

libDaisy

Generated by Doxygen 1.8.18

1 libdaisy	1
1.1 Documentation available on our wiki!	1
1.2 Using libdaisy	1
1.2.1 daisy.h	2
1.2.2 daisy_seed.h	2
1.2.3 daisy_platform.h	2
2 Module Index	3
2.1 Modules	3
3 Namespace Index	5
3.1 Namespace List	5
4 Class Index	7
4.1 Class List	7
5 File Index	9
5.1 File List	9
6 Module Documentation	11
6.1 LIBDAISY	11
6.1.1 Detailed Description	11
6.2 HUMAN_INTERFACE	12
6.2.1 Detailed Description	12
6.3 AUDIO	13
6.3.1 Detailed Description	13
6.3.2 Typedef Documentation	13
6.3.2.1 dsy_audio_callback	13
6.3.2.2 dsy_audio_mc_callback	13
6.3.3 Enumeration Type Documentation	13
6.3.3.1 anonymous enum	13
6.3.4 Function Documentation	14
6.3.4.1 dsy_audio_enter_bypass()	14
6.3.4.2 dsy_audio_exit_bypass()	14
6.3.4.3 dsy_audio_init()	14
6.3.4.4 dsy_audio_passthru()	14
6.3.4.5 dsy_audio_set_blocksize()	14
6.3.4.6 dsy_audio_set_callback()	14
6.3.4.7 dsy_audio_set_mc_callback()	14
6.3.4.8 dsy_audio_silence()	15
6.3.4.9 dsy_audio_start()	15
6.3.4.10 dsy_audio_stop()	15
6.4 CONTROLS	16
6.4.1 Detailed Description	16

6.5 FEEDBACK	17
6.5.1 Detailed Description	17
6.6 EXTERNAL	18
6.6.1 Detailed Description	18
6.6.2 Enumeration Type Documentation	18
6.6.2.1 MidiMessageType	18
6.7 PERIPHERAL	19
6.7.1 Detailed Description	19
6.8 SERIAL	20
6.8.1 Detailed Description	21
6.8.2 Enumeration Type Documentation	21
6.8.2.1 anonymous enum	21
6.8.2.2 dsy_audio_bitdepth	21
6.8.2.3 dsy_audio_device	21
6.8.2.4 dsy_audio_dir	22
6.8.2.5 dsy_audio_sai	22
6.8.2.6 dsy_audio_samplerate	22
6.8.2.7 dsy_audio_sync	22
6.8.2.8 dsy_i2c_periph	23
6.8.2.9 dsy_i2c_pin	23
6.8.2.10 dsy_i2c_speed	23
6.8.2.11 dsy_qspi_device	23
6.8.2.12 dsy_qspi_mode	24
6.8.2.13 dsy_qspi_pin	24
6.8.2.14 dsy_sai_pin	24
6.8.2.15 SpiPeriph	25
6.8.2.16 SpiPin	25
6.8.3 Function Documentation	25
6.8.3.1 dsy_i2c_init()	25
6.8.3.2 dsy_qspi_deinit()	25
6.8.3.3 dsy_qspi_erase()	25
6.8.3.4 dsy_qspi_erasector()	26
6.8.3.5 dsy_qspi_init()	26
6.8.3.6 dsy_qspi_write()	26
6.8.3.7 dsy_qspi_writepage()	27
6.8.3.8 dsy_sai_init()	27
6.8.3.9 dsy_sai_init_from_handle()	28
6.8.4 Variable Documentation	28
6.8.4.1 kUartMaxBufferSize	28
6.9 ANALOG_DIGITAL_CONVERSION	29
6.9.1 Detailed Description	29
6.9.2 Enumeration Type Documentation	29

6.9.2.1 dsy_dac_bitdepth	29
6.9.2.2 dsy_dac_channel	29
6.9.2.3 dsy_dac_mode	30
6.9.3 Function Documentation	30
6.9.3.1 dsy_dac_init()	30
6.9.3.2 dsy_dac_start()	30
6.9.3.3 dsy_dac_write()	30
6.10 OTHER	31
6.10.1 Detailed Description	31
6.10.2 Enumeration Type Documentation	31
6.10.2.1 dsy_gpio_mode	31
6.10.2.2 dsy_gpio_pull	32
6.10.2.3 SdmmcBitWidth	32
6.10.2.4 SdmmcMode	32
6.10.2.5 SdmmcSpeed	32
6.10.3 Function Documentation	32
6.10.3.1 dsy_gpio_deinit()	33
6.10.3.2 dsy_gpio_init()	33
6.10.3.3 dsy_gpio_read()	33
6.10.3.4 dsy_gpio_toggle()	33
6.10.3.5 dsy_gpio_write()	33
6.10.3.6 dsy_tim_delay_ms()	34
6.10.3.7 dsy_tim_delay_tick()	34
6.10.3.8 dsy_tim_delay_us()	34
6.10.3.9 dsy_tim_get_ms()	34
6.10.3.10 dsy_tim_get_tick()	34
6.10.3.11 dsy_tim_get_us()	35
6.10.3.12 dsy_tim_init()	35
6.10.3.13 dsy_tim_start()	35
6.11 SYSTEM	36
6.11.1 Detailed Description	36
6.11.2 Function Documentation	36
6.11.2.1 dsy_dma_init()	36
6.11.2.2 dsy_system_delay()	36
6.11.2.3 dsy_system_getnow()	36
6.11.2.4 dsy_system_init()	36
6.11.2.5 dsy_system_jumpto()	37
6.11.2.6 dsy_system_jumptoqspi()	37
6.12 DEVICE	38
6.12.1 Detailed Description	38
6.13 SHIFTREGISTER	39
6.13.1 Detailed Description	39

6.13.2 Enumeration Type Documentation	39
6.13.2.1 anonymous enum	39
6.13.3 Function Documentation	39
6.13.3.1 dsy_sr_4021_init()	39
6.13.3.2 dsy_sr_4021_state()	40
6.13.3.3 dsy_sr_4021_update()	40
6.14 FLASH	41
6.14.1 Detailed Description	44
6.14.2 Macro Definition Documentation	44
6.14.2.1 BLOCK_ERASE_32K_CMD [1/2]	44
6.14.2.2 BLOCK_ERASE_32K_CMD [2/2]	44
6.14.2.3 CLEAR_FLAG_STATUS_REG_CMD [1/2]	44
6.14.2.4 CLEAR_FLAG_STATUS_REG_CMD [2/2]	45
6.14.2.5 DIE_ERASE_CMD [1/2]	45
6.14.2.6 DIE_ERASE_CMD [2/2]	45
6.14.2.7 DUAL_IN_FAST_PROG_CMD [1/2]	45
6.14.2.8 DUAL_IN_FAST_PROG_CMD [2/2]	45
6.14.2.9 DUAL_INOUT_FAST_READ_4_BYTE_ADDR_CMD [1/2]	45
6.14.2.10 DUAL_INOUT_FAST_READ_4_BYTE_ADDR_CMD [2/2]	45
6.14.2.11 DUAL_INOUT_FAST_READ_CMD [1/2]	45
6.14.2.12 DUAL_INOUT_FAST_READ_CMD [2/2]	45
6.14.2.13 DUAL_INOUT_FAST_READ_DTR_CMD [1/2]	45
6.14.2.14 DUAL_INOUT_FAST_READ_DTR_CMD [2/2]	45
6.14.2.15 DUAL_OUT_FAST_READ_4_BYTE_ADDR_CMD [1/2]	45
6.14.2.16 DUAL_OUT_FAST_READ_4_BYTE_ADDR_CMD [2/2]	46
6.14.2.17 DUAL_OUT_FAST_READ_CMD [1/2]	46
6.14.2.18 DUAL_OUT_FAST_READ_CMD [2/2]	46
6.14.2.19 DUAL_OUT_FAST_READ_DTR_CMD [1/2]	46
6.14.2.20 DUAL_OUT_FAST_READ_DTR_CMD [2/2]	46
6.14.2.21 ENTER_4_BYTE_ADDR_MODE_CMD [1/2]	46
6.14.2.22 ENTER_4_BYTE_ADDR_MODE_CMD [2/2]	46
6.14.2.23 ENTER_QUAD_CMD [1/2]	46
6.14.2.24 ENTER_QUAD_CMD [2/2]	46
6.14.2.25 EXIT_4_BYTE_ADDR_MODE_CMD [1/2]	46
6.14.2.26 EXIT_4_BYTE_ADDR_MODE_CMD [2/2]	46
6.14.2.27 EXIT_QUAD_CMD [1/2]	46
6.14.2.28 EXIT_QUAD_CMD [2/2]	47
6.14.2.29 EXT_DUAL_IN_FAST_PROG_CMD [1/2]	47
6.14.2.30 EXT_DUAL_IN_FAST_PROG_CMD [2/2]	47
6.14.2.31 EXT_QUAD_IN_FAST_PROG_CMD [1/2]	47
6.14.2.32 EXT_QUAD_IN_FAST_PROG_CMD [2/2]	47
6.14.2.33 FAST_READ_4_BYTE_ADDR_CMD [1/2]	47

6.14.2.34 FAST_READ_4_BYTE_ADDR_CMD [2/2]	47
6.14.2.35 FAST_READ_CMD [1/2]	47
6.14.2.36 FAST_READ_CMD [2/2]	47
6.14.2.37 FAST_READ_DTR_CMD [1/2]	47
6.14.2.38 FAST_READ_DTR_CMD [2/2]	47
6.14.2.39 IS25LP064A_EAR_HIGHEST_SE	47
6.14.2.40 IS25LP064A_EAR_LOWEST_SEG	48
6.14.2.41 IS25LP064A_EAR_SECOND_SEG	48
6.14.2.42 IS25LP064A_EAR_THIRD_SEG	48
6.14.2.43 IS25LP064A_EVCR_DTRP	48
6.14.2.44 IS25LP064A_EVCR_DUAL	48
6.14.2.45 IS25LP064A_EVCR_ODS	48
6.14.2.46 IS25LP064A_EVCR_QUAD	48
6.14.2.47 IS25LP064A_EVCR_RH	48
6.14.2.48 IS25LP064A_FSR_ERERR	48
6.14.2.49 IS25LP064A_FSR_ERSUS	48
6.14.2.50 IS25LP064A_FSR_NBADDR	48
6.14.2.51 IS25LP064A_FSR_PGERR	48
6.14.2.52 IS25LP064A_FSR_PGSUS	49
6.14.2.53 IS25LP064A_FSR_PRERR	49
6.14.2.54 IS25LP064A_FSR_READY	49
6.14.2.55 IS25LP064A_NVCR_DTRP	49
6.14.2.56 IS25LP064A_NVCR_DUAL	49
6.14.2.57 IS25LP064A_NVCR_NB_DUMMY	49
6.14.2.58 IS25LP064A_NVCR_NBADDR	49
6.14.2.59 IS25LP064A_NVCR_ODS	49
6.14.2.60 IS25LP064A_NVCR_QUAB	49
6.14.2.61 IS25LP064A_NVCR_RH	49
6.14.2.62 IS25LP064A_NVCR_SEGMENT	49
6.14.2.63 IS25LP064A_NVCR_XIP	49
6.14.2.64 IS25LP064A_SR_QE	50
6.14.2.65 IS25LP064A_SR_SRWREN	50
6.14.2.66 IS25LP064A_SR_WIP	50
6.14.2.67 IS25LP064A_SR_WREN	50
6.14.2.68 IS25LP064A_VCR_NB_DUMMY	50
6.14.2.69 IS25LP064A_VCR_WRAP	50
6.14.2.70 IS25LP064A_VCR_XIP	50
6.14.2.71 IS25LP080D_EAR_HIGHEST_SE	50
6.14.2.72 IS25LP080D_EAR_LOWEST_SEG	50
6.14.2.73 IS25LP080D_EAR_SECOND_SEG	50
6.14.2.74 IS25LP080D_EAR_THIRD_SEG	50
6.14.2.75 IS25LP080D_EVCR_DTRP	50

6.14.2.76 IS25LP080D_EVCR_DUAL	51
6.14.2.77 IS25LP080D_EVCR_ODS	51
6.14.2.78 IS25LP080D_EVCR_QUAD	51
6.14.2.79 IS25LP080D_EVCR_RH	51
6.14.2.80 IS25LP080D_FSR_ERERR	51
6.14.2.81 IS25LP080D_FSR_ERSUS	51
6.14.2.82 IS25LP080D_FSR_NBADDR	51
6.14.2.83 IS25LP080D_FSR_PGERR	51
6.14.2.84 IS25LP080D_FSR_PGSUS	51
6.14.2.85 IS25LP080D_FSR_PRERR	51
6.14.2.86 IS25LP080D_FSR_READY	51
6.14.2.87 IS25LP080D_NVCR_DTRP	51
6.14.2.88 IS25LP080D_NVCR_DUAL	52
6.14.2.89 IS25LP080D_NVCR_NB_DUMMY	52
6.14.2.90 IS25LP080D_NVCR_NBADDR	52
6.14.2.91 IS25LP080D_NVCR_ODS	52
6.14.2.92 IS25LP080D_NVCR_QUAB	52
6.14.2.93 IS25LP080D_NVCR_RH	52
6.14.2.94 IS25LP080D_NVCR_SEGMENT	52
6.14.2.95 IS25LP080D_NVCR_XIP	52
6.14.2.96 IS25LP080D_SR_QE	52
6.14.2.97 IS25LP080D_SR_SRWREN	52
6.14.2.98 IS25LP080D_SR_WIP	52
6.14.2.99 IS25LP080D_SR_WREN	52
6.14.2.100 IS25LP080D_VCR_NB_DUMMY	53
6.14.2.101 IS25LP080D_VCR_WRAP	53
6.14.2.102 IS25LP080D_VCR_XIP	53
6.14.2.103 MULTIPLE_IO_READ_ID_CMD [1/2]	53
6.14.2.104 MULTIPLE_IO_READ_ID_CMD [2/2]	53
6.14.2.105 PAGE_PROG_4_BYTE_ADDR_CMD [1/2]	53
6.14.2.106 PAGE_PROG_4_BYTE_ADDR_CMD [2/2]	53
6.14.2.107 PAGE_PROG_CMD [1/2]	53
6.14.2.108 PAGE_PROG_CMD [2/2]	53
6.14.2.109 PROG_ERASE_RESUME_CMD [1/2]	53
6.14.2.110 PROG_ERASE_RESUME_CMD [2/2]	53
6.14.2.111 PROG_ERASE_SUSPEND_CMD [1/2]	53
6.14.2.112 PROG_ERASE_SUSPEND_CMD [2/2]	54
6.14.2.113 PROG_OTP_ARRAY_CMD [1/2]	54
6.14.2.114 PROG_OTP_ARRAY_CMD [2/2]	54
6.14.2.115 QUAD_IN_FAST_PROG_4_BYTE_ADDR_CMD [1/2]	54
6.14.2.116 QUAD_IN_FAST_PROG_4_BYTE_ADDR_CMD [2/2]	54
6.14.2.117 QUAD_IN_FAST_PROG_CMD [1/2]	54

6.14.2.118 QUAD_IN_FAST_PROG_CMD [2/2]	54
6.14.2.119 QUAD_INOUT_FAST_READ_4_BYTE_ADDR_CMD [1/2]	54
6.14.2.120 QUAD_INOUT_FAST_READ_4_BYTE_ADDR_CMD [2/2]	54
6.14.2.121 QUAD_INOUT_FAST_READ_CMD [1/2]	54
6.14.2.122 QUAD_INOUT_FAST_READ_CMD [2/2]	54
6.14.2.123 QUAD_INOUT_FAST_READ_DTR_CMD [1/2]	54
6.14.2.124 QUAD_INOUT_FAST_READ_DTR_CMD [2/2]	55
6.14.2.125 QUAD_OUT_FAST_READ_4_BYTE_ADDR_CMD [1/2]	55
6.14.2.126 QUAD_OUT_FAST_READ_4_BYTE_ADDR_CMD [2/2]	55
6.14.2.127 QUAD_OUT_FAST_READ_CMD [1/2]	55
6.14.2.128 QUAD_OUT_FAST_READ_CMD [2/2]	55
6.14.2.129 QUAD_OUT_FAST_READ_DTR_CMD [1/2]	55
6.14.2.130 QUAD_OUT_FAST_READ_DTR_CMD [2/2]	55
6.14.2.131 READ_4_BYTE_ADDR_CMD [1/2]	55
6.14.2.132 READ_4_BYTE_ADDR_CMD [2/2]	55
6.14.2.133 READ_CMD [1/2]	55
6.14.2.134 READ_CMD [2/2]	55
6.14.2.135 READ_ENHANCED_VOL_CFG_REG_CMD [1/2]	55
6.14.2.136 READ_ENHANCED_VOL_CFG_REG_CMD [2/2]	56
6.14.2.137 READ_EXT_ADDR_REG_CMD [1/2]	56
6.14.2.138 READ_EXT_ADDR_REG_CMD [2/2]	56
6.14.2.139 READ_FLAG_STATUS_REG_CMD [1/2]	56
6.14.2.140 READ_FLAG_STATUS_REG_CMD [2/2]	56
6.14.2.141 READ_ID_CMD [1/2]	56
6.14.2.142 READ_ID_CMD [2/2]	56
6.14.2.143 READ_ID_CMD2 [1/2]	56
6.14.2.144 READ_ID_CMD2 [2/2]	56
6.14.2.145 READ_LOCK_REG_CMD [1/2]	56
6.14.2.146 READ_LOCK_REG_CMD [2/2]	56
6.14.2.147 READ_NONVOL_CFG_REG_CMD [1/2]	56
6.14.2.148 READ_NONVOL_CFG_REG_CMD [2/2]	57
6.14.2.149 READ_OTP_ARRAY_CMD [1/2]	57
6.14.2.150 READ_OTP_ARRAY_CMD [2/2]	57
6.14.2.151 READ_READ_PARAM_REG_CMD [1/2]	57
6.14.2.152 READ_READ_PARAM_REG_CMD [2/2]	57
6.14.2.153 READ_SERIAL_FLASH_DISCO_PARAM_CMD [1/2]	57
6.14.2.154 READ_SERIAL_FLASH_DISCO_PARAM_CMD [2/2]	57
6.14.2.155 READ_STATUS_REG_CMD [1/2]	57
6.14.2.156 READ_STATUS_REG_CMD [2/2]	57
6.14.2.157 RESET_ENABLE_CMD [1/2]	57
6.14.2.158 RESET_ENABLE_CMD [2/2]	57
6.14.2.159 RESET_MEMORY_CMD [1/2]	57

6.14.2.160 RESET_MEMORY_CMD [2/2]	58
6.14.2.161 SECTOR_ERASE_4_BYTE_ADDR_CMD [1/2]	58
6.14.2.162 SECTOR_ERASE_4_BYTE_ADDR_CMD [2/2]	58
6.14.2.163 SECTOR_ERASE_CMD [1/2]	58
6.14.2.164 SECTOR_ERASE_CMD [2/2]	58
6.14.2.165 SUBSECTOR_ERASE_4_BYTE_ADDR_CMD [1/2]	58
6.14.2.166 SUBSECTOR_ERASE_4_BYTE_ADDR_CMD [2/2]	58
6.14.2.167 SUBSECTOR_ERASE_CMD [1/2]	58
6.14.2.168 SUBSECTOR_ERASE_CMD [2/2]	58
6.14.2.169 SUBSECTOR_ERASE_QPI_CMD [1/2]	58
6.14.2.170 SUBSECTOR_ERASE_QPI_CMD [2/2]	58
6.14.2.171 WRITE_DISABLE_CMD [1/2]	58
6.14.2.172 WRITE_DISABLE_CMD [2/2]	59
6.14.2.173 WRITE_ENABLE_CMD [1/2]	59
6.14.2.174 WRITE_ENABLE_CMD [2/2]	59
6.14.2.175 WRITE_ENHANCED_VOL_CFG_REG_CMD [1/2]	59
6.14.2.176 WRITE_ENHANCED_VOL_CFG_REG_CMD [2/2]	59
6.14.2.177 WRITE_EXT_ADDR_REG_CMD [1/2]	59
6.14.2.178 WRITE_EXT_ADDR_REG_CMD [2/2]	59
6.14.2.179 WRITE_LOCK_REG_CMD [1/2]	59
6.14.2.180 WRITE_LOCK_REG_CMD [2/2]	59
6.14.2.181 WRITE_NONVOL_CFG_REG_CMD [1/2]	59
6.14.2.182 WRITE_NONVOL_CFG_REG_CMD [2/2]	59
6.14.2.183 WRITE_READ_PARAM_REG_CMD [1/2]	59
6.14.2.184 WRITE_READ_PARAM_REG_CMD [2/2]	60
6.14.2.185 WRITE_STATUS_REG_CMD [1/2]	60
6.14.2.186 WRITE_STATUS_REG_CMD [2/2]	60
6.15 CODEC	61
6.15.1 Detailed Description	61
6.15.2 Typedef Documentation	61
6.15.2.1 sa_audio_callback	61
6.15.3 Function Documentation	61
6.15.3.1 codec_ak4556_init()	61
6.15.3.2 codec_pcm3060_init()	61
6.15.3.3 codec_wm8731_enter_bypass()	62
6.15.3.4 codec_wm8731_exit_bypass()	62
6.15.3.5 codec_wm8731_init()	62
6.16 LED	63
6.16.1 Detailed Description	63
6.16.2 Enumeration Type Documentation	63
6.16.2.1 anonymous enum	63
6.16.3 Function Documentation	63

6.16.3.1 dsy_led_driver_color_by_name()	63
6.16.3.2 dsy_led_driver_init()	64
6.16.3.3 dsy_led_driver_set_led()	64
6.16.3.4 dsy_led_driver_update()	64
6.17 SDRAM	65
6.17.1 Detailed Description	65
6.17.2 Macro Definition Documentation	65
6.17.2.1 DSY_SDRAM_BSS	65
6.17.2.2 DSY_SDRAM_DATA	65
6.17.3 Enumeration Type Documentation	65
6.17.3.1 anonymous enum	65
6.17.3.2 dsy_sdram_pin	66
6.17.3.3 dsy_sdram_state	66
6.17.4 Function Documentation	66
6.17.4.1 dsy_sdram_init()	66
6.18 BOARDS	67
6.18.1 Detailed Description	67
6.18.2 Enumeration Type Documentation	67
6.18.2.1 anonymous enum	67
6.18.2.2 anonymous enum	68
6.18.2.3 anonymous enum	68
6.18.2.4 anonymous enum	68
6.18.3 Function Documentation	69
6.18.3.1 daisy_field_init()	69
6.18.3.2 f2s16()	70
6.18.3.3 f2s24()	70
6.18.3.4 s162f()	70
6.18.3.5 s242f()	70
6.19 UTILITY	71
6.19.1 Detailed Description	72
6.19.2 Macro Definition Documentation	72
6.19.2.1 BSP_SD_CardInfo	72
6.19.2.2 DMA_BUFFER_MEM_SECTION	72
6.19.2.3 DTCM_MEM_SECTION	72
6.19.2.4 MSD_ERROR	72
6.19.2.5 MSD_ERROR_SD_NOT_PRESENT	72
6.19.2.6 MSD_OK	72
6.19.2.7 SD_DATATIMEOUT	72
6.19.2.8 SD_NOT_PRESENT	72
6.19.2.9 SD_PRESENT	73
6.19.2.10 SD_TRANSFER_BUSY	73
6.19.2.11 SD_TRANSFER_OK	73

6.19.3 Enumeration Type Documentation	73
6.19.3.1 dsy_gpio_port	73
6.19.4 Function Documentation	73
6.19.4.1 BSP_SD_AbortCallback()	73
6.19.4.2 BSP_SD_Erase()	73
6.19.4.3 BSP_SD_GetCardInfo()	74
6.19.4.4 BSP_SD_GetCardState()	74
6.19.4.5 BSP_SD_Init()	74
6.19.4.6 BSP_SD_IsDetected()	74
6.19.4.7 BSP_SD_ITConfig()	75
6.19.4.8 BSP_SD_ReadBlocks()	75
6.19.4.9 BSP_SD_ReadBlocks_DMA()	75
6.19.4.10 BSP_SD_ReadCpltCallback()	75
6.19.4.11 BSP_SD_WriteBlocks()	75
6.19.4.12 BSP_SD_WriteBlocks_DMA()	76
6.19.4.13 BSP_SD_WriteCpltCallback()	76
6.19.4.14 cube()	76
6.19.4.15 dsy_get_unique_id()	77
6.19.4.16 dsy_hal_map_get_i2c()	77
6.19.4.17 dsy_hal_map_get_pin()	77
6.19.4.18 dsy_hal_map_get_port()	77
6.19.4.19 dsy_pin()	78
6.19.4.20 dsy_pin_cmp()	78
6.19.5 Variable Documentation	78
6.19.5.1 Font_11x18	78
6.19.5.2 Font_16x26	78
6.19.5.3 Font_6x8	78
6.19.5.4 Font_7x10	78
6.19.5.5 hi2c1	78
6.19.5.6 hi2c2	79
6.19.5.7 hi2c3	79
6.19.5.8 hi2c4	79
6.20 USBDCDC_IF	80
6.20.1 Detailed Description	80
6.21 USBDCDC_IF_Exported_Defines	81
6.22 USBDCDC_IF_Exported_Types	82
6.22.1 Detailed Description	82
6.22.2 Typedef Documentation	82
6.22.2.1 CDC_ReceiveCallback	82
6.23 USBDCDC_IF_Exported_Macros	83
6.24 USBDCDC_IF_Exported_Variables	84
6.24.1 Detailed Description	84

6.24.2 Variable Documentation	84
6.24.2.1 USBD_Interface_fops_FS	84
6.24.2.2 USBD_Interface_fops_HS	84
6.25 USBD_CDC_IF_Exported_FunctionsPrototype	85
6.25.1 Detailed Description	85
6.25.2 Function Documentation	85
6.25.2.1 CDC_Set_Rx_Callback_FS()	85
6.25.2.2 CDC_Transmit_FS()	85
6.25.2.3 CDC_Transmit_HS()	85
6.26 USBD_CONF	86
6.26.1 Detailed Description	86
6.27 USBD_CONF_Exported_Variables	87
6.28 USBD_CONF_Exported_Defines	88
6.28.1 Detailed Description	88
6.28.2 Macro Definition Documentation	88
6.28.2.1 DEVICE_FS	88
6.28.2.2 DEVICE_HS	88
6.28.2.3 USBD_DEBUG_LEVEL	88
6.28.2.4 USBD_LPM_ENABLED	88
6.28.2.5 USBD_MAX_NUM_CONFIGURATION	88
6.28.2.6 USBD_MAX_NUM_INTERFACES	88
6.28.2.7 USBD_MAX_STR_DESC_SIZ	88
6.28.2.8 USBD_SELF_POWERED	89
6.28.2.9 USBD_SUPPORT_USER_STRING	89
6.29 USBD_CONF_Exported_Macros	90
6.29.1 Detailed Description	90
6.29.2 Macro Definition Documentation	90
6.29.2.1 USBD_DbgLog	90
6.29.2.2 USBD_Delay	90
6.29.2.3 USBD_ErrLog	90
6.29.2.4 USBD_free	90
6.29.2.5 USBD_malloc	90
6.29.2.6 USBD_memcpy	91
6.29.2.7 USBD_memset	91
6.29.2.8 USBD_UsrLog	91
6.30 USBD_CONF_Exported_Types	92
6.31 USBD_CONF_Exported_FunctionsPrototype	93
6.32 USBD_DESC	94
6.32.1 Detailed Description	94
6.33 USBD_DESC_Exported_Constants	95
6.33.1 Detailed Description	95
6.33.2 Macro Definition Documentation	95

6.33.2.1 DEVICE_ID1	95
6.33.2.2 DEVICE_ID2	95
6.33.2.3 DEVICE_ID3	95
6.33.2.4 USB_SIZ_STRING_SERIAL	95
6.34 USBD_DESC_Exported_Defines	96
6.35 USBD_DESC_Exported_TypesDefinitions	97
6.36 USBD_DESC_Exported_Macros	98
6.37 USBD_DESC_Exported_Variables	99
6.37.1 Detailed Description	99
6.37.2 Variable Documentation	99
6.37.2.1 FS_Desc	99
6.37.2.2 HS_Desc	99
6.38 USBD_DESC_Exported_FunctionsPrototype	100
6.39 Externals	101
6.40 STM32_USB_OTG_DEVICE_LIBRARY	102
6.40.1 Detailed Description	102
6.41 USBD_OTG_DRIVER	103
6.41.1 Detailed Description	103
7 Namespace Documentation	105
7.1 daisy Namespace Reference	105
7.1.1 Detailed Description	106
8 Class Documentation	107
8.1 daisy::AdcChannelConfig Struct Reference	107
8.1.1 Detailed Description	107
8.1.2 Member Enumeration Documentation	107
8.1.2.1 MuxPin	107
8.1.3 Member Function Documentation	108
8.1.3.1 InitMux()	108
8.1.3.2 InitSingle()	108
8.1.4 Member Data Documentation	108
8.1.4.1 mux_channels_	108
8.1.4.2 mux_pin_	108
8.1.4.3 pin_	108
8.2 daisy::AdcHandle Class Reference	109
8.2.1 Detailed Description	109
8.2.2 Member Enumeration Documentation	109
8.2.2.1 OverSampling	109
8.2.3 Member Function Documentation	109
8.2.3.1 Get()	110
8.2.3.2 GetFloat()	110
8.2.3.3 GetMux()	110

8.2.3.4 GetMuxFloat()	110
8.2.3.5 GetMuxPtr()	111
8.2.3.6 GetPtr()	111
8.2.3.7 Init()	111
8.2.3.8 Start()	112
8.2.3.9 Stop()	112
8.3 daisy::AnalogControl Class Reference	112
8.3.1 Detailed Description	112
8.3.2 Constructor & Destructor Documentation	112
8.3.2.1 AnalogControl()	112
8.3.2.2 ~AnalogControl()	112
8.3.3 Member Function Documentation	113
8.3.3.1 Init()	113
8.3.3.2 InitBipolarCv()	113
8.3.3.3 Process()	113
8.3.3.4 Value()	113
8.4 codec_frame_t Struct Reference	113
8.4.1 Detailed Description	114
8.4.2 Member Data Documentation	114
8.4.2.1 l	114
8.4.2.2 r	114
8.5 color Struct Reference	114
8.5.1 Detailed Description	114
8.5.2 Member Data Documentation	114
8.5.2.1 blue	114
8.5.2.2 green	114
8.5.2.3 red	115
8.6 daisy::Color Class Reference	115
8.6.1 Detailed Description	115
8.6.2 Member Enumeration Documentation	115
8.6.2.1 PresetColor	115
8.6.3 Member Function Documentation	115
8.6.3.1 Blue()	116
8.6.3.2 Green()	116
8.6.3.3 Init() [1/2]	116
8.6.3.4 Init() [2/2]	116
8.6.3.5 Red()	116
8.7 daisy::ControlChangeEvent Struct Reference	116
8.7.1 Detailed Description	116
8.7.2 Member Data Documentation	117
8.7.2.1 channel	117
8.7.2.2 control_number	117

8.7.2.3 value	117
8.8 daisy::daisy_field Struct Reference	117
8.8.1 Detailed Description	117
8.8.2 Member Data Documentation	117
8.8.2.1 cvs	117
8.8.2.2 gate_in	117
8.8.2.3 gate_out	117
8.8.2.4 keyboard_sr	118
8.8.2.5 knobs	118
8.8.2.6 seed	118
8.8.2.7 switches	118
8.9 daisy::DaisyPatch Class Reference	118
8.9.1 Detailed Description	119
8.9.2 Member Enumeration Documentation	119
8.9.2.1 Ctrl	119
8.9.2.2 GateInput	119
8.9.3 Constructor & Destructor Documentation	119
8.9.3.1 DaisyPatch()	119
8.9.3.2 ~DaisyPatch()	119
8.9.4 Member Function Documentation	119
8.9.4.1 AudioBlockSize()	120
8.9.4.2 AudioCallbackRate()	120
8.9.4.3 AudioSampleRate()	120
8.9.4.4 ChangeAudioCallback()	120
8.9.4.5 DebounceControls()	120
8.9.4.6 DelayMs()	120
8.9.4.7 DisplayControls()	120
8.9.4.8 GetCtrlValue()	120
8.9.4.9 Init()	121
8.9.4.10 SetAudioBlockSize()	121
8.9.4.11 StartAdc()	121
8.9.4.12 StartAudio()	121
8.9.4.13 UpdateAnalogControls()	121
8.9.5 Member Data Documentation	121
8.9.5.1 controls	121
8.9.5.2 display	121
8.9.5.3 encoder	121
8.9.5.4 gate_input	122
8.9.5.5 gate_output	122
8.9.5.6 midi	122
8.9.5.7 seed	122
8.10 daisy::DaisyPetal Class Reference	122

8.10.1 Detailed Description	123
8.10.2 Member Enumeration Documentation	123
8.10.2.1 FootswitchLed	123
8.10.2.2 Knob	123
8.10.2.3 RingLed	124
8.10.2.4 Sw	124
8.10.3 Constructor & Destructor Documentation	124
8.10.3.1 DaisyPetal()	124
8.10.3.2 ~DaisyPetal()	124
8.10.4 Member Function Documentation	124
8.10.4.1 AudioBlockSize()	125
8.10.4.2 AudioCallbackRate()	125
8.10.4.3 AudioSampleRate()	125
8.10.4.4 ChangeAudioCallback()	125
8.10.4.5 ClearLeds()	125
8.10.4.6 DebounceControls()	125
8.10.4.7 DelayMs()	125
8.10.4.8 GetExpression()	125
8.10.4.9 GetKnobValue()	125
8.10.4.10 Init()	126
8.10.4.11 SetAudioBlockSize()	126
8.10.4.12 SetFootswitchLed()	126
8.10.4.13 SetRingLed()	126
8.10.4.14 StartAdc()	127
8.10.4.15 StartAudio()	127
8.10.4.16 UpdateAnalogControls()	127
8.10.4.17 UpdateLeds()	127
8.10.5 Member Data Documentation	127
8.10.5.1 encoder	127
8.10.5.2 expression	127
8.10.5.3 footswitch_led	127
8.10.5.4 knob	127
8.10.5.5 ring_led	127
8.10.5.6 seed	127
8.10.5.7 switches	128
8.11 daisy::DaisyPod Class Reference	128
8.11.1 Detailed Description	128
8.11.2 Member Enumeration Documentation	129
8.11.2.1 Knob	129
8.11.2.2 Sw	129
8.11.3 Member Function Documentation	129
8.11.3.1 AudioBlockSize()	129

8.11.3.2 AudioCallbackRate()	129
8.11.3.3 AudioSampleRate()	129
8.11.3.4 ChangeAudioCallback()	129
8.11.3.5 ClearLeds()	130
8.11.3.6 DebounceControls()	130
8.11.3.7 DelayMs()	130
8.11.3.8 GetKnobValue()	130
8.11.3.9 Init()	130
8.11.3.10 SetAudioBlockSize()	130
8.11.3.11 StartAdc()	130
8.11.3.12 StartAudio()	130
8.11.3.13 UpdateAnalogControls()	131
8.11.3.14 UpdateLeds()	131
8.11.4 Member Data Documentation	131
8.11.4.1 button1	131
8.11.4.2 button2	131
8.11.4.3 buttons	131
8.11.4.4 encoder	131
8.11.4.5 knob1	131
8.11.4.6 knob2	131
8.11.4.7 knobs	131
8.11.4.8 led1	131
8.11.4.9 led2	132
8.11.4.10 seed	132
8.11.5 autotoc_md8	132
8.12 daisy::DaisySeed Class Reference	132
8.12.1 Detailed Description	132
8.12.2 Member Function Documentation	132
8.12.2.1 AudioSampleRate()	132
8.12.2.2 Configure()	133
8.12.2.3 GetPin()	133
8.12.2.4 Init()	133
8.12.2.5 SetAudioBlockSize()	133
8.12.2.6 SetLed()	133
8.12.2.7 SetTestPoint()	133
8.12.2.8 StartAudio()	133
8.12.3 Member Data Documentation	133
8.12.3.1 adc	133
8.12.3.2 audio_handle	133
8.12.3.3 dac_handle	134
8.12.3.4 i2c1_handle	134
8.12.3.5 i2c2_handle	134

8.12.3.6 qspi_handle	134
8.12.3.7 sai_handle	134
8.12.3.8 sdram_handle	134
8.12.3.9 usb_handle	134
8.13 dsy_audio_handle Struct Reference	134
8.13.1 Detailed Description	134
8.13.2 Member Data Documentation	134
8.13.2.1 block_size	135
8.13.2.2 dev0_i2c	135
8.13.2.3 dev1_i2c	135
8.13.2.4 sai	135
8.14 dsy_dac_handle Struct Reference	135
8.14.1 Detailed Description	135
8.14.2 Member Data Documentation	135
8.14.2.1 bitdepth	135
8.14.2.2 mode	135
8.14.2.3 pin_config	135
8.15 dsy_gpio Struct Reference	136
8.15.1 Detailed Description	136
8.15.2 Member Data Documentation	136
8.15.2.1 mode	136
8.15.2.2 pin	136
8.15.2.3 pull	136
8.16 dsy_gpio_pin Struct Reference	136
8.16.1 Detailed Description	136
8.16.2 Member Data Documentation	136
8.16.2.1 pin	136
8.16.2.2 port	137
8.17 dsy_i2c_handle Struct Reference	137
8.17.1 Detailed Description	137
8.17.2 Member Data Documentation	137
8.17.2.1 periph	137
8.17.2.2 pin_config	137
8.17.2.3 speed	137
8.18 dsy_qspi_handle Struct Reference	137
8.18.1 Detailed Description	137
8.18.2 Member Data Documentation	138
8.18.2.1 device	138
8.18.2.2 mode	138
8.18.2.3 pin_config	138
8.19 dsy_sai_handle Struct Reference	138
8.19.1 Detailed Description	138

8.19.2 Member Data Documentation	138
8.19.2.1 a_direction	138
8.19.2.2 b_direction	138
8.19.2.3 bitdepth	139
8.19.2.4 device	139
8.19.2.5 init	139
8.19.2.6 sai1_pin_config	139
8.19.2.7 sai2_pin_config	139
8.19.2.8 samplerate	139
8.19.2.9 sync_config	139
8.20 DSY_SD_CardInfoTypeDef Struct Reference	139
8.20.1 Detailed Description	139
8.20.2 Member Data Documentation	140
8.20.2.1 BlockNbr	140
8.20.2.2 BlockSize	140
8.20.2.3 CardSpeed	140
8.20.2.4 CardType	140
8.20.2.5 CardVersion	140
8.20.2.6 Class	140
8.20.2.7 LogBlockNbr	140
8.20.2.8 LogBlockSize	140
8.20.2.9 RelCardAdd	140
8.21 dsy_sdram_handle Struct Reference	140
8.21.1 Detailed Description	141
8.21.2 Member Data Documentation	141
8.21.2.1 pin_config	141
8.21.2.2 state	141
8.22 dsy_sr_4021_handle Struct Reference	141
8.22.1 Detailed Description	141
8.22.2 Member Data Documentation	141
8.22.2.1 clk	141
8.22.2.2 cs	141
8.22.2.3 data	142
8.22.2.4 num_daisy chained	142
8.22.2.5 num_parallel	142
8.22.2.6 pin_config	142
8.22.2.7 states	142
8.23 daisy::Encoder Class Reference	142
8.23.1 Detailed Description	142
8.23.2 Member Function Documentation	142
8.23.2.1 Debounce()	143
8.23.2.2 FallingEdge()	143

8.23.2.3 Increment()	143
8.23.2.4 Init()	143
8.23.2.5 Pressed()	143
8.23.2.6 RisingEdge()	143
8.23.2.7 TimeHeldMs()	143
8.24 FontDef Struct Reference	143
8.24.1 Detailed Description	143
8.24.2 Member Data Documentation	144
8.24.2.1 data	144
8.24.2.2 FontHeight	144
8.24.2.3 FontWidth	144
8.25 daisy::GateIn Class Reference	144
8.25.1 Detailed Description	144
8.25.2 Constructor & Destructor Documentation	144
8.25.2.1 GateIn()	144
8.25.2.2 ~GateIn()	145
8.25.3 Member Function Documentation	145
8.25.3.1 Init()	145
8.25.3.2 Trig()	145
8.26 daisy::Led Class Reference	145
8.26.1 Detailed Description	145
8.26.2 Member Function Documentation	145
8.26.2.1 Init()	146
8.26.2.2 Set()	147
8.26.2.3 Update()	147
8.27 daisy::MidiEvent Struct Reference	147
8.27.1 Detailed Description	147
8.27.2 Member Function Documentation	147
8.27.2.1 AsControlChange()	147
8.27.2.2 AsNoteOn()	148
8.27.3 Member Data Documentation	148
8.27.3.1 channel	148
8.27.3.2 data	148
8.27.3.3 type	148
8.28 daisy::MidiHandler Class Reference	148
8.28.1 Detailed Description	148
8.28.2 Member Enumeration Documentation	149
8.28.2.1 MidiInputMode	149
8.28.2.2 MidiOutputMode	149
8.28.3 Member Function Documentation	149
8.28.3.1 HasEvents()	149
8.28.3.2 Init()	149

8.28.3.3 Listen()	150
8.28.3.4 Parse()	150
8.28.3.5 PopEvent()	150
8.28.3.6 StartReceive()	150
8.29 daisy::NoteOnEvent Struct Reference	150
8.29.1 Detailed Description	150
8.29.2 Member Data Documentation	150
8.29.2.1 channel	150
8.29.2.2 note	151
8.29.2.3 velocity	151
8.30 daisy::OledDisplay Class Reference	151
8.30.1 Detailed Description	151
8.30.2 Member Enumeration Documentation	151
8.30.2.1 Pins	151
8.30.3 Member Function Documentation	151
8.30.3.1 DrawPixel()	152
8.30.3.2 Fill()	152
8.30.3.3 Init()	152
8.30.3.4 SetCursor()	152
8.30.3.5 Update()	152
8.30.3.6 WriteChar()	153
8.30.3.7 WriteString()	153
8.31 daisy::Parameter Class Reference	153
8.31.1 Detailed Description	154
8.31.2 Member Enumeration Documentation	154
8.31.2.1 Curve	154
8.31.3 Constructor & Destructor Documentation	154
8.31.3.1 Parameter()	154
8.31.3.2 ~Parameter()	154
8.31.4 Member Function Documentation	154
8.31.4.1 Init()	154
8.31.4.2 Process()	155
8.31.4.3 Value()	155
8.32 daisy::RgbLed Class Reference	155
8.32.1 Detailed Description	155
8.32.2 Member Function Documentation	155
8.32.2.1 Init()	155
8.32.2.2 Set()	156
8.32.2.3 SetColor()	156
8.32.2.4 Update()	156
8.33 daisy::RingBuffer< T, size > Class Template Reference	156
8.33.1 Detailed Description	157

8.33.2 Member Function Documentation	157
8.33.2.1 capacity()	157
8.33.2.2 Flush()	157
8.33.2.3 ImmediateRead() [1/2]	157
8.33.2.4 ImmediateRead() [2/2]	157
8.33.2.5 Init()	157
8.33.2.6 Overwrite() [1/2]	158
8.33.2.7 Overwrite() [2/2]	158
8.33.2.8 Read()	158
8.33.2.9 readable()	158
8.33.2.10 Swallow()	158
8.33.2.11 writable()	159
8.33.2.12 Write()	159
8.34 daisy::RingBuffer< T, 0 > Class Template Reference	159
8.34.1 Detailed Description	159
8.34.2 Member Function Documentation	159
8.34.2.1 capacity()	160
8.34.2.2 Flush()	160
8.34.2.3 ImmediateRead() [1/2]	160
8.34.2.4 ImmediateRead() [2/2]	160
8.34.2.5 Init()	160
8.34.2.6 Overwrite() [1/2]	160
8.34.2.7 Overwrite() [2/2]	161
8.34.2.8 Read()	161
8.34.2.9 readable()	161
8.34.2.10 writable()	161
8.34.2.11 Write()	161
8.35 daisy::SdmmcHandler Class Reference	161
8.35.1 Detailed Description	162
8.35.2 Member Function Documentation	162
8.35.2.1 Init()	162
8.36 daisy::SdmmcHandlerInit Struct Reference	162
8.36.1 Detailed Description	162
8.36.2 Member Data Documentation	162
8.36.2.1 bitdepth	162
8.36.2.2 speed	162
8.37 ShiftRegister595 Class Reference	162
8.37.1 Detailed Description	163
8.37.2 Member Enumeration Documentation	163
8.37.2.1 Pins	163
8.37.3 Member Function Documentation	163
8.37.3.1 Init()	163

8.37.3.2 Set()	163
8.37.3.3 Write()	164
8.38 daisy::SpiHandle Class Reference	164
8.38.1 Detailed Description	164
8.38.2 Member Function Documentation	164
8.38.2.1 BlockingTransmit()	164
8.38.2.2 Init()	164
8.39 daisy::Switch Class Reference	164
8.39.1 Detailed Description	165
8.39.2 Member Enumeration Documentation	165
8.39.2.1 Polarity	165
8.39.2.2 Pull	165
8.39.2.3 Type	166
8.39.3 Member Function Documentation	166
8.39.3.1 Debounce()	166
8.39.3.2 FallingEdge()	166
8.39.3.3 Init() [1/2]	166
8.39.3.4 Init() [2/2]	166
8.39.3.5 Pressed()	167
8.39.3.6 RisingEdge()	167
8.39.3.7 TimeHeldMs()	167
8.40 daisy::UartHandler Class Reference	167
8.40.1 Detailed Description	168
8.40.2 Member Function Documentation	168
8.40.2.1 CheckError()	168
8.40.2.2 FlushRx()	168
8.40.2.3 Init()	168
8.40.2.4 PollReceive()	168
8.40.2.5 PollTx()	169
8.40.2.6 PopRx()	169
8.40.2.7 Readable()	169
8.40.2.8 RxActive()	169
8.40.2.9 StartRx()	169
8.41 UsbHandle Class Reference	170
8.41.1 Detailed Description	170
8.41.2 Member Typedef Documentation	170
8.41.2.1 ReceiveCallback [1/2]	170
8.41.2.2 ReceiveCallback [2/2]	171
8.41.3 Member Enumeration Documentation	171
8.41.3.1 UsbPeriph [1/2]	171
8.41.3.2 UsbPeriph [2/2]	171
8.41.4 Member Function Documentation	171

8.41.4.1 Init() [1/2]	171
8.41.4.2 Init() [2/2]	172
8.41.4.3 SetReceiveCallback() [1/2]	172
8.41.4.4 SetReceiveCallback() [2/2]	172
8.41.4.5 TransmitExternal() [1/2]	172
8.41.4.6 TransmitExternal() [2/2]	172
8.41.4.7 TransmitInternal() [1/2]	173
8.41.4.8 TransmitInternal() [2/2]	173
8.42 WAV_FormatTypeDef Struct Reference	173
8.42.1 Detailed Description	174
8.42.2 Member Data Documentation	174
8.42.2.1 AudioFormat	174
8.42.2.2 BitPerSample	174
8.42.2.3 BlockAlign	174
8.42.2.4 ByteRate	174
8.42.2.5 ChunkId	174
8.42.2.6 FileFormat	174
8.42.2.7 FileSize	174
8.42.2.8 NbrChannels	174
8.42.2.9 SampleRate	174
8.42.2.10 SubChunk1ID	174
8.42.2.11 SubChunk1Size	175
8.42.2.12 SubChunk2ID	175
8.42.2.13 SubCHunk2Size	175
8.43 daisy::WavFileInfo Struct Reference	175
8.43.1 Detailed Description	175
8.43.2 Member Data Documentation	175
8.43.2.1 name	175
8.43.2.2 raw_data	175
8.44 daisy::WavPlayer Class Reference	175
8.44.1 Detailed Description	176
8.44.2 Member Function Documentation	176
8.44.2.1 Close()	176
8.44.2.2 GetCurrentFile()	176
8.44.2.3 GetLooping()	176
8.44.2.4 GetNumberFiles()	176
8.44.2.5 Init()	176
8.44.2.6 Open()	177
8.44.2.7 Prepare()	178
8.44.2.8 Restart()	178
8.44.2.9 SetLooping()	178
8.44.2.10 Stream()	178

9 File Documentation	179
9.1 src/ffconf.h File Reference	179
9.1.1 Detailed Description	180
9.1.2 Macro Definition Documentation	180
9.1.2.1 _CODE_PAGE	180
9.1.2.2 _FFCONF	180
9.1.2.3 _FS_EXFAT	180
9.1.2.4 _FS_LOCK	180
9.1.2.5 _FS_MINIMIZE	180
9.1.2.6 _FS_NOFSINFO	181
9.1.2.7 _FS_NORTC	181
9.1.2.8 _FS_READONLY	181
9.1.2.9 _FS_REENTRANT	181
9.1.2.10 _FS_RPATH	181
9.1.2.11 _FS_TIMEOUT	181
9.1.2.12 _FS_TINY	181
9.1.2.13 _LFN_UNICODE	181
9.1.2.14 _MAX_LFN	181
9.1.2.15 _MAX_SS	182
9.1.2.16 _MIN_SS	182
9.1.2.17 _MULTI_PARTITION	182
9.1.2.18 _NORTC_MDAY	182
9.1.2.19 _NORTC_MON	182
9.1.2.20 _NORTC_YEAR	182
9.1.2.21 _STR_VOLUME_ID	182
9.1.2.22 _STRF_ENCODE	182
9.1.2.23 _SYNC_t	183
9.1.2.24 _USE_CHMOD	183
9.1.2.25 _USE_EXPAND	183
9.1.2.26 _USE_FASTSEEK	183
9.1.2.27 _USE_FIND	183
9.1.2.28 _USE_FORWARD	183
9.1.2.29 _USE_LABEL	183
9.1.2.30 _USE_LFN	183
9.1.2.31 _USE_MKFS	183
9.1.2.32 _USE_STRFUNC	183
9.1.2.33 _USE_TRIM	184
9.1.2.34 _VOLUME_STRS	184
9.1.2.35 _VOLUMES	184
9.1.2.36 ff_free	184
9.1.2.37 ff_malloc	184
9.2 src/hid_gatin.h File Reference	184

9.3 src/hid_wavplayer.h File Reference	184
9.3.1 Macro Definition Documentation	185
9.3.1.1 DSY_WAVPLAYER_H	185
9.3.1.2 WAV_FILENAME_MAX	185
9.4 src/usbd_cdc_if.h File Reference	185
9.4.1 Detailed Description	185
9.5 src/usbd_conf.h File Reference	186
9.5.1 Detailed Description	186
9.6 src/usbd_desc.h File Reference	187
9.6.1 Detailed Description	187
Index	189

Chapter 1

libdaisy

1.1 [↔](https://github.com/electro-smith/DaisyWiki/wiki) Documentation available on our wiki!

Multi-layer hardware abstraction library for Daisy Product family

On STM32H7 MCUs

Lower-levels use STM32 HAL (local copy w/ modifications in Drivers/)

Prefixes and their meanings:

- sys - System level configuration (clocks, dma, etc.)
- per - Peripheral level, internal to MCU (i2c, spi, etc.)
- dev - External device support (external flash chips, DACs, codecs, etc.)
- hid - User level interface elements (encoders, switches, audio, etc.)
- util - library level elements used within the library (not included via [daisy.h](#))
- daisy - core API files (specific boards, platforms have extended user APIs that configure libdaisy more below).

Also included is a core/ folder containing:

- a generic Makefile that can be included in a project Makefile to simplify getting started
- a linker script for defining the sections of memory used by the firmware
- core files for starting the hardware (system_stm32h7xx.c, startup_stm32h750xx.s, etc.)

1.2 Using libdaisy

Due to the amount of hardware configuration and flexibility of the daisy platform, (in the present, and the future), a user can use libdaisy to define their own custom hardware, or include one of our supported board files to jumpstart the creativity, and hack on an existing piece of hardware.

If you are getting started, and have one of the Daisy Family Products, you can skip ahead to that section below.

1.2.1 daisy.h

The base-level include file. This is all you need to include to create your own custom hardware that uses libdaisy. [daisy_seed.h](#) is an example of a board level file that utilizes libdaisy to define some hardware, and provide flexible access.

1.2.2 daisy_seed.h

The SOM-level include file. This can be used with any boards that use the Daisy Seed hardware. Additional configuration files, with more specific hardware access are provided below for our supported hardware platforms.

1.2.3 daisy_platform.h

Several other pairs of files exist in the repo for each of the supported hardware platforms that work with Daisy Seed. These are:

- daisy_field
- daisy_patch
- daisy_petal
- daisy_pod

With these files a number of additional initialization, and configuration is done by the library.

This allows a user to jump right into their new product with a simple api to do things without having a full understanding of what's going on under the hood.

With this flexible approach to the hardware configuration, we hope to promote a lot of fantastic hardware along with code to go with it.

Chapter 2

Module Index

2.1 Modules

Here is a list of all modules:

LIBDAISY	11
HUMAN_INTERFACE	12
AUDIO	13
CONTROLS	16
FEEDBACK	17
EXTERNAL	18
PERIPHERAL	19
SERIAL	20
ANALOG_DIGITAL_CONVERSION	29
OTHER	31
SYSTEM	36
DEVICE	38
SHIFTREGISTER	39
FLASH	41
CODEC	61
LED	63
SDRAM	65
BOARDS	67
UTILITY	71
Externals	101
STM32_USB_OTG_DEVICE_LIBRARY	102
USBD_CDC_IF	80
USBD_CDC_IF_Exported_Defines	81
USBD_CDC_IF_Exported_Types	82
USBD_CDC_IF_Exported_Macros	83
USBD_CDC_IF_Exported_Variables	84
USBD_CDC_IF_Exported_FunctionsPrototype	85
USBD_DESC	94
USBD_DESC_Exported_Constants	95
USBD_DESC_Exported_Defines	96
USBD_DESC_Exported_TypesDefinitions	97
USBD_DESC_Exported_Macros	98
USBD_DESC_Exported_Variables	99
USBD_DESC_Exported_FunctionsPrototype	100
USBD_OTG_DRIVER	103
USBD_CONF	86
USBD_CONF_Exported_Variables	87
USBD_CONF_Exported_Defines	88
USBD_CONF_Exported_Macros	90

USBD_CONF_Exported_Types	92
USBD_CONF_Exported_FunctionsPrototype	93

Chapter 3

Namespace Index

3.1 Namespace List

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

daisy	Hardware defines and helpers for daisy field platform	105
-----------------------	---	---------------------

Chapter 4

Class Index

4.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

daisy::AdcChannelConfig	107
daisy::AdcHandle	109
daisy::AnalogControl	
Hardware Interface for control inputs	
Primarily designed for ADC input controls such as potentiometers, and control voltage.	
112	
codec_frame_t	113
color	114
daisy::Color	115
daisy::ControlChangeEvent	116
daisy::daisy_field	117
daisy::DaisyPatch	
Class that handles initializing all of the hardware specific to the Daisy Patch Board.	
Helper funtions are also in place to provide easy access to built-in controls and peripherals	118
daisy::DaisyPetal	
Helpers and hardware definitions for daisy petal	122
daisy::DaisyPod	
Class that handles initializing all of the hardware specific to the Daisy Patch Board.	
Helper funtions are also in place to provide easy access to built-in controls and peripherals	128
daisy::DaisySeed	
This is the higher-level interface for the Daisy board.	
All basic peripheral configuration/initialization is setup here	132
dsy_audio_handle	134
dsy_dac_handle	135
dsy_gpio	136
dsy_gpio_pin	136
dsy_i2c_handle	137
dsy_qspi_handle	137
dsy_sai_handle	138
DSY_SD_CardInfoTypeDef	139
dsy_sdram_handle	140
dsy_sr_4021_handle	141
daisy::Encoder	
Generic Class for handling Quadrature Encoders	
Inspired/influenced by Mutable Instruments (pichenettes) Encoder classes	142
FontDef	143
daisy::GateIn	
Generic Class for handling gate inputs through GPIO	144

daisy::Led	
LED Class providing simple Software PWM ability, etc	
Eventually this will work with hardware PWM, and external LED Driver devices as well	145
daisy::MidiEvent	147
daisy::MidiHandler	
Simple MIDI Handler	
Parses bytes from an input into valid MidiEvents.	
The MidiEvents fill a FIFO queue that the user can pop messages from	148
daisy::NoteOnEvent	150
daisy::OledDisplay	151
daisy::Parameter	153
daisy::RgbLed	155
daisy::RingBuffer< T, size >	156
daisy::RingBuffer< T, 0 >	159
daisy::SdmmcHandler	161
daisy::SdmmcHandlerInit	162
ShiftRegister595	
Device Driver for 8-bit shift register.	
CD74HC595 - 8-bit serial to parallel output shift	162
daisy::SpiHandle	164
daisy::Switch	164
daisy::UartHandler	167
UsbHandle	
Interface for initializing and using the USB Peripherals on the daisy	170
WAV_FormatTypeDef	173
daisy::WavFileInfo	175
daisy::WavPlayer	175

Chapter 5

File Index

5.1 File List

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

src/ daisy.h	??
src/ daisy_core.h	??
src/ daisy_field.h	??
src/ daisy_patch.h	??
src/ daisy_petal.h	??
src/ daisy_pod.h	??
src/ daisy_seed.h	??
src/ dev_codec_ak4556.h	??
src/ dev_codec_pcm3060.h	??
src/ dev_codec_wm8731.h	??
src/ dev_codec_wm8731_frame.h	??
src/ dev_flash_IS25LP064A.h	??
src/ dev_flash_IS25LP080D.h	??
src/ dev_leddriver.h	??
src/ dev_sdram.h	??
src/ dev_sr_4021.h	??
src/ dev_sr_595.h	??
src/ fatfs.h	??
src/ ffconf.h	179
src/ hid_audio.h	??
src/ hid_ctrl.h	??
src/ hid_encoder.h	??
src/ hid_gatein.h	184
src/ hid_led.h	??
src/ hid_midi.h	??
src/ hid_oled_display.h	??
src/ hid_parameter.h	??
src/ hid_rgb_led.h	??
src/ hid_switch.h	??
src/ hid_usb.h	??
src/ hid_wavplayer.h	184
src/ per_adc.h	??
src/ per_dac.h	??
src/ per_gpio.h	??
src/ per_i2c.h	??
src/ per_qspi.h	??
src/ per_sai.h	??
src/ per_sdmmc.h	??
src/ per_spi.h	??
src/ per_tim.h	??

src/ per_uart.h	??
src/ stm32h7xx_hal_conf.h	??
src/ sys_dma.h	??
src/ sys_system.h	??
src/ usbd_cdc_if.h	
: Header for usbd_cdc_if.c file	185
src/ usbd_conf.h	
: Header for usbd_conf.c file	186
src/ usbd_desc.h	
: Header for usbd_conf.c file	187
src/ util_bsp_sd_diskio.h	??
src/ util_color.h	??
src/ util_hal_map.h	??
src/ util_oled_fonts.h	??
src/ util_ringbuffer.h	??
src/ util_sd_diskio.h	??
src/ util_unique_id.h	??
src/ util_wav_format.h	??

Chapter 6

Module Documentation

6.1 LIBDAISY

The daisy library.

Modules

- [HUMAN_INTERFACE](#)
Interface with the world.
- [PERIPHERAL](#)
Peripheral devices, not meant for human interaction.
- [SYSTEM](#)
Deals with system. DMA, clocks, etc.
- [DEVICE](#)
Low level devices. Led drivers, codecs, etc.
- [BOARDS](#)
Daisy devices. Pod, seed, etc.
- [UTILITY](#)
General utilities. Ringbuffers, LED colors, OLED stuff, etc.

6.1.1 Detailed Description

The daisy library.

6.2 HUMAN_INTERFACE

Interface with the world.

Modules

- [AUDIO](#)
Embedded Audio Engine.
- [CONTROLS](#)
Hardware Controls.
- [FEEDBACK](#)
Screens, leds, etc.
- [EXTERNAL](#)
External interface devices.

6.2.1 Detailed Description

Interface with the world.

6.3 AUDIO

Embedded Audio Engine.

- enum { [DSY_AUDIO_INTERNAL](#), [DSY_AUDIO_EXTERNAL](#), [DSY_AUDIO_LAST](#) }
- typedef void(* [dsy_audio_callback](#)) (float *, float *, size_t)
- typedef void(* [dsy_audio_mc_callback](#)) (float **, float **, size_t)
- void [dsy_audio_init](#) ([dsy_audio_handle](#) *handle)
- void [dsy_audio_set_callback](#) (uint8_t intext, [dsy_audio_callback](#) cb)
- void [dsy_audio_set_mc_callback](#) ([dsy_audio_mc_callback](#) cb)
- void [dsy_audio_set_blocksize](#) (uint8_t intext, size_t blocksize)
- void [dsy_audio_start](#) (uint8_t intext)
- void [dsy_audio_stop](#) (uint8_t intext)
- void [dsy_audio_enter_bypass](#) (uint8_t intext)
- void [dsy_audio_exit_bypass](#) (uint8_t intext)
- void [dsy_audio_passthru](#) (float *in, float *out, size_t size)
- void [dsy_audio_silence](#) (float *in, float *out, size_t size)

6.3.1 Detailed Description

Embedded Audio Engine.

6.3.2 Typedef Documentation

6.3.2.1 [dsy_audio_callback](#)

```
typedef void(* dsy_audio_callback) (float *, float *, size_t)
```

These are user-defineable callbacks that are called when audio data is ready to be received/transmitted. Function to define for using a single Stereo device for I/O audio is packed as: { LEFT | RIGHT | LEFT | RIGHT }

6.3.2.2 [dsy_audio_mc_callback](#)

```
typedef void(* dsy_audio_mc_callback) (float **, float **, size_t)
```

Defaults to 4 channels, and is fixed for now.

(still works for stereo, but will still fill buffers)

audio is packed as:

```
{ LEFT | LEFT + 1 | . . . | LEFT + SIZE | RIGHT | RIGHT + 1 | . . . | RIGHT + SIZE }
```

6.3.3 Enumeration Type Documentation

6.3.3.1 anonymous enum

anonymous enum

Internally, there are two separate 'audio blocks' that can be configured together or separately

Enumerator

DSY_AUDIO_INTERNAL	&
DSY_AUDIO_EXTERNAL	&
DSY_AUDIO_LAST	&

6.3.4 Function Documentation

6.3.4.1 dsy_audio_enter_bypass()

```
void dsy_audio_enter_bypass (
    uint8_t intext )
```

If the device supports hardware bypass, enter that mode.

6.3.4.2 dsy_audio_exit_bypass()

```
void dsy_audio_exit_bypass (
    uint8_t intext )
```

If the device supports hardware bypass, exit that mode.

6.3.4.3 dsy_audio_init()

```
void dsy_audio_init (
    dsy_audio_handle * handle )
```

Initializes the Audio Engine using configurations set to the sai_handle
i2c_handles can be set to NULL if not needed.

6.3.4.4 dsy_audio_passthru()

```
void dsy_audio_passthru (
    float * in,
    float * out,
    size_t size )
```

A few useful stereo-interleaved callbacks
Passes the input to the output

6.3.4.5 dsy_audio_set_blocksize()

```
void dsy_audio_set_blocksize (
    uint8_t intext,
    size_t blocksize )
```

Sets the number of samples (per-channel) to be handled in a single audio frame.

6.3.4.6 dsy_audio_set_callback()

```
void dsy_audio_set_callback (
    uint8_t intext,
    dsy_audio_callback cb )
```

Sets the user defined, interleaving callback to be called when audio data is ready.
intext is a specifier for DSY_AUDIO_INT/EXT (which audio peripheral to use).
When using this, each 'audio block' can have completely independent callbacks.

6.3.4.7 dsy_audio_set_mc_callback()

```
void dsy_audio_set_mc_callback (
    dsy_audio_mc_callback cb )
```

Sets the user defined, non-interleaving callback to be called when audio data is ready.
This will always use both DSY_AUDIO_INT and DSY_AUDIO_EXT blocks together.
To ensure clean audio you'll want to make sure the two SAls are set to the same samplerate

6.3.4.8 dsy_audio_silence()

```
void dsy_audio_silence (
    float * in,
    float * out,
    size_t size )
```

sets outputs to 0 without stopping the Audio Engine

6.3.4.9 dsy_audio_start()

```
void dsy_audio_start (
    uint8_t intext )
```

Starts Audio Engine, callbacks will begin getting called.

When using with dsy_audio_mc_callback (for 4 channels), this function should be called for both audio blocks

6.3.4.10 dsy_audio_stop()

```
void dsy_audio_stop (
    uint8_t intext )
```

Stops transmitting/receiving audio on the specified audio block.

6.4 CONTROLS

Hardware Controls.

Classes

- class [daisy::AnalogControl](#)
Hardware Interface for control inputs
Primarily designed for ADC input controls such as potentiometers, and control voltage.
.
- class [daisy::Encoder](#)
Generic Class for handling Quadrature Encoders
Inspired/influenced by Mutable Instruments (pichenettes) [Encoder](#) classes.
- class [daisy::GateIn](#)
Generic Class for handling gate inputs through GPIO.
- class [daisy::Parameter](#)
- class [daisy::Switch](#)

6.4.1 Detailed Description

Hardware Controls.

6.5 FEEDBACK

Screens, leds, etc.

Classes

- class `daisy::Led`
LED Class providing simple Software PWM ability, etc
Eventually this will work with hardware PWM, and external LED Driver devices as well.
- class `daisy::OledDisplay`
- class `daisy::RgbLed`

6.5.1 Detailed Description

Screens, leds, etc.

6.6 EXTERNAL

External interface devices.

Classes

- struct [daisy::NoteOnEvent](#)
- struct [daisy::ControlChangeEvent](#)
- struct [daisy::MidiEvent](#)
- class [daisy::MidiHandler](#)

Simple MIDI Handler

Parses bytes from an input into valid MidiEvents.

The MidiEvents fill a FIFO queue that the user can pop messages from.

Enumerations

- enum [daisy::MidiMessageType](#) {
[daisy::NoteOff](#), [daisy::NoteOn](#), [daisy::PolyphonicKeyPressure](#), [daisy::ControlChange](#),
[daisy::ProgramChange](#), [daisy::ChannelPressure](#), [daisy::PitchBend](#), [daisy::MessageLast](#) }

6.6.1 Detailed Description

External interface devices.

6.6.2 Enumeration Type Documentation

6.6.2.1 MidiMessageType

enum [daisy::MidiMessageType](#)

Parsed from the Status Byte, these are the common Midi Messages that can be handled.

At this time only 3-byte messages are correctly parsed into MidiEvents.

Enumerator

NoteOff	&
NoteOn	&
PolyphonicKeyPressure	&
ControlChange	&
ProgramChange	&
ChannelPressure	&
PitchBend	&
MessageLast	&

6.7 PERIPHERAL

Peripheral devices, not meant for human interaction.

Modules

- [SERIAL](#)
Serial Communications.
- [ANALOG_DIGITAL_CONVERSION](#)
Convert from digital to analog, or vice-versa.
- [OTHER](#)
GPIO, timers, and SDMMC.

6.7.1 Detailed Description

Peripheral devices, not meant for human interaction.

6.8 SERIAL

Serial Communications.

Classes

- struct [dsy_i2c_handle](#)
- struct [dsy_qspi_handle](#)
- struct [dsy_sai_handle](#)
- class [daisy::SpiHandle](#)
- class [daisy::UartHandler](#)

Enumerations

- enum [dsy_i2c_periph](#) { [DSY_I2C_PERIPH_1](#), [DSY_I2C_PERIPH_2](#), [DSY_I2C_PERIPH_3](#), [DSY_I2C_PERIPH_4](#) }
- enum [dsy_i2c_pin](#) { [DSY_I2C_PIN_SCL](#), [DSY_I2C_PIN_SDA](#), [DSY_I2C_PIN_LAST](#) }
- enum [dsy_i2c_speed](#) { [DSY_I2C_SPEED_100KHZ](#), [DSY_I2C_SPEED_400KHZ](#), [DSY_I2C_SPEED_1MHZ](#), [DSY_I2C_SPEED_LAST](#) }
- enum [dsy_qspi_pin](#) { [DSY_QSPI_PIN_IO0](#), [DSY_QSPI_PIN_IO1](#), [DSY_QSPI_PIN_IO2](#), [DSY_QSPI_PIN_IO3](#), [DSY_QSPI_PIN_CLK](#), [DSY_QSPI_PIN_NCS](#), [DSY_QSPI_PIN_LAST](#) }
- enum [dsy_qspi_mode](#) { [DSY_QSPI_MODE_DSY_MEMORY_MAPPED](#), [DSY_QSPI_MODE_INDIRECT_POLLING](#), [DSY_QSPI_MODE_LAST](#) }
- enum [dsy_qspi_device](#) { [DSY_QSPI_DEVICE_IS25LP080D](#), [DSY_QSPI_DEVICE_IS25LP064A](#), [DSY_QSPI_DEVICE_LAST](#) }
- enum [dsy_audio_sai](#) { [DSY_AUDIO_INIT_SAI1](#), [DSY_AUDIO_INIT_SAI2](#), [DSY_AUDIO_INIT_BOTH](#), [DSY_AUDIO_INIT_NONE](#), [DSY_AUDIO_INIT_LAST](#) }
- enum [dsy_audio_samplerate](#) { [DSY_AUDIO_SAMPLERATE_32K](#), [DSY_AUDIO_SAMPLERATE_48K](#), [DSY_AUDIO_SAMPLERATE_96K](#), [DSY_AUDIO_SAMPLERATE_LAST](#) }
- enum [dsy_audio_bitdepth](#) { [DSY_AUDIO_BITDEPTH_16](#), [DSY_AUDIO_BITDEPTH_24](#), [DSY_AUDIO_BITDEPTH_LAST](#) }
- enum [dsy_audio_sync](#) { [DSY_AUDIO_SYNC_MASTER](#), [DSY_AUDIO_SYNC_SLAVE](#), [DSY_AUDIO_SYNC_LAST](#) }
- enum [dsy_audio_dir](#) { [DSY_AUDIO_RX](#), [DSY_AUDIO_TX](#) }
- enum [dsy_sai_pin](#) { [DSY_SAI_PIN_MCLK](#), [DSY_SAI_PIN_FS](#), [DSY_SAI_PIN_SCK](#), [DSY_SAI_PIN_SIN](#), [DSY_SAI_PIN_SOUT](#), [DSY_SAI_PIN_LAST](#) }
- enum [dsy_audio_device](#) { [DSY_AUDIO_NONE](#), [DSY_AUDIO_DEVICE_PCM3060](#), [DSY_AUDIO_DEVICE_WM8731](#), [DSY_AUDIO_DEVICE_AK4556](#), [DSY_AUDIO_DEVICE_LAST](#) }
- enum { [DSY_SAI_1](#), [DSY_SAI_2](#), [DSY_SAI_LAST](#) }
- enum [daisy::SpiPeriph](#) { [daisy::SPI_PERIPH_1](#), [daisy::SPI_PERIPH_3](#), [daisy::SPI_PERIPH_6](#) }
- enum [daisy::SpiPin](#) { [daisy::SPI_PIN_CS](#), [daisy::SPI_PIN_SCK](#), [daisy::SPI_PIN_MOSI](#), [daisy::SPI_PIN_MISO](#) }

Functions

- void [dsy_i2c_init](#) ([dsy_i2c_handle](#) *dsy_hi2c)
- int [dsy_qspi_init](#) ([dsy_qspi_handle](#) *hqspi)
- int [dsy_qspi_deinit](#) ()
- int [dsy_qspi_writepage](#) (uint32_t adr, uint32_t sz, uint8_t *buf)
- int [dsy_qspi_write](#) (uint32_t address, uint32_t size, uint8_t *buffer)
- int [dsy_qspi_erase](#) (uint32_t start_adr, uint32_t end_adr)
- int [dsy_qspi_erasesector](#) (uint32_t addr)

- void `dsy_sai_init` (`dsy_audio_sai` init, `dsy_audio_samplerate` sr[2], `dsy_audio_bitdepth` bitdepth[2], `dsy_audio_sync` sync_config[2], `dsy_gpio_pin` *sai1_pin_list, `dsy_gpio_pin` *sai2_pin_list)
- void `dsy_sai_init_from_handle` (`dsy_sai_handle` *hsai)

Variables

- const size_t `daisy::kUartMaxBufferSize` = 32

6.8.1 Detailed Description

Serial Communications.

6.8.2 Enumeration Type Documentation

6.8.2.1 anonymous enum

anonymous enum

Index for the several arrays in the sai_handle struct below.

Enumerator

DSY_SAI_1	&
DSY_SAI_2	&
DSY_SAI_LAST	&

6.8.2.2 dsy_audio_bitdepth

enum `dsy_audio_bitdepth`

Specifies the bitdepth of the hardware connected to the SAI peripheral

Enumerator

DSY_AUDIO_BITDEPTH_16	&
DSY_AUDIO_BITDEPTH_24	&
DSY_AUDIO_BITDEPTH_LAST	&

6.8.2.3 dsy_audio_device

enum `dsy_audio_device`

List of devices with built in support. Devices not listed here, will need to have initialization done externally.

Enumerator

DSY_AUDIO_NONE	For unsupported, or custom devices.
DSY_AUDIO_DEVICE_PCM3060	&
DSY_AUDIO_DEVICE_WM8731	&
DSY_AUDIO_DEVICE_AK4556	&
DSY_AUDIO_DEVICE_LAST	&

6.8.2.4 dsy_audio_dir

enum `dsy_audio_dir`

Each SAI has two datalines, they can independently be configured as inputs or outputs.

Enumerator

DSY_AUDIO_RX	&
DSY_AUDIO_TX	&

6.8.2.5 dsy_audio_sai

enum `dsy_audio_sai`

Driver for the SAI peripheral Supports SAI1 and SAI2 with several configuration options selects which SAI (or both/none) to initialize

Enumerator

DSY_AUDIO_INIT_SAI1	&
DSY_AUDIO_INIT_SAI2	&
DSY_AUDIO_INIT_BOTH	&
DSY_AUDIO_INIT_NONE	&
DSY_AUDIO_INIT_LAST	&

6.8.2.6 dsy_audio_samplerate

enum `dsy_audio_samplerate`

Currently Sample Rates are not correctly supported. All audio is currently run at 48kHz

Enumerator

DSY_AUDIO_SAMPLERATE_32K	&
DSY_AUDIO_SAMPLERATE_48K	&
DSY_AUDIO_SAMPLERATE_96K	&
DSY_AUDIO_SAMPLERATE_LAST	&

6.8.2.7 dsy_audio_sync

enum `dsy_audio_sync`

Setting for each SAI that sets whether the processor is generating the MCLK signal or not.

Enumerator

DSY_AUDIO_SYNC_MASTER	No Crystal
DSY_AUDIO_SYNC_SLAVE	Crystal
DSY_AUDIO_SYNC_LAST	&

6.8.2.8 dsy_i2c_periph

enum [dsy_i2c_periph](#)

Driver for controlling I2C devices Specifies the internal peripheral to use (these are mapped to different pins on the hardware).

Enumerator

DSY_I2C_PERIPH↔ _1	&
DSY_I2C_PERIPH↔ _2	&
DSY_I2C_PERIPH↔ _3	&
DSY_I2C_PERIPH↔ _4	&

6.8.2.9 dsy_i2c_pin

enum [dsy_i2c_pin](#)

List of pins associated with the peripheral. These must be set in the handle's pin_config.

Enumerator

DSY_I2C_PIN_SCL	&
DSY_I2C_PIN_SDA	&
DSY_I2C_PIN_LAST	&

6.8.2.10 dsy_i2c_speed

enum [dsy_i2c_speed](#)

Rate at which the clock/data will be sent/received. The device being used will have maximum speeds. 1MHZ Mode is currently 886kHz**

Enumerator

DSY_I2C_SPEED_100KHZ	&
DSY_I2C_SPEED_400KHZ	&
DSY_I2C_SPEED_1MHZ	&
DSY_I2C_SPEED_LAST	&

6.8.2.11 dsy_qspi_device

enum [dsy_qspi_device](#)

Flash Devices supported. (Both of these are more-or-less the same, just different sizes).

Enumerator

DSY_QSPI_DEVICE_IS25LP080D	&
DSY_QSPI_DEVICE_IS25LP064A	&
DSY_QSPI_DEVICE_LAST	&

6.8.2.12 dsy_qspi_mode

enum [dsy_qspi_mode](#)

Modes of operation. Memory Mapped mode: QSPI configured so that the QSPI can be read from starting address 0x90000000. Writing is not possible in this mode.

Indirect Polling mode: Device driver enabled.

Read/Write possible via dsy_qspi_* functions

Enumerator

DSY_QSPI_MODE_DSY_MEMORY_MAPPED	&
DSY_QSPI_MODE_INDIRECT_POLLING	&
DSY_QSPI_MODE_LAST	&

6.8.2.13 dsy_qspi_pin

enum [dsy_qspi_pin](#)

Driver for QSPI peripheral to interface with external flash memory.

Currently supported QSPI Devices:

IS25LP080D List of Pins used in QSPI (passed in during Init)

Enumerator

DSY_QSPI_PIN_IO0	&
DSY_QSPI_PIN_IO1	&
DSY_QSPI_PIN_IO2	&
DSY_QSPI_PIN_IO3	&
DSY_QSPI_PIN_CLK	&
DSY_QSPI_PIN_NCS	&
DSY_QSPI_PIN_LAST	&

6.8.2.14 dsy_sai_pin

enum [dsy_sai_pin](#)

List of the pins that need to be initialized SIN/SOUT is a bit misleading, and should be turned into A/B since it is possible to configure two inputs or two outputs on a single SAI.

Enumerator

DSY_SAI_PIN_MCLK	&
DSY_SAI_PIN_FS	&
DSY_SAI_PIN_SCK	&
DSY_SAI_PIN_SIN	&
DSY_SAI_PIN_SOUT	&
DSY_SAI_PIN_LAST	&

6.8.2.15 SpiPeriph

enum `daisy::SpiPeriph`
 SPI peripheral enum

Enumerator

<code>SPI_PERIPH_1</code>	SPI peripheral 1
<code>SPI_PERIPH_3</code>	SPI peripheral 3
<code>SPI_PERIPH_6</code>	SPI peripheral 3

6.8.2.16 SpiPin

enum `daisy::SpiPin`
 SPI pins

Enumerator

<code>SPI_PIN_CS</code>	CS pin
<code>SPI_PIN_SCK</code>	SCK pin
<code>SPI_PIN_MOSI</code>	MOSI pin
<code>SPI_PIN_MISO</code>	MISO pin

6.8.3 Function Documentation

6.8.3.1 `dsy_i2c_init()`

```
void dsy_i2c_init (
    dsy_i2c_handle * dsy_hi2c )
```

Initializes an I2C peripheral with the data given from the handle.

Parameters

<code>*dsy_hi2c</code>	Required to initialize.
------------------------	-------------------------

6.8.3.2 `dsy_qspi_deinit()`

```
int dsy_qspi_deinit ( )
```

Deinitializes the peripheral This should be called before reinitializing QSPI in a different mode.

Returns

DSY_MEMORY_OK or DSY_MEMORY_ERROR

6.8.3.3 `dsy_qspi_erase()`

```
int dsy_qspi_erase (
```

```
uint32_t start_adr,
uint32_t end_adr )
```

Erases the area specified on the chip. Erasures will happen by 4K, 32K or 64K increments. Smallest erase possible is 4kB at a time. (on IS25LP*)

Parameters

<i>start_adr</i>	Address to begin erasing from
<i>end_adr</i>	Address to stop erasing at

Returns

DSY_MEMORY_OK or DSY_MEMORY_ERROR

6.8.3.4 dsy_qspi_erasesector()

```
int dsy_qspi_erasesector (
    uint32_t addr )
```

Erases a single sector of the chip.

TODO: Document the size of this function.

Parameters

<i>addr</i>	Address of sector to erase
-------------	----------------------------

Returns

DSY_MEMORY_OK or DSY_MEMORY_ERROR

6.8.3.5 dsy_qspi_init()

```
int dsy_qspi_init (
    dsy_qspi_handle * hqspi )
```

Initializes QSPI peripheral, and Resets, and prepares memory for access.

Parameters

<i>hqspi</i>	should be populated with the mode, device and pin_config before calling this function.
--------------	--

Returns

DSY_MEMORY_OK or DSY_MEMORY_ERROR

6.8.3.6 dsy_qspi_write()

```
int dsy_qspi_write (
    uint32_t address,
    uint32_t size,
    uint8_t * buffer )
```

Writes data in buffer to to the QSPI. Starting at address to address+size

Parameters

<i>address</i>	Address to write to
<i>size</i>	Buffer size
<i>buffer</i>	Buffer to write

Returns

DSY_MEMORY_OK or DSY_MEMORY_ERROR

6.8.3.7 dsy_qspi_writepage()

```
int dsy_qspi_writepage (
    uint32_t adr,
    uint32_t sz,
    uint8_t * buf )
```

Writes a single page to the specified address on the QSPI chip. For IS25LP* page size is 256 bytes.

Parameters

<i>adr</i>	Address to write to
<i>sz</i>	Buff size
<i>buf</i>	Buffer to write

Returns

DSY_MEMORY_OK or DSY_MEMORY_ERROR

6.8.3.8 dsy_sai_init()

```
void dsy_sai_init (
    dsy_audio_sai init,
    dsy_audio_samplerate sr[2],
    dsy_audio_bitdepth bitdepth[2],
    dsy_audio_sync sync_config[2],
    dsy_gpio_pin * sai1_pin_list,
    dsy_gpio_pin * sai2_pin_list )
```

Initializes the SAI peripheral(s) with the specified settings. Pinlists should be arrays of DSY_SAI_PIN_LAST elements

Parameters

<i>init</i>	&
<i>sr[]</i>	Sample rate per chan: 0, 1
<i>bitdepth[]</i>	Bitdepth per chan: 0, 1
<i>sync_config[]</i>	& sync config per chan: 0, 1
<i>*sai1_pin_list</i>	&
<i>*sai2_pin_list</i>	&

6.8.3.9 dsy_sai_init_from_handle()

```
void dsy_sai_init_from_handle (
    dsy_sai_handle * hsai )
```

Uses the data within *hsai to initialize the peripheral(s)

Parameters

<i>hsai</i>	&
-------------	---

6.8.4 Variable Documentation

6.8.4.1 kUartMaxBufferSize

```
const size_t daisy::kUartMaxBufferSize = 32
```

Maximum Queue buffer size

6.9 ANALOG_DIGITAL_CONVERSION

Convert from digital to analog, or vice-versa.

Classes

- struct [daisy::AdcChannelConfig](#)
- class [daisy::AdcHandle](#)
- struct [dsy_dac_handle](#)

Enumerations

- enum [dsy_dac_mode](#) { [DSY_DAC_MODE_POLLING](#), [DSY_DAC_MODE_LAST](#) }
- enum [dsy_dac_bitdepth](#) { [DSY_DAC_BITS_8](#), [DSY_DAC_BITS_12](#), [DSY_DAC_BITS_LAST](#) }
- enum [dsy_dac_channel](#) { [DSY_DAC_CHN1](#), [DSY_DAC_CHN2](#), [DSY_DAC_CHN_LAST](#), [DSY_DAC_CHN_BOTH](#) }

Functions

- void [dsy_dac_init](#) ([dsy_dac_handle](#) *dsy_hdac, [dsy_dac_channel](#) channel)
- void [dsy_dac_start](#) ([dsy_dac_channel](#) channel)
- void [dsy_dac_write](#) ([dsy_dac_channel](#) channel, uint16_t val)

6.9.1 Detailed Description

Convert from digital to analog, or vice-versa.

6.9.2 Enumeration Type Documentation

6.9.2.1 dsy_dac_bitdepth

enum [dsy_dac_bitdepth](#)

Sets the bit depth of the DAC output This can be set independently for each channel.

Enumerator

DSY_DAC_BITS_8	&
DSY_DAC_BITS_12	&
DSY_DAC_BITS_LAST	&

6.9.2.2 dsy_dac_channel

enum [dsy_dac_channel](#)

Sets which channel(s) are initialized with the settings chosen.

Enumerator

DSY_DAC_CHN1	&
DSY_DAC_CHN2	&
DSY_DAC_CHN_LAST	&
DSY_DAC_CHN_BOTH	&

6.9.2.3 dsy_dac_mode

enum `dsy_dac_mode`

Driver for the built in DAC on the STM32 The STM32 has 2 Channels of independently configurable DACs, with up to 12-bit resolution. Currently only Polling is supported.

Enumerator

<code>DSY_DAC_MODE_POLLING</code>	Polling mode
<code>DSY_DAC_MODE_LAST</code>	3

6.9.3 Function Documentation

6.9.3.1 dsy_dac_init()

```
void dsy_dac_init (
    dsy_dac_handle * dsy_hdac,
    dsy_dac_channel channel )
```

Initializes the specified channel(s) of the DAC

Parameters

<i>*dsy_hdac</i>	Dac to initialize
<i>channel</i>	Channels to init

6.9.3.2 dsy_dac_start()

```
void dsy_dac_start (
    dsy_dac_channel channel )
```

Turns on the DAC and turns on any internal timer if necessary.

Parameters

<i>channel</i>	Channel to start
----------------	------------------

6.9.3.3 dsy_dac_write()

```
void dsy_dac_write (
    dsy_dac_channel channel,
    uint16_t val )
```

Sets the specified channel of the dac to the value (within bitdepth) resolution. When set to 8-bit, val should be 0-255
When set to 12-bit, val should be 0-4095

Parameters

<i>channel</i>	Channel to write to
<i>val</i>	Value to write

6.10 OTHER

GPIO, timers, and SDMMC.

Classes

- struct [dsy_gpio](#)
- struct [daisy::SdmmcHandlerInit](#)
- class [daisy::SdmmcHandler](#)

Enumerations

- enum [dsy_gpio_mode](#) {
[DSY_GPIO_MODE_INPUT](#), [DSY_GPIO_MODE_OUTPUT_PP](#), [DSY_GPIO_MODE_OUTPUT_OD](#),
[DSY_GPIO_MODE_ANALOG](#),
[DSY_GPIO_MODE_LAST](#) }
- enum [dsy_gpio_pull](#) { [DSY_GPIO_NOPULL](#), [DSY_GPIO_PULLUP](#), [DSY_GPIO_PULLDOWN](#) }
- enum [daisy::SdmmcMode](#) { [daisy::SDMMC_MODE_FATFS](#) }
- enum [daisy::SdmmcBitWidth](#) { [daisy::SDMMC_BITS_1](#), [daisy::SDMMC_BITS_4](#) }
- enum [daisy::SdmmcSpeed](#) { [daisy::SDMMC_SPEED_400KHZ](#), [daisy::SDMMC_SPEED_12MHZ](#) }

Functions

- void [dsy_gpio_init](#) ([dsy_gpio](#) *p)
- void [dsy_gpio_deinit](#) ([dsy_gpio](#) *p)
- uint8_t [dsy_gpio_read](#) ([dsy_gpio](#) *p)
- void [dsy_gpio_write](#) ([dsy_gpio](#) *p, uint8_t state)
- void [dsy_gpio_toggle](#) ([dsy_gpio](#) *p)
- void [dsy_tim_init](#) ()
- void [dsy_tim_start](#) ()
- uint32_t [dsy_tim_get_tick](#) ()
- void [dsy_tim_delay_tick](#) (uint32_t cnt)
- uint32_t [dsy_tim_get_ms](#) ()
- void [dsy_tim_delay_ms](#) (uint32_t cnt)
- uint32_t [dsy_tim_get_us](#) ()
- void [dsy_tim_delay_us](#) (uint32_t cnt)

6.10.1 Detailed Description

GPIO, timers, and SDMMC.

General Purpose IO driver

6.10.2 Enumeration Type Documentation

6.10.2.1 dsy_gpio_mode

enum [dsy_gpio_mode](#)

Sets the mode of the GPIO

Enumerator

DSY_GPIO_MODE_INPUT	&
DSY_GPIO_MODE_OUTPUT_PP	Push-Pull
DSY_GPIO_MODE_OUTPUT_OD	Open-Drain
DSY_GPIO_MODE_ANALOG	&
DSY_GPIO_MODE_LAST	&

6.10.2.2 dsy_gpio_pull

enum [dsy_gpio_pull](#)

Configures whether an internal Pull up or Pull down resistor is used

Enumerator

DSY_GPIO_NOPULL	&
DSY_GPIO_PULLUP	&
DSY_GPIO_PULLDOWN	&

6.10.2.3 SdmmcBitWidth

enum [daisy::SdmmcBitWidth](#)

Sets whether 4-bit mode or 1-bit mode is used for the SDMMC

Enumerator

SDMMC_BITS↔ _1	&
SDMMC_BITS↔ _4	&

6.10.2.4 SdmmcMode

enum [daisy::SdmmcMode](#)

Operating Mode. Currently only FatFS is supported.

Enumerator

SDMMC_MODE_FATFS	&
------------------	---

6.10.2.5 SdmmcSpeed

enum [daisy::SdmmcSpeed](#)

Sets the desired clock speed of the SD card bus.

Initialization is always done at or below 400kHz, and then the user speed is set.

Enumerator

SDMMC_SPEED_400KHZ	&
SDMMC_SPEED_12MHZ	&

6.10.3 Function Documentation

6.10.3.1 dsy_gpio_deinit()

```
void dsy_gpio_deinit (
    dsy_gpio * p )
```

Deinitializes the gpio pin

Parameters

<i>*p</i>	Pin pointer
-----------	-------------

6.10.3.2 dsy_gpio_init()

```
void dsy_gpio_init (
    dsy_gpio * p )
```

Initializes the gpio with the settings configured.

Parameters

<i>*p</i>	Pin pointer
-----------	-------------

6.10.3.3 dsy_gpio_read()

```
uint8_t dsy_gpio_read (
    dsy_gpio * p )
```

Reads the state of the gpio pin

Parameters

<i>*p</i>	Pin pointer
-----------	-------------

Returns

1 if the pin is HIGH, and 0 if the pin is LOW

6.10.3.4 dsy_gpio_toggle()

```
void dsy_gpio_toggle (
    dsy_gpio * p )
```

Toggles the state of the pin so that it is not at the same state as it was previously.

Parameters

<i>*p</i>	Pin pointer
-----------	-------------

6.10.3.5 dsy_gpio_write()

```
void dsy_gpio_write (
    dsy_gpio * p,
    uint8_t state )
```

Writes the state to the gpio pin Pin will be set to 3v3 when state is 1, and 0V when state is 0

Parameters

<i>*p</i>	Pin pointer
<i>state</i>	State to write

6.10.3.6 dsy_tim_delay_ms()

```
void dsy_tim_delay_ms (
    uint32_t cnt )
```

blocking delay of cnt milliseconds.

Parameters

<i>cnt</i>	Delay time in ms
------------	------------------

6.10.3.7 dsy_tim_delay_tick()

```
void dsy_tim_delay_tick (
    uint32_t cnt )
```

blocking delay of cnt timer ticks.

Parameters

<i>cnt</i>	Number of ticks
------------	-----------------

6.10.3.8 dsy_tim_delay_us()

```
void dsy_tim_delay_us (
    uint32_t cnt )
```

blocking delay of cnt microseconds.

Parameters

<i>cnt</i>	Delay time in us
------------	------------------

6.10.3.9 dsy_tim_get_ms()

```
uint32_t dsy_tim_get_ms ( )
```

These functions are converted to use milliseconds as their time base.

Returns

the number of milliseconds through the timer period.

6.10.3.10 dsy_tim_get_tick()

```
uint32_t dsy_tim_get_tick ( )
```

These functions are specific to the actual clock ticks at the timer frequency which is currently fixed at 200MHz

Returns

a number 0x00000000-0xffffffff of the current tick

6.10.3.11 dsy_tim_get_us()

```
uint32_t dsy_tim_get_us ( )
```

These functions are converted to use microseconds as their time base.

Returns

the number of microseconds through the timer period.

6.10.3.12 dsy_tim_init()

```
void dsy_tim_init ( )
```

General purpose timer for delays and general timing. initializes the TIM2 peripheral with maximum counter autoreload, and no prescalers.

6.10.3.13 dsy_tim_start()

```
void dsy_tim_start ( )
```

Starts the timer ticking.

6.11 SYSTEM

Deals with system. DMA, clocks, etc.

Functions

- void [dsy_dma_init](#) (void)
- void [dsy_system_init](#) ()
- void [dsy_system_jumpto](#) (uint32_t addr)
- void [dsy_system_jumptoqspi](#) ()
- uint32_t [dsy_system_getnow](#) ()
- void [dsy_system_delay](#) (uint32_t delay_ms)

6.11.1 Detailed Description

Deals with system. DMA, clocks, etc.

Low level System Configuration

6.11.2 Function Documentation

6.11.2.1 dsy_dma_init()

```
void dsy_dma_init (
    void )
```

Initializes the Direct Memory Access Peripheral used by many internal elements of libdaisy. Initializes the DMA (specifically for the modules used within the library)

6.11.2.2 dsy_system_delay()

```
void dsy_system_delay (
    uint32_t delay_ms )
```

Blocking Delay that uses the SysTick (1ms callback) to wait.

Parameters

<i>delay_ms</i>	Time to delay in ms
-----------------	---------------------

6.11.2.3 dsy_system_getnow()

```
uint32_t dsy_system_getnow ( )
```

Returns

a uint32_t value of milliseconds since the SysTick started

Note! This is a HAL_GetTick()

6.11.2.4 dsy_system_init()

```
void dsy_system_init ( )
```

Initializes Clock tree, MPU, and internal memories voltage regulators.

This function *must* be called at the beginning of any program using libdaisy Higher level daisy_ files call this through the DaisySeed object.

6.11.2.5 dsy_system_jumpto()

```
void dsy_system_jumpto (
    uint32_t addr )
```

Jump to an address within the internal memory

This may not work correctly, and may not be very useful with the single sector of memory on the stm32h750**

Parameters

<i>addr</i>	Address to jump to
-------------	--------------------

6.11.2.6 dsy_system_jumptoqspi()

```
void dsy_system_jumptoqspi ( )
```

Jumps to the first address of the external flash chip (0x90000000) If there is no code there, the chip will likely fall through to the while() loop TODO: Documentation/Loader for using external flash coming soon.

6.12 DEVICE

Low level devices. Led drivers, codecs, etc.

Modules

- [SHIFTREGISTER](#)
Digital shift registers.
- [FLASH](#)
Flash memory.
- [CODEC](#)
Audio codecs.
- [LED](#)
LED driver devices.
- [SDRAM](#)
SDRAM devices.

6.12.1 Detailed Description

Low level devices. Led drivers, codecs, etc.

6.13 SHIFTREGISTER

Digital shift registers.

Classes

- struct [dsy_sr_4021_handle](#)
- class [ShiftRegister595](#)
Device Driver for 8-bit shift register.
CD74HC595 - 8-bit serial to parallel output shift.

Enumerations

- enum {
[DSY_SR_4021_PIN_CS](#), [DSY_SR_4021_PIN_CLK](#), [DSY_SR_4021_PIN_DATA](#), [DSY_SR_4021_PIN_DATA2](#),
[DSY_SR_4021_PIN_LAST](#) }

Functions

- void [dsy_sr_4021_init](#) ([dsy_sr_4021_handle](#) *sr)
- void [dsy_sr_4021_update](#) ([dsy_sr_4021_handle](#) *sr)
- uint8_t [dsy_sr_4021_state](#) ([dsy_sr_4021_handle](#) *sr, uint8_t idx)

6.13.1 Detailed Description

Digital shift registers.

Device driver for the CD4021. Bit-banged serial shift input.

6.13.2 Enumeration Type Documentation

6.13.2.1 anonymous enum

anonymous enum

Pins that need to be configured to use. DATA2 only needs to be set if num_parallel is > 1

Enumerator

DSY_SR_4021_PIN_CS	CS Pin
DSY_SR_4021_PIN_CLK	CLK Pin
DSY_SR_4021_PIN_DATA	DATA pin
DSY_SR_4021_PIN_DATA2	DATA2 Pin, optional
DSY_SR_4021_PIN_LAST	Enum Last

6.13.3 Function Documentation

6.13.3.1 dsy_sr_4021_init()

```
void dsy_sr_4021_init (
    dsy\_sr\_4021\_handle * sr )
```

Initialize CD4021 with settings from sr_4021_handle

Parameters

<i>sr</i>	handle to initialize
-----------	----------------------

6.13.3.2 dsy_sr_4021_state()

```
uint8_t dsy_sr_4021_state (
    dsy_sr_4021_handle * sr,
    uint8_t idx )
```

Returns the state of a pin at a given index.

Parameters

<i>*sr</i>	Handle containing desired pin
<i>idx</i>	Pin index

6.13.3.3 dsy_sr_4021_update()

```
void dsy_sr_4021_update (
    dsy_sr_4021_handle * sr )
```

Fills internal states with CD4021 data states.

Parameters

<i>*sr</i>	Handle to update
------------	------------------

6.14 FLASH

Flash memory.

Macros

- #define [RESET_ENABLE_CMD](#) 0x66
- #define [RESET_MEMORY_CMD](#) 0x99
- #define [READ_ID_CMD](#) 0x9E
- #define [READ_ID_CMD2](#) 0x9F
- #define [MULTIPLE_IO_READ_ID_CMD](#) 0xAF
- #define [READ_SERIAL_FLASH_DISCO_PARAM_CMD](#) 0x5A
- #define [READ_CMD](#) 0x03
- #define [READ_4_BYTE_ADDR_CMD](#) 0x13
- #define [FAST_READ_CMD](#) 0x0B
- #define [FAST_READ_DTR_CMD](#) 0x0D
- #define [FAST_READ_4_BYTE_ADDR_CMD](#) 0x0C
- #define [DUAL_OUT_FAST_READ_CMD](#) 0x3B
- #define [DUAL_OUT_FAST_READ_DTR_CMD](#) 0x3D
- #define [DUAL_OUT_FAST_READ_4_BYTE_ADDR_CMD](#) 0x3C
- #define [DUAL_INOUT_FAST_READ_CMD](#) 0xBB
- #define [DUAL_INOUT_FAST_READ_DTR_CMD](#) 0xBD
- #define [DUAL_INOUT_FAST_READ_4_BYTE_ADDR_CMD](#) 0xBC
- #define [QUAD_OUT_FAST_READ_CMD](#) 0x6B
- #define [QUAD_OUT_FAST_READ_DTR_CMD](#) 0x0D
- #define [QUAD_OUT_FAST_READ_4_BYTE_ADDR_CMD](#) 0x6C
- #define [QUAD_INOUT_FAST_READ_CMD](#) 0xEB
- #define [QUAD_INOUT_FAST_READ_DTR_CMD](#) 0xED
- #define [QUAD_INOUT_FAST_READ_4_BYTE_ADDR_CMD](#) 0xEC
- #define [WRITE_ENABLE_CMD](#) 0x06
- #define [WRITE_DISABLE_CMD](#) 0x04
- #define [READ_STATUS_REG_CMD](#) 0x05
- #define [WRITE_STATUS_REG_CMD](#) 0x01
- #define [READ_LOCK_REG_CMD](#) 0xE8
- #define [WRITE_LOCK_REG_CMD](#) 0xE5
- #define [READ_FLAG_STATUS_REG_CMD](#) 0x70
- #define [CLEAR_FLAG_STATUS_REG_CMD](#) 0x50
- #define [READ_NONVOL_CFG_REG_CMD](#) 0xB5
- #define [WRITE_NONVOL_CFG_REG_CMD](#) 0xB1
- #define [READ_READ_PARAM_REG_CMD](#) 0x61
- #define [WRITE_READ_PARAM_REG_CMD](#) 0xC0
- #define [READ_ENHANCED_VOL_CFG_REG_CMD](#) 0x81
- #define [WRITE_ENHANCED_VOL_CFG_REG_CMD](#) 0x85
- #define [READ_EXT_ADDR_REG_CMD](#) 0xC8
- #define [WRITE_EXT_ADDR_REG_CMD](#) 0xC5
- #define [PAGE_PROG_CMD](#) 0x02
- #define [PAGE_PROG_4_BYTE_ADDR_CMD](#) 0x12
- #define [DUAL_IN_FAST_PROG_CMD](#) 0xA2
- #define [EXT_DUAL_IN_FAST_PROG_CMD](#) 0xD2
- #define [QUAD_IN_FAST_PROG_CMD](#) 0x32
- #define [EXT_QUAD_IN_FAST_PROG_CMD](#) 0x38
- #define [QUAD_IN_FAST_PROG_4_BYTE_ADDR_CMD](#) 0x34
- #define [SUBSECTOR_ERASE_CMD](#) 0xd7
- #define [SUBSECTOR_ERASE_QPI_CMD](#) 0x20
- #define [SUBSECTOR_ERASE_4_BYTE_ADDR_CMD](#) 0x21

- #define `SECTOR_ERASE_CMD` 0xD8
- #define `SECTOR_ERASE_4_BYTE_ADDR_CMD` 0xDC
- #define `BLOCK_ERASE_32K_CMD` 0x52
- #define `DIE_ERASE_CMD` 0xC4
- #define `PROG_ERASE_RESUME_CMD` 0x7A
- #define `PROG_ERASE_SUSPEND_CMD` 0x75
- #define `READ_OTP_ARRAY_CMD` 0x4B
- #define `PROG_OTP_ARRAY_CMD` 0x42
- #define `ENTER_4_BYTE_ADDR_MODE_CMD` 0xB7
- #define `EXIT_4_BYTE_ADDR_MODE_CMD` 0xE9
- #define `ENTER_QUAD_CMD` 0x35
- #define `EXIT_QUAD_CMD` 0xF5
- #define `IS25LP064A_SR_WIP` ((uint8_t)0x01)

IS25LP08D Registers

- #define `IS25LP064A_SR_WREN` ((uint8_t)0x02)
- #define `IS25LP064A_SR_SRWREN` ((uint8_t)0x80)
- #define `IS25LP064A_SR_QE` ((uint8_t)0x40)
- #define `IS25LP064A_NVCR_NBADDR` ((uint16_t)0x0001)
- #define `IS25LP064A_NVCR_SEGMENT` ((uint16_t)0x0002)
- #define `IS25LP064A_NVCR_DUAL` ((uint16_t)0x0004)
- #define `IS25LP064A_NVCR_QUAB` ((uint16_t)0x0008)
- #define `IS25LP064A_NVCR_RH` ((uint16_t)0x0010)
- #define `IS25LP064A_NVCR_DTRP` ((uint16_t)0x0020)
- #define `IS25LP064A_NVCR_ODS` ((uint16_t)0x01C0)
- #define `IS25LP064A_NVCR_XIP` ((uint16_t)0x0E00)
- #define `IS25LP064A_NVCR_NB_DUMMY` ((uint16_t)0xF000)
- #define `IS25LP064A_VCR_WRAP` ((uint8_t)0x03)
- #define `IS25LP064A_VCR_XIP` ((uint8_t)0x08)
- #define `IS25LP064A_VCR_NB_DUMMY` ((uint8_t)0xF0)
- #define `IS25LP064A_EAR_HIGHEST_SE` ((uint8_t)0x03)
- #define `IS25LP064A_EAR_THIRD_SEG` ((uint8_t)0x02)
- #define `IS25LP064A_EAR_SECOND_SEG` ((uint8_t)0x01)
- #define `IS25LP064A_EAR_LOWEST_SEG` ((uint8_t)0x00)
- #define `IS25LP064A_EVCR_ODS` ((uint8_t)0x07)
- #define `IS25LP064A_EVCR_RH` ((uint8_t)0x10)
- #define `IS25LP064A_EVCR_DTRP` ((uint8_t)0x20)
- #define `IS25LP064A_EVCR_DUAL` ((uint8_t)0x40)
- #define `IS25LP064A_EVCR_QUAD` ((uint8_t)0x80)
- #define `IS25LP064A_FSR_NBADDR` ((uint8_t)0x01)
- #define `IS25LP064A_FSR_PRERR` ((uint8_t)0x02)
- #define `IS25LP064A_FSR_PGSUS` ((uint8_t)0x04)
- #define `IS25LP064A_FSR_PGERR` ((uint8_t)0x10)
- #define `IS25LP064A_FSR_ERERR` ((uint8_t)0x20)
- #define `IS25LP064A_FSR_ERSUS` ((uint8_t)0x40)
- #define `IS25LP064A_FSR_READY` ((uint8_t)0x80)
- #define `RESET_ENABLE_CMD` 0x66
- #define `RESET_MEMORY_CMD` 0x99
- #define `READ_ID_CMD` 0x9E
- #define `READ_ID_CMD2` 0x9F
- #define `MULTIPLE_IO_READ_ID_CMD` 0xAF
- #define `READ_SERIAL_FLASH_DISCO_PARAM_CMD` 0x5A
- #define `READ_CMD` 0x03
- #define `READ_4_BYTE_ADDR_CMD` 0x13

- #define FAST_READ_CMD 0x0B
- #define FAST_READ_DTR_CMD 0x0D
- #define FAST_READ_4_BYTE_ADDR_CMD 0x0C
- #define DUAL_OUT_FAST_READ_CMD 0x3B
- #define DUAL_OUT_FAST_READ_DTR_CMD 0x3D
- #define DUAL_OUT_FAST_READ_4_BYTE_ADDR_CMD 0x3C
- #define DUAL_INOUT_FAST_READ_CMD 0xBB
- #define DUAL_INOUT_FAST_READ_DTR_CMD 0xBD
- #define DUAL_INOUT_FAST_READ_4_BYTE_ADDR_CMD 0xBC
- #define QUAD_OUT_FAST_READ_CMD 0x6B
- #define QUAD_OUT_FAST_READ_DTR_CMD 0x0D
- #define QUAD_OUT_FAST_READ_4_BYTE_ADDR_CMD 0x6C
- #define QUAD_INOUT_FAST_READ_CMD 0xEB
- #define QUAD_INOUT_FAST_READ_DTR_CMD 0xED
- #define QUAD_INOUT_FAST_READ_4_BYTE_ADDR_CMD 0xEC
- #define WRITE_ENABLE_CMD 0x06
- #define WRITE_DISABLE_CMD 0x04
- #define READ_STATUS_REG_CMD 0x05
- #define WRITE_STATUS_REG_CMD 0x01
- #define READ_LOCK_REG_CMD 0xE8
- #define WRITE_LOCK_REG_CMD 0xE5
- #define READ_FLAG_STATUS_REG_CMD 0x70
- #define CLEAR_FLAG_STATUS_REG_CMD 0x50
- #define READ_NONVOL_CFG_REG_CMD 0xB5
- #define WRITE_NONVOL_CFG_REG_CMD 0xB1
- #define READ_READ_PARAM_REG_CMD 0x61
- #define WRITE_READ_PARAM_REG_CMD 0xC0
- #define READ_ENHANCED_VOL_CFG_REG_CMD 0x81
- #define WRITE_ENHANCED_VOL_CFG_REG_CMD 0x85
- #define READ_EXT_ADDR_REG_CMD 0xC8
- #define WRITE_EXT_ADDR_REG_CMD 0xC5
- #define PAGE_PROG_CMD 0x02
- #define PAGE_PROG_4_BYTE_ADDR_CMD 0x12
- #define DUAL_IN_FAST_PROG_CMD 0xA2
- #define EXT_DUAL_IN_FAST_PROG_CMD 0xD2
- #define QUAD_IN_FAST_PROG_CMD 0x32
- #define EXT_QUAD_IN_FAST_PROG_CMD 0x38
- #define QUAD_IN_FAST_PROG_4_BYTE_ADDR_CMD 0x34
- #define SUBSECTOR_ERASE_CMD 0xd7
- #define SUBSECTOR_ERASE_QPI_CMD 0x20
- #define SUBSECTOR_ERASE_4_BYTE_ADDR_CMD 0x21
- #define SECTOR_ERASE_CMD 0xD8
- #define SECTOR_ERASE_4_BYTE_ADDR_CMD 0xDC
- #define BLOCK_ERASE_32K_CMD 0x52
- #define DIE_ERASE_CMD 0xC4
- #define PROG_ERASE_RESUME_CMD 0x7A
- #define PROG_ERASE_SUSPEND_CMD 0x75
- #define READ_OTP_ARRAY_CMD 0x4B
- #define PROG_OTP_ARRAY_CMD 0x42
- #define ENTER_4_BYTE_ADDR_MODE_CMD 0xB7
- #define EXIT_4_BYTE_ADDR_MODE_CMD 0xE9
- #define ENTER_QUAD_CMD 0x35
- #define EXIT_QUAD_CMD 0xF5
- #define IS25LP080D_SR_WIP ((uint8_t)0x01)

IS25LP08D Registers

- `#define IS25LP080D_SR_WREN ((uint8_t)0x02)`
- `#define IS25LP080D_SR_SRWREN ((uint8_t)0x80)`
- `#define IS25LP080D_SR_QE ((uint8_t)0x40)`
- `#define IS25LP080D_NVCR_NBADDR ((uint16_t)0x0001)`
- `#define IS25LP080D_NVCR_SEGMENT ((uint16_t)0x0002)`
- `#define IS25LP080D_NVCR_DUAL ((uint16_t)0x0004)`
- `#define IS25LP080D_NVCR_QUAB ((uint16_t)0x0008)`
- `#define IS25LP080D_NVCR_RH ((uint16_t)0x0010)`
- `#define IS25LP080D_NVCR_DTRP ((uint16_t)0x0020)`
- `#define IS25LP080D_NVCR_ODS ((uint16_t)0x01C0)`
- `#define IS25LP080D_NVCR_XIP ((uint16_t)0x0E00)`
- `#define IS25LP080D_NVCR_NB_DUMMY ((uint16_t)0xF000)`
- `#define IS25LP080D_VCR_WRAP ((uint8_t)0x03)`
- `#define IS25LP080D_VCR_XIP ((uint8_t)0x08)`
- `#define IS25LP080D_VCR_NB_DUMMY ((uint8_t)0xF0)`
- `#define IS25LP080D_EAR_HIGHEST_SE ((uint8_t)0x03)`
- `#define IS25LP080D_EAR_THIRD_SEG ((uint8_t)0x02)`
- `#define IS25LP080D_EAR_SECOND_SEG ((uint8_t)0x01)`
- `#define IS25LP080D_EAR_LOWEST_SEG ((uint8_t)0x00)`
- `#define IS25LP080D_EVCR_ODS ((uint8_t)0x07)`
- `#define IS25LP080D_EVCR_RH ((uint8_t)0x10)`
- `#define IS25LP080D_EVCR_DTRP ((uint8_t)0x20)`
- `#define IS25LP080D_EVCR_DUAL ((uint8_t)0x40)`
- `#define IS25LP080D_EVCR_QUAD ((uint8_t)0x80)`
- `#define IS25LP080D_FSR_NBADDR ((uint8_t)0x01)`
- `#define IS25LP080D_FSR_PRERR ((uint8_t)0x02)`
- `#define IS25LP080D_FSR_PGSUS ((uint8_t)0x04)`
- `#define IS25LP080D_FSR_PGERR ((uint8_t)0x10)`
- `#define IS25LP080D_FSR_ERERR ((uint8_t)0x20)`
- `#define IS25LP080D_FSR_ERSUS ((uint8_t)0x40)`
- `#define IS25LP080D_FSR_READY ((uint8_t)0x80)`

6.14.1 Detailed Description

Flash memory.

IS25LP08D Commands

6.14.2 Macro Definition Documentation**6.14.2.1 BLOCK_ERASE_32K_CMD [1/2]**

```
#define BLOCK_ERASE_32K_CMD 0x52
&
```

6.14.2.2 BLOCK_ERASE_32K_CMD [2/2]

```
#define BLOCK_ERASE_32K_CMD 0x52
&
```

6.14.2.3 CLEAR_FLAG_STATUS_REG_CMD [1/2]

```
#define CLEAR_FLAG_STATUS_REG_CMD 0x50
&
```


6.14.2.4 CLEAR_FLAG_STATUS_REG_CMD [2/2]

```
#define CLEAR_FLAG_STATUS_REG_CMD 0x50
&
```

6.14.2.5 DIE_ERASE_CMD [1/2]

```
#define DIE_ERASE_CMD 0xC4
&
```

6.14.2.6 DIE_ERASE_CMD [2/2]

```
#define DIE_ERASE_CMD 0xC4
&
```

6.14.2.7 DUAL_IN_FAST_PROG_CMD [1/2]

```
#define DUAL_IN_FAST_PROG_CMD 0xA2
&
```

6.14.2.8 DUAL_IN_FAST_PROG_CMD [2/2]

```
#define DUAL_IN_FAST_PROG_CMD 0xA2
&
```

6.14.2.9 DUAL_INOUT_FAST_READ_4_BYTE_ADDR_CMD [1/2]

```
#define DUAL_INOUT_FAST_READ_4_BYTE_ADDR_CMD 0xBC
&
```

6.14.2.10 DUAL_INOUT_FAST_READ_4_BYTE_ADDR_CMD [2/2]

```
#define DUAL_INOUT_FAST_READ_4_BYTE_ADDR_CMD 0xBC
&
```

6.14.2.11 DUAL_INOUT_FAST_READ_CMD [1/2]

```
#define DUAL_INOUT_FAST_READ_CMD 0xBB
&
```

6.14.2.12 DUAL_INOUT_FAST_READ_CMD [2/2]

```
#define DUAL_INOUT_FAST_READ_CMD 0xBB
&
```

6.14.2.13 DUAL_INOUT_FAST_READ_DTR_CMD [1/2]

```
#define DUAL_INOUT_FAST_READ_DTR_CMD 0xBD
&
```

6.14.2.14 DUAL_INOUT_FAST_READ_DTR_CMD [2/2]

```
#define DUAL_INOUT_FAST_READ_DTR_CMD 0xBD
&
```

6.14.2.15 DUAL_OUT_FAST_READ_4_BYTE_ADDR_CMD [1/2]

```
#define DUAL_OUT_FAST_READ_4_BYTE_ADDR_CMD 0x3C
&
```

6.14.2.16 DUAL_OUT_FAST_READ_4_BYTE_ADDR_CMD [2/2]

```
#define DUAL_OUT_FAST_READ_4_BYTE_ADDR_CMD 0x3C
&
```

6.14.2.17 DUAL_OUT_FAST_READ_CMD [1/2]

```
#define DUAL_OUT_FAST_READ_CMD 0x3B
&
```

6.14.2.18 DUAL_OUT_FAST_READ_CMD [2/2]

```
#define DUAL_OUT_FAST_READ_CMD 0x3B
&
```

6.14.2.19 DUAL_OUT_FAST_READ_DTR_CMD [1/2]

```
#define DUAL_OUT_FAST_READ_DTR_CMD 0x3D
&
```

6.14.2.20 DUAL_OUT_FAST_READ_DTR_CMD [2/2]

```
#define DUAL_OUT_FAST_READ_DTR_CMD 0x3D
&
```

6.14.2.21 ENTER_4_BYTE_ADDR_MODE_CMD [1/2]

```
#define ENTER_4_BYTE_ADDR_MODE_CMD 0xB7
4-byte Address Mode Operations
```

6.14.2.22 ENTER_4_BYTE_ADDR_MODE_CMD [2/2]

```
#define ENTER_4_BYTE_ADDR_MODE_CMD 0xB7
4-byte Address Mode Operations
```

6.14.2.23 ENTER_QUAD_CMD [1/2]

```
#define ENTER_QUAD_CMD 0x35
Quad Operations
```

6.14.2.24 ENTER_QUAD_CMD [2/2]

```
#define ENTER_QUAD_CMD 0x35
Quad Operations
```

6.14.2.25 EXIT_4_BYTE_ADDR_MODE_CMD [1/2]

```
#define EXIT_4_BYTE_ADDR_MODE_CMD 0xE9
&
```

6.14.2.26 EXIT_4_BYTE_ADDR_MODE_CMD [2/2]

```
#define EXIT_4_BYTE_ADDR_MODE_CMD 0xE9
&
```

6.14.2.27 EXIT_QUAD_CMD [1/2]

```
#define EXIT_QUAD_CMD 0xF5
&
```

6.14.2.28 EXIT_QUAD_CMD [2/2]

```
#define EXIT_QUAD_CMD 0xF5
&
```

6.14.2.29 EXT_DUAL_IN_FAST_PROG_CMD [1/2]

```
#define EXT_DUAL_IN_FAST_PROG_CMD 0xD2
&
```

6.14.2.30 EXT_DUAL_IN_FAST_PROG_CMD [2/2]

```
#define EXT_DUAL_IN_FAST_PROG_CMD 0xD2
&
```

6.14.2.31 EXT_QUAD_IN_FAST_PROG_CMD [1/2]

```
#define EXT_QUAD_IN_FAST_PROG_CMD 0x38
&
```

6.14.2.32 EXT_QUAD_IN_FAST_PROG_CMD [2/2]

```
#define EXT_QUAD_IN_FAST_PROG_CMD 0x38
&
```

6.14.2.33 FAST_READ_4_BYTE_ADDR_CMD [1/2]

```
#define FAST_READ_4_BYTE_ADDR_CMD 0x0C
&
```

6.14.2.34 FAST_READ_4_BYTE_ADDR_CMD [2/2]

```
#define FAST_READ_4_BYTE_ADDR_CMD 0x0C
&
```

6.14.2.35 FAST_READ_CMD [1/2]

```
#define FAST_READ_CMD 0x0B
&
```

6.14.2.36 FAST_READ_CMD [2/2]

```
#define FAST_READ_CMD 0x0B
&
```

6.14.2.37 FAST_READ_DTR_CMD [1/2]

```
#define FAST_READ_DTR_CMD 0x0D
&
```

6.14.2.38 FAST_READ_DTR_CMD [2/2]

```
#define FAST_READ_DTR_CMD 0x0D
&
```

6.14.2.39 IS25LP064A_EAR_HIGHEST_SE

```
#define IS25LP064A_EAR_HIGHEST_SE ((uint8_t)0x03)
Select the Highest 128Mb segment
```

6.14.2.40 IS25LP064A_EAR_LOWEST_SEG

```
#define IS25LP064A_EAR_LOWEST_SEG ((uint8_t)0x00)
```

Select the Lowest 128Mb segment (default)

6.14.2.41 IS25LP064A_EAR_SECOND_SEG

```
#define IS25LP064A_EAR_SECOND_SEG ((uint8_t)0x01)
```

Select the Second 128Mb segment

6.14.2.42 IS25LP064A_EAR_THIRD_SEG

```
#define IS25LP064A_EAR_THIRD_SEG ((uint8_t)0x02)
```

Select the Third 128Mb segment

6.14.2.43 IS25LP064A_EVCR_DTRP

```
#define IS25LP064A_EVCR_DTRP ((uint8_t)0x20)
```

Double transfer rate protocol

6.14.2.44 IS25LP064A_EVCR_DUAL

```
#define IS25LP064A_EVCR_DUAL ((uint8_t)0x40)
```

Dual I/O protocol

6.14.2.45 IS25LP064A_EVCR_ODS

```
#define IS25LP064A_EVCR_ODS ((uint8_t)0x07)
```

Output driver strength

6.14.2.46 IS25LP064A_EVCR_QUAD

```
#define IS25LP064A_EVCR_QUAD ((uint8_t)0x80)
```

Quad I/O protocol

6.14.2.47 IS25LP064A_EVCR_RH

```
#define IS25LP064A_EVCR_RH ((uint8_t)0x10)
```

Reset/hold

6.14.2.48 IS25LP064A_FSR_ERERR

```
#define IS25LP064A_FSR_ERERR ((uint8_t)0x20)
```

Erase error

6.14.2.49 IS25LP064A_FSR_ERSUS

```
#define IS25LP064A_FSR_ERSUS ((uint8_t)0x40)
```

Erase operation suspended

6.14.2.50 IS25LP064A_FSR_NBADDR

```
#define IS25LP064A_FSR_NBADDR ((uint8_t)0x01)
```

3-bytes or 4-bytes addressing

6.14.2.51 IS25LP064A_FSR_PGERR

```
#define IS25LP064A_FSR_PGERR ((uint8_t)0x10)
```

Program error

6.14.2.52 IS25LP064A_FSR_PGSUS

```
#define IS25LP064A_FSR_PGSUS ((uint8_t)0x04)
```

Program operation suspended

6.14.2.53 IS25LP064A_FSR_PRERR

```
#define IS25LP064A_FSR_PRERR ((uint8_t)0x02)
```

Protection error

6.14.2.54 IS25LP064A_FSR_READY

```
#define IS25LP064A_FSR_READY ((uint8_t)0x80)
```

Ready or command in progress

6.14.2.55 IS25LP064A_NVCR_DTRP

```
#define IS25LP064A_NVCR_DTRP ((uint16_t)0x0020)
```

Double transfer rate protocol

6.14.2.56 IS25LP064A_NVCR_DUAL

```
#define IS25LP064A_NVCR_DUAL ((uint16_t)0x0004)
```

Dual I/O protocol

6.14.2.57 IS25LP064A_NVCR_NB_DUMMY

```
#define IS25LP064A_NVCR_NB_DUMMY ((uint16_t)0xF000)
```

Number of dummy clock cycles

6.14.2.58 IS25LP064A_NVCR_NBADDR

```
#define IS25LP064A_NVCR_NBADDR ((uint16_t)0x0001)
```

3-bytes or 4-bytes addressing

6.14.2.59 IS25LP064A_NVCR_ODS

```
#define IS25LP064A_NVCR_ODS ((uint16_t)0x01C0)
```

Output driver strength

6.14.2.60 IS25LP064A_NVCR_QUAB

```
#define IS25LP064A_NVCR_QUAB ((uint16_t)0x0008)
```

Quad I/O protocol

6.14.2.61 IS25LP064A_NVCR_RH

```
#define IS25LP064A_NVCR_RH ((uint16_t)0x0010)
```

Reset/hold

6.14.2.62 IS25LP064A_NVCR_SEGMENT

```
#define IS25LP064A_NVCR_SEGMENT ((uint16_t)0x0002)
```

Upper or lower 128Mb segment selected by default

6.14.2.63 IS25LP064A_NVCR_XIP

```
#define IS25LP064A_NVCR_XIP ((uint16_t)0x0E00)
```

XIP mode at power-on reset

6.14.2.64 IS25LP064A_SR_QE

```
#define IS25LP064A_SR_QE ((uint8_t)0x40)
&
```

6.14.2.65 IS25LP064A_SR_SRWREN

```
#define IS25LP064A_SR_SRWREN ((uint8_t)0x80)
Status register write enable/disable
```

6.14.2.66 IS25LP064A_SR_WIP

```
#define IS25LP064A_SR_WIP ((uint8_t)0x01)
IS25LP08D Registers
```

Write in progress

6.14.2.67 IS25LP064A_SR_WREN

```
#define IS25LP064A_SR_WREN ((uint8_t)0x02)
Write enable latch
```

6.14.2.68 IS25LP064A_VCR_NB_DUMMY

```
#define IS25LP064A_VCR_NB_DUMMY ((uint8_t)0xF0)
Number of dummy clock cycles
```

6.14.2.69 IS25LP064A_VCR_WRAP

```
#define IS25LP064A_VCR_WRAP ((uint8_t)0x03)
Wrap
```

6.14.2.70 IS25LP064A_VCR_XIP

```
#define IS25LP064A_VCR_XIP ((uint8_t)0x08)
XIP
```

6.14.2.71 IS25LP080D_EAR_HIGHEST_SE

```
#define IS25LP080D_EAR_HIGHEST_SE ((uint8_t)0x03)
Select the Highest 128Mb segment
```

6.14.2.72 IS25LP080D_EAR_LOWEST_SEG

```
#define IS25LP080D_EAR_LOWEST_SEG ((uint8_t)0x00)
Select the Lowest 128Mb segment (default)
```

6.14.2.73 IS25LP080D_EAR_SECOND_SEG

```
#define IS25LP080D_EAR_SECOND_SEG ((uint8_t)0x01)
Select the Second 128Mb segment
```

6.14.2.74 IS25LP080D_EAR_THIRD_SEG

```
#define IS25LP080D_EAR_THIRD_SEG ((uint8_t)0x02)
Select the Third 128Mb segment
```

6.14.2.75 IS25LP080D_EVCR_DTRP

```
#define IS25LP080D_EVCR_DTRP ((uint8_t)0x20)
Double transfer rate protocol
```

6.14.2.76 IS25LP080D_EVCR_DUAL

```
#define IS25LP080D_EVCR_DUAL ((uint8_t)0x40)
```

Dual I/O protocol

6.14.2.77 IS25LP080D_EVCR_ODS

```
#define IS25LP080D_EVCR_ODS ((uint8_t)0x07)
```

Output driver strength

6.14.2.78 IS25LP080D_EVCR_QUAD

```
#define IS25LP080D_EVCR_QUAD ((uint8_t)0x80)
```

Quad I/O protocol

6.14.2.79 IS25LP080D_EVCR_RH

```
#define IS25LP080D_EVCR_RH ((uint8_t)0x10)
```

Reset/hold

6.14.2.80 IS25LP080D_FSR_ERERR

```
#define IS25LP080D_FSR_ERERR ((uint8_t)0x20)
```

Erase error

6.14.2.81 IS25LP080D_FSR_ERSUS

```
#define IS25LP080D_FSR_ERSUS ((uint8_t)0x40)
```

Erase operation suspended

6.14.2.82 IS25LP080D_FSR_NBADDR

```
#define IS25LP080D_FSR_NBADDR ((uint8_t)0x01)
```

3-bytes or 4-bytes addressing

6.14.2.83 IS25LP080D_FSR_PGERR

```
#define IS25LP080D_FSR_PGERR ((uint8_t)0x10)
```

Program error

6.14.2.84 IS25LP080D_FSR_PGSUS

```
#define IS25LP080D_FSR_PGSUS ((uint8_t)0x04)
```

Program operation suspended

6.14.2.85 IS25LP080D_FSR_PRERR

```
#define IS25LP080D_FSR_PRERR ((uint8_t)0x02)
```

Protection error

6.14.2.86 IS25LP080D_FSR_READY

```
#define IS25LP080D_FSR_READY ((uint8_t)0x80)
```

Ready or command in progress

6.14.2.87 IS25LP080D_NVCR_DTRP

```
#define IS25LP080D_NVCR_DTRP ((uint16_t)0x0020)
```

Double transfer rate protocol

6.14.2.88 IS25LP080D_NVCR_DUAL

```
#define IS25LP080D_NVCR_DUAL ((uint16_t)0x0004)
```

Dual I/O protocol

6.14.2.89 IS25LP080D_NVCR_NB_DUMMY

```
#define IS25LP080D_NVCR_NB_DUMMY ((uint16_t)0xF000)
```

Number of dummy clock cycles

6.14.2.90 IS25LP080D_NVCR_NBADDR

```
#define IS25LP080D_NVCR_NBADDR ((uint16_t)0x0001)
```

3-bytes or 4-bytes addressing

6.14.2.91 IS25LP080D_NVCR_ODS

```
#define IS25LP080D_NVCR_ODS ((uint16_t)0x01C0)
```

Output driver strength

6.14.2.92 IS25LP080D_NVCR_QUAB

```
#define IS25LP080D_NVCR_QUAB ((uint16_t)0x0008)
```

Quad I/O protocol

6.14.2.93 IS25LP080D_NVCR_RH

```
#define IS25LP080D_NVCR_RH ((uint16_t)0x0010)
```

Reset/hold

6.14.2.94 IS25LP080D_NVCR_SEGMENT

```
#define IS25LP080D_NVCR_SEGMENT ((uint16_t)0x0002)
```

Upper or lower 128Mb segment selected by default

6.14.2.95 IS25LP080D_NVCR_XIP

```
#define IS25LP080D_NVCR_XIP ((uint16_t)0x0E00)
```

XIP mode at power-on reset

6.14.2.96 IS25LP080D_SR_QE

```
#define IS25LP080D_SR_QE ((uint8_t)0x40)
```

&

6.14.2.97 IS25LP080D_SR_SRWREN

```
#define IS25LP080D_SR_SRWREN ((uint8_t)0x80)
```

Status register write enable/disable

6.14.2.98 IS25LP080D_SR_WIP

```
#define IS25LP080D_SR_WIP ((uint8_t)0x01)
```

IS25LP08D Registers

Status Register Write in progress

6.14.2.99 IS25LP080D_SR_WREN

```
#define IS25LP080D_SR_WREN ((uint8_t)0x02)
```

Write enable latch

6.14.2.100 IS25LP080D_VCR_NB_DUMMY

```
#define IS25LP080D_VCR_NB_DUMMY ((uint8_t)0xF0)
Number of dummy clock cycles
```

6.14.2.101 IS25LP080D_VCR_WRAP

```
#define IS25LP080D_VCR_WRAP ((uint8_t)0x03)
Wrap
```

6.14.2.102 IS25LP080D_VCR_XIP

```
#define IS25LP080D_VCR_XIP ((uint8_t)0x08)
XIP
```

6.14.2.103 MULTIPLE_IO_READ_ID_CMD [1/2]

```
#define MULTIPLE_IO_READ_ID_CMD 0xAF
&
```

6.14.2.104 MULTIPLE_IO_READ_ID_CMD [2/2]

```
#define MULTIPLE_IO_READ_ID_CMD 0xAF
&
```

6.14.2.105 PAGE_PROG_4_BYTE_ADDR_CMD [1/2]

```
#define PAGE_PROG_4_BYTE_ADDR_CMD 0x12
&
```

6.14.2.106 PAGE_PROG_4_BYTE_ADDR_CMD [2/2]

```
#define PAGE_PROG_4_BYTE_ADDR_CMD 0x12
&
```

6.14.2.107 PAGE_PROG_CMD [1/2]

```
#define PAGE_PROG_CMD 0x02
Program Operations
```

6.14.2.108 PAGE_PROG_CMD [2/2]

```
#define PAGE_PROG_CMD 0x02
Program Operations
```

6.14.2.109 PROG_ERASE_RESUME_CMD [1/2]

```
#define PROG_ERASE_RESUME_CMD 0x7A
&
```

6.14.2.110 PROG_ERASE_RESUME_CMD [2/2]

```
#define PROG_ERASE_RESUME_CMD 0x7A
&
```

6.14.2.111 PROG_ERASE_SUSPEND_CMD [1/2]

```
#define PROG_ERASE_SUSPEND_CMD 0x75
&
```

6.14.2.112 PROG_ERASE_SUSPEND_CMD [2/2]

```
#define PROG_ERASE_SUSPEND_CMD 0x75
&
```

6.14.2.113 PROG_OTP_ARRAY_CMD [1/2]

```
#define PROG_OTP_ARRAY_CMD 0x42
&
```

6.14.2.114 PROG_OTP_ARRAY_CMD [2/2]

```
#define PROG_OTP_ARRAY_CMD 0x42
&
```

6.14.2.115 QUAD_IN_FAST_PROG_4_BYTE_ADDR_CMD [1/2]

```
#define QUAD_IN_FAST_PROG_4_BYTE_ADDR_CMD 0x34
&
```

6.14.2.116 QUAD_IN_FAST_PROG_4_BYTE_ADDR_CMD [2/2]

```
#define QUAD_IN_FAST_PROG_4_BYTE_ADDR_CMD 0x34
&
```

6.14.2.117 QUAD_IN_FAST_PROG_CMD [1/2]

```
#define QUAD_IN_FAST_PROG_CMD 0x32
&
```

6.14.2.118 QUAD_IN_FAST_PROG_CMD [2/2]

```
#define QUAD_IN_FAST_PROG_CMD 0x32
&
```

6.14.2.119 QUAD_INOUT_FAST_READ_4_BYTE_ADDR_CMD [1/2]

```
#define QUAD_INOUT_FAST_READ_4_BYTE_ADDR_CMD 0xEC
&
```

6.14.2.120 QUAD_INOUT_FAST_READ_4_BYTE_ADDR_CMD [2/2]

```
#define QUAD_INOUT_FAST_READ_4_BYTE_ADDR_CMD 0xEC
&
```

6.14.2.121 QUAD_INOUT_FAST_READ_CMD [1/2]

```
#define QUAD_INOUT_FAST_READ_CMD 0xEB
&
```

6.14.2.122 QUAD_INOUT_FAST_READ_CMD [2/2]

```
#define QUAD_INOUT_FAST_READ_CMD 0xEB
&
```

6.14.2.123 QUAD_INOUT_FAST_READ_DTR_CMD [1/2]

```
#define QUAD_INOUT_FAST_READ_DTR_CMD 0xED
&
```

6.14.2.124 QUAD_INOUT_FAST_READ_DTR_CMD [2/2]

```
#define QUAD_INOUT_FAST_READ_DTR_CMD 0xED
&
```

6.14.2.125 QUAD_OUT_FAST_READ_4_BYTE_ADDR_CMD [1/2]

```
#define QUAD_OUT_FAST_READ_4_BYTE_ADDR_CMD 0x6C
&
```

6.14.2.126 QUAD_OUT_FAST_READ_4_BYTE_ADDR_CMD [2/2]

```
#define QUAD_OUT_FAST_READ_4_BYTE_ADDR_CMD 0x6C
&
```

6.14.2.127 QUAD_OUT_FAST_READ_CMD [1/2]

```
#define QUAD_OUT_FAST_READ_CMD 0x6B
&
```

6.14.2.128 QUAD_OUT_FAST_READ_CMD [2/2]

```
#define QUAD_OUT_FAST_READ_CMD 0x6B
&
```

6.14.2.129 QUAD_OUT_FAST_READ_DTR_CMD [1/2]

```
#define QUAD_OUT_FAST_READ_DTR_CMD 0x0D
&
```

6.14.2.130 QUAD_OUT_FAST_READ_DTR_CMD [2/2]

```
#define QUAD_OUT_FAST_READ_DTR_CMD 0x0D
&
```

6.14.2.131 READ_4_BYTE_ADDR_CMD [1/2]

```
#define READ_4_BYTE_ADDR_CMD 0x13
&
```

6.14.2.132 READ_4_BYTE_ADDR_CMD [2/2]

```
#define READ_4_BYTE_ADDR_CMD 0x13
&
```

6.14.2.133 READ_CMD [1/2]

```
#define READ_CMD 0x03
Read Operations
```

6.14.2.134 READ_CMD [2/2]

```
#define READ_CMD 0x03
Read Operations
```

6.14.2.135 READ_ENHANCED_VOL_CFG_REG_CMD [1/2]

```
#define READ_ENHANCED_VOL_CFG_REG_CMD 0x81
&
```

6.14.2.136 READ_ENHANCED_VOL_CFG_REG_CMD [2/2]

```
#define READ_ENHANCED_VOL_CFG_REG_CMD 0x81
&
```

6.14.2.137 READ_EXT_ADDR_REG_CMD [1/2]

```
#define READ_EXT_ADDR_REG_CMD 0xC8
&
```

6.14.2.138 READ_EXT_ADDR_REG_CMD [2/2]

```
#define READ_EXT_ADDR_REG_CMD 0xC8
&
```

6.14.2.139 READ_FLAG_STATUS_REG_CMD [1/2]

```
#define READ_FLAG_STATUS_REG_CMD 0x70
&
```

6.14.2.140 READ_FLAG_STATUS_REG_CMD [2/2]

```
#define READ_FLAG_STATUS_REG_CMD 0x70
&
```

6.14.2.141 READ_ID_CMD [1/2]

```
#define READ_ID_CMD 0x9E
Identification Operations
```

6.14.2.142 READ_ID_CMD [2/2]

```
#define READ_ID_CMD 0x9E
Identification Operations
```

6.14.2.143 READ_ID_CMD2 [1/2]

```
#define READ_ID_CMD2 0x9F
&
```

6.14.2.144 READ_ID_CMD2 [2/2]

```
#define READ_ID_CMD2 0x9F
&
```

6.14.2.145 READ_LOCK_REG_CMD [1/2]

```
#define READ_LOCK_REG_CMD 0xE8
&
```

6.14.2.146 READ_LOCK_REG_CMD [2/2]

```
#define READ_LOCK_REG_CMD 0xE8
&
```

6.14.2.147 READ_NONVOL_CFG_REG_CMD [1/2]

```
#define READ_NONVOL_CFG_REG_CMD 0xB5
&
```

6.14.2.148 READ_NONVOL_CFG_REG_CMD [2/2]

```
#define READ_NONVOL_CFG_REG_CMD 0xB5  
&
```

6.14.2.149 READ_OTP_ARRAY_CMD [1/2]

```
#define READ_OTP_ARRAY_CMD 0x4B  
One-Time Programmable Operations
```

6.14.2.150 READ_OTP_ARRAY_CMD [2/2]

```
#define READ_OTP_ARRAY_CMD 0x4B  
One-Time Programmable Operations
```

6.14.2.151 READ_READ_PARAM_REG_CMD [1/2]

```
#define READ_READ_PARAM_REG_CMD 0x61  
&
```

6.14.2.152 READ_READ_PARAM_REG_CMD [2/2]

```
#define READ_READ_PARAM_REG_CMD 0x61  
&
```

6.14.2.153 READ_SERIAL_FLASH_DISCO_PARAM_CMD [1/2]

```
#define READ_SERIAL_FLASH_DISCO_PARAM_CMD 0x5A  
&
```

6.14.2.154 READ_SERIAL_FLASH_DISCO_PARAM_CMD [2/2]

```
#define READ_SERIAL_FLASH_DISCO_PARAM_CMD 0x5A  
&
```

6.14.2.155 READ_STATUS_REG_CMD [1/2]

```
#define READ_STATUS_REG_CMD 0x05  
Register Operations
```

6.14.2.156 READ_STATUS_REG_CMD [2/2]

```
#define READ_STATUS_REG_CMD 0x05  
Register Operations
```

6.14.2.157 RESET_ENABLE_CMD [1/2]

```
#define RESET_ENABLE_CMD 0x66  
Reset Operations
```

6.14.2.158 RESET_ENABLE_CMD [2/2]

```
#define RESET_ENABLE_CMD 0x66  
Reset Operations
```

6.14.2.159 RESET_MEMORY_CMD [1/2]

```
#define RESET_MEMORY_CMD 0x99  
&
```

6.14.2.160 RESET_MEMORY_CMD [2/2]

```
#define RESET_MEMORY_CMD 0x99
&
```

6.14.2.161 SECTOR_ERASE_4_BYTE_ADDR_CMD [1/2]

```
#define SECTOR_ERASE_4_BYTE_ADDR_CMD 0xDC
&
```

6.14.2.162 SECTOR_ERASE_4_BYTE_ADDR_CMD [2/2]

```
#define SECTOR_ERASE_4_BYTE_ADDR_CMD 0xDC
&
```

6.14.2.163 SECTOR_ERASE_CMD [1/2]

```
#define SECTOR_ERASE_CMD 0xD8
&
```

6.14.2.164 SECTOR_ERASE_CMD [2/2]

```
#define SECTOR_ERASE_CMD 0xD8
&
```

6.14.2.165 SUBSECTOR_ERASE_4_BYTE_ADDR_CMD [1/2]

```
#define SUBSECTOR_ERASE_4_BYTE_ADDR_CMD 0x21
&
```

6.14.2.166 SUBSECTOR_ERASE_4_BYTE_ADDR_CMD [2/2]

```
#define SUBSECTOR_ERASE_4_BYTE_ADDR_CMD 0x21
&
```

6.14.2.167 SUBSECTOR_ERASE_CMD [1/2]

```
#define SUBSECTOR_ERASE_CMD 0xd7
Erase Operations
```

6.14.2.168 SUBSECTOR_ERASE_CMD [2/2]

```
#define SUBSECTOR_ERASE_CMD 0xd7
Erase Operations
```

6.14.2.169 SUBSECTOR_ERASE_QPI_CMD [1/2]

```
#define SUBSECTOR_ERASE_QPI_CMD 0x20
&
```

6.14.2.170 SUBSECTOR_ERASE_QPI_CMD [2/2]

```
#define SUBSECTOR_ERASE_QPI_CMD 0x20
&
```

6.14.2.171 WRITE_DISABLE_CMD [1/2]

```
#define WRITE_DISABLE_CMD 0x04
&
```

6.14.2.172 WRITE_DISABLE_CMD [2/2]

```
#define WRITE_DISABLE_CMD 0x04
&
```

6.14.2.173 WRITE_ENABLE_CMD [1/2]

```
#define WRITE_ENABLE_CMD 0x06
Write Operations
```

6.14.2.174 WRITE_ENABLE_CMD [2/2]

```
#define WRITE_ENABLE_CMD 0x06
Write Operations
```

6.14.2.175 WRITE_ENHANCED_VOL_CFG_REG_CMD [1/2]

```
#define WRITE_ENHANCED_VOL_CFG_REG_CMD 0x85
&
```

6.14.2.176 WRITE_ENHANCED_VOL_CFG_REG_CMD [2/2]

```
#define WRITE_ENHANCED_VOL_CFG_REG_CMD 0x85
&
```

6.14.2.177 WRITE_EXT_ADDR_REG_CMD [1/2]

```
#define WRITE_EXT_ADDR_REG_CMD 0xC5
&
```

6.14.2.178 WRITE_EXT_ADDR_REG_CMD [2/2]

```
#define WRITE_EXT_ADDR_REG_CMD 0xC5
&
```

6.14.2.179 WRITE_LOCK_REG_CMD [1/2]

```
#define WRITE_LOCK_REG_CMD 0xE5
&
```

6.14.2.180 WRITE_LOCK_REG_CMD [2/2]

```
#define WRITE_LOCK_REG_CMD 0xE5
&
```

6.14.2.181 WRITE_NONVOL_CFG_REG_CMD [1/2]

```
#define WRITE_NONVOL_CFG_REG_CMD 0xB1
&
```

6.14.2.182 WRITE_NONVOL_CFG_REG_CMD [2/2]

```
#define WRITE_NONVOL_CFG_REG_CMD 0xB1
&
```

6.14.2.183 WRITE_READ_PARAM_REG_CMD [1/2]

```
#define WRITE_READ_PARAM_REG_CMD 0xC0
&
```

6.14.2.184 WRITE_READ_PARAM_REG_CMD [2/2]

```
#define WRITE_READ_PARAM_REG_CMD 0xC0  
&
```

6.14.2.185 WRITE_STATUS_REG_CMD [1/2]

```
#define WRITE_STATUS_REG_CMD 0x01  
&
```

6.14.2.186 WRITE_STATUS_REG_CMD [2/2]

```
#define WRITE_STATUS_REG_CMD 0x01  
&
```


6.15 CODEC

Audio codecs.

Classes

- struct [codec_frame_t](#)

Typedefs

- typedef void(* [sa_audio_callback](#)) ([codec_frame_t](#) *, [codec_frame_t](#) *, [size_t](#))

Functions

- void [codec_ak4556_init](#) ([dsy_gpio_pin](#) reset_pin)
- void [codec_pcm3060_init](#) ([dsy_i2c_handle](#) *hi2c)
- [uint8_t](#) [codec_wm8731_init](#) ([dsy_i2c_handle](#) *hi2c, [uint8_t](#) mcu_is_master, [int32_t](#) sample_rate, [uint8_t](#) bitdepth)
- [uint8_t](#) [codec_wm8731_enter_bypass](#) ([dsy_i2c_handle](#) *hi2c)
- [uint8_t](#) [codec_wm8731_exit_bypass](#) ([dsy_i2c_handle](#) *hi2c)

6.15.1 Detailed Description

Audio codecs.

WM8731 Codec framework.

Driver for the WM8731 Codec.

Driver for the PCM3060 Codec.

Driver for the AK4556 Stereo Codec.

6.15.2 Typedef Documentation

6.15.2.1 sa_audio_callback

```
typedef void(* sa_audio_callback) (codec\_frame\_t *, codec\_frame\_t *, size\_t)
&
```

6.15.3 Function Documentation

6.15.3.1 codec_ak4556_init()

```
void codec_ak4556_init (
    dsy\_gpio\_pin reset_pin )
```

Resets the AK4556

Parameters

<i>reset_pin</i>	should be a dsy_gpio_pin that is connected to the RST pin of the AK4556
------------------	---

6.15.3.2 codec_pcm3060_init()

```
void codec_pcm3060_init (
    dsy\_i2c\_handle * hi2c )
```

Resets the PCM060

Parameters

<i>*hi2c</i>	array of pins handling i2c?
--------------	-----------------------------

6.15.3.3 codec_wm8731_enter_bypass()

```
uint8_t codec_wm8731_enter_bypass (
    dsy_i2c_handle * hi2c )
```

Put codec into bypass mode

Parameters

<i>*hi2c</i>	pins handling i2c
--------------	-------------------

6.15.3.4 codec_wm8731_exit_bypass()

```
uint8_t codec_wm8731_exit_bypass (
    dsy_i2c_handle * hi2c )
```

Take codec out of bypass mode

Parameters

<i>*hi2c</i>	pins handling i2c
--------------	-------------------

6.15.3.5 codec_wm8731_init()

```
uint8_t codec_wm8731_init (
    dsy_i2c_handle * hi2c,
    uint8_t mcu_is_master,
    int32_t sample_rate,
    uint8_t bitdepth )
```

Resets the WM8731

Parameters

<i>*hi2c</i>	array of pins handling i2c?
<i>mcu_is_master</i>	&
<i>sample_rate</i>	Sample rate to run codec at
<i>bitdepth</i>	Bit depth to run codec at

6.16 LED

LED driver devices.

Classes

- struct `color`

Enumerations

- enum {
`LED_COLOR_RED`, `LED_COLOR_GREEN`, `LED_COLOR_BLUE`, `LED_COLOR_WHITE`,
`LED_COLOR_PURPLE`, `LED_COLOR_CYAN`, `LED_COLOR_GOLD`, `LED_COLOR_OFF`,
`LED_COLOR_LAST` }

Functions

- void `dsy_led_driver_init` (`dsy_i2c_handle` *dsy_i2c, uint8_t *addr, uint8_t addr_cnt)
- void `dsy_led_driver_update` ()
- void `dsy_led_driver_set_led` (uint8_t idx, float bright)
- `color` * `dsy_led_driver_color_by_name` (uint8_t name)

6.16.1 Detailed Description

LED driver devices.

Device driver for PCA9685 16-channel 12-bit PWM generator.

6.16.2 Enumeration Type Documentation

6.16.2.1 anonymous enum

anonymous enum

Different Led colors

Enumerator

<code>LED_COLOR_RED</code>	<code>&</code>
<code>LED_COLOR_GREEN</code>	<code>&</code>
<code>LED_COLOR_BLUE</code>	<code>&</code>
<code>LED_COLOR_WHITE</code>	<code>&</code>
<code>LED_COLOR_PURPLE</code>	<code>&</code>
<code>LED_COLOR_CYAN</code>	<code>&</code>
<code>LED_COLOR_GOLD</code>	<code>&</code>
<code>LED_COLOR_OFF</code>	<code>&</code>
<code>LED_COLOR_LAST</code>	<code>&</code>

6.16.3 Function Documentation

6.16.3.1 `dsy_led_driver_color_by_name()`

```
color* dsy_led_driver_color_by_name (
    uint8_t name )
```

Passing in one of the preset colors will return a pointer to a color struct

Parameters

<i>name</i>	Preset color
-------------	--------------

6.16.3.2 dsy_led_driver_init()

```
void dsy_led_driver_init (
    dsy_i2c_handle * dsy_i2c,
    uint8_t * addr,
    uint8_t addr_cnt )
```

Initializes the LED Driver(s) on the specified I2C Bus

Parameters

<i>*dsy_i2c</i>	should be any dsy_i2c_handle with pins and speed configured.
<i>addr</i>	is either a pointer to 1 device address, or an array of addresses for multiple devices
<i>addr_cnt</i>	is the number of addresses passed in (use '1' for a single device)

6.16.3.3 dsy_led_driver_set_led()

```
void dsy_led_driver_set_led (
    uint8_t idx,
    float bright )
```

sets the LED at the index to the specified brightness (0-1) Index is sequential so device 0 will have idx 0-15, while device 1 will have idx 16-31, etc.

Parameters

<i>idx</i>	Index
<i>bright</i>	Brightness

6.16.3.4 dsy_led_driver_update()

```
void dsy_led_driver_update ( )
```

Updates the LED Driver with the values set using the set function

Currently only updates one driver at a time due to the time it takes to update all of the devices. This can likely be set up to use DMA so that the function doesn't block for so long.

6.17 SDRAM

SDRAM devices.

Classes

- struct [dsy_sdram_handle](#)

Macros

- #define [DSY_SDRAM_DATA](#) __attribute__((section(".sram_data")))
- #define [DSY_SDRAM_BSS](#) __attribute__((section(".sram_bss")))

Enumerations

- enum { [DSY_SDRAM_OK](#), [DSY_SDRAM_ERR](#) }
- enum [dsy_sdram_state](#) { [DSY_SDRAM_STATE_ENABLE](#), [DSY_SDRAM_STATE_DISABLE](#), [DSY_SDRAM_STATE_LAST](#) }
- enum [dsy_sdram_pin](#) { [DSY_SDRAM_PIN_SDNWE](#), [DSY_SDRAM_PIN_LAST](#) }

Functions

- uint8_t [dsy_sdram_init](#) ([dsy_sdram_handle](#) *dsy_hsdram)

6.17.1 Detailed Description

SDRAM devices.

6.17.2 Macro Definition Documentation

6.17.2.1 DSY_SDRAM_BSS

```
#define DSY_SDRAM_BSS __attribute__((section(".sram_bss")))
```

Variables placed here will not be initialized.

Usage

E.g. int DSY_SDRAM_BSS uninitialized_var;

6.17.2.2 DSY_SDRAM_DATA

```
#define DSY_SDRAM_DATA __attribute__((section(".sram_data")))
```

Usage:

E.g. int DSY_SDRAM_DATA initialized_var = 1;

6.17.3 Enumeration Type Documentation

6.17.3.1 anonymous enum

anonymous enum

Enumerator

DSY_SDRAM_OK	&
DSY_SDRAM_ERR	&

6.17.3.2 dsy_sdram_pin

enum `dsy_sdram_pin`

This is PH5 on Daisy

Enumerator

DSY_SDRAM_PIN_SDNWE	&
DSY_SDRAM_PIN_LAST	&

6.17.3.3 dsy_sdram_state

enum `dsy_sdram_state`

Determines whether chip is initialized, and activated.

Enumerator

DSY_SDRAM_STATE_ENABLE	&
DSY_SDRAM_STATE_DISABLE	&
DSY_SDRAM_STATE_LAST	&

6.17.4 Function Documentation

6.17.4.1 dsy_sdram_init()

```
uint8_t dsy_sdram_init (
    dsy_sdram_handle * dsy_hsdram )
```

Initializes the SDRAM peripheral

6.18 BOARDS

Daisy devices. Pod, seed, etc.

Classes

- struct [daisy::daisy_field](#)
- class [daisy::DaisyPatch](#)
*Class that handles initializing all of the hardware specific to the Daisy Patch Board.
 Helper funtions are also in place to provide easy access to built-in controls and peripherals.*
- class [daisy::DaisyPetal](#)
Helpers and hardware definitions for daisy petal.
- class [daisy::DaisyPod](#)
*Class that handles initializing all of the hardware specific to the Daisy Patch Board.
 Helper funtions are also in place to provide easy access to built-in controls and peripherals.*
- class [daisy::DaisySeed](#)
*This is the higher-level interface for the Daisy board.
 All basic peripheral configuration/initialization is setup here.*

Enumerations

- enum { [daisy::SW_2](#), [daisy::SW_1](#), [daisy::SW_3](#), [daisy::SW_LAST](#) }
- enum {
[daisy::KNOB_1](#), [daisy::KNOB_3](#), [daisy::KNOB_5](#), [daisy::KNOB_2](#),
[daisy::KNOB_4](#), [daisy::KNOB_6](#), [daisy::KNOB_7](#), [daisy::KNOB_8](#),
[daisy::KNOB_LAST](#) }
- enum {
[CV_1](#), [daisy::CV_2](#), [daisy::CV_3](#), [daisy::CV_4](#),
[daisy::CV_LAST](#) }
- enum {
[daisy::LED_KEY_A8](#), [daisy::LED_KEY_A7](#), [daisy::LED_KEY_A6](#), [daisy::LED_KEY_A5](#),
[daisy::LED_KEY_A4](#), [daisy::LED_KEY_A3](#), [daisy::LED_KEY_A2](#), [daisy::LED_KEY_A1](#),
[daisy::LED_KEY_B1](#), [daisy::LED_KEY_B2](#), [daisy::LED_KEY_B3](#), [daisy::LED_KEY_B4](#),
[daisy::LED_KEY_B5](#), [daisy::LED_KEY_B6](#), [daisy::LED_KEY_B7](#), [daisy::LED_KEY_B8](#),
[daisy::LED_KNOB_1](#), [daisy::LED_KNOB_2](#), [daisy::LED_KNOB_3](#), [daisy::LED_KNOB_4](#),
[daisy::LED_KNOB_5](#), [daisy::LED_KNOB_6](#), [daisy::LED_KNOB_7](#), [daisy::LED_KNOB_8](#),
[daisy::LED_SW_1](#), [daisy::LED_SW_2](#), [daisy::LED_LAST](#) }

Functions

- FORCE_INLINE float [s162f](#) (int16_t x)
- FORCE_INLINE int16_t [f2s16](#) (float x)
- FORCE_INLINE float [s242f](#) (int32_t x)
- FORCE_INLINE int32_t [f2s24](#) (float x)
- FORCE_INLINE void [daisy::daisy_field_init](#) (daisy_field *p)

6.18.1 Detailed Description

Daisy devices. Pod, seed, etc.

6.18.2 Enumeration Type Documentation

6.18.2.1 anonymous enum

anonymous enum
 enums for controls, etc.

Enumerator

SW_2	tactile switch
SW_1	tactile switch
SW_3	toggle
SW_LAST	&

6.18.2.2 anonymous enum

`anonymous enum`

All knobs connect to ADC1_INP10 via CD4051 mux

Enumerator

KNOB_1	&
KNOB_3	&
KNOB_5	&
KNOB_2	&
KNOB_4	&
KNOB_6	&
KNOB_7	&
KNOB_8	&
KNOB_LAST	&

6.18.2.3 anonymous enum

`anonymous enum`

Enumerator

CV_2	Connected to ADC1_INP17
CV_3	Connected to ADC1_INP15
CV_4	Connected to ADC1_INP4
CV_LAST	Connected to ADC1_INP11 &

6.18.2.4 anonymous enum

`anonymous enum`

Enumerator

LED_KEY_A8	&
LED_KEY_A7	&
LED_KEY_A6	&
LED_KEY_A5	&
LED_KEY_A4	&
LED_KEY_A3	&
LED_KEY_A2	&
LED_KEY_A1	&

Enumerator

LED_KEY_B1	&
LED_KEY_B2	&
LED_KEY_B3	&
LED_KEY_B4	&
LED_KEY_B5	&
LED_KEY_B6	&
LED_KEY_B7	&
LED_KEY_B8	&
LED_KNOB↔ _1	&
LED_KNOB↔ _2	&
LED_KNOB↔ _3	&
LED_KNOB↔ _4	&
LED_KNOB↔ _5	&
LED_KNOB↔ _6	&
LED_KNOB↔ _7	&
LED_KNOB↔ _8	&
LED_SW_1	&
LED_SW_2	&
LED_LAST	&

6.18.3 Function Documentation

6.18.3.1 daisy_field_init()

```
FORCE_INLINE void daisy::daisy_field_init (
    daisy_field * p )
```

Initializes daisy field

Parameters

<i>p</i>	<code>daisy_field</code> struct to initialize
----------	---

```
< &
< &
< &
< &
< &
< &
< &
< &
< &
< &
```

< &
 < &
 < &
 < &
 < &
 < &
 < &
 < &

6.18.3.2 f2s16()

```
FORCE_INLINE int16_t f2s16 (
    float x )
& < close to 1.0f-LSB at 16 bit
< - (1 - LSB)
< close to 1.0f-LSB at 16 bit
< - (1 - LSB)
< close to 1.0f-LSB at 16 bit
< close to 1.0f-LSB at 16 bit
< (2 ** 15) - 1
```

6.18.3.3 f2s24()

```
FORCE_INLINE int32_t f2s24 (
    float x )
& < close to 1.0f-LSB at 16 bit
< - (1 - LSB)
< close to 1.0f-LSB at 16 bit
< - (1 - LSB)
< close to 1.0f-LSB at 16 bit
< close to 1.0f-LSB at 16 bit
< 2 ** 23
```

6.18.3.4 s162f()

```
FORCE_INLINE float s162f (
    int16_t x )
Scales float by 1/(2 ^ 15)
```

Parameters

x	Number to be scaled.
-----	----------------------

Returns

Scaled number.

< 1 / (2** 15)

6.18.3.5 s242f()

```
FORCE_INLINE float s242f (
    int32_t x )
# < 2 ** 23
< 2 ** 23
< 1 / (2 ** 23)
```

6.19 UTILITY

General utilities. Ringbuffers, LED colors, OLED stuff, etc.

Classes

- struct [dsy_gpio_pin](#)
- struct [DSY_SD_CardInfoTypeDef](#)
- class [daisy::Color](#)
- struct [FontDef](#)
- class [daisy::RingBuffer< T, size >](#)
- class [daisy::RingBuffer< T, 0 >](#)
- struct [WAV_FormatTypeDef](#)

Macros

- `#define DMA_BUFFER_MEM_SECTION __attribute__((section(".sram1_bss")))`
- `#define DTCM_MEM_SECTION __attribute__((section(".dcmram_bss")))`
- `#define BSP_SD_CardInfo DSY_SD_CardInfoTypeDef`
- `#define MSD_OK ((uint8_t)0x00)`
- `#define MSD_ERROR ((uint8_t)0x01)`
- `#define MSD_ERROR_SD_NOT_PRESENT ((uint8_t)0x02)`
- `#define SD_TRANSFER_OK ((uint8_t)0x00)`
- `#define SD_TRANSFER_BUSY ((uint8_t)0x01)`
- `#define SD_PRESENT ((uint8_t)0x01)`
- `#define SD_NOT_PRESENT ((uint8_t)0x00)`
- `#define SD_DATATIMEOUT ((uint32_t)100000000)`

Enumerations

- enum [dsy_gpio_port](#) {
[DSY_GPIOA](#), [DSY_GPIOB](#), [DSY_GPIOC](#), [DSY_GPIOD](#),
[DSY_GPIOE](#), [DSY_GPIOF](#), [DSY_GPIOG](#), [DSY_GPIOH](#),
[DSY_GPIOI](#), [DSY_GPIOJ](#), [DSY_GPIOK](#), [DSY_GPIOX](#),
[DSY_GPIO_LAST](#) }

Functions

- `FORCE_INLINE float cube (float x)`
- `FORCE_INLINE dsy_gpio_pin dsy_pin (dsy_gpio_port port, uint8_t pin)`
- `FORCE_INLINE uint8_t dsy_pin_cmp (dsy_gpio_pin *a, dsy_gpio_pin *b)`
- `uint8_t BSP_SD_Init (void)`
- `uint8_t BSP_SD_ITConfig (void)`
- `uint8_t BSP_SD_ReadBlocks (uint32_t *pData, uint32_t ReadAddr, uint32_t NumOfBlocks, uint32_t Timeout)`
- `uint8_t BSP_SD_WriteBlocks (uint32_t *pData, uint32_t WriteAddr, uint32_t NumOfBlocks, uint32_t Timeout)`
- `uint8_t BSP_SD_ReadBlocks_DMA (uint32_t *pData, uint32_t ReadAddr, uint32_t NumOfBlocks)`
- `uint8_t BSP_SD_WriteBlocks_DMA (uint32_t *pData, uint32_t WriteAddr, uint32_t NumOfBlocks)`
- `uint8_t BSP_SD_Erase (uint32_t StartAddr, uint32_t EndAddr)`
- `uint8_t BSP_SD_GetCardState (void)`
- `void BSP_SD_GetCardInfo (DSY_SD_CardInfoTypeDef *CardInfo)`
- `uint8_t BSP_SD_IsDetected (void)`
- `void BSP_SD_AbortCallback (void)`
- `void BSP_SD_WriteCpltCallback (void)`
- `void BSP_SD_ReadCpltCallback (void)`
- `GPIO_TypeDef * dsy_hal_map_get_port (dsy_gpio_pin *p)`
- `uint16_t dsy_hal_map_get_pin (dsy_gpio_pin *p)`
- `I2C_HandleTypeDef * dsy_hal_map_get_i2c (dsy_i2c_handle *p)`
- `void dsy_get_unique_id (uint32_t *w0, uint32_t *w1, uint32_t *w2)`

Variables

- I2C_HandleTypeDef [hi2c1](#)
- I2C_HandleTypeDef [hi2c2](#)
- I2C_HandleTypeDef [hi2c3](#)
- I2C_HandleTypeDef [hi2c4](#)
- FontDef Font_6x8
- FontDef Font_7x10
- FontDef Font_11x18
- FontDef Font_16x26

6.19.1 Detailed Description

General utilities. Ringbuffers, LED colors, OLED stuff, etc.

6.19.2 Macro Definition Documentation

6.19.2.1 BSP_SD_CardInfo

```
#define BSP_SD_CardInfo DSY_SD_CardInfoTypeDef
&
```

6.19.2.2 DMA_BUFFER_MEM_SECTION

```
#define DMA_BUFFER_MEM_SECTION __attribute__((section(".sram1_bss")))
```

Macro for area of memory that is configured as cacheless This should be used primarily for DMA buffers, and the like.

6.19.2.3 DTCM_MEM_SECTION

```
#define DTCM_MEM_SECTION __attribute__((section(".dttcmram_bss")))
```

THE DTCM RAM section is also non-cached. However, is not suitable for DMA transfers. Performance is on par with internal SRAM w/ cache enabled.

6.19.2.4 MSD_ERROR

```
#define MSD_ERROR ((uint8_t)0x01)
&
```

6.19.2.5 MSD_ERROR_SD_NOT_PRESENT

```
#define MSD_ERROR_SD_NOT_PRESENT ((uint8_t)0x02)
&
```

6.19.2.6 MSD_OK

```
#define MSD_OK ((uint8_t)0x00)
&
```

6.19.2.7 SD_DATATIMEOUT

```
#define SD_DATATIMEOUT ((uint32_t)1000000000)
&
```

6.19.2.8 SD_NOT_PRESENT

```
#define SD_NOT_PRESENT ((uint8_t)0x00)
&
```

6.19.2.9 SD_PRESENT

```
#define SD_PRESENT ((uint8_t)0x01)
&
```

6.19.2.10 SD_TRANSFER_BUSY

```
#define SD_TRANSFER_BUSY ((uint8_t)0x01)
&
```

6.19.2.11 SD_TRANSFER_OK

```
#define SD_TRANSFER_OK ((uint8_t)0x00)
&
```

6.19.3 Enumeration Type Documentation**6.19.3.1 dsy_gpio_port**

```
enum dsy_gpio_port
```

Enums and a simple struct for defining a hardware pin on the MCU These correlate with the stm32 datasheet, and are used to configure the hardware.

Enumerator

DSY_GPIOA	&
DSY_GPIOB	&
DSY_GPIOC	&
DSY_GPIOD	&
DSY_GPIOE	&
DSY_GPIOF	&
DSY_GPIOG	&
DSY_GPIOH	&
DSY_GPIOI	&
DSY_GPIOJ	&
DSY_GPIOK	&
DSY_GPIO_LAST	This is a non-existent port for unsupported bits of hardware.

6.19.4 Function Documentation**6.19.4.1 BSP_SD_AbortCallback()**

```
void BSP_SD_AbortCallback (
    void )
```

These functions can be modified in case the current settings (e.g. DMA stream) need to be changed for specific application needs /n

Abort the callback

6.19.4.2 BSP_SD_Erase()

```
uint8_t BSP_SD_Erase (
    uint32_t StartAddr,
    uint32_t EndAddr )
```

Erase a section of memory

Parameters

<i>StartAddr</i>	Address to start erasing at
<i>EndAddr</i>	Address to stop erasing at

Returns

card state, ERROR, etc.

6.19.4.3 BSP_SD_GetCardInfo()

```
void BSP_SD_GetCardInfo (
    DSY_SD_CardInfoTypeDef * CardInfo )
```

Parameters

<i>*CardInfo</i>	Pointer to write card info to
------------------	-------------------------------

Parameters

<i>CardInfo</i>	&
-----------------	---

6.19.4.4 BSP_SD_GetCardState()

```
uint8_t BSP_SD_GetCardState (
    void )
```

Returns

card state, ERROR, etc.

6.19.4.5 BSP_SD_Init()

```
uint8_t BSP_SD_Init (
    void )
```

Returns

card state, ERROR, etc.

6.19.4.6 BSP_SD_IsDetected()

```
uint8_t BSP_SD_IsDetected (
    void )
```

Returns

Is card detected

6.19.4.7 BSP_SD_ITConfig()

```
uint8_t BSP_SD_ITConfig (
    void )
```

Returns

card state, ERROR, etc.

6.19.4.8 BSP_SD_ReadBlocks()

```
uint8_t BSP_SD_ReadBlocks (
    uint32_t * pData,
    uint32_t ReadAddr,
    uint32_t NumOfBlocks,
    uint32_t Timeout )
```

Parameters

<i>*pData</i>	&
<i>ReadAddr</i>	Address to read from
<i>NumOfBlocks</i>	Number of blocks to be read
<i>Timeout</i>	Timeout len in ms

Returns

OK ERROR, etc.

6.19.4.9 BSP_SD_ReadBlocks_DMA()

```
uint8_t BSP_SD_ReadBlocks_DMA (
    uint32_t * pData,
    uint32_t ReadAddr,
    uint32_t NumOfBlocks )
```

No timeout

Parameters

<i>*pData</i>	&
<i>ReadAddr</i>	Address to read from
<i>NumOfBlocks</i>	Number of blocks to be read

Returns

card state, ERROR, etc.

6.19.4.10 BSP_SD_ReadCpltCallback()

```
void BSP_SD_ReadCpltCallback (
    void )
```

Write complete callback

6.19.4.11 BSP_SD_WriteBlocks()

```
uint8_t BSP_SD_WriteBlocks (
```

```
uint32_t * pData,
uint32_t WriteAddr,
uint32_t NumOfBlocks,
uint32_t Timeout )
```

Parameters

<i>*pData</i>	&
<i>WriteAddr</i>	Address to write to
<i>NumOfBlocks</i>	Number of blocks to be written
<i>Timeout</i>	Timeout len in ms

Returns

card state, ERROR, etc.

6.19.4.12 BSP_SD_WriteBlocks_DMA()

```
uint8_t BSP_SD_WriteBlocks_DMA (
    uint32_t * pData,
    uint32_t WriteAddr,
    uint32_t NumOfBlocks )
```

No timeout

Parameters

<i>*pData</i>	&
<i>WriteAddr</i>	Address to write to
<i>NumOfBlocks</i>	Number of blocks to be read

Returns

card state, ERROR, etc.

6.19.4.13 BSP_SD_WriteCpltCallback()

```
void BSP_SD_WriteCpltCallback (
    void )
```

Read complete callback

6.19.4.14 cube()

```
FORCE_INLINE float cube (
    float x )
```

Computes cube.

Parameters

<i>x</i>	Number to be cubed
----------	--------------------

Returns

x^3

6.19.4.15 dsy_get_unique_id()

```
void dsy_get_unique_id (
    uint32_t * w0,
    uint32_t * w1,
    uint32_t * w2 )
```

Returns 96-bit Unique ID of the MCU

Author

shensley

Date

May 2020 fills the three pointer arguments with the unique ID of the MCU.

Parameters

*w0	First pointer
*w1	Second pointer
*w2	Third pointer

6.19.4.16 dsy_hal_map_get_i2c()

```
I2C_HandleTypeDef* dsy_hal_map_get_i2c (
    dsy_i2c_handle * p )
```

Parameters

*p	dsy_i2c_handle to get
----	-----------------------

Returns

The I2C_HandleTypeDef for the given *p

6.19.4.17 dsy_hal_map_get_pin()

```
uint16_t dsy_hal_map_get_pin (
    dsy_gpio_pin * p )
```

Parameters

*p	Pin pin to get
----	----------------

Returns

HAL GPIO Pin as used in the HAL from a dsy_gpio_pin input.

6.19.4.18 dsy_hal_map_get_port()

```
GPIO_TypeDef* dsy_hal_map_get_port (
    dsy_gpio_pin * p )
```

Parameters

<code>*p</code>	Pin pin to get
-----------------	----------------

Returns

HAL GPIO_TypeDef as used in the HAL from a [dsy_gpio_pin](#) input.

6.19.4.19 dsy_pin()

```
FORCE_INLINE dsy_gpio_pin dsy_pin (
    dsy_gpio_port port,
    uint8_t pin )
```

Helper for creating pins from port/pin combos easily

6.19.4.20 dsy_pin_cmp()

```
FORCE_INLINE uint8_t dsy_pin_cmp (
    dsy_gpio_pin * a,
    dsy_gpio_pin * b )
```

Helper for testing sameness of two dsy_gpio_pins

Returns

1 if same, 0 if different

6.19.5 Variable Documentation**6.19.5.1 Font_11x18**

```
FontDef Font_11x18
&
```

6.19.5.2 Font_16x26

```
FontDef Font_16x26
&
```

6.19.5.3 Font_6x8

```
FontDef Font_6x8
```

These are the different sizes of fonts (width x height in pixels per character)

6.19.5.4 Font_7x10

```
FontDef Font_7x10
&
```

6.19.5.5 hi2c1

```
I2C_HandleTypeDef hi2c1
```

global structs, and helper functions for interfacing with the stm32 HAL library while it remains a dependency. This file should only be included from source files (c/cpp) Including it from a header within libdaisy would expose the entire HAL to the users. This should be an option for users, but should not be required. externs of HAL handles...

6.19.5.6 hi2c2

I2C_HandleTypeDef hi2c2
externs of HAL handles...

6.19.5.7 hi2c3

I2C_HandleTypeDef hi2c3
externs of HAL handles...

6.19.5.8 hi2c4

I2C_HandleTypeDef hi2c4
externs of HAL handles...

6.20 USBDCDC_IF

Usb VCP device module.

Modules

- [USBDCDC_IF_Exported_Defines](#)
Defines.
- [USBDCDC_IF_Exported_Types](#)
Types.
- [USBDCDC_IF_Exported_Macros](#)
Aliases.
- [USBDCDC_IF_Exported_Variables](#)
Public variables.
- [USBDCDC_IF_Exported_FunctionsPrototype](#)
Public functions declaration.

6.20.1 Detailed Description

Usb VCP device module.

6.21 USBD_CDC_IF_Exported_Defines

Defines.

Defines.

6.22 USB_D_CDC_IF_Exported_Types

Types.

Typedefs

- typedef void(* [CDC_ReceiveCallback](#)) (uint8_t *buf, uint32_t *size)

6.22.1 Detailed Description

Types.

6.22.2 Typedef Documentation

6.22.2.1 CDC_ReceiveCallback

```
typedef void(* CDC_ReceiveCallback) (uint8_t *buf, uint32_t *size)
```

Parameters

<i>buf</i>	buffer
<i>size</i>	buffer size

6.23 USB_D_CDC_IF_Exported_Macros

Aliases.

Aliases.

6.24 USBD_CDC_IF_Exported_Variables

Public variables.

Variables

- USBD_CDC_ItfTypeDef [USBInterface_fops_FS](#)
- USBD_CDC_ItfTypeDef [USBInterface_fops_HS](#)

6.24.1 Detailed Description

Public variables.

6.24.2 Variable Documentation

6.24.2.1 USBInterface_fops_FS

USBDCDC_ItfTypeDef USBInterface_fops_FS
CDC Interface callback.

6.24.2.2 USBInterface_fops_HS

USBDCDC_ItfTypeDef USBInterface_fops_HS
CDC Interface callback.

6.25 USB_D_CDC_IF_Exported_FunctionsPrototype

Public functions declaration.

Functions

- void [CDC_Set_Rx_Callback_FS](#) ([CDC_ReceiveCallback](#) cb)
- [uint8_t CDC_Transmit_FS](#) ([uint8_t](#) *Buf, [uint16_t](#) Len)
- [uint8_t CDC_Transmit_HS](#) ([uint8_t](#) *Buf, [uint16_t](#) Len)

6.25.1 Detailed Description

Public functions declaration.

6.25.2 Function Documentation

6.25.2.1 CDC_Set_Rx_Callback_FS()

```
void CDC_Set_Rx_Callback_FS (  
    CDC\_ReceiveCallback cb )  
&
```

6.25.2.2 CDC_Transmit_FS()

```
uint8\_t CDC_Transmit_FS (  
    uint8\_t * Buf,  
    uint16\_t Len )  
&
```

6.25.2.3 CDC_Transmit_HS()

```
uint8\_t CDC_Transmit_HS (  
    uint8\_t * Buf,  
    uint16\_t Len )  
&
```

6.26 USBD_CONF

Configuration file for Usb otg low level driver.

Modules

- [USB_D_CONF_Exported_Variables](#)
Public variables.
- [USB_D_CONF_Exported_Defines](#)
Defines for configuration of the Usb device.
- [USB_D_CONF_Exported_Macros](#)
Aliases.
- [USB_D_CONF_Exported_Types](#)
Types.
- [USB_D_CONF_Exported_FunctionsPrototype](#)
Declaration of public functions for Usb device.

6.26.1 Detailed Description

Configuration file for Usb otg low level driver.

6.27 USBD_CONF_Exported_Variables

Public variables.

Public variables.

6.28 USBD_CONF_Exported_Defines

Defines for configuration of the Usb device.

Macros

- `#define USBD_MAX_NUM_INTERFACES 1U`
- `#define USBD_MAX_NUM_CONFIGURATION 1U`
- `#define USBD_MAX_STR_DESC_SIZ 512U`
- `#define USBD_SUPPORT_USER_STRING 0U`
- `#define USBD_DEBUG_LEVEL 3U`
- `#define USBD_LPM_ENABLED 0U`
- `#define USBD_SELF_POWERED 1U`
- `#define DEVICE_FS 0`
- `#define DEVICE_HS 1`

6.28.1 Detailed Description

Defines for configuration of the Usb device.

6.28.2 Macro Definition Documentation

6.28.2.1 DEVICE_FS

```
#define DEVICE_FS 0
```

FS and HS identification

6.28.2.2 DEVICE_HS

```
#define DEVICE_HS 1
```

&

6.28.2.3 USBD_DEBUG_LEVEL

```
#define USBD_DEBUG_LEVEL 3U
```

&

6.28.2.4 USBD_LPM_ENABLED

```
#define USBD_LPM_ENABLED 0U
```

&

6.28.2.5 USBD_MAX_NUM_CONFIGURATION

```
#define USBD_MAX_NUM_CONFIGURATION 1U
```

&

6.28.2.6 USBD_MAX_NUM_INTERFACES

```
#define USBD_MAX_NUM_INTERFACES 1U
```

&

6.28.2.7 USBD_MAX_STR_DESC_SIZ

```
#define USBD_MAX_STR_DESC_SIZ 512U
```

&

6.28.2.8 USBD_SELF_POWERED

```
#define USBD_SELF_POWERED 1U  
&
```

6.28.2.9 USBD_SUPPORT_USER_STRING

```
#define USBD_SUPPORT_USER_STRING 0U  
&
```

6.29 USBD_CONF_Exported_Macros

Aliases.

Macros

- #define `USBBD_malloc` `malloc`
- #define `USBBD_free` `free`
- #define `USBBD_memset` `memset`
- #define `USBBD_memcpy` `memcpy`
- #define `USBBD_Delay` `HAL_Delay`
- #define `USBBD_UsrLog(...)`
- #define `USBBD_ErrLog(...)`
- #define `USBBD_DbgLog(...)`

6.29.1 Detailed Description

Aliases.

6.29.2 Macro Definition Documentation

6.29.2.1 USBBD_DbgLog

```
#define USBBD_DbgLog(  
    ... )
```

Value:

```
printf("DEBUG : "); \  
printf(__VA_ARGS__); \  
printf("\n");
```

&

6.29.2.2 USBBD_Delay

```
#define USBBD_Delay HAL_Delay
```

Alias for delay.

6.29.2.3 USBBD_ErrLog

```
#define USBBD_ErrLog(  
    ... )
```

Value:

```
printf("ERROR: "); \  
printf(__VA_ARGS__); \  
printf("\n");
```

&

6.29.2.4 USBBD_free

```
#define USBBD_free free
```

Alias for memory release.

6.29.2.5 USBBD_malloc

```
#define USBBD_malloc malloc
```

Alias for memory allocation.

6.29.2.6 USBD_memcpy

```
#define USBD_memcpy memcpy
```

Alias for memory copy.

6.29.2.7 USBD_memset

```
#define USBD_memset memset
```

Alias for memory set.

6.29.2.8 USBD_UsrLog

```
#define USBD_UsrLog(  
    ... )
```

Value:

```
    printf(__VA_ARGS__); \  
    printf("\n");  
&
```

6.30 USBD_CONF_Exported_Types

Types.

Types.

6.31 USBD_CONF_Exported_FunctionsPrototype

Declaration of public functions for Usb device.

Declaration of public functions for Usb device.

6.32 USBD_DESC

Usb device descriptors module.

Modules

- [USB_D_DESC_Exported_Constants](#)
Constants.
- [USB_D_DESC_Exported_Defines](#)
Defines.
- [USB_D_DESC_Exported_TypesDefinitions](#)
Types.
- [USB_D_DESC_Exported_Macros](#)
Aliases.
- [USB_D_DESC_Exported_Variables](#)
Public variables.
- [USB_D_DESC_Exported_FunctionsPrototype](#)
Public functions declaration.

6.32.1 Detailed Description

Usb device descriptors module.

6.33 USBD_DESC_Exported_Constants

Constants.

Macros

- `#define DEVICE_ID1 (UID_BASE)`
- `#define DEVICE_ID2 (UID_BASE + 0x4)`
- `#define DEVICE_ID3 (UID_BASE + 0x8)`
- `#define USB_SIZ_STRING_SERIAL 0x1A`

6.33.1 Detailed Description

Constants.

6.33.2 Macro Definition Documentation

6.33.2.1 DEVICE_ID1

```
#define DEVICE_ID1 (UID_BASE)
&
```

6.33.2.2 DEVICE_ID2

```
#define DEVICE_ID2 (UID_BASE + 0x4)
&
```

6.33.2.3 DEVICE_ID3

```
#define DEVICE_ID3 (UID_BASE + 0x8)
&
```

6.33.2.4 USB_SIZ_STRING_SERIAL

```
#define USB_SIZ_STRING_SERIAL 0x1A
&
```

6.34 USBD_DESC_Exported_Defines

Defines.

Defines.

6.35 USBD_DESC_Exported_TypesDefinitions

Types.

Types.

6.36 USBD_DESC_Exported_Macros

Aliases.

Aliases.

6.37 USBD_DESC_Exported_Variables

Public variables.

Variables

- USBD_DescriptorsTypeDef [HS_Desc](#)
- USBD_DescriptorsTypeDef [FS_Desc](#)

6.37.1 Detailed Description

Public variables.

6.37.2 Variable Documentation

6.37.2.1 FS_Desc

USB_DescriptorsTypeDef FS_Desc
Descriptor for the Usb device.

6.37.2.2 HS_Desc

USB_DescriptorsTypeDef HS_Desc
Descriptor for the Usb device.

6.38 USBD_DESC_Exported_FunctionsPrototype

Public functions declaration.

Public functions declaration.

6.39 Externals

6.40 STM32_USB_OTG_DEVICE_LIBRARY

For Usb device.

Modules

- [USBD_CDC_IF](#)
Usb VCP device module.
- [USBD_DESC](#)
Usb device descriptors module.

6.40.1 Detailed Description

For Usb device.

< Define to prevent recursive inclusion -----

6.41 USBD_OTG_DRIVER

Modules

- [USB_CONF](#)

Configuration file for Usb otg low level driver.

6.41.1 Detailed Description

Chapter 7

Namespace Documentation

7.1 daisy Namespace Reference

Hardware defines and helpers for daisy field platform.

Classes

- struct [AdcChannelConfig](#)
- class [AdcHandle](#)
- class [AnalogControl](#)
 - Hardware Interface for control inputs*
 - Primarily designed for ADC input controls such as potentiometers, and control voltage.*
- class [Color](#)
- struct [ControlChangeEvent](#)
- struct [daisy_field](#)
- class [DaisyPatch](#)
 - Class that handles initializing all of the hardware specific to the Daisy Patch Board.*
 - Helper funtions are also in place to provide easy access to built-in controls and peripherals.*
- class [DaisyPetal](#)
 - Helpers and hardware definitions for daisy petal.*
- class [DaisyPod](#)
 - Class that handles initializing all of the hardware specific to the Daisy Patch Board.*
 - Helper funtions are also in place to provide easy access to built-in controls and peripherals.*
- class [DaisySeed](#)
 - This is the higher-level interface for the Daisy board.*
 - All basic peripheral configuration/initialization is setup here.*
- class [Encoder](#)
 - Generic Class for handling Quadrature Encoders*
 - Inspired/influenced by Mutable Instruments (pichenettes) [Encoder](#) classes.*
- class [GateIn](#)
 - Generic Class for handling gate inputs through GPIO.*
- class [Led](#)
 - LED Class providing simple Software PWM ability, etc*
 - Eventually this will work with hardware PWM, and external LED Driver devices as well.*
- struct [MidiEvent](#)
- class [MidiHandler](#)
 - Simple MIDI Handler*
 - Parses bytes from an input into valid MidiEvents.*
 - The MidiEvents fill a FIFO queue that the user can pop messages from.*
- struct [NoteOnEvent](#)

- class [OledDisplay](#)
- class [Parameter](#)
- class [RgbLed](#)
- class [RingBuffer](#)
- class [RingBuffer< T, 0 >](#)
- class [SdmmcHandler](#)
- struct [SdmmcHandlerInit](#)
- class [SpiHandle](#)
- class [Switch](#)
- class [UartHandler](#)
- struct [WavFileInfo](#)
- class [WavPlayer](#)

Enumerations

- enum { [SW_2](#), [SW_1](#), [SW_3](#), [SW_LAST](#) }
- enum { [KNOB_1](#), [KNOB_3](#), [KNOB_5](#), [KNOB_2](#), [KNOB_4](#), [KNOB_6](#), [KNOB_7](#), [KNOB_8](#), [KNOB_LAST](#) }
- enum { [CV_1](#), [CV_2](#), [CV_3](#), [CV_4](#), [CV_LAST](#) }
- enum { [LED_KEY_A8](#), [LED_KEY_A7](#), [LED_KEY_A6](#), [LED_KEY_A5](#), [LED_KEY_A4](#), [LED_KEY_A3](#), [LED_KEY_A2](#), [LED_KEY_A1](#), [LED_KEY_B1](#), [LED_KEY_B2](#), [LED_KEY_B3](#), [LED_KEY_B4](#), [LED_KEY_B5](#), [LED_KEY_B6](#), [LED_KEY_B7](#), [LED_KEY_B8](#), [LED_KNOB_1](#), [LED_KNOB_2](#), [LED_KNOB_3](#), [LED_KNOB_4](#), [LED_KNOB_5](#), [LED_KNOB_6](#), [LED_KNOB_7](#), [LED_KNOB_8](#), [LED_SW_1](#), [LED_SW_2](#), [LED_LAST](#) }
- enum [MidiMessageType](#) { [NoteOff](#), [NoteOn](#), [PolyphonicKeyPressure](#), [ControlChange](#), [ProgramChange](#), [ChannelPressure](#), [PitchBend](#), [MessageLast](#) }
- enum [SdmmcMode](#) { [SDMMC_MODE_FATFS](#) }
- enum [SdmmcBitWidth](#) { [SDMMC_BITS_1](#), [SDMMC_BITS_4](#) }
- enum [SdmmcSpeed](#) { [SDMMC_SPEED_400KHZ](#), [SDMMC_SPEED_12MHZ](#) }
- enum [SpiPeriph](#) { [SPI_PERIPH_1](#), [SPI_PERIPH_3](#), [SPI_PERIPH_6](#) }
- enum [SpiPin](#) { [SPI_PIN_CS](#), [SPI_PIN_SCK](#), [SPI_PIN_MOSI](#), [SPI_PIN_MISO](#) }

Functions

- `FORCE_INLINE void daisy_field_init (daisy_field *p)`

Variables

- `const size_t kUartMaxBufferSize = 32`

7.1.1 Detailed Description

Hardware defines and helpers for daisy field platform.

Chapter 8

Class Documentation

8.1 daisy::AdcChannelConfig Struct Reference

```
#include <per_adc.h>
```

Public Types

- enum [MuxPin](#) { [MUX_SEL_0](#), [MUX_SEL_1](#), [MUX_SEL_2](#), [MUX_SEL_LAST](#) }

Public Member Functions

- void [InitSingle](#) ([dsy_gpio_pin](#) pin)
- void [InitMux](#) ([dsy_gpio_pin](#) adc_pin, [dsy_gpio_pin](#) mux_0, [dsy_gpio_pin](#) mux_1, [dsy_gpio_pin](#) mux_2, size_t channels)

Public Attributes

- [dsy_gpio_pin_](#)
- [dsy_gpio_mux_pin_](#) [[MUX_SEL_LAST](#)]
- uint8_t [mux_channels_](#)

8.1.1 Detailed Description

Configuration Structure for a given channel

8.1.2 Member Enumeration Documentation

8.1.2.1 MuxPin

```
enum daisy::AdcChannelConfig::MuxPin
```

Which pin to use for multiplexing

Enumerator

MUX_SEL_0	&
MUX_SEL_1	&
MUX_SEL_2	&
MUX_SEL_LAST	&

8.1.3 Member Function Documentation

8.1.3.1 InitMux()

```
void daisy::AdcChannelConfig::InitMux (
    dsy_gpio_pin adc_pin,
    dsy_gpio_pin mux_0,
    dsy_gpio_pin mux_1,
    dsy_gpio_pin mux_2,
    size_t channels )
```

Initializes a single ADC pin as a Multiplexed ADC. Requires a CD4051 Multiplexor connected to the pin Internal Callbacks handle the pin addressing.

Parameters

<i>channels</i>	must be 1-8
<i>mux_0</i>	First mux pin
<i>mux_1</i>	Second mux pin
<i>mux_2</i>	Third mux pin
<i>adc_pin</i>	&

8.1.3.2 InitSingle()

```
void daisy::AdcChannelConfig::InitSingle (
    dsy_gpio_pin pin )
```

Initializes a single ADC pin as an ADC.

Parameters

<i>pin</i>	Pin to init.
------------	--------------

8.1.4 Member Data Documentation

8.1.4.1 mux_channels_

```
uint8_t daisy::AdcChannelConfig::mux_channels_
&
```

8.1.4.2 mux_pin_

```
dsy_gpio daisy::AdcChannelConfig::mux_pin_[MUX_SEL_LAST]
&
```

8.1.4.3 pin_

```
dsy_gpio daisy::AdcChannelConfig::pin_
&
```

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

- src/per_adc.h

8.2 daisy::AdcHandle Class Reference

```
#include <per_adc.h>
```

Public Types

- enum [OverSampling](#) {
[OVS_NONE](#), [OVS_4](#), [OVS_8](#), [OVS_16](#),
[OVS_32](#), [OVS_64](#), [OVS_128](#), [OVS_256](#),
[OVS_512](#), [OVS_1024](#), [OVS_LAST](#) }

Public Member Functions

- void [Init](#) ([AdcChannelConfig](#) *cfg, size_t num_channels, [OverSampling](#) ovs=[OVS_32](#))
- void [Start](#) ()
- void [Stop](#) ()
- uint16_t [Get](#) (uint8_t chn)
- uint16_t * [GetPtr](#) (uint8_t chn)
- float [GetFloat](#) (uint8_t chn)
- uint16_t [GetMux](#) (uint8_t chn, uint8_t idx)
- uint16_t * [GetMuxPtr](#) (uint8_t chn, uint8_t idx)
- float [GetMuxFloat](#) (uint8_t chn, uint8_t idx)

8.2.1 Detailed Description

Handler for analog to digital conversion

8.2.2 Member Enumeration Documentation

8.2.2.1 OverSampling

```
enum daisy::AdcHandle::OverSampling
```

Supported oversampling amounts

Enumerator

OVS_NONE	&
OVS_4	&
OVS_8	&
OVS_16	&
OVS_32	&
OVS_64	&
OVS_128	&
OVS_256	&
OVS_512	&
OVS_1024	&
OVS_LAST	&

8.2.3 Member Function Documentation

8.2.3.1 Get()

```
uint16_t daisy::AdcHandle::Get (
    uint8_t chn )
```

Single channel getter

Parameters

<i>chn</i>	channel to get
------------	----------------

Returns

Converted value

8.2.3.2 GetFloat()

```
float daisy::AdcHandle::GetFloat (
    uint8_t chn )
```

Get floating point from single channel

Parameters

<i>chn</i>	Channel to get from
------------	---------------------

Returns

Floating point converted value

8.2.3.3 GetMux()

```
uint16_t daisy::AdcHandle::GetMux (
    uint8_t chn,
    uint8_t idx )
```

Getters for multiplexed inputs on a single channel (up to 8 per ADC input).

Parameters

<i>chn</i>	Channel to get from
<i>idx</i>	&

Returns

data

8.2.3.4 GetMuxFloat()

```
float daisy::AdcHandle::GetMuxFloat (
    uint8_t chn,
    uint8_t idx )
```

Getters for multiplexed inputs on a single channel (up to 8 per ADC input).

Parameters

<i>chn</i>	Channel to get from
<i>idx</i>	&

Returns

Floating point data

8.2.3.5 GetMuxPtr()

```
uint16_t* daisy::AdcHandle::GetMuxPtr (
    uint8_t chn,
    uint8_t idx )
```

Getters for multiplexed inputs on a single channel. (Max 8 per chan)

Parameters

<i>chn</i>	Channel to get from
<i>idx</i>	&

Returns

Pointer to data

8.2.3.6 GetPtr()

```
uint16_t* daisy::AdcHandle::GetPtr (
    uint8_t chn )
```

Get pointer to a value from a single channel

Parameters

<i>chn</i>	
------------	--

Returns

Pointer to converted value

8.2.3.7 Init()

```
void daisy::AdcHandle::Init (
    AdcChannelConfig * cfg,
    size_t num_channels,
    OverSampling ovs = OVS\_32 )
```

Initializes the ADC with the pins passed in.

Parameters

<i>*cfg</i>	an array of AdcChannelConfig of the desired channel
<i>num_channels</i>	number of ADC channels to initialize
<i>ovs</i>	Oversampling amount - Defaults to OVS_32

8.2.3.8 Start()

```
void daisy::AdcHandle::Start ( )
```

Starts reading from the ADC

8.2.3.9 Stop()

```
void daisy::AdcHandle::Stop ( )
```

Stops reading from the ADC

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

- src/per_adc.h

8.3 daisy::AnalogControl Class Reference

Hardware Interface for control inputs

Primarily designed for ADC input controls such as potentiometers, and control voltage.

```
.
#include <hid_ctrl.h>
```

Public Member Functions

- [AnalogControl](#) ()
- [~AnalogControl](#) ()
- void [Init](#) (uint16_t *adcptr, float sr, bool flip=false, bool invert=false, float slew_seconds=0.002f)
- void [InitBipolarCv](#) (uint16_t *adcptr, float sr)
- float [Process](#) ()
- float [Value](#) () const

8.3.1 Detailed Description

Hardware Interface for control inputs

Primarily designed for ADC input controls such as potentiometers, and control voltage.

.

Author

Stephen Hensley

Date

November 2019

8.3.2 Constructor & Destructor Documentation

8.3.2.1 AnalogControl()

```
daisy::AnalogControl::AnalogControl ( ) [inline]
```

Constructor

8.3.2.2 ~AnalogControl()

```
daisy::AnalogControl::~~AnalogControl ( ) [inline]
```

destructor

8.3.3 Member Function Documentation

8.3.3.1 Init()

```
void daisy::AnalogControl::Init (
    uint16_t * adcptr,
    float sr,
    bool flip = false,
    bool invert = false,
    float slew_seconds = 0.002f )
```

Initializes the control

Parameters

<i>*adcptr</i>	is a pointer to the raw adc read value – This can be acquired with <code>dsy_adc_get_rawptr()</code> , or <code>dsy_adc_get_mux_rawptr()</code>
<i>sr</i>	is the samplerate in Hz that the Process function will be called at.
<i>flip</i>	determines whether the input is flipped (i.e. 1.f - input) or not before being processed.1
<i>invert</i>	determines whether the input is inverted (i.e. -1.f * input) or not before being processed.
<i>slew_seconds</i>	is the slew time in seconds that it takes for the control to change to a new value.

8.3.3.2 InitBipolarCv()

```
void daisy::AnalogControl::InitBipolarCv (
    uint16_t * adcptr,
    float sr )
```

This Initializes the [AnalogControl](#) for a -5V to 5V inverted input All of the Init details are the same otherwise

Parameters

<i>*adcptr</i>	Pointer to analog digital converter
<i>sr</i>	Audio engine sample rate

8.3.3.3 Process()

```
float daisy::AnalogControl::Process ( )
```

Filters, and transforms a raw ADC read into a normalized range. this should be called at the rate of specified by samplerate at Init time.

Default Initializations will return 0.0 -> 1.0 Bi-polar CV inputs will return -1.0 -> 1.0

8.3.3.4 Value()

```
float daisy::AnalogControl::Value ( ) const [inline]
```

Returns the current stored value, without reprocessing

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

- `src/hid_ctrl.h`

8.4 codec_frame_t Struct Reference

```
#include <dev_codec_wm8731_frame.h>
```

Public Attributes

- short [l](#)
- short [r](#)

8.4.1 Detailed Description

&

8.4.2 Member Data Documentation

8.4.2.1 [l](#)

```
short codec_frame_t::l
&
```

8.4.2.2 [r](#)

```
short codec_frame_t::r
&
```

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

- [src/dev_codec_wm8731_frame.h](#)

8.5 color Struct Reference

```
#include <dev_leddriver.h>
```

Public Attributes

- uint16_t [red](#)
- uint16_t [green](#)
- uint16_t [blue](#)

8.5.1 Detailed Description

Simple color struct Different from util_color only in type (0-4095 vs 0-1)
This could easily be migrated to work with those instead.

8.5.2 Member Data Documentation

8.5.2.1 [blue](#)

```
uint16_t color::blue
&
```

8.5.2.2 [green](#)

```
uint16_t color::green
&
```

8.5.2.3 red

```
uint16_t color::red
&
```

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

- src/dev_leddriver.h

8.6 daisy::Color Class Reference

```
#include <util_color.h>
```

Public Types

- enum [PresetColor](#) {
 [RED](#), [GREEN](#), [BLUE](#), [WHITE](#),
 [PURPLE](#), [CYAN](#), [GOLD](#), [OFF](#),
 [LAST](#) }

Public Member Functions

- void [Init](#) ([PresetColor](#) c)
- void [Init](#) (float red, float green, float blue)
- float [Red](#) () const
- float [Green](#) () const
- float [Blue](#) () const

8.6.1 Detailed Description

Class for handling simple colors

8.6.2 Member Enumeration Documentation

8.6.2.1 PresetColor

```
enum daisy::Color::PresetColor
```

List of colors that have a preset RGB value

Enumerator

RED	&
GREEN	&
BLUE	&
WHITE	&
PURPLE	&
CYAN	&
GOLD	&
OFF	&
LAST	&

8.6.3 Member Function Documentation

8.6.3.1 Blue()

```
float daisy::Color::Blue ( ) const [inline]
```

Returns the 0-1 value for Blue

8.6.3.2 Green()

```
float daisy::Color::Green ( ) const [inline]
```

Returns the 0-1 value for Green

8.6.3.3 Init() [1/2]

```
void daisy::Color::Init (
    float red,
    float green,
    float blue )
```

Initializes the [Color](#) with a specific RGB value red, green, and blue should be floats between 0 and 1

Parameters

<i>red</i>	Red value
<i>green</i>	Green value
<i>blue</i>	Blue value

8.6.3.4 Init() [2/2]

```
void daisy::Color::Init (
    PresetColor c )
```

Initializes the [Color](#) with a given preset.

Parameters

<i>c</i>	Color to init to
----------	----------------------------------

8.6.3.5 Red()

```
float daisy::Color::Red ( ) const [inline]
```

Returns the 0-1 value for Red

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

- `src/util_color.h`

8.7 daisy::ControlChangeEvent Struct Reference

```
#include <hid_midi.h>
```

Public Attributes

- int [channel](#)
- uint8_t [control_number](#)
- uint8_t [value](#)

8.7.1 Detailed Description

Struct containing control number, and value for a given channel. Can be made from [MidiEvent](#)

8.7.2 Member Data Documentation

8.7.2.1 channel

```
int daisy::ControlChangeEvent::channel
&
```

8.7.2.2 control_number

```
uint8_t daisy::ControlChangeEvent::control_number
&
```

8.7.2.3 value

```
uint8_t daisy::ControlChangeEvent::value
&
```

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

- src/hid_midi.h

8.8 daisy::daisy_field Struct Reference

```
#include <daisy_field.h>
```

Public Attributes

- [daisy::DaisySeed seed](#)
- [daisy::Switch switches \[SW_LAST\]](#)
- [dsy_gpio gate_in](#)
- [dsy_gpio gate_out](#)
- [dsy_sr_4021_handle keyboard_sr](#)
- [AnalogControl knobs \[KNOB_LAST\]](#)
- [AnalogControl cvs \[CV_LAST\]](#)

8.8.1 Detailed Description

Struct containing hardware defines and daisy seed

8.8.2 Member Data Documentation

8.8.2.1 cvs

```
AnalogControl daisy::daisy_field::cvs[CV_LAST]
Array of cv ins
```

8.8.2.2 gate_in

```
dsy_gpio daisy::daisy_field::gate_in
Gate input.
```

8.8.2.3 gate_out

```
dsy_gpio daisy::daisy_field::gate_out
Gate output
```

8.8.2.4 keyboard_sr

`dsy_sr_4021_handle` `daisy::daisy_field::keyboard_sr`
Keyboard shift register

8.8.2.5 knobs

`AnalogControl` `daisy::daisy_field::knobs[KNOB_LAST]`
Array of hardware knobs

8.8.2.6 seed

`daisy::DaisySeed` `daisy::daisy_field::seed`
Daisy seed

8.8.2.7 switches

`daisy::Switch` `daisy::daisy_field::switches[SW_LAST]`
Array of hardware switches
The documentation for this struct was generated from the following file:

- `src/daisy_field.h`

8.9 daisy::DaisyPatch Class Reference

Class that handles initializing all of the hardware specific to the Daisy Patch Board.
Helper funtions are also in place to provide easy access to built-in controls and peripherals.
`#include <daisy_patch.h>`

Public Types

- enum `Ctrl` {
 `CTRL_1`, `CTRL_2`, `CTRL_3`, `CTRL_4`,
 `CTRL_LAST` }
- enum `GateInput` { `GATE_IN_1`, `GATE_IN_2`, `GATE_IN_LAST` }

Public Member Functions

- `DaisyPatch` ()
- `~DaisyPatch` ()
- void `Init` ()
- void `DelayMs` (size_t del)
- void `SetAudioBlockSize` (size_t size)
- void `StartAudio` (dsy_audio_mc_callback cb)
- void `ChangeAudioCallback` (dsy_audio_callback cb)
- void `StartAdc` ()
- float `AudioSampleRate` ()
- size_t `AudioBlockSize` ()
- float `AudioCallbackRate` ()
- void `UpdateAnalogControls` ()
- float `GetCtrlValue` (Ctrl k)
- void `DebounceControls` ()
- void `DisplayControls` (bool invert=true)

Public Attributes

- [DaisySeed](#) seed
- [Encoder](#) encoder
- [AnalogControl](#) controls [CTRL_LAST]
- [GateIn](#) gate_input [GATE_IN_LAST]
- [MidiHandler](#) midi
- [OledDisplay](#) display
- [dsy_gpio](#) gate_output

8.9.1 Detailed Description

Class that handles initializing all of the hardware specific to the Daisy Patch Board. Helper funtions are also in place to provide easy access to built-in controls and peripherals.

Author

Stephen Hensley

Date

November 2019

8.9.2 Member Enumeration Documentation

8.9.2.1 Ctrl

```
enum daisy::DaisyPatch::Ctrl
```

Enum of Ctrl's to represent the four CV/Knob combos on the Patch

8.9.2.2 GateInput

```
enum daisy::DaisyPatch::GateInput
```

Daisy patch gate inputs

Enumerator

GATE_IN_LAST	<
--------------	---

8.9.3 Constructor & Destructor Documentation

8.9.3.1 DaisyPatch()

```
daisy::DaisyPatch::DaisyPatch ( ) [inline]
```

Constructor

8.9.3.2 ~DaisyPatch()

```
daisy::DaisyPatch::~~DaisyPatch ( ) [inline]
```

Destructor

8.9.4 Member Function Documentation

8.9.4.1 AudioBlockSize()

```
size_t daisy::DaisyPatch::AudioBlockSize ( )
```

Get block size

8.9.4.2 AudioCallbackRate()

```
float daisy::DaisyPatch::AudioCallbackRate ( )
```

Get callback rate

8.9.4.3 AudioSampleRate()

```
float daisy::DaisyPatch::AudioSampleRate ( )
```

Get sample rate

8.9.4.4 ChangeAudioCallback()

```
void daisy::DaisyPatch::ChangeAudioCallback (
    dsy_audio_callback cb )
```

Change to a different callback function.

Parameters

<i>cb</i>	New callback function.
-----------	------------------------

8.9.4.5 DebounceControls()

```
void daisy::DaisyPatch::DebounceControls ( )
```

Debounce analog controls. Call at same rate as reading controls.

8.9.4.6 DelayMs()

```
void daisy::DaisyPatch::DelayMs (
    size_t del )
```

Wait some ms before going on.

Parameters

<i>del</i>	Delay time in ms.
------------	-------------------

8.9.4.7 DisplayControls()

```
void daisy::DaisyPatch::DisplayControls (
    bool invert = true )
```

Control the display

8.9.4.8 GetCtrlValue()

```
float daisy::DaisyPatch::GetCtrlValue (
    Ctrl k )
```

Get value for a particular control

Parameters

<i>k</i>	Which control to get
----------	----------------------

8.9.4.9 Init()

```
void daisy::DaisyPatch::Init ( )
```

Initializes the daisy seed, and patch hardware.

8.9.4.10 SetAudioBlockSize()

```
void daisy::DaisyPatch::SetAudioBlockSize (
    size_t size )
```

Audio Block size defaults to 48. Change it using this function before StartingAudio

Parameters

<i>size</i>	Audio block size.
-------------	-------------------

8.9.4.11 StartAdc()

```
void daisy::DaisyPatch::StartAdc ( )
```

Start analog to digital conversion.

8.9.4.12 StartAudio()

```
void daisy::DaisyPatch::StartAudio (
    dsy_audio_mc_callback cb )
```

Start audio output.

Parameters

<i>cb</i>	Audio callback function
-----------	-------------------------

8.9.4.13 UpdateAnalogControls()

```
void daisy::DaisyPatch::UpdateAnalogControls ( )
```

Call at same rate as reading controls for good reads.

8.9.5 Member Data Documentation

8.9.5.1 controls

```
AnalogControl daisy::DaisyPatch::controls[CTRL_LAST]
```

Array of controls

8.9.5.2 display

```
OledDisplay daisy::DaisyPatch::display
&
```

8.9.5.3 encoder

```
Encoder daisy::DaisyPatch::encoder
```

Encoder object

8.9.5.4 gate_input

`GateIn` daisy::DaisyPatch::gate_input [GATE_IN_LAST]
Gate inputs

8.9.5.5 gate_output

`dsy_gpio` daisy::DaisyPatch::gate_output
&

8.9.5.6 midi

`MidiHandler` daisy::DaisyPatch::midi
Handles midi

8.9.5.7 seed

`DaisySeed` daisy::DaisyPatch::seed
Seed object

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

- src/daisy_patch.h

8.10 daisy::DaisyPetal Class Reference

Helpers and hardware definitions for daisy petal.

```
#include <daisy_petal.h>
```

Public Types

- enum `Sw` {
 `SW_1`, `SW_2`, `SW_3`, `SW_4`,
 `SW_5`, `SW_6`, `SW_7`, `SW_LAST` }
- enum `Knob` {
 `KNOB_1`, `KNOB_2`, `KNOB_3`, `KNOB_4`,
 `KNOB_5`, `KNOB_6`, `KNOB_LAST` }
- enum `RingLed` {
 `RING_LED_1`, `RING_LED_2`, `RING_LED_3`, `RING_LED_4`,
 `RING_LED_5`, `RING_LED_6`, `RING_LED_7`, `RING_LED_8`,
 `RING_LED_LAST` }
- enum `FootswitchLed` {
 `FOOTSWITCH_LED_1`, `FOOTSWITCH_LED_2`, `FOOTSWITCH_LED_3`, `FOOTSWITCH_LED_4`,
 `FOOTSWITCH_LED_LAST` }

Public Member Functions

- `DaisyPetal` ()
- `~DaisyPetal` ()
- void `Init` ()
- void `DelayMs` (size_t del)
- void `SetAudioBlockSize` (size_t size)
- void `StartAudio` (dsy_audio_callback cb)
- void `ChangeAudioCallback` (dsy_audio_callback cb)
- void `StartAdc` ()
- float `AudioSampleRate` ()
- size_t `AudioBlockSize` ()

- float [AudioCallbackRate](#) ()
- void [UpdateAnalogControls](#) ()
- float [GetKnobValue](#) ([Knob](#) k)
- float [GetExpression](#) ()
- void [DebounceControls](#) ()
- void [ClearLeds](#) ()
- void [UpdateLeds](#) ()
- void [SetRingLed](#) ([RingLed](#) idx, float r, float g, float b)
- void [SetFootswitchLed](#) ([FootswitchLed](#) idx, float bright)

Public Attributes

- [DaisySeed](#) seed
- [Encoder](#) encoder
- [AnalogControl](#) knob [[KNOB_LAST](#)]
- [AnalogControl](#) expression
- [Switch](#) switches [[SW_LAST](#)]
- [RgbLed](#) ring_led [8]
- [Led](#) footswitch_led [4]

8.10.1 Detailed Description

Helpers and hardware definitions for daisy petal.

8.10.2 Member Enumeration Documentation

8.10.2.1 FootswitchLed

enum [daisy::DaisyPetal::FootswitchLed](#)
footswitch leds

Enumerator

FOOTSWITCH_LED_1	&
FOOTSWITCH_LED_2	&
FOOTSWITCH_LED_3	&
FOOTSWITCH_LED_4	&
FOOTSWITCH_LED_LAST	&

8.10.2.2 Knob

enum [daisy::DaisyPetal::Knob](#)
Knobs

Enumerator

KNOB_1	&
KNOB_2	&
KNOB_3	&
KNOB_4	&
KNOB_5	&
KNOB_6	&
KNOB_LAST	&

8.10.2.3 RingLed

enum `daisy::DaisyPetal::RingLed`

Leds in ringled

Enumerator

RING_LED_1	&
RING_LED_2	&
RING_LED_3	&
RING_LED_4	&
RING_LED_5	&
RING_LED_6	&
RING_LED_7	&
RING_LED_8	&
RING_LED_LAST	&

8.10.2.4 Sw

enum `daisy::DaisyPetal::Sw`

Switches

Enumerator

SW_1	Footswitch
SW_2	Footswitch
SW_3	Footswitch
SW_4	Footswitch
SW_5	Toggle
SW_6	Toggle
SW_7	Toggle
SW_LAST	Last enum item

8.10.3 Constructor & Destructor Documentation

8.10.3.1 DaisyPetal()

`daisy::DaisyPetal::DaisyPetal () [inline]`

Constructor

8.10.3.2 ~DaisyPetal()

`daisy::DaisyPetal::~~DaisyPetal () [inline]`

Destructor

8.10.4 Member Function Documentation

8.10.4.1 AudioBlockSize()

```
size_t daisy::DaisyPetal::AudioBlockSize ( )
```

Get audio block size

8.10.4.2 AudioCallbackRate()

```
float daisy::DaisyPetal::AudioCallbackRate ( )
```

Get callback rate

8.10.4.3 AudioSampleRate()

```
float daisy::DaisyPetal::AudioSampleRate ( )
```

Device audio sample rate.

8.10.4.4 ChangeAudioCallback()

```
void daisy::DaisyPetal::ChangeAudioCallback (
    dsy_audio_callback cb )
```

Change callback function

Parameters

<i>cb</i>	New callback function.
-----------	------------------------

8.10.4.5 ClearLeds()

```
void daisy::DaisyPetal::ClearLeds ( )
```

Turn all leds off

8.10.4.6 DebounceControls()

```
void daisy::DaisyPetal::DebounceControls ( )
```

Debounce inputs.

8.10.4.7 DelayMs()

```
void daisy::DaisyPetal::DelayMs (
    size_t del )
```

Wait before moving on.

Parameters

<i>del</i>	Delay time in ms.
------------	-------------------

8.10.4.8 GetExpression()

```
float daisy::DaisyPetal::GetExpression ( )
```

&

8.10.4.9 GetKnobValue()

```
float daisy::DaisyPetal::GetKnobValue (
    Knob k )
```

Get value per knob.

Parameters

<i>k</i>	Which knob to get
----------	-------------------

Returns

Floating point knob position.

8.10.4.10 Init()

```
void daisy::DaisyPetal::Init ( )
```

Initialize daisy petal

8.10.4.11 SetAudioBlockSize()

```
void daisy::DaisyPetal::SetAudioBlockSize (
    size_t size )
```

Set size of audio blocks.

Parameters

<i>size</i>	Audio block size
-------------	------------------

8.10.4.12 SetFootswitchLed()

```
void daisy::DaisyPetal::SetFootswitchLed (
    FootswitchLed idx,
    float bright )
```

Set footswitch LED

Parameters

<i>idx</i>	Led Index
<i>bright</i>	Brightness

8.10.4.13 SetRingLed()

```
void daisy::DaisyPetal::SetRingLed (
    RingLed idx,
    float r,
    float g,
    float b )
```

Set ring LED colors

Parameters

<i>idx</i>	Index to set
<i>r</i>	Red value
<i>g</i>	Green value
<i>b</i>	Blue value

8.10.4.14 StartAdc()

```
void daisy::DaisyPetal::StartAdc ( )
```

Start analog to digital conversion.

8.10.4.15 StartAudio()

```
void daisy::DaisyPetal::StartAudio (
    dsy_audio_callback cb )
```

Start audio callback

Parameters

<i>cb</i>	Callback function.
-----------	--------------------

8.10.4.16 UpdateAnalogControls()

```
void daisy::DaisyPetal::UpdateAnalogControls ( )
```

Call at the same frequency as controls are read for stable readings.

8.10.4.17 UpdateLeds()

```
void daisy::DaisyPetal::UpdateLeds ( )
```

Update Leds to values you had set.

8.10.5 Member Data Documentation

8.10.5.1 encoder

```
Encoder daisy::DaisyPetal::encoder
```

&

8.10.5.2 expression

```
AnalogControl daisy::DaisyPetal::expression
```

&

8.10.5.3 footswitch_led

```
Led daisy::DaisyPetal::footswitch_led[4]
```

&

8.10.5.4 knob

```
AnalogControl daisy::DaisyPetal::knob[KNOB_LAST]
```

&

8.10.5.5 ring_led

```
RgbLed daisy::DaisyPetal::ring_led[8]
```

&

8.10.5.6 seed

```
DaisySeed daisy::DaisyPetal::seed
```

&

8.10.5.7 switches

```
Switch daisy::DaisyPetal::switches[SW_LAST]
< &
```

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

- src/daisy_petal.h

8.11 daisy::DaisyPod Class Reference

Class that handles initializing all of the hardware specific to the Daisy Patch Board.

Helper funtions are also in place to provide easy access to built-in controls and peripherals.

```
#include <daisy_pod.h>
```

Public Types

- enum [Sw](#) { [BUTTON_1](#), [BUTTON_2](#), [BUTTON_LAST](#) }
- enum [Knob](#) { [KNOB_1](#), [KNOB_2](#), [KNOB_LAST](#) }

Public Member Functions

- void [Init](#) ()
- void [DelayMs](#) (size_t del)
- void [SetAudioBlockSize](#) (size_t size)
- void [StartAudio](#) (dsy_audio_callback cb)
- void [ChangeAudioCallback](#) (dsy_audio_callback cb)
- void [StartAdc](#) ()
- float [AudioSampleRate](#) ()
- size_t [AudioBlockSize](#) ()
- float [AudioCallbackRate](#) ()
- void [UpdateAnalogControls](#) ()
- float [GetKnobValue](#) ([Knob](#) k)
- void [DebounceControls](#) ()
- void [ClearLeds](#) ()
- void [UpdateLeds](#) ()

Public Attributes

- [DaisySeed](#) seed
- [Encoder](#) encoder
- [AnalogControl](#) knob1
- [AnalogControl](#) knob2
- [AnalogControl](#) * knobs [[KNOB_LAST](#)]
- [Switch](#) button1
- [Switch](#) button2
- [Switch](#) * buttons [[BUTTON_LAST](#)]
- [RgbLed](#) led1
- [RgbLed](#) led2

8.11.1 Detailed Description

Class that handles initializing all of the hardware specific to the Daisy Patch Board.

Helper funtions are also in place to provide easy access to built-in controls and peripherals.

Author

Stephen Hensley

Date

November 2019

8.11.2 Member Enumeration Documentation

8.11.2.1 Knob

```
enum daisy::DaisyPod::Knob
Knobs
```

Enumerator

KNOB_2	&
KNOB_LAST	&

8.11.2.2 Sw

```
enum daisy::DaisyPod::Sw
Switches
```

Enumerator

BUTTON_2	&
BUTTON_LAST	&

8.11.3 Member Function Documentation

8.11.3.1 AudioBlockSize()

```
size_t daisy::DaisyPod::AudioBlockSize ( )
Get block size
```

8.11.3.2 AudioCallbackRate()

```
float daisy::DaisyPod::AudioCallbackRate ( )
Get callback rate
```

8.11.3.3 AudioSampleRate()

```
float daisy::DaisyPod::AudioSampleRate ( )
Get sample rate
```

8.11.3.4 ChangeAudioCallback()

```
void daisy::DaisyPod::ChangeAudioCallback (
    dsy_audio_callback cb )
Switch callback functions
```

Parameters

<i>cb</i>	New callback function.
-----------	------------------------

8.11.3.5 ClearLeds()

```
void daisy::DaisyPod::ClearLeds ( )
```

Reset Leds

8.11.3.6 DebounceControls()

```
void daisy::DaisyPod::DebounceControls ( )
```

&

8.11.3.7 DelayMs()

```
void daisy::DaisyPod::DelayMs (
    size_t del )
```

Wait for a bit

Parameters

<i>del</i>	Time to wait in ms.
------------	---------------------

8.11.3.8 GetKnobValue()

```
float daisy::DaisyPod::GetKnobValue (
    Knob k )
```

&

8.11.3.9 Init()

```
void daisy::DaisyPod::Init ( )
```

Init related stuff.

8.11.3.10 SetAudioBlockSize()

```
void daisy::DaisyPod::SetAudioBlockSize (
    size_t size )
```

Audio Block size defaults to 48. Change it using this function before StartingAudio.

Parameters

<i>size</i>	Block size to set.
-------------	--------------------

8.11.3.11 StartAdc()

```
void daisy::DaisyPod::StartAdc ( )
```

Start analog to digital conversion.

8.11.3.12 StartAudio()

```
void daisy::DaisyPod::StartAudio (
    dsy_audio_callback cb )
```

Start audio callback

Parameters

<i>cb</i>	Callback function.
-----------	--------------------

8.11.3.13 UpdateAnalogControls()

`void daisy::DaisyPod::UpdateAnalogControls ()`
Call at same rate as analog reads for smooth reading.

8.11.3.14 UpdateLeds()

`void daisy::DaisyPod::UpdateLeds ()`
Update Leds to set colors

8.11.4 Member Data Documentation**8.11.4.1 button1**

`Switch daisy::DaisyPod::button1`
&

8.11.4.2 button2

`Switch daisy::DaisyPod::button2`
&

8.11.4.3 buttons

`Switch * daisy::DaisyPod::buttons[BUTTON_LAST]`
&

8.11.4.4 encoder

`Encoder daisy::DaisyPod::encoder`
&

8.11.4.5 knob1

`AnalogControl daisy::DaisyPod::knob1`
&

8.11.4.6 knob2

`AnalogControl daisy::DaisyPod::knob2`
&

8.11.4.7 knobs

`AnalogControl * daisy::DaisyPod::knobs[KNOB_LAST]`
&

8.11.4.8 led1

`RgbLed daisy::DaisyPod::led1`
&

8.11.4.9 led2

```
RgbLed daisy::DaisyPod::led2
&
```

8.11.4.10 seed

```
DaisySeed daisy::DaisyPod::seed
```

Public Members

8.11.5 autotoc_md8

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

- src/daisy_pod.h

8.12 daisy::DaisySeed Class Reference

This is the higher-level interface for the Daisy board.
All basic peripheral configuration/initialization is setup here.
`#include <daisy_seed.h>`

Public Member Functions

- void [Configure](#) ()
- void [Init](#) ()
- [dsy_gpio_pin GetPin](#) (uint8_t pin_idx)
- void [StartAudio](#) ([dsy_audio_callback](#) cb)
- void [SetLed](#) (bool state)
- void [SetTestPoint](#) (bool state)
- float [AudioSampleRate](#) ()
- void [SetAudioBlockSize](#) (size_t blocksize)

Public Attributes

- [dsy_sdr_handle sdr_handle](#)
- [dsy_qspi_handle qspi_handle](#)
- [dsy_audio_handle audio_handle](#)
- [dsy_sai_handle sai_handle](#)
- [dsy_i2c_handle i2c1_handle](#)
- [dsy_i2c_handle i2c2_handle](#)
- [AdcHandle adc](#)
- [dsy_dac_handle dac_handle](#)
- [UsbHandle usb_handle](#)

8.12.1 Detailed Description

This is the higher-level interface for the Daisy board.
All basic peripheral configuration/initialization is setup here.

8.12.2 Member Function Documentation

8.12.2.1 AudioSampleRate()

```
float daisy::DaisySeed::AudioSampleRate ( )
```

Returns the audio sample rate in Hz as a floating point number.

8.12.2.2 Configure()

```
void daisy::DaisySeed::Configure ( )
```

Configures the settings for all internal peripherals, but does not initialize them. This allows for modification of the configuration handles prior to initialization. &

8.12.2.3 GetPin()

```
dsy_gpio_pin daisy::DaisySeed::GetPin (
    uint8_t pin_idx )
```

Returns the gpio_pin corresponding to the index 0-31. For the given GPIO on the Daisy Seed (labeled 1-32 in docs).

8.12.2.4 Init()

```
void daisy::DaisySeed::Init ( )
```

Initializes the Daisy Seed and the following peripherals: SDRAM, QSPI, 24-bit 48kHz Audio via AK4556, Internal USB, as well as the built-in LED and Testpoint.

ADCs, DACs, and other special peripherals (such as I2C, SPI, etc.) can be initialized using their specific initializers within libdaisy for a specific application.

8.12.2.5 SetAudioBlockSize()

```
void daisy::DaisySeed::SetAudioBlockSize (
    size_t blocksize )
```

Sets the number of samples processed per channel by the audio callback.

8.12.2.6 SetLed()

```
void daisy::DaisySeed::SetLed (
    bool state )
```

Sets the state of the built in LED

8.12.2.7 SetTestPoint()

```
void daisy::DaisySeed::SetTestPoint (
    bool state )
```

Sets the state of the test point near pin 10

8.12.2.8 StartAudio()

```
void daisy::DaisySeed::StartAudio (
    dsy_audio_callback cb )
```

Begins the audio for the seeds builtin audio. the specified callback will get called whenever new data is ready to be prepared.

8.12.3 Member Data Documentation

8.12.3.1 adc

```
AdcHandle daisy::DaisySeed::adc
&
```

8.12.3.2 audio_handle

```
dsy_audio_handle daisy::DaisySeed::audio_handle
&
```

8.12.3.3 dac_handle

```
dsy_dac_handle daisy::DaisySeed::dac_handle  
&
```

8.12.3.4 i2c1_handle

```
dsy_i2c_handle daisy::DaisySeed::i2c1_handle  
&
```

8.12.3.5 i2c2_handle

```
dsy_i2c_handle daisy::DaisySeed::i2c2_handle  
&
```

8.12.3.6 qspi_handle

```
dsy_qspi_handle daisy::DaisySeed::qspi_handle  
&
```

8.12.3.7 sai_handle

```
dsy_sai_handle daisy::DaisySeed::sai_handle  
&
```

8.12.3.8 sdram_handle

```
dsy_sdram_handle daisy::DaisySeed::sdram_handle  
&
```

8.12.3.9 usb_handle

```
UsbHandle daisy::DaisySeed::usb_handle  
&
```

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

- src/daisy_seed.h

8.13 dsy_audio_handle Struct Reference

```
#include <hid_audio.h>
```

Public Attributes

- `size_t` [block_size](#)
- `dsy_sai_handle` * [sai](#)
- `dsy_i2c_handle` * [dev0_i2c](#)
- `dsy_i2c_handle` * [dev1_i2c](#)

8.13.1 Detailed Description

Simple config struct that holds peripheral drivers.

8.13.2 Member Data Documentation

8.13.2.1 block_size

```
size_t dsy_audio_handle::block_size  
&
```

8.13.2.2 dev0_i2c

```
dsy_i2c_handle* dsy_audio_handle::dev0_i2c  
&
```

8.13.2.3 dev1_i2c

```
dsy_i2c_handle* dsy_audio_handle::dev1_i2c  
&
```

8.13.2.4 sai

```
dsy_sai_handle* dsy_audio_handle::sai  
&
```

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

- src/hid_audio.h

8.14 dsy_dac_handle Struct Reference

```
#include <per_dac.h>
```

Public Attributes

- [dsy_dac_mode](#) mode
- [dsy_dac_bitdepth](#) bitdepth
- [dsy_gpio_pin](#) pin_config [DSY_DAC_CHN_LAST]

8.14.1 Detailed Description

Configuration structure for DAC initialization and settings. pin_config must be filled out. However, the DACs are pretty consistently on pins PA4, and PA5 across all STM32 MCUs that I've used.

8.14.2 Member Data Documentation

8.14.2.1 bitdepth

```
dsy_dac_bitdepth dsy_dac_handle::bitdepth  
&
```

8.14.2.2 mode

```
dsy_dac_mode dsy_dac_handle::mode  
&
```

8.14.2.3 pin_config

```
dsy_gpio_pin dsy_dac_handle::pin_config[DSY_DAC_CHN_LAST]  
&
```

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

- src/per_dac.h

8.15 dsy_gpio Struct Reference

```
#include <per_gpio.h>
```

Public Attributes

- [dsy_gpio_pin](#) pin
- [dsy_gpio_mode](#) mode
- [dsy_gpio_pull](#) pull

8.15.1 Detailed Description

Struct for holding the pin, and configuration

8.15.2 Member Data Documentation

8.15.2.1 mode

```
dsy_gpio_mode dsy_gpio::mode  
&
```

8.15.2.2 pin

```
dsy_gpio_pin dsy_gpio::pin  
&
```

8.15.2.3 pull

```
dsy_gpio_pull dsy_gpio::pull  
&
```

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

- src/per_gpio.h

8.16 dsy_gpio_pin Struct Reference

```
#include <daisy_core.h>
```

Public Attributes

- [dsy_gpio_port](#) port
- [uint8_t](#) pin

8.16.1 Detailed Description

Hardware define pins

8.16.2 Member Data Documentation

8.16.2.1 pin

```
uint8_t dsy_gpio_pin::pin  
number 0-15
```

8.16.2.2 port

```
dsy_gpio_port dsy_gpio_pin::port
&
```

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

- src/daisy_core.h

8.17 dsy_i2c_handle Struct Reference

```
#include <per_i2c.h>
```

Public Attributes

- [dsy_i2c_periph](#) periph
- [dsy_gpio_pin](#) pin_config [DSY_I2C_PIN_LAST]
- [dsy_i2c_speed](#) speed

8.17.1 Detailed Description

this object will be used to initialize the I2C interface, and can be passed to dev_ drivers that require I2C.

8.17.2 Member Data Documentation

8.17.2.1 periph

```
dsy_i2c_periph dsy_i2c_handle::periph
&
```

8.17.2.2 pin_config

```
dsy_gpio_pin dsy_i2c_handle::pin_config[DSY_I2C_PIN_LAST]
&
```

8.17.2.3 speed

```
dsy_i2c_speed dsy_i2c_handle::speed
&
```

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

- src/per_i2c.h

8.18 dsy_qspi_handle Struct Reference

```
#include <per_qspi.h>
```

Public Attributes

- [dsy_qspi_mode](#) mode
- [dsy_qspi_device](#) device
- [dsy_gpio_pin](#) pin_config [DSY_QSPI_PIN_LAST]

8.18.1 Detailed Description

Configuration structure for interfacing with QSPI Driver

8.18.2 Member Data Documentation

8.18.2.1 device

```
dsy_qspi_device dsy_qspi_handle::device
&
```

8.18.2.2 mode

```
dsy_qspi_mode dsy_qspi_handle::mode
&
```

8.18.2.3 pin_config

```
dsy_gpio_pin dsy_qspi_handle::pin_config[DSY_QSPI_PIN_LAST]
&
```

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

- src/per_qspi.h

8.19 dsy_sai_handle Struct Reference

```
#include <per_sai.h>
```

Public Attributes

- [dsy_audio_sai](#) init
- [dsy_audio_samplerate](#) samplerate [DSY_SAI_LAST]
- [dsy_audio_bitdepth](#) bitdepth [DSY_SAI_LAST]
- [dsy_audio_dir a_direction](#) [DSY_SAI_LAST]
- [dsy_audio_dir b_direction](#) [DSY_SAI_LAST]
- [dsy_audio_sync](#) sync_config [DSY_SAI_LAST]
- [dsy_audio_device](#) device [DSY_SAI_LAST]
- [dsy_gpio_pin sai1_pin_config](#) [DSY_SAI_PIN_LAST]
- [dsy_gpio_pin sai2_pin_config](#) [DSY_SAI_PIN_LAST]

8.19.1 Detailed Description

Configuration structure for SAI contains all above settings, and passes them to internal structure for hardware initialization.

8.19.2 Member Data Documentation

8.19.2.1 a_direction

```
dsy_audio_dir dsy_sai_handle::a_direction[DSY_SAI_LAST]
&
```

8.19.2.2 b_direction

```
dsy_audio_dir dsy_sai_handle::b_direction[DSY_SAI_LAST]
&
```

8.19.2.3 bitdepth

```
dsy_audio_bitdepth dsy_sai_handle::bitdepth[DSY_SAI_LAST]
&
```

8.19.2.4 device

```
dsy_audio_device dsy_sai_handle::device[DSY_SAI_LAST]
&
```

8.19.2.5 init

```
dsy_audio_sai dsy_sai_handle::init
&
```

8.19.2.6 sai1_pin_config

```
dsy_gpio_pin dsy_sai_handle::sai1_pin_config[DSY_SAI_PIN_LAST]
&
```

8.19.2.7 sai2_pin_config

```
dsy_gpio_pin dsy_sai_handle::sai2_pin_config[DSY_SAI_PIN_LAST]
&
```

8.19.2.8 samplerate

```
dsy_audio_samplerate dsy_sai_handle::samplerate[DSY_SAI_LAST]
&
```

8.19.2.9 sync_config

```
dsy_audio_sync dsy_sai_handle::sync_config[DSY_SAI_LAST]
&
```

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

- src/per_sai.h

8.20 DSY_SD_CardInfoTypeDef Struct Reference

```
#include <util_bsp_sd_diskio.h>
```

Public Attributes

- uint32_t [CardType](#)
- uint32_t [CardVersion](#)
- uint32_t [Class](#)
- uint32_t [RelCardAdd](#)
- uint32_t [BlockNbr](#)
- uint32_t [BlockSize](#)
- uint32_t [LogBlockNbr](#)
- uint32_t [LogBlockSize](#)
- uint32_t [CardSpeed](#)

8.20.1 Detailed Description

Functions for handling DiskIO via SDMMC These are usually configured through the FatFS driver/interface, and won't need to be accessed directly often.

8.20.2 Member Data Documentation

8.20.2.1 BlockNbr

```
uint32_t DSY_SD_CardInfoTypeDef::BlockNbr
```

Specifies the Card Capacity in blocks

8.20.2.2 BlockSize

```
uint32_t DSY_SD_CardInfoTypeDef::BlockSize
```

Specifies one block size in bytes

8.20.2.3 CardSpeed

```
uint32_t DSY_SD_CardInfoTypeDef::CardSpeed
```

Specifies the card Speed

8.20.2.4 CardType

```
uint32_t DSY_SD_CardInfoTypeDef::CardType
```

Specifies the card Type

8.20.2.5 CardVersion

```
uint32_t DSY_SD_CardInfoTypeDef::CardVersion
```

Specifies the card version

8.20.2.6 Class

```
uint32_t DSY_SD_CardInfoTypeDef::Class
```

Specifies the class of the card class

8.20.2.7 LogBlockNbr

```
uint32_t DSY_SD_CardInfoTypeDef::LogBlockNbr
```

Specifies the Card logical Capacity in blocks

8.20.2.8 LogBlockSize

```
uint32_t DSY_SD_CardInfoTypeDef::LogBlockSize
```

Specifies logical block size in bytes

8.20.2.9 RelCardAdd

```
uint32_t DSY_SD_CardInfoTypeDef::RelCardAdd
```

Specifies the Relative Card Address

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

- src/util_bsp_sd_diskio.h

8.21 dsy_sdram_handle Struct Reference

```
#include <dev_sdram.h>
```


Public Attributes

- [dsy_sdram_state](#) state
- [dsy_gpio_pin pin_config](#) [DSY_SDRAM_PIN_LAST]

8.21.1 Detailed Description

Configuration struct for passing to initialization

8.21.2 Member Data Documentation

8.21.2.1 pin_config

```
dsy_gpio_pin dsy_sdram_handle::pin_config[DSY_SDRAM_PIN_LAST]
&
```

8.21.2.2 state

```
dsy_sdram_state dsy_sdram_handle::state
&
```

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

- src/dev_sdram.h

8.22 dsy_sr_4021_handle Struct Reference

```
#include <dev_sr_4021.h>
```

Public Attributes

- [dsy_gpio_pin pin_config](#) [DSY_SR_4021_PIN_LAST]
- [uint8_t num_parallel](#)
- [uint8_t num_daisy chained](#)
- [dsy_gpio cs](#)
- [dsy_gpio clk](#)
- [dsy_gpio data](#) [2]
- [uint8_t states](#) [8 * 1 * 2]

8.22.1 Detailed Description

configuration struct for 4021 pin config is used to initialize the [dsy_gpio](#) num_parallel is the number of devices connected that share the same clk/cs, etc. but have independent data num_daisy chained is the number of devices in a daisy-chain configuration

8.22.2 Member Data Documentation

8.22.2.1 clk

```
dsy_gpio dsy_sr_4021_handle::clk
clk pin
```

8.22.2.2 cs

```
dsy_gpio dsy_sr_4021_handle::cs
cs pin
```

8.22.2.3 data

`dsy_gpio` `dsy_sr_4021_handle::data[2]`
array of data pins

8.22.2.4 num_daisy chained

`uint8_t` `dsy_sr_4021_handle::num_daisy chained`
Number of devices daisy chained

8.22.2.5 num_parallel

`uint8_t` `dsy_sr_4021_handle::num_parallel`
number of devices connected

8.22.2.6 pin_config

`dsy_gpio_pin` `dsy_sr_4021_handle::pin_config[DSY_SR_4021_PIN_LAST]`
used to initialize the `dsy_gpio`

8.22.2.7 states

`uint8_t` `dsy_sr_4021_handle::states[8 * 1 * 2]`
array of states

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

- `src/dev_sr_4021.h`

8.23 daisy::Encoder Class Reference

Generic Class for handling Quadrature Encoders

Inspired/influenced by Mutable Instruments (pichenettes) [Encoder](#) classes.

```
#include <hid_encoder.h>
```

Public Member Functions

- void [Init](#) ([dsy_gpio_pin](#) a, [dsy_gpio_pin](#) b, [dsy_gpio_pin](#) click, float update_rate)
- void [Debounce](#) ()
- `int32_t` [Increment](#) () const
- bool [RisingEdge](#) () const
- bool [FallingEdge](#) () const
- bool [Pressed](#) () const
- float [TimeHeldMs](#) () const

8.23.1 Detailed Description

Generic Class for handling Quadrature Encoders

Inspired/influenced by Mutable Instruments (pichenettes) [Encoder](#) classes.

Author

Stephen Hensley

Date

December 2019

8.23.2 Member Function Documentation

8.23.2.1 Debounce()

```
void daisy::Encoder::Debounce ( )
```

Called at `update_rate` to debounce and handle timing for the switch. In order for events not to be missed, its important that the Edge/Pressed checks be made at the same rate as the debounce function is being called.

8.23.2.2 FallingEdge()

```
bool daisy::Encoder::FallingEdge ( ) const [inline]
```

Returns true if the encoder was just released.

8.23.2.3 Increment()

```
int32_t daisy::Encoder::Increment ( ) const [inline]
```

Returns +1 if the encoder was turned clockwise, -1 if it was turned counter-clockwise, or 0 if it was not just turned.

8.23.2.4 Init()

```
void daisy::Encoder::Init (
    dsy_gpio_pin a,
    dsy_gpio_pin b,
    dsy_gpio_pin click,
    float update_rate )
```

Initializes the encoder with the specified hardware pins. Update rate should be the rate at which [Debounce\(\)](#) gets called in Hertz.

8.23.2.5 Pressed()

```
bool daisy::Encoder::Pressed ( ) const [inline]
```

Returns true while the encoder is held down.

8.23.2.6 RisingEdge()

```
bool daisy::Encoder::RisingEdge ( ) const [inline]
```

Returns true if the encoder was just pressed.

8.23.2.7 TimeHeldMs()

```
float daisy::Encoder::TimeHeldMs ( ) const [inline]
```

Returns the time in milliseconds that the encoder has been held down.

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

- `src/hid_encoder.h`

8.24 FontDef Struct Reference

```
#include <util_oled_fonts.h>
```

Public Attributes

- `const uint8_t` [FontWidth](#)
- `uint8_t` [FontHeight](#)
- `const uint16_t *` [data](#)

8.24.1 Detailed Description

Utility for displaying fonts on OLED displays

Migrated to work with libdaisy from stm32-ssd1306

Author

afiskon on github. Font struct

8.24.2 Member Data Documentation**8.24.2.1 data**

```
const uint16_t* FontDef::data
```

Pointer to data font data array

8.24.2.2 FontHeight

```
uint8_t FontDef::FontHeight
```

Font height in pixels

8.24.2.3 FontWidth

```
const uint8_t FontDef::FontWidth
```

Font width in pixels

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

- `src/util_oled_fonts.h`

8.25 daisy::GateIn Class Reference

Generic Class for handling gate inputs through GPIO.
`#include <hid_gatein.h>`

Public Member Functions

- [GateIn](#) ()
- [~GateIn](#) ()
- void [Init](#) ([dsy_gpio_pin](#) *pin_cfg)
- bool [Trig](#) ()

8.25.1 Detailed Description

Generic Class for handling gate inputs through GPIO.

Author

Stephen Hensley

Date

March 2020

8.25.2 Constructor & Destructor Documentation**8.25.2.1 GateIn()**

```
daisy::GateIn::GateIn ( ) [inline]
```

[GateIn](#) Constructor

8.25.2.2 ~GateIn()

```
daisy::GateIn::~GateIn ( ) [inline]
GateIn~ Destructor
```

8.25.3 Member Function Documentation

8.25.3.1 Init()

```
void daisy::GateIn::Init (
    dsy_gpio_pin * pin_cfg )
Init Initializes the gate input with specified hardware pin
```

8.25.3.2 Trig()

```
bool daisy::GateIn::Trig ( )
Trig Checks current state of gate input.
```

Returns

FALSE if pin is low, and TRUE if high

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

- src/[hid_gatein.h](#)

8.26 daisy::Led Class Reference

LED Class providing simple Software PWM ability, etc
Eventually this will work with hardware PWM, and external LED Driver devices as well.
`#include <hid_led.h>`

Public Member Functions

- void [Init](#) ([dsy_gpio_pin](#) pin, bool invert, float samplerate=1000.0f)
- void [Set](#) (float val)
- void [Update](#) ()

8.26.1 Detailed Description

LED Class providing simple Software PWM ability, etc
Eventually this will work with hardware PWM, and external LED Driver devices as well.

Author

shensley

Date

March 2020

8.26.2 Member Function Documentation

8.26.2.1 Init()

```
void daisy::Led::Init (
    dsy_gpio_pin pin,
    bool invert,
    float samplerate = 1000.0f )
```

Initializes an LED using the specified hardware pin.

Parameters

<i>pin</i>	chooses LED pin
<i>invert</i>	will set whether to internally invert the brightness due to hardware config.
<i>samplerate</i>	sets the rate at which 'Update()' will be called (used for software PWM)

8.26.2.2 Set()

```
void daisy::Led::Set (
    float val )
```

Sets the brightness of the [Led](#).

Parameters

<i>val</i>	will be cubed for gamma correction, and then quantized to 8-bit values for Software PWM 8-bit is for more flexible update rate options, as 12-bit or more would require faster update rates.
------------	--

8.26.2.3 Update()

```
void daisy::Led::Update ( )
```

This processes the pwm of the LED sets the hardware accordingly.

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

- `src/hid_led.h`

8.27 daisy::MidiEvent Struct Reference

```
#include <hid_midi.h>
```

Public Member Functions

- [NoteOnEvent AsNoteOn](#) ()
- [ControlChangeEvent AsControlChange](#) ()

Public Attributes

- [MidiMessageType](#) type
- int [channel](#)
- uint8_t [data](#) [2]

8.27.1 Detailed Description

Simple [MidiEvent](#) with message type, channel, and data[2] members.

8.27.2 Member Function Documentation

8.27.2.1 AsControlChange()

```
ControlChangeEvent daisy::MidiEvent::AsControlChange ( ) [inline]
```

Returns the data within the [MidiEvent](#) as a [NoteOnEvent](#) struct.

8.27.2.2 AsNoteOn()

`NoteOnEvent` `daisy::MidiEvent::AsNoteOn () [inline]`
 Returns the data within the `MidiEvent` as a `NoteOnEvent` struct

8.27.3 Member Data Documentation

8.27.3.1 channel

```
int daisy::MidiEvent::channel
&
```

8.27.3.2 data

```
uint8_t daisy::MidiEvent::data[2]
&
```

8.27.3.3 type

```
MidiMessageType daisy::MidiEvent::type
&
```

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

- `src/hid_midi.h`

8.28 daisy::MidiHandler Class Reference

Simple MIDI Handler

Parses bytes from an input into valid `MidiEvents`.

The `MidiEvents` fill a FIFO queue that the user can pop messages from.

```
#include <hid_midi.h>
```

Public Types

- enum `MidiInputMode` { `INPUT_MODE_NONE` = 0x00, `INPUT_MODE_UART1` = 0x01, `INPUT_MODE_USB_INT` = 0x02, `INPUT_MODE_USB_EXT` = 0x04 }
- enum `MidiOutputMode` { `OUTPUT_MODE_NONE` = 0x00, `OUTPUT_MODE_UART1` = 0x01, `OUTPUT_MODE_USB_INT` = 0x02, `OUTPUT_MODE_USB_EXT` = 0x04 }

Public Member Functions

- void `Init` (`MidiInputMode` in_mode, `MidiOutputMode` out_mode)
- void `StartReceive` ()
- void `Listen` ()
- void `Parse` (uint8_t byte)
- bool `HasEvents` () const
- `MidiEvent` `PopEvent` ()

8.28.1 Detailed Description

Simple MIDI Handler

Parses bytes from an input into valid `MidiEvents`.

The `MidiEvents` fill a FIFO queue that the user can pop messages from.

Author

shensley

Date

March 2020

8.28.2 Member Enumeration Documentation

8.28.2.1 MidiInputMode

```
enum daisy::MidiHandler::MidiInputMode
```

Input and Output can be configured separately Multiple Input modes can be selected by OR'ing the values.

Enumerator

INPUT_MODE_NONE	&
INPUT_MODE_UART1	&
INPUT_MODE_USB_INT	&
INPUT_MODE_USB_EXT	&

8.28.2.2 MidiOutputMode

```
enum daisy::MidiHandler::MidiOutputMode
```

Output mode

Enumerator

OUTPUT_MODE_NONE	&
OUTPUT_MODE_UART1	&
OUTPUT_MODE_USB_INT	&
OUTPUT_MODE_USB_EXT	&

8.28.3 Member Function Documentation

8.28.3.1 HasEvents()

```
bool daisy::MidiHandler::HasEvents ( ) const [inline]
```

Checks if there are unhandled messages in the queue

Returns

True if there are events to be handled, else false.

8.28.3.2 Init()

```
void daisy::MidiHandler::Init (
    MidiInputMode in_mode,
    MidiOutputMode out_mode )
```

Initializes the [MidiHandler](#)

Parameters

<i>in_mode</i>	Input mode
<i>out_mode</i>	Output mode

8.28.3.3 Listen()

```
void daisy::MidiHandler::Listen ( )
```

Start listening

8.28.3.4 Parse()

```
void daisy::MidiHandler::Parse (
    uint8_t byte )
```

Feed in bytes to state machine from a queue. Populates internal FIFO queue with MIDI Messages For example with
uart: midi.Parse(uart.PopRx());

Parameters

<i>byte</i>	&
-------------	---

8.28.3.5 PopEvent()

```
MidiEvent daisy::MidiHandler::PopEvent ( ) [inline]
```

Pops the oldest unhandled [MidiEvent](#) from the internal queue

Returns

The event to be handled

8.28.3.6 StartReceive()

```
void daisy::MidiHandler::StartReceive ( )
```

Starts listening on the selected input mode(s). [MidiEvent](#) Queue will begin to fill, and can be checked with
The documentation for this class was generated from the following file:

- src/hid_midi.h

8.29 daisy::NoteOnEvent Struct Reference

```
#include <hid_midi.h>
```

Public Attributes

- int [channel](#)
- uint8_t [note](#)
- uint8_t [velocity](#)

8.29.1 Detailed Description

Struct containing note, and velocity data for a given channel. Can be made from [MidiEvent](#)

8.29.2 Member Data Documentation

8.29.2.1 channel

```
int daisy::NoteOnEvent::channel
&
```

8.29.2.2 note

```
uint8_t daisy::NoteOnEvent::note
&
```

8.29.2.3 velocity

```
uint8_t daisy::NoteOnEvent::velocity
&
```

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

- src/hid_midi.h

8.30 daisy::OledDisplay Class Reference

```
#include <hid_oled_display.h>
```

Public Types

- enum Pins { DATA_COMMAND, RESET, NUM_PINS }

Public Member Functions

- void Init (dsy_gpio_pin *pin_cfg)
- void Fill (bool on)
- void DrawPixel (uint8_t x, uint8_t y, bool on)
- char WriteChar (char ch, FontDef font, bool on)
- char WriteString (char *str, FontDef font, bool on)
- void SetCursor (uint8_t x, uint8_t y)
- void Update ()

8.30.1 Detailed Description

Human Interface Driver for using an OLED Display (SSD1309) For all `bool on` arguments: true is on, false is off. Credit to Aleksander Alekseev (github.com/afiskon/stm32-ssd1306) on github for a great starting point. adapted for SSD1309 and H7 by shensley, 2020

8.30.2 Member Enumeration Documentation**8.30.2.1 Pins**

```
enum daisy::OledDisplay::Pins
```

GPIO Pins that need to be used independent of peripheral used.

Enumerator

DATA_COMMAND	Data command pin.
RESET	Reset pin
NUM_PINS	Num pins

8.30.3 Member Function Documentation

8.30.3.1 DrawPixel()

```
void daisy::OledDisplay::DrawPixel (
    uint8_t x,
    uint8_t y,
    bool on )
```

Sets the pixel at the specified coordinate to be on/off.

Parameters

<i>x</i>	x Coordinate
<i>y</i>	y coordinate
<i>on</i>	on or off

8.30.3.2 Fill()

```
void daisy::OledDisplay::Fill (
    bool on )
```

Fills the entire display with either on/off.

Parameters

<i>on</i>	Sets on or off.
-----------	-----------------

8.30.3.3 Init()

```
void daisy::OledDisplay::Init (
    dsy_gpio_pin * pin_cfg )
```

Takes an argument for the pin cfg

Parameters

<i>pin_cfg</i>	should be a pointer to an array of OledDisplay::NUM_PINS dsy_gpio_pins
----------------	--

8.30.3.4 SetCursor()

```
void daisy::OledDisplay::SetCursor (
    uint8_t x,
    uint8_t y )
```

Moves the 'Cursor' position used for WriteChar, and WriteStr to the specified coordinate.

Parameters

<i>x</i>	x pos
<i>y</i>	y pos

8.30.3.5 Update()

```
void daisy::OledDisplay::Update ( )
```

Writes the current display buffer to the OLED device using SPI or I2C depending on how the object was initialized.

8.30.3.6 WriteChar()

```
char daisy::OledDisplay::WriteChar (
    char ch,
    FontDef font,
    bool on )
```

Writes the character with the specific [FontDef](#) to the display buffer at the current Cursor position.

Parameters

<i>ch</i>	character to be written
<i>font</i>	font to be written in
<i>on</i>	on or off

Returns

&

8.30.3.7 WriteString()

```
char daisy::OledDisplay::WriteString (
    char * str,
    FontDef font,
    bool on )
```

Similar to WriteChar, except it will handle an entire String. Wrapping does not happen automatically, so the width of the string must be kept within the dimensions of the screen.

Parameters

<i>str</i>	string to be written
<i>font</i>	font to use
<i>on</i>	on or off

Returns

&

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

- src/hid_oled_display.h

8.31 daisy::Parameter Class Reference

```
#include <hid_parameter.h>
```

Public Types

- enum [Curve](#) {
[LINEAR](#), [EXPONENTIAL](#), [LOGARITHMIC](#), [CUBE](#),
[LAST](#) }

Public Member Functions

- [Parameter](#) ()
- [~Parameter](#) ()
- void [Init](#) ([AnalogControl](#) input, float min, float max, [Curve](#) curve)

- float [Process](#) ()
- float [Value](#) ()

8.31.1 Detailed Description

Simple parameter mapping tool that takes a 0-1 input from an hid_ctrl.

8.31.2 Member Enumeration Documentation

8.31.2.1 Curve

```
enum daisy::Parameter::Curve
```

Curves are applied to the output signal

Enumerator

LINEAR	Linear curve
EXPONENTIAL	Exponential curve
LOGARITHMIC	Logarithmic curve
CUBE	Cubic curve
LAST	Final enum element.

8.31.3 Constructor & Destructor Documentation

8.31.3.1 Parameter()

```
daisy::Parameter::Parameter ( ) [inline]
```

Constructor

8.31.3.2 ~Parameter()

```
daisy::Parameter::~~Parameter ( ) [inline]
```

Destructor

8.31.4 Member Function Documentation

8.31.4.1 Init()

```
void daisy::Parameter::Init (
    AnalogControl input,
    float min,
    float max,
    Curve curve )
```

initialize a parameter using an hid_ctrl object.

Parameters

<i>input</i>	- object containing the direct link to a hardware control source.
<i>min</i>	- bottom of range. (when input is 0.0)
<i>max</i>	- top of range (when input is 1.0)
<i>curve</i>	- the scaling curve for the input->output transformation.

8.31.4.2 Process()

```
float daisy::Parameter::Process ( )
```

processes the input signal, this should be called at the samplerate of the hid_ctrl passed in.

Returns

a float with the specified transformation applied.

8.31.4.3 Value()

```
float daisy::Parameter::Value ( ) [inline]
```

Returns

the current value from the parameter without processing another sample. this is useful if you need to use the value multiple times, and don't store

the output of process in a local variable.

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

- src/hid_parameter.h

8.32 daisy::RgbLed Class Reference

```
#include <hid_rgb_led.h>
```

Public Member Functions

- void [Init](#) ([dsy_gpio_pin](#) red, [dsy_gpio_pin](#) green, [dsy_gpio_pin](#) blue, bool invert)
- void [Set](#) (float r, float g, float b)
- void [SetColor](#) ([Color](#) c)
- void [Update](#) ()

8.32.1 Detailed Description

3x LEDs configured as an RGB for ease of use.

8.32.2 Member Function Documentation

8.32.2.1 Init()

```
void daisy::RgbLed::Init (
    dsy\_gpio\_pin red,
    dsy\_gpio\_pin green,
    dsy\_gpio\_pin blue,
    bool invert )
```

Initializes 3x GPIO Pins as red, green, and blue elements of an RGB LED

Parameters

<i>red</i>	Red element
<i>green</i>	Green element
<i>blue</i>	Blue element
<i>invert</i>	Flips led polarity

8.32.2.2 Set()

```
void daisy::RgbLed::Set (
    float r,
    float g,
    float b )
```

Sets each element of the LED with a floating point number 0-1

Parameters

<i>r</i>	Red element
<i>g</i>	Green element
<i>b</i>	Blue element

8.32.2.3 SetColor()

```
void daisy::RgbLed::SetColor (
    Color c )
```

Sets the RGB using a [Color](#) object.

Parameters

<i>c</i>	Color object to set.
----------	--------------------------------------

8.32.2.4 Update()

```
void daisy::RgbLed::Update ( )
```

Updates the PWM of the LED based on the current values. Should be called at a regular interval. (i.e. 1kHz/1ms)
The documentation for this class was generated from the following file:

- `src/hid_rgb_led.h`

8.33 daisy::RingBuffer< T, size > Class Template Reference

```
#include <util_ringbuffer.h>
```

Public Member Functions

- void [Init](#) ()
- size_t [capacity](#) () const
- size_t [writable](#) () const
- size_t [readable](#) () const
- void [Write](#) (T v)
- void [Overwrite](#) (T v)
- T [Read](#) ()
- T [ImmediateRead](#) ()
- void [Flush](#) ()
- void [Swallow](#) (size_t n)
- void [ImmediateRead](#) (T *destination, size_t num_elements)
- void [Overwrite](#) (const T *source, size_t num_elements)

8.33.1 Detailed Description

```
template<typename T, size_t size>
class daisy::RingBuffer< T, size >
```

Utility Ring Buffer

imported from pichenettes/stmlib

8.33.2 Member Function Documentation

8.33.2.1 capacity()

```
template<typename T , size_t size>
size_t daisy::RingBuffer< T, size >::capacity ( ) const [inline]
```

Returns

The total size of the ring buffer

8.33.2.2 Flush()

```
template<typename T , size_t size>
void daisy::RingBuffer< T, size >::Flush ( ) [inline]
Flushes unread elements from the ring buffer
```

8.33.2.3 ImmediateRead() [1/2]

```
template<typename T , size_t size>
T daisy::RingBuffer< T, size >::ImmediateRead ( ) [inline]
Reads next element from ring buffer immediately
```

Returns

read value

8.33.2.4 ImmediateRead() [2/2]

```
template<typename T , size_t size>
void daisy::RingBuffer< T, size >::ImmediateRead (
    T * destination,
    size_t num_elements ) [inline]
```

Reads a number of elements into a buffer immediately

Parameters

<i>destination</i>	buffer to write to
<i>num_elements</i>	number of elements in buffer

8.33.2.5 Init()

```
template<typename T , size_t size>
void daisy::RingBuffer< T, size >::Init ( ) [inline]
Initializes the Ring Buffer
```

8.33.2.6 Overwrite() [1/2]

```
template<typename T , size_t size>
void daisy::RingBuffer< T, size >::Overwrite (
    const T * source,
    size_t num_elements ) [inline]
```

Overwrites a number of elements using the source buffer as input.

Parameters

<i>source</i>	Input buffer
<i>num_elements</i>	Number of elements in source

8.33.2.7 Overwrite() [2/2]

```
template<typename T , size_t size>
void daisy::RingBuffer< T, size >::Overwrite (
    T v ) [inline]
```

Writes the new element to the ring buffer, overwriting unread data if necessary.

Parameters

<i>v</i>	Value to overwrite
----------	--------------------

8.33.2.8 Read()

```
template<typename T , size_t size>
T daisy::RingBuffer< T, size >::Read ( ) [inline]
```

Reads the first available element from the ring buffer

Returns

read value

8.33.2.9 readable()

```
template<typename T , size_t size>
size_t daisy::RingBuffer< T, size >::readable ( ) const [inline]
```

Returns

number of unread elements in ring buffer

8.33.2.10 Swallow()

```
template<typename T , size_t size>
void daisy::RingBuffer< T, size >::Swallow (
    size_t n ) [inline]
```

Read enough samples to make it possible to read 1 sample.

Parameters

<i>n</i>	Size of T ?
----------	-------------

8.33.2.11 writable()

```
template<typename T , size_t size>
size_t daisy::RingBuffer< T, size >::writable ( ) const [inline]
```

Returns

the number of samples that can be written to ring buffer without overwriting unread data.

8.33.2.12 Write()

```
template<typename T , size_t size>
void daisy::RingBuffer< T, size >::Write (
    T v ) [inline]
```

Writes the value to the next available position in the ring buffer

Parameters

v	Value to write
----------	----------------

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

- src/util_ringbuffer.h

8.34 daisy::RingBuffer< T, 0 > Class Template Reference

```
#include <util_ringbuffer.h>
```

Public Member Functions

- void [Init](#) ()
- size_t [capacity](#) () const
- size_t [writable](#) () const
- size_t [readable](#) () const
- void [Write](#) (T v)
- void [Overwrite](#) (T v)
- T [Read](#) ()
- T [ImmediateRead](#) ()
- void [Flush](#) ()
- void [ImmediateRead](#) (T *destination, size_t num_elements)
- void [Overwrite](#) (const T *source, size_t num_elements)

8.34.1 Detailed Description

```
template<typename T>
class daisy::RingBuffer< T, 0 >
```

Utility Ring Buffer imported from pichenettes/stmlib

8.34.2 Member Function Documentation

8.34.2.1 capacity()

```
template<typename T >
size_t daisy::RingBuffer< T, 0 >::capacity ( ) const [inline]
```

Returns

0

8.34.2.2 Flush()

```
template<typename T >
void daisy::RingBuffer< T, 0 >::Flush ( ) [inline]
```

Flush the buffer

8.34.2.3 ImmediateRead() [1/2]

```
template<typename T >
T daisy::RingBuffer< T, 0 >::ImmediateRead ( ) [inline]
```

Returns

Read value

8.34.2.4 ImmediateRead() [2/2]

```
template<typename T >
void daisy::RingBuffer< T, 0 >::ImmediateRead (
    T * destination,
    size_t num_elements ) [inline]
```

Parameters

<i>destination</i>	&
<i>num_elements</i>	&

8.34.2.5 Init()

```
template<typename T >
void daisy::RingBuffer< T, 0 >::Init ( ) [inline]
```

Initialize ringbuffer

8.34.2.6 Overwrite() [1/2]

```
template<typename T >
void daisy::RingBuffer< T, 0 >::Overwrite (
    const T * source,
    size_t num_elements ) [inline]
```

Parameters

<i>source</i>	3
<i>num_elements</i>	&

8.34.2.7 Overwrite() [2/2]

```
template<typename T >
void daisy::RingBuffer< T, 0 >::Overwrite (
    T v ) [inline]
```

Parameters

<i>v</i>	Value to overwrite
----------	--------------------

8.34.2.8 Read()

```
template<typename T >
T daisy::RingBuffer< T, 0 >::Read ( ) [inline]
```

Returns

Read value

8.34.2.9 readable()

```
template<typename T >
size_t daisy::RingBuffer< T, 0 >::readable ( ) const [inline]
```

Returns

0

8.34.2.10 writable()

```
template<typename T >
size_t daisy::RingBuffer< T, 0 >::writable ( ) const [inline]
```

Returns

0

8.34.2.11 Write()

```
template<typename T >
void daisy::RingBuffer< T, 0 >::Write (
    T v ) [inline]
```

Parameters

<i>v</i>	Value to write
----------	----------------

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

- src/util_ringbuffer.h

8.35 daisy::SdmmcHandler Class Reference

```
#include <per_sdmmc.h>
```

Public Member Functions

- void [Init](#) ()

8.35.1 Detailed Description

Configuration for interfacing with SD cards. Currently only supports operation using FatFS filesystem

8.35.2 Member Function Documentation

8.35.2.1 Init()

```
void daisy::SdmmcHandler::Init ( )
```

Initializes the SD Card Interface For now all settings are fixed (See todo at top of section)

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

- [src/per_sdmmc.h](#)

8.36 daisy::SdmmcHandlerInit Struct Reference

```
#include <per_sdmmc.h>
```

Public Attributes

- [SdmmcBitWidth](#) [bitdepth](#)
- [SdmmcSpeed](#) [speed](#)

8.36.1 Detailed Description

Structure for setting the options above. Used to intialize [SdmmcHandler](#)

8.36.2 Member Data Documentation

8.36.2.1 bitdepth

```
SdmmcBitWidth daisy::SdmmcHandlerInit::bitdepth
&
```

8.36.2.2 speed

```
SdmmcSpeed daisy::SdmmcHandlerInit::speed
&
```

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

- [src/per_sdmmc.h](#)

8.37 ShiftRegister595 Class Reference

Device Driver for 8-bit shift register.

CD74HC595 - 8-bit serial to parallel output shift.

```
#include <dev_sr_595.h>
```

Public Types

- enum [Pins](#) { [PIN_LATCH](#), [PIN_CLK](#), [PIN_DATA](#), [NUM_PINS](#) }

Public Member Functions

- void [Init](#) ([dsy_gpio_pin](#) *pin_cfg, size_t num_daisy_chained=1)
- void [Set](#) (uint8_t idx, bool state)
- void [Write](#) ()

8.37.1 Detailed Description

Device Driver for 8-bit shift register.

CD74HC595 - 8-bit serial to parallel output shift.

Author

shensley

Date

May 2020

8.37.2 Member Enumeration Documentation

8.37.2.1 Pins

enum [ShiftRegister595::Pins](#)

The following pins correspond to the hardware connections to the 595.

Enumerator

PIN_CLK	LATCH corresponds to Pin 12 "RCLK"
PIN_DATA	CLK corresponds to Pin 11 "SRCLK"
NUM_PINS	DATA corresponds to Pin 14 "SER"

8.37.3 Member Function Documentation

8.37.3.1 Init()

```
void ShiftRegister595::Init (
    dsy\_gpio\_pin * pin_cfg,
    size_t num_daisy_chained = 1 )
```

Initializes the GPIO, and data for the ShiftRegister

Parameters

<i>pin_cfg</i>	is an array of dsy_gpio_pin corresponding the the Pins enum above.
<i>num_daisy_chained</i>	(default = 1) is the number of 595 devices daisy chained together.

8.37.3.2 Set()

```
void ShiftRegister595::Set (
    uint8_t idx,
    bool state )
```

Sets the state of the specified output.

Parameters

<i>idx</i>	The index starts with QA on the first device and ends with QH on the last device.
<i>state</i>	A true state will set the output HIGH, while a false state will set the output LOW.

8.37.3.3 Write()

```
void ShiftRegister595::Write ( )
```

Writes the states of shift register out to the connected devices.

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

- src/dev_sr_595.h

8.38 daisy::SpiHandle Class Reference

```
#include <per_spi.h>
```

Public Member Functions

- void [Init](#) ()
- void [BlockingTransmit](#) (uint8_t *buff, size_t size)

8.38.1 Detailed Description

Handler for serial peripheral interface

8.38.2 Member Function Documentation**8.38.2.1 BlockingTransmit()**

```
void daisy::SpiHandle::BlockingTransmit (
    uint8_t * buff,
    size_t size )
```

Blocking transmit

Parameters

<i>*buff</i>	input buffer
<i>size</i>	buffer size

8.38.2.2 Init()

```
void daisy::SpiHandle::Init ( )
```

Initializes handler

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

- src/per_spi.h

8.39 daisy::Switch Class Reference

```
#include <hid_switch.h>
```


Public Types

- enum [Type](#) { [TYPE_TOGGLE](#), [TYPE_MOMENTARY](#) }
- enum [Polarity](#) { [POLARITY_NORMAL](#), [POLARITY_INVERTED](#) }
- enum [Pull](#) { [PULL_UP](#), [PULL_DOWN](#), [PULL_NONE](#) }

Public Member Functions

- void [Init](#) ([dsy_gpio_pin](#) pin, float update_rate, [Type](#) t, [Polarity](#) pol, [Pull](#) pu)
- void [Init](#) ([dsy_gpio_pin](#) pin, float update_rate)
- void [Debounce](#) ()
- bool [RisingEdge](#) () const
- bool [FallingEdge](#) () const
- bool [Pressed](#) () const
- float [TimeHeldMs](#) () const

8.39.1 Detailed Description

Generic Class for handling momentary/latching switches
 Inspired/influenced by Mutable Instruments (pichenettes) [Switch](#) classes

Author

Stephen Hensley

Date

December 2019

8.39.2 Member Enumeration Documentation

8.39.2.1 Polarity

enum [daisy::Switch::Polarity](#)

Specifies whether the pressed is HIGH or LOW.

Enumerator

POLARITY_NORMAL	&
POLARITY_INVERTED	&

8.39.2.2 Pull

enum [daisy::Switch::Pull](#)

Specifies whether to use built-in Pull Up/Down resistors to hold button at a given state when not engaged.

Enumerator

PULL_UP	&
PULL_DOWN	&
PULL_NONE	&

8.39.2.3 Type

```
enum daisy::Switch::Type
```

Specifies the expected behavior of the switch

Enumerator

TYPE_TOGGLE	&
TYPE_MOMENTARY	&

8.39.3 Member Function Documentation

8.39.3.1 Debounce()

```
void daisy::Switch::Debounce ( )
```

Called at `update_rate` to debounce and handle timing for the switch. In order for events not to be missed, its important that the Edge/Pressed checks be made at the same rate as the debounce function is being called.

8.39.3.2 FallingEdge()

```
bool daisy::Switch::FallingEdge ( ) const [inline]
```

Returns

true if the button was just released

8.39.3.3 Init() [1/2]

```
void daisy::Switch::Init (
    dsy_gpio_pin pin,
    float update_rate )
```

Simplified Init.

Parameters

<i>pin</i>	port/pin object to tell the switch which hardware pin to use.
<i>update_rate</i>	the rate at which the Debounce() function will be called. (used for timing).

8.39.3.4 Init() [2/2]

```
void daisy::Switch::Init (
    dsy_gpio_pin pin,
    float update_rate,
    Type t,
    Polarity pol,
    Pull pu )
```

Initializes the switch object with a given port/pin combo.

Parameters

<i>pin</i>	port/pin object to tell the switch which hardware pin to use.
<i>update_rate</i>	the rate at which the Debounce() function will be called. (used for timing).

Parameters

<i>t</i>	switch type – Default: TYPE_MOMENTARY
<i>pol</i>	switch polarity – Default: POLARITY_INVERTED
<i>pu</i>	switch pull up/down – Default: PULL_UP

8.39.3.5 Pressed()

```
bool daisy::Switch::Pressed ( ) const [inline]
```

Returns

true if the button is held down (or if the toggle is on)

8.39.3.6 RisingEdge()

```
bool daisy::Switch::RisingEdge ( ) const [inline]
```

Returns

true if a button was just pressed.

8.39.3.7 TimeHeldMs()

```
float daisy::Switch::TimeHeldMs ( ) const [inline]
```

Returns

the time in milliseconds that the button has been held (or toggle has been on)

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

- src/hid_switch.h

8.40 daisy::UartHandler Class Reference

```
#include <per_uart.h>
```

Public Member Functions

- void [Init](#) ()
- int [PollReceive](#) (uint8_t *buff, size_t size, uint32_t timeout)
- int [StartRx](#) (size_t size)
- bool [RxActive](#) ()
- int [FlushRx](#) ()
- int [PollTx](#) (uint8_t *buff, size_t size)
- uint8_t [PopRx](#) ()
- size_t [Readable](#) ()
- int [CheckError](#) ()

8.40.1 Detailed Description

Uart Peripheral

Author

shensley

Date

March 2020

8.40.2 Member Function Documentation

8.40.2.1 CheckError()

```
int daisy::UartHandler::CheckError ( )
```

Returns

the result of HAL_UART_GetError() to the user.

8.40.2.2 FlushRx()

```
int daisy::UartHandler::FlushRx ( )
```

Flushes the Receive Queue

Returns

OK or ERROR

8.40.2.3 Init()

```
void daisy::UartHandler::Init ( )
```

Initializes the UART Peripheral

8.40.2.4 PollReceive()

```
int daisy::UartHandler::PollReceive (
    uint8_t * buff,
    size_t size,
    uint32_t timeout )
```

Reads the amount of bytes in blocking mode with a 10ms timeout.

Parameters

<i>*buff</i>	Buffer to read to
<i>size</i>	Buff size
<i>timeout</i>	How long to timeout for (10ms?)

Returns

Data received

8.40.2.5 PollTx()

```
int daisy::UartHandler::PollTx (
    uint8_t * buff,
    size_t size )
```

Sends an amount of data in blocking mode.

Parameters

<i>*buff</i>	Buffer of data to send
<i>size</i>	Buffer size

Returns

OK or ERROR

8.40.2.6 PopRx()

```
uint8_t daisy::UartHandler::PopRx ( )
```

Pops the oldest byte from the FIFO.

Returns

Popped byte

8.40.2.7 Readable()

```
size_t daisy::UartHandler::Readable ( )
```

Checks if there are any unread bytes in the FIFO

Returns

1 or 0 ??

8.40.2.8 RxActive()

```
bool daisy::UartHandler::RxActive ( )
```

Returns

whether Rx DMA is listening or not.

8.40.2.9 StartRx()

```
int daisy::UartHandler::StartRx (
    size_t size )
```

Starts a DMA Receive callback to fill a buffer of specified size. Data is populated into a FIFO queue, and can be queried with the functions below. Maximum Buffer size is defined above. If a value outside of the maximum is specified, the size will be set to the maximum.

Parameters

<i>size</i>	Queue size
-------------	------------

Returns

OK or ERROR

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

- src/per_uart.h

8.41 UsbHandle Class Reference

Interface for initializing and using the USB Peripherals on the daisy.

```
#include <hid_usb.h>
```

Public Types

- enum [UsbPeriph](#) {
 FS_INTERNAL, FS_EXTERNAL, FS_BOTH, FS_INTERNAL,
 FS_EXTERNAL, FS_BOTH }
- enum [UsbPeriph](#) {
 FS_INTERNAL, FS_EXTERNAL, FS_BOTH, FS_INTERNAL,
 FS_EXTERNAL, FS_BOTH }
- typedef void(* [ReceiveCallback](#)) (uint8_t *buff, uint32_t *len)
- typedef void(* [ReceiveCallback](#)) (uint8_t *buff, uint32_t *len)

Public Member Functions

- void [Init](#) ([UsbPeriph](#) dev)
- void [TransmitInternal](#) (uint8_t *buff, size_t size)
- void [TransmitExternal](#) (uint8_t *buff, size_t size)
- void [SetReceiveCallback](#) ([ReceiveCallback](#) cb)
- void [Init](#) ([UsbPeriph](#) dev)
- void [TransmitInternal](#) (uint8_t *buff, size_t size)
- void [TransmitExternal](#) (uint8_t *buff, size_t size)
- void [SetReceiveCallback](#) ([ReceiveCallback](#) cb)

8.41.1 Detailed Description

Interface for initializing and using the USB Peripherals on the daisy.

Author

Stephen Hensley

Date

December 2019

8.41.2 Member Typedef Documentation

8.41.2.1 [ReceiveCallback](#) [1/2]

```
typedef void(* UsbHandle::ReceiveCallback) (uint8_t *buff, uint32_t *len)
```

Function called upon reception of a buffer

8.41.2.2 ReceiveCallback [2/2]

```
typedef void(* UsbHandle::ReceiveCallback) (uint8_t *buff, uint32_t *len)
```

Function called upon reception of a buffer

8.41.3 Member Enumeration Documentation

8.41.3.1 UsbPeriph [1/2]

```
enum UsbHandle::UsbPeriph
```

Specified which of the two USB Peripherals to initialize.

Enumerator

FS_INTERNAL	Internal pin
FS_EXTERNAL	FS External D+ pin is Pin 38 (GPIO32). FS External D- pin is Pin 37 (GPIO31)
FS_BOTH	Both
FS_INTERNAL	Internal pin
FS_EXTERNAL	FS External D+ pin is Pin 38 (GPIO32). FS External D- pin is Pin 37 (GPIO31)
FS_BOTH	Both

8.41.3.2 UsbPeriph [2/2]

```
enum UsbHandle::UsbPeriph
```

Specified which of the two USB Peripherals to initialize.

Enumerator

FS_INTERNAL	Internal pin
FS_EXTERNAL	FS External D+ pin is Pin 38 (GPIO32). FS External D- pin is Pin 37 (GPIO31)
FS_BOTH	Both
FS_INTERNAL	Internal pin
FS_EXTERNAL	FS External D+ pin is Pin 38 (GPIO32). FS External D- pin is Pin 37 (GPIO31)
FS_BOTH	Both

8.41.4 Member Function Documentation

8.41.4.1 Init() [1/2]

```
void UsbHandle::Init (  
    UsbPeriph dev )
```

Initializes the specified peripheral(s) as USB CDC Devices

Parameters

<i>dev</i>	Device to initialize
------------	----------------------

8.41.4.2 Init() [2/2]

```
void UsbHandle::Init (
    UsbPeriph dev )
```

Initializes the specified peripheral(s) as USB CDC Devices

Parameters

<i>dev</i>	Device to initialize
------------	----------------------

8.41.4.3 SetReceiveCallback() [1/2]

```
void UsbHandle::SetReceiveCallback (
    ReceiveCallback cb )
```

sets the callback to be called upon reception of new data

Parameters

<i>cb</i>	Function to serve as callback
-----------	-------------------------------

8.41.4.4 SetReceiveCallback() [2/2]

```
void UsbHandle::SetReceiveCallback (
    ReceiveCallback cb )
```

sets the callback to be called upon reception of new data

Parameters

<i>cb</i>	Function to serve as callback
-----------	-------------------------------

8.41.4.5 TransmitExternal() [1/2]

```
void UsbHandle::TransmitExternal (
    uint8_t * buff,
    size_t size )
```

Transmits a buffer of 'size' bytes from a USB port connected to the external USB Pins of the daisy seed.

Parameters

<i>buff</i>	Buffer to transmit
<i>size</i>	Buffer size

8.41.4.6 TransmitExternal() [2/2]

```
void UsbHandle::TransmitExternal (
    uint8_t * buff,
    size_t size )
```

Transmits a buffer of 'size' bytes from a USB port connected to the external USB Pins of the daisy seed.

Parameters

<i>buff</i>	Buffer to transmit
<i>size</i>	Buffer size

8.41.4.7 TransmitInternal() [1/2]

```
void UsbHandle::TransmitInternal (
    uint8_t * buff,
    size_t size )
```

Transmits a buffer of 'size' bytes from the on board USB FS port.

Parameters

<i>buff</i>	Buffer to transmit
<i>size</i>	Buffer size

8.41.4.8 TransmitInternal() [2/2]

```
void UsbHandle::TransmitInternal (
    uint8_t * buff,
    size_t size )
```

Transmits a buffer of 'size' bytes from the on board USB FS port.

Parameters

<i>buff</i>	Buffer to transmit
<i>size</i>	Buffer size

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

- src/hid_usb.h

8.42 WAV_FormatTypeDef Struct Reference

```
#include <util_wav_format.h>
```

Public Attributes

- uint32_t [ChunkId](#)
- uint32_t [FileSize](#)
- uint32_t [FileFormat](#)
- uint32_t [SubChunk1ID](#)
- uint32_t [SubChunk1Size](#)
- uint16_t [AudioFormat](#)
- uint16_t [NbrChannels](#)
- uint32_t [SampleRate](#)
- uint32_t [ByteRate](#)
- uint16_t [BlockAlign](#)
- uint16_t [BitPerSample](#)
- uint32_t [SubChunk2ID](#)
- uint32_t [SubCHunk2Size](#)

8.42.1 Detailed Description

Helper struct for handling the WAV file format

8.42.2 Member Data Documentation

8.42.2.1 AudioFormat

```
uint16_t WAV_FormatTypeDef::AudioFormat  
&
```

8.42.2.2 BitPerSample

```
uint16_t WAV_FormatTypeDef::BitPerSample  
&
```

8.42.2.3 BlockAlign

```
uint16_t WAV_FormatTypeDef::BlockAlign  
&
```

8.42.2.4 ByteRate

```
uint32_t WAV_FormatTypeDef::ByteRate  
&
```

8.42.2.5 ChunkId

```
uint32_t WAV_FormatTypeDef::ChunkId  
&
```

8.42.2.6 FileFormat

```
uint32_t WAV_FormatTypeDef::FileFormat  
&
```

8.42.2.7 FileSize

```
uint32_t WAV_FormatTypeDef::FileSize  
&
```

8.42.2.8 NbrChannels

```
uint16_t WAV_FormatTypeDef::NbrChannels  
&
```

8.42.2.9 SampleRate

```
uint32_t WAV_FormatTypeDef::SampleRate  
&
```

8.42.2.10 SubChunk1ID

```
uint32_t WAV_FormatTypeDef::SubChunk1ID  
&
```

8.42.2.11 SubChunk1Size

```
uint32_t WAV_FormatTypeDef::SubChunk1Size  
&
```

8.42.2.12 SubChunk2ID

```
uint32_t WAV_FormatTypeDef::SubChunk2ID  
&
```

8.42.2.13 SubCHunk2Size

```
uint32_t WAV_FormatTypeDef::SubCHunk2Size  
&
```

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

- `src/util_wav_format.h`

8.43 daisy::WavFileInfo Struct Reference

```
#include <hid_wavplayer.h>
```

Public Attributes

- [WAV_FormatTypeDef raw_data](#)
- `char name` [256]

8.43.1 Detailed Description

Struct containing details of Wav File.

8.43.2 Member Data Documentation

8.43.2.1 name

```
char daisy::WavFileInfo::name[256]  
Wav filename
```

8.43.2.2 raw_data

```
WAV_FormatTypeDef daisy::WavFileInfo::raw_data  
Raw wav data
```

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

- `src/hid_wavplayer.h`

8.44 daisy::WavPlayer Class Reference

```
#include <hid_wavplayer.h>
```

Public Member Functions

- `void Init ()`
- `int Open (size_t sel)`
- `int Close ()`
- `int16_t Stream ()`

- void [Prepare](#) ()
- void [Restart](#) ()
- void [SetLooping](#) (bool loop)
- bool [GetLooping](#) () const
- size_t [GetNumberFiles](#) () const
- size_t [GetCurrentFile](#) () const

8.44.1 Detailed Description

Wav Player that will load .wav files from an SD Card, and then provide a method of accessing the samples with double-buffering.

8.44.2 Member Function Documentation

8.44.2.1 Close()

```
int daisy::WavPlayer::Close ( )
```

Closes whatever file is currently open.

Returns

&

8.44.2.2 GetCurrentFile()

```
size_t daisy::WavPlayer::GetCurrentFile ( ) const [inline]
```

Returns

currently selected file.

8.44.2.3 GetLooping()

```
bool daisy::WavPlayer::GetLooping ( ) const [inline]
```

Returns

Whether the [WavPlayer](#) is looping or not.

8.44.2.4 GetNumberFiles()

```
size_t daisy::WavPlayer::GetNumberFiles ( ) const [inline]
```

Returns

The number of files loaded by the [WavPlayer](#)

8.44.2.5 Init()

```
void daisy::WavPlayer::Init ( )
```

Initializes the [WavPlayer](#), loading up to max_files of wav files from an SD Card.

8.44.2.6 Open()

```
int daisy::WavPlayer::Open (  
    size_t sel )
```

Opens the file at index sel for reading.

Parameters

<i>sel</i>	File to open
------------	--------------

8.44.2.7 Prepare()

```
void daisy::WavPlayer::Prepare ( )
```

Collects buffer for playback when needed.

8.44.2.8 Restart()

```
void daisy::WavPlayer::Restart ( )
```

Resets the playback position to the beginning of the file immediately

8.44.2.9 SetLooping()

```
void daisy::WavPlayer::SetLooping (
    bool loop ) [inline]
```

Sets whether or not the current file will repeat after completing playback.

Parameters

<i>loop</i>	To loop or not to loop.
-------------	-------------------------

8.44.2.10 Stream()

```
int16_t daisy::WavPlayer::Stream ( )
```

Returns

The next sample if playing, otherwise returns 0

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

- [src/hid_wavplayer.h](#)

Chapter 9

File Documentation

9.1 src/ffconf.h File Reference

```
#include "util_bsp_sd_diskio.h"  
#include <stdlib.h>
```

Macros

- `#define _FFCONF 68300`
- `#define _FS_READONLY 0`
- `#define _FS_MINIMIZE 0`
- `#define _USE_STRFUNC 2`
- `#define _USE_FIND 0`
- `#define _USE_MKFS 1`
- `#define _USE_FASTSEEK 1`
- `#define _USE_EXPAND 0`
- `#define _USE_CHMOD 0`
- `#define _USE_LABEL 0`
- `#define _USE_FORWARD 0`
- `#define _CODE_PAGE 850`
- `#define _USE_LFN 1`
- `#define _MAX_LFN 255`
- `#define _LFN_UNICODE 0`
- `#define _STRF_ENCODE 3`
- `#define _FS_RPATH 0`
- `#define _VOLUMES 1`
- `#define _STR_VOLUME_ID 0`
- `#define _VOLUME_STRS`
- `#define _MULTI_PARTITION 0`
- `#define _MIN_SS 512`
- `#define _MAX_SS 512`
- `#define _USE_TRIM 0`
- `#define _FS_NOFSINFO 0`
- `#define _FS_TINY 0`
- `#define _FS_EXFAT 0`
- `#define _FS_NORTC 0`
- `#define _NORTC_MON 6`
- `#define _NORTC_MDAY 4`
- `#define _NORTC_YEAR 2015`
- `#define _FS_LOCK 2`
- `#define _FS_REENTRANT 0`

- `#define _FS_TIMEOUT 1000`
- `#define _SYNC_t osSemaphoreId`
- `#define ff_malloc malloc`
- `#define ff_free free`

9.1.1 Detailed Description

Further fatfs support.

9.1.2 Macro Definition Documentation

9.1.2.1 _CODE_PAGE

```
#define _CODE_PAGE 850
```

This option specifies the OEM code page to be used on the target system. / Incorrect setting of the code page can cause a file open failure. // 1 - ASCII (No extended character. Non-LFN cfg. only) / 437 - U.S. / 720 - Arabic / 737 - Greek / 771 - KBL / 775 - Baltic / 850 - Latin 1 / 852 - Latin 2 / 855 - Cyrillic / 857 - Turkish / 860 - Portuguese / 861 - Icelandic / 862 - Hebrew / 863 - Canadian French / 864 - Arabic / 865 - Nordic / 866 - Russian / 869 - Greek 2 / 932 - Japanese (DBCS) / 936 - Simplified Chinese (DBCS) / 949 - Korean (DBCS) / 950 - Traditional Chinese (DBCS)

9.1.2.2 _FFCONF

```
#define _FFCONF 68300
```

FatFs - Generic FAT file system module R0.12c (C)ChaN, 2017

Attention

© Copyright (c) 2019 STMicroelectronics. All rights reserved.

This software component is licensed by ST under Ultimate Liberty license SLA0044, the "License"; You may not use this file except in compliance with the License. You may obtain a copy of the License at: www.st.com/SLA0044 Revision ID

9.1.2.3 _FS_EXFAT

```
#define _FS_EXFAT 0
```

This option switches support of exFAT file system. (0:Disable or 1:Enable) / When enable exFAT, also LFN needs to be enabled. (`_USE_LFN >= 1`) / Note that enabling exFAT discards C89 compatibility.

9.1.2.4 _FS_LOCK

```
#define _FS_LOCK 2
```

0:Disable or ≥ 1 :Enable The option `_FS_LOCK` switches file lock function to control duplicated file open / and illegal operation to open objects. This option must be 0 when `_FS_READONLY` / is 1. // 0: Disable file lock function. To avoid volume corruption, application program / should avoid illegal open, remove and rename to the open objects. / > 0 : Enable file lock function. The value defines how many files/sub-directories / can be opened simultaneously under file lock control. Note that the file / lock control is independent of re-entrancy.

9.1.2.5 _FS_MINIMIZE

```
#define _FS_MINIMIZE 0
```

0 to 3 This option defines minimization level to remove some basic API functions. // 0: All basic functions are enabled. / 1: `f_stat()`, `f_getfree()`, `f_unlink()`, `f_mkdir()`, `f_truncate()` and `f_rename()` / are removed. / 2: `f_opendir()`, `f_readdir()` and `f_closedir()` are removed in addition to 1. / 3: `f_lseek()` function is removed in addition to 2.

9.1.2.6 `_FS_NOFSINFO`

```
#define _FS_NOFSINFO 0
```

0,1,2 or 3 If you need to know correct free space on the FAT32 volume, set bit 0 of this / option, and `f_getfree()` function at first time after volume mount will force / a full FAT scan. Bit 1 controls the use of last allocated cluster number. // bit0=0: Use free cluster count in the FSINFO if available. / bit0=1: Do not trust free cluster count in the FSINFO. / bit1=0: Use last allocated cluster number in the FSINFO if available. / bit1=1: Do not trust last allocated cluster number in the FSINFO.

9.1.2.7 `_FS_NORTC`

```
#define _FS_NORTC 0
&
```

9.1.2.8 `_FS_READONLY`

```
#define _FS_READONLY 0
```

0:Read/Write or 1:Read only This option switches read-only configuration. (0:Read/Write or 1:Read-only) / Read-only configuration removes writing API functions, `f_write()`, `f_sync()`, `f_unlink()`, `f_mkdir()`, `f_chmod()`, `f_rename()`, `f_truncate()`, `f_getfree()` / and optional writing functions as well.

9.1.2.9 `_FS_REENTRANT`

```
#define _FS_REENTRANT 0
```

0:Disable or 1:Enable

9.1.2.10 `_FS_RPATH`

```
#define _FS_RPATH 0
```

0 to 2 This option configures support of relative path. // 0: Disable relative path and remove related functions. / 1: Enable relative path. `f_chdir()` and `f_chdrive()` are available. / 2: `f_getcwd()` function is available in addition to 1.

9.1.2.11 `_FS_TIMEOUT`

```
#define _FS_TIMEOUT 1000
```

Timeout period in unit of time ticks

9.1.2.12 `_FS_TINY`

```
#define _FS_TINY 0
```

0:Normal or 1:Tiny This option switches tiny buffer configuration. (0:Normal or 1:Tiny) / At the tiny configuration, size of file object (FIL) is reduced `_MAX_SS` bytes. / Instead of private sector buffer eliminated from the file object, common sector / buffer in the file system object (FATFS) is used for the file data transfer.

9.1.2.13 `_LFN_UNICODE`

```
#define _LFN_UNICODE 0
```

0:ANSI/OEM or 1:Unicode This option switches character encoding on the API. (0:ANSI/OEM or 1:UTF-16) / To use Unicode string for the path name, enable LFN and set `_LFN_UNICODE = 1`. / This option also affects behavior of string I/O functions.

9.1.2.14 `_MAX_LFN`

```
#define _MAX_LFN 255
```

Maximum LFN length to handle (12 to 255) The `_USE_LFN` switches the support of long file name (LFN). // 0: Disable support of LFN. `_MAX_LFN` has no effect. / 1: Enable LFN with static working buffer on the BSS. Always NOT thread-safe. / 2: Enable LFN with dynamic working buffer on the STACK. / 3: Enable LFN with dynamic working buffer on the HEAP. // To enable the LFN, Unicode handling functions (option/unicode.c) must be added / to the project. The working buffer occupies $(_MAX_LFN + 1) * 2$ bytes and / additional 608 bytes at exFAT enabled. `_MAX_LFN` can be in range from 12 to 255. / It should be set 255 to support full featured LFN operations. / When

use stack for the working buffer, take care on stack overflow. When use heap / memory for the working buffer, memory management functions, `ff_memalloc()` and `ff_memfree()`, must be added to the project.

9.1.2.15 `_MAX_SS`

```
#define _MAX_SS 512
```

512, 1024, 2048 or 4096 These options configure the range of sector size to be supported. (512, 1024, / 2048 or 4096) Always set both 512 for most systems, all type of memory cards and / harddisk. But a larger value may be required for on-board flash memory and some / type of optical media. When `_MAX_SS` is larger than `_MIN_SS`, FatFs is configured / to variable sector size and `GET_SECTOR_SIZE` command must be implemented to the / `disk_ioctl()` function.

9.1.2.16 `_MIN_SS`

```
#define _MIN_SS 512
```

512, 1024, 2048 or 4096

9.1.2.17 `_MULTI_PARTITION`

```
#define _MULTI_PARTITION 0
```

0:Single partition, 1:Multiple partition This option switches support of multi-partition on a physical drive. / By default (0), each logical drive number is bound to the same physical drive / number and only an FAT volume found on the physical drive will be mounted. / When multi-partition is enabled (1), each logical drive number can be bound to / arbitrary physical drive and partition listed in the `VolToPart[]`. Also `f_fdisk()` / `funciton` will be available.

9.1.2.18 `_NORTC_MDAY`

```
#define _NORTC_MDAY 4
```

&

9.1.2.19 `_NORTC_MON`

```
#define _NORTC_MON 6
```

&

9.1.2.20 `_NORTC_YEAR`

```
#define _NORTC_YEAR 2015
```

The option `_FS_NORTC` switches timestamp function. If the system does not have / any RTC function or valid timestamp is not needed, set `_FS_NORTC = 1` to disable / the timestamp function. All objects modified by FatFs will have a fixed timestamp / defined by `_NORTC_MON`, `_NORTC_MDAY` and `_NORTC_YEAR` in local time. / To enable timestamp function (`_FS_NORTC = 0`), `get_fattime()` function need to be / added to the project to get current time form real-time clock. `_NORTC_MON`, / `_NORTC_MDAY` and `_NORTC_YEAR` have no effect. / These options have no effect at read-only configuration (`_FS_READONLY = 1`).

9.1.2.21 `_STR_VOLUME_ID`

```
#define _STR_VOLUME_ID 0
```

0:Use only 0-9 for drive ID, 1:Use strings for drive ID

9.1.2.22 `_STRF_ENCODE`

```
#define _STRF_ENCODE 3
```

When `_LFN_UNICODE == 1`, this option selects the character encoding ON THE FILE to / be read/written via string I/O functions, `f_gets()`, `f_putc()`, `f_puts` and `f_printf()`. // 0: ANSI/OEM / 1: UTF-16LE / 2: UTF-16BE / 3: UTF-8 // This option has no effect when `_LFN_UNICODE == 0`.

9.1.2.23 _SYNC_t

```
#define _SYNC_t osSemaphoreId
```

The option `_FS_REENTRANT` switches the re-entrancy (thread safe) of the FatFs / module itself. Note that regardless of this option, file access to different / volume is always re-entrant and volume control functions, `f_mount()`, `f_mkfs()` / and `f_fdisk()` function, are always not re-entrant. Only file/directory access / to the same volume is under control of this function. // 0: Disable re-entrancy. `_FS_TIMEOUT` and `_SYNC_t` have no effect. / 1: Enable re-entrancy. Also user provided synchronization handlers, / `ff_req_grant()`, `ff_rel_grant()`, `ff_del_syncobj()` and `ff_cre_syncobj()` / function, must be added to the project. Samples are available in / option/syscall.c. // The `_FS_TIMEOUT` defines timeout period in unit of time tick. / The `_SYNC_t` defines O/S dependent sync object type. e.g. `HANDLE`, `ID`, `OS_EVENT*`, / `SemaphoreHandle_t` and etc.. A header file for O/S definitions needs to be / included somewhere in the scope of `ff.h`.

9.1.2.24 _USE_CHMOD

```
#define _USE_CHMOD 0
```

This option switches attribute manipulation functions, `f_chmod()` and `f_utime()`. / (0:Disable or 1:Enable) Also `_F_S_READONLY` needs to be 0 to enable this option.

9.1.2.25 _USE_EXPAND

```
#define _USE_EXPAND 0
```

This option switches `f_expand` function. (0:Disable or 1:Enable)

9.1.2.26 _USE_FASTSEEK

```
#define _USE_FASTSEEK 1
```

This option switches fast seek feature. (0:Disable or 1:Enable)

9.1.2.27 _USE_FIND

```
#define _USE_FIND 0
```

This option switches filtered directory read functions, `f_findfirst()` and / `f_findnext()`. (0:Disable, 1:Enable 2:Enable with matching `altname[]` too)

9.1.2.28 _USE_FORWARD

```
#define _USE_FORWARD 0
```

This option switches `f_forward()` function. (0:Disable or 1:Enable)

9.1.2.29 _USE_LABEL

```
#define _USE_LABEL 0
```

This option switches volume label functions, `f_getlabel()` and `f_setlabel()`. / (0:Disable or 1:Enable)

9.1.2.30 _USE_LFN

```
#define _USE_LFN 1
```

0 to 3

9.1.2.31 _USE_MKFS

```
#define _USE_MKFS 1
```

This option switches `f_mkfs()` function. (0:Disable or 1:Enable)

9.1.2.32 _USE_STRFUNC

```
#define _USE_STRFUNC 2
```

0:Disable or 1-2:Enable This option switches string functions, `f_gets()`, `f_putc()`, `f_puts()` and / `f_printf()`. // 0: Disable string functions. / 1: Enable without LF-CRLF conversion. / 2: Enable with LF-CRLF conversion.

9.1.2.33 `_USE_TRIM`

```
#define _USE_TRIM 0
```

This option switches support of ATA-TRIM. (0:Disable or 1:Enable) / To enable Trim function, also CTRL_TRIM command should be implemented to the / disk_ioctl() function.

9.1.2.34 `_VOLUME_STRS`

```
#define _VOLUME_STRS
```

Value:

```
"RAM", "NAND", "CF", "SD1", "SD2", "USB1", "USB2", \
"USB3"
```

`_STR_VOLUME_ID` switches string support of volume ID. / When `_STR_VOLUME_ID` is set to 1, also pre-defined strings can be used as drive / number in the path name. `_VOLUME_STRS` defines the drive ID strings for each / logical drives. Number of items must be equal to `_VOLUMES`. Valid characters for / the drive ID strings are: A-Z and 0-9.

9.1.2.35 `_VOLUMES`

```
#define _VOLUMES 1
```

Number of volumes (logical drives) to be used.

9.1.2.36 `ff_free`

```
#define ff_free free
```

define the `ff_malloc` `ff_free` macros as standard malloc free

9.1.2.37 `ff_malloc`

```
#define ff_malloc malloc
```

define the `ff_malloc` `ff_free` macros as standard malloc free

9.2 `src/hid_gatein.h` File Reference

```
#include "per_gpio.h"
```

Classes

- class [daisy::GateIn](#)
Generic Class for handling gate inputs through GPIO.

Namespaces

- [daisy](#)
Hardware defines and helpers for daisy field platform.

9.3 `src/hid_wavplayer.h` File Reference

```
#include "daisy_core.h"
```

```
#include "util_wav_format.h"
```

Classes

- struct [daisy::WavFileInfo](#)
- class [daisy::WavPlayer](#)

Namespaces

- [daisy](#)

Hardware defines and helpers for daisy field platform.

Macros

- `#define` [DSY_WAVPLAYER_H](#)
- `#define` [WAV_FILENAME_MAX](#) 256

9.3.1 Macro Definition Documentation

9.3.1.1 DSY_WAVPLAYER_H

```
#define DSY_WAVPLAYER_H  
Macro
```

9.3.1.2 WAV_FILENAME_MAX

```
#define WAV_FILENAME_MAX 256  
Maximum LFN (set to same in FatFs (ffconf.h))
```

9.4 src/usbd_cdc_if.h File Reference

: Header for usbd_cdc_if.c file.

```
#include "usbd_cdc.h"
```

Typedefs

- `typedef void(*` [CDC_ReceiveCallback](#)`)(uint8_t *buf, uint32_t *size)`

Functions

- `void` [CDC_Set_Rx_Callback_FS](#) ([CDC_ReceiveCallback](#) cb)
- `uint8_t` [CDC_Transmit_FS](#) (`uint8_t *Buf`, `uint16_t Len`)
- `uint8_t` [CDC_Transmit_HS](#) (`uint8_t *Buf`, `uint16_t Len`)

Variables

- `USBD_CDC_ItfTypeDef` [USBD_Interface_fops_FS](#)
- `USBD_CDC_ItfTypeDef` [USBD_Interface_fops_HS](#)

9.4.1 Detailed Description

: Header for usbd_cdc_if.c file.

Version

: v1.0_Cube

Attention

© Copyright (c) 2019 STMicroelectronics. All rights reserved.

This software component is licensed by ST under Ultimate Liberty license SLA0044, the "License"; You may not use this file except in compliance with the License. You may obtain a copy of the License at: www.st.com/SLA0044

9.5 src/usbd_conf.h File Reference

: Header for usbd_conf.c file.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "stm32h7xx.h"
#include "stm32h7xx_hal.h"
```

Macros

- #define [USBD_MAX_NUM_INTERFACES](#) 1U
- #define [USBD_MAX_NUM_CONFIGURATION](#) 1U
- #define [USBD_MAX_STR_DESC_SIZ](#) 512U
- #define [USBD_SUPPORT_USER_STRING](#) 0U
- #define [USBD_DEBUG_LEVEL](#) 3U
- #define [USBD_LPM_ENABLED](#) 0U
- #define [USBD_SELF_POWERED](#) 1U
- #define [DEVICE_FS](#) 0
- #define [DEVICE_HS](#) 1
- #define [USBD_malloc](#) malloc
- #define [USBD_free](#) free
- #define [USBD_memset](#) memset
- #define [USBD_memcpy](#) memcpy
- #define [USBD_Delay](#) HAL_Delay
- #define [USBD_UsrLog](#)(...)
- #define [USBD_ErrLog](#)(...)
- #define [USBD_DbgLog](#)(...)

9.5.1 Detailed Description

: Header for usbd_conf.c file.

Version

: v1.0_Cube

Attention

© Copyright (c) 2019 STMicroelectronics. All rights reserved.

This software component is licensed by ST under Ultimate Liberty license SLA0044, the "License"; You may not use this file except in compliance with the License. You may obtain a copy of the License at: www.st.com/SLA0044

9.6 src/usbd_desc.h File Reference

: Header for usbd_conf.c file.

```
#include "usbd_def.h"
```

Macros

- #define [DEVICE_ID1](#) (UID_BASE)
- #define [DEVICE_ID2](#) (UID_BASE + 0x4)
- #define [DEVICE_ID3](#) (UID_BASE + 0x8)
- #define [USB_SIZ_STRING_SERIAL](#) 0x1A

Variables

- USBD_DescriptorsTypeDef [HS_Desc](#)
- USBD_DescriptorsTypeDef [FS_Desc](#)

9.6.1 Detailed Description

: Header for usbd_conf.c file.

Version

: v1.0_Cube

Attention

© Copyright (c) 2019 STMicroelectronics. All rights reserved.

This software component is licensed by ST under Ultimate Liberty license SLA0044, the "License"; You may not use this file except in compliance with the License. You may obtain a copy of the License at: www.st.com/SLA0044

Index

- `_CODE_PAGE`
 - `ffconf.h`, [180](#)
- `_FFCONF`
 - `ffconf.h`, [180](#)
- `_FS_EXFAT`
 - `ffconf.h`, [180](#)
- `_FS_LOCK`
 - `ffconf.h`, [180](#)
- `_FS_MINIMIZE`
 - `ffconf.h`, [180](#)
- `_FS_NOFSINFO`
 - `ffconf.h`, [180](#)
- `_FS_NORTC`
 - `ffconf.h`, [181](#)
- `_FS_READONLY`
 - `ffconf.h`, [181](#)
- `_FS_REENTRANT`
 - `ffconf.h`, [181](#)
- `_FS_RPATH`
 - `ffconf.h`, [181](#)
- `_FS_TIMEOUT`
 - `ffconf.h`, [181](#)
- `_FS_TINY`
 - `ffconf.h`, [181](#)
- `_LFN_UNICODE`
 - `ffconf.h`, [181](#)
- `_MAX_LFN`
 - `ffconf.h`, [181](#)
- `_MAX_SS`
 - `ffconf.h`, [182](#)
- `_MIN_SS`
 - `ffconf.h`, [182](#)
- `_MULTI_PARTITION`
 - `ffconf.h`, [182](#)
- `_NORTC_MDAY`
 - `ffconf.h`, [182](#)
- `_NORTC_MON`
 - `ffconf.h`, [182](#)
- `_NORTC_YEAR`
 - `ffconf.h`, [182](#)
- `_STRF_ENCODE`
 - `ffconf.h`, [182](#)
- `_STR_VOLUME_ID`
 - `ffconf.h`, [182](#)
- `_SYNC_t`
 - `ffconf.h`, [182](#)
- `_USE_CHMOD`
 - `ffconf.h`, [183](#)
- `_USE_EXPAND`
 - `ffconf.h`, [183](#)
- `_USE_FASTSEEK`
 - `ffconf.h`, [183](#)
- `_USE_FIND`
 - `ffconf.h`, [183](#)
- `_USE_FORWARD`
 - `ffconf.h`, [183](#)
- `_USE_LABEL`
 - `ffconf.h`, [183](#)
- `_USE_LFN`
 - `ffconf.h`, [183](#)
- `_USE_MKFS`
 - `ffconf.h`, [183](#)
- `_USE_STRFUNC`
 - `ffconf.h`, [183](#)
- `_USE_TRIM`
 - `ffconf.h`, [183](#)
- `_VOLUMES`
 - `ffconf.h`, [184](#)
- `_VOLUME_STRS`
 - `ffconf.h`, [184](#)
- `~AnalogControl`
 - `daisy::AnalogControl`, [112](#)
- `~DaisyPatch`
 - `daisy::DaisyPatch`, [119](#)
- `~DaisyPetal`
 - `daisy::DaisyPetal`, [124](#)
- `~GateIn`
 - `daisy::GateIn`, [144](#)
- `~Parameter`
 - `daisy::Parameter`, [154](#)
- `a_direction`
 - `dsy_sai_handle`, [138](#)
- `adc`
 - `daisy::DaisySeed`, [133](#)
- `ANALOG_DIGITAL_CONVERSION`, [29](#)
 - `dsy_dac_bitdepth`, [29](#)
 - `DSY_DAC_BITS_12`, [29](#)
 - `DSY_DAC_BITS_8`, [29](#)
 - `DSY_DAC_BITS_LAST`, [29](#)
 - `dsy_dac_channel`, [29](#)
 - `DSY_DAC_CHN1`, [29](#)
 - `DSY_DAC_CHN2`, [29](#)
 - `DSY_DAC_CHN_BOTH`, [29](#)
 - `DSY_DAC_CHN_LAST`, [29](#)
 - `dsy_dac_init`, [30](#)
 - `dsy_dac_mode`, [29](#)
 - `DSY_DAC_MODE_LAST`, [30](#)
 - `DSY_DAC_MODE_POLLING`, [30](#)

- dsy_dac_start, [30](#)
 - dsy_dac_write, [30](#)
- AnalogControl
 - daisy::AnalogControl, [112](#)
- AsControlChange
 - daisy::MidiEvent, [147](#)
- AsNoteOn
 - daisy::MidiEvent, [147](#)
- AUDIO, [13](#)
 - dsy_audio_callback, [13](#)
 - dsy_audio_enter_bypass, [14](#)
 - dsy_audio_exit_bypass, [14](#)
 - DSY_AUDIO_EXTERNAL, [13](#)
 - dsy_audio_init, [14](#)
 - DSY_AUDIO_INTERNAL, [13](#)
 - DSY_AUDIO_LAST, [13](#)
 - dsy_audio_mc_callback, [13](#)
 - dsy_audio_passthru, [14](#)
 - dsy_audio_set_blocksize, [14](#)
 - dsy_audio_set_callback, [14](#)
 - dsy_audio_set_mc_callback, [14](#)
 - dsy_audio_silence, [14](#)
 - dsy_audio_start, [15](#)
 - dsy_audio_stop, [15](#)
- audio_handle
 - daisy::DaisySeed, [133](#)
- AudioBlockSize
 - daisy::DaisyPatch, [119](#)
 - daisy::DaisyPetal, [124](#)
 - daisy::DaisyPod, [129](#)
- AudioCallbackRate
 - daisy::DaisyPatch, [120](#)
 - daisy::DaisyPetal, [125](#)
 - daisy::DaisyPod, [129](#)
- AudioFormat
 - WAV_FormatTypeDef, [174](#)
- AudioSampleRate
 - daisy::DaisyPatch, [120](#)
 - daisy::DaisyPetal, [125](#)
 - daisy::DaisyPod, [129](#)
 - daisy::DaisySeed, [132](#)
- b_direction
 - dsy_sai_handle, [138](#)
- bitdepth
 - daisy::SdmmcHandlerInit, [162](#)
 - dsy_dac_handle, [135](#)
 - dsy_sai_handle, [138](#)
- BitPerSample
 - WAV_FormatTypeDef, [174](#)
- BLOCK_ERASE_32K_CMD
 - FLASH, [44](#)
- block_size
 - dsy_audio_handle, [134](#)
- BlockAlign
 - WAV_FormatTypeDef, [174](#)
- BlockingTransmit
 - daisy::SpiHandle, [164](#)
- BlockNbr
 - DSY_SD_CardInfoTypeDef, [140](#)
- BlockSize
 - DSY_SD_CardInfoTypeDef, [140](#)
- BLUE
 - daisy::Color, [115](#)
- Blue
 - daisy::Color, [115](#)
- blue
 - color, [114](#)
- BOARDS, [67](#)
 - CV_2, [68](#)
 - CV_3, [68](#)
 - CV_4, [68](#)
 - CV_LAST, [68](#)
 - daisy_field_init, [69](#)
 - f2s16, [70](#)
 - f2s24, [70](#)
 - KNOB_1, [68](#)
 - KNOB_2, [68](#)
 - KNOB_3, [68](#)
 - KNOB_4, [68](#)
 - KNOB_5, [68](#)
 - KNOB_6, [68](#)
 - KNOB_7, [68](#)
 - KNOB_8, [68](#)
 - KNOB_LAST, [68](#)
 - LED_KEY_A1, [68](#)
 - LED_KEY_A2, [68](#)
 - LED_KEY_A3, [68](#)
 - LED_KEY_A4, [68](#)
 - LED_KEY_A5, [68](#)
 - LED_KEY_A6, [68](#)
 - LED_KEY_A7, [68](#)
 - LED_KEY_A8, [68](#)
 - LED_KEY_B1, [69](#)
 - LED_KEY_B2, [69](#)
 - LED_KEY_B3, [69](#)
 - LED_KEY_B4, [69](#)
 - LED_KEY_B5, [69](#)
 - LED_KEY_B6, [69](#)
 - LED_KEY_B7, [69](#)
 - LED_KEY_B8, [69](#)
 - LED_KNOB_1, [69](#)
 - LED_KNOB_2, [69](#)
 - LED_KNOB_3, [69](#)
 - LED_KNOB_4, [69](#)
 - LED_KNOB_5, [69](#)
 - LED_KNOB_6, [69](#)
 - LED_KNOB_7, [69](#)
 - LED_KNOB_8, [69](#)
 - LED_LAST, [69](#)
 - LED_SW_1, [69](#)
 - LED_SW_2, [69](#)
 - s162f, [70](#)
 - s242f, [70](#)
 - SW_1, [68](#)
 - SW_2, [68](#)
 - SW_3, [68](#)

- SW_LAST, [68](#)
- BSP_SD_AbortCallback
 - UTILITY, [73](#)
- BSP_SD_CardInfo
 - UTILITY, [72](#)
- BSP_SD_Erase
 - UTILITY, [73](#)
- BSP_SD_GetCardInfo
 - UTILITY, [74](#)
- BSP_SD_GetCardState
 - UTILITY, [74](#)
- BSP_SD_Init
 - UTILITY, [74](#)
- BSP_SD_IsDetected
 - UTILITY, [74](#)
- BSP_SD_ITConfig
 - UTILITY, [74](#)
- BSP_SD_ReadBlocks
 - UTILITY, [75](#)
- BSP_SD_ReadBlocks_DMA
 - UTILITY, [75](#)
- BSP_SD_ReadCpltCallback
 - UTILITY, [75](#)
- BSP_SD_WriteBlocks
 - UTILITY, [75](#)
- BSP_SD_WriteBlocks_DMA
 - UTILITY, [76](#)
- BSP_SD_WriteCpltCallback
 - UTILITY, [76](#)
- button1
 - daisy::DaisyPod, [131](#)
- button2
 - daisy::DaisyPod, [131](#)
- BUTTON_2
 - daisy::DaisyPod, [129](#)
- BUTTON_LAST
 - daisy::DaisyPod, [129](#)
- buttons
 - daisy::DaisyPod, [131](#)
- ByteRate
 - WAV_FormatTypeDef, [174](#)
- capacity
 - daisy::RingBuffer< T, 0 >, [159](#)
 - daisy::RingBuffer< T, size >, [157](#)
- CardSpeed
 - DSY_SD_CardInfoTypeDef, [140](#)
- CardType
 - DSY_SD_CardInfoTypeDef, [140](#)
- CardVersion
 - DSY_SD_CardInfoTypeDef, [140](#)
- CDC_ReceiveCallback
 - USBDCDC_IF_Exported_Types, [82](#)
- CDC_Set_Rx_Callback_FS
 - USBDCDC_IF_Exported_FunctionsPrototype, [85](#)
- CDC_Transmit_FS
 - USBDCDC_IF_Exported_FunctionsPrototype, [85](#)
- CDC_Transmit_HS
 - USBDCDC_IF_Exported_FunctionsPrototype, [85](#)
- ChangeAudioCallback
 - daisy::DaisyPatch, [120](#)
 - daisy::DaisyPetal, [125](#)
 - daisy::DaisyPod, [129](#)
- channel
 - daisy::ControlChangeEvent, [117](#)
 - daisy::MidiEvent, [148](#)
 - daisy::NoteOnEvent, [150](#)
- ChannelPressure
 - EXTERNAL, [18](#)
- CheckError
 - daisy::UartHandler, [168](#)
- ChunkId
 - WAV_FormatTypeDef, [174](#)
- Class
 - DSY_SD_CardInfoTypeDef, [140](#)
- CLEAR_FLAG_STATUS_REG_CMD
 - FLASH, [44](#)
- ClearLeds
 - daisy::DaisyPetal, [125](#)
 - daisy::DaisyPod, [130](#)
- clk
 - dsy_sr_4021_handle, [141](#)
- Close
 - daisy::WavPlayer, [176](#)
- CODEC, [61](#)
 - codec_ak4556_init, [61](#)
 - codec_pcm3060_init, [61](#)
 - codec_wm8731_enter_bypass, [62](#)
 - codec_wm8731_exit_bypass, [62](#)
 - codec_wm8731_init, [62](#)
 - sa_audio_callback, [61](#)
- codec_ak4556_init
 - CODEC, [61](#)
- codec_frame_t, [113](#)
 - l, [114](#)
 - r, [114](#)
- codec_pcm3060_init
 - CODEC, [61](#)
- codec_wm8731_enter_bypass
 - CODEC, [62](#)
- codec_wm8731_exit_bypass
 - CODEC, [62](#)
- codec_wm8731_init
 - CODEC, [62](#)
- color, [114](#)
 - blue, [114](#)
 - green, [114](#)
 - red, [114](#)
- Configure
 - daisy::DaisySeed, [132](#)
- control_number
 - daisy::ControlChangeEvent, [117](#)
- ControlChange
 - EXTERNAL, [18](#)
- CONTROLS, [16](#)
- controls
 - daisy::DaisyPatch, [121](#)

- cs
 - dsy_sr_4021_handle, 141
- Ctrl
 - daisy::DaisyPatch, 119
- CUBE
 - daisy::Parameter, 154
- cube
 - UTILITY, 76
- Curve
 - daisy::Parameter, 154
- CV_2
 - BOARDS, 68
- CV_3
 - BOARDS, 68
- CV_4
 - BOARDS, 68
- CV_LAST
 - BOARDS, 68
- cvs
 - daisy::daisy_field, 117
- CYAN
 - daisy::Color, 115
- dac_handle
 - daisy::DaisySeed, 133
- daisy, 105
- daisy::AdcChannelConfig, 107
 - InitMux, 108
 - InitSingle, 108
 - mux_channels_, 108
 - mux_pin_, 108
 - MUX_SEL_0, 107
 - MUX_SEL_1, 107
 - MUX_SEL_2, 107
 - MUX_SEL_LAST, 107
 - MuxPin, 107
 - pin_, 108
- daisy::AdcHandle, 109
 - Get, 109
 - GetFloat, 110
 - GetMux, 110
 - GetMuxFloat, 110
 - GetMuxPtr, 111
 - GetPtr, 111
 - Init, 111
 - OverSampling, 109
 - OVS_1024, 109
 - OVS_128, 109
 - OVS_16, 109
 - OVS_256, 109
 - OVS_32, 109
 - OVS_4, 109
 - OVS_512, 109
 - OVS_64, 109
 - OVS_8, 109
 - OVS_LAST, 109
 - OVS_NONE, 109
 - Start, 112
 - Stop, 112
- daisy::AnalogControl, 112
 - ~AnalogControl, 112
 - AnalogControl, 112
 - Init, 113
 - InitBipolarCv, 113
 - Process, 113
 - Value, 113
- daisy::Color, 115
 - BLUE, 115
 - Blue, 115
 - CYAN, 115
 - GOLD, 115
 - GREEN, 115
 - Green, 116
 - Init, 116
 - LAST, 115
 - OFF, 115
 - PresetColor, 115
 - PURPLE, 115
 - RED, 115
 - Red, 116
 - WHITE, 115
- daisy::ControlChangeEvent, 116
 - channel, 117
 - control_number, 117
 - value, 117
- daisy::daisy_field, 117
 - cvs, 117
 - gate_in, 117
 - gate_out, 117
 - keyboard_sr, 117
 - knobs, 118
 - seed, 118
 - switches, 118
- daisy::DaisyPatch, 118
 - ~DaisyPatch, 119
 - AudioBlockSize, 119
 - AudioCallbackRate, 120
 - AudioSampleRate, 120
 - ChangeAudioCallback, 120
 - controls, 121
 - Ctrl, 119
 - DaisyPatch, 119
 - DebounceControls, 120
 - DelayMs, 120
 - display, 121
 - DisplayControls, 120
 - encoder, 121
 - GATE_IN_LAST, 119
 - gate_input, 121
 - gate_output, 122
 - GateInput, 119
 - GetCtrlValue, 120
 - Init, 121
 - midi, 122
 - seed, 122
 - SetAudioBlockSize, 121
 - StartAdc, 121

- StartAudio, 121
- UpdateAnalogControls, 121
- daisy::DaisyPetal, 122
 - ~DaisyPetal, 124
 - AudioBlockSize, 124
 - AudioCallbackRate, 125
 - AudioSampleRate, 125
 - ChangeAudioCallback, 125
 - ClearLeds, 125
 - DaisyPetal, 124
 - DebounceControls, 125
 - DelayMs, 125
 - encoder, 127
 - expression, 127
 - footswitch_led, 127
 - FOOTSWITCH_LED_1, 123
 - FOOTSWITCH_LED_2, 123
 - FOOTSWITCH_LED_3, 123
 - FOOTSWITCH_LED_4, 123
 - FOOTSWITCH_LED_LAST, 123
 - FootswitchLed, 123
 - GetExpression, 125
 - GetKnobValue, 125
 - Init, 126
 - Knob, 123
 - knob, 127
 - KNOB_1, 123
 - KNOB_2, 123
 - KNOB_3, 123
 - KNOB_4, 123
 - KNOB_5, 123
 - KNOB_6, 123
 - KNOB_LAST, 123
 - ring_led, 127
 - RING_LED_1, 124
 - RING_LED_2, 124
 - RING_LED_3, 124
 - RING_LED_4, 124
 - RING_LED_5, 124
 - RING_LED_6, 124
 - RING_LED_7, 124
 - RING_LED_8, 124
 - RING_LED_LAST, 124
 - RingLed, 124
 - seed, 127
 - SetAudioBlockSize, 126
 - SetFootswitchLed, 126
 - SetRingLed, 126
 - StartAdc, 126
 - StartAudio, 127
 - Sw, 124
 - SW_1, 124
 - SW_2, 124
 - SW_3, 124
 - SW_4, 124
 - SW_5, 124
 - SW_6, 124
 - SW_7, 124
 - SW_LAST, 124
 - switches, 127
 - UpdateAnalogControls, 127
 - UpdateLeds, 127
- daisy::DaisyPod, 128
 - AudioBlockSize, 129
 - AudioCallbackRate, 129
 - AudioSampleRate, 129
 - button1, 131
 - button2, 131
 - BUTTON_2, 129
 - BUTTON_LAST, 129
 - buttons, 131
 - ChangeAudioCallback, 129
 - ClearLeds, 130
 - DebounceControls, 130
 - DelayMs, 130
 - encoder, 131
 - GetKnobValue, 130
 - Init, 130
 - Knob, 129
 - knob1, 131
 - knob2, 131
 - KNOB_2, 129
 - KNOB_LAST, 129
 - knobs, 131
 - led1, 131
 - led2, 131
 - seed, 132
 - SetAudioBlockSize, 130
 - StartAdc, 130
 - StartAudio, 130
 - Sw, 129
 - UpdateAnalogControls, 131
 - UpdateLeds, 131
- daisy::DaisySeed, 132
 - adc, 133
 - audio_handle, 133
 - AudioSampleRate, 132
 - Configure, 132
 - dac_handle, 133
 - GetPin, 133
 - i2c1_handle, 134
 - i2c2_handle, 134
 - Init, 133
 - qspi_handle, 134
 - sai_handle, 134
 - sdram_handle, 134
 - SetAudioBlockSize, 133
 - SetLed, 133
 - SetTestPoint, 133
 - StartAudio, 133
 - usb_handle, 134
- daisy::Encoder, 142
 - Debounce, 142
 - FallingEdge, 143
 - Increment, 143
 - Init, 143

- Pressed, 143
- RisingEdge, 143
- TimeHeldMs, 143
- daisy::GateIn, 144
 - ~GateIn, 144
 - GateIn, 144
 - Init, 145
 - Trig, 145
- daisy::Led, 145
 - Init, 145
 - Set, 147
 - Update, 147
- daisy::MidiEvent, 147
 - AsControlChange, 147
 - AsNoteOn, 147
 - channel, 148
 - data, 148
 - type, 148
- daisy::MidiHandler, 148
 - HasEvents, 149
 - Init, 149
 - INPUT_MODE_NONE, 149
 - INPUT_MODE_UART1, 149
 - INPUT_MODE_USB_EXT, 149
 - INPUT_MODE_USB_INT, 149
 - Listen, 150
 - MidiInputMode, 149
 - MidiOutputMode, 149
 - OUTPUT_MODE_NONE, 149
 - OUTPUT_MODE_UART1, 149
 - OUTPUT_MODE_USB_EXT, 149
 - OUTPUT_MODE_USB_INT, 149
 - Parse, 150
 - PopEvent, 150
 - StartReceive, 150
- daisy::NoteOnEvent, 150
 - channel, 150
 - note, 150
 - velocity, 151
- daisy::OledDisplay, 151
 - DATA_COMMAND, 151
 - DrawPixel, 151
 - Fill, 152
 - Init, 152
 - NUM_PINS, 151
 - Pins, 151
 - RESET, 151
 - SetCursor, 152
 - Update, 152
 - WriteChar, 152
 - WriteString, 153
- daisy::Parameter, 153
 - ~Parameter, 154
 - CUBE, 154
 - Curve, 154
 - EXPONENTIAL, 154
 - Init, 154
 - LAST, 154
 - LINEAR, 154
 - LOGARITHMIC, 154
 - Parameter, 154
 - Process, 155
 - Value, 155
- daisy::RgbLed, 155
 - Init, 155
 - Set, 156
 - SetColor, 156
 - Update, 156
- daisy::RingBuffer< T, 0 >, 159
 - capacity, 159
 - Flush, 160
 - ImmediateRead, 160
 - Init, 160
 - Overwrite, 160
 - Read, 161
 - readable, 161
 - writable, 161
 - Write, 161
- daisy::RingBuffer< T, size >, 156
 - capacity, 157
 - Flush, 157
 - ImmediateRead, 157
 - Init, 157
 - Overwrite, 157, 158
 - Read, 158
 - readable, 158
 - Swallow, 158
 - writable, 159
 - Write, 159
- daisy::SdmmcHandler, 161
 - Init, 162
- daisy::SdmmcHandlerInit, 162
 - bitdepth, 162
 - speed, 162
- daisy::SpiHandle, 164
 - BlockingTransmit, 164
 - Init, 164
- daisy::Switch, 164
 - Debounce, 166
 - FallingEdge, 166
 - Init, 166
 - Polarity, 165
 - POLARITY_INVERTED, 165
 - POLARITY_NORMAL, 165
 - Pressed, 167
 - Pull, 165
 - PULL_DOWN, 165
 - PULL_NONE, 165
 - PULL_UP, 165
 - RisingEdge, 167
 - TimeHeldMs, 167
 - Type, 165
 - TYPE_MOMENTARY, 166
 - TYPE_TOGGLE, 166
- daisy::UartHandler, 167
 - CheckError, 168

- FlushRx, [168](#)
- Init, [168](#)
- PollReceive, [168](#)
- PollTx, [169](#)
- PopRx, [169](#)
- Readable, [169](#)
- RxActive, [169](#)
- StartRx, [169](#)
- daisy::WavFileInfo, [175](#)
 - name, [175](#)
 - raw_data, [175](#)
- daisy::WavPlayer, [175](#)
 - Close, [176](#)
 - GetCurrentFile, [176](#)
 - GetLooping, [176](#)
 - GetNumberFiles, [176](#)
 - Init, [176](#)
 - Open, [176](#)
 - Prepare, [178](#)
 - Restart, [178](#)
 - SetLooping, [178](#)
 - Stream, [178](#)
- daisy_field_init
 - BOARDS, [69](#)
- DaisyPatch
 - daisy::DaisyPatch, [119](#)
- DaisyPetal
 - daisy::DaisyPetal, [124](#)
- data
 - daisy::MidiEvent, [148](#)
 - dsy_sr_4021_handle, [141](#)
 - FontDef, [144](#)
- DATA_COMMAND
 - daisy::OledDisplay, [151](#)
- Debounce
 - daisy::Encoder, [142](#)
 - daisy::Switch, [166](#)
- DebounceControls
 - daisy::DaisyPatch, [120](#)
 - daisy::DaisyPetal, [125](#)
 - daisy::DaisyPod, [130](#)
- DelayMs
 - daisy::DaisyPatch, [120](#)
 - daisy::DaisyPetal, [125](#)
 - daisy::DaisyPod, [130](#)
- dev0_i2c
 - dsy_audio_handle, [135](#)
- dev1_i2c
 - dsy_audio_handle, [135](#)
- DEVICE, [38](#)
- device
 - dsy_qspi_handle, [138](#)
 - dsy_sai_handle, [139](#)
- DEVICE_FS
 - USBD_DESC_Exported_Defines, [88](#)
- DEVICE_HS
 - USBD_CONF_Exported_Defines, [88](#)
- DEVICE_ID1
 - USBD_DESC_Exported_Constants, [95](#)
- DEVICE_ID2
 - USBD_DESC_Exported_Constants, [95](#)
- DEVICE_ID3
 - USBD_DESC_Exported_Constants, [95](#)
- DIE_ERASE_CMD
 - FLASH, [45](#)
- display
 - daisy::DaisyPatch, [121](#)
- DisplayControls
 - daisy::DaisyPatch, [120](#)
- DMA_BUFFER_MEM_SECTION
 - UTILITY, [72](#)
- DrawPixel
 - daisy::OledDisplay, [151](#)
- dsy_audio_bitdepth
 - SERIAL, [21](#)
- DSY_AUDIO_BITDEPTH_16
 - SERIAL, [21](#)
- DSY_AUDIO_BITDEPTH_24
 - SERIAL, [21](#)
- DSY_AUDIO_BITDEPTH_LAST
 - SERIAL, [21](#)
- dsy_audio_callback
 - AUDIO, [13](#)
- dsy_audio_device
 - SERIAL, [21](#)
- DSY_AUDIO_DEVICE_AK4556
 - SERIAL, [21](#)
- DSY_AUDIO_DEVICE_LAST
 - SERIAL, [21](#)
- DSY_AUDIO_DEVICE_PCM3060
 - SERIAL, [21](#)
- DSY_AUDIO_DEVICE_WM8731
 - SERIAL, [21](#)
- dsy_audio_dir
 - SERIAL, [21](#)
- dsy_audio_enter_bypass
 - AUDIO, [14](#)
- dsy_audio_exit_bypass
 - AUDIO, [14](#)
- DSY_AUDIO_EXTERNAL
 - AUDIO, [13](#)
- dsy_audio_handle, [134](#)
 - block_size, [134](#)
 - dev0_i2c, [135](#)
 - dev1_i2c, [135](#)
 - sai, [135](#)
- dsy_audio_init
 - AUDIO, [14](#)
- DSY_AUDIO_INIT_BOTH
 - SERIAL, [22](#)
- DSY_AUDIO_INIT_LAST
 - SERIAL, [22](#)
- DSY_AUDIO_INIT_NONE
 - SERIAL, [22](#)
- DSY_AUDIO_INIT_SAI1
 - SERIAL, [22](#)

DSY_AUDIO_INIT_SAI2
 SERIAL, [22](#)
 DSY_AUDIO_INTERNAL
 AUDIO, [13](#)
 DSY_AUDIO_LAST
 AUDIO, [13](#)
 dsy_audio_mc_callback
 AUDIO, [13](#)
 DSY_AUDIO_NONE
 SERIAL, [21](#)
 dsy_audio_passthru
 AUDIO, [14](#)
 DSY_AUDIO_RX
 SERIAL, [22](#)
 dsy_audio_sai
 SERIAL, [22](#)
 dsy_audio_samplerate
 SERIAL, [22](#)
 DSY_AUDIO_SAMPLERATE_32K
 SERIAL, [22](#)
 DSY_AUDIO_SAMPLERATE_48K
 SERIAL, [22](#)
 DSY_AUDIO_SAMPLERATE_96K
 SERIAL, [22](#)
 DSY_AUDIO_SAMPLERATE_LAST
 SERIAL, [22](#)
 dsy_audio_set_blocksize
 AUDIO, [14](#)
 dsy_audio_set_callback
 AUDIO, [14](#)
 dsy_audio_set_mc_callback
 AUDIO, [14](#)
 dsy_audio_silence
 AUDIO, [14](#)
 dsy_audio_start
 AUDIO, [15](#)
 dsy_audio_stop
 AUDIO, [15](#)
 dsy_audio_sync
 SERIAL, [22](#)
 DSY_AUDIO_SYNC_LAST
 SERIAL, [22](#)
 DSY_AUDIO_SYNC_MASTER
 SERIAL, [22](#)
 DSY_AUDIO_SYNC_SLAVE
 SERIAL, [22](#)
 DSY_AUDIO_TX
 SERIAL, [22](#)
 dsy_dac_bitdepth
 ANALOG_DIGITAL_CONVERSION, [29](#)
 DSY_DAC_BITS_12
 ANALOG_DIGITAL_CONVERSION, [29](#)
 DSY_DAC_BITS_8
 ANALOG_DIGITAL_CONVERSION, [29](#)
 DSY_DAC_BITS_LAST
 ANALOG_DIGITAL_CONVERSION, [29](#)
 dsy_dac_channel
 ANALOG_DIGITAL_CONVERSION, [29](#)
 DSY_DAC_CHN1
 ANALOG_DIGITAL_CONVERSION, [29](#)
 DSY_DAC_CHN2
 ANALOG_DIGITAL_CONVERSION, [29](#)
 DSY_DAC_CHN_BOTH
 ANALOG_DIGITAL_CONVERSION, [29](#)
 DSY_DAC_CHN_LAST
 ANALOG_DIGITAL_CONVERSION, [29](#)
 dsy_dac_handle, [135](#)
 bitdepth, [135](#)
 mode, [135](#)
 pin_config, [135](#)
 dsy_dac_init
 ANALOG_DIGITAL_CONVERSION, [30](#)
 dsy_dac_mode
 ANALOG_DIGITAL_CONVERSION, [29](#)
 DSY_DAC_MODE_LAST
 ANALOG_DIGITAL_CONVERSION, [30](#)
 DSY_DAC_MODE_POLLING
 ANALOG_DIGITAL_CONVERSION, [30](#)
 dsy_dac_start
 ANALOG_DIGITAL_CONVERSION, [30](#)
 dsy_dac_write
 ANALOG_DIGITAL_CONVERSION, [30](#)
 dsy_dma_init
 SYSTEM, [36](#)
 dsy_get_unique_id
 UTILITY, [76](#)
 dsy_gpio, [136](#)
 mode, [136](#)
 pin, [136](#)
 pull, [136](#)
 dsy_gpio_deinit
 OTHER, [32](#)
 dsy_gpio_init
 OTHER, [33](#)
 DSY_GPIO_LAST
 UTILITY, [73](#)
 dsy_gpio_mode
 OTHER, [31](#)
 DSY_GPIO_MODE_ANALOG
 OTHER, [31](#)
 DSY_GPIO_MODE_INPUT
 OTHER, [31](#)
 DSY_GPIO_MODE_LAST
 OTHER, [31](#)
 DSY_GPIO_MODE_OUTPUT_OD
 OTHER, [31](#)
 DSY_GPIO_MODE_OUTPUT_PP
 OTHER, [31](#)
 DSY_GPIO_NOPULL
 OTHER, [32](#)
 dsy_gpio_pin, [136](#)
 pin, [136](#)
 port, [136](#)
 dsy_gpio_port
 UTILITY, [73](#)
 dsy_gpio_pull

- OTHER, [32](#)
- DSY_GPIO_PULLDOWN
 - OTHER, [32](#)
- DSY_GPIO_PULLUP
 - OTHER, [32](#)
- dsy_gpio_read
 - OTHER, [33](#)
- dsy_gpio_toggle
 - OTHER, [33](#)
- dsy_gpio_write
 - OTHER, [33](#)
- DSY_GPIOA
 - UTILITY, [73](#)
- DSY_GPIOB
 - UTILITY, [73](#)
- DSY_GPIOC
 - UTILITY, [73](#)
- DSY_GPIOD
 - UTILITY, [73](#)
- DSY_GPIOE
 - UTILITY, [73](#)
- DSY_GPIOF
 - UTILITY, [73](#)
- DSY_GPIOG
 - UTILITY, [73](#)
- DSY_GPIOH
 - UTILITY, [73](#)
- DSY_GPIOI
 - UTILITY, [73](#)
- DSY_GPIOJ
 - UTILITY, [73](#)
- DSY_GPIOK
 - UTILITY, [73](#)
- dsy_hal_map_get_i2c
 - UTILITY, [77](#)
- dsy_hal_map_get_pin
 - UTILITY, [77](#)
- dsy_hal_map_get_port
 - UTILITY, [77](#)
- dsy_i2c_handle, [137](#)
 - periph, [137](#)
 - pin_config, [137](#)
 - speed, [137](#)
- dsy_i2c_init
 - SERIAL, [23](#)
- dsy_i2c_periph
 - SERIAL, [23](#)
- DSY_I2C_PERIPH_1
 - SERIAL, [23](#)
- DSY_I2C_PERIPH_2
 - SERIAL, [23](#)
- DSY_I2C_PERIPH_3
 - SERIAL, [23](#)
- DSY_I2C_PERIPH_4
 - SERIAL, [23](#)
- dsy_i2c_pin
 - SERIAL, [23](#)
- DSY_I2C_PIN_LAST
 - SERIAL, [23](#)
- DSY_I2C_PIN_SCL
 - SERIAL, [23](#)
- DSY_I2C_PIN_SDA
 - SERIAL, [23](#)
- dsy_i2c_speed
 - SERIAL, [23](#)
- DSY_I2C_SPEED_100KHZ
 - SERIAL, [23](#)
- DSY_I2C_SPEED_1MHZ
 - SERIAL, [23](#)
- DSY_I2C_SPEED_400KHZ
 - SERIAL, [23](#)
- DSY_I2C_SPEED_LAST
 - SERIAL, [23](#)
- dsy_led_driver_color_by_name
 - LED, [63](#)
- dsy_led_driver_init
 - LED, [64](#)
- dsy_led_driver_set_led
 - LED, [64](#)
- dsy_led_driver_update
 - LED, [64](#)
- dsy_pin
 - UTILITY, [78](#)
- dsy_pin_cmp
 - UTILITY, [78](#)
- dsy_qspi_deinit
 - SERIAL, [25](#)
- dsy_qspi_device
 - SERIAL, [23](#)
- DSY_QSPI_DEVICE_IS25LP064A
 - SERIAL, [23](#)
- DSY_QSPI_DEVICE_IS25LP080D
 - SERIAL, [23](#)
- DSY_QSPI_DEVICE_LAST
 - SERIAL, [23](#)
- dsy_qspi_erase
 - SERIAL, [25](#)
- dsy_qspi_erasesector
 - SERIAL, [26](#)
- dsy_qspi_handle, [137](#)
 - device, [138](#)
 - mode, [138](#)
 - pin_config, [138](#)
- dsy_qspi_init
 - SERIAL, [26](#)
- dsy_qspi_mode
 - SERIAL, [24](#)
- DSY_QSPI_MODE_DSY_MEMORY_MAPPED
 - SERIAL, [24](#)
- DSY_QSPI_MODE_INDIRECT_POLLING
 - SERIAL, [24](#)
- DSY_QSPI_MODE_LAST
 - SERIAL, [24](#)
- dsy_qspi_pin
 - SERIAL, [24](#)
- DSY_QSPI_PIN_CLK

- SERIAL, [24](#)
- DSY_QSPI_PIN_IO0
 - SERIAL, [24](#)
- DSY_QSPI_PIN_IO1
 - SERIAL, [24](#)
- DSY_QSPI_PIN_IO2
 - SERIAL, [24](#)
- DSY_QSPI_PIN_IO3
 - SERIAL, [24](#)
- DSY_QSPI_PIN_LAST
 - SERIAL, [24](#)
- DSY_QSPI_PIN_NCS
 - SERIAL, [24](#)
- dsy_qspi_write
 - SERIAL, [26](#)
- dsy_qspi_writepage
 - SERIAL, [27](#)
- DSY_SAI_1
 - SERIAL, [21](#)
- DSY_SAI_2
 - SERIAL, [21](#)
- dsy_sai_handle, [138](#)
 - a_direction, [138](#)
 - b_direction, [138](#)
 - bitdepth, [138](#)
 - device, [139](#)
 - init, [139](#)
 - sai1_pin_config, [139](#)
 - sai2_pin_config, [139](#)
 - samplerate, [139](#)
 - sync_config, [139](#)
- dsy_sai_init
 - SERIAL, [27](#)
- dsy_sai_init_from_handle
 - SERIAL, [27](#)
- DSY_SAI_LAST
 - SERIAL, [21](#)
- dsy_sai_pin
 - SERIAL, [24](#)
- DSY_SAI_PIN_FS
 - SERIAL, [24](#)
- DSY_SAI_PIN_LAST
 - SERIAL, [24](#)
- DSY_SAI_PIN_MCLK
 - SERIAL, [24](#)
- DSY_SAI_PIN_SCK
 - SERIAL, [24](#)
- DSY_SAI_PIN_SIN
 - SERIAL, [24](#)
- DSY_SAI_PIN_SOUT
 - SERIAL, [24](#)
- DSY_SD_CardInfoTypeDef, [139](#)
 - BlockNbr, [140](#)
 - BlockSize, [140](#)
 - CardSpeed, [140](#)
 - CardType, [140](#)
 - CardVersion, [140](#)
 - Class, [140](#)
 - LogBlockNbr, [140](#)
 - LogBlockSize, [140](#)
 - RelCardAdd, [140](#)
- DSY_SDRAM_BSS
 - SDRAM, [65](#)
- DSY_SDRAM_DATA
 - SDRAM, [65](#)
- DSY_SDRAM_ERR
 - SDRAM, [65](#)
- dsy_sdram_handle, [140](#)
 - pin_config, [141](#)
 - state, [141](#)
- dsy_sdram_init
 - SDRAM, [66](#)
- DSY_SDRAM_OK
 - SDRAM, [65](#)
- dsy_sdram_pin
 - SDRAM, [66](#)
- DSY_SDRAM_PIN_LAST
 - SDRAM, [66](#)
- DSY_SDRAM_PIN_SDNWE
 - SDRAM, [66](#)
- dsy_sdram_state
 - SDRAM, [66](#)
- DSY_SDRAM_STATE_DISABLE
 - SDRAM, [66](#)
- DSY_SDRAM_STATE_ENABLE
 - SDRAM, [66](#)
- DSY_SDRAM_STATE_LAST
 - SDRAM, [66](#)
- dsy_sr_4021_handle, [141](#)
 - clk, [141](#)
 - cs, [141](#)
 - data, [141](#)
 - num_daisy chained, [142](#)
 - num_parallel, [142](#)
 - pin_config, [142](#)
 - states, [142](#)
- dsy_sr_4021_init
 - SHIFTREGISTER, [39](#)
- DSY_SR_4021_PIN_CLK
 - SHIFTREGISTER, [39](#)
- DSY_SR_4021_PIN_CS
 - SHIFTREGISTER, [39](#)
- DSY_SR_4021_PIN_DATA
 - SHIFTREGISTER, [39](#)
- DSY_SR_4021_PIN_DATA2
 - SHIFTREGISTER, [39](#)
- DSY_SR_4021_PIN_LAST
 - SHIFTREGISTER, [39](#)
- dsy_sr_4021_state
 - SHIFTREGISTER, [40](#)
- dsy_sr_4021_update
 - SHIFTREGISTER, [40](#)
- dsy_system_delay
 - SYSTEM, [36](#)
- dsy_system_getnow
 - SYSTEM, [36](#)

- dsy_system_init
 - SYSTEM, [36](#)
- dsy_system_jumpto
 - SYSTEM, [36](#)
- dsy_system_jumptoqspi
 - SYSTEM, [37](#)
- dsy_tim_delay_ms
 - OTHER, [34](#)
- dsy_tim_delay_tick
 - OTHER, [34](#)
- dsy_tim_delay_us
 - OTHER, [34](#)
- dsy_tim_get_ms
 - OTHER, [34](#)
- dsy_tim_get_tick
 - OTHER, [34](#)
- dsy_tim_get_us
 - OTHER, [35](#)
- dsy_tim_init
 - OTHER, [35](#)
- dsy_tim_start
 - OTHER, [35](#)
- DSY_WAVPLAYER_H
 - hid_wavplayer.h, [185](#)
- DTCM_MEM_SECTION
 - UTILITY, [72](#)
- DUAL_IN_FAST_PROG_CMD
 - FLASH, [45](#)
- DUAL_INOUT_FAST_READ_4_BYTE_ADDR_CMD
 - FLASH, [45](#)
- DUAL_INOUT_FAST_READ_CMD
 - FLASH, [45](#)
- DUAL_INOUT_FAST_READ_DTR_CMD
 - FLASH, [45](#)
- DUAL_OUT_FAST_READ_4_BYTE_ADDR_CMD
 - FLASH, [45](#)
- DUAL_OUT_FAST_READ_CMD
 - FLASH, [46](#)
- DUAL_OUT_FAST_READ_DTR_CMD
 - FLASH, [46](#)
- encoder
 - daisy::DaisyPatch, [121](#)
 - daisy::DaisyPetal, [127](#)
 - daisy::DaisyPod, [131](#)
- ENTER_4_BYTE_ADDR_MODE_CMD
 - FLASH, [46](#)
- ENTER_QUAD_CMD
 - FLASH, [46](#)
- EXIT_4_BYTE_ADDR_MODE_CMD
 - FLASH, [46](#)
- EXIT_QUAD_CMD
 - FLASH, [46](#)
- EXPONENTIAL
 - daisy::Parameter, [154](#)
- expression
 - daisy::DaisyPetal, [127](#)
- EXT_DUAL_IN_FAST_PROG_CMD
 - FLASH, [47](#)
- EXT_QUAD_IN_FAST_PROG_CMD
 - FLASH, [47](#)
- EXTERNAL, [18](#)
 - ChannelPressure, [18](#)
 - ControlChange, [18](#)
 - MessageLast, [18](#)
 - MidiMessageType, [18](#)
 - NoteOff, [18](#)
 - NoteOn, [18](#)
 - PitchBend, [18](#)
 - PolyphonicKeyPressure, [18](#)
 - ProgramChange, [18](#)
- Externals, [101](#)
- f2s16
 - BOARDS, [70](#)
- f2s24
 - BOARDS, [70](#)
- FallingEdge
 - daisy::Encoder, [143](#)
 - daisy::Switch, [166](#)
- FAST_READ_4_BYTE_ADDR_CMD
 - FLASH, [47](#)
- FAST_READ_CMD
 - FLASH, [47](#)
- FAST_READ_DTR_CMD
 - FLASH, [47](#)
- FEEDBACK, [17](#)
- ff_free
 - ffconf.h, [184](#)
- ff_malloc
 - ffconf.h, [184](#)
- ffconf.h
 - _CODE_PAGE, [180](#)
 - _FFCONF, [180](#)
 - _FS_EXFAT, [180](#)
 - _FS_LOCK, [180](#)
 - _FS_MINIMIZE, [180](#)
 - _FS_NOFSINFO, [180](#)
 - _FS_NORTC, [181](#)
 - _FS_READONLY, [181](#)
 - _FS_REENTRANT, [181](#)
 - _FS_RPATH, [181](#)
 - _FS_TIMEOUT, [181](#)
 - _FS_TINY, [181](#)
 - _LFN_UNICODE, [181](#)
 - _MAX_LFN, [181](#)
 - _MAX_SS, [182](#)
 - _MIN_SS, [182](#)
 - _MULTI_PARTITION, [182](#)
 - _NORTC_MDAY, [182](#)
 - _NORTC_MON, [182](#)
 - _NORTC_YEAR, [182](#)
 - _STRF_ENCODE, [182](#)
 - _STR_VOLUME_ID, [182](#)
 - _SYNC_t, [182](#)
 - _USE_CHMOD, [183](#)
 - _USE_EXPAND, [183](#)
 - _USE_FASTSEEK, [183](#)

- [_USE_FIND, 183](#)
- [_USE_FORWARD, 183](#)
- [_USE_LABEL, 183](#)
- [_USE_LFN, 183](#)
- [_USE_MKFS, 183](#)
- [_USE_STRFUNC, 183](#)
- [_USE_TRIM, 183](#)
- [_VOLUMES, 184](#)
- [_VOLUME_STRS, 184](#)
- [ff_free, 184](#)
- [ff_malloc, 184](#)
- FileFormat
 - [WAV_FormatTypeDef, 174](#)
- FileSize
 - [WAV_FormatTypeDef, 174](#)
- Fill
 - [daisy::OledDisplay, 152](#)
- FLASH, 41
 - [BLOCK_ERASE_32K_CMD, 44](#)
 - [CLEAR_FLAG_STATUS_REG_CMD, 44](#)
 - [DIE_ERASE_CMD, 45](#)
 - [DUAL_IN_FAST_PROG_CMD, 45](#)
 - [DUAL_INOUT_FAST_READ_4_BYTE_ADDR_CMD, 45](#)
 - [DUAL_INOUT_FAST_READ_CMD, 45](#)
 - [DUAL_INOUT_FAST_READ_DTR_CMD, 45](#)
 - [DUAL_OUT_FAST_READ_4_BYTE_ADDR_CMD, 45](#)
 - [DUAL_OUT_FAST_READ_CMD, 46](#)
 - [DUAL_OUT_FAST_READ_DTR_CMD, 46](#)
 - [ENTER_4_BYTE_ADDR_MODE_CMD, 46](#)
 - [ENTER_QUAD_CMD, 46](#)
 - [EXIT_4_BYTE_ADDR_MODE_CMD, 46](#)
 - [EXIT_QUAD_CMD, 46](#)
 - [EXT_DUAL_IN_FAST_PROG_CMD, 47](#)
 - [EXT_QUAD_IN_FAST_PROG_CMD, 47](#)
 - [FAST_READ_4_BYTE_ADDR_CMD, 47](#)
 - [FAST_READ_CMD, 47](#)
 - [FAST_READ_DTR_CMD, 47](#)
 - [IS25LP064A_EAR_HIGHEST_SE, 47](#)
 - [IS25LP064A_EAR_LOWEST_SEG, 47](#)
 - [IS25LP064A_EAR_SECOND_SEG, 48](#)
 - [IS25LP064A_EAR_THIRD_SEG, 48](#)
 - [IS25LP064A_EVCR_DTRP, 48](#)
 - [IS25LP064A_EVCR_DUAL, 48](#)
 - [IS25LP064A_EVCR_ODS, 48](#)
 - [IS25LP064A_EVCR_QUAD, 48](#)
 - [IS25LP064A_EVCR_RH, 48](#)
 - [IS25LP064A_FSR_ERERR, 48](#)
 - [IS25LP064A_FSR_ERSUS, 48](#)
 - [IS25LP064A_FSR_NBADDR, 48](#)
 - [IS25LP064A_FSR_PGERR, 48](#)
 - [IS25LP064A_FSR_PGSUS, 48](#)
 - [IS25LP064A_FSR_PRERR, 49](#)
 - [IS25LP064A_FSR_READY, 49](#)
 - [IS25LP064A_NVCR_DTRP, 49](#)
 - [IS25LP064A_NVCR_DUAL, 49](#)
 - [IS25LP064A_NVCR_NB_DUMMY, 49](#)
 - [IS25LP064A_NVCR_NBADDR, 49](#)
 - [IS25LP064A_NVCR_ODS, 49](#)
 - [IS25LP064A_NVCR_QUAB, 49](#)
 - [IS25LP064A_NVCR_RH, 49](#)
 - [IS25LP064A_NVCR_SEGMENT, 49](#)
 - [IS25LP064A_NVCR_XIP, 49](#)
 - [IS25LP064A_SR_QE, 49](#)
 - [IS25LP064A_SR_SRWREN, 50](#)
 - [IS25LP064A_SR_WIP, 50](#)
 - [IS25LP064A_SR_WREN, 50](#)
 - [IS25LP064A_VCR_NB_DUMMY, 50](#)
 - [IS25LP064A_VCR_WRAP, 50](#)
 - [IS25LP064A_VCR_XIP, 50](#)
 - [IS25LP080D_EAR_HIGHEST_SE, 50](#)
 - [IS25LP080D_EAR_LOWEST_SEG, 50](#)
 - [IS25LP080D_EAR_SECOND_SEG, 50](#)
 - [IS25LP080D_EAR_THIRD_SEG, 50](#)
 - [IS25LP080D_EVCR_DTRP, 50](#)
 - [IS25LP080D_EVCR_DUAL, 50](#)
 - [IS25LP080D_EVCR_ODS, 51](#)
 - [IS25LP080D_EVCR_QUAD, 51](#)
 - [IS25LP080D_EVCR_RH, 51](#)
 - [IS25LP080D_FSR_ERERR, 51](#)
 - [IS25LP080D_FSR_ERSUS, 51](#)
 - [IS25LP080D_FSR_NBADDR, 51](#)
 - [IS25LP080D_FSR_PGERR, 51](#)
 - [IS25LP080D_FSR_PGSUS, 51](#)
 - [IS25LP080D_FSR_PRERR, 51](#)
 - [IS25LP080D_FSR_READY, 51](#)
 - [IS25LP080D_NVCR_DTRP, 51](#)
 - [IS25LP080D_NVCR_DUAL, 51](#)
 - [IS25LP080D_NVCR_NB_DUMMY, 52](#)
 - [IS25LP080D_NVCR_NBADDR, 52](#)
 - [IS25LP080D_NVCR_ODS, 52](#)
 - [IS25LP080D_NVCR_QUAB, 52](#)
 - [IS25LP080D_NVCR_RH, 52](#)
 - [IS25LP080D_NVCR_SEGMENT, 52](#)
 - [IS25LP080D_NVCR_XIP, 52](#)
 - [IS25LP080D_SR_QE, 52](#)
 - [IS25LP080D_SR_SRWREN, 52](#)
 - [IS25LP080D_SR_WIP, 52](#)
 - [IS25LP080D_SR_WREN, 52](#)
 - [IS25LP080D_VCR_NB_DUMMY, 52](#)
 - [IS25LP080D_VCR_WRAP, 53](#)
 - [IS25LP080D_VCR_XIP, 53](#)
 - [MULTIPLE_IO_READ_ID_CMD, 53](#)
 - [PAGE_PROG_4_BYTE_ADDR_CMD, 53](#)
 - [PAGE_PROG_CMD, 53](#)
 - [PROG_ERASE_RESUME_CMD, 53](#)
 - [PROG_ERASE_SUSPEND_CMD, 53](#)
 - [PROG_OTP_ARRAY_CMD, 54](#)
 - [QUAD_IN_FAST_PROG_4_BYTE_ADDR_CMD, 54](#)
 - [QUAD_IN_FAST_PROG_CMD, 54](#)
 - [QUAD_INOUT_FAST_READ_4_BYTE_ADDR_CMD, 54](#)
 - [QUAD_INOUT_FAST_READ_CMD, 54](#)
 - [QUAD_INOUT_FAST_READ_DTR_CMD, 54](#)

- QUAD_OUT_FAST_READ_4_BYTE_ADDR_CMD, 55
- QUAD_OUT_FAST_READ_CMD, 55
- QUAD_OUT_FAST_READ_DTR_CMD, 55
- READ_4_BYTE_ADDR_CMD, 55
- READ_CMD, 55
- READ_ENHANCED_VOL_CFG_REG_CMD, 55
- READ_EXT_ADDR_REG_CMD, 56
- READ_FLAG_STATUS_REG_CMD, 56
- READ_ID_CMD, 56
- READ_ID_CMD2, 56
- READ_LOCK_REG_CMD, 56
- READ_NONVOL_CFG_REG_CMD, 56
- READ_OTP_ARRAY_CMD, 57
- READ_READ_PARAM_REG_CMD, 57
- READ_SERIAL_FLASH_DISCO_PARAM_CMD, 57
- READ_STATUS_REG_CMD, 57
- RESET_ENABLE_CMD, 57
- RESET_MEMORY_CMD, 57
- SECTOR_ERASE_4_BYTE_ADDR_CMD, 58
- SECTOR_ERASE_CMD, 58
- SUBSECTOR_ERASE_4_BYTE_ADDR_CMD, 58
- SUBSECTOR_ERASE_CMD, 58
- SUBSECTOR_ERASE_QPI_CMD, 58
- WRITE_DISABLE_CMD, 58
- WRITE_ENABLE_CMD, 59
- WRITE_ENHANCED_VOL_CFG_REG_CMD, 59
- WRITE_EXT_ADDR_REG_CMD, 59
- WRITE_LOCK_REG_CMD, 59
- WRITE_NONVOL_CFG_REG_CMD, 59
- WRITE_READ_PARAM_REG_CMD, 59
- WRITE_STATUS_REG_CMD, 60
- Flush
 - daisy::RingBuffer< T, 0 >, 160
 - daisy::RingBuffer< T, size >, 157
- FlushRx
 - daisy::UartHandler, 168
- Font_11x18
 - UTILITY, 78
- Font_16x26
 - UTILITY, 78
- Font_6x8
 - UTILITY, 78
- Font_7x10
 - UTILITY, 78
- FontDef, 143
 - data, 144
 - FontHeight, 144
 - FontWidth, 144
- FontHeight
 - FontDef, 144
- FontWidth
 - FontDef, 144
- footswitch_led
 - daisy::DaisyPetal, 127
- FOOTSWITCH_LED_1
 - daisy::DaisyPetal, 123
- FOOTSWITCH_LED_2
 - daisy::DaisyPetal, 123
- FOOTSWITCH_LED_3
 - daisy::DaisyPetal, 123
- FOOTSWITCH_LED_4
 - daisy::DaisyPetal, 123
- FOOTSWITCH_LED_LAST
 - daisy::DaisyPetal, 123
- FootswitchLed
 - daisy::DaisyPetal, 123
- FS_BOTH
 - UsbHandle, 171
- FS_Desc
 - USBD_DESC_Exported_Variables, 99
- FS_EXTERNAL
 - UsbHandle, 171
- FS_INTERNAL
 - UsbHandle, 171
- gate_in
 - daisy::daisy_field, 117
- GATE_IN_LAST
 - daisy::DaisyPatch, 119
- gate_input
 - daisy::DaisyPatch, 121
- gate_out
 - daisy::daisy_field, 117
- gate_output
 - daisy::DaisyPatch, 122
- Gateln
 - daisy::Gateln, 144
- GatelnInput
 - daisy::DaisyPatch, 119
- Get
 - daisy::AdcHandle, 109
- GetCtrlValue
 - daisy::DaisyPatch, 120
- GetCurrentFile
 - daisy::WavPlayer, 176
- GetExpression
 - daisy::DaisyPetal, 125
- GetFloat
 - daisy::AdcHandle, 110
- GetKnobValue
 - daisy::DaisyPetal, 125
 - daisy::DaisyPod, 130
- GetLooping
 - daisy::WavPlayer, 176
- GetMux
 - daisy::AdcHandle, 110
- GetMuxFloat
 - daisy::AdcHandle, 110
- GetMuxPtr
 - daisy::AdcHandle, 111
- GetNumberFiles
 - daisy::WavPlayer, 176
- GetPin
 - daisy::DaisySeed, 133
- GetPtr

- daisy::AdcHandle, [111](#)
- GOLD
 - daisy::Color, [115](#)
- GREEN
 - daisy::Color, [115](#)
- Green
 - daisy::Color, [116](#)
- green
 - color, [114](#)
- HasEvents
 - daisy::MidiHandler, [149](#)
- hi2c1
 - UTILITY, [78](#)
- hi2c2
 - UTILITY, [78](#)
- hi2c3
 - UTILITY, [79](#)
- hi2c4
 - UTILITY, [79](#)
- hid_wavplayer.h
 - DSY_WAVPLAYER_H, [185](#)
 - WAV_FILENAME_MAX, [185](#)
- HS_Desc
 - USBD_DESC_Exported_Variables, [99](#)
- HUMAN_INTERFACE, [12](#)
- i2c1_handle
 - daisy::DaisySeed, [134](#)
- i2c2_handle
 - daisy::DaisySeed, [134](#)
- ImmediateRead
 - daisy::RingBuffer< T, 0 >, [160](#)
 - daisy::RingBuffer< T, size >, [157](#)
- Increment
 - daisy::Encoder, [143](#)
- Init
 - daisy::AdcHandle, [111](#)
 - daisy::AnalogControl, [113](#)
 - daisy::Color, [116](#)
 - daisy::DaisyPatch, [121](#)
 - daisy::DaisyPetal, [126](#)
 - daisy::DaisyPod, [130](#)
 - daisy::DaisySeed, [133](#)
 - daisy::Encoder, [143](#)
 - daisy::GateIn, [145](#)
 - daisy::Led, [145](#)
 - daisy::MidiHandler, [149](#)
 - daisy::OledDisplay, [152](#)
 - daisy::Parameter, [154](#)
 - daisy::RgbLed, [155](#)
 - daisy::RingBuffer< T, 0 >, [160](#)
 - daisy::RingBuffer< T, size >, [157](#)
 - daisy::SdmmcHandler, [162](#)
 - daisy::SpiHandle, [164](#)
 - daisy::Switch, [166](#)
 - daisy::UartHandler, [168](#)
 - daisy::WavPlayer, [176](#)
 - ShiftRegister595, [163](#)
 - UsbHandle, [171](#)
- init
 - dsy_sai_handle, [139](#)
- InitBipolarCv
 - daisy::AnalogControl, [113](#)
- InitMux
 - daisy::AdcChannelConfig, [108](#)
- InitSingle
 - daisy::AdcChannelConfig, [108](#)
- INPUT_MODE_NONE
 - daisy::MidiHandler, [149](#)
- INPUT_MODE_UART1
 - daisy::MidiHandler, [149](#)
- INPUT_MODE_USB_EXT
 - daisy::MidiHandler, [149](#)
- INPUT_MODE_USB_INT
 - daisy::MidiHandler, [149](#)
- IS25LP064A_EAR_HIGHEST_SE
 - FLASH, [47](#)
- IS25LP064A_EAR_LOWEST_SEG
 - FLASH, [47](#)
- IS25LP064A_EAR_SECOND_SEG
 - FLASH, [48](#)
- IS25LP064A_EAR_THIRD_SEG
 - FLASH, [48](#)
- IS25LP064A_EVCR_DTRP
 - FLASH, [48](#)
- IS25LP064A_EVCR_DUAL
 - FLASH, [48](#)
- IS25LP064A_EVCR_ODS
 - FLASH, [48](#)
- IS25LP064A_EVCR_QUAD
 - FLASH, [48](#)
- IS25LP064A_EVCR_RH
 - FLASH, [48](#)
- IS25LP064A_FSR_ERERR
 - FLASH, [48](#)
- IS25LP064A_FSR_ERSUS
 - FLASH, [48](#)
- IS25LP064A_FSR_NBADDR
 - FLASH, [48](#)
- IS25LP064A_FSR_PGERR
 - FLASH, [48](#)
- IS25LP064A_FSR_PGSUS
 - FLASH, [48](#)
- IS25LP064A_FSR_PRERR
 - FLASH, [49](#)
- IS25LP064A_FSR_READY
 - FLASH, [49](#)
- IS25LP064A_NVCR_DTRP
 - FLASH, [49](#)
- IS25LP064A_NVCR_DUAL
 - FLASH, [49](#)
- IS25LP064A_NVCR_NB_DUMMY
 - FLASH, [49](#)
- IS25LP064A_NVCR_NBADDR
 - FLASH, [49](#)
- IS25LP064A_NVCR_ODS

- FLASH, [49](#)
- IS25LP064A_NVCR_QUAB
 - FLASH, [49](#)
- IS25LP064A_NVCR_RH
 - FLASH, [49](#)
- IS25LP064A_NVCR_SEGMENT
 - FLASH, [49](#)
- IS25LP064A_NVCR_XIP
 - FLASH, [49](#)
- IS25LP064A_SR_QE
 - FLASH, [49](#)
- IS25LP064A_SR_SRWREN
 - FLASH, [50](#)
- IS25LP064A_SR_WIP
 - FLASH, [50](#)
- IS25LP064A_SR_WREN
 - FLASH, [50](#)
- IS25LP064A_VCR_NB_DUMMY
 - FLASH, [50](#)
- IS25LP064A_VCR_WRAP
 - FLASH, [50](#)
- IS25LP064A_VCR_XIP
 - FLASH, [50](#)
- IS25LP080D_EAR_HIGHEST_SE
 - FLASH, [50](#)
- IS25LP080D_EAR_LOWEST_SEG
 - FLASH, [50](#)
- IS25LP080D_EAR_SECOND_SEG
 - FLASH, [50](#)
- IS25LP080D_EAR_THIRD_SEG
 - FLASH, [50](#)
- IS25LP080D_EVCR_DTRP
 - FLASH, [50](#)
- IS25LP080D_EVCR_DUAL
 - FLASH, [50](#)
- IS25LP080D_EVCR_ODS
 - FLASH, [51](#)
- IS25LP080D_EVCR_QUAD
 - FLASH, [51](#)
- IS25LP080D_EVCR_RH
 - FLASH, [51](#)
- IS25LP080D_FSR_ERERR
 - FLASH, [51](#)
- IS25LP080D_FSR_ERSUS
 - FLASH, [51](#)
- IS25LP080D_FSR_NBADDR
 - FLASH, [51](#)
- IS25LP080D_FSR_PGERR
 - FLASH, [51](#)
- IS25LP080D_FSR_PGSUS
 - FLASH, [51](#)
- IS25LP080D_FSR_PRERR
 - FLASH, [51](#)
- IS25LP080D_FSR_READY
 - FLASH, [51](#)
- IS25LP080D_NVCR_DTRP
 - FLASH, [51](#)
- IS25LP080D_NVCR_DUAL
 - FLASH, [51](#)
- IS25LP080D_NVCR_NB_DUMMY
 - FLASH, [52](#)
- IS25LP080D_NVCR_NBADDR
 - FLASH, [52](#)
- IS25LP080D_NVCR_ODS
 - FLASH, [52](#)
- IS25LP080D_NVCR_QUAB
 - FLASH, [52](#)
- IS25LP080D_NVCR_RH
 - FLASH, [52](#)
- IS25LP080D_NVCR_SEGMENT
 - FLASH, [52](#)
- IS25LP080D_NVCR_XIP
 - FLASH, [52](#)
- IS25LP080D_SR_QE
 - FLASH, [52](#)
- IS25LP080D_SR_SRWREN
 - FLASH, [52](#)
- IS25LP080D_SR_WIP
 - FLASH, [52](#)
- IS25LP080D_SR_WREN
 - FLASH, [52](#)
- IS25LP080D_VCR_NB_DUMMY
 - FLASH, [52](#)
- IS25LP080D_VCR_WRAP
 - FLASH, [53](#)
- IS25LP080D_VCR_XIP
 - FLASH, [53](#)
- keyboard_sr
 - daisy::daisy_field, [117](#)
- Knob
 - daisy::DaisyPetal, [123](#)
 - daisy::DaisyPod, [129](#)
- knob
 - daisy::DaisyPetal, [127](#)
- knob1
 - daisy::DaisyPod, [131](#)
- knob2
 - daisy::DaisyPod, [131](#)
- KNOB_1
 - BOARDS, [68](#)
 - daisy::DaisyPetal, [123](#)
- KNOB_2
 - BOARDS, [68](#)
 - daisy::DaisyPetal, [123](#)
 - daisy::DaisyPod, [129](#)
- KNOB_3
 - BOARDS, [68](#)
 - daisy::DaisyPetal, [123](#)
- KNOB_4
 - BOARDS, [68](#)
 - daisy::DaisyPetal, [123](#)
- KNOB_5
 - BOARDS, [68](#)
 - daisy::DaisyPetal, [123](#)
- KNOB_6
 - BOARDS, [68](#)

- daisy::DaisyPetal, 123
- KNOB_7
 - BOARDS, 68
- KNOB_8
 - BOARDS, 68
- KNOB_LAST
 - BOARDS, 68
 - daisy::DaisyPetal, 123
 - daisy::DaisyPod, 129
- knobs
 - daisy::daisy_field, 118
 - daisy::DaisyPod, 131
- kUartMaxBufferSize
 - SERIAL, 28
- I
 - codec_frame_t, 114
- LAST
 - daisy::Color, 115
 - daisy::Parameter, 154
- LED, 63
 - dsy_led_driver_color_by_name, 63
 - dsy_led_driver_init, 64
 - dsy_led_driver_set_led, 64
 - dsy_led_driver_update, 64
 - LED_COLOR_BLUE, 63
 - LED_COLOR_CYAN, 63
 - LED_COLOR_GOLD, 63
 - LED_COLOR_GREEN, 63
 - LED_COLOR_LAST, 63
 - LED_COLOR_OFF, 63
 - LED_COLOR_PURPLE, 63
 - LED_COLOR_RED, 63
 - LED_COLOR_WHITE, 63
- led1
 - daisy::DaisyPod, 131
- led2
 - daisy::DaisyPod, 131
- LED_COLOR_BLUE
 - LED, 63
- LED_COLOR_CYAN
 - LED, 63
- LED_COLOR_GOLD
 - LED, 63
- LED_COLOR_GREEN
 - LED, 63
- LED_COLOR_LAST
 - LED, 63
- LED_COLOR_OFF
 - LED, 63
- LED_COLOR_PURPLE
 - LED, 63
- LED_COLOR_RED
 - LED, 63
- LED_COLOR_WHITE
 - LED, 63
- LED_KEY_A1
 - BOARDS, 68
- LED_KEY_A2
 - BOARDS, 68
- LED_KEY_A3
 - BOARDS, 68
- LED_KEY_A4
 - BOARDS, 68
- LED_KEY_A5
 - BOARDS, 68
- LED_KEY_A6
 - BOARDS, 68
- LED_KEY_A7
 - BOARDS, 68
- LED_KEY_A8
 - BOARDS, 68
- LED_KEY_B1
 - BOARDS, 69
- LED_KEY_B2
 - BOARDS, 69
- LED_KEY_B3
 - BOARDS, 69
- LED_KEY_B4
 - BOARDS, 69
- LED_KEY_B5
 - BOARDS, 69
- LED_KEY_B6
 - BOARDS, 69
- LED_KEY_B7
 - BOARDS, 69
- LED_KEY_B8
 - BOARDS, 69
- LED_KNOB_1
 - BOARDS, 69
- LED_KNOB_2
 - BOARDS, 69
- LED_KNOB_3
 - BOARDS, 69
- LED_KNOB_4
 - BOARDS, 69
- LED_KNOB_5
 - BOARDS, 69
- LED_KNOB_6
 - BOARDS, 69
- LED_KNOB_7
 - BOARDS, 69
- LED_KNOB_8
 - BOARDS, 69
- LED_LAST
 - BOARDS, 69
- LED_SW_1
 - BOARDS, 69
- LED_SW_2
 - BOARDS, 69
- LIBDAISY, 11
- LINEAR
 - daisy::Parameter, 154
- Listen
 - daisy::MidiHandler, 150
- LOGARITHMIC
 - daisy::Parameter, 154

- LogBlockNbr
 - DSY_SD_CardInfoTypeDef, [140](#)
- LogBlockSize
 - DSY_SD_CardInfoTypeDef, [140](#)
- MessageLast
 - EXTERNAL, [18](#)
- midi
 - daisy::DaisyPatch, [122](#)
- MidInputMode
 - daisy::MidiHandler, [149](#)
- MidiMessageType
 - EXTERNAL, [18](#)
- MidiOutputMode
 - daisy::MidiHandler, [149](#)
- mode
 - dsy_dac_handle, [135](#)
 - dsy_gpio, [136](#)
 - dsy_qspi_handle, [138](#)
- MSD_ERROR
 - UTILITY, [72](#)
- MSD_ERROR_SD_NOT_PRESENT
 - UTILITY, [72](#)
- MSD_OK
 - UTILITY, [72](#)
- MULTIPLE_IO_READ_ID_CMD
 - FLASH, [53](#)
- mux_channels_
 - daisy::AdcChannelConfig, [108](#)
- mux_pin_
 - daisy::AdcChannelConfig, [108](#)
- MUX_SEL_0
 - daisy::AdcChannelConfig, [107](#)
- MUX_SEL_1
 - daisy::AdcChannelConfig, [107](#)
- MUX_SEL_2
 - daisy::AdcChannelConfig, [107](#)
- MUX_SEL_LAST
 - daisy::AdcChannelConfig, [107](#)
- MuxPin
 - daisy::AdcChannelConfig, [107](#)
- name
 - daisy::WavFileInfo, [175](#)
- NbrChannels
 - WAV_FormatTypeDef, [174](#)
- note
 - daisy::NoteOnEvent, [150](#)
- NoteOff
 - EXTERNAL, [18](#)
- NoteOn
 - EXTERNAL, [18](#)
- num_daisy chained
 - dsy_sr_4021_handle, [142](#)
- num_parallel
 - dsy_sr_4021_handle, [142](#)
- NUM_PINS
 - daisy::OledDisplay, [151](#)
 - ShiftRegister595, [163](#)
- OFF
 - daisy::Color, [115](#)
- Open
 - daisy::WavPlayer, [176](#)
- OTHER, [31](#)
 - dsy_gpio_deinit, [32](#)
 - dsy_gpio_init, [33](#)
 - dsy_gpio_mode, [31](#)
 - DSY_GPIO_MODE_ANALOG, [31](#)
 - DSY_GPIO_MODE_INPUT, [31](#)
 - DSY_GPIO_MODE_LAST, [31](#)
 - DSY_GPIO_MODE_OUTPUT_OD, [31](#)
 - DSY_GPIO_MODE_OUTPUT_PP, [31](#)
 - DSY_GPIO_NOPULL, [32](#)
 - dsy_gpio_pull, [32](#)
 - DSY_GPIO_PULLDOWN, [32](#)
 - DSY_GPIO_PULLUP, [32](#)
 - dsy_gpio_read, [33](#)
 - dsy_gpio_toggle, [33](#)
 - dsy_gpio_write, [33](#)
 - dsy_tim_delay_ms, [34](#)
 - dsy_tim_delay_tick, [34](#)
 - dsy_tim_delay_us, [34](#)
 - dsy_tim_get_ms, [34](#)
 - dsy_tim_get_tick, [34](#)
 - dsy_tim_get_us, [35](#)
 - dsy_tim_init, [35](#)
 - dsy_tim_start, [35](#)
 - SDMMC_BITS_1, [32](#)
 - SDMMC_BITS_4, [32](#)
 - SDMMC_MODE_FATFS, [32](#)
 - SDMMC_SPEED_12MHZ, [32](#)
 - SDMMC_SPEED_400KHZ, [32](#)
 - SdmmcBitWidth, [32](#)
 - SdmmcMode, [32](#)
 - SdmmcSpeed, [32](#)
- OUTPUT_MODE_NONE
 - daisy::MidiHandler, [149](#)
- OUTPUT_MODE_UART1
 - daisy::MidiHandler, [149](#)
- OUTPUT_MODE_USB_EXT
 - daisy::MidiHandler, [149](#)
- OUTPUT_MODE_USB_INT
 - daisy::MidiHandler, [149](#)
- OverSampling
 - daisy::AdcHandle, [109](#)
- Overwrite
 - daisy::RingBuffer< T, 0 >, [160](#)
 - daisy::RingBuffer< T, size >, [157](#), [158](#)
- OVS_1024
 - daisy::AdcHandle, [109](#)
- OVS_128
 - daisy::AdcHandle, [109](#)
- OVS_16
 - daisy::AdcHandle, [109](#)
- OVS_256
 - daisy::AdcHandle, [109](#)
- OVS_32

- daisy::AdcHandle, 109
- OVS_4
 - daisy::AdcHandle, 109
- OVS_512
 - daisy::AdcHandle, 109
- OVS_64
 - daisy::AdcHandle, 109
- OVS_8
 - daisy::AdcHandle, 109
- OVS_LAST
 - daisy::AdcHandle, 109
- OVS_NONE
 - daisy::AdcHandle, 109
- PAGE_PROG_4_BYTE_ADDR_CMD
 - FLASH, 53
- PAGE_PROG_CMD
 - FLASH, 53
- Parameter
 - daisy::Parameter, 154
- Parse
 - daisy::MidiHandler, 150
- periph
 - dsy_i2c_handle, 137
- PERIPHERAL, 19
- pin
 - dsy_gpio, 136
 - dsy_gpio_pin, 136
- pin_
 - daisy::AdcChannelConfig, 108
- PIN_CLK
 - ShiftRegister595, 163
- pin_config
 - dsy_dac_handle, 135
 - dsy_i2c_handle, 137
 - dsy_qspi_handle, 138
 - dsy_sdram_handle, 141
 - dsy_sr_4021_handle, 142
- PIN_DATA
 - ShiftRegister595, 163
- Pins
 - daisy::OledDisplay, 151
 - ShiftRegister595, 163
- PitchBend
 - EXTERNAL, 18
- Polarity
 - daisy::Switch, 165
- POLARITY_INVERTED
 - daisy::Switch, 165
- POLARITY_NORMAL
 - daisy::Switch, 165
- PollReceive
 - daisy::UartHandler, 168
- PollTx
 - daisy::UartHandler, 169
- PolyphonicKeyPressure
 - EXTERNAL, 18
- PopEvent
 - daisy::MidiHandler, 150
- PopRx
 - daisy::UartHandler, 169
- port
 - dsy_gpio_pin, 136
- Prepare
 - daisy::WavPlayer, 178
- PresetColor
 - daisy::Color, 115
- Pressed
 - daisy::Encoder, 143
 - daisy::Switch, 167
- Process
 - daisy::AnalogControl, 113
 - daisy::Parameter, 155
- PROG_ERASE_RESUME_CMD
 - FLASH, 53
- PROG_ERASE_SUSPEND_CMD
 - FLASH, 53
- PROG_OTP_ARRAY_CMD
 - FLASH, 54
- ProgramChange
 - EXTERNAL, 18
- Pull
 - daisy::Switch, 165
- pull
 - dsy_gpio, 136
- PULL_DOWN
 - daisy::Switch, 165
- PULL_NONE
 - daisy::Switch, 165
- PULL_UP
 - daisy::Switch, 165
- PURPLE
 - daisy::Color, 115
- qspi_handle
 - daisy::DaisySeed, 134
- QUAD_IN_FAST_PROG_4_BYTE_ADDR_CMD
 - FLASH, 54
- QUAD_IN_FAST_PROG_CMD
 - FLASH, 54
- QUAD_INOUT_FAST_READ_4_BYTE_ADDR_CMD
 - FLASH, 54
- QUAD_INOUT_FAST_READ_CMD
 - FLASH, 54
- QUAD_INOUT_FAST_READ_DTR_CMD
 - FLASH, 54
- QUAD_OUT_FAST_READ_4_BYTE_ADDR_CMD
 - FLASH, 55
- QUAD_OUT_FAST_READ_CMD
 - FLASH, 55
- QUAD_OUT_FAST_READ_DTR_CMD
 - FLASH, 55
- r
 - codec_frame_t, 114
- raw_data
 - daisy::WavFileInfo, 175
- Read

- daisy::RingBuffer< T, 0 >, [161](#)
- daisy::RingBuffer< T, size >, [158](#)
- READ_4_BYTE_ADDR_CMD
 - FLASH, [55](#)
- READ_CMD
 - FLASH, [55](#)
- READ_ENHANCED_VOL_CFG_REG_CMD
 - FLASH, [55](#)
- READ_EXT_ADDR_REG_CMD
 - FLASH, [56](#)
- READ_FLAG_STATUS_REG_CMD
 - FLASH, [56](#)
- READ_ID_CMD
 - FLASH, [56](#)
- READ_ID_CMD2
 - FLASH, [56](#)
- READ_LOCK_REG_CMD
 - FLASH, [56](#)
- READ_NONVOL_CFG_REG_CMD
 - FLASH, [56](#)
- READ_OTP_ARRAY_CMD
 - FLASH, [57](#)
- READ_READ_PARAM_REG_CMD
 - FLASH, [57](#)
- READ_SERIAL_FLASH_DISCO_PARAM_CMD
 - FLASH, [57](#)
- READ_STATUS_REG_CMD
 - FLASH, [57](#)
- Readable
 - daisy::UartHandler, [169](#)
- readable
 - daisy::RingBuffer< T, 0 >, [161](#)
 - daisy::RingBuffer< T, size >, [158](#)
- ReceiveCallback
 - UsbHandle, [170](#)
- RED
 - daisy::Color, [115](#)
- Red
 - daisy::Color, [116](#)
- red
 - color, [114](#)
- RelCardAdd
 - DSY_SD_CardInfoTypeDef, [140](#)
- RESET
 - daisy::OledDisplay, [151](#)
- RESET_ENABLE_CMD
 - FLASH, [57](#)
- RESET_MEMORY_CMD
 - FLASH, [57](#)
- Restart
 - daisy::WavPlayer, [178](#)
- ring_led
 - daisy::DaisyPetal, [127](#)
- RING_LED_1
 - daisy::DaisyPetal, [124](#)
- RING_LED_2
 - daisy::DaisyPetal, [124](#)
- RING_LED_3
 - daisy::DaisyPetal, [124](#)
- RING_LED_4
 - daisy::DaisyPetal, [124](#)
- RING_LED_5
 - daisy::DaisyPetal, [124](#)
- RING_LED_6
 - daisy::DaisyPetal, [124](#)
- RING_LED_7
 - daisy::DaisyPetal, [124](#)
- RING_LED_8
 - daisy::DaisyPetal, [124](#)
- RING_LED_LAST
 - daisy::DaisyPetal, [124](#)
- RingLed
 - daisy::DaisyPetal, [124](#)
- RisingEdge
 - daisy::Encoder, [143](#)
 - daisy::Switch, [167](#)
- RxActive
 - daisy::UartHandler, [169](#)
- s162f
 - BOARDS, [70](#)
- s242f
 - BOARDS, [70](#)
- sa_audio_callback
 - CODEC, [61](#)
- sai
 - dsy_audio_handle, [135](#)
- sai1_pin_config
 - dsy_sai_handle, [139](#)
- sai2_pin_config
 - dsy_sai_handle, [139](#)
- sai_handle
 - daisy::DaisySeed, [134](#)
- SampleRate
 - WAV_FormatTypeDef, [174](#)
- samplerate
 - dsy_sai_handle, [139](#)
- SD_DATATIMEOUT
 - UTILITY, [72](#)
- SD_NOT_PRESENT
 - UTILITY, [72](#)
- SD_PRESENT
 - UTILITY, [72](#)
- SD_TRANSFER_BUSY
 - UTILITY, [73](#)
- SD_TRANSFER_OK
 - UTILITY, [73](#)
- SDMMC_BITS_1
 - OTHER, [32](#)
- SDMMC_BITS_4
 - OTHER, [32](#)
- SDMMC_MODE_FATFS
 - OTHER, [32](#)
- SDMMC_SPEED_12MHZ
 - OTHER, [32](#)
- SDMMC_SPEED_400KHZ
 - OTHER, [32](#)

SdmmcBitWidth
 OTHER, 32
 SdmmcMode
 OTHER, 32
 SdmmcSpeed
 OTHER, 32
 SDRAM, 65
 DSY_SDRAM_BSS, 65
 DSY_SDRAM_DATA, 65
 DSY_SDRAM_ERR, 65
 dsy_sdram_init, 66
 DSY_SDRAM_OK, 65
 dsy_sdram_pin, 66
 DSY_SDRAM_PIN_LAST, 66
 DSY_SDRAM_PIN_SDNWE, 66
 dsy_sdram_state, 66
 DSY_SDRAM_STATE_DISABLE, 66
 DSY_SDRAM_STATE_ENABLE, 66
 DSY_SDRAM_STATE_LAST, 66
 sdram_handle
 daisy::DaisySeed, 134
 SECTOR_ERASE_4_BYTE_ADDR_CMD
 FLASH, 58
 SECTOR_ERASE_CMD
 FLASH, 58
 seed
 daisy::daisy_field, 118
 daisy::DaisyPatch, 122
 daisy::DaisyPetal, 127
 daisy::DaisyPod, 132
 SERIAL, 20
 dsy_audio_bitdepth, 21
 DSY_AUDIO_BITDEPTH_16, 21
 DSY_AUDIO_BITDEPTH_24, 21
 DSY_AUDIO_BITDEPTH_LAST, 21
 dsy_audio_device, 21
 DSY_AUDIO_DEVICE_AK4556, 21
 DSY_AUDIO_DEVICE_LAST, 21
 DSY_AUDIO_DEVICE_PCM3060, 21
 DSY_AUDIO_DEVICE_WM8731, 21
 dsy_audio_dir, 21
 DSY_AUDIO_INIT_BOTH, 22
 DSY_AUDIO_INIT_LAST, 22
 DSY_AUDIO_INIT_NONE, 22
 DSY_AUDIO_INIT_SAI1, 22
 DSY_AUDIO_INIT_SAI2, 22
 DSY_AUDIO_NONE, 21
 DSY_AUDIO_RX, 22
 dsy_audio_sai, 22
 dsy_audio_samplerate, 22
 DSY_AUDIO_SAMPLERATE_32K, 22
 DSY_AUDIO_SAMPLERATE_48K, 22
 DSY_AUDIO_SAMPLERATE_96K, 22
 DSY_AUDIO_SAMPLERATE_LAST, 22
 dsy_audio_sync, 22
 DSY_AUDIO_SYNC_LAST, 22
 DSY_AUDIO_SYNC_MASTER, 22
 DSY_AUDIO_SYNC_SLAVE, 22
 DSY_AUDIO_TX, 22
 dsy_i2c_init, 25
 dsy_i2c_periph, 22
 DSY_I2C_PERIPH_1, 23
 DSY_I2C_PERIPH_2, 23
 DSY_I2C_PERIPH_3, 23
 DSY_I2C_PERIPH_4, 23
 dsy_i2c_pin, 23
 DSY_I2C_PIN_LAST, 23
 DSY_I2C_PIN_SCL, 23
 DSY_I2C_PIN_SDA, 23
 dsy_i2c_speed, 23
 DSY_I2C_SPEED_100KHZ, 23
 DSY_I2C_SPEED_1MHZ, 23
 DSY_I2C_SPEED_400KHZ, 23
 DSY_I2C_SPEED_LAST, 23
 dsy_qspi_deinit, 25
 dsy_qspi_device, 23
 DSY_QSPI_DEVICE_IS25LP064A, 23
 DSY_QSPI_DEVICE_IS25LP080D, 23
 DSY_QSPI_DEVICE_LAST, 23
 dsy_qspi_erase, 25
 dsy_qspi_erasesector, 26
 dsy_qspi_init, 26
 dsy_qspi_mode, 24
 DSY_QSPI_MODE_DSY_MEMORY_MAPPED, 24
 DSY_QSPI_MODE_INDIRECT_POLLING, 24
 DSY_QSPI_MODE_LAST, 24
 dsy_qspi_pin, 24
 DSY_QSPI_PIN_CLK, 24
 DSY_QSPI_PIN_IO0, 24
 DSY_QSPI_PIN_IO1, 24
 DSY_QSPI_PIN_IO2, 24
 DSY_QSPI_PIN_IO3, 24
 DSY_QSPI_PIN_LAST, 24
 DSY_QSPI_PIN_NCS, 24
 dsy_qspi_write, 26
 dsy_qspi_writepage, 27
 DSY_SAI_1, 21
 DSY_SAI_2, 21
 dsy_sai_init, 27
 dsy_sai_init_from_handle, 27
 DSY_SAI_LAST, 21
 dsy_sai_pin, 24
 DSY_SAI_PIN_FS, 24
 DSY_SAI_PIN_LAST, 24
 DSY_SAI_PIN_MCLK, 24
 DSY_SAI_PIN_SCK, 24
 DSY_SAI_PIN_SIN, 24
 DSY_SAI_PIN_SOUT, 24
 kUartMaxBufferSize, 28
 SPI_PERIPH_1, 25
 SPI_PERIPH_3, 25
 SPI_PERIPH_6, 25
 SPI_PIN_CS, 25
 SPI_PIN_MISO, 25
 SPI_PIN_MOSI, 25

- SPI_PIN_SCK, [25](#)
- SpiPeriph, [24](#)
- SpiPin, [25](#)
- Set
 - daisy::Led, [147](#)
 - daisy::RgbLed, [156](#)
 - ShiftRegister595, [163](#)
- SetAudioBlockSize
 - daisy::DaisyPatch, [121](#)
 - daisy::DaisyPetal, [126](#)
 - daisy::DaisyPod, [130](#)
 - daisy::DaisySeed, [133](#)
- SetColor
 - daisy::RgbLed, [156](#)
- SetCursor
 - daisy::OledDisplay, [152](#)
- SetFootswitchLed
 - daisy::DaisyPetal, [126](#)
- SetLed
 - daisy::DaisySeed, [133](#)
- SetLooping
 - daisy::WavPlayer, [178](#)
- SetReceiveCallback
 - UsbHandle, [172](#)
- SetRingLed
 - daisy::DaisyPetal, [126](#)
- SetTestPoint
 - daisy::DaisySeed, [133](#)
- SHIFTREGISTER, [39](#)
 - dsy_sr_4021_init, [39](#)
 - DSY_SR_4021_PIN_CLK, [39](#)
 - DSY_SR_4021_PIN_CS, [39](#)
 - DSY_SR_4021_PIN_DATA, [39](#)
 - DSY_SR_4021_PIN_DATA2, [39](#)
 - DSY_SR_4021_PIN_LAST, [39](#)
 - dsy_sr_4021_state, [40](#)
 - dsy_sr_4021_update, [40](#)
- ShiftRegister595, [162](#)
 - Init, [163](#)
 - NUM_PINS, [163](#)
 - PIN_CLK, [163](#)
 - PIN_DATA, [163](#)
 - Pins, [163](#)
 - Set, [163](#)
 - Write, [164](#)
- speed
 - daisy::SdmmcHandlerInit, [162](#)
 - dsy_i2c_handle, [137](#)
- SPI_PERIPH_1
 - SERIAL, [25](#)
- SPI_PERIPH_3
 - SERIAL, [25](#)
- SPI_PERIPH_6
 - SERIAL, [25](#)
- SPI_PIN_CS
 - SERIAL, [25](#)
- SPI_PIN_MISO
 - SERIAL, [25](#)
- SPI_PIN_MOSI
 - SERIAL, [25](#)
- SPI_PIN_SCK
 - SERIAL, [25](#)
- SpiPeriph
 - SERIAL, [24](#)
- SpiPin
 - SERIAL, [25](#)
- src/ffconf.h, [179](#)
- src/hid_gatein.h, [184](#)
- src/hid_wavplayer.h, [184](#)
- src/usbd_cdc_if.h, [185](#)
- src/usbd_conf.h, [186](#)
- src/usbd_desc.h, [187](#)
- Start
 - daisy::AdcHandle, [112](#)
- StartAdc
 - daisy::DaisyPatch, [121](#)
 - daisy::DaisyPetal, [126](#)
 - daisy::DaisyPod, [130](#)
- StartAudio
 - daisy::DaisyPatch, [121](#)
 - daisy::DaisyPetal, [127](#)
 - daisy::DaisyPod, [130](#)
 - daisy::DaisySeed, [133](#)
- StartReceive
 - daisy::MidiHandler, [150](#)
- StartRx
 - daisy::UartHandler, [169](#)
- state
 - dsy_sdram_handle, [141](#)
- states
 - dsy_sr_4021_handle, [142](#)
- STM32_USB_OTG_DEVICE_LIBRARY, [102](#)
- Stop
 - daisy::AdcHandle, [112](#)
- Stream
 - daisy::WavPlayer, [178](#)
- SubChunk1ID
 - WAV_FormatTypeDef, [174](#)
- SubChunk1Size
 - WAV_FormatTypeDef, [174](#)
- SubChunk2ID
 - WAV_FormatTypeDef, [175](#)
- SubCHunk2Size
 - WAV_FormatTypeDef, [175](#)
- SUBSECTOR_ERASE_4_BYTE_ADDR_CMD
 - FLASH, [58](#)
- SUBSECTOR_ERASE_CMD
 - FLASH, [58](#)
- SUBSECTOR_ERASE_QPI_CMD
 - FLASH, [58](#)
- Sw
 - daisy::DaisyPetal, [124](#)
 - daisy::DaisyPod, [129](#)
- SW_1
 - BOARDS, [68](#)
 - daisy::DaisyPetal, [124](#)

- SW_2
 - BOARDS, 68
 - daisy::DaisyPetal, 124
- SW_3
 - BOARDS, 68
 - daisy::DaisyPetal, 124
- SW_4
 - daisy::DaisyPetal, 124
- SW_5
 - daisy::DaisyPetal, 124
- SW_6
 - daisy::DaisyPetal, 124
- SW_7
 - daisy::DaisyPetal, 124
- SW_LAST
 - BOARDS, 68
 - daisy::DaisyPetal, 124
- Swallow
 - daisy::RingBuffer< T, size >, 158
- switches
 - daisy::daisy_field, 118
 - daisy::DaisyPetal, 127
- sync_config
 - dsy_sai_handle, 139
- SYSTEM, 36
 - dsy_dma_init, 36
 - dsy_system_delay, 36
 - dsy_system_getnow, 36
 - dsy_system_init, 36
 - dsy_system_jumpton, 36
 - dsy_system_jumptonqspi, 37
- TimeHeldMs
 - daisy::Encoder, 143
 - daisy::Switch, 167
- TransmitExternal
 - UsbHandle, 172
- TransmitInternal
 - UsbHandle, 173
- Trig
 - daisy::GateIn, 145
- Type
 - daisy::Switch, 165
- type
 - daisy::MidiEvent, 148
- TYPE_MOMENTARY
 - daisy::Switch, 166
- TYPE_TOGGLE
 - daisy::Switch, 166
- Update
 - daisy::Led, 147
 - daisy::OledDisplay, 152
 - daisy::RgbLed, 156
- UpdateAnalogControls
 - daisy::DaisyPatch, 121
 - daisy::DaisyPetal, 127
 - daisy::DaisyPod, 131
- UpdateLeds
 - daisy::DaisyPetal, 127
 - daisy::DaisyPod, 131
- usb_handle
 - daisy::DaisySeed, 134
- USB_SIZ_STRING_SERIAL
 - USBD_DESC_Exported_Constants, 95
- USBD_CDC_IF, 80
 - USBD_CDC_IF_Exported_Defines, 81
 - USBD_CDC_IF_Exported_FunctionsPrototype, 85
 - CDC_Set_Rx_Callback_FS, 85
 - CDC_Transmit_FS, 85
 - CDC_Transmit_HS, 85
- USBD_CDC_IF_Exported_Macros, 83
- USBD_CDC_IF_Exported_Types, 82
 - CDC_ReceiveCallback, 82
- USBD_CDC_IF_Exported_Variables, 84
 - USBD_Interface_fops_FS, 84
 - USBD_Interface_fops_HS, 84
- USBD_CONF, 86
 - USBD_CONF_Exported_Defines, 88
 - DEVICE_FS, 88
 - DEVICE_HS, 88
 - USBD_DEBUG_LEVEL, 88
 - USBD_LPM_ENABLED, 88
 - USBD_MAX_NUM_CONFIGURATION, 88
 - USBD_MAX_NUM_INTERFACES, 88
 - USBD_MAX_STR_DESC_SIZ, 88
 - USBD_SELF_POWERED, 88
 - USBD_SUPPORT_USER_STRING, 89
- USBD_CONF_Exported_FunctionsPrototype, 93
- USBD_CONF_Exported_Macros, 90
 - USBD_DbgLog, 90
 - USBD_Delay, 90
 - USBD_ErrLog, 90
 - USBD_free, 90
 - USBD_malloc, 90
 - USBD_memcpy, 90
 - USBD_memset, 91
 - USBD_UsrLog, 91
- USBD_CONF_Exported_Types, 92
- USBD_CONF_Exported_Variables, 87
- USBD_DbgLog
 - USBD_CONF_Exported_Macros, 90
- USBD_DEBUG_LEVEL
 - USBD_CONF_Exported_Defines, 88
- USBD_Delay
 - USBD_CONF_Exported_Macros, 90
- USBD_DESC, 94
 - USBD_DESC_Exported_Constants, 95
 - DEVICE_ID1, 95
 - DEVICE_ID2, 95
 - DEVICE_ID3, 95
 - USB_SIZ_STRING_SERIAL, 95
- USBD_DESC_Exported_Defines, 96
- USBD_DESC_Exported_FunctionsPrototype, 100
- USBD_DESC_Exported_Macros, 98
- USBD_DESC_Exported_TypesDefinitions, 97
- USBD_DESC_Exported_Variables, 99

- FS_Desc, [99](#)
- HS_Desc, [99](#)
- USBD_ErrLog
 - USBD_CONF_Exported_Macros, [90](#)
- USBD_free
 - USBD_CONF_Exported_Macros, [90](#)
- USBD_Interface_fops_FS
 - USBD_CDC_IF_Exported_Variables, [84](#)
- USBD_Interface_fops_HS
 - USBD_CDC_IF_Exported_Variables, [84](#)
- USBD_LPM_ENABLED
 - USBD_CONF_Exported_Defines, [88](#)
- USBD_malloc
 - USBD_CONF_Exported_Macros, [90](#)
- USBD_MAX_NUM_CONFIGURATION
 - USBD_CONF_Exported_Defines, [88](#)
- USBD_MAX_NUM_INTERFACES
 - USBD_CONF_Exported_Defines, [88](#)
- USBD_MAX_STR_DESC_SIZ
 - USBD_CONF_Exported_Defines, [88](#)
- USBD_memcpy
 - USBD_CONF_Exported_Macros, [90](#)
- USBD_memset
 - USBD_CONF_Exported_Macros, [91](#)
- USBD_OTG_DRIVER, [103](#)
- USBD_SELF_POWERED
 - USBD_CONF_Exported_Defines, [88](#)
- USBD_SUPPORT_USER_STRING
 - USBD_CONF_Exported_Defines, [89](#)
- USBD_UsrLog
 - USBD_CONF_Exported_Macros, [91](#)
- UsbHandle, [170](#)
 - FS_BOTH, [171](#)
 - FS_EXTERNAL, [171](#)
 - FS_INTERNAL, [171](#)
 - Init, [171](#)
 - ReceiveCallback, [170](#)
 - SetReceiveCallback, [172](#)
 - TransmitExternal, [172](#)
 - TransmitInternal, [173](#)
 - UsbPeriph, [171](#)
- UsbPeriph
 - UsbHandle, [171](#)
- UTILITY, [71](#)
 - BSP_SD_AbortCallback, [73](#)
 - BSP_SD_CardInfo, [72](#)
 - BSP_SD_Erase, [73](#)
 - BSP_SD_GetCardInfo, [74](#)
 - BSP_SD_GetCardState, [74](#)
 - BSP_SD_Init, [74](#)
 - BSP_SD_IsDetected, [74](#)
 - BSP_SD_ITConfig, [74](#)
 - BSP_SD_ReadBlocks, [75](#)
 - BSP_SD_ReadBlocks_DMA, [75](#)
 - BSP_SD_ReadCpltCallback, [75](#)
 - BSP_SD_WriteBlocks, [75](#)
 - BSP_SD_WriteBlocks_DMA, [76](#)
 - BSP_SD_WriteCpltCallback, [76](#)
- cube, [76](#)
- DMA_BUFFER_MEM_SECTION, [72](#)
- dsy_get_unique_id, [76](#)
- DSY_GPIO_LAST, [73](#)
- dsy_gpio_port, [73](#)
- DSY_GPIOA, [73](#)
- DSY_GPIOB, [73](#)
- DSY_GPIOC, [73](#)
- DSY_GPIOD, [73](#)
- DSY_GPIOE, [73](#)
- DSY_GPIOF, [73](#)
- DSY_GPIOG, [73](#)
- DSY_GPIOH, [73](#)
- DSY_GPIOI, [73](#)
- DSY_GPIOJ, [73](#)
- DSY_GPIOK, [73](#)
- dsy_hal_map_get_i2c, [77](#)
- dsy_hal_map_get_pin, [77](#)
- dsy_hal_map_get_port, [77](#)
- dsy_pin, [78](#)
- dsy_pin_cmp, [78](#)
- DTCM_MEM_SECTION, [72](#)
- Font_11x18, [78](#)
- Font_16x26, [78](#)
- Font_6x8, [78](#)
- Font_7x10, [78](#)
- hi2c1, [78](#)
- hi2c2, [78](#)
- hi2c3, [79](#)
- hi2c4, [79](#)
- MSD_ERROR, [72](#)
- MSD_ERROR_SD_NOT_PRESENT, [72](#)
- MSD_OK, [72](#)
- SD_DATATIMEOUT, [72](#)
- SD_NOT_PRESENT, [72](#)
- SD_PRESENT, [72](#)
- SD_TRANSFER_BUSY, [73](#)
- SD_TRANSFER_OK, [73](#)
- Value
 - daisy::AnalogControl, [113](#)
 - daisy::Parameter, [155](#)
- value
 - daisy::ControlChangeEvent, [117](#)
- velocity
 - daisy::NoteOnEvent, [151](#)
- WAV_FILENAME_MAX
 - hid_wavplayer.h, [185](#)
- WAV_FormatTypeDef, [173](#)
 - AudioFormat, [174](#)
 - BitPerSample, [174](#)
 - BlockAlign, [174](#)
 - ByteRate, [174](#)
 - ChunkId, [174](#)
 - FileFormat, [174](#)
 - FileSize, [174](#)
 - NbrChannels, [174](#)
 - SampleRate, [174](#)

- SubChunk1ID, [174](#)
- SubChunk1Size, [174](#)
- SubChunk2ID, [175](#)
- SubCHunk2Size, [175](#)
- WHITE
 - daisy::Color, [115](#)
- writable
 - daisy::RingBuffer< T, 0 >, [161](#)
 - daisy::RingBuffer< T, size >, [159](#)
- Write
 - daisy::RingBuffer< T, 0 >, [161](#)
 - daisy::RingBuffer< T, size >, [159](#)
 - ShiftRegister595, [164](#)
- WRITE_DISABLE_CMD
 - FLASH, [58](#)
- WRITE_ENABLE_CMD
 - FLASH, [59](#)
- WRITE_ENHANCED_VOL_CFG_REG_CMD
 - FLASH, [59](#)
- WRITE_EXT_ADDR_REG_CMD
 - FLASH, [59](#)
- WRITE_LOCK_REG_CMD
 - FLASH, [59](#)
- WRITE_NONVOL_CFG_REG_CMD
 - FLASH, [59](#)
- WRITE_READ_PARAM_REG_CMD
 - FLASH, [59](#)
- WRITE_STATUS_REG_CMD
 - FLASH, [60](#)
- WriteChar
 - daisy::OledDisplay, [152](#)
- WriteString
 - daisy::OledDisplay, [153](#)