN, 26
- DAVIS128\_CONFIG\_BIAS\_LOCALBUFBN, 26
- DAVIS128\_CONFIG\_BIAS\_OFFBN, 27
- DAVIS128\_CONFIG\_BIAS\_ONBN, 27
- DAVIS128\_CONFIG\_BIAS\_PADFOLLBN, 27
- DAVIS128\_CONFIG\_BIAS\_PIXINBN, 28
- DAVIS128\_CONFIG\_BIAS\_PRBP, 28
- DAVIS128\_CONFIG\_BIAS\_PRSFBN, 28
- DAVIS128\_CONFIG\_BIAS\_READOUTBUFBP, 29
- DAVIS128\_CONFIG\_BIAS\_REFRBP, 29
- DAVIS128\_CONFIG\_BIAS\_SSN, 29
- DAVIS128\_CONFIG\_BIAS\_SSP, 30
- DAVIS128\_CONFIG\_CHIP\_AERNAROW, 30
- DAVIS128\_CONFIG\_CHIP\_ANALOGMUX0, 30
- DAVIS128\_CONFIG\_CHIP\_ANALOGMUX1, 30
- DAVIS128\_CONFIG\_CHIP\_ANALOGMUX2, 31
- DAVIS128\_CONFIG\_CHIP\_BIASMUX0, 31
- DAVIS128\_CONFIG\_CHIP\_DIGITALMUX0, 31
- DAVIS128\_CONFIG\_CHIP\_DIGITALMUX1, 31
- DAVIS128\_CONFIG\_CHIP\_DIGITALMUX2, 31
- DAVIS128\_CONFIG\_CHIP\_DIGITALMUX3, 31
- DAVIS128\_CONFIG\_CHIP\_GLOBAL\_SHUTTER, 32
- DAVIS128\_CONFIG\_CHIP\_RESETCALIBNEU↵RON, 32
- DAVIS128\_CONFIG\_CHIP\_RESETTESTPIXEL, 32
- DAVIS128\_CONFIG\_CHIP\_SELECTGRAYCO↵UNTER, 32
- DAVIS128\_CONFIG\_CHIP\_TYPENCALIBNEU↵RON, 32
- DAVIS128\_CONFIG\_CHIP\_USEAOUT, 32
- DAVIS208\_CONFIG\_BIAS\_ADCCOMPBP, 33
- DAVIS208\_CONFIG\_BIAS\_ADCREFHIGH, 33
- DAVIS208\_CONFIG\_BIAS\_ADCREFLOW, 33
- DAVIS208\_CONFIG\_BIAS\_AEPDBN, 33
- DAVIS208\_CONFIG\_BIAS\_AEPUXBP, 34
- DAVIS208\_CONFIG\_BIAS\_AEPUYBP, 34
- DAVIS208\_CONFIG\_BIAS\_APSCAS, 34
- DAVIS208\_CONFIG\_BIAS\_APSOEVERFLOWLE↵VEL, 35
- DAVIS208\_CONFIG\_BIAS\_APSROSFBN, 35
- DAVIS208\_CONFIG\_BIAS\_BIASBUFFER, 35
- DAVIS208\_CONFIG\_BIAS\_COLSELOWBN, 36
- DAVIS208\_CONFIG\_BIAS\_DACBUFBP, 36
- DAVIS208\_CONFIG\_BIAS\_DIFFBN, 36
- DAVIS208\_CONFIG\_BIAS\_IFREFRBN, 37
- DAVIS208\_CONFIG\_BIAS\_IFTHRBN, 37
- DAVIS208\_CONFIG\_BIAS\_LCOLTIMEOUTBN, 37
- DAVIS208\_CONFIG\_BIAS\_LOCALBUFBN, 38
- DAVIS208\_CONFIG\_BIAS\_OFFBN, 38
- DAVIS208\_CONFIG\_BIAS\_ONBN, 38
- DAVIS208\_CONFIG\_BIAS\_PADFOLLBN, 39
- DAVIS208\_CONFIG\_BIAS\_PIXINBN, 39
- DAVIS208\_CONFIG\_BIAS\_PRBP, 39
- DAVIS208\_CONFIG\_BIAS\_PRSFBN, 40
- DAVIS208\_CONFIG\_BIAS\_READOUTBUFBP, 40
- DAVIS208\_CONFIG\_BIAS\_REFRBP, 40
- DAVIS208\_CONFIG\_BIAS\_REFSSBN, 41
- DAVIS208\_CONFIG\_BIAS\_REFSS, 41
- DAVIS208\_CONFIG\_BIAS\_REGBIASBP, 41
- DAVIS208\_CONFIG\_BIAS\_RESETHIGHPASS, 42
- DAVIS208\_CONFIG\_BIAS\_SSN, 42
- DAVIS208\_CONFIG\_BIAS\_SSP, 42
- DAVIS208\_CONFIG\_CHIP\_AERNAROW, 43
- DAVIS208\_CONFIG\_CHIP\_ANALOGMUX0, 43
- DAVIS208\_CONFIG\_CHIP\_ANALOGMUX1, 43
- DAVIS208\_CONFIG\_CHIP\_ANALOGMUX2, 43
- DAVIS208\_CONFIG\_CHIP\_BIASMUX0, 43
- DAVIS208\_CONFIG\_CHIP\_DIGITALMUX0, 44
- DAVIS208\_CONFIG\_CHIP\_DIGITALMUX1, 44
- DAVIS208\_CONFIG\_CHIP\_DIGITALMUX2, 44
- DAVIS208\_CONFIG\_CHIP\_DIGITALMUX3, 44
- DAVIS208\_CONFIG\_CHIP\_GLOBAL\_SHUTTER, 44
- DAVIS208\_CONFIG\_CHIP\_RESETCALIBNEU↵RON, 44
- DAVIS208\_CONFIG\_CHIP\_RESETTESTPIXEL, 45
- DAVIS208\_CONFIG\_CHIP\_SELECTBIASREFSS, 45
- DAVIS208\_CONFIG\_CHIP\_SELECTGRAYCO↵UNTER, 45
- DAVIS208\_CONFIG\_CHIP\_SELECTHIGHPASS, 45
- DAVIS208\_CONFIG\_CHIP\_SELECTPOSFB, 45
- DAVIS208\_CONFIG\_CHIP\_SELECTPREAMPA↵VG, 45
- DAVIS208\_CONFIG\_CHIP\_SELECTSENSE, 46
- DAVIS208\_CONFIG\_CHIP\_TYPENCALIBNEU↵RON, 46
- DAVIS208\_CONFIG\_CHIP\_USEAOUT, 46
- DAVIS240\_CONFIG\_BIAS\_AEPDBN, 46
- DAVIS240\_CONFIG\_BIAS\_AEPUXBP, 46
- DAVIS240\_CONFIG\_BIAS\_AEPUYBP, 47
- DAVIS240\_CONFIG\_BIAS\_APSCASEPC, 47
- DAVIS240\_CONFIG\_BIAS\_APSOEVERFLOWLE↵VELBN, 47
- DAVIS240\_CONFIG\_BIAS\_APSROSFBN, 47
- DAVIS240\_CONFIG\_BIAS\_BIASBUFFER, 48
- DAVIS240\_CONFIG\_BIAS\_DIFFBN, 48
- DAVIS240\_CONFIG\_BIAS\_DIFFCASBNC, 48
- DAVIS240\_CONFIG\_BIAS\_IFREFRBN, 48
- DAVIS240\_CONFIG\_BIAS\_IFTHRBN, 49
- DAVIS240\_CONFIG\_BIAS\_LCOLTIMEOUTBN, 49
- DAVIS240\_CONFIG\_BIAS\_LOCALBUFBN, 49
- DAVIS240\_CONFIG\_BIAS\_OFFBN, 49
- DAVIS240\_CONFIG\_BIAS\_ONBN, 50
- DAVIS240\_CONFIG\_BIAS\_PADFOLLBN, 50
- DAVIS240\_CONFIG\_BIAS\_PIXINBN, 50
- DAVIS240\_CONFIG\_BIAS\_PRBP, 50
- DAVIS240\_CONFIG\_BIAS\_PRSFBN, 51



- DAVIS240\_CONFIG\_BIAS\_REFRBP, [51](#)
- DAVIS240\_CONFIG\_BIAS\_SSN, [51](#)
- DAVIS240\_CONFIG\_BIAS\_SSP, [51](#)
- DAVIS240\_CONFIG\_CHIP\_AERNAROW, [52](#)
- DAVIS240\_CONFIG\_CHIP\_ANALOGMUX0, [52](#)
- DAVIS240\_CONFIG\_CHIP\_ANALOGMUX1, [52](#)
- DAVIS240\_CONFIG\_CHIP\_ANALOGMUX2, [52](#)
- DAVIS240\_CONFIG\_CHIP\_BIASMUX0, [52](#)
- DAVIS240\_CONFIG\_CHIP\_DIGITALMUX0, [53](#)
- DAVIS240\_CONFIG\_CHIP\_DIGITALMUX1, [53](#)
- DAVIS240\_CONFIG\_CHIP\_DIGITALMUX2, [53](#)
- DAVIS240\_CONFIG\_CHIP\_DIGITALMUX3, [53](#)
- DAVIS240\_CONFIG\_CHIP\_GLOBAL\_SHUTTER, [53](#)
- DAVIS240\_CONFIG\_CHIP\_RESETCALIBNEU↵RON, [53](#)
- DAVIS240\_CONFIG\_CHIP\_RESETTESTPIXEL, [54](#)
- DAVIS240\_CONFIG\_CHIP\_SPECIALPIXELCO↵NTROL, [54](#)
- DAVIS240\_CONFIG\_CHIP\_TYPENCALIBNEU↵RON, [54](#)
- DAVIS240\_CONFIG\_CHIP\_USEAOUT, [54](#)
- DAVIS346\_CONFIG\_BIAS\_ADCCOMPBP, [54](#)
- DAVIS346\_CONFIG\_BIAS\_ADCREFHIGH, [55](#)
- DAVIS346\_CONFIG\_BIAS\_ADCREFLOW, [55](#)
- DAVIS346\_CONFIG\_BIAS\_ADCTESTVOLTAGE, [55](#)
- DAVIS346\_CONFIG\_BIAS\_AEPDBN, [56](#)
- DAVIS346\_CONFIG\_BIAS\_AEPUXBP, [56](#)
- DAVIS346\_CONFIG\_BIAS\_AEPUYBP, [56](#)
- DAVIS346\_CONFIG\_BIAS\_APSCAS, [57](#)
- DAVIS346\_CONFIG\_BIAS\_APSOEVERFLOWLE↵VEL, [57](#)
- DAVIS346\_CONFIG\_BIAS\_APSROSFBN, [57](#)
- DAVIS346\_CONFIG\_BIAS\_BIASBUFFER, [58](#)
- DAVIS346\_CONFIG\_BIAS\_COLSEOLLOWBN, [58](#)
- DAVIS346\_CONFIG\_BIAS\_DACBUFBP, [58](#)
- DAVIS346\_CONFIG\_BIAS\_DIFFBN, [59](#)
- DAVIS346\_CONFIG\_BIAS\_IFREFRBN, [59](#)
- DAVIS346\_CONFIG\_BIAS\_IFTHRBN, [59](#)
- DAVIS346\_CONFIG\_BIAS\_LCOLTIMEOUTBN, [60](#)
- DAVIS346\_CONFIG\_BIAS\_LOCALBUFBN, [60](#)
- DAVIS346\_CONFIG\_BIAS\_OFFBN, [60](#)
- DAVIS346\_CONFIG\_BIAS\_ONBN, [61](#)
- DAVIS346\_CONFIG\_BIAS\_PADFOLLBN, [61](#)
- DAVIS346\_CONFIG\_BIAS\_PIXINVBN, [61](#)
- DAVIS346\_CONFIG\_BIAS\_PRBP, [62](#)
- DAVIS346\_CONFIG\_BIAS\_PRSFBP, [62](#)
- DAVIS346\_CONFIG\_BIAS\_READOUTBUFBP, [62](#)
- DAVIS346\_CONFIG\_BIAS\_REFRBP, [63](#)
- DAVIS346\_CONFIG\_BIAS\_SSN, [63](#)
- DAVIS346\_CONFIG\_BIAS\_SSP, [63](#)
- DAVIS346\_CONFIG\_CHIP\_AERNAROW, [64](#)
- DAVIS346\_CONFIG\_CHIP\_ANALOGMUX0, [64](#)
- DAVIS346\_CONFIG\_CHIP\_ANALOGMUX1, [64](#)
- DAVIS346\_CONFIG\_CHIP\_ANALOGMUX2, [64](#)
- DAVIS346\_CONFIG\_CHIP\_BIASMUX0, [64](#)
- DAVIS346\_CONFIG\_CHIP\_DIGITALMUX0, [65](#)
- DAVIS346\_CONFIG\_CHIP\_DIGITALMUX1, [65](#)
- DAVIS346\_CONFIG\_CHIP\_DIGITALMUX2, [65](#)
- DAVIS346\_CONFIG\_CHIP\_DIGITALMUX3, [65](#)
- DAVIS346\_CONFIG\_CHIP\_GLOBAL\_SHUTTER, [65](#)
- DAVIS346\_CONFIG\_CHIP\_RESETCALIBNEU↵RON, [65](#)
- DAVIS346\_CONFIG\_CHIP\_RESETTESTPIXEL, [66](#)
- DAVIS346\_CONFIG\_CHIP\_SELECTGRAYCO↵UNTER, [66](#)
- DAVIS346\_CONFIG\_CHIP\_TESTADC, [66](#)
- DAVIS346\_CONFIG\_CHIP\_TYPENCALIBNEU↵RON, [66](#)
- DAVIS346\_CONFIG\_CHIP\_USEAOUT, [66](#)
- DAVIS640\_CONFIG\_BIAS\_ADCCOMPBP, [66](#)
- DAVIS640\_CONFIG\_BIAS\_ADCREFHIGH, [67](#)
- DAVIS640\_CONFIG\_BIAS\_ADCREFLOW, [67](#)
- DAVIS640\_CONFIG\_BIAS\_ADCTESTVOLTAGE, [67](#)
- DAVIS640\_CONFIG\_BIAS\_AEPDBN, [68](#)
- DAVIS640\_CONFIG\_BIAS\_AEPUXBP, [68](#)
- DAVIS640\_CONFIG\_BIAS\_AEPUYBP, [68](#)
- DAVIS640\_CONFIG\_BIAS\_APSCAS, [69](#)
- DAVIS640\_CONFIG\_BIAS\_APSOEVERFLOWLE↵VEL, [69](#)
- DAVIS640\_CONFIG\_BIAS\_APSROSFBN, [69](#)
- DAVIS640\_CONFIG\_BIAS\_BIASBUFFER, [70](#)
- DAVIS640\_CONFIG\_BIAS\_COLSEOLLOWBN, [70](#)
- DAVIS640\_CONFIG\_BIAS\_DACBUFBP, [70](#)
- DAVIS640\_CONFIG\_BIAS\_DIFFBN, [71](#)
- DAVIS640\_CONFIG\_BIAS\_IFREFRBN, [71](#)
- DAVIS640\_CONFIG\_BIAS\_IFTHRBN, [71](#)
- DAVIS640\_CONFIG\_BIAS\_LCOLTIMEOUTBN, [72](#)
- DAVIS640\_CONFIG\_BIAS\_LOCALBUFBN, [72](#)
- DAVIS640\_CONFIG\_BIAS\_OFFBN, [72](#)
- DAVIS640\_CONFIG\_BIAS\_ONBN, [73](#)
- DAVIS640\_CONFIG\_BIAS\_PADFOLLBN, [73](#)
- DAVIS640\_CONFIG\_BIAS\_PIXINVBN, [73](#)
- DAVIS640\_CONFIG\_BIAS\_PRBP, [74](#)
- DAVIS640\_CONFIG\_BIAS\_PRSFBP, [74](#)
- DAVIS640\_CONFIG\_BIAS\_READOUTBUFBP, [74](#)
- DAVIS640\_CONFIG\_BIAS\_REFRBP, [75](#)
- DAVIS640\_CONFIG\_BIAS\_SSN, [75](#)
- DAVIS640\_CONFIG\_BIAS\_SSP, [75](#)
- DAVIS640\_CONFIG\_CHIP\_AERNAROW, [76](#)
- DAVIS640\_CONFIG\_CHIP\_ANALOGMUX0, [76](#)
- DAVIS640\_CONFIG\_CHIP\_ANALOGMUX1, [76](#)
- DAVIS640\_CONFIG\_CHIP\_ANALOGMUX2, [76](#)
- DAVIS640\_CONFIG\_CHIP\_BIASMUX0, [76](#)
- DAVIS640\_CONFIG\_CHIP\_DIGITALMUX0, [77](#)
- DAVIS640\_CONFIG\_CHIP\_DIGITALMUX1, [77](#)
- DAVIS640\_CONFIG\_CHIP\_DIGITALMUX2, [77](#)
- DAVIS640\_CONFIG\_CHIP\_DIGITALMUX3, [77](#)

- DAVIS640\_CONFIG\_CHIP\_GLOBAL\_SHUTTER, 77
- DAVIS640\_CONFIG\_CHIP\_RESETCALIBNEU↔RON, 77
- DAVIS640\_CONFIG\_CHIP\_RESETTESTPIXEL, 78
- DAVIS640\_CONFIG\_CHIP\_SELECTGRAYCO↔UNTER, 78
- DAVIS640\_CONFIG\_CHIP\_TESTADC, 78
- DAVIS640\_CONFIG\_CHIP\_TYPENCALIBNEU↔RON, 78
- DAVIS640\_CONFIG\_CHIP\_USEAOUT, 78
- DAVIS\_CHIP\_DAVIS128, 78
- DAVIS\_CHIP\_DAVIS208, 79
- DAVIS\_CHIP\_DAVIS240A, 79
- DAVIS\_CHIP\_DAVIS240B, 79
- DAVIS\_CHIP\_DAVIS240C, 79
- DAVIS\_CHIP\_DAVIS346A, 79
- DAVIS\_CHIP\_DAVIS346B, 79
- DAVIS\_CHIP\_DAVIS346C, 79
- DAVIS\_CHIP\_DAVIS640, 79
- DAVIS\_CHIP\_DAVISRGB, 80
- DAVIS\_CONFIG\_APS\_ADC\_TEST\_MODE, 80
- DAVIS\_CONFIG\_APS\_AUTOEXPOSURE, 80
- DAVIS\_CONFIG\_APS\_COLOR\_FILTER, 80
- DAVIS\_CONFIG\_APS\_COLUMN\_SETTLE, 80
- DAVIS\_CONFIG\_APS\_END\_COLUMN\_0, 80
- DAVIS\_CONFIG\_APS\_END\_COLUMN\_1, 81
- DAVIS\_CONFIG\_APS\_END\_COLUMN\_2, 81
- DAVIS\_CONFIG\_APS\_END\_COLUMN\_3, 81
- DAVIS\_CONFIG\_APS\_END\_ROW\_0, 81
- DAVIS\_CONFIG\_APS\_END\_ROW\_1, 81
- DAVIS\_CONFIG\_APS\_END\_ROW\_2, 81
- DAVIS\_CONFIG\_APS\_END\_ROW\_3, 81
- DAVIS\_CONFIG\_APS\_EXPOSURE, 82
- DAVIS\_CONFIG\_APS\_FRAME\_DELAY, 82
- DAVIS\_CONFIG\_APS\_GLOBAL\_SHUTTER, 82
- DAVIS\_CONFIG\_APS\_HAS\_EXTERNAL\_ADC, 82
- DAVIS\_CONFIG\_APS\_HAS\_GLOBAL\_SHUTT↔ER, 82
- DAVIS\_CONFIG\_APS\_HAS\_INTERNAL\_ADC, 82
- DAVIS\_CONFIG\_APS\_HAS\_QUAD\_ROI, 83
- DAVIS\_CONFIG\_APS\_NULL\_SETTLE, 83
- DAVIS\_CONFIG\_APS\_ORIENTATION\_INFO, 83
- DAVIS\_CONFIG\_APS\_RAMP\_RESET, 83
- DAVIS\_CONFIG\_APS\_RAMP\_SHORT\_RESET, 83
- DAVIS\_CONFIG\_APS\_RESET\_READ, 83
- DAVIS\_CONFIG\_APS\_RESET\_SETTLE, 84
- DAVIS\_CONFIG\_APS\_ROW\_SETTLE, 84
- DAVIS\_CONFIG\_APS\_RUN, 84
- DAVIS\_CONFIG\_APS\_SAMPLE\_ENABLE, 84
- DAVIS\_CONFIG\_APS\_SAMPLE\_SETTLE, 84
- DAVIS\_CONFIG\_APS\_SIZE\_COLUMNS, 84
- DAVIS\_CONFIG\_APS\_SIZE\_ROWS, 84
- DAVIS\_CONFIG\_APS\_SNAPSHOT, 85
- DAVIS\_CONFIG\_APS\_START\_COLUMN\_0, 85
- DAVIS\_CONFIG\_APS\_START\_COLUMN\_1, 85
- DAVIS\_CONFIG\_APS\_START\_COLUMN\_2, 85
- DAVIS\_CONFIG\_APS\_START\_COLUMN\_3, 85
- DAVIS\_CONFIG\_APS\_START\_ROW\_0, 85
- DAVIS\_CONFIG\_APS\_START\_ROW\_1, 86
- DAVIS\_CONFIG\_APS\_START\_ROW\_2, 86
- DAVIS\_CONFIG\_APS\_START\_ROW\_3, 86
- DAVIS\_CONFIG\_APS\_USE\_INTERNAL\_ADC, 86
- DAVIS\_CONFIG\_APS\_WAIT\_ON\_TRANSFER↔\_STALL, 86
- DAVIS\_CONFIG\_APS, 80
- DAVIS\_CONFIG\_BIAS, 86
- DAVIS\_CONFIG\_CHIP, 87
- DAVIS\_CONFIG\_DVS\_ACK\_DELAY\_COLUMN, 87
- DAVIS\_CONFIG\_DVS\_ACK\_DELAY\_ROW, 87
- DAVIS\_CONFIG\_DVS\_ACK\_EXTENSION\_CO↔LUMN, 87
- DAVIS\_CONFIG\_DVS\_ACK\_EXTENSION\_ROW, 87
- DAVIS\_CONFIG\_DVS\_EXTERNAL\_AER\_CON↔TROL, 87
- DAVIS\_CONFIG\_DVS\_FILTER\_BACKGROUN↔D\_ACTIVITY\_DELTAT, 88
- DAVIS\_CONFIG\_DVS\_FILTER\_BACKGROUN↔D\_ACTIVITY, 88
- DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_0\_COL↔UMN, 88
- DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_0\_ROW, 88
- DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_1\_COL↔UMN, 88
- DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_1\_ROW, 88
- DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_2\_COL↔UMN, 89
- DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_2\_ROW, 89
- DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_3\_COL↔UMN, 89
- DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_3\_ROW, 89
- DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_4\_COL↔UMN, 89
- DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_4\_ROW, 89
- DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_5\_COL↔UMN, 89
- DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_5\_ROW, 90
- DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_6\_COL↔UMN, 90
- DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_6\_ROW, 90
- DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_7\_COL↔UMN, 90
- DAVIS\_CONFIG\_DVS\_FILTER\_PIXEL\_7\_ROW, 90

- DAVIS\_CONFIG\_DVS\_FILTER\_ROW\_ONLY\_↵  
EVENTS, [90](#)
- DAVIS\_CONFIG\_DVS\_HAS\_BACKGROUND\_↵  
ACTIVITY\_FILTER, [90](#)
- DAVIS\_CONFIG\_DVS\_HAS\_PIXEL\_FILTER, [91](#)
- DAVIS\_CONFIG\_DVS\_HAS\_TEST\_EVENT\_G↵  
ENERATOR, [91](#)
- DAVIS\_CONFIG\_DVS\_ORIENTATION\_INFO, [91](#)
- DAVIS\_CONFIG\_DVS\_RUN, [91](#)
- DAVIS\_CONFIG\_DVS\_SIZE\_COLUMNS, [91](#)
- DAVIS\_CONFIG\_DVS\_SIZE\_ROWS, [91](#)
- DAVIS\_CONFIG\_DVS\_TEST\_EVENT\_GENER↵  
ATOR\_ENABLE, [92](#)
- DAVIS\_CONFIG\_DVS\_WAIT\_ON\_TRANSFER↵  
\_STALL, [92](#)
- DAVIS\_CONFIG\_DVS, [87](#)
- DAVIS\_CONFIG\_EXTINPUT\_DETECT\_FALLIN↵  
G\_EDGES1, [92](#)
- DAVIS\_CONFIG\_EXTINPUT\_DETECT\_FALLIN↵  
G\_EDGES2, [92](#)
- DAVIS\_CONFIG\_EXTINPUT\_DETECT\_FALLIN↵  
G\_EDGES, [92](#)
- DAVIS\_CONFIG\_EXTINPUT\_DETECT\_PULSE↵  
\_LENGTH1, [93](#)
- DAVIS\_CONFIG\_EXTINPUT\_DETECT\_PULSE↵  
\_LENGTH2, [93](#)
- DAVIS\_CONFIG\_EXTINPUT\_DETECT\_PULSE↵  
\_LENGTH, [93](#)
- DAVIS\_CONFIG\_EXTINPUT\_DETECT\_PULSE↵  
\_POLARITY1, [93](#)
- DAVIS\_CONFIG\_EXTINPUT\_DETECT\_PULSE↵  
\_POLARITY2, [93](#)
- DAVIS\_CONFIG\_EXTINPUT\_DETECT\_PULSE↵  
\_POLARITY, [93](#)
- DAVIS\_CONFIG\_EXTINPUT\_DETECT\_PULSE↵  
S1, [94](#)
- DAVIS\_CONFIG\_EXTINPUT\_DETECT\_PULSE↵  
S2, [94](#)
- DAVIS\_CONFIG\_EXTINPUT\_DETECT\_PULSES,  
[94](#)
- DAVIS\_CONFIG\_EXTINPUT\_DETECT\_RISING↵  
\_EDGES1, [94](#)
- DAVIS\_CONFIG\_EXTINPUT\_DETECT\_RISING↵  
\_EDGES2, [94](#)
- DAVIS\_CONFIG\_EXTINPUT\_DETECT\_RISING↵  
\_EDGES, [94](#)
- DAVIS\_CONFIG\_EXTINPUT\_GENERATE\_INJ↵  
ECT\_ON\_FALLING\_EDGE, [95](#)
- DAVIS\_CONFIG\_EXTINPUT\_GENERATE\_INJ↵  
ECT\_ON\_RISING\_EDGE, [95](#)
- DAVIS\_CONFIG\_EXTINPUT\_GENERATE\_PUL↵  
SE\_INTERVAL, [95](#)
- DAVIS\_CONFIG\_EXTINPUT\_GENERATE\_PUL↵  
SE\_LENGTH, [95](#)
- DAVIS\_CONFIG\_EXTINPUT\_GENERATE\_PUL↵  
SE\_POLARITY, [95](#)
- DAVIS\_CONFIG\_EXTINPUT\_GENERATE\_US↵  
E\_CUSTOM\_SIGNAL, [95](#)
- DAVIS\_CONFIG\_EXTINPUT\_HAS\_EXTRA\_DE↵  
TECTORS, [96](#)
- DAVIS\_CONFIG\_EXTINPUT\_HAS\_GENERAT↵  
OR, [96](#)
- DAVIS\_CONFIG\_EXTINPUT\_RUN\_DETECTO↵  
R1, [96](#)
- DAVIS\_CONFIG\_EXTINPUT\_RUN\_DETECTO↵  
R2, [96](#)
- DAVIS\_CONFIG\_EXTINPUT\_RUN\_DETECTOR,  
[96](#)
- DAVIS\_CONFIG\_EXTINPUT\_RUN\_GENERAT↵  
OR, [96](#)
- DAVIS\_CONFIG\_EXTINPUT, [92](#)
- DAVIS\_CONFIG\_IMU\_ACCEL\_FULL\_SCALE, [97](#)
- DAVIS\_CONFIG\_IMU\_ACCEL\_STANDBY, [97](#)
- DAVIS\_CONFIG\_IMU\_DIGITAL\_LOW\_PASS\_F↵  
ILTER, [97](#)
- DAVIS\_CONFIG\_IMU\_GYRO\_FULL\_SCALE, [97](#)
- DAVIS\_CONFIG\_IMU\_GYRO\_STANDBY, [97](#)
- DAVIS\_CONFIG\_IMU\_LP\_CYCLE, [98](#)
- DAVIS\_CONFIG\_IMU\_LP\_WAKEUP, [98](#)
- DAVIS\_CONFIG\_IMU\_ORIENTATION\_INFO, [98](#)
- DAVIS\_CONFIG\_IMU\_RUN, [98](#)
- DAVIS\_CONFIG\_IMU\_SAMPLE\_RATE\_DIVID↵  
ER, [98](#)
- DAVIS\_CONFIG\_IMU\_TEMP\_STANDBY, [98](#)
- DAVIS\_CONFIG\_IMU, [97](#)
- DAVIS\_CONFIG\_MICROPHONE\_RUN, [99](#)
- DAVIS\_CONFIG\_MICROPHONE\_SAMPLE\_FR↵  
QUENCY, [99](#)
- DAVIS\_CONFIG\_MICROPHONE, [99](#)
- DAVIS\_CONFIG\_MUX\_DROP\_APS\_ON\_TRA↵  
NSFER\_STALL, [99](#)
- DAVIS\_CONFIG\_MUX\_DROP\_DVS\_ON\_TRA↵  
NSFER\_STALL, [99](#)
- DAVIS\_CONFIG\_MUX\_DROP\_EXTINPUT\_ON↵  
\_TRANSFER\_STALL, [100](#)
- DAVIS\_CONFIG\_MUX\_DROP\_IMU\_ON\_TRAN↵  
SFER\_STALL, [100](#)
- DAVIS\_CONFIG\_MUX\_DROP\_MIC\_ON\_TRAN↵  
SFER\_STALL, [100](#)
- DAVIS\_CONFIG\_MUX\_FORCE\_CHIP\_BIAS\_E↵  
NABLE, [100](#)
- DAVIS\_CONFIG\_MUX\_RUN, [100](#)
- DAVIS\_CONFIG\_MUX\_TIMESTAMP\_RESET,  
[100](#)
- DAVIS\_CONFIG\_MUX\_TIMESTAMP\_RUN, [101](#)
- DAVIS\_CONFIG\_MUX, [99](#)
- DAVIS\_CONFIG\_SYSINFO\_ADC\_CLOCK, [101](#)
- DAVIS\_CONFIG\_SYSINFO\_CHIP\_IDENTIFIER,  
[101](#)
- DAVIS\_CONFIG\_SYSINFO\_DEVICE\_IS\_MAST↵  
ER, [101](#)
- DAVIS\_CONFIG\_SYSINFO\_LOGIC\_CLOCK, [101](#)
- DAVIS\_CONFIG\_SYSINFO\_LOGIC\_VERSION,  
[102](#)
- DAVIS\_CONFIG\_SYSINFO, [101](#)
- DAVIS\_CONFIG\_USB\_EARLY\_PACKET\_DELAY,

- 102
- DAVIS\_CONFIG\_USB\_RUN, 102
- DAVIS\_CONFIG\_USB, 102
- DAVISRGB\_CONFIG\_APS\_GSFDRESET, 102
- DAVISRGB\_CONFIG\_APS\_GSPDRESET, 102
- DAVISRGB\_CONFIG\_APS\_GSRESETFALL, 103
- DAVISRGB\_CONFIG\_APS\_GSTXFALL, 103
- DAVISRGB\_CONFIG\_APS\_RSFDSETTLE, 103
- DAVISRGB\_CONFIG\_APS\_TRANSFER, 103
- DAVISRGB\_CONFIG\_BIAS\_ADCCOMPBP, 103
- DAVISRGB\_CONFIG\_BIAS\_ADCREFHIGH, 103
- DAVISRGB\_CONFIG\_BIAS\_ADCREFLOW, 104
- DAVISRGB\_CONFIG\_BIAS\_ADCTESTVOLTA↔GE, 104
- DAVISRGB\_CONFIG\_BIAS\_AEPDBN, 104
- DAVISRGB\_CONFIG\_BIAS\_AEPUXBP, 105
- DAVISRGB\_CONFIG\_BIAS\_AEPUYBP, 105
- DAVISRGB\_CONFIG\_BIAS\_APSCAS, 105
- DAVISRGB\_CONFIG\_BIAS\_APSROSFBN, 106
- DAVISRGB\_CONFIG\_BIAS\_ARRAYBIASBUFF↔ERBN, 106
- DAVISRGB\_CONFIG\_BIAS\_ARRAYLOGICBU↔FFERBN, 106
- DAVISRGB\_CONFIG\_BIAS\_BIASBUFFER, 107
- DAVISRGB\_CONFIG\_BIAS\_DACBUFBP, 107
- DAVISRGB\_CONFIG\_BIAS\_DIFFBN, 107
- DAVISRGB\_CONFIG\_BIAS\_FALLTIMEBN, 108
- DAVISRGB\_CONFIG\_BIAS\_GND07, 108
- DAVISRGB\_CONFIG\_BIAS\_IFREFRBN, 108
- DAVISRGB\_CONFIG\_BIAS\_IFTHRBN, 109
- DAVISRGB\_CONFIG\_BIAS\_LCOLTIMEOUTBN, 109
- DAVISRGB\_CONFIG\_BIAS\_LOCALBUFBN, 109
- DAVISRGB\_CONFIG\_BIAS\_OFFBN, 110
- DAVISRGB\_CONFIG\_BIAS\_ONBN, 110
- DAVISRGB\_CONFIG\_BIAS\_OVG1LO, 110
- DAVISRGB\_CONFIG\_BIAS\_OVG2LO, 111
- DAVISRGB\_CONFIG\_BIAS\_PADFOLLBN, 111
- DAVISRGB\_CONFIG\_BIAS\_PIXINBN, 111
- DAVISRGB\_CONFIG\_BIAS\_PRBP, 112
- DAVISRGB\_CONFIG\_BIAS\_PRSFBP, 112
- DAVISRGB\_CONFIG\_BIAS\_READOUTBUFBP, 112
- DAVISRGB\_CONFIG\_BIAS\_REFRBP, 113
- DAVISRGB\_CONFIG\_BIAS\_RISETIMEBP, 113
- DAVISRGB\_CONFIG\_BIAS\_SSN, 113
- DAVISRGB\_CONFIG\_BIAS\_SSP, 114
- DAVISRGB\_CONFIG\_BIAS\_TX2OVG2HI, 114
- DAVISRGB\_CONFIG\_CHIP\_ADJUSTOVG1LO, 114
- DAVISRGB\_CONFIG\_CHIP\_ADJUSTOVG2LO, 115
- DAVISRGB\_CONFIG\_CHIP\_ADJUSTTX2OV↔G2HI, 115
- DAVISRGB\_CONFIG\_CHIP\_AERNAROW, 115
- DAVISRGB\_CONFIG\_CHIP\_ANALOGMUX0, 115
- DAVISRGB\_CONFIG\_CHIP\_ANALOGMUX1, 115
- DAVISRGB\_CONFIG\_CHIP\_ANALOGMUX2, 115
- DAVISRGB\_CONFIG\_CHIP\_BIASMUX0, 116
- DAVISRGB\_CONFIG\_CHIP\_DIGITALMUX0, 116
- DAVISRGB\_CONFIG\_CHIP\_DIGITALMUX1, 116
- DAVISRGB\_CONFIG\_CHIP\_DIGITALMUX2, 116
- DAVISRGB\_CONFIG\_CHIP\_DIGITALMUX3, 116
- DAVISRGB\_CONFIG\_CHIP\_RESETCALIBNEU↔RON, 116
- DAVISRGB\_CONFIG\_CHIP\_RESETTESTPIXEL, 117
- DAVISRGB\_CONFIG\_CHIP\_SELECTGRAYCO↔UNTER, 117
- DAVISRGB\_CONFIG\_CHIP\_TESTADC, 117
- DAVISRGB\_CONFIG\_CHIP\_TYPENCALIBNEU↔RON, 117
- DAVISRGB\_CONFIG\_CHIP\_USEAOUT, 117
- IS\_DAVIS128, 117
- IS\_DAVIS208, 118
- IS\_DAVIS240, 118
- IS\_DAVIS240A, 118
- IS\_DAVIS240B, 118
- IS\_DAVIS240C, 118
- IS\_DAVIS346, 118
- IS\_DAVIS346A, 119
- IS\_DAVIS346B, 119
- IS\_DAVIS346C, 119
- IS\_DAVIS640, 119
- IS\_DAVISRGB, 119
- device.h
  - CAER\_HOST\_CONFIG\_DATAEXCHANGE\_BL↔OCKING, 125
  - CAER\_HOST\_CONFIG\_DATAEXCHANGE\_BU↔FFER\_SIZE, 125
  - CAER\_HOST\_CONFIG\_DATAEXCHANGE\_ST↔ART\_PRODUCERS, 125
  - CAER\_HOST\_CONFIG\_DATAEXCHANGE\_ST↔OP\_PRODUCERS, 125
  - CAER\_HOST\_CONFIG\_DATAEXCHANGE, 124
  - CAER\_HOST\_CONFIG\_LOG\_LEVEL, 125
  - CAER\_HOST\_CONFIG\_LOG, 125
  - CAER\_HOST\_CONFIG\_PACKETS\_MAX\_CON↔TAINER\_INTERVAL, 126
  - CAER\_HOST\_CONFIG\_PACKETS\_MAX\_CON↔TAINER\_PACKET\_SIZE, 126
  - CAER\_HOST\_CONFIG\_PACKETS, 126
  - caerDeviceClose, 126
  - caerDeviceConfigGet, 127
  - caerDeviceConfigSet, 127
  - caerDeviceDataGet, 128
  - caerDeviceDataStart, 128
  - caerDeviceDataStop, 129
  - caerDeviceHandle, 126
  - caerDeviceSendDefaultConfig, 129
- devices/davis.h, 11
- devices/device.h, 124
- devices/dvs128.h, 130
- devices/dynapse.h, 134
- devices/edvs.h, 157
- devices/serial.h, 161

devices/usb.h, [163](#)

dvs128.h

CAER\_DEVICE\_DVS128, [131](#)  
 caerDVS128InfoGet, [133](#)  
 DVS128\_CONFIG\_BIAS\_CAS, [131](#)  
 DVS128\_CONFIG\_BIAS\_DIFFOFF, [131](#)  
 DVS128\_CONFIG\_BIAS\_DIFFON, [131](#)  
 DVS128\_CONFIG\_BIAS\_DIFF, [131](#)  
 DVS128\_CONFIG\_BIAS\_FOLL, [131](#)  
 DVS128\_CONFIG\_BIAS\_INJGND, [132](#)  
 DVS128\_CONFIG\_BIAS\_PUX, [132](#)  
 DVS128\_CONFIG\_BIAS\_PUY, [132](#)  
 DVS128\_CONFIG\_BIAS\_PR, [132](#)  
 DVS128\_CONFIG\_BIAS\_REFR, [132](#)  
 DVS128\_CONFIG\_BIAS\_REQPD, [132](#)  
 DVS128\_CONFIG\_BIAS\_REQ, [132](#)  
 DVS128\_CONFIG\_BIAS, [131](#)  
 DVS128\_CONFIG\_DVS\_ARRAY\_RESET, [133](#)  
 DVS128\_CONFIG\_DVS\_RUN, [133](#)  
 DVS128\_CONFIG\_DVS\_TIMESTAMP\_RESET, [133](#)  
 DVS128\_CONFIG\_DVS\_TS\_MASTER, [133](#)  
 DVS128\_CONFIG\_DVS, [133](#)

dynapse.h

CAER\_DEVICE\_DYNAPSE, [139](#)  
 caerBiasDynapseGenerate, [149](#)  
 caerBiasDynapseParse, [149](#)  
 caerDynapseCoreAddrToNeuronId, [150](#)  
 caerDynapseCoreXYToNeuronId, [150](#)  
 caerDynapseGenerateCamBits, [150](#)  
 caerDynapseGenerateSramBits, [151](#)  
 caerDynapseInfoGet, [152](#)  
 caerDynapseSendDataToUSB, [152](#)  
 caerDynapseSpikeEventFromXY, [152](#)  
 caerDynapseSpikeEventGetX, [153](#)  
 caerDynapseSpikeEventGetY, [153](#)  
 caerDynapseWriteCam, [154](#)  
 caerDynapseWritePoissonSpikeRate, [154](#)  
 caerDynapseWriteSram, [155](#)  
 caerDynapseWriteSramWords, [156](#)  
 caerDynapseWriteSramN, [155](#)  
 DYNAPSE\_CHIP\_DYNAPSE, [139](#)  
 DYNAPSE\_CONFIG\_AER\_ACK\_DELAY, [139](#)  
 DYNAPSE\_CONFIG\_AER\_ACK\_EXTENSION, [139](#)  
 DYNAPSE\_CONFIG\_AER\_EXTERNAL\_AER\_↔  
 CONTROL, [139](#)  
 DYNAPSE\_CONFIG\_AER\_RUN, [139](#)  
 DYNAPSE\_CONFIG\_AER\_WAIT\_ON\_TRANSF↔  
 ER\_STALL, [140](#)  
 DYNAPSE\_CONFIG\_AER, [139](#)  
 DYNAPSE\_CONFIG\_BIAS\_C0\_PULSE\_PWLK↔  
 \_P, [140](#)  
 DYNAPSE\_CONFIG\_CHIP\_CONTENT, [140](#)  
 DYNAPSE\_CONFIG\_CHIP\_ID, [140](#)  
 DYNAPSE\_CONFIG\_CHIP\_REQ\_DELAY, [140](#)  
 DYNAPSE\_CONFIG\_CHIP\_REQ\_EXTENSION, [141](#)

DYNAPSE\_CONFIG\_CHIP\_RUN, [141](#)  
 DYNAPSE\_CONFIG\_CHIP, [140](#)  
 DYNAPSE\_CONFIG\_CLEAR\_CAM, [141](#)  
 DYNAPSE\_CONFIG\_DEFAULT\_SRAM\_EMPTY, [141](#)  
 DYNAPSE\_CONFIG\_DEFAULT\_SRAM, [141](#)  
 DYNAPSE\_CONFIG\_MONITOR\_NEU, [141](#)  
 DYNAPSE\_CONFIG\_MUX\_DROP\_AER\_ON\_T↔  
 RANSFER\_STALL, [142](#)  
 DYNAPSE\_CONFIG\_MUX\_FORCE\_CHIP\_BIA↔  
 S\_ENABLE, [142](#)  
 DYNAPSE\_CONFIG\_MUX\_RUN, [142](#)  
 DYNAPSE\_CONFIG\_MUX\_TIMESTAMP\_RES↔  
 ET, [142](#)  
 DYNAPSE\_CONFIG\_MUX\_TIMESTAMP\_RUN, [142](#)  
 DYNAPSE\_CONFIG\_MUX, [141](#)  
 DYNAPSE\_CONFIG\_POISSONSPIKEGEN\_CH↔  
 IPID, [143](#)  
 DYNAPSE\_CONFIG\_POISSONSPIKEGEN\_RUN, [143](#)  
 DYNAPSE\_CONFIG\_POISSONSPIKEGEN\_W↔  
 RITEADDRESS, [143](#)  
 DYNAPSE\_CONFIG\_POISSONSPIKEGEN\_W↔  
 RITEDATA, [143](#)  
 DYNAPSE\_CONFIG\_POISSONSPIKEGEN, [142](#)  
 DYNAPSE\_CONFIG\_SPIKEGEN\_BASEADDR, [143](#)  
 DYNAPSE\_CONFIG\_SPIKEGEN\_ISIBASE, [144](#)  
 DYNAPSE\_CONFIG\_SPIKEGEN\_ISI, [143](#)  
 DYNAPSE\_CONFIG\_SPIKEGEN\_REPEAT, [144](#)  
 DYNAPSE\_CONFIG\_SPIKEGEN\_RUN, [144](#)  
 DYNAPSE\_CONFIG\_SPIKEGEN\_STIMCOUNT, [144](#)  
 DYNAPSE\_CONFIG\_SPIKEGEN\_VARMODE, [144](#)  
 DYNAPSE\_CONFIG\_SPIKEGEN, [143](#)  
 DYNAPSE\_CONFIG\_SRAM\_ADDRESS, [144](#)  
 DYNAPSE\_CONFIG\_SRAM\_BURSTMODE, [145](#)  
 DYNAPSE\_CONFIG\_SRAM\_DIRECTION\_POS, [145](#)  
 DYNAPSE\_CONFIG\_SRAM\_READDATA, [145](#)  
 DYNAPSE\_CONFIG\_SRAM\_READ, [145](#)  
 DYNAPSE\_CONFIG\_SRAM\_RWCOMMAND, [145](#)  
 DYNAPSE\_CONFIG\_SRAM\_WRITEDATA, [146](#)  
 DYNAPSE\_CONFIG\_SRAM\_WRITE, [145](#)  
 DYNAPSE\_CONFIG\_SRAM, [144](#)  
 DYNAPSE\_CONFIG\_SYNAPSERECONFIG\_C↔  
 HIPSELECT, [146](#)  
 DYNAPSE\_CONFIG\_SYNAPSERECONFIG\_G↔  
 LOBALKERNEL, [146](#)  
 DYNAPSE\_CONFIG\_SYNAPSERECONFIG\_R↔  
 UN, [146](#)  
 DYNAPSE\_CONFIG\_SYNAPSERECONFIG\_S↔  
 RAMBASEADDR, [146](#)  
 DYNAPSE\_CONFIG\_SYNAPSERECONFIG\_U↔  
 SESRAMKERNELS, [147](#)  
 DYNAPSE\_CONFIG\_SYNAPSERECONFIG, [146](#)



- DYNAPSE\_CONFIG\_SYSINFO\_CHIP\_IDENTITY, 147
- DYNAPSE\_CONFIG\_SYSINFO\_DEVICE\_IS\_MASTER, 147
- DYNAPSE\_CONFIG\_SYSINFO\_LOGIC\_CLOCK, 147
- DYNAPSE\_CONFIG\_SYSINFO\_LOGIC\_VERSION, 147
- DYNAPSE\_CONFIG\_SYSINFO, 147
- DYNAPSE\_CONFIG\_USB\_EARLY\_PACKET\_DELAY, 148
- DYNAPSE\_CONFIG\_USB\_RUN, 148
- DYNAPSE\_CONFIG\_USB, 148
- DYNAPSE\_X4BOARD\_COREX, 148
- DYNAPSE\_X4BOARD\_COREY, 148
- DYNAPSE\_X4BOARD\_NEUX, 148
- DYNAPSE\_X4BOARD\_NEUY, 148
- DYNAPSE\_X4BOARD\_NUMCHIPS, 149
- EAR\_CHANNEL\_MASK
  - ear.h, 197
- EAR\_CHANNEL\_SHIFT
  - ear.h, 197
- EAR\_FILTER\_MASK
  - ear.h, 198
- EAR\_FILTER\_SHIFT
  - ear.h, 198
- EAR\_MASK
  - ear.h, 198
- EAR\_NEURON\_MASK
  - ear.h, 198
- EAR\_NEURON\_SHIFT
  - ear.h, 198
- EAR\_SHIFT
  - ear.h, 198
- EDVS\_CONFIG\_BIAS\_CAS
  - edvs.h, 158
- EDVS\_CONFIG\_BIAS\_DIFFOFF
  - edvs.h, 158
- EDVS\_CONFIG\_BIAS\_DIFFON
  - edvs.h, 158
- EDVS\_CONFIG\_BIAS\_DIFF
  - edvs.h, 158
- EDVS\_CONFIG\_BIAS\_FOLL
  - edvs.h, 158
- EDVS\_CONFIG\_BIAS\_INJGND
  - edvs.h, 159
- EDVS\_CONFIG\_BIAS\_PUX
  - edvs.h, 159
- EDVS\_CONFIG\_BIAS\_PUY
  - edvs.h, 159
- EDVS\_CONFIG\_BIAS\_PR
  - edvs.h, 159
- EDVS\_CONFIG\_BIAS\_REFR
  - edvs.h, 159
- EDVS\_CONFIG\_BIAS\_REQPD
  - edvs.h, 159
- EDVS\_CONFIG\_BIAS\_REQ
  - edvs.h, 159
- EDVS\_CONFIG\_BIAS
  - edvs.h, 158
- EDVS\_CONFIG\_DVS\_RUN
  - edvs.h, 160
- EDVS\_CONFIG\_DVS\_TIMESTAMP\_RESET
  - edvs.h, 160
- EDVS\_CONFIG\_DVS
  - edvs.h, 160
- ear.h
  - CAER\_EAR\_CONST\_ITERATOR\_ALL\_START, 194
  - CAER\_EAR\_CONST\_ITERATOR\_VALID\_START, 194
  - CAER\_EAR\_CONST\_REVERSE\_ITERATOR\_ALL\_START, 194
  - CAER\_EAR\_CONST\_REVERSE\_ITERATOR\_VALID\_START, 195
  - CAER\_EAR\_ITERATOR\_ALL\_END, 195
  - CAER\_EAR\_ITERATOR\_ALL\_START, 195
  - CAER\_EAR\_ITERATOR\_VALID\_END, 196
  - CAER\_EAR\_ITERATOR\_VALID\_START, 196
  - CAER\_EAR\_REVERSE\_ITERATOR\_ALL\_END, 196
  - CAER\_EAR\_REVERSE\_ITERATOR\_ALL\_START, 196
  - CAER\_EAR\_REVERSE\_ITERATOR\_VALID\_END, 197
  - CAER\_EAR\_REVERSE\_ITERATOR\_VALID\_START, 197
  - caerEarEvent, 199
  - caerEarEventGetChannel, 199
  - caerEarEventGetEar, 199
  - caerEarEventGetTimestamp, 200
  - caerEarEventGetTimestamp64, 200
  - caerEarEventInvalidate, 201
  - caerEarEventIsValid, 201
  - caerEarEventPacket, 199
  - caerEarEventPacketAllocate, 201
  - caerEarEventPacketGetEvent, 202
  - caerEarEventPacketGetEventConst, 202
  - caerEarEventSetChannel, 203
  - caerEarEventSetEar, 203
  - caerEarEventSetTimestamp, 203
  - caerEarEventValidate, 204
  - EAR\_CHANNEL\_MASK, 197
  - EAR\_CHANNEL\_SHIFT, 197
  - EAR\_FILTER\_MASK, 198
  - EAR\_FILTER\_SHIFT, 198
  - EAR\_MASK, 198
  - EAR\_NEURON\_MASK, 198
  - EAR\_NEURON\_SHIFT, 198
  - EAR\_SHIFT, 198
  - PACKED\_STRUCT, 204
- edvs.h
  - CAER\_DEVICE\_EDVS, 158
  - caerEDVSInfoGet, 160
  - EDVS\_CONFIG\_BIAS\_CAS, 158
  - EDVS\_CONFIG\_BIAS\_DIFFOFF, 158

- EDVS\_CONFIG\_BIAS\_DIFFON, 158
- EDVS\_CONFIG\_BIAS\_DIFF, 158
- EDVS\_CONFIG\_BIAS\_FOLL, 158
- EDVS\_CONFIG\_BIAS\_INJGND, 159
- EDVS\_CONFIG\_BIAS\_PUX, 159
- EDVS\_CONFIG\_BIAS\_PUY, 159
- EDVS\_CONFIG\_BIAS\_PR, 159
- EDVS\_CONFIG\_BIAS\_REFR, 159
- EDVS\_CONFIG\_BIAS\_REQPD, 159
- EDVS\_CONFIG\_BIAS\_REQ, 159
- EDVS\_CONFIG\_BIAS, 158
- EDVS\_CONFIG\_DVS\_RUN, 160
- EDVS\_CONFIG\_DVS\_TIMESTAMP\_RESET, 160
- EDVS\_CONFIG\_DVS, 160
- events/common.h, 164
- events/config.h, 181
- events/ear.h, 192
- events/frame.h, 204
- events/imu6.h, 232
- events/imu9.h, 245
- events/packetContainer.h, 262
- events/point1d.h, 271
- events/point2d.h, 283
- events/point3d.h, 296
- events/point4d.h, 309
- events/polarity.h, 323
- events/sample.h, 335
- events/special.h, 346
- events/spike.h, 360
- FRAME\_COLOR\_CHANNELS\_MASK
  - frame.h, 210
- FRAME\_COLOR\_CHANNELS\_SHIFT
  - frame.h, 210
- FRAME\_COLOR\_FILTER\_MASK
  - frame.h, 211
- FRAME\_COLOR\_FILTER\_SHIFT
  - frame.h, 211
- FRAME\_ROI\_IDENTIFIER\_MASK
  - frame.h, 211
- FRAME\_ROI\_IDENTIFIER\_SHIFT
  - frame.h, 211
- frame.h
  - CAER\_FRAME\_CONST\_ITERATOR\_ALL\_STA↔RT, 207
  - CAER\_FRAME\_CONST\_ITERATOR\_VALID\_S↔TART, 207
  - CAER\_FRAME\_CONST\_REVERSE\_ITERATO↔R\_ALL\_START, 207
  - CAER\_FRAME\_CONST\_REVERSE\_ITERATO↔R\_VALID\_START, 208
  - CAER\_FRAME\_ITERATOR\_ALL\_END, 208
  - CAER\_FRAME\_ITERATOR\_ALL\_START, 208
  - CAER\_FRAME\_ITERATOR\_VALID\_END, 209
  - CAER\_FRAME\_ITERATOR\_VALID\_START, 209
  - CAER\_FRAME\_REVERSE\_ITERATOR\_ALL\_E↔ND, 209
  - CAER\_FRAME\_REVERSE\_ITERATOR\_ALL\_S↔TART, 209
  - CAER\_FRAME\_REVERSE\_ITERATOR\_VALID↔\_END, 210
  - CAER\_FRAME\_REVERSE\_ITERATOR\_VALID↔\_START, 210
  - caer\_frame\_event\_color\_channels, 212
  - caer\_frame\_event\_color\_filter, 212
  - caerFrameEvent, 212
  - caerFrameEventGetChannelNumber, 213
  - caerFrameEventGetColorFilter, 213
  - caerFrameEventGetExposureLength, 213
  - caerFrameEventGetLengthX, 214
  - caerFrameEventGetLengthY, 214
  - caerFrameEventGetPixel, 215
  - caerFrameEventGetPixelArrayUnsafe, 215
  - caerFrameEventGetPixelArrayUnsafeConst, 215
  - caerFrameEventGetPixelForChannel, 216
  - caerFrameEventGetPixelForChannelUnsafe, 216
  - caerFrameEventGetPixelUnsafe, 217
  - caerFrameEventGetPixelsMaxIndex, 217
  - caerFrameEventGetPixelsSize, 217
  - caerFrameEventGetPositionX, 218
  - caerFrameEventGetPositionY, 218
  - caerFrameEventGetROIIdentifier, 219
  - caerFrameEventGetTSEndOfExposure, 220
  - caerFrameEventGetTSEndOfExposure64, 220
  - caerFrameEventGetTSEndOfFrame, 221
  - caerFrameEventGetTSEndOfFrame64, 221
  - caerFrameEventGetTSSStartOfExposure, 222
  - caerFrameEventGetTSSStartOfExposure64, 222
  - caerFrameEventGetTSSStartOfFrame, 222
  - caerFrameEventGetTSSStartOfFrame64, 223
  - caerFrameEventGetTimestamp, 219
  - caerFrameEventGetTimestamp64, 219
  - caerFrameEventInvalidate, 223
  - caerFrameEventsIsValid, 223
  - caerFrameEventPacket, 212
  - caerFrameEventPacketAllocate, 224
  - caerFrameEventPacketGetEvent, 224
  - caerFrameEventPacketGetEventConst, 225
  - caerFrameEventPacketGetPixelsMaxIndex, 225
  - caerFrameEventPacketGetPixelsSize, 226
  - caerFrameEventSetColorFilter, 226
  - caerFrameEventSetLengthXLengthYChannel↔Number, 226
  - caerFrameEventSetPixel, 227
  - caerFrameEventSetPixelForChannel, 227
  - caerFrameEventSetPixelForChannelUnsafe, 228
  - caerFrameEventSetPixelUnsafe, 228
  - caerFrameEventSetPositionX, 229
  - caerFrameEventSetPositionY, 229
  - caerFrameEventSetROIIdentifier, 229
  - caerFrameEventSetTSEndOfExposure, 229
  - caerFrameEventSetTSEndOfFrame, 230
  - caerFrameEventSetTSSStartOfExposure, 230
  - caerFrameEventSetTSSStartOfFrame, 230
  - caerFrameEventValidate, 231
  - FRAME\_COLOR\_CHANNELS\_MASK, 210
  - FRAME\_COLOR\_CHANNELS\_SHIFT, 210

- FRAME\_COLOR\_FILTER\_MASK, [211](#)
- FRAME\_COLOR\_FILTER\_SHIFT, [211](#)
- FRAME\_ROI\_IDENTIFIER\_MASK, [211](#)
- FRAME\_ROI\_IDENTIFIER\_SHIFT, [211](#)
- PACKED\_STRUCT, [231](#)
- frame\_utils.h, [372](#)
- GET\_NUMBITS16
  - libcaer.h, [374](#)
- GET\_NUMBITS32
  - libcaer.h, [374](#)
- GET\_NUMBITS8
  - libcaer.h, [374](#)
- I16T
  - libcaer.h, [375](#)
- I32T
  - libcaer.h, [375](#)
- I64T
  - libcaer.h, [375](#)
- I8T
  - libcaer.h, [375](#)
- IS\_DAVIS128
  - davis.h, [117](#)
- IS\_DAVIS208
  - davis.h, [118](#)
- IS\_DAVIS240
  - davis.h, [118](#)
- IS\_DAVIS240A
  - davis.h, [118](#)
- IS\_DAVIS240B
  - davis.h, [118](#)
- IS\_DAVIS240C
  - davis.h, [118](#)
- IS\_DAVIS346
  - davis.h, [118](#)
- IS\_DAVIS346A
  - davis.h, [119](#)
- IS\_DAVIS346B
  - davis.h, [119](#)
- IS\_DAVIS346C
  - davis.h, [119](#)
- IS\_DAVIS640
  - davis.h, [119](#)
- IS\_DAVISRGB
  - davis.h, [119](#)
- imu6.h
  - CAER\_IMU6\_CONST\_ITERATOR\_ALL\_START, [233](#)
  - CAER\_IMU6\_CONST\_ITERATOR\_VALID\_START, [233](#)
  - CAER\_IMU6\_CONST\_REVERSE\_ITERATOR\_ALL\_START, [233](#)
  - CAER\_IMU6\_CONST\_REVERSE\_ITERATOR\_VALID\_START, [234](#)
  - CAER\_IMU6\_ITERATOR\_ALL\_END, [234](#)
  - CAER\_IMU6\_ITERATOR\_ALL\_START, [234](#)
  - CAER\_IMU6\_ITERATOR\_VALID\_END, [235](#)
  - CAER\_IMU6\_ITERATOR\_VALID\_START, [235](#)
  - CAER\_IMU6\_REVERSE\_ITERATOR\_ALL\_END, [235](#)
  - CAER\_IMU6\_REVERSE\_ITERATOR\_ALL\_START, [235](#)
  - CAER\_IMU6\_REVERSE\_ITERATOR\_VALID\_END, [236](#)
  - CAER\_IMU6\_REVERSE\_ITERATOR\_VALID\_START, [236](#)
  - caerIMU6Event, [236](#)
  - caerIMU6EventGetAccelX, [237](#)
  - caerIMU6EventGetAccelY, [237](#)
  - caerIMU6EventGetAccelZ, [238](#)
  - caerIMU6EventGetGyroX, [238](#)
  - caerIMU6EventGetGyroY, [238](#)
  - caerIMU6EventGetGyroZ, [239](#)
  - caerIMU6EventGetTemp, [239](#)
  - caerIMU6EventGetTimestamp, [239](#)
  - caerIMU6EventGetTimestamp64, [240](#)
  - caerIMU6EventInvalidate, [240](#)
  - caerIMU6EventIsValid, [241](#)
  - caerIMU6EventPacket, [237](#)
  - caerIMU6EventPacketAllocate, [241](#)
  - caerIMU6EventPacketGetEvent, [241](#)
  - caerIMU6EventPacketGetEventConst, [242](#)
  - caerIMU6EventSetAccelX, [242](#)
  - caerIMU6EventSetAccelY, [242](#)
  - caerIMU6EventSetAccelZ, [243](#)
  - caerIMU6EventSetGyroX, [243](#)
  - caerIMU6EventSetGyroY, [243](#)
  - caerIMU6EventSetGyroZ, [244](#)
  - caerIMU6EventSetTemp, [244](#)
  - caerIMU6EventSetTimestamp, [244](#)
  - caerIMU6EventValidate, [245](#)
  - PACKED\_STRUCT, [245](#)
- imu9.h
  - CAER\_IMU9\_CONST\_ITERATOR\_ALL\_START, [247](#)
  - CAER\_IMU9\_CONST\_ITERATOR\_VALID\_START, [247](#)
  - CAER\_IMU9\_CONST\_REVERSE\_ITERATOR\_ALL\_START, [247](#)
  - CAER\_IMU9\_CONST\_REVERSE\_ITERATOR\_VALID\_START, [248](#)
  - CAER\_IMU9\_ITERATOR\_ALL\_END, [248](#)
  - CAER\_IMU9\_ITERATOR\_ALL\_START, [248](#)
  - CAER\_IMU9\_ITERATOR\_VALID\_END, [249](#)
  - CAER\_IMU9\_ITERATOR\_VALID\_START, [249](#)
  - CAER\_IMU9\_REVERSE\_ITERATOR\_ALL\_END, [249](#)
  - CAER\_IMU9\_REVERSE\_ITERATOR\_ALL\_START, [249](#)
  - CAER\_IMU9\_REVERSE\_ITERATOR\_VALID\_END, [250](#)
  - CAER\_IMU9\_REVERSE\_ITERATOR\_VALID\_START, [250](#)
  - caerIMU9Event, [250](#)
  - caerIMU9EventGetAccelX, [251](#)
  - caerIMU9EventGetAccelY, [251](#)



- caerIMU9EventGetAccelZ, [252](#)
- caerIMU9EventGetCompX, [252](#)
- caerIMU9EventGetCompY, [252](#)
- caerIMU9EventGetCompZ, [253](#)
- caerIMU9EventGetGyroX, [253](#)
- caerIMU9EventGetGyroY, [253](#)
- caerIMU9EventGetGyroZ, [255](#)
- caerIMU9EventGetTemp, [255](#)
- caerIMU9EventGetTimestamp, [255](#)
- caerIMU9EventGetTimestamp64, [256](#)
- caerIMU9EventInvalidate, [256](#)
- caerIMU9EventIsValid, [257](#)
- caerIMU9EventPacket, [251](#)
- caerIMU9EventPacketAllocate, [257](#)
- caerIMU9EventPacketGetEvent, [257](#)
- caerIMU9EventPacketGetEventConst, [258](#)
- caerIMU9EventSetAccelX, [258](#)
- caerIMU9EventSetAccelY, [258](#)
- caerIMU9EventSetAccelZ, [259](#)
- caerIMU9EventSetCompX, [259](#)
- caerIMU9EventSetCompY, [259](#)
- caerIMU9EventSetCompZ, [260](#)
- caerIMU9EventSetGyroX, [260](#)
- caerIMU9EventSetGyroY, [260](#)
- caerIMU9EventSetGyroZ, [261](#)
- caerIMU9EventSetTemp, [261](#)
- caerIMU9EventSetTimestamp, [261](#)
- caerIMU9EventValidate, [262](#)
- PACKED\_STRUCT, [262](#)
- LIBCAER\_HAVE\_OPENCV
  - libcaer.h, [375](#)
- LIBCAER\_HAVE\_SERIALDEV
  - libcaer.h, [375](#)
- LIBCAER\_NAME\_STRING
  - libcaer.h, [375](#)
- LIBCAER\_VERSION\_STRING
  - libcaer.h, [376](#)
- LIBCAER\_VERSION
  - libcaer.h, [376](#)
- libcaer.h, [372](#)
  - CLEAR\_NUMBITS16, [374](#)
  - CLEAR\_NUMBITS32, [374](#)
  - CLEAR\_NUMBITS8, [374](#)
  - caerByteArrayToInteger, [378](#)
  - caerIntegerToByteArray, [378](#)
  - caerStrEquals, [378](#)
  - caerStrEqualsUpTo, [380](#)
  - GET\_NUMBITS16, [374](#)
  - GET\_NUMBITS32, [374](#)
  - GET\_NUMBITS8, [374](#)
  - I16T, [375](#)
  - I32T, [375](#)
  - I64T, [375](#)
  - I8T, [375](#)
  - LIBCAER\_HAVE\_OPENCV, [375](#)
  - LIBCAER\_HAVE\_SERIALDEV, [375](#)
  - LIBCAER\_NAME\_STRING, [375](#)
  - LIBCAER\_VERSION\_STRING, [376](#)
  - LIBCAER\_VERSION, [376](#)
  - MASK\_NUMBITS32, [376](#)
  - MASK\_NUMBITS64, [376](#)
  - SET\_NUMBITS16, [376](#)
  - SET\_NUMBITS32, [376](#)
  - SET\_NUMBITS8, [376](#)
  - SWAP\_VAR, [377](#)
  - U16T, [377](#)
  - U32T, [377](#)
  - U64T, [377](#)
  - U8T, [377](#)
- log.h, [380](#)
  - caer\_log\_level, [381](#)
  - caerLog, [381](#)
  - caerLogFileDescriptorsGetFirst, [382](#)
  - caerLogFileDescriptorsGetSecond, [382](#)
  - caerLogFileDescriptorsSet, [382](#)
  - caerLogLevelGet, [383](#)
  - caerLogLevelSet, [383](#)
  - caerLogVAFull, [384](#)
  - caerLogVA, [383](#)
- MASK\_NUMBITS32
  - libcaer.h, [376](#)
- MASK\_NUMBITS64
  - libcaer.h, [376](#)
- network.h, [384](#)
- PACKED\_STRUCT
  - common.h, [181](#)
  - config.h, [192](#)
  - ear.h, [204](#)
  - frame.h, [231](#)
  - imu6.h, [245](#)
  - imu9.h, [262](#)
  - packetContainer.h, [271](#)
  - point1d.h, [283](#)
  - point2d.h, [295](#), [296](#)
  - point3d.h, [309](#)
  - point4d.h, [322](#)
  - polarity.h, [334](#)
  - sample.h, [345](#)
  - special.h, [359](#)
  - spike.h, [371](#)
- POINT1D\_SCALE\_MASK
  - point1d.h, [276](#)
- POINT1D\_SCALE\_SHIFT
  - point1d.h, [276](#)
- POINT1D\_TYPE\_MASK
  - point1d.h, [277](#)
- POINT1D\_TYPE\_SHIFT
  - point1d.h, [277](#)
- POINT2D\_SCALE\_MASK
  - point2d.h, [288](#)
- POINT2D\_SCALE\_SHIFT
  - point2d.h, [288](#)
- POINT2D\_TYPE\_MASK
  - point2d.h, [289](#)

- POINT2D\_TYPE\_SHIFT
  - point2d.h, [289](#)
- POINT3D\_SCALE\_MASK
  - point3d.h, [301](#)
- POINT3D\_SCALE\_SHIFT
  - point3d.h, [301](#)
- POINT3D\_TYPE\_MASK
  - point3d.h, [301](#)
- POINT3D\_TYPE\_SHIFT
  - point3d.h, [301](#)
- POINT4D\_SCALE\_MASK
  - point4d.h, [314](#)
- POINT4D\_SCALE\_SHIFT
  - point4d.h, [314](#)
- POINT4D\_TYPE\_MASK
  - point4d.h, [314](#)
- POINT4D\_TYPE\_SHIFT
  - point4d.h, [314](#)
- POLARITY\_MASK
  - polarity.h, [327](#)
- POLARITY\_SHIFT
  - polarity.h, [328](#)
- POLARITY\_X\_ADDR\_MASK
  - polarity.h, [328](#)
- POLARITY\_X\_ADDR\_SHIFT
  - polarity.h, [328](#)
- POLARITY\_Y\_ADDR\_MASK
  - polarity.h, [328](#)
- POLARITY\_Y\_ADDR\_SHIFT
  - polarity.h, [328](#)
- packetContainer.h
  - CAER\_EVENT\_PACKET\_CONTAINER\_CONST\_ITERATOR\_START, [264](#)
  - CAER\_EVENT\_PACKET\_CONTAINER\_ITERATOR\_END, [264](#)
  - CAER\_EVENT\_PACKET\_CONTAINER\_ITERATOR\_START, [265](#)
  - caerEventPacketContainer, [265](#)
  - caerEventPacketContainerAllocate, [265](#)
  - caerEventPacketContainerCopyAllEvents, [266](#)
  - caerEventPacketContainerCopyValidEvents, [266](#)
  - caerEventPacketContainerFindEventPacketByType, [266](#)
  - caerEventPacketContainerFindEventPacketByTypeConst, [267](#)
  - caerEventPacketContainerFree, [267](#)
  - caerEventPacketContainerGetEventPacket, [268](#)
  - caerEventPacketContainerGetEventPacketConst, [268](#)
  - caerEventPacketContainerGetEventPacketsNumber, [268](#)
  - caerEventPacketContainerGetEventsNumber, [269](#)
  - caerEventPacketContainerGetEventsValidNumber, [269](#)
  - caerEventPacketContainerGetHighestEventTimestamp, [269](#)
  - caerEventPacketContainerGetLowestEventTimestamp, [270](#)
  - caerEventPacketContainerSetEventPacket, [270](#)
  - caerEventPacketContainerSetEventPacketsNumber, [271](#)
  - caerEventPacketContainerUpdateStatistics, [271](#)
  - PACKED\_STRUCT, [271](#)
- point1d.h
  - CAER\_POINT1D\_CONST\_ITERATOR\_ALL\_START, [273](#)
  - CAER\_POINT1D\_CONST\_ITERATOR\_VALID\_START, [273](#)
  - CAER\_POINT1D\_CONST\_REVERSE\_ITERATOR\_OR\_ALL\_START, [273](#)
  - CAER\_POINT1D\_CONST\_REVERSE\_ITERATOR\_OR\_VALID\_START, [274](#)
  - CAER\_POINT1D\_ITERATOR\_ALL\_END, [274](#)
  - CAER\_POINT1D\_ITERATOR\_ALL\_START, [274](#)
  - CAER\_POINT1D\_ITERATOR\_VALID\_END, [275](#)
  - CAER\_POINT1D\_ITERATOR\_VALID\_START, [275](#)
  - CAER\_POINT1D\_REVERSE\_ITERATOR\_ALL\_END, [275](#)
  - CAER\_POINT1D\_REVERSE\_ITERATOR\_ALL\_START, [275](#)
  - CAER\_POINT1D\_REVERSE\_ITERATOR\_VALID\_END, [276](#)
  - CAER\_POINT1D\_REVERSE\_ITERATOR\_VALID\_START, [276](#)
  - caerPoint1DEvent, [277](#)
  - caerPoint1DEventGetScale, [277](#)
  - caerPoint1DEventGetTimestamp, [278](#)
  - caerPoint1DEventGetTimestamp64, [278](#)
  - caerPoint1DEventGetType, [278](#)
  - caerPoint1DEventGetX, [279](#)
  - caerPoint1DEventInvalidate, [279](#)
  - caerPoint1DEventsIsValid, [280](#)
  - caerPoint1DEventPacket, [277](#)
  - caerPoint1DEventPacketAllocate, [280](#)
  - caerPoint1DEventPacketGetEvent, [280](#)
  - caerPoint1DEventPacketGetEventConst, [281](#)
  - caerPoint1DEventSetScale, [281](#)
  - caerPoint1DEventSetTimestamp, [281](#)
  - caerPoint1DEventSetType, [282](#)
  - caerPoint1DEventSetX, [282](#)
  - caerPoint1DEventValidate, [282](#)
  - PACKED\_STRUCT, [283](#)
  - POINT1D\_SCALE\_MASK, [276](#)
  - POINT1D\_SCALE\_SHIFT, [276](#)
  - POINT1D\_TYPE\_MASK, [277](#)
  - POINT1D\_TYPE\_SHIFT, [277](#)
- point2d.h
  - CAER\_POINT2D\_CONST\_ITERATOR\_ALL\_START, [285](#)
  - CAER\_POINT2D\_CONST\_ITERATOR\_VALID\_START, [285](#)
  - CAER\_POINT2D\_CONST\_REVERSE\_ITERATOR\_OR\_ALL\_START, [285](#)
  - CAER\_POINT2D\_CONST\_REVERSE\_ITERATOR\_OR\_VALID\_START, [286](#)

- CAER\_POINT2D\_ITERATOR\_ALL\_END, [286](#)
- CAER\_POINT2D\_ITERATOR\_ALL\_START, [286](#)
- CAER\_POINT2D\_ITERATOR\_VALID\_END, [287](#)
- CAER\_POINT2D\_ITERATOR\_VALID\_START, [287](#)
- CAER\_POINT2D\_REVERSE\_ITERATOR\_ALL\_↔END, [287](#)
- CAER\_POINT2D\_REVERSE\_ITERATOR\_ALL\_↔START, [287](#)
- CAER\_POINT2D\_REVERSE\_ITERATOR\_VALI↔D\_END, [288](#)
- CAER\_POINT2D\_REVERSE\_ITERATOR\_VALI↔D\_START, [288](#)
- caerPoint2DEvent, [289](#)
- caerPoint2DEventGetScale, [289](#)
- caerPoint2DEventGetTimestamp, [290](#)
- caerPoint2DEventGetTimestamp64, [290](#)
- caerPoint2DEventGetType, [290](#)
- caerPoint2DEventGetX, [291](#)
- caerPoint2DEventGetY, [291](#)
- caerPoint2DEventInvalidate, [292](#)
- caerPoint2DEventIsValid, [292](#)
- caerPoint2DEventPacket, [289](#)
- caerPoint2DEventPacketAllocate, [292](#)
- caerPoint2DEventPacketGetEvent, [293](#)
- caerPoint2DEventPacketGetEventConst, [293](#)
- caerPoint2DEventSetScale, [294](#)
- caerPoint2DEventSetTimestamp, [294](#)
- caerPoint2DEventSetType, [294](#)
- caerPoint2DEventSetX, [295](#)
- caerPoint2DEventSetY, [295](#)
- caerPoint2DEventValidate, [295](#)
- PACKED\_STRUCT, [295](#), [296](#)
- POINT2D\_SCALE\_MASK, [288](#)
- POINT2D\_SCALE\_SHIFT, [288](#)
- POINT2D\_TYPE\_MASK, [289](#)
- POINT2D\_TYPE\_SHIFT, [289](#)
- point3d.h
  - CAER\_POINT3D\_CONST\_ITERATOR\_ALL\_ST↔ART, [297](#)
  - CAER\_POINT3D\_CONST\_ITERATOR\_VALID\_↔START, [298](#)
  - CAER\_POINT3D\_CONST\_REVERSE\_ITERAT↔OR\_ALL\_START, [298](#)
  - CAER\_POINT3D\_CONST\_REVERSE\_ITERAT↔OR\_VALID\_START, [298](#)
  - CAER\_POINT3D\_ITERATOR\_ALL\_END, [299](#)
  - CAER\_POINT3D\_ITERATOR\_ALL\_START, [299](#)
  - CAER\_POINT3D\_ITERATOR\_VALID\_END, [299](#)
  - CAER\_POINT3D\_ITERATOR\_VALID\_START, [299](#)
  - CAER\_POINT3D\_REVERSE\_ITERATOR\_ALL\_↔END, [300](#)
  - CAER\_POINT3D\_REVERSE\_ITERATOR\_ALL\_↔START, [300](#)
  - CAER\_POINT3D\_REVERSE\_ITERATOR\_VALI↔D\_END, [300](#)
  - CAER\_POINT3D\_REVERSE\_ITERATOR\_VALI↔D\_START, [300](#)
  - caerPoint3DEvent, [302](#)
  - caerPoint3DEventGetScale, [302](#)
  - caerPoint3DEventGetTimestamp, [302](#)
  - caerPoint3DEventGetTimestamp64, [303](#)
  - caerPoint3DEventGetType, [303](#)
  - caerPoint3DEventGetX, [303](#)
  - caerPoint3DEventGetY, [304](#)
  - caerPoint3DEventGetZ, [304](#)
  - caerPoint3DEventInvalidate, [305](#)
  - caerPoint3DEventIsValid, [305](#)
  - caerPoint3DEventPacket, [302](#)
  - caerPoint3DEventPacketAllocate, [305](#)
  - caerPoint3DEventPacketGetEvent, [306](#)
  - caerPoint3DEventPacketGetEventConst, [306](#)
  - caerPoint3DEventSetScale, [306](#)
  - caerPoint3DEventSetTimestamp, [307](#)
  - caerPoint3DEventSetType, [307](#)
  - caerPoint3DEventSetX, [307](#)
  - caerPoint3DEventSetY, [308](#)
  - caerPoint3DEventSetZ, [308](#)
  - caerPoint3DEventValidate, [308](#)
  - PACKED\_STRUCT, [309](#)
  - POINT3D\_SCALE\_MASK, [301](#)
  - POINT3D\_SCALE\_SHIFT, [301](#)
  - POINT3D\_TYPE\_MASK, [301](#)
  - POINT3D\_TYPE\_SHIFT, [301](#)
- point4d.h
  - CAER\_POINT4D\_CONST\_ITERATOR\_ALL\_ST↔ART, [310](#)
  - CAER\_POINT4D\_CONST\_ITERATOR\_VALID\_↔START, [311](#)
  - CAER\_POINT4D\_CONST\_REVERSE\_ITERAT↔OR\_ALL\_START, [311](#)
  - CAER\_POINT4D\_CONST\_REVERSE\_ITERAT↔OR\_VALID\_START, [311](#)
  - CAER\_POINT4D\_ITERATOR\_ALL\_END, [312](#)
  - CAER\_POINT4D\_ITERATOR\_ALL\_START, [312](#)
  - CAER\_POINT4D\_ITERATOR\_VALID\_END, [312](#)
  - CAER\_POINT4D\_ITERATOR\_VALID\_START, [312](#)
  - CAER\_POINT4D\_REVERSE\_ITERATOR\_ALL\_↔END, [313](#)
  - CAER\_POINT4D\_REVERSE\_ITERATOR\_ALL\_↔START, [313](#)
  - CAER\_POINT4D\_REVERSE\_ITERATOR\_VALI↔D\_END, [313](#)
  - CAER\_POINT4D\_REVERSE\_ITERATOR\_VALI↔D\_START, [313](#)
  - caerPoint4DEvent, [315](#)
  - caerPoint4DEventGetScale, [315](#)
  - caerPoint4DEventGetTimestamp, [315](#)
  - caerPoint4DEventGetTimestamp64, [316](#)
  - caerPoint4DEventGetType, [316](#)
  - caerPoint4DEventGetW, [316](#)
  - caerPoint4DEventGetX, [317](#)
  - caerPoint4DEventGetY, [317](#)

- caerPoint4DEventGetZ, 318
- caerPoint4DEventInvalidate, 318
- caerPoint4DEventIsValid, 318
- caerPoint4DEventPacket, 315
- caerPoint4DEventPacketAllocate, 319
- caerPoint4DEventPacketGetEvent, 319
- caerPoint4DEventPacketGetEventConst, 319
- caerPoint4DEventSetScale, 320
- caerPoint4DEventSetTimestamp, 320
- caerPoint4DEventSetType, 320
- caerPoint4DEventSetW, 321
- caerPoint4DEventSetX, 321
- caerPoint4DEventSetY, 321
- caerPoint4DEventSetZ, 322
- caerPoint4DEventValidate, 322
- PACKED\_STRUCT, 322
- POINT4D\_SCALE\_MASK, 314
- POINT4D\_SCALE\_SHIFT, 314
- POINT4D\_TYPE\_MASK, 314
- POINT4D\_TYPE\_SHIFT, 314
- polarity.h
  - CAER\_POLARITY\_CONST\_ITERATOR\_ALL\_START, 324
  - CAER\_POLARITY\_CONST\_ITERATOR\_VALID\_START, 324
  - CAER\_POLARITY\_CONST\_REVERSE\_ITERATOR\_ALL\_START, 325
  - CAER\_POLARITY\_CONST\_REVERSE\_ITERATOR\_VALID\_START, 325
  - CAER\_POLARITY\_ITERATOR\_ALL\_END, 325
  - CAER\_POLARITY\_ITERATOR\_ALL\_START, 326
  - CAER\_POLARITY\_ITERATOR\_VALID\_END, 326
  - CAER\_POLARITY\_ITERATOR\_VALID\_START, 326
  - CAER\_POLARITY\_REVERSE\_ITERATOR\_ALL\_END, 326
  - CAER\_POLARITY\_REVERSE\_ITERATOR\_ALL\_START, 327
  - CAER\_POLARITY\_REVERSE\_ITERATOR\_VALID\_END, 327
  - CAER\_POLARITY\_REVERSE\_ITERATOR\_VALID\_START, 327
  - caerPolarityEvent, 328
  - caerPolarityEventGetPolarity, 329
  - caerPolarityEventGetTimestamp, 329
  - caerPolarityEventGetTimestamp64, 330
  - caerPolarityEventGetX, 330
  - caerPolarityEventGetY, 330
  - caerPolarityEventInvalidate, 331
  - caerPolarityEventIsValid, 331
  - caerPolarityEventPacket, 329
  - caerPolarityEventPacketAllocate, 331
  - caerPolarityEventPacketGetEvent, 332
  - caerPolarityEventPacketGetEventConst, 332
  - caerPolarityEventSetPolarity, 333
  - caerPolarityEventSetTimestamp, 333
  - caerPolarityEventSetX, 333
  - caerPolarityEventSetY, 334
  - caerPolarityEventValidate, 334
  - PACKED\_STRUCT, 334
  - POLARITY\_MASK, 327
  - POLARITY\_SHIFT, 328
  - POLARITY\_X\_ADDR\_MASK, 328
  - POLARITY\_X\_ADDR\_SHIFT, 328
  - POLARITY\_Y\_ADDR\_MASK, 328
  - POLARITY\_Y\_ADDR\_SHIFT, 328
  - portable\_endian.h, 385
  - SAMPLE\_MASK
    - sample.h, 339
  - SAMPLE\_SHIFT
    - sample.h, 340
  - SAMPLE\_TYPE\_MASK
    - sample.h, 340
  - SAMPLE\_TYPE\_SHIFT
    - sample.h, 340
  - SET\_NUMBITS16
    - libcaer.h, 376
  - SET\_NUMBITS32
    - libcaer.h, 376
  - SET\_NUMBITS8
    - libcaer.h, 376
  - SPECIAL\_DATA\_MASK
    - special.h, 351
  - SPECIAL\_DATA\_SHIFT
    - special.h, 351
  - SPECIAL\_TYPE\_MASK
    - special.h, 351
  - SPECIAL\_TYPE\_SHIFT
    - special.h, 351
  - SPIKE\_CHIP\_ID\_MASK
    - spike.h, 364
  - SPIKE\_CHIP\_ID\_SHIFT
    - spike.h, 365
  - SPIKE\_NEURON\_ID\_MASK
    - spike.h, 365
  - SPIKE\_NEURON\_ID\_SHIFT
    - spike.h, 365
  - SPIKE\_SOURCE\_CORE\_ID\_MASK
    - spike.h, 365
  - SPIKE\_SOURCE\_CORE\_ID\_SHIFT
    - spike.h, 365
  - SWAP\_VAR
    - libcaer.h, 377
  - sample.h
    - CAER\_SAMPLE\_CONST\_ITERATOR\_ALL\_START, 336
    - CAER\_SAMPLE\_CONST\_ITERATOR\_VALID\_START, 336
    - CAER\_SAMPLE\_CONST\_REVERSE\_ITERATOR\_ALL\_START, 337
    - CAER\_SAMPLE\_CONST\_REVERSE\_ITERATOR\_VALID\_START, 337
    - CAER\_SAMPLE\_ITERATOR\_ALL\_END, 337
    - CAER\_SAMPLE\_ITERATOR\_ALL\_START, 338
    - CAER\_SAMPLE\_ITERATOR\_VALID\_END, 338
    - CAER\_SAMPLE\_ITERATOR\_VALID\_START, 338

- CAER\_SAMPLE\_REVERSE\_ITERATOR\_ALL\_↔  
END, 338
- CAER\_SAMPLE\_REVERSE\_ITERATOR\_ALL\_↔  
START, 339
- CAER\_SAMPLE\_REVERSE\_ITERATOR\_VALI↔  
D\_END, 339
- CAER\_SAMPLE\_REVERSE\_ITERATOR\_VALI↔  
D\_START, 339
- caerSampleEvent, 340
- caerSampleEventGetSample, 341
- caerSampleEventGetTimestamp, 341
- caerSampleEventGetTimestamp64, 341
- caerSampleEventGetType, 342
- caerSampleEventInvalidate, 342
- caerSampleEventsValid, 342
- caerSampleEventPacket, 340
- caerSampleEventPacketAllocate, 343
- caerSampleEventPacketGetEvent, 343
- caerSampleEventPacketGetEventConst, 344
- caerSampleEventSetSample, 344
- caerSampleEventSetTimestamp, 344
- caerSampleEventSetType, 345
- caerSampleEventValidate, 345
- PACKED\_STRUCT, 345
- SAMPLE\_MASK, 339
- SAMPLE\_SHIFT, 340
- SAMPLE\_TYPE\_MASK, 340
- SAMPLE\_TYPE\_SHIFT, 340
- serial.h
  - CAER\_HOST\_CONFIG\_SERIAL\_BAUD\_RATE↔  
\_12M, 161
  - CAER\_HOST\_CONFIG\_SERIAL\_BAUD\_RATE↔  
\_2M, 161
  - CAER\_HOST\_CONFIG\_SERIAL\_BAUD\_RATE↔  
\_4M, 162
  - CAER\_HOST\_CONFIG\_SERIAL\_BAUD\_RATE↔  
\_8M, 162
  - CAER\_HOST\_CONFIG\_SERIAL\_READ\_SIZE,  
162
  - CAER\_HOST\_CONFIG\_SERIAL, 161
  - caerDeviceOpenSerial, 162
- special.h
  - CAER\_SPECIAL\_CONST\_ITERATOR\_ALL\_ST↔  
ART, 347
  - CAER\_SPECIAL\_CONST\_ITERATOR\_VALID\_↔  
START, 348
  - CAER\_SPECIAL\_CONST\_REVERSE\_ITERAT↔  
OR\_ALL\_START, 348
  - CAER\_SPECIAL\_CONST\_REVERSE\_ITERAT↔  
OR\_VALID\_START, 348
  - CAER\_SPECIAL\_ITERATOR\_ALL\_END, 349
  - CAER\_SPECIAL\_ITERATOR\_ALL\_START, 349
  - CAER\_SPECIAL\_ITERATOR\_VALID\_END, 349
  - CAER\_SPECIAL\_ITERATOR\_VALID\_START,  
349
  - CAER\_SPECIAL\_REVERSE\_ITERATOR\_ALL\_↔  
END, 350
  - CAER\_SPECIAL\_REVERSE\_ITERATOR\_ALL\_↔  
START, 350
  - CAER\_SPECIAL\_REVERSE\_ITERATOR\_VALI↔  
D\_END, 350
  - CAER\_SPECIAL\_REVERSE\_ITERATOR\_VALI↔  
D\_START, 350
  - caer\_special\_event\_types, 352
  - caerSpecialEvent, 352
  - caerSpecialEventGetData, 353
  - caerSpecialEventGetTimestamp, 353
  - caerSpecialEventGetTimestamp64, 354
  - caerSpecialEventGetType, 354
  - caerSpecialEventInvalidate, 354
  - caerSpecialEventsValid, 355
  - caerSpecialEventPacket, 352
  - caerSpecialEventPacketAllocate, 355
  - caerSpecialEventPacketFindEventByType, 356
  - caerSpecialEventPacketFindEventByTypeConst,  
356
  - caerSpecialEventPacketFindValidEventByType,  
356
  - caerSpecialEventPacketFindValidEventByType↔  
Const, 357
  - caerSpecialEventPacketGetEvent, 357
  - caerSpecialEventPacketGetEventConst, 358
  - caerSpecialEventSetData, 358
  - caerSpecialEventSetTimestamp, 358
  - caerSpecialEventSetType, 359
  - caerSpecialEventValidate, 359
  - PACKED\_STRUCT, 359
  - SPECIAL\_DATA\_MASK, 351
  - SPECIAL\_DATA\_SHIFT, 351
  - SPECIAL\_TYPE\_MASK, 351
  - SPECIAL\_TYPE\_SHIFT, 351
- spike.h
  - CAER\_SPIKE\_CONST\_ITERATOR\_ALL\_START,  
361
  - CAER\_SPIKE\_CONST\_ITERATOR\_VALID\_ST↔  
ART, 361
  - CAER\_SPIKE\_CONST\_REVERSE\_ITERATOR↔  
\_ALL\_START, 362
  - CAER\_SPIKE\_CONST\_REVERSE\_ITERATOR↔  
\_VALID\_START, 362
  - CAER\_SPIKE\_ITERATOR\_ALL\_END, 362
  - CAER\_SPIKE\_ITERATOR\_ALL\_START, 363
  - CAER\_SPIKE\_ITERATOR\_VALID\_END, 363
  - CAER\_SPIKE\_ITERATOR\_VALID\_START, 363
  - CAER\_SPIKE\_REVERSE\_ITERATOR\_ALL\_END,  
363
  - CAER\_SPIKE\_REVERSE\_ITERATOR\_ALL\_ST↔  
ART, 364
  - CAER\_SPIKE\_REVERSE\_ITERATOR\_VALID\_↔  
END, 364
  - CAER\_SPIKE\_REVERSE\_ITERATOR\_VALID\_↔  
START, 364
  - caerSpikeEvent, 365
  - caerSpikeEventGetChipID, 366
  - caerSpikeEventGetNeuronID, 366
  - caerSpikeEventGetSourceCoreID, 367

- caerSpikeEventGetTimestamp, [367](#)
- caerSpikeEventGetTimestamp64, [367](#)
- caerSpikeEventInvalidate, [368](#)
- caerSpikeEventIsValid, [368](#)
- caerSpikeEventPacket, [366](#)
- caerSpikeEventPacketAllocate, [368](#)
- caerSpikeEventPacketGetEvent, [369](#)
- caerSpikeEventPacketGetEventConst, [369](#)
- caerSpikeEventSetChipID, [370](#)
- caerSpikeEventSetNeuronID, [370](#)
- caerSpikeEventSetSourceCoreID, [370](#)
- caerSpikeEventSetTimestamp, [371](#)
- caerSpikeEventValidate, [371](#)
- PACKED\_STRUCT, [371](#)
- SPIKE\_CHIP\_ID\_MASK, [364](#)
- SPIKE\_CHIP\_ID\_SHIFT, [365](#)
- SPIKE\_NEURON\_ID\_MASK, [365](#)
- SPIKE\_NEURON\_ID\_SHIFT, [365](#)
- SPIKE\_SOURCE\_CORE\_ID\_MASK, [365](#)
- SPIKE\_SOURCE\_CORE\_ID\_SHIFT, [365](#)
- TS\_OVERFLOW\_SHIFT
  - common.h, [167](#)
- U16T
  - libcaer.h, [377](#)
- U32T
  - libcaer.h, [377](#)
- U64T
  - libcaer.h, [377](#)
- U8T
  - libcaer.h, [377](#)
- usb.h
  - CAER\_HOST\_CONFIG\_USB\_BUFFER\_NUMBERS, [163](#)
  - CAER\_HOST\_CONFIG\_USB\_BUFFER\_SIZE, [163](#)
  - CAER\_HOST\_CONFIG\_USB, [163](#)
  - caerDeviceOpen, [164](#)
- VALID\_MARK\_MASK
  - common.h, [168](#)
- VALID\_MARK\_SHIFT
  - common.h, [168](#)