

libDaisy

Generated by Doxygen 1.8.13



# Contents

<b>1</b>	<b>libdaisy</b>	<b>1</b>
1.1	Using libdaisy . . . . .	1
1.1.1	daisy.h . . . . .	2
1.1.2	daisy_seed.h . . . . .	2
1.1.3	daisy_platform.h . . . . .	2
<b>2</b>	<b>Module Index</b>	<b>3</b>
2.1	Modules . . . . .	3
<b>3</b>	<b>Namespace Index</b>	<b>5</b>
3.1	Namespace List . . . . .	5
<b>4</b>	<b>Class Index</b>	<b>7</b>
4.1	Class List . . . . .	7
<b>5</b>	<b>File Index</b>	<b>9</b>
5.1	File List . . . . .	9
<b>6</b>	<b>Module Documentation</b>	<b>11</b>
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	<a href="#">dsy_audio_callback</a>	13
6.3.2.2	<a href="#">dsy_audio_mc_callback</a>	13
6.3.3	<a href="#">Enumeration Type Documentation</a>	13
6.3.3.1	<a href="#">anonymous enum</a>	13
6.3.4	<a href="#">Function Documentation</a>	14
6.3.4.1	<a href="#">dsy_audio_enter_bypass()</a>	14
6.3.4.2	<a href="#">dsy_audio_exit_bypass()</a>	14
6.3.4.3	<a href="#">dsy_audio_init()</a>	14
6.3.4.4	<a href="#">dsy_audio_passthru()</a>	14
6.3.4.5	<a href="#">dsy_audio_set_blocksize()</a>	15
6.3.4.6	<a href="#">dsy_audio_set_callback()</a>	15
6.3.4.7	<a href="#">dsy_audio_set_mc_callback()</a>	15
6.3.4.8	<a href="#">dsy_audio_silence()</a>	15
6.3.4.9	<a href="#">dsy_audio_start()</a>	15
6.3.4.10	<a href="#">dsy_audio_stop()</a>	15
6.4	<a href="#">CONTROLS</a>	16
6.4.1	<a href="#">Detailed Description</a>	17
6.4.2	<a href="#">Enumeration Type Documentation</a>	17
6.4.2.1	<a href="#">Curve</a>	17
6.4.2.2	<a href="#">Polarity</a>	17
6.4.2.3	<a href="#">Pull</a>	17
6.4.2.4	<a href="#">Type</a>	18
6.4.3	<a href="#">Function Documentation</a>	18
6.4.3.1	<a href="#">AnalogControl()</a>	18
6.4.3.2	<a href="#">Debounce()</a> <small>[1/2]</small>	18
6.4.3.3	<a href="#">Debounce()</a> <small>[2/2]</small>	18
6.4.3.4	<a href="#">FallingEdge()</a> <small>[1/2]</small>	18
6.4.3.5	<a href="#">FallingEdge()</a> <small>[2/2]</small>	19
6.4.3.6	<a href="#">Increment()</a>	19
6.4.3.7	<a href="#">Init()</a> <small>[1/5]</small>	19

6.4.3.8	Init() [2/5]	19
6.4.3.9	Init() [3/5]	19
6.4.3.10	Init() [4/5]	20
6.4.3.11	Init() [5/5]	20
6.4.3.12	InitBipolarCv()	21
6.4.3.13	Parameter()	21
6.4.3.14	Pressed() [1/2]	21
6.4.3.15	Pressed() [2/2]	21
6.4.3.16	Process() [1/2]	22
6.4.3.17	Process() [2/2]	22
6.4.3.18	RisingEdge() [1/2]	22
6.4.3.19	RisingEdge() [2/2]	22
6.4.3.20	TimeHeldMs() [1/2]	22
6.4.3.21	TimeHeldMs() [2/2]	22
6.4.3.22	Value() [1/2]	23
6.4.3.23	Value() [2/2]	23
6.4.3.24	~AnalogControl()	23
6.4.3.25	~Parameter()	23
6.5	FEEDBACK	24
6.5.1	Detailed Description	24
6.5.2	Enumeration Type Documentation	24
6.5.2.1	Pins	24
6.5.3	Function Documentation	25
6.5.3.1	DrawPixel()	25
6.5.3.2	Fill()	25
6.5.3.3	Init() [1/3]	25
6.5.3.4	Init() [2/3]	26
6.5.3.5	Init() [3/3]	26
6.5.3.6	Set() [1/2]	26
6.5.3.7	Set() [2/2]	27

6.5.3.8	<a href="#">SetColor()</a>	27
6.5.3.9	<a href="#">SetCursor()</a>	27
6.5.3.10	<a href="#">Update()</a> [1/3]	28
6.5.3.11	<a href="#">Update()</a> [2/3]	28
6.5.3.12	<a href="#">Update()</a> [3/3]	28
6.5.3.13	<a href="#">WriteChar()</a>	28
6.5.3.14	<a href="#">WriteString()</a>	29
6.6	<a href="#">EXTERNAL</a>	30
6.6.1	<a href="#">Detailed Description</a>	31
6.6.2	<a href="#">Enumeration Type Documentation</a>	31
6.6.2.1	<a href="#">MidiInputMode</a>	31
6.6.2.2	<a href="#">MidiMessageType</a>	31
6.6.2.3	<a href="#">MidiOutputMode</a>	31
6.6.3	<a href="#">Function Documentation</a>	33
6.6.3.1	<a href="#">AsControlChange()</a>	33
6.6.3.2	<a href="#">AsNoteOn()</a>	33
6.6.3.3	<a href="#">HasEvents()</a>	33
6.6.3.4	<a href="#">Init()</a>	33
6.6.3.5	<a href="#">Listen()</a>	34
6.6.3.6	<a href="#">Parse()</a>	34
6.6.3.7	<a href="#">PopEvent()</a>	34
6.6.3.8	<a href="#">StartReceive()</a>	34
6.6.4	<a href="#">Variable Documentation</a>	34
6.6.4.1	<a href="#">channel</a> [1/3]	35
6.6.4.2	<a href="#">channel</a> [2/3]	35
6.6.4.3	<a href="#">channel</a> [3/3]	35
6.6.4.4	<a href="#">control_number</a>	35
6.6.4.5	<a href="#">data</a>	35
6.6.4.6	<a href="#">note</a>	35
6.6.4.7	<a href="#">type</a>	35

6.6.4.8	value	35
6.6.4.9	velocity	35
6.7	PERIPHERAL	36
6.7.1	Detailed Description	36
6.8	SERIAL	37
6.8.1	Detailed Description	38
6.8.2	Enumeration Type Documentation	38
6.8.2.1	anonymous enum	38
6.8.2.2	dsy_audio_bitdepth	39
6.8.2.3	dsy_audio_device	39
6.8.2.4	dsy_audio_dir	39
6.8.2.5	dsy_audio_sai	40
6.8.2.6	dsy_audio_samplerate	40
6.8.2.7	dsy_audio_sync	40
6.8.2.8	dsy_i2c_periph	41
6.8.2.9	dsy_i2c_pin	41
6.8.2.10	dsy_i2c_speed	41
6.8.2.11	dsy_qspi_device	42
6.8.2.12	dsy_qspi_mode	42
6.8.2.13	dsy_qspi_pin	42
6.8.2.14	dsy_sai_pin	43
6.8.2.15	SpiPeriph	43
6.8.2.16	SpiPin	43
6.8.3	Function Documentation	44
6.8.3.1	BlockingTransmit()	44
6.8.3.2	CheckError()	44
6.8.3.3	dsy_i2c_init()	44
6.8.3.4	dsy_qspi_deinit()	44
6.8.3.5	dsy_qspi_erase()	45
6.8.3.6	dsy_qspi_erasesector()	45

6.8.3.7	<code>dsy_qspi_init()</code>	45
6.8.3.8	<code>dsy_qspi_write()</code>	46
6.8.3.9	<code>dsy_qspi_writepage()</code>	46
6.8.3.10	<code>dsy_sai_init()</code>	47
6.8.3.11	<code>dsy_sai_init_from_handle()</code>	47
6.8.3.12	<code>FlushRx()</code>	47
6.8.3.13	<code>Init()</code> [1/2]	48
6.8.3.14	<code>Init()</code> [2/2]	48
6.8.3.15	<code>PollReceive()</code>	48
6.8.3.16	<code>PollTx()</code>	48
6.8.3.17	<code>PopRx()</code>	49
6.8.3.18	<code>Readable()</code>	49
6.8.3.19	<code>RxActive()</code>	49
6.8.3.20	<code>StartRx()</code>	49
6.8.4	Variable Documentation	50
6.8.4.1	<code>kUartMaxBufferSize</code>	50
6.9	ANALOG_DIGITAL_CONVERSION	51
6.9.1	Detailed Description	52
6.9.2	Enumeration Type Documentation	52
6.9.2.1	<code>dsy_dac_bitdepth</code>	52
6.9.2.2	<code>dsy_dac_channel</code>	52
6.9.2.3	<code>dsy_dac_mode</code>	52
6.9.2.4	<code>MuxPin</code>	53
6.9.2.5	<code>OverSampling</code>	53
6.9.3	Function Documentation	53
6.9.3.1	<code>dsy_dac_init()</code>	53
6.9.3.2	<code>dsy_dac_start()</code>	54
6.9.3.3	<code>dsy_dac_write()</code>	54
6.9.3.4	<code>Get()</code>	54
6.9.3.5	<code>GetFloat()</code>	55



6.9.3.6	<a href="#">GetMux()</a> . . . . .	55
6.9.3.7	<a href="#">GetMuxFloat()</a> . . . . .	55
6.9.3.8	<a href="#">GetMuxPtr()</a> . . . . .	56
6.9.3.9	<a href="#">GetPtr()</a> . . . . .	56
6.9.3.10	<a href="#">Init()</a> . . . . .	57
6.9.3.11	<a href="#">InitMux()</a> . . . . .	57
6.9.3.12	<a href="#">InitSingle()</a> . . . . .	57
6.9.3.13	<a href="#">Start()</a> . . . . .	58
6.9.3.14	<a href="#">Stop()</a> . . . . .	58
6.9.4	<a href="#">Variable Documentation</a> . . . . .	58
6.9.4.1	<a href="#">mux_channels_</a> . . . . .	58
6.9.4.2	<a href="#">mux_pin_</a> . . . . .	58
6.9.4.3	<a href="#">pin_</a> . . . . .	58
6.10	<a href="#">OTHER</a> . . . . .	59
6.10.1	<a href="#">Detailed Description</a> . . . . .	59
6.10.2	<a href="#">Enumeration Type Documentation</a> . . . . .	59
6.10.2.1	<a href="#">dsy_gpio_mode</a> . . . . .	59
6.10.2.2	<a href="#">dsy_gpio_pull</a> . . . . .	60
6.10.3	<a href="#">Function Documentation</a> . . . . .	60
6.10.3.1	<a href="#">dsy_gpio_deinit()</a> . . . . .	60
6.10.3.2	<a href="#">dsy_gpio_init()</a> . . . . .	60
6.10.3.3	<a href="#">dsy_gpio_read()</a> . . . . .	61
6.10.3.4	<a href="#">dsy_gpio_toggle()</a> . . . . .	61
6.10.3.5	<a href="#">dsy_gpio_write()</a> . . . . .	61
6.10.3.6	<a href="#">dsy_tim_delay_ms()</a> . . . . .	62
6.10.3.7	<a href="#">dsy_tim_delay_tick()</a> . . . . .	62
6.10.3.8	<a href="#">dsy_tim_delay_us()</a> . . . . .	62
6.10.3.9	<a href="#">dsy_tim_get_ms()</a> . . . . .	62
6.10.3.10	<a href="#">dsy_tim_get_tick()</a> . . . . .	63
6.10.3.11	<a href="#">dsy_tim_get_us()</a> . . . . .	63

6.10.3.12	<a href="#">dsy_tim_init()</a>	63
6.10.3.13	<a href="#">dsy_tim_start()</a>	63
6.11	<a href="#">SYSTEM</a>	64
6.11.1	<a href="#">Detailed Description</a>	64
6.11.2	<a href="#">Function Documentation</a>	64
6.11.2.1	<a href="#">dsy_dma_init()</a>	64
6.11.2.2	<a href="#">dsy_system_delay()</a>	64
6.11.2.3	<a href="#">dsy_system_getnow()</a>	65
6.11.2.4	<a href="#">dsy_system_init()</a>	65
6.11.2.5	<a href="#">dsy_system_jumpto()</a>	65
6.11.2.6	<a href="#">dsy_system_jumptoqspi()</a>	65
6.12	<a href="#">DEVICE</a>	66
6.12.1	<a href="#">Detailed Description</a>	66
6.13	<a href="#">SHIFTREGISTER</a>	67
6.13.1	<a href="#">Detailed Description</a>	67
6.13.2	<a href="#">Enumeration Type Documentation</a>	67
6.13.2.1	<a href="#">anonymous enum</a>	67
6.13.3	<a href="#">Function Documentation</a>	68
6.13.3.1	<a href="#">dsy_sr_4021_init()</a>	68
6.13.3.2	<a href="#">dsy_sr_4021_state()</a>	68
6.13.3.3	<a href="#">dsy_sr_4021_update()</a>	68
6.14	<a href="#">FLASH</a>	69
6.14.1	<a href="#">Detailed Description</a>	72
6.14.2	<a href="#">Macro Definition Documentation</a>	72
6.14.2.1	<a href="#">BLOCK_ERASE_32K_CMD</a> [1/2]	72
6.14.2.2	<a href="#">BLOCK_ERASE_32K_CMD</a> [2/2]	73
6.14.2.3	<a href="#">CLEAR_FLAG_STATUS_REG_CMD</a> [1/2]	73
6.14.2.4	<a href="#">CLEAR_FLAG_STATUS_REG_CMD</a> [2/2]	73
6.14.2.5	<a href="#">DIE_ERASE_CMD</a> [1/2]	73
6.14.2.6	<a href="#">DIE_ERASE_CMD</a> [2/2]	73

6.14.2.7 DUAL_IN_FAST_PROG_CMD [1/2] . . . . .	73
6.14.2.8 DUAL_IN_FAST_PROG_CMD [2/2] . . . . .	73
6.14.2.9 DUAL_INOUT_FAST_READ_4_BYTE_ADDR_CMD [1/2] . . . . .	73
6.14.2.10 DUAL_INOUT_FAST_READ_4_BYTE_ADDR_CMD [2/2] . . . . .	74
6.14.2.11 DUAL_INOUT_FAST_READ_CMD [1/2] . . . . .	74
6.14.2.12 DUAL_INOUT_FAST_READ_CMD [2/2] . . . . .	74
6.14.2.13 DUAL_INOUT_FAST_READ_DTR_CMD [1/2] . . . . .	74
6.14.2.14 DUAL_INOUT_FAST_READ_DTR_CMD [2/2] . . . . .	74
6.14.2.15 DUAL_OUT_FAST_READ_4_BYTE_ADDR_CMD [1/2] . . . . .	74
6.14.2.16 DUAL_OUT_FAST_READ_4_BYTE_ADDR_CMD [2/2] . . . . .	74
6.14.2.17 DUAL_OUT_FAST_READ_CMD [1/2] . . . . .	74
6.14.2.18 DUAL_OUT_FAST_READ_CMD [2/2] . . . . .	75
6.14.2.19 DUAL_OUT_FAST_READ_DTR_CMD [1/2] . . . . .	75
6.14.2.20 DUAL_OUT_FAST_READ_DTR_CMD [2/2] . . . . .	75
6.14.2.21 ENTER_4_BYTE_ADDR_MODE_CMD [1/2] . . . . .	75
6.14.2.22 ENTER_4_BYTE_ADDR_MODE_CMD [2/2] . . . . .	75
6.14.2.23 ENTER_QUAD_CMD [1/2] . . . . .	75
6.14.2.24 ENTER_QUAD_CMD [2/2] . . . . .	75
6.14.2.25 EXIT_4_BYTE_ADDR_MODE_CMD [1/2] . . . . .	75
6.14.2.26 EXIT_4_BYTE_ADDR_MODE_CMD [2/2] . . . . .	76
6.14.2.27 EXIT_QUAD_CMD [1/2] . . . . .	76
6.14.2.28 EXIT_QUAD_CMD [2/2] . . . . .	76
6.14.2.29 EXT_DUAL_IN_FAST_PROG_CMD [1/2] . . . . .	76
6.14.2.30 EXT_DUAL_IN_FAST_PROG_CMD [2/2] . . . . .	76
6.14.2.31 EXT_QUAD_IN_FAST_PROG_CMD [1/2] . . . . .	76
6.14.2.32 EXT_QUAD_IN_FAST_PROG_CMD [2/2] . . . . .	76
6.14.2.33 FAST_READ_4_BYTE_ADDR_CMD [1/2] . . . . .	76
6.14.2.34 FAST_READ_4_BYTE_ADDR_CMD [2/2] . . . . .	77
6.14.2.35 FAST_READ_CMD [1/2] . . . . .	77
6.14.2.36 FAST_READ_CMD [2/2] . . . . .	77

6.14.2.37 FAST_READ_DTR_CMD [1/2]	77
6.14.2.38 FAST_READ_DTR_CMD [2/2]	77
6.14.2.39 IS25LP064A_EAR_HIGHEST_SE	77
6.14.2.40 IS25LP064A_EAR_LOWEST_SEG	77
6.14.2.41 IS25LP064A_EAR_SECOND_SEG	77
6.14.2.42 IS25LP064A_EAR_THIRD_SEG	78
6.14.2.43 IS25LP064A_EVCR_DTRP	78
6.14.2.44 IS25LP064A_EVCR_DUAL	78
6.14.2.45 IS25LP064A_EVCR_ODS	78
6.14.2.46 IS25LP064A_EVCR_QUAD	78
6.14.2.47 IS25LP064A_EVCR_RH	78
6.14.2.48 IS25LP064A_FSR_ERERR	78
6.14.2.49 IS25LP064A_FSR_ERSUS	78
6.14.2.50 IS25LP064A_FSR_NBADDR	79
6.14.2.51 IS25LP064A_FSR_PGERR	79
6.14.2.52 IS25LP064A_FSR_PGSUS	79
6.14.2.53 IS25LP064A_FSR_PRERR	79
6.14.2.54 IS25LP064A_FSR_READY	79
6.14.2.55 IS25LP064A_NVCR_DTRP	79
6.14.2.56 IS25LP064A_NVCR_DUAL	79
6.14.2.57 IS25LP064A_NVCR_NB_DUMMY	79
6.14.2.58 IS25LP064A_NVCR_NBADDR	80
6.14.2.59 IS25LP064A_NVCR_ODS	80
6.14.2.60 IS25LP064A_NVCR_QUAB	80
6.14.2.61 IS25LP064A_NVCR_RH	80
6.14.2.62 IS25LP064A_NVCR_SEGMENT	80
6.14.2.63 IS25LP064A_NVCR_XIP	80
6.14.2.64 IS25LP064A_SR_QE	80
6.14.2.65 IS25LP064A_SR_SRWREN	80
6.14.2.66 IS25LP064A_SR_WIP	81

6.14.2.67 IS25LP064A_SR_WREN . . . . .	81
6.14.2.68 IS25LP064A_VCR_NB_DUMMY . . . . .	81
6.14.2.69 IS25LP064A_VCR_WRAP . . . . .	81
6.14.2.70 IS25LP064A_VCR_XIP . . . . .	81
6.14.2.71 IS25LP080D_EAR_HIGHEST_SEG . . . . .	81
6.14.2.72 IS25LP080D_EAR_LOWEST_SEG . . . . .	81
6.14.2.73 IS25LP080D_EAR_SECOND_SEG . . . . .	82
6.14.2.74 IS25LP080D_EAR_THIRD_SEG . . . . .	82
6.14.2.75 IS25LP080D_EVCR_DTRP . . . . .	82
6.14.2.76 IS25LP080D_EVCR_DUAL . . . . .	82
6.14.2.77 IS25LP080D_EVCR_ODS . . . . .	82
6.14.2.78 IS25LP080D_EVCR_QUAD . . . . .	82
6.14.2.79 IS25LP080D_EVCR_RH . . . . .	82
6.14.2.80 IS25LP080D_FSR_ERERR . . . . .	82
6.14.2.81 IS25LP080D_FSR_ERSUS . . . . .	83
6.14.2.82 IS25LP080D_FSR_NBADDR . . . . .	83
6.14.2.83 IS25LP080D_FSR_PGERR . . . . .	83
6.14.2.84 IS25LP080D_FSR_PGSUS . . . . .	83
6.14.2.85 IS25LP080D_FSR_PRERR . . . . .	83
6.14.2.86 IS25LP080D_FSR_READY . . . . .	83
6.14.2.87 IS25LP080D_NVCR_DTRP . . . . .	83
6.14.2.88 IS25LP080D_NVCR_DUAL . . . . .	83
6.14.2.89 IS25LP080D_NVCR_NB_DUMMY . . . . .	84
6.14.2.90 IS25LP080D_NVCR_NBADDR . . . . .	84
6.14.2.91 IS25LP080D_NVCR_ODS . . . . .	84
6.14.2.92 IS25LP080D_NVCR_QUAB . . . . .	84
6.14.2.93 IS25LP080D_NVCR_RH . . . . .	84
6.14.2.94 IS25LP080D_NVCR_SEGMENT . . . . .	84
6.14.2.95 IS25LP080D_NVCR_XIP . . . . .	84
6.14.2.96 IS25LP080D_SR_QE . . . . .	84

6.14.2.97 IS25LP080D_SR_SRWREN . . . . .	85
6.14.2.98 IS25LP080D_SR_WIP . . . . .	85
6.14.2.99 IS25LP080D_SR_WREN . . . . .	85
6.14.2.100 IS25LP080D_VCR_NB_DUMMY . . . . .	85
6.14.2.101 IS25LP080D_VCR_WRAP . . . . .	85
6.14.2.102 IS25LP080D_VCR_XIP . . . . .	85
6.14.2.103 MULTIPLE_IO_READ_ID_CMD [1/2] . . . . .	85
6.14.2.104 MULTIPLE_IO_READ_ID_CMD [2/2] . . . . .	86
6.14.2.105 PAGE_PROG_4_BYTE_ADDR_CMD [1/2] . . . . .	86
6.14.2.106 PAGE_PROG_4_BYTE_ADDR_CMD [2/2] . . . . .	86
6.14.2.107 PAGE_PROG_CMD [1/2] . . . . .	86
6.14.2.108 PAGE_PROG_CMD [2/2] . . . . .	86
6.14.2.109 PROG_ERASE_RESUME_CMD [1/2] . . . . .	86
6.14.2.110 PROG_ERASE_RESUME_CMD [2/2] . . . . .	86
6.14.2.111 PROG_ERASE_SUSPEND_CMD [1/2] . . . . .	86
6.14.2.112 PROG_ERASE_SUSPEND_CMD [2/2] . . . . .	87
6.14.2.113 PROG_OTP_ARRAY_CMD [1/2] . . . . .	87
6.14.2.114 PROG_OTP_ARRAY_CMD [2/2] . . . . .	87
6.14.2.115 QUAD_IN_FAST_PROG_4_BYTE_ADDR_CMD [1/2] . . . . .	87
6.14.2.116 QUAD_IN_FAST_PROG_4_BYTE_ADDR_CMD [2/2] . . . . .	87
6.14.2.117 QUAD_IN_FAST_PROG_CMD [1/2] . . . . .	87
6.14.2.118 QUAD_IN_FAST_PROG_CMD [2/2] . . . . .	87
6.14.2.119 QUAD_INOUT_FAST_READ_4_BYTE_ADDR_CMD [1/2] . . . . .	87
6.14.2.120 QUAD_INOUT_FAST_READ_4_BYTE_ADDR_CMD [2/2] . . . . .	88
6.14.2.121 QUAD_INOUT_FAST_READ_CMD [1/2] . . . . .	88
6.14.2.122 QUAD_INOUT_FAST_READ_CMD [2/2] . . . . .	88
6.14.2.123 QUAD_INOUT_FAST_READ_DTR_CMD [1/2] . . . . .	88
6.14.2.124 QUAD_INOUT_FAST_READ_DTR_CMD [2/2] . . . . .	88
6.14.2.125 QUAD_OUT_FAST_READ_4_BYTE_ADDR_CMD [1/2] . . . . .	88
6.14.2.126 QUAD_OUT_FAST_READ_4_BYTE_ADDR_CMD [2/2] . . . . .	88

6.14.2.127	QUAD_OUT_FAST_READ_CMD [1/2]	88
6.14.2.128	QUAD_OUT_FAST_READ_CMD [2/2]	89
6.14.2.129	QUAD_OUT_FAST_READ_DTR_CMD [1/2]	89
6.14.2.130	QUAD_OUT_FAST_READ_DTR_CMD [2/2]	89
6.14.2.131	READ_4_BYTE_ADDR_CMD [1/2]	89
6.14.2.132	READ_4_BYTE_ADDR_CMD [2/2]	89
6.14.2.133	READ_CMD [1/2]	89
6.14.2.134	READ_CMD [2/2]	89
6.14.2.135	READ_ENHANCED_VOL_CFG_REG_CMD [1/2]	89
6.14.2.136	READ_ENHANCED_VOL_CFG_REG_CMD [2/2]	90
6.14.2.137	READ_EXT_ADDR_REG_CMD [1/2]	90
6.14.2.138	READ_EXT_ADDR_REG_CMD [2/2]	90
6.14.2.139	READ_FLAG_STATUS_REG_CMD [1/2]	90
6.14.2.140	READ_FLAG_STATUS_REG_CMD [2/2]	90
6.14.2.141	READ_ID_CMD [1/2]	90
6.14.2.142	READ_ID_CMD [2/2]	90
6.14.2.143	READ_ID_CMD2 [1/2]	90
6.14.2.144	READ_ID_CMD2 [2/2]	91
6.14.2.145	READ_LOCK_REG_CMD [1/2]	91
6.14.2.146	READ_LOCK_REG_CMD [2/2]	91
6.14.2.147	READ_NONVOL_CFG_REG_CMD [1/2]	91
6.14.2.148	READ_NONVOL_CFG_REG_CMD [2/2]	91
6.14.2.149	READ_OTP_ARRAY_CMD [1/2]	91
6.14.2.150	READ_OTP_ARRAY_CMD [2/2]	91
6.14.2.151	READ_READ_PARAM_REG_CMD [1/2]	91
6.14.2.152	READ_READ_PARAM_REG_CMD [2/2]	92
6.14.2.153	READ_SERIAL_FLASH_DISCO_PARAM_CMD [1/2]	92
6.14.2.154	READ_SERIAL_FLASH_DISCO_PARAM_CMD [2/2]	92
6.14.2.155	READ_STATUS_REG_CMD [1/2]	92
6.14.2.156	READ_STATUS_REG_CMD [2/2]	92

6.14.2.157	RESET_ENABLE_CMD [1/2]	92
6.14.2.158	RESET_ENABLE_CMD [2/2]	92
6.14.2.159	RESET_MEMORY_CMD [1/2]	92
6.14.2.160	RESET_MEMORY_CMD [2/2]	93
6.14.2.161	SECTOR_ERASE_4_BYTE_ADDR_CMD [1/2]	93
6.14.2.162	SECTOR_ERASE_4_BYTE_ADDR_CMD [2/2]	93
6.14.2.163	SECTOR_ERASE_CMD [1/2]	93
6.14.2.164	SECTOR_ERASE_CMD [2/2]	93
6.14.2.165	SUBSECTOR_ERASE_4_BYTE_ADDR_CMD [1/2]	93
6.14.2.166	SUBSECTOR_ERASE_4_BYTE_ADDR_CMD [2/2]	93
6.14.2.167	SUBSECTOR_ERASE_CMD [1/2]	93
6.14.2.168	SUBSECTOR_ERASE_CMD [2/2]	94
6.14.2.169	SUBSECTOR_ERASE_QPI_CMD [1/2]	94
6.14.2.170	SUBSECTOR_ERASE_QPI_CMD [2/2]	94
6.14.2.171	WRITE_DISABLE_CMD [1/2]	94
6.14.2.172	WRITE_DISABLE_CMD [2/2]	94
6.14.2.173	WRITE_ENABLE_CMD [1/2]	94
6.14.2.174	WRITE_ENABLE_CMD [2/2]	94
6.14.2.175	WRITE_ENHANCED_VOL_CFG_REG_CMD [1/2]	94
6.14.2.176	WRITE_ENHANCED_VOL_CFG_REG_CMD [2/2]	95
6.14.2.177	WRITE_EXT_ADDR_REG_CMD [1/2]	95
6.14.2.178	WRITE_EXT_ADDR_REG_CMD [2/2]	95
6.14.2.179	WRITE_LOCK_REG_CMD [1/2]	95
6.14.2.180	WRITE_LOCK_REG_CMD [2/2]	95
6.14.2.181	WRITE_NONVOL_CFG_REG_CMD [1/2]	95
6.14.2.182	WRITE_NONVOL_CFG_REG_CMD [2/2]	95
6.14.2.183	WRITE_READ_PARAM_REG_CMD [1/2]	95
6.14.2.184	WRITE_READ_PARAM_REG_CMD [2/2]	96
6.14.2.185	WRITE_STATUS_REG_CMD [1/2]	96
6.14.2.186	WRITE_STATUS_REG_CMD [2/2]	96



6.15 CODEC	97
6.15.1 Detailed Description	97
6.15.2 Typedef Documentation	97
6.15.2.1 sa_audio_callback	97
6.15.3 Function Documentation	97
6.15.3.1 codec_ak4556_init()	97
6.15.3.2 codec_pcm3060_init()	98
6.15.3.3 codec_wm8731_enter_bypass()	98
6.15.3.4 codec_wm8731_exit_bypass()	98
6.15.3.5 codec_wm8731_init()	99
6.16 LED	100
6.16.1 Detailed Description	100
6.16.2 Enumeration Type Documentation	100
6.16.2.1 anonymous enum	100
6.16.3 Function Documentation	101
6.16.3.1 dsy_led_driver_color_by_name()	101
6.16.3.2 dsy_led_driver_init()	101
6.16.3.3 dsy_led_driver_set_led()	101
6.16.3.4 dsy_led_driver_update()	102
6.17 SDRAM	103
6.17.1 Detailed Description	103
6.17.2 Macro Definition Documentation	103
6.17.2.1 DSY_SDRAM_BSS	103
6.17.2.2 DSY_SDRAM_DATA	103
6.17.3 Enumeration Type Documentation	103
6.17.3.1 anonymous enum	103
6.17.3.2 dsy_sdram_pin	104
6.17.3.3 dsy_sdram_state	104
6.17.4 Function Documentation	104
6.17.4.1 dsy_sdram_init()	104

6.18	BOARDS	105
6.18.1	Detailed Description	108
6.18.2	Enumeration Type Documentation	108
6.18.2.1	anonymous enum	108
6.18.2.2	anonymous enum	108
6.18.2.3	anonymous enum	109
6.18.2.4	anonymous enum	109
6.18.2.5	Ctrl	110
6.18.2.6	FootswitchLed	110
6.18.2.7	GateInput	110
6.18.2.8	Knob [1/2]	110
6.18.2.9	Knob [2/2]	111
6.18.2.10	RingLed	111
6.18.2.11	Sw [1/2]	112
6.18.2.12	Sw [2/2]	112
6.18.3	Function Documentation	112
6.18.3.1	AudioBlockSize() [1/3]	112
6.18.3.2	AudioBlockSize() [2/3]	112
6.18.3.3	AudioBlockSize() [3/3]	113
6.18.3.4	AudioCallbackRate() [1/3]	113
6.18.3.5	AudioCallbackRate() [2/3]	113
6.18.3.6	AudioCallbackRate() [3/3]	113
6.18.3.7	AudioSampleRate() [1/4]	113
6.18.3.8	AudioSampleRate() [2/4]	113
6.18.3.9	AudioSampleRate() [3/4]	113
6.18.3.10	AudioSampleRate() [4/4]	113
6.18.3.11	ChangeAudioCallback() [1/3]	113
6.18.3.12	ChangeAudioCallback() [2/3]	114
6.18.3.13	ChangeAudioCallback() [3/3]	114
6.18.3.14	ClearLeds() [1/2]	114

6.18.3.15 ClearLeds() [2/2] . . . . .	114
6.18.3.16 Configure() . . . . .	115
6.18.3.17 daisy_field_init() . . . . .	115
6.18.3.18 DaisyPatch() . . . . .	116
6.18.3.19 DaisyPetal() . . . . .	116
6.18.3.20 DebounceControls() [1/3] . . . . .	116
6.18.3.21 DebounceControls() [2/3] . . . . .	116
6.18.3.22 DebounceControls() [3/3] . . . . .	116
6.18.3.23 DelayMs() [1/3] . . . . .	116
6.18.3.24 DelayMs() [2/3] . . . . .	116
6.18.3.25 DelayMs() [3/3] . . . . .	117
6.18.3.26 DisplayControls() . . . . .	117
6.18.3.27 f2s16() . . . . .	117
6.18.3.28 f2s24() . . . . .	118
6.18.3.29 GetCtrlValue() . . . . .	118
6.18.3.30 GetExpression() . . . . .	118
6.18.3.31 GetKnobValue() [1/2] . . . . .	118
6.18.3.32 GetKnobValue() [2/2] . . . . .	118
6.18.3.33 GetPin() . . . . .	119
6.18.3.34 Init() [1/4] . . . . .	119
6.18.3.35 Init() [2/4] . . . . .	119
6.18.3.36 Init() [3/4] . . . . .	119
6.18.3.37 Init() [4/4] . . . . .	119
6.18.3.38 s162f() . . . . .	119
6.18.3.39 s242f() . . . . .	120
6.18.3.40 SetAudioBlockSize() [1/4] . . . . .	120
6.18.3.41 SetAudioBlockSize() [2/4] . . . . .	120
6.18.3.42 SetAudioBlockSize() [3/4] . . . . .	121
6.18.3.43 SetAudioBlockSize() [4/4] . . . . .	121
6.18.3.44 SetFootswitchLed() . . . . .	121

6.18.3.45 SetLed()	121
6.18.3.46 SetRingLed()	121
6.18.3.47 SetTestPoint()	122
6.18.3.48 StartAdc() [1/3]	122
6.18.3.49 StartAdc() [2/3]	122
6.18.3.50 StartAdc() [3/3]	122
6.18.3.51 StartAudio() [1/4]	122
6.18.3.52 StartAudio() [2/4]	123
6.18.3.53 StartAudio() [3/4]	123
6.18.3.54 StartAudio() [4/4]	123
6.18.3.55 UpdateAnalogControls() [1/3]	123
6.18.3.56 UpdateAnalogControls() [2/3]	123
6.18.3.57 UpdateAnalogControls() [3/3]	124
6.18.3.58 UpdateLeds() [1/2]	124
6.18.3.59 UpdateLeds() [2/2]	124
6.18.3.60 ~DaisyPatch()	124
6.18.3.61 ~DaisyPetal()	124
6.18.4 Variable Documentation	124
6.18.4.1 adc	124
6.18.4.2 audio_handle	124
6.18.4.3 button1	125
6.18.4.4 button2	125
6.18.4.5 buttons	125
6.18.4.6 controls	125
6.18.4.7 cvs	125
6.18.4.8 dac_handle	125
6.18.4.9 display	125
6.18.4.10 encoder [1/3]	125
6.18.4.11 encoder [2/3]	126
6.18.4.12 encoder [3/3]	126

6.18.4.13 expression . . . . .	126
6.18.4.14 footswitch_led . . . . .	126
6.18.4.15 gate_in . . . . .	126
6.18.4.16 gate_input . . . . .	126
6.18.4.17 gate_out . . . . .	126
6.18.4.18 gate_output . . . . .	126
6.18.4.19 i2c1_handle . . . . .	127
6.18.4.20 i2c2_handle . . . . .	127
6.18.4.21 keyboard_sr . . . . .	127
6.18.4.22 knob . . . . .	127
6.18.4.23 knob1 . . . . .	127
6.18.4.24 knob2 . . . . .	127
6.18.4.25 knobs [1/2] . . . . .	127
6.18.4.26 knobs [2/2] . . . . .	127
6.18.4.27 led1 . . . . .	128
6.18.4.28 led2 . . . . .	128
6.18.4.29 midi . . . . .	128
6.18.4.30 qspi_handle . . . . .	128
6.18.4.31 ring_led . . . . .	128
6.18.4.32 sai_handle . . . . .	128
6.18.4.33 sdram_handle . . . . .	128
6.18.4.34 seed [1/4] . . . . .	128
6.18.4.35 seed [2/4] . . . . .	129
6.18.4.36 seed [3/4] . . . . .	129
6.18.4.37 seed [4/4] . . . . .	129
6.18.4.38 switches [1/2] . . . . .	129
6.18.4.39 switches [2/2] . . . . .	129
6.18.4.40 usb_handle . . . . .	129
6.19 UTILITY . . . . .	130
6.19.1 Detailed Description . . . . .	132

6.19.2	Macro Definition Documentation	132
6.19.2.1	BSP_SD_CardInfo	132
6.19.2.2	DMA_BUFFER_MEM_SECTION	132
6.19.2.3	DTCM_MEM_SECTION	132
6.19.2.4	MSD_ERROR	132
6.19.2.5	MSD_ERROR_SD_NOT_PRESENT	132
6.19.2.6	MSD_OK	132
6.19.2.7	SD_DATATIMEOUT	133
6.19.2.8	SD_NOT_PRESENT	133
6.19.2.9	SD_PRESENT	133
6.19.2.10	SD_TRANSFER_BUSY	133
6.19.2.11	SD_TRANSFER_OK	133
6.19.3	Enumeration Type Documentation	133
6.19.3.1	dsy_gpio_port	133
6.19.3.2	PresetColor	134
6.19.4	Function Documentation	134
6.19.4.1	Blue()	134
6.19.4.2	BSP_SD_AbortCallback()	134
6.19.4.3	BSP_SD_Erase()	134
6.19.4.4	BSP_SD_GetCardInfo()	135
6.19.4.5	BSP_SD_GetCardState()	135
6.19.4.6	BSP_SD_Init()	135
6.19.4.7	BSP_SD_IsDetected()	136
6.19.4.8	BSP_SD_ITConfig()	136
6.19.4.9	BSP_SD_ReadBlocks()	136
6.19.4.10	BSP_SD_ReadBlocks_DMA()	136
6.19.4.11	BSP_SD_ReadCpltCallback()	137
6.19.4.12	BSP_SD_WriteBlocks()	137
6.19.4.13	BSP_SD_WriteBlocks_DMA()	137
6.19.4.14	BSP_SD_WriteCpltCallback()	138

6.19.4.15 capacity() [1/2]	138
6.19.4.16 capacity() [2/2]	138
6.19.4.17 cube()	138
6.19.4.18 dsy_get_unique_id()	139
6.19.4.19 dsy_hal_map_get_i2c()	139
6.19.4.20 dsy_hal_map_get_pin()	140
6.19.4.21 dsy_hal_map_get_port()	140
6.19.4.22 dsy_pin()	140
6.19.4.23 dsy_pin_cmp()	140
6.19.4.24 Flush() [1/2]	141
6.19.4.25 Flush() [2/2]	141
6.19.4.26 Green()	141
6.19.4.27 ImmediateRead() [1/4]	141
6.19.4.28 ImmediateRead() [2/4]	141
6.19.4.29 ImmediateRead() [3/4]	142
6.19.4.30 ImmediateRead() [4/4]	142
6.19.4.31 Init() [1/4]	142
6.19.4.32 Init() [2/4]	142
6.19.4.33 Init() [3/4]	142
6.19.4.34 Init() [4/4]	143
6.19.4.35 Overwrite() [1/4]	143
6.19.4.36 Overwrite() [2/4]	143
6.19.4.37 Overwrite() [3/4]	144
6.19.4.38 Overwrite() [4/4]	144
6.19.4.39 Read() [1/2]	144
6.19.4.40 Read() [2/2]	144
6.19.4.41 readable() [1/2]	145
6.19.4.42 readable() [2/2]	145
6.19.4.43 Red()	145
6.19.4.44 Swallow()	145

6.19.4.45	writable() [1/2]	145
6.19.4.46	writable() [2/2]	146
6.19.4.47	Write() [1/2]	146
6.19.4.48	Write() [2/2]	146
6.19.5	Variable Documentation	146
6.19.5.1	Font_11x18	147
6.19.5.2	Font_16x26	147
6.19.5.3	Font_6x8	147
6.19.5.4	Font_7x10	147
6.19.5.5	hi2c1	147
6.19.5.6	hi2c2	147
6.19.5.7	hi2c3	147
6.19.5.8	hi2c4	147
6.20	USBD_CDC_IF	148
6.20.1	Detailed Description	148
6.21	USBD_CDC_IF_Exported_Defines	149
6.22	USBD_CDC_IF_Exported_Types	150
6.22.1	Detailed Description	150
6.22.2	Typedef Documentation	150
6.22.2.1	CDC_ReceiveCallback	150
6.23	USBD_CDC_IF_Exported_Macros	151
6.24	USBD_CDC_IF_Exported_Variables	152
6.24.1	Detailed Description	152
6.24.2	Variable Documentation	152
6.24.2.1	USBD_Interface_fops_FS	152
6.24.2.2	USBD_Interface_fops_HS	152
6.25	USBD_CDC_IF_Exported_FunctionsPrototype	153
6.25.1	Detailed Description	153
6.25.2	Function Documentation	153
6.25.2.1	CDC_Set_Rx_Callback_FS()	153



6.25.2.2	CDC_Transmit_FS()	153
6.25.2.3	CDC_Transmit_HS()	153
6.26	USBD_CONF	154
6.26.1	Detailed Description	154
6.27	USBD_CONF_Exported_Variables	155
6.28	USBD_CONF_Exported_Defines	156
6.28.1	Detailed Description	156
6.28.2	Macro Definition Documentation	156
6.28.2.1	DEVICE_FS	156
6.28.2.2	DEVICE_HS	156
6.28.2.3	USBD_DEBUG_LEVEL	156
6.28.2.4	USBD_LPM_ENABLED	157
6.28.2.5	USBD_MAX_NUM_CONFIGURATION	157
6.28.2.6	USBD_MAX_NUM_INTERFACES	157
6.28.2.7	USBD_MAX_STR_DESC_SIZ	157
6.28.2.8	USBD_SELF_POWERED	157
6.28.2.9	USBD_SUPPORT_USER_STRING	157
6.29	USBD_CONF_Exported_Macros	158
6.29.1	Detailed Description	158
6.29.2	Macro Definition Documentation	158
6.29.2.1	USBD_DbgLog	158
6.29.2.2	USBD_Delay	158
6.29.2.3	USBD_ErrLog	159
6.29.2.4	USBD_free	159
6.29.2.5	USBD_malloc	159
6.29.2.6	USBD_memcpy	159
6.29.2.7	USBD_memset	159
6.29.2.8	USBD_UsrLog	159
6.30	USBD_CONF_Exported_Types	160
6.31	USBD_CONF_Exported_FunctionsPrototype	161

6.32	USBD_DESC	162
6.32.1	Detailed Description	162
6.33	USBD_DESC_Exported_Constants	163
6.33.1	Detailed Description	163
6.33.2	Macro Definition Documentation	163
6.33.2.1	DEVICE_ID1	163
6.33.2.2	DEVICE_ID2	163
6.33.2.3	DEVICE_ID3	163
6.33.2.4	USB_SIZ_STRING_SERIAL	163
6.34	USBD_DESC_Exported_Defines	164
6.35	USBD_DESC_Exported_TypesDefinitions	165
6.36	USBD_DESC_Exported_Macros	166
6.37	USBD_DESC_Exported_Variables	167
6.37.1	Detailed Description	167
6.37.2	Variable Documentation	167
6.37.2.1	FS_Desc	167
6.37.2.2	HS_Desc	167
6.38	USBD_DESC_Exported_FunctionsPrototype	168
6.39	Externals	169
6.40	STM32_USB_OTG_DEVICE_LIBRARY	170
6.40.1	Detailed Description	170
6.41	USBD_OTG_DRIVER	171
6.41.1	Detailed Description	171
<b>7</b>	<b>Namespace Documentation</b>	<b>173</b>
7.1	daisy Namespace Reference	173
7.1.1	Detailed Description	174

<b>8</b>	<b>Class Documentation</b>	<b>175</b>
8.1	<a href="#">daisy::AdcChannelConfig Struct Reference</a>	175
8.1.1	<a href="#">Detailed Description</a>	175
8.2	<a href="#">daisy::AdcHandle Class Reference</a>	175
8.2.1	<a href="#">Detailed Description</a>	176
8.3	<a href="#">daisy::AnalogControl Class Reference</a>	176
8.3.1	<a href="#">Detailed Description</a>	177
8.4	<a href="#">codec_frame_t Struct Reference</a>	177
8.4.1	<a href="#">Detailed Description</a>	177
8.4.2	<a href="#">Member Data Documentation</a>	177
8.4.2.1	<a href="#">l</a>	177
8.4.2.2	<a href="#">r</a>	178
8.5	<a href="#">color Struct Reference</a>	178
8.5.1	<a href="#">Detailed Description</a>	178
8.5.2	<a href="#">Member Data Documentation</a>	178
8.5.2.1	<a href="#">blue</a>	178
8.5.2.2	<a href="#">green</a>	178
8.5.2.3	<a href="#">red</a>	179
8.6	<a href="#">daisy::Color Class Reference</a>	179
8.6.1	<a href="#">Detailed Description</a>	179
8.7	<a href="#">daisy::ControlChangeEvent Struct Reference</a>	179
8.7.1	<a href="#">Detailed Description</a>	180
8.8	<a href="#">daisy::daisy_field Struct Reference</a>	180
8.8.1	<a href="#">Detailed Description</a>	180
8.9	<a href="#">daisy::DaisyPatch Class Reference</a>	180
8.9.1	<a href="#">Detailed Description</a>	181
8.10	<a href="#">daisy::DaisyPetal Class Reference</a>	181
8.10.1	<a href="#">Detailed Description</a>	182
8.11	<a href="#">daisy::DaisyPod Class Reference</a>	183
8.11.1	<a href="#">Detailed Description</a>	183

8.12 daisy::DaisySeed Class Reference . . . . .	184
8.12.1 Detailed Description . . . . .	184
8.13 dsy_audio_handle Struct Reference . . . . .	184
8.13.1 Detailed Description . . . . .	185
8.13.2 Member Data Documentation . . . . .	185
8.13.2.1 block_size . . . . .	185
8.13.2.2 dev0_i2c . . . . .	185
8.13.2.3 dev1_i2c . . . . .	185
8.13.2.4 sai . . . . .	185
8.14 dsy_dac_handle Struct Reference . . . . .	185
8.14.1 Detailed Description . . . . .	186
8.14.2 Member Data Documentation . . . . .	186
8.14.2.1 bitdepth . . . . .	186
8.14.2.2 mode . . . . .	186
8.14.2.3 pin_config . . . . .	186
8.15 dsy_gpio Struct Reference . . . . .	186
8.15.1 Detailed Description . . . . .	186
8.15.2 Member Data Documentation . . . . .	187
8.15.2.1 mode . . . . .	187
8.15.2.2 pin . . . . .	187
8.15.2.3 pull . . . . .	187
8.16 dsy_gpio_pin Struct Reference . . . . .	187
8.16.1 Detailed Description . . . . .	187
8.16.2 Member Data Documentation . . . . .	187
8.16.2.1 pin . . . . .	188
8.16.2.2 port . . . . .	188
8.17 dsy_i2c_handle Struct Reference . . . . .	188
8.17.1 Detailed Description . . . . .	188
8.17.2 Member Data Documentation . . . . .	188
8.17.2.1 periph . . . . .	188

8.17.2.2	pin_config	188
8.17.2.3	speed	189
8.18	dsy_qspi_handle Struct Reference	189
8.18.1	Detailed Description	189
8.18.2	Member Data Documentation	189
8.18.2.1	device	189
8.18.2.2	mode	189
8.18.2.3	pin_config	190
8.19	dsy_sai_handle Struct Reference	190
8.19.1	Detailed Description	190
8.19.2	Member Data Documentation	190
8.19.2.1	a_direction	190
8.19.2.2	b_direction	190
8.19.2.3	bitdepth	191
8.19.2.4	device	191
8.19.2.5	init	191
8.19.2.6	sai1_pin_config	191
8.19.2.7	sai2_pin_config	191
8.19.2.8	samplerate	191
8.19.2.9	sync_config	191
8.20	DSY_SD_CardInfoTypeDef Struct Reference	192
8.20.1	Detailed Description	192
8.20.2	Member Data Documentation	192
8.20.2.1	BlockNbr	192
8.20.2.2	BlockSize	192
8.20.2.3	CardSpeed	192
8.20.2.4	CardType	193
8.20.2.5	CardVersion	193
8.20.2.6	Class	193
8.20.2.7	LogBlockNbr	193

8.20.2.8	LogBlockSize	193
8.20.2.9	RelCardAdd	193
8.21	dsy_sdram_handle Struct Reference	193
8.21.1	Detailed Description	194
8.21.2	Member Data Documentation	194
8.21.2.1	pin_config	194
8.21.2.2	state	194
8.22	dsy_sr_4021_handle Struct Reference	194
8.22.1	Detailed Description	195
8.22.2	Member Data Documentation	195
8.22.2.1	clk	195
8.22.2.2	cs	195
8.22.2.3	data	195
8.22.2.4	num_daisy chained	195
8.22.2.5	num_parallel	195
8.22.2.6	pin_config	195
8.22.2.7	states	196
8.23	daisy::Encoder Class Reference	196
8.23.1	Detailed Description	196
8.24	FontDef Struct Reference	196
8.24.1	Detailed Description	197
8.24.2	Member Data Documentation	197
8.24.2.1	data	197
8.24.2.2	FontHeight	197
8.24.2.3	FontWidth	197
8.25	daisy::GateIn Class Reference	197
8.25.1	Detailed Description	198
8.25.2	Constructor & Destructor Documentation	198
8.25.2.1	GateIn()	198
8.25.2.2	~GateIn()	198

8.25.3	Member Function Documentation	198
8.25.3.1	Init()	198
8.25.3.2	Trig()	199
8.26	daisy::Led Class Reference	199
8.26.1	Detailed Description	199
8.27	daisy::MidiEvent Struct Reference	199
8.27.1	Detailed Description	200
8.28	daisy::MidiHandler Class Reference	200
8.28.1	Detailed Description	201
8.29	daisy::NoteOnEvent Struct Reference	201
8.29.1	Detailed Description	201
8.30	daisy::OledDisplay Class Reference	201
8.30.1	Detailed Description	202
8.31	daisy::Parameter Class Reference	202
8.31.1	Detailed Description	202
8.32	daisy::RgbLed Class Reference	203
8.32.1	Detailed Description	203
8.33	daisy::RingBuffer< T, size > Class Template Reference	203
8.33.1	Detailed Description	203
8.34	daisy::RingBuffer< T, 0 > Class Template Reference	204
8.34.1	Detailed Description	204
8.35	ShiftRegister595 Class Reference	204
8.35.1	Detailed Description	205
8.35.2	Member Enumeration Documentation	205
8.35.2.1	Pins	205
8.35.3	Member Function Documentation	205
8.35.3.1	Init()	205
8.35.3.2	Set()	206
8.35.3.3	Write()	206
8.36	daisy::SpiHandle Class Reference	206

8.36.1 Detailed Description . . . . .	206
8.37 daisy::Switch Class Reference . . . . .	207
8.37.1 Detailed Description . . . . .	207
8.38 daisy::UartHandler Class Reference . . . . .	207
8.38.1 Detailed Description . . . . .	208
8.39 UsbHandle Class Reference . . . . .	208
8.39.1 Detailed Description . . . . .	209
8.39.2 Member Typedef Documentation . . . . .	209
8.39.2.1 ReceiveCallback [1/2] . . . . .	209
8.39.2.2 ReceiveCallback [2/2] . . . . .	209
8.39.3 Member Enumeration Documentation . . . . .	209
8.39.3.1 UsbPeriph [1/2] . . . . .	209
8.39.3.2 UsbPeriph [2/2] . . . . .	210
8.39.4 Member Function Documentation . . . . .	210
8.39.4.1 Init() [1/2] . . . . .	210
8.39.4.2 Init() [2/2] . . . . .	210
8.39.4.3 SetReceiveCallback() [1/2] . . . . .	211
8.39.4.4 SetReceiveCallback() [2/2] . . . . .	211
8.39.4.5 TransmitExternal() [1/2] . . . . .	211
8.39.4.6 TransmitExternal() [2/2] . . . . .	212
8.39.4.7 TransmitInternal() [1/2] . . . . .	212
8.39.4.8 TransmitInternal() [2/2] . . . . .	212
8.40 WAV_FormatTypeDef Struct Reference . . . . .	213
8.40.1 Detailed Description . . . . .	213
8.40.2 Member Data Documentation . . . . .	213
8.40.2.1 AudioFormat . . . . .	213
8.40.2.2 BitPerSample . . . . .	213
8.40.2.3 BlockAlign . . . . .	213
8.40.2.4 ByteRate . . . . .	214
8.40.2.5 ChunkId . . . . .	214



8.40.2.6	FileFormat	214
8.40.2.7	FileSize	214
8.40.2.8	NbrChannels	214
8.40.2.9	SampleRate	214
8.40.2.10	SubChunk1ID	214
8.40.2.11	SubChunk1Size	214
8.40.2.12	SubChunk2ID	215
8.40.2.13	SubCHunk2Size	215
8.41	daisy::WavFileInfo Struct Reference	215
8.41.1	Detailed Description	215
8.41.2	Member Data Documentation	215
8.41.2.1	name	215
8.41.2.2	raw_data	216
8.42	daisy::WavPlayer Class Reference	216
8.42.1	Detailed Description	216
8.42.2	Member Function Documentation	216
8.42.2.1	Close()	216
8.42.2.2	GetCurrentFile()	217
8.42.2.3	GetLooping()	217
8.42.2.4	GetNumberFiles()	217
8.42.2.5	Init()	217
8.42.2.6	Open()	217
8.42.2.7	Prepare()	218
8.42.2.8	Restart()	218
8.42.2.9	SetLooping()	218
8.42.2.10	Stream()	218

<b>9 File Documentation</b>	<b>219</b>
9.1 src/ffconf.h File Reference . . . . .	219
9.1.1 Detailed Description . . . . .	220
9.1.2 Macro Definition Documentation . . . . .	220
9.1.2.1 _CODE_PAGE . . . . .	220
9.1.2.2 _FFCONF . . . . .	220
9.1.2.3 _FS_EXFAT . . . . .	221
9.1.2.4 _FS_LOCK . . . . .	221
9.1.2.5 _FS_MINIMIZE . . . . .	221
9.1.2.6 _FS_NOFSINFO . . . . .	221
9.1.2.7 _FS_NORTC . . . . .	221
9.1.2.8 _FS_READONLY . . . . .	221
9.1.2.9 _FS_REENTRANT . . . . .	222
9.1.2.10 _FS_RPATH . . . . .	222
9.1.2.11 _FS_TIMEOUT . . . . .	222
9.1.2.12 _FS_TINY . . . . .	222
9.1.2.13 _LFN_UNICODE . . . . .	222
9.1.2.14 _MAX_LFN . . . . .	222
9.1.2.15 _MAX_SS . . . . .	223
9.1.2.16 _MIN_SS . . . . .	223
9.1.2.17 _MULTI_PARTITION . . . . .	223
9.1.2.18 _NORTC_MDAY . . . . .	223
9.1.2.19 _NORTC_MON . . . . .	223
9.1.2.20 _NORTC_YEAR . . . . .	223
9.1.2.21 _STR_VOLUME_ID . . . . .	224
9.1.2.22 _STRF_ENCODE . . . . .	224
9.1.2.23 _SYNC_t . . . . .	224
9.1.2.24 _USE_CHMOD . . . . .	224
9.1.2.25 _USE_EXPAND . . . . .	224
9.1.2.26 _USE_FASTSEEK . . . . .	224

9.1.2.27	<code>_USE_FIND</code>	225
9.1.2.28	<code>_USE_FORWARD</code>	225
9.1.2.29	<code>_USE_LABEL</code>	225
9.1.2.30	<code>_USE_LFN</code>	225
9.1.2.31	<code>_USE_MKFS</code>	225
9.1.2.32	<code>_USE_STRFUNC</code>	225
9.1.2.33	<code>_USE_TRIM</code>	225
9.1.2.34	<code>_VOLUME_STRS</code>	226
9.1.2.35	<code>_VOLUMES</code>	226
9.1.2.36	<code>ff_free</code>	226
9.1.2.37	<code>ff_malloc</code>	226
9.2	<code>src/hid_gatein.h</code> File Reference	226
9.3	<code>src/hid_wavplayer.h</code> File Reference	227
9.3.1	Macro Definition Documentation	227
9.3.1.1	<code>DSY_WAVPLAYER_H</code>	227
9.3.1.2	<code>WAV_FILENAME_MAX</code>	227
9.4	<code>src/usbd_cdc_if.h</code> File Reference	227
9.4.1	Detailed Description	228
9.5	<code>src/usbd_conf.h</code> File Reference	228
9.5.1	Detailed Description	229
9.6	<code>src/usbd_desc.h</code> File Reference	229
9.6.1	Detailed Description	230
	<b>Index</b>	<b>231</b>



# Chapter 1

## libdaisy

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.)

### 1.1 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.

### 1.1.1 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.

### 1.1.2 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.

### 1.1.3 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 . . . . .	24
EXTERNAL . . . . .	30
PERIPHERAL . . . . .	36
SERIAL . . . . .	37
ANALOG_DIGITAL_CONVERSION . . . . .	51
OTHER . . . . .	59
SYSTEM . . . . .	64
DEVICE . . . . .	66
SHIFTREGISTER . . . . .	67
FLASH . . . . .	69
CODEC . . . . .	97
LED . . . . .	100
SDRAM . . . . .	103
BOARDS . . . . .	105
UTILITY . . . . .	130
Externals . . . . .	169
STM32_USB_OTG_DEVICE_LIBRARY . . . . .	170
USBDCDC_IF . . . . .	148
USBDCDC_IF_Exported_Defines . . . . .	149
USBDCDC_IF_Exported_Types . . . . .	150
USBDCDC_IF_Exported_Macros . . . . .	151
USBDCDC_IF_Exported_Variables . . . . .	152
USBDCDC_IF_Exported_FunctionsPrototype . . . . .	153
USBDESC . . . . .	162
USBDESC_Exported_Constants . . . . .	163
USBDESC_Exported_Defines . . . . .	164
USBDESC_Exported_TypesDefinitions . . . . .	165
USBDESC_Exported_Macros . . . . .	166
USBDESC_Exported_Variables . . . . .	167
USBDESC_Exported_FunctionsPrototype . . . . .	168

USBD_OTG_DRIVER . . . . .	171
USBD_CONF . . . . .	154
USBD_CONF_Exported_Variables . . . . .	155
USBD_CONF_Exported_Defines . . . . .	156
USBD_CONF_Exported_Macros . . . . .	158
USBD_CONF_Exported_Types . . . . .	160
USBD_CONF_Exported_FunctionsPrototype . . . . .	161



## Chapter 3

# Namespace Index

### 3.1 Namespace List

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

<a href="#">daisy</a>	Hardware defines and helpers for daisy field platform . . . . .	<a href="#">173</a>
-----------------------	---	---------------------



## Chapter 4

# Class Index

### 4.1 Class List

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

<a href="#">daisy::AdcChannelConfig</a>	175
<a href="#">daisy::AdcHandle</a>	175
<a href="#">daisy::AnalogControl</a>	
Hardware Interface for control inputs	
Primarily designed for ADC input controls such as	
potentiometers, and control voltage.	
176	
<a href="#">codec_frame_t</a>	177
<a href="#">color</a>	178
<a href="#">daisy::Color</a>	179
<a href="#">daisy::ControlChangeEvent</a>	179
<a href="#">daisy::daisy_field</a>	180
<a href="#">daisy::DaisyPatch</a>	
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	180
<a href="#">daisy::DaisyPetal</a>	
Helpers and hardware definitions for daisy petal	181
<a href="#">daisy::DaisyPod</a>	
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	183
<a href="#">daisy::DaisySeed</a>	
This is the higher-level interface for the Daisy board.	
All basic peripheral configuration/initialization is setup here	184
<a href="#">dsy_audio_handle</a>	184
<a href="#">dsy_dac_handle</a>	185
<a href="#">dsy_gpio</a>	186
<a href="#">dsy_gpio_pin</a>	187
<a href="#">dsy_i2c_handle</a>	188
<a href="#">dsy_qspi_handle</a>	189
<a href="#">dsy_sai_handle</a>	190
<a href="#">DSY_SD_CardInfoTypeDef</a>	192
<a href="#">dsy_sdram_handle</a>	193
<a href="#">dsy_sr_4021_handle</a>	194
<a href="#">daisy::Encoder</a>	
Generic Class for handling Quadrature Encoders	
Inspired/influenced by Mutable Instruments (pichenettes) <a href="#">Encoder</a> classes	196

FontDef	196
daisy::GateIn	
Generic Class for handling gate inputs through GPIO	197
daisy::Led	
LED Class providing simple Software PWM ability, etc	
Eventually this will work with hardware PWM, and external LED Driver devices as well	199
daisy::MidiEvent	199
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	200
daisy::NoteOnEvent	201
daisy::OledDisplay	201
daisy::Parameter	202
daisy::RgbLed	203
daisy::RingBuffer< T, size >	203
daisy::RingBuffer< T, 0 >	204
ShiftRegister595	
Device Driver for 8-bit shift register.	
CD74HC595 - 8-bit serial to parallel output shift	204
daisy::SpiHandle	206
daisy::Switch	207
daisy::UartHandler	207
UsbHandle	
Interface for initializing and using the USB Peripherals on the daisy	208
WAV_FormatTypeDef	213
daisy::WavFileInfo	215
daisy::WavPlayer	216

## Chapter 5

# File Index

### 5.1 File List

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

src/ <b>daisy.h</b>	??
src/ <b>daisy_core.h</b>	??
src/ <b>daisy_field.h</b>	??
src/ <b>daisy_patch.h</b>	??
src/ <b>daisy_petal.h</b>	??
src/ <b>daisy_pod.h</b>	??
src/ <b>daisy_seed.h</b>	??
src/ <b>dev_codec_ak4556.h</b>	??
src/ <b>dev_codec_pcm3060.h</b>	??
src/ <b>dev_codec_wm8731.h</b>	??
src/ <b>dev_codec_wm8731_frame.h</b>	??
src/ <b>dev_flash_IS25LP064A.h</b>	??
src/ <b>dev_flash_IS25LP080D.h</b>	??
src/ <b>dev_leddriver.h</b>	??
src/ <b>dev_sdram.h</b>	??
src/ <b>dev_sr_4021.h</b>	??
src/ <b>dev_sr_595.h</b>	??
src/ <b>fatfs.h</b>	??
src/ <b>ffconf.h</b>	219
src/ <b>hid_audio.h</b>	??
src/ <b>hid_ctrl.h</b>	??
src/ <b>hid_encoder.h</b>	??
src/ <b>hid_gatein.h</b>	226
src/ <b>hid_led.h</b>	??
src/ <b>hid_midi.h</b>	??
src/ <b>hid_oled_display.h</b>	??
src/ <b>hid_parameter.h</b>	??
src/ <b>hid_rgb_led.h</b>	??
src/ <b>hid_switch.h</b>	??
src/ <b>hid_usb.h</b>	??
src/ <b>hid_wavplayer.h</b>	227
src/ <b>per_adc.h</b>	??
src/ <b>per_dac.h</b>	??
src/ <b>per_gpio.h</b>	??
src/ <b>per_i2c.h</b>	??

src/ <b>per_qspi.h</b> . . . . .	??
src/ <b>per_sai.h</b> . . . . .	??
src/ <b>per_sdmmc.h</b> . . . . .	??
src/ <b>per_spi.h</b> . . . . .	??
src/ <b>per_tim.h</b> . . . . .	??
src/ <b>per_uart.h</b> . . . . .	??
src/ <b>stm32h7xx_hal_conf.h</b> . . . . .	??
src/ <b>sys_dma.h</b> . . . . .	??
src/ <b>sys_system.h</b> . . . . .	??
src/ <a href="#">usbd_cdc_if.h</a>	
: Header for usbd_cdc_if.c file . . . . .	<a href="#">227</a>
src/ <a href="#">usbd_conf.h</a>	
: Header for usbd_conf.c file . . . . .	<a href="#">228</a>
src/ <a href="#">usbd_desc.h</a>	
: Header for usbd_conf.c file . . . . .	<a href="#">229</a>
src/ <b>util_bsp_sd_diskio.h</b> . . . . .	??
src/ <b>util_color.h</b> . . . . .	??
src/ <b>util_hal_map.h</b> . . . . .	??
src/ <b>util_oled_fonts.h</b> . . . . .	??
src/ <b>util_ringbuffer.h</b> . . . . .	??
src/ <b>util_sd_diskio.h</b> . . . . .	??
src/ <b>util_unique_id.h</b> . . . . .	??
src/ <b>util_wav_format.h</b> . . . . .	??

## 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 SAs 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::Parameter](#)
- class [daisy::Switch](#)

### Enumerations

- enum [daisy::Parameter::Curve](#) {  
[daisy::Parameter::LINEAR](#), [daisy::Parameter::EXPONENTIAL](#), [daisy::Parameter::LOGARITHMIC](#), [daisy::Parameter::CUBE](#),  
[daisy::Parameter::LAST](#) }
- enum [daisy::Switch::Type](#) { [daisy::Switch::TYPE\\_TOGGLE](#), [daisy::Switch::TYPE\\_MOMENTARY](#) }
- enum [daisy::Switch::Polarity](#) { [daisy::Switch::POLARITY\\_NORMAL](#), [daisy::Switch::POLARITY\\_INVERTED](#) }
- enum [daisy::Switch::Pull](#) { [daisy::Switch::PULL\\_UP](#), [daisy::Switch::PULL\\_DOWN](#), [daisy::Switch::PULL\\_NONE](#) }

### Functions

- [daisy::AnalogControl::AnalogControl](#) ()
- [daisy::AnalogControl::~~AnalogControl](#) ()
- void [daisy::AnalogControl::Init](#) (uint16\_t \*adcptr, float sr, bool flip=false, bool invert=false, float slew\_rate=0.002f)
- void [daisy::AnalogControl::InitBipolarCv](#) (uint16\_t \*adcptr, float sr)
- float [daisy::AnalogControl::Process](#) ()
- float [daisy::AnalogControl::Value](#) () const
- void [daisy::Encoder::Init](#) (dsy\_gpio\_pin a, dsy\_gpio\_pin b, dsy\_gpio\_pin click, float update\_rate)
- void [daisy::Encoder::Debounce](#) ()
- int32\_t [daisy::Encoder::Increment](#) () const
- bool [daisy::Encoder::RisingEdge](#) () const
- bool [daisy::Encoder::FallingEdge](#) () const
- bool [daisy::Encoder::Pressed](#) () const
- float [daisy::Encoder::TimeHeldMs](#) () const
- [daisy::Parameter::Parameter](#) ()
- [daisy::Parameter::~~Parameter](#) ()
- void [daisy::Parameter::Init](#) ([AnalogControl](#) input, float min, float max, [Curve](#) curve)
- float [daisy::Parameter::Process](#) ()
- float [daisy::Parameter::Value](#) ()
- void [daisy::Switch::Init](#) (dsy\_gpio\_pin pin, float update\_rate, [Type](#) t, [Polarity](#) pol, [Pull](#) pu)
- void [daisy::Switch::Init](#) (dsy\_gpio\_pin pin, float update\_rate)
- void [daisy::Switch::Debounce](#) ()
- bool [daisy::Switch::RisingEdge](#) () const
- bool [daisy::Switch::FallingEdge](#) () const
- bool [daisy::Switch::Pressed](#) () const
- float [daisy::Switch::TimeHeldMs](#) () const

### 6.4.1 Detailed Description

Hardware Controls.

### 6.4.2 Enumeration Type Documentation

#### 6.4.2.1 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.

#### 6.4.2.2 Polarity

```
enum daisy::Switch::Polarity
```

Specifies whether the pressed is HIGH or LOW.

Enumerator

POLARITY_NORMAL	&
POLARITY_INVERTED	&

#### 6.4.2.3 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	&

#### 6.4.2.4 Type

```
enum daisy::Switch::Type
```

Specifies the expected behavior of the switch

Enumerator

TYPE_TOGGLE	&
TYPE_MOMENTARY	&

### 6.4.3 Function Documentation

#### 6.4.3.1 AnalogControl()

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

Constructor

#### 6.4.3.2 Debounce() [1/2]

```
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.

#### 6.4.3.3 Debounce() [2/2]

```
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.

#### 6.4.3.4 FallingEdge() [1/2]

```
bool daisy::Encoder::FallingEdge ( ) const [inline]
```

Returns true if the encoder was just released.

**6.4.3.5 FallingEdge()** [2/2]

```
bool daisy::Switch::FallingEdge ( ) const [inline]
```

**Returns**

true if the button was just released

**6.4.3.6 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.

**6.4.3.7 Init()** [1/5]

```
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.

**6.4.3.8 Init()** [2/5]

```
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.

**6.4.3.9 Init()** [3/5]

```
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.

#### 6.4.3.10 Init() [4/5]

```

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 <a href="#">Debounce()</a> function will be called. (used for timing).
<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

#### 6.4.3.11 Init() [5/5]

```

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 <a href="#">Debounce()</a> function will be called. (used for timing).

## 6.4.3.12 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

## 6.4.3.13 Parameter()

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

## Constructor

## 6.4.3.14 Pressed() [1/2]

```
bool daisy::Encoder::Pressed ( ) const [inline]
```

Returns true while the encoder is held down.

## 6.4.3.15 Pressed() [2/2]

```
bool daisy::Switch::Pressed ( ) const [inline]
```

## Returns

true if the button is held down (or if the toggle is on)

**6.4.3.16 Process()** [1/2]

```
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.

**6.4.3.17 Process()** [2/2]

```
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

**6.4.3.18 RisingEdge()** [1/2]

```
bool daisy::Encoder::RisingEdge ( ) const [inline]
```

Returns true if the encoder was just pressed.

**6.4.3.19 RisingEdge()** [2/2]

```
bool daisy::Switch::RisingEdge ( ) const [inline]
```

**Returns**

true if a button was just pressed.

**6.4.3.20 TimeHeldMs()** [1/2]

```
float daisy::Encoder::TimeHeldMs ( ) const [inline]
```

Returns the time in milliseconds that the encoder has been held down.

**6.4.3.21 TimeHeldMs()** [2/2]

```
float daisy::Switch::TimeHeldMs ( ) const [inline]
```

**Returns**

the time in milliseconds that the button has been held (or toggle has been on)

**6.4.3.22 Value()** [1/2]

```
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.

**6.4.3.23 Value()** [2/2]

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

Returns the current stored value, without reprocessing

**6.4.3.24 ~AnalogControl()**

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

destructor

**6.4.3.25 ~Parameter()**

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

Destructor

## 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`

### Enumerations

- enum `daisy::OledDisplay::Pins` { `daisy::OledDisplay::DATA_COMMAND`, `daisy::OledDisplay::RESET`, `daisy::OledDisplay::NUM_PINS` }

### Functions

- void `daisy::Led::Init` (`dsy_gpio_pin` pin, bool invert, float samplerate=1000.0f)
- void `daisy::Led::Set` (float val)
- void `daisy::Led::Update` ()
- void `daisy::OledDisplay::Init` (`dsy_gpio_pin` \*pin\_cfg)
- void `daisy::OledDisplay::Fill` (bool on)
- void `daisy::OledDisplay::DrawPixel` (uint8\_t x, uint8\_t y, bool on)
- char `daisy::OledDisplay::WriteChar` (char ch, `FontDef` font, bool on)
- char `daisy::OledDisplay::WriteString` (char \*str, `FontDef` font, bool on)
- void `daisy::OledDisplay::SetCursor` (uint8\_t x, uint8\_t y)
- void `daisy::OledDisplay::Update` ()
- void `daisy::RgbLed::Init` (`dsy_gpio_pin` red, `dsy_gpio_pin` green, `dsy_gpio_pin` blue, bool invert)
- void `daisy::RgbLed::Set` (float r, float g, float b)
- void `daisy::RgbLed::SetColor` (`Color` c)
- void `daisy::RgbLed::Update` ()

#### 6.5.1 Detailed Description

Screens, leds, etc.

#### 6.5.2 Enumeration Type Documentation

##### 6.5.2.1 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

## 6.5.3 Function Documentation

## 6.5.3.1 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

## 6.5.3.2 Fill()

```
void daisy::OledDisplay::Fill (
    bool on )
```

Fills the entire display with either on/off.

## Parameters

<i>on</i>	Sets on or off.
-----------	-----------------

## 6.5.3.3 Init() [1/3]

```
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

#### 6.5.3.4 Init() [2/3]

```

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)

#### 6.5.3.5 Init() [3/3]

```

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 <a href="#">OledDisplay::NUM_PINS</a> dsy_gpio_pins
----------------	--

#### 6.5.3.6 Set() [1/2]

```

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

#### 6.5.3.7 Set() [2/2]

```
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.
------------	--

#### 6.5.3.8 SetColor()

```
void daisy::RgbLed::SetColor (
    Color c )
```

Sets the RGB using a [Color](#) object.

#### Parameters

<i>c</i>	<a href="#">Color</a> object to set.
----------	--------------------------------------

#### 6.5.3.9 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

**6.5.3.10 Update()** [1/3]

```
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)

**6.5.3.11 Update()** [2/3]

```
void daisy::Led::Update ( )
```

This processes the pwm of the LED sets the hardware accordingly.

**6.5.3.12 Update()** [3/3]

```
void daisy::OledDisplay::Update ( )
```

Writes the current display buffer to the OLED device using SPI or I2C depending on how the object was initialized.

**6.5.3.13 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**

&



#### 6.5.3.14 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

&

## 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](#) }
- enum [daisy::MidiHandler::MidiInputMode](#) { [daisy::MidiHandler::INPUT\\_MODE\\_NONE](#) = 0x00, [daisy::MidiHandler::INPUT\\_MODE\\_UART1](#) = 0x01, [daisy::MidiHandler::INPUT\\_MODE\\_USB\\_INT](#) = 0x02, [daisy::MidiHandler::INPUT\\_MODE\\_USB\\_EXT](#) = 0x04 }
- enum [daisy::MidiHandler::MidiOutputMode](#) { [daisy::MidiHandler::OUTPUT\\_MODE\\_NONE](#) = 0x00, [daisy::MidiHandler::OUTPUT\\_MODE\\_UART1](#) = 0x01, [daisy::MidiHandler::OUTPUT\\_MODE\\_USB\\_INT](#) = 0x02, [daisy::MidiHandler::OUTPUT\\_MODE\\_USB\\_EXT](#) = 0x04 }

### Functions

- [NoteOnEvent](#) [daisy::MidiEvent::AsNoteOn](#) ()
- [ControlChangeEvent](#) [daisy::MidiEvent::AsControlChange](#) ()
- void [daisy::MidiHandler::Init](#) ([MidiInputMode](#) in\_mode, [MidiOutputMode](#) out\_mode)
- void [daisy::MidiHandler::StartReceive](#) ()
- void [daisy::MidiHandler::Listen](#) ()
- void [daisy::MidiHandler::Parse](#) (uint8\_t byte)
- bool [daisy::MidiHandler::HasEvents](#) () const
- [MidiEvent](#) [daisy::MidiHandler::PopEvent](#) ()

### Variables

- int [daisy::NoteOnEvent::channel](#)
- uint8\_t [daisy::NoteOnEvent::note](#)
- uint8\_t [daisy::NoteOnEvent::velocity](#)
- int [daisy::ControlChangeEvent::channel](#)
- uint8\_t [daisy::ControlChangeEvent::control\\_number](#)
- uint8\_t [daisy::ControlChangeEvent::value](#)
- [MidiMessageType](#) [daisy::MidiEvent::type](#)
- int [daisy::MidiEvent::channel](#)
- uint8\_t [daisy::MidiEvent::data](#) [2]

### 6.6.1 Detailed Description

External interface devices.

### 6.6.2 Enumeration Type Documentation

#### 6.6.2.1 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	&

#### 6.6.2.2 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	&

#### 6.6.2.3 MidiOutputMode

```
enum daisy::MidiHandler::MidiOutputMode
```

Output mode

## Enumerator

OUTPUT_MODE_NONE	&
OUTPUT_MODE_UART1	&
OUTPUT_MODE_USB_INT	&
OUTPUT_MODE_USB_EXT	&

## 6.6.3 Function Documentation

## 6.6.3.1 AsControlChange()

```
ControlChangeEvent daisy::MidiEvent::AsControlChange ( ) [inline]
```

Returns the data within the [MidiEvent](#) as a [NoteOnEvent](#) struct.

## 6.6.3.2 AsNoteOn()

```
NoteOnEvent daisy::MidiEvent::AsNoteOn ( ) [inline]
```

Returns the data within the [MidiEvent](#) as a [NoteOnEvent](#) struct

## 6.6.3.3 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.

## 6.6.3.4 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

#### 6.6.3.5 Listen()

```
void daisy::MidiHandler::Listen ( )
```

Start listening

#### 6.6.3.6 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>	&
-------------	---

#### 6.6.3.7 PopEvent()

```
MidiEvent daisy::MidiHandler::PopEvent ( ) [inline]
```

Pops the oldest unhandled [MidiEvent](#) from the internal queue

##### Returns

The event to be handled

#### 6.6.3.8 StartReceive()

```
void daisy::MidiHandler::StartReceive ( )
```

Starts listening on the selected input mode(s). [MidiEvent](#) Queue will begin to fill, and can be checked with

### 6.6.4 Variable Documentation

**6.6.4.1 channel** [1/3]

```
int daisy::NoteOnEvent::channel  
&
```

**6.6.4.2 channel** [2/3]

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

**6.6.4.3 channel** [3/3]

```
int daisy::MidiEvent::channel  
&
```

**6.6.4.4 control\_number**

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

**6.6.4.5 data**

```
uint8_t daisy::MidiEvent::data[2]  
&
```

**6.6.4.6 note**

```
uint8_t daisy::NoteOnEvent::note  
&
```

**6.6.4.7 type**

```
MidiMessageType daisy::MidiEvent::type  
&
```

**6.6.4.8 value**

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

**6.6.4.9 velocity**

```
uint8_t daisy::NoteOnEvent::velocity  
&
```

## 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)
- void `daisy::SpiHandle::Init` ()
- void `daisy::SpiHandle::BlockingTransmit` (uint8\_t \*buff, size\_t size)
- void `daisy::UartHandler::Init` ()
- int `daisy::UartHandler::PollReceive` (uint8\_t \*buff, size\_t size, uint32\_t timeout)
- int `daisy::UartHandler::StartRx` (size\_t size)
- bool `daisy::UartHandler::RxActive` ()
- int `daisy::UartHandler::FlushRx` ()
- int `daisy::UartHandler::PollTx` (uint8\_t \*buff, size\_t size)
- uint8\_t `daisy::UartHandler::PopRx` ()
- size\_t `daisy::UartHandler::Readable` ()
- int `daisy::UartHandler::CheckError` ()

## 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

SPI_PERIPH↔ _1	SPI peripheral 1
SPI_PERIPH↔ _3	SPI peripheral 3
SPI_PERIPH↔ _6	SPI peripheral 3

## 6.8.2.16 SpiPin

```
enum daisy::SpiPin
```

SPI pins

## Enumerator

SPI_PIN_CS	CS pin
SPI_PIN_SCK	SCK pin
SPI_PIN_MOSI	MOSI pin
SPI_PIN_MISO	MISO pin

## 6.8.3 Function Documentation

### 6.8.3.1 BlockingTransmit()

```
void daisy::SpiHandle::BlockingTransmit (
    uint8_t * buff,
    size_t size )
```

Blocking transmit

#### Parameters

<i>*buff</i>	input buffer
<i>size</i>	buffer size

### 6.8.3.2 CheckError()

```
int daisy::UartHandler::CheckError ( )
```

#### Returns

the result of HAL\_UART\_GetError() to the user.

### 6.8.3.3 dsy\_i2c\_init()

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

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

#### Parameters

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

### 6.8.3.4 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

**6.8.3.5 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

**6.8.3.6 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

**6.8.3.7 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

**6.8.3.8 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

**6.8.3.9 dsy\_qspi\_writepage()**

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

Writes a single page to 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

**6.8.3.10 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>	&

**6.8.3.11 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>	&
-------------	---

**6.8.3.12 FlushRx()**

```
int daisy::UartHandler::FlushRx ( )
```

Flushes the Receive Queue

**Returns**

OK or ERROR

**6.8.3.13 Init()** [1/2]

```
void daisy::UartHandler::Init ( )
```

Initializes the UART Peripheral

**6.8.3.14 Init()** [2/2]

```
void daisy::SpiHandle::Init ( )
```

Initializes handler

**6.8.3.15 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

**6.8.3.16 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

**6.8.3.17 PopRx()**

```
uint8_t daisy::UartHandler::PopRx ( )
```

Pops the oldest byte from the FIFO.

**Returns**

Popped byte

**6.8.3.18 Readable()**

```
size_t daisy::UartHandler::Readable ( )
```

Checks if there are any unread bytes in the FIFO

**Returns**

1 or 0 ??

**6.8.3.19 RxActive()**

```
bool daisy::UartHandler::RxActive ( )
```

**Returns**

whether Rx DMA is listening or not.

**6.8.3.20 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

## 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 [daisy::AdcChannelConfig::MuxPin](#) { [daisy::AdcChannelConfig::MUX\\_SEL\\_0](#), [daisy::AdcChannelConfig::MUX\\_SEL\\_1](#), [daisy::AdcChannelConfig::MUX\\_SEL\\_2](#), [daisy::AdcChannelConfig::MUX\\_SEL\\_LAST](#) }
- enum [daisy::AdcHandle::OverSampling](#) { [daisy::AdcHandle::OVS\\_NONE](#), [daisy::AdcHandle::OVS\\_4](#), [daisy::AdcHandle::OVS\\_8](#), [daisy::AdcHandle::OVS\\_16](#), [daisy::AdcHandle::OVS\\_32](#), [daisy::AdcHandle::OVS\\_64](#), [daisy::AdcHandle::OVS\\_128](#), [daisy::AdcHandle::OVS\\_256](#), [daisy::AdcHandle::OVS\\_512](#), [daisy::AdcHandle::OVS\\_1024](#), [daisy::AdcHandle::OVS\\_LAST](#) }
- 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 [daisy::AdcChannelConfig::InitSingle](#) ([dsy\\_gpio\\_pin](#) pin)
- 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)
- void [daisy::AdcHandle::Init](#) ([AdcChannelConfig](#) \*cfg, [size\\_t](#) num\_channels, [OverSampling](#) ovs=[OVS\\_32](#))
- void [daisy::AdcHandle::Start](#) ()
- void [daisy::AdcHandle::Stop](#) ()
- [uint16\\_t](#) [daisy::AdcHandle::Get](#) ([uint8\\_t](#) chn)
- [uint16\\_t](#) \* [daisy::AdcHandle::GetPtr](#) ([uint8\\_t](#) chn)
- [float](#) [daisy::AdcHandle::GetFloat](#) ([uint8\\_t](#) chn)
- [uint16\\_t](#) [daisy::AdcHandle::GetMux](#) ([uint8\\_t](#) chn, [uint8\\_t](#) idx)
- [uint16\\_t](#) \* [daisy::AdcHandle::GetMuxPtr](#) ([uint8\\_t](#) chn, [uint8\\_t](#) idx)
- [float](#) [daisy::AdcHandle::GetMuxFloat](#) ([uint8\\_t](#) chn, [uint8\\_t](#) idx)
- 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)

### Variables

- [dsy\\_gpio](#) [daisy::AdcChannelConfig::pin\\_](#)
- [dsy\\_gpio](#) [daisy::AdcChannelConfig::mux\\_pin\\_](#) [[MUX\\_SEL\\_LAST](#)]
- [uint8\\_t](#) [daisy::AdcChannelConfig::mux\\_channels\\_](#)

### 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

DSY_DAC_MODE_POLLING	Polling mode
DSY_DAC_MODE_LAST	3



#### 6.9.2.4 MuxPin

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

Which pin to use for multiplexing

Enumerator

MUX_SEL_0	&
MUX_SEL_1	&
MUX_SEL_2	&
MUX_SEL_LAST	&

#### 6.9.2.5 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	&

### 6.9.3 Function Documentation

#### 6.9.3.1 dsy\_dac\_init()

```
void dsy_dac_init (
    daisy_dac_handle * dsy_hdac,
    daisy_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

**6.9.3.4 Get()**

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

Single channel getter

**Parameters**

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

**Returns**

Converted value

**6.9.3.5 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

**6.9.3.6 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

**6.9.3.7 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

**6.9.3.8 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

**6.9.3.9 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

## 6.9.3.10 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 <a href="#">AdcChannelConfig</a> of the desired channel
<i>num_channels</i>	number of ADC channels to initialize
<i>ovs</i>	Oversampling amount - Defaults to OVS_32

## 6.9.3.11 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>	&

## 6.9.3.12 InitSingle()

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

Initializes a single ADC pin as an ADC.

## Parameters

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

#### 6.9.3.13 Start()

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

Starts reading from the ADC

#### 6.9.3.14 Stop()

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

Stops reading from the ADC

### 6.9.4 Variable Documentation

#### 6.9.4.1 mux\_channels\_

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

&

#### 6.9.4.2 mux\_pin\_

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

&

#### 6.9.4.3 pin\_

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

&

## 6.10 OTHER

GPIO, timers, and SDMMC.

### Classes

- struct [dsy\\_gpio](#)

### 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](#) }

### 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	&

**6.10.3 Function Documentation****6.10.3.1 dsy\_gpio\_deinit()**

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

Deinitializes the gpio pin

**Parameters**

<code>*<i>p</i></code>	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 SHIFREGISTER

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

<a href="#">DSY_SR_4021_PIN_CS</a>	CS Pin
<a href="#">DSY_SR_4021_PIN_CLK</a>	CLK Pin
<a href="#">DSY_SR_4021_PIN_DATA</a>	DATA pin
<a href="#">DSY_SR_4021_PIN_DATA2</a>	DATA2 Pin, optional
<a href="#">DSY_SR_4021_PIN_LAST</a>	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

<code>reset_pin</code>	should be a <a href="#">dsy_gpio_pin</a> 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

<code>*hi2c</code>	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

<code>*hi2c</code>	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

<code>*hi2c</code>	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

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	&



### 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 <a href="#">dsy_i2c_handle</a> 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](#) }
- enum [daisy::DaisyPatch::Ctrl](#) {  
[CTRL\\_1](#), [CTRL\\_2](#), [CTRL\\_3](#), [CTRL\\_4](#),  
[CTRL\\_LAST](#) }
- enum [daisy::DaisyPatch::GateInput](#) { [GATE\\_IN\\_1](#), [GATE\\_IN\\_2](#), [daisy::DaisyPatch::GATE\\_IN\\_LAST](#) }
- enum [daisy::DaisyPetal::Sw](#) {  
[daisy::DaisyPetal::SW\\_1](#), [daisy::DaisyPetal::SW\\_2](#), [daisy::DaisyPetal::SW\\_3](#), [daisy::DaisyPetal::SW\\_4](#),  
[daisy::DaisyPetal::SW\\_5](#), [daisy::DaisyPetal::SW\\_6](#), [daisy::DaisyPetal::SW\\_7](#), [daisy::DaisyPetal::SW\\_LAST](#) }
- enum [daisy::DaisyPetal::Knob](#) {  
[daisy::DaisyPetal::KNOB\\_1](#), [daisy::DaisyPetal::KNOB\\_2](#), [daisy::DaisyPetal::KNOB\\_3](#), [daisy::DaisyPetal::KNOB\\_4](#),  
[daisy::DaisyPetal::KNOB\\_5](#), [daisy::DaisyPetal::KNOB\\_6](#), [daisy::DaisyPetal::KNOB\\_LAST](#) }
- enum [daisy::DaisyPetal::RingLed](#) {  
[daisy::DaisyPetal::RING\\_LED\\_1](#), [daisy::DaisyPetal::RING\\_LED\\_2](#), [daisy::DaisyPetal::RING\\_LED\\_3](#), [daisy::DaisyPetal::RING\\_LED\\_4](#),  
[daisy::DaisyPetal::RING\\_LED\\_5](#), [daisy::DaisyPetal::RING\\_LED\\_6](#), [daisy::DaisyPetal::RING\\_LED\\_7](#), [daisy::DaisyPetal::RING\\_LED\\_8](#),  
[daisy::DaisyPetal::RING\\_LED\\_LAST](#) }

- enum `daisy::DaisyPetal::FootswitchLed` {  
`daisy::DaisyPetal::FOOTSWITCH_LED_1`, `daisy::DaisyPetal::FOOTSWITCH_LED_2`, `daisy::DaisyPetal::FOOTSWITCH_LED_3`, `daisy::DaisyPetal::FOOTSWITCH_LED_4`,  
`daisy::DaisyPetal::FOOTSWITCH_LED_LAST` }
- enum `daisy::DaisyPod::Sw` { `BUTTON_1`, `daisy::DaisyPod::BUTTON_2`, `daisy::DaisyPod::BUTTON_LAST` }
- enum `daisy::DaisyPod::Knob` { `KNOB_1`, `daisy::DaisyPod::KNOB_2`, `daisy::DaisyPod::KNOB_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)`
- `daisy::DaisyPatch::DaisyPatch ()`
- `daisy::DaisyPatch::~~DaisyPatch ()`
- `void daisy::DaisyPatch::Init ()`
- `void daisy::DaisyPatch::DelayMs (size_t del)`
- `void daisy::DaisyPatch::SetAudioBlockSize (size_t size)`
- `void daisy::DaisyPatch::StartAudio (dsy_audio_mc_callback cb)`
- `void daisy::DaisyPatch::ChangeAudioCallback (dsy_audio_callback cb)`
- `void daisy::DaisyPatch::StartAdc ()`
- `float daisy::DaisyPatch::AudioSampleRate ()`
- `size_t daisy::DaisyPatch::AudioBlockSize ()`
- `float daisy::DaisyPatch::AudioCallbackRate ()`
- `void daisy::DaisyPatch::UpdateAnalogControls ()`
- `float daisy::DaisyPatch::GetCtrlValue (Ctrl k)`
- `void daisy::DaisyPatch::DebounceControls ()`
- `void daisy::DaisyPatch::DisplayControls (bool invert=true)`
- `daisy::DaisyPetal::DaisyPetal ()`
- `daisy::DaisyPetal::~~DaisyPetal ()`
- `void daisy::DaisyPetal::Init ()`
- `void daisy::DaisyPetal::DelayMs (size_t del)`
- `void daisy::DaisyPetal::SetAudioBlockSize (size_t size)`
- `void daisy::DaisyPetal::StartAudio (dsy_audio_callback cb)`
- `void daisy::DaisyPetal::ChangeAudioCallback (dsy_audio_callback cb)`
- `void daisy::DaisyPetal::StartAdc ()`
- `float daisy::DaisyPetal::AudioSampleRate ()`
- `size_t daisy::DaisyPetal::AudioBlockSize ()`
- `float daisy::DaisyPetal::AudioCallbackRate ()`
- `void daisy::DaisyPetal::UpdateAnalogControls ()`
- `float daisy::DaisyPetal::GetKnobValue (Knob k)`
- `float daisy::DaisyPetal::GetExpression ()`
- `void daisy::DaisyPetal::DebounceControls ()`
- `void daisy::DaisyPetal::ClearLeds ()`
- `void daisy::DaisyPetal::UpdateLeds ()`
- `void daisy::DaisyPetal::SetRingLed (RingLed idx, float r, float g, float b)`
- `void daisy::DaisyPetal::SetFootswitchLed (FootswitchLed idx, float bright)`
- `void daisy::DaisyPod::Init ()`
- `void daisy::DaisyPod::DelayMs (size_t del)`
- `void daisy::DaisyPod::SetAudioBlockSize (size_t size)`
- `void daisy::DaisyPod::StartAudio (dsy_audio_callback cb)`
- `void daisy::DaisyPod::ChangeAudioCallback (dsy_audio_callback cb)`
- `void daisy::DaisyPod::StartAdc ()`

- float `daisy::DaisyPod::AudioSampleRate` ()
- size\_t `daisy::DaisyPod::AudioBlockSize` ()
- float `daisy::DaisyPod::AudioCallbackRate` ()
- void `daisy::DaisyPod::UpdateAnalogControls` ()
- float `daisy::DaisyPod::GetKnobValue` (Knob k)
- void `daisy::DaisyPod::DebounceControls` ()
- void `daisy::DaisyPod::ClearLeds` ()
- void `daisy::DaisyPod::UpdateLeds` ()
- void `daisy::DaisySeed::Configure` ()
- void `daisy::DaisySeed::Init` ()
- dsy\_gpio\_pin `daisy::DaisySeed::GetPin` (uint8\_t pin\_idx)
- void `daisy::DaisySeed::StartAudio` (dsy\_audio\_callback cb)
- void `daisy::DaisySeed::SetLed` (bool state)
- void `daisy::DaisySeed::SetTestPoint` (bool state)
- float `daisy::DaisySeed::AudioSampleRate` ()
- void `daisy::DaisySeed::SetAudioBlockSize` (size\_t blocksize)

## Variables

- `daisy::DaisySeed daisy::daisy_field::seed`
- `daisy::Switch daisy::daisy_field::switches [SW_LAST]`
- `dsy_gpio daisy::daisy_field::gate_in`
- `dsy_gpio daisy::daisy_field::gate_out`
- `dsy_sr_4021_handle daisy::daisy_field::keyboard_sr`
- `AnalogControl daisy::daisy_field::knobs [KNOB_LAST]`
- `AnalogControl daisy::daisy_field::cvs [CV_LAST]`
- `DaisySeed daisy::DaisyPatch::seed`
- `Encoder daisy::DaisyPatch::encoder`
- `AnalogControl daisy::DaisyPatch::controls [CTRL_LAST]`
- `GateIn daisy::DaisyPatch::gate_input [GATE_IN_LAST]`
- `MidiHandler daisy::DaisyPatch::midi`
- `OledDisplay daisy::DaisyPatch::display`
- `dsy_gpio daisy::DaisyPatch::gate_output`
- `DaisySeed daisy::DaisyPetal::seed`
- `Encoder daisy::DaisyPetal::encoder`
- `AnalogControl daisy::DaisyPetal::knob [KNOB_LAST]`
- `AnalogControl daisy::DaisyPetal::expression`
- `Switch daisy::DaisyPetal::switches [SW_LAST]`
- `RgbLed daisy::DaisyPetal::ring_led [8]`
- `Led daisy::DaisyPetal::footswitch_led [4]`
- `DaisySeed daisy::DaisyPod::seed`
- `Encoder daisy::DaisyPod::encoder`
- `AnalogControl daisy::DaisyPod::knob1`
- `AnalogControl daisy::DaisyPod::knob2`
- `AnalogControl * daisy::DaisyPod::knobs [KNOB_LAST]`
- `Switch daisy::DaisyPod::button1`
- `Switch daisy::DaisyPod::button2`
- `Switch * daisy::DaisyPod::buttons [BUTTON_LAST]`
- `RgbLed daisy::DaisyPod::led1`
- `RgbLed daisy::DaisyPod::led2`
- `dsy_sdram_handle daisy::DaisySeed::sdram_handle`
- `dsy_qspi_handle daisy::DaisySeed::qspi_handle`
- `dsy_audio_handle daisy::DaisySeed::audio_handle`
- `dsy_sai_handle daisy::DaisySeed::sai_handle`

- [dsy\\_i2c\\_handle daisy::DaisySeed::i2c1\\_handle](#)
- [dsy\\_i2c\\_handle daisy::DaisySeed::i2c2\\_handle](#)
- [AdcHandle daisy::DaisySeed::adc](#)
- [dsy\\_dac\\_handle daisy::DaisySeed::dac\\_handle](#)
- [UsbHandle daisy::DaisySeed::usb\\_handle](#)

### 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	&
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	&

**Enumerator**

LED_KNOB↔ _8	&
LED_SW_1	&
LED_SW_2	&
LED_LAST	&

**6.18.2.5 Ctrl**

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

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

**6.18.2.6 FootswitchLed**

```
enum daisy::DaisyPetal::FootswitchLed
```

footswitch leds

**Enumerator**

FOOTSWITCH_LED_1	&
FOOTSWITCH_LED_2	&
FOOTSWITCH_LED_3	&
FOOTSWITCH_LED_4	&
FOOTSWITCH_LED_LAST	&

**6.18.2.7 GateInput**

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

Daisy patch gate inputs

**Enumerator**

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

**6.18.2.8 Knob** [1/2]

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

Knobs

Enumerator

KNOB_2	&
KNOB_LAST	&

#### 6.18.2.9 Knob [2/2]

```
enum daisy::DaisyPetal::Knob
```

Knobs

Enumerator

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

#### 6.18.2.10 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	&

**6.18.2.11 Sw** [1/2]

```
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

**6.18.2.12 Sw** [2/2]

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

**Switches****Enumerator**

BUTTON_2	&
BUTTON_LAST	&

**6.18.3 Function Documentation****6.18.3.1 AudioBlockSize()** [1/3]

```
size_t daisy::DaisyPod::AudioBlockSize ( )
```

Get block size

**6.18.3.2 AudioBlockSize()** [2/3]

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

Get block size

**6.18.3.3 AudioBlockSize()** [3/3]

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

Get audio block size

**6.18.3.4 AudioCallbackRate()** [1/3]

```
float daisy::DaisyPod::AudioCallbackRate ( )
```

Get callback rate

**6.18.3.5 AudioCallbackRate()** [2/3]

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

Get callback rate

**6.18.3.6 AudioCallbackRate()** [3/3]

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

Get callback rate

**6.18.3.7 AudioSampleRate()** [1/4]

```
float daisy::DaisyPod::AudioSampleRate ( )
```

Get sample rate

**6.18.3.8 AudioSampleRate()** [2/4]

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

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

**6.18.3.9 AudioSampleRate()** [3/4]

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

Get sample rate

**6.18.3.10 AudioSampleRate()** [4/4]

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

Device audio sample rate.

**6.18.3.11 ChangeAudioCallback()** [1/3]

```
void daisy::DaisyPod::ChangeAudioCallback (
    daisy_audio_callback cb )
```

Switch callback functions

## Parameters

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

**6.18.3.12 ChangeAudioCallback()** [2/3]

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

Change to a different callback function.

## Parameters

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

**6.18.3.13 ChangeAudioCallback()** [3/3]

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

Change callback function

## Parameters

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

**6.18.3.14 ClearLeds()** [1/2]

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

Reset Leds

**6.18.3.15 ClearLeds()** [2/2]

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

Turn all leds off

6.18.3.16 `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.&

6.18.3.17 `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
----------	---

< &

< &

< &

< &

< &

< &

< &

< &

< &

< &

< &

< &

< &

< &

< &

< &

< &

< &

**6.18.3.18 DaisyPatch()**

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

Constructor

**6.18.3.19 DaisyPetal()**

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

Constructor

**6.18.3.20 DebounceControls()** [1/3]

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

&

**6.18.3.21 DebounceControls()** [2/3]

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

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

**6.18.3.22 DebounceControls()** [3/3]

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

Debounce inputs.

**6.18.3.23 DelayMs()** [1/3]

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

Wait for a bit

Parameters

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

**6.18.3.24 DelayMs()** [2/3]

```
void daisy::DaisyPatch::DelayMs (
```



```
size_t del )
```

Wait some ms before going on.

#### Parameters

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

#### 6.18.3.25 DelayMs() [3/3]

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

Wait before moving on.

#### Parameters

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

#### 6.18.3.26 DisplayControls()

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

Control the display

#### 6.18.3.27 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

**6.18.3.28 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

**6.18.3.29 GetCtrlValue()**

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

Get value for a partiular control

**Parameters**

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

**6.18.3.30 GetExpression()**

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

&

**6.18.3.31 GetKnobValue()** [1/2]

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

&

**6.18.3.32 GetKnobValue()** [2/2]

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

Get value per knob.

## Parameters

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

## Returns

Floating point knob position.

## 6.18.3.33 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).

## 6.18.3.34 Init() [1/4]

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

Init related stuff.

## 6.18.3.35 Init() [2/4]

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

Initializes the daisy seed, and patch hardware.

## 6.18.3.36 Init() [3/4]

```
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.

## 6.18.3.37 Init() [4/4]

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

Initialize daisy petal

## 6.18.3.38 s162f()

```
FORCE_INLINE float s162f (
    int16_t x )
```

Scales float by  $1/(2^{15})$

**Parameters**

<i>x</i>	Number to be scaled.
----------	----------------------

**Returns**

Scaled number.

$< 1 / (2^{**} 15)$

**6.18.3.39 s242f()**

```
FORCE_INLINE float s242f (
    int32_t x )
```

#  $< 2^{**} 23$

$< 2^{**} 23$

$< 1 / (2^{**} 23)$

**6.18.3.40 SetAudioBlockSize()** [1/4]

```
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.
-------------	--------------------

**6.18.3.41 SetAudioBlockSize()** [2/4]

```
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.
-------------	-------------------

**6.18.3.42 SetAudioBlockSize()** [3/4]

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

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

**6.18.3.43 SetAudioBlockSize()** [4/4]

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

Set size of audio blocks.

**Parameters**

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

**6.18.3.44 SetFootswitchLed()**

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

Set footswitch LED

**Parameters**

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

**6.18.3.45 SetLed()**

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

Sets the state of the built in LED

**6.18.3.46 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

**6.18.3.47 SetTestPoint()**

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

Sets the state of the test point near pin 10

**6.18.3.48 StartAdc()** [1/3]

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

Start analog to digital conversion.

**6.18.3.49 StartAdc()** [2/3]

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

Start analog to digital conversion.

**6.18.3.50 StartAdc()** [3/3]

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

Start analog to digital conversion.

**6.18.3.51 StartAudio()** [1/4]

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

Start audio callback

**Parameters**

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

**6.18.3.52 StartAudio()** [2/4]

```
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.

**6.18.3.53 StartAudio()** [3/4]

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

Start audio output.

**Parameters**

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

**6.18.3.54 StartAudio()** [4/4]

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

Start audio callback

**Parameters**

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

**6.18.3.55 UpdateAnalogControls()** [1/3]

```
void daisy::DaisyPod::UpdateAnalogControls ( )
```

Call at same rate as analog reads for smooth reading.

**6.18.3.56 UpdateAnalogControls()** [2/3]

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

Call at same rate as reading controls for good reads.

#### 6.18.3.57 UpdateAnalogControls() [3/3]

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

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

#### 6.18.3.58 UpdateLeds() [1/2]

```
void daisy::DaisyPod::UpdateLeds ( )
```

Update Leds to set colors

#### 6.18.3.59 UpdateLeds() [2/2]

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

Update Leds to values you had set.

#### 6.18.3.60 ~DaisyPatch()

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

Destructor

#### 6.18.3.61 ~DaisyPetal()

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

Destructor

### 6.18.4 Variable Documentation

#### 6.18.4.1 adc

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

&

#### 6.18.4.2 audio\_handle

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

&



#### 6.18.4.3 button1

```
Switch daisy::DaisyPod::button1
```

&

#### 6.18.4.4 button2

```
Switch daisy::DaisyPod::button2
```

&

#### 6.18.4.5 buttons

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

&

#### 6.18.4.6 controls

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

Array of controls

#### 6.18.4.7 cvs

```
AnalogControl daisy::daisy_field::cvs[CV_LAST]
```

Array of cv ins

#### 6.18.4.8 dac\_handle

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

&

#### 6.18.4.9 display

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

&

#### 6.18.4.10 encoder [1/3]

```
Encoder daisy::DaisyPod::encoder
```

&

#### 6.18.4.11 encoder [2/3]

`Encoder` daisy::DaisyPatch::encoder

`Encoder` object

#### 6.18.4.12 encoder [3/3]

`Encoder` daisy::DaisyPetal::encoder

&

#### 6.18.4.13 expression

`AnalogControl` daisy::DaisyPetal::expression

&

#### 6.18.4.14 footswitch\_led

`Led` daisy::DaisyPetal::footswitch\_led[4]

&

#### 6.18.4.15 gate\_in

`dsy_gpio` daisy::daisy\_field::gate\_in

Gate input.

#### 6.18.4.16 gate\_input

`GateIn` daisy::DaisyPatch::gate\_input[GATE\_IN\_LAST]

Gate inputs

#### 6.18.4.17 gate\_out

`dsy_gpio` daisy::daisy\_field::gate\_out

Gate output

#### 6.18.4.18 gate\_output

`dsy_gpio` daisy::DaisyPatch::gate\_output

&

#### 6.18.4.19 i2c1\_handle

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

&

#### 6.18.4.20 i2c2\_handle

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

&

#### 6.18.4.21 keyboard\_sr

```
dsy_sr_4021_handle daisy::daisy_field::keyboard_sr
```

Keyboard shift register

#### 6.18.4.22 knob

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

&

#### 6.18.4.23 knob1

```
AnalogControl daisy::DaisyPod::knob1
```

&

#### 6.18.4.24 knob2

```
AnalogControl daisy::DaisyPod::knob2
```

&

#### 6.18.4.25 knobs [1/2]

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

&

#### 6.18.4.26 knobs [2/2]

```
AnalogControl daisy::daisy_field::knobs[KNOB_LAST]
```

Array of hardware knobs

#### 6.18.4.27 led1

```
RgbLed daisy::DaisyPod::led1
```

&

#### 6.18.4.28 led2

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

&

#### 6.18.4.29 midi

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

Handles midi

#### 6.18.4.30 qspi\_handle

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

&

#### 6.18.4.31 ring\_led

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

&

#### 6.18.4.32 sai\_handle

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

&

#### 6.18.4.33 sdram\_handle

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

&

#### 6.18.4.34 seed [1/4]

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

Public Members #

**6.18.4.35 seed** [2/4]

```
DaisySeed daisy::DaisyPatch::seed
```

Seed object

**6.18.4.36 seed** [3/4]

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

Daisy seed

**6.18.4.37 seed** [4/4]

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

&

**6.18.4.38 switches** [1/2]

```
daisy::Switch daisy::daisy_field::switches[SW_LAST]
```

Array of hardware switches

**6.18.4.39 switches** [2/2]

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

< &

**6.18.4.40 usb\_handle**

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

&

## 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](#) }
- enum [daisy::Color::PresetColor](#) {  
[daisy::Color::RED](#), [daisy::Color::GREEN](#), [daisy::Color::BLUE](#), [daisy::Color::WHITE](#),  
[daisy::Color::PURPLE](#), [daisy::Color::CYAN](#), [daisy::Color::GOLD](#), [daisy::Color::OFF](#),  
[daisy::Color::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`)
- `void daisy::Color::Init` (`PresetColor c`)
- `void daisy::Color::Init` (`float red`, `float green`, `float blue`)
- `float daisy::Color::Red` (`() const`)
- `float daisy::Color::Green` (`() const`)
- `float daisy::Color::Blue` (`() const`)
- `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 daisy::RingBuffer< T, size >::Init` (`()`)
- `size_t daisy::RingBuffer< T, size >::capacity` (`() const`)
- `size_t daisy::RingBuffer< T, size >::writable` (`() const`)
- `size_t daisy::RingBuffer< T, size >::readable` (`() const`)
- `void daisy::RingBuffer< T, size >::Write` (`T v`)
- `void daisy::RingBuffer< T, size >::Overwrite` (`T v`)
- `T daisy::RingBuffer< T, size >::Read` (`()`)
- `T daisy::RingBuffer< T, size >::ImmediateRead` (`()`)
- `void daisy::RingBuffer< T, size >::Flush` (`()`)
- `void daisy::RingBuffer< T, size >::Swallow` (`size_t n`)
- `void daisy::RingBuffer< T, size >::ImmediateRead` (`T *destination`, `size_t num_elements`)
- `void daisy::RingBuffer< T, size >::Overwrite` (`const T *source`, `size_t num_elements`)
- `void daisy::RingBuffer< T, 0 >::Init` (`()`)
- `size_t daisy::RingBuffer< T, 0 >::capacity` (`() const`)
- `size_t daisy::RingBuffer< T, 0 >::writable` (`() const`)
- `size_t daisy::RingBuffer< T, 0 >::readable` (`() const`)
- `void daisy::RingBuffer< T, 0 >::Write` (`T v`)
- `void daisy::RingBuffer< T, 0 >::Overwrite` (`T v`)
- `T daisy::RingBuffer< T, 0 >::Read` (`()`)
- `T daisy::RingBuffer< T, 0 >::ImmediateRead` (`()`)
- `void daisy::RingBuffer< T, 0 >::Flush` (`()`)
- `void daisy::RingBuffer< T, 0 >::ImmediateRead` (`T *destination`, `size_t num_elements`)
- `void daisy::RingBuffer< T, 0 >::Overwrite` (`const T *source`, `size_t num_elements`)
- `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(".dtcmram_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)100000000)
```

&

## 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	&

### 6.19.3.2 PresetColor

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

List of colors that have a preset RGB value

Enumerator

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

## 6.19.4 Function Documentation

### 6.19.4.1 Blue()

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

Returns the 0-1 value for Blue

### 6.19.4.2 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

### 6.19.4.3 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.

**6.19.4.4 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>	&
-----------------	---

**6.19.4.5 BSP\_SD\_GetCardState()**

```
uint8_t BSP_SD_GetCardState (
    void )
```

**Returns**

card state, ERROR, etc.

**6.19.4.6 BSP\_SD\_Init()**

```
uint8_t BSP_SD_Init (
    void )
```

**Returns**

card state, ERROR, etc.

#### 6.19.4.7 BSP\_SD\_IsDetected()

```
uint8_t BSP_SD_IsDetected (
    void )
```

##### Returns

Is card detected

#### 6.19.4.8 BSP\_SD\_ITConfig()

```
uint8_t BSP_SD_ITConfig (
    void )
```

##### Returns

card state, ERROR, etc.

#### 6.19.4.9 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.

#### 6.19.4.10 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.

## 6.19.4.11 BSP\_SD\_ReadCpltCallback()

```
void BSP_SD_ReadCpltCallback (
    void )
```

Write complete callback

## 6.19.4.12 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.

## 6.19.4.13 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.

**6.19.4.14 BSP\_SD\_WriteCpltCallback()**

```
void BSP_SD_WriteCpltCallback (
    void )
```

Read complete callback

**6.19.4.15 capacity()** [1/2]

```
template<typename T, size_t size>
size_t daisy::RingBuffer< T, size >::capacity ( ) const [inline]
```

**Returns**

The total size of the ring buffer

**6.19.4.16 capacity()** [2/2]

```
template<typename T >
size_t daisy::RingBuffer< T, 0 >::capacity ( ) const [inline]
```

**Returns**

0

**6.19.4.17 cube()**

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

Computes cube.

**Parameters**

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

**Returns** $x^3$ **6.19.4.18 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**

<i>*w0</i>	First pointer
<i>*w1</i>	Second pointer
<i>*w2</i>	Third pointer

**6.19.4.19 dsy\_hal\_map\_get\_i2c()**

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

**Parameters**

<i>*p</i>	<a href="#">dsy_i2c_handle</a> to get
-----------	---------------------------------------

**Returns**

The I2C\_HandleTypeDef for the given \*p

#### 6.19.4.20 dsy\_hal\_map\_get\_pin()

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

##### Parameters

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

##### Returns

HAL GPIO Pin as used in the HAL from a [dsy\\_gpio\\_pin](#) input.

#### 6.19.4.21 dsy\_hal\_map\_get\_port()

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

##### Parameters

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

##### Returns

HAL GPIO\_TypeDef as used in the HAL from a [dsy\\_gpio\\_pin](#) input.

#### 6.19.4.22 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

#### 6.19.4.23 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



**6.19.4.24 Flush()** [1/2]

```
template<typename T, size_t size>
void daisy::RingBuffer< T, size >::Flush ( ) [inline]
```

Flushes unread elements from the ring buffer

**6.19.4.25 Flush()** [2/2]

```
template<typename T >
void daisy::RingBuffer< T, 0 >::Flush ( ) [inline]
```

Flush the buffer

**6.19.4.26 Green()**

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

Returns the 0-1 value for Green

**6.19.4.27 ImmediateRead()** [1/4]

```
template<typename T, size_t size>
T daisy::RingBuffer< T, size >::ImmediateRead ( ) [inline]
```

Reads next element from ring buffer immediately

**Returns**

read value

**6.19.4.28 ImmediateRead()** [2/4]

```
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

**6.19.4.29 ImmediateRead()** [3/4]

```
template<typename T >
T daisy::RingBuffer< T, 0 >::ImmediateRead ( ) [inline]
```

**Returns**

Read value

**6.19.4.30 ImmediateRead()** [4/4]

```
template<typename T >
void daisy::RingBuffer< T, 0 >::ImmediateRead (
    T * destination,
    size_t num_elements ) [inline]
```

**Parameters**

<i>destination</i>	&
<i>num_elements</i>	&

**6.19.4.31 Init()** [1/4]

```
template<typename T, size_t size>
void daisy::RingBuffer< T, size >::Init ( ) [inline]
```

Initializes the Ring Buffer

**6.19.4.32 Init()** [2/4]

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

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

**Parameters**

<i>c</i>	<a href="#">Color</a> to init to
----------	----------------------------------

**6.19.4.33 Init()** [3/4]

```
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

#### 6.19.4.34 Init() [4/4]

```
template<typename T >
void daisy::RingBuffer< T, 0 >::Init ( ) [inline]
```

Initialize ringbuffer

#### 6.19.4.35 Overwrite() [1/4]

```
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
----------	--------------------

#### 6.19.4.36 Overwrite() [2/4]

```
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

**6.19.4.37 Overwrite()** [3/4]

```
template<typename T >
void daisy::RingBuffer< T, 0 >::Overwrite (
    T v ) [inline]
```

**Parameters**

<i>v</i>	Value to overwrite
----------	--------------------

**6.19.4.38 Overwrite()** [4/4]

```
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>	&

**6.19.4.39 Read()** [1/2]

```
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

**6.19.4.40 Read()** [2/2]

```
template<typename T >
T daisy::RingBuffer< T, 0 >::Read ( ) [inline]
```

**Returns**

Read value

**6.19.4.41 readable()** [1/2]

```
template<typename T, size_t size>
size_t daisy::RingBuffer< T, size >::readable ( ) const [inline]
```

**Returns**

number of unread elements in ring buffer

**6.19.4.42 readable()** [2/2]

```
template<typename T >
size_t daisy::RingBuffer< T, 0 >::readable ( ) const [inline]
```

**Returns**

0

**6.19.4.43 Red()**

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

Returns the 0-1 value for Red

**6.19.4.44 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**

$n$	Size of T ?
-----	-------------

**6.19.4.45 writable()** [1/2]

```
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.

**6.19.4.46 writable()** [2/2]

```
template<typename T >
size_t daisy::RingBuffer< T, 0 >::writable ( ) const [inline]
```

**Returns**

0

**6.19.4.47 Write()** [1/2]

```
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**

<i>v</i>	Value to write
----------	----------------

**6.19.4.48 Write()** [2/2]

```
template<typename T >
void daisy::RingBuffer< T, 0 >::Write (
    T v ) [inline]
```

**Parameters**

<i>v</i>	Value to write
----------	----------------

**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 USB\_D\_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 USBD\_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\\_CONF\\_Exported\\_Variables](#)  
*Public variables.*
- [USB\\_CONF\\_Exported\\_Defines](#)  
*Defines for configuration of the Usb device.*
- [USB\\_CONF\\_Exported\\_Macros](#)  
*Aliases.*
- [USB\\_CONF\\_Exported\\_Types](#)  
*Types.*
- [USB\\_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 [USB\\_D\\_malloc](#) malloc
- #define [USB\\_D\\_free](#) free
- #define [USB\\_D\\_memset](#) memset
- #define [USB\\_D\\_memcpy](#) memcpy
- #define [USB\\_D\\_Delay](#) HAL\_Delay
- #define [USB\\_D\\_UsrLog](#)(...)
- #define [USB\\_D\\_ErrLog](#)(...)
- #define [USB\\_D\\_DbgLog](#)(...)

### 6.29.1 Detailed Description

Aliases.

### 6.29.2 Macro Definition Documentation

#### 6.29.2.1 USB\_D\_DbgLog

```
#define USB_D_DbgLog(  
    ... )
```

**Value:**

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

&

#### 6.29.2.2 USB\_D\_Delay

```
#define USB_D_Delay HAL_Delay
```

Alias for delay.

### 6.29.2.3 USBD\_ErrLog

```
#define USBD_ErrLog(  
    ... )
```

**Value:**

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

&

### 6.29.2.4 USBD\_free

```
#define USBD_free free
```

Alias for memory release.

### 6.29.2.5 USBD\_malloc

```
#define USBD_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\\_DDESC\\_Exported\\_Constants](#)  
*Constants.*
- [USB\\_DDESC\\_Exported\\_Defines](#)  
*Defines.*
- [USB\\_DDESC\\_Exported\\_TypesDefinitions](#)  
*Types.*
- [USB\\_DDESC\\_Exported\\_Macros](#)  
*Aliases.*
- [USB\\_DDESC\\_Exported\\_Variables](#)  
*Public variables.*
- [USB\\_DDESC\\_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

- [USBD\\_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 [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 [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](#) pin\_
- [dsy\\_gpio\\_mux\\_pin](#) [[MUX\\_SEL\\_LAST](#)]
- [uint8\\_t](#) mux\_channels\_

#### 8.1.1 Detailed Description

Configuration Structure for a given channel

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

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

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

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](#)

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

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

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.

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

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.

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 strucutre 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

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

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.

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

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](#)

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](https://github.com/afiskon/stm32-ssd1306)) on github for a great starting point. adapted for SSD1309 and H7 by shensley, 2020

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`.

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.

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

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

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

- src/util\_ringbuffer.h

## 8.35 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](#) ()



### 8.35.1 Detailed Description

Device Driver for 8-bit shift register.  
CD74HC595 - 8-bit serial to parallel output shift.

#### Author

shensley

#### Date

May 2020

### 8.35.2 Member Enumeration Documentation

#### 8.35.2.1 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"

### 8.35.3 Member Function Documentation

#### 8.35.3.1 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 <a href="#">dsy_gpio_pin</a> corresponding the the Pins enum above.
<i>num_daisy_chained</i>	(default = 1) is the number of 595 devices daisy chained together.

### 8.35.3.2 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.

### 8.35.3.3 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

## 8.36 daisy::SpiHandle Class Reference

```
#include <per_spi.h>
```

### Public Member Functions

- void [Init](#) ()
- void [BlockingTransmit](#) (uint8\_t \*buff, size\_t size)

### 8.36.1 Detailed Description

Handler for serial peripheral interface

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

- src/per\_spi.h

## 8.37 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

#### 8.37.1 Detailed Description

Generic Class for handling momentary/latching switches  
Inspired/influenced by Mutable Instruments (pichenettes) [Switch](#) classes

#### Author

Stephen Hensley

#### Date

December 2019

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

- [src/hid\\_switch.h](#)

## 8.38 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](#) ()

### 8.38.1 Detailed Description

Uart Peripheral

Author

shensley

Date

March 2020

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

- `src/per_uart.h`

## 8.39 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)

### 8.39.1 Detailed Description

Interface for initializing and using the USB Peripherals on the daisy.

#### Author

Stephen Hensley

#### Date

December 2019

### 8.39.2 Member Typedef Documentation

#### 8.39.2.1 [ReceiveCallback](#) [1/2]

```
typedef void(* UsbHandle::ReceiveCallback) (uint8_t *buff, uint32_t *len)
```

Function called upon reception of a buffer

#### 8.39.2.2 [ReceiveCallback](#) [2/2]

```
typedef void(* UsbHandle::ReceiveCallback) (uint8_t *buff, uint32_t *len)
```

Function called upon reception of a buffer

### 8.39.3 Member Enumeration Documentation

#### 8.39.3.1 [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

**8.39.3.2 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

**8.39.4 Member Function Documentation****8.39.4.1 Init()** [1/2]

```
void UsbHandle::Init (
    UsbPeriph dev )
```

Initializes the specified peripheral(s) as USB CDC Devices

**Parameters**

<i>dev</i>	Device to initialize
------------	----------------------

**8.39.4.2 Init()** [2/2]

```
void UsbHandle::Init (
```

```
UsbPeriph dev )
```

Initializes the specified peripheral(s) as USB CDC Devices

#### Parameters

<i>dev</i>	Device to initialize
------------	----------------------

#### 8.39.4.3 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
-----------	-------------------------------

#### 8.39.4.4 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
-----------	-------------------------------

#### 8.39.4.5 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

#### 8.39.4.6 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

#### 8.39.4.7 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

#### 8.39.4.8 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



## 8.40 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](#)

### 8.40.1 Detailed Description

Helper struct for handling the WAV file format

### 8.40.2 Member Data Documentation

#### 8.40.2.1 AudioFormat

```
uint16_t WAV_FormatTypeDef::AudioFormat
```

&

#### 8.40.2.2 BitPerSample

```
uint16_t WAV_FormatTypeDef::BitPerSample
```

&

#### 8.40.2.3 BlockAlign

```
uint16_t WAV_FormatTypeDef::BlockAlign
```

&

#### 8.40.2.4 ByteRate

```
uint32_t WAV_FormatTypeDef::ByteRate
```

&

#### 8.40.2.5 ChunkId

```
uint32_t WAV_FormatTypeDef::ChunkId
```

&

#### 8.40.2.6 FileFormat

```
uint32_t WAV_FormatTypeDef::FileFormat
```

&

#### 8.40.2.7 FileSize

```
uint32_t WAV_FormatTypeDef::FileSize
```

&

#### 8.40.2.8 NbrChannels

```
uint16_t WAV_FormatTypeDef::NbrChannels
```

&

#### 8.40.2.9 SampleRate

```
uint32_t WAV_FormatTypeDef::SampleRate
```

&

#### 8.40.2.10 SubChunk1ID

```
uint32_t WAV_FormatTypeDef::SubChunk1ID
```

&

#### 8.40.2.11 SubChunk1Size

```
uint32_t WAV_FormatTypeDef::SubChunk1Size
```

&

#### 8.40.2.12 SubChunk2ID

```
uint32_t WAV_FormatTypeDef::SubChunk2ID
```

&

#### 8.40.2.13 SubCHunk2Size

```
uint32_t WAV_FormatTypeDef::SubCHunk2Size
```

&

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

- `src/util_wav_format.h`

## 8.41 daisy::WavFileInfo Struct Reference

```
#include <hid_wavplayer.h>
```

### Public Attributes

- [WAV\\_FormatTypeDef raw\\_data](#)
- `char name [256]`

### 8.41.1 Detailed Description

Struct containing details of Wav File.

### 8.41.2 Member Data Documentation

#### 8.41.2.1 name

```
char daisy::WavFileInfo::name[256]
```

Wav filename

### 8.41.2.2 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](#)

## 8.42 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

### 8.42.1 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.

### 8.42.2 Member Function Documentation

#### 8.42.2.1 Close()

```
int daisy::WavPlayer::Close ( )
```

Closes whatever file is currently open.

Returns

&

#### 8.42.2.2 GetCurrentFile()

```
size_t daisy::WavPlayer::GetCurrentFile ( ) const [inline]
```

##### Returns

currently selected file.

#### 8.42.2.3 GetLooping()

```
bool daisy::WavPlayer::GetLooping ( ) const [inline]
```

##### Returns

Whether the [WavPlayer](#) is looping or not.

#### 8.42.2.4 GetNumberFiles()

```
size_t daisy::WavPlayer::GetNumberFiles ( ) const [inline]
```

##### Returns

The number of files loaded by the [WavPlayer](#)

#### 8.42.2.5 Init()

```
void daisy::WavPlayer::Init ( )
```

Initializes the [WavPlayer](#), loading up to max\_files of wav files from an SD Card.

#### 8.42.2.6 Open()

```
int daisy::WavPlayer::Open (
    size_t sel )
```

Opens the file at index sel for reading.

##### Parameters

<i>sel</i>	File to open
------------	--------------

#### 8.42.2.7 Prepare()

```
void daisy::WavPlayer::Prepare ( )
```

Collects buffer for playback when needed.

#### 8.42.2.8 Restart()

```
void daisy::WavPlayer::Restart ( )
```

Resets the playback position to the beginning of the file immediately

#### 8.42.2.9 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.
-------------	-------------------------

#### 8.42.2.10 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](http://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  $\geq 1$ :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() / function 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 from 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 alname[] 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

- USBDCDC\_TypeDef [USBDCDC\\_Interface\\_fops\\_FS](#)
- USBDCDC\_TypeDef [USBDCDC\\_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](http://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](http://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](http://www.st.com/SLA0044)

# Index

- [\\_CODE\\_PAGE](#)
  - [ffconf.h, 220](#)
- [\\_FFCONF](#)
  - [ffconf.h, 220](#)
- [\\_FS\\_EXFAT](#)
  - [ffconf.h, 220](#)
- [\\_FS\\_LOCK](#)
  - [ffconf.h, 221](#)
- [\\_FS\\_MINIMIZE](#)
  - [ffconf.h, 221](#)
- [\\_FS\\_NOFSINFO](#)
  - [ffconf.h, 221](#)
- [\\_FS\\_NORTC](#)
  - [ffconf.h, 221](#)
- [\\_FS\\_READONLY](#)
  - [ffconf.h, 221](#)
- [\\_FS\\_REENTRANT](#)
  - [ffconf.h, 221](#)
- [\\_FS\\_RPATH](#)
  - [ffconf.h, 222](#)
- [\\_FS\\_TIMEOUT](#)
  - [ffconf.h, 222](#)
- [\\_FS\\_TINY](#)
  - [ffconf.h, 222](#)
- [\\_LFN\\_UNICODE](#)
  - [ffconf.h, 222](#)
- [\\_MAX\\_LFN](#)
  - [ffconf.h, 222](#)
- [\\_MAX\\_SS](#)
  - [ffconf.h, 222](#)
- [\\_MIN\\_SS](#)
  - [ffconf.h, 223](#)
- [\\_MULTI\\_PARTITION](#)
  - [ffconf.h, 223](#)
- [\\_NORTC\\_MDAY](#)
  - [ffconf.h, 223](#)
- [\\_NORTC\\_MON](#)
  - [ffconf.h, 223](#)
- [\\_NORTC\\_YEAR](#)
  - [ffconf.h, 223](#)
- [\\_STRF\\_ENCODE](#)
  - [ffconf.h, 224](#)
- [\\_STR\\_VOLUME\\_ID](#)
  - [ffconf.h, 223](#)
- [\\_SYNC\\_t](#)
  - [ffconf.h, 224](#)
- [\\_USE\\_CHMOD](#)
  - [ffconf.h, 224](#)
- [\\_USE\\_EXPAND](#)
  - [ffconf.h, 224](#)
- [\\_USE\\_FASTSEEK](#)
  - [ffconf.h, 224](#)
- [\\_USE\\_FIND](#)
  - [ffconf.h, 224](#)
- [\\_USE\\_FORWARD](#)
  - [ffconf.h, 225](#)
- [\\_USE\\_LABEL](#)
  - [ffconf.h, 225](#)
- [\\_USE\\_LFN](#)
  - [ffconf.h, 225](#)
- [\\_USE\\_MKFS](#)
  - [ffconf.h, 225](#)
- [\\_USE\\_STRFUNC](#)
  - [ffconf.h, 225](#)
- [\\_USE\\_TRIM](#)
  - [ffconf.h, 225](#)
- [\\_VOLUMES](#)
  - [ffconf.h, 226](#)
- [\\_VOLUME\\_STRS](#)
  - [ffconf.h, 225](#)
- [~AnalogControl](#)
  - [CONTROLS, 23](#)
- [~DaisyPatch](#)
  - [BOARDS, 124](#)
- [~DaisyPetal](#)
  - [BOARDS, 124](#)
- [~GateIn](#)
  - [daisy::GateIn, 198](#)
- [~Parameter](#)
  - [CONTROLS, 23](#)
- [a\\_direction](#)
  - [dsy\\_sai\\_handle, 190](#)
- [ANALOG\\_DIGITAL\\_CONVERSION, 51](#)
  - [dsy\\_dac\\_bitdepth, 52](#)
  - [dsy\\_dac\\_channel, 52](#)
  - [dsy\\_dac\\_init, 53](#)
  - [dsy\\_dac\\_mode, 52](#)
  - [dsy\\_dac\\_start, 54](#)
  - [dsy\\_dac\\_write, 54](#)
  - [Get, 54](#)
  - [GetFloat, 55](#)
  - [GetMux, 55](#)
  - [GetMuxFloat, 55](#)
  - [GetMuxPtr, 56](#)
  - [GetPtr, 56](#)
  - [Init, 56](#)
  - [InitMux, 57](#)
  - [InitSingle, 57](#)

- [mux\\_channels\\_](#), 58
  - [mux\\_pin\\_](#), 58
  - [MuxPin](#), 53
  - [OverSampling](#), 53
  - [pin\\_](#), 58
  - [Start](#), 58
  - [Stop](#), 58
- [AUDIO](#), 13
  - [dsy\\_audio\\_callback](#), 13
  - [dsy\\_audio\\_enter\\_bypass](#), 14
  - [dsy\\_audio\\_exit\\_bypass](#), 14
  - [dsy\\_audio\\_init](#), 14
  - [dsy\\_audio\\_mc\\_callback](#), 13
  - [dsy\\_audio\\_passthru](#), 14
  - [dsy\\_audio\\_set\\_blocksize](#), 14
  - [dsy\\_audio\\_set\\_callback](#), 15
  - [dsy\\_audio\\_set\\_mc\\_callback](#), 15
  - [dsy\\_audio\\_silence](#), 15
  - [dsy\\_audio\\_start](#), 15
  - [dsy\\_audio\\_stop](#), 15
- [adc](#)
  - [BOARDS](#), 124
- [AnalogControl](#)
  - [CONTROLS](#), 18
- [AsControlChange](#)
  - [EXTERNAL](#), 33
- [AsNoteOn](#)
  - [EXTERNAL](#), 33
- [audio\\_handle](#)
  - [BOARDS](#), 124
- [AudioBlockSize](#)
  - [BOARDS](#), 112
- [AudioCallbackRate](#)
  - [BOARDS](#), 113
- [AudioFormat](#)
  - [WAV\\_FormatTypeDef](#), 213
- [AudioSampleRate](#)
  - [BOARDS](#), 113
- [b\\_direction](#)
  - [dsy\\_sai\\_handle](#), 190
- [BLOCK\\_ERASE\\_32K\\_CMD](#)
  - [FLASH](#), 72
- [BOARDS](#), 105
  - [~DaisyPatch](#), 124
  - [~DaisyPetal](#), 124
  - [adc](#), 124
  - [audio\\_handle](#), 124
  - [AudioBlockSize](#), 112
  - [AudioCallbackRate](#), 113
  - [AudioSampleRate](#), 113
  - [button1](#), 124
  - [button2](#), 125
  - [buttons](#), 125
  - [ChangeAudioCallback](#), 113, 114
  - [ClearLeds](#), 114
  - [Configure](#), 114
  - [controls](#), 125
  - [Ctrl](#), 110
  - [cvs](#), 125
  - [dac\\_handle](#), 125
  - [daisy\\_field\\_init](#), 115
  - [DaisyPatch](#), 115
  - [DaisyPetal](#), 116
  - [DebounceControls](#), 116
  - [DelayMs](#), 116, 117
  - [display](#), 125
  - [DisplayControls](#), 117
  - [encoder](#), 125, 126
  - [expression](#), 126
  - [f2s16](#), 117
  - [f2s24](#), 117
  - [footswitch\\_led](#), 126
  - [FootswitchLed](#), 110
  - [gate\\_in](#), 126
  - [gate\\_input](#), 126
  - [gate\\_out](#), 126
  - [gate\\_output](#), 126
  - [GateInput](#), 110
  - [GetCtrlValue](#), 118
  - [GetExpression](#), 118
  - [GetKnobValue](#), 118
  - [GetPin](#), 119
  - [i2c1\\_handle](#), 126
  - [i2c2\\_handle](#), 127
  - [Init](#), 119
  - [keyboard\\_sr](#), 127
  - [Knob](#), 110, 111
  - [knob](#), 127
  - [knob1](#), 127
  - [knob2](#), 127
  - [knobs](#), 127
  - [led1](#), 127
  - [led2](#), 128
  - [midi](#), 128
  - [qspi\\_handle](#), 128
  - [ring\\_led](#), 128
  - [RingLed](#), 111
  - [s162f](#), 119
  - [s242f](#), 120
  - [sai\\_handle](#), 128
  - [sdram\\_handle](#), 128
  - [seed](#), 128, 129
  - [SetAudioBlockSize](#), 120, 121
  - [SetFootswitchLed](#), 121
  - [SetLed](#), 121
  - [SetRingLed](#), 121
  - [SetTestPoint](#), 122
  - [StartAdc](#), 122
  - [StartAudio](#), 122, 123
  - [Sw](#), 111, 112
  - [switches](#), 129
  - [UpdateAnalogControls](#), 123
  - [UpdateLeds](#), 124
  - [usb\\_handle](#), 129
- [BSP\\_SD\\_AbortCallback](#)
  - [UTILITY](#), 134

- BSP\_SD\_CardInfo
  - UTILITY, [132](#)
- BSP\_SD\_Erase
  - UTILITY, [134](#)
- BSP\_SD\_GetCardInfo
  - UTILITY, [135](#)
- BSP\_SD\_GetCardState
  - UTILITY, [135](#)
- BSP\_SD\_ITConfig
  - UTILITY, [136](#)
- BSP\_SD\_Init
  - UTILITY, [135](#)
- BSP\_SD\_IsDetected
  - UTILITY, [135](#)
- BSP\_SD\_ReadBlocks
  - UTILITY, [136](#)
- BSP\_SD\_ReadBlocks\_DMA
  - UTILITY, [136](#)
- BSP\_SD\_ReadCpltCallback
  - UTILITY, [137](#)
- BSP\_SD\_WriteBlocks
  - UTILITY, [137](#)
- BSP\_SD\_WriteBlocks\_DMA
  - UTILITY, [137](#)
- BSP\_SD\_WriteCpltCallback
  - UTILITY, [138](#)
- BitPerSample
  - WAV\_FormatTypeDef, [213](#)
- bitdepth
  - dsy\_dac\_handle, [186](#)
  - dsy\_sai\_handle, [190](#)
- block\_size
  - dsy\_audio\_handle, [185](#)
- BlockAlign
  - WAV\_FormatTypeDef, [213](#)
- BlockNbr
  - DSY\_SD\_CardInfoTypeDef, [192](#)
- BlockSize
  - DSY\_SD\_CardInfoTypeDef, [192](#)
- BlockingTransmit
  - SERIAL, [44](#)
- Blue
  - UTILITY, [134](#)
- blue
  - color, [178](#)
- button1
  - BOARDS, [124](#)
- button2
  - BOARDS, [125](#)
- buttons
  - BOARDS, [125](#)
- ByteRate
  - WAV\_FormatTypeDef, [213](#)
- CDC\_ReceiveCallback
  - USBDCDC\_IF\_Exported\_Types, [150](#)
- CDC\_Set\_Rx\_Callback\_FS
  - USBDCDC\_IF\_Exported\_FunctionsPrototype, [153](#)
- CDC\_Transmit\_FS
  - USBDCDC\_IF\_Exported\_FunctionsPrototype, [153](#)
- CDC\_Transmit\_HS
  - USBDCDC\_IF\_Exported\_FunctionsPrototype, [153](#)
- CLEAR\_FLAG\_STATUS\_REG\_CMD
  - FLASH, [73](#)
- CODEC, [97](#)
  - codec\_ak4556\_init, [97](#)
  - codec\_pcm3060\_init, [98](#)
  - codec\_wm8731\_enter\_bypass, [98](#)
  - codec\_wm8731\_exit\_bypass, [98](#)
  - codec\_wm8731\_init, [98](#)
  - sa\_audio\_callback, [97](#)
- CONTROLS, [16](#)
  - ~AnalogControl, [23](#)
  - ~Parameter, [23](#)
  - AnalogControl, [18](#)
  - Curve, [17](#)
  - Debounce, [18](#)
  - FallingEdge, [18](#)
  - Increment, [19](#)
  - Init, [19](#), [20](#)
  - InitBipolarCv, [21](#)
  - Parameter, [21](#)
  - Polarity, [17](#)
  - Pressed, [21](#)
  - Process, [21](#), [22](#)
  - Pull, [17](#)
  - RisingEdge, [22](#)
  - TimeHeldMs, [22](#)
  - Type, [18](#)
  - Value, [22](#), [23](#)
- capacity
  - UTILITY, [138](#)
- CardSpeed
  - DSY\_SD\_CardInfoTypeDef, [192](#)
- CardType
  - DSY\_SD\_CardInfoTypeDef, [192](#)
- CardVersion
  - DSY\_SD\_CardInfoTypeDef, [193](#)
- ChangeAudioCallback
  - BOARDS, [113](#), [114](#)
- channel
  - EXTERNAL, [34](#), [35](#)
- CheckError
  - SERIAL, [44](#)
- ChunkId
  - WAV\_FormatTypeDef, [214](#)
- Class
  - DSY\_SD\_CardInfoTypeDef, [193](#)
- ClearLeds
  - BOARDS, [114](#)
- clk
  - dsy\_sr\_4021\_handle, [195](#)
- Close
  - daisy::WavPlayer, [216](#)

- codec\_ak4556\_init
  - CODEC, [97](#)
- codec\_frame\_t, [177](#)
  - l, [177](#)
  - r, [177](#)
- codec\_pcm3060\_init
  - CODEC, [98](#)
- codec\_wm8731\_enter\_bypass
  - CODEC, [98](#)
- codec\_wm8731\_exit\_bypass
  - CODEC, [98](#)
- codec\_wm8731\_init
  - CODEC, [98](#)
- color, [178](#)
  - blue, [178](#)
  - green, [178](#)
  - red, [178](#)
- Configure
  - BOARDS, [114](#)
- control\_number
  - EXTERNAL, [35](#)
- controls
  - BOARDS, [125](#)
- cs
  - dsy\_sr\_4021\_handle, [195](#)
- Ctrl
  - BOARDS, [110](#)
- cube
  - UTILITY, [138](#)
- Curve
  - CONTROLS, [17](#)
- cvs
  - BOARDS, [125](#)
- DEVICE\_FS
  - USBD\_CONF\_Exported\_Defines, [156](#)
- DEVICE\_HS
  - USBD\_CONF\_Exported\_Defines, [156](#)
- DEVICE\_ID1
  - USBD\_DESC\_Exported\_Constants, [163](#)
- DEVICE\_ID2
  - USBD\_DESC\_Exported\_Constants, [163](#)
- DEVICE\_ID3
  - USBD\_DESC\_Exported\_Constants, [163](#)
- DEVICE, [66](#)
- DIE\_ERASE\_CMD
  - FLASH, [73](#)
- DMA\_BUFFER\_MEM\_SECTION
  - UTILITY, [132](#)
- DSY\_SD\_CardInfoTypeDef, [192](#)
  - BlockNbr, [192](#)
  - BlockSize, [192](#)
  - CardSpeed, [192](#)
  - CardType, [192](#)
  - CardVersion, [193](#)
  - Class, [193](#)
  - LogBlockNbr, [193](#)
  - LogBlockSize, [193](#)
  - RelCardAdd, [193](#)
- DSY\_SDRAM\_BSS
  - SDRAM, [103](#)
- DSY\_SDRAM\_DATA
  - SDRAM, [103](#)
- DSY\_WAVPLAYER\_H
  - hid\_wavplayer.h, [227](#)
- DTCM\_MEM\_SECTION
  - UTILITY, [132](#)
- DUAL\_IN\_FAST\_PROG\_CMD
  - FLASH, [73](#)
- DUAL\_INOUT\_FAST\_READ\_4\_BYTE\_ADDR\_CMD
  - FLASH, [73](#)
- DUAL\_INOUT\_FAST\_READ\_CMD
  - FLASH, [74](#)
- DUAL\_INOUT\_FAST\_READ\_DTR\_CMD
  - FLASH, [74](#)
- DUAL\_OUT\_FAST\_READ\_4\_BYTE\_ADDR\_CMD
  - FLASH, [74](#)
- DUAL\_OUT\_FAST\_READ\_CMD
  - FLASH, [74](#)
- DUAL\_OUT\_FAST\_READ\_DTR\_CMD
  - FLASH, [75](#)
- dac\_handle
  - BOARDS, [125](#)
- daisy, [173](#)
  - daisy::AdcChannelConfig, [175](#)
  - daisy::AdcHandle, [175](#)
  - daisy::AnalogControl, [176](#)
  - daisy::Color, [179](#)
  - daisy::ControlChangeEvent, [179](#)
  - daisy::DaisyPatch, [180](#)
  - daisy::DaisyPetal, [181](#)
  - daisy::DaisyPod, [183](#)
  - daisy::DaisySeed, [184](#)
  - daisy::Encoder, [196](#)
  - daisy::GateIn, [197](#)
    - ~GateIn, [198](#)
    - GateIn, [198](#)
    - Init, [198](#)
    - Trig, [198](#)
  - daisy::Led, [199](#)
  - daisy::MidiEvent, [199](#)
  - daisy::MidiHandler, [200](#)
  - daisy::NoteOnEvent, [201](#)
  - daisy::OledDisplay, [201](#)
  - daisy::Parameter, [202](#)
  - daisy::RgbLed, [203](#)
  - daisy::RingBuffer< T, 0 >, [204](#)
  - daisy::RingBuffer< T, size >, [203](#)
  - daisy::SpiHandle, [206](#)
  - daisy::Switch, [207](#)
  - daisy::UartHandler, [207](#)
  - daisy::WavFileInfo, [215](#)
    - name, [215](#)
    - raw\_data, [215](#)
  - daisy::WavPlayer, [216](#)
    - Close, [216](#)
    - GetCurrentFile, [216](#)

- GetLooping, [217](#)
- GetNumberFiles, [217](#)
- Init, [217](#)
- Open, [217](#)
- Prepare, [218](#)
- Restart, [218](#)
- SetLooping, [218](#)
- Stream, [218](#)
- daisy::daisy\_field, [180](#)
- daisy\_field\_init
  - BOARDS, [115](#)
- DaisyPatch
  - BOARDS, [115](#)
- DaisyPetal
  - BOARDS, [116](#)
- data
  - dsy\_sr\_4021\_handle, [195](#)
  - EXTERNAL, [35](#)
  - FontDef, [197](#)
- Debounce
  - CONTROLS, [18](#)
- DebounceControls
  - BOARDS, [116](#)
- DelayMs
  - BOARDS, [116](#), [117](#)
- dev0\_i2c
  - dsy\_audio\_handle, [185](#)
- dev1\_i2c
  - dsy\_audio\_handle, [185](#)
- device
  - dsy\_qspi\_handle, [189](#)
  - dsy\_sai\_handle, [191](#)
- display
  - BOARDS, [125](#)
- DisplayControls
  - BOARDS, [117](#)
- DrawPixel
  - FEEDBACK, [25](#)
- dsy\_audio\_bitdepth
  - SERIAL, [39](#)
- dsy\_audio\_callback
  - AUDIO, [13](#)
- dsy\_audio\_device
  - SERIAL, [39](#)
- dsy\_audio\_dir
  - SERIAL, [39](#)
- dsy\_audio\_enter\_bypass
  - AUDIO, [14](#)
- dsy\_audio\_exit\_bypass
  - AUDIO, [14](#)
- dsy\_audio\_handle, [184](#)
  - block\_size, [185](#)
  - dev0\_i2c, [185](#)
  - dev1\_i2c, [185](#)
  - sai, [185](#)
- dsy\_audio\_init
  - AUDIO, [14](#)
- dsy\_audio\_mc\_callback
  - AUDIO, [13](#)
- dsy\_audio\_passthru
  - AUDIO, [14](#)
- dsy\_audio\_sai
  - SERIAL, [39](#)
- dsy\_audio\_samplerate
  - SERIAL, [40](#)
- dsy\_audio\_set\_blocksize
  - AUDIO, [14](#)
- dsy\_audio\_set\_callback
  - AUDIO, [15](#)
- dsy\_audio\_set\_mc\_callback
  - AUDIO, [15](#)
- dsy\_audio\_silence
  - AUDIO, [15](#)
- dsy\_audio\_start
  - AUDIO, [15](#)
- dsy\_audio\_stop
  - AUDIO, [15](#)
- dsy\_audio\_sync
  - SERIAL, [40](#)
- dsy\_dac\_bitdepth
  - ANALOG\_DIGITAL\_CONVERSION, [52](#)
- dsy\_dac\_channel
  - ANALOG\_DIGITAL\_CONVERSION, [52](#)
- dsy\_dac\_handle, [185](#)
  - bitdepth, [186](#)
  - mode, [186](#)
  - pin\_config, [186](#)
- dsy\_dac\_init
  - ANALOG\_DIGITAL\_CONVERSION, [53](#)
- dsy\_dac\_mode
  - ANALOG\_DIGITAL\_CONVERSION, [52](#)
- dsy\_dac\_start
  - ANALOG\_DIGITAL\_CONVERSION, [54](#)
- dsy\_dac\_write
  - ANALOG\_DIGITAL\_CONVERSION, [54](#)
- dsy\_dma\_init
  - SYSTEM, [64](#)
- dsy\_get\_unique\_id
  - UTILITY, [139](#)
- dsy\_gpio, [186](#)
  - mode, [187](#)
  - pin, [187](#)
  - pull, [187](#)
- dsy\_gpio\_deinit
  - OTHER, [60](#)
- dsy\_gpio\_init
  - OTHER, [60](#)
- dsy\_gpio\_mode
  - OTHER, [59](#)
- dsy\_gpio\_pin, [187](#)
  - pin, [187](#)
  - port, [188](#)
- dsy\_gpio\_port
  - UTILITY, [133](#)
- dsy\_gpio\_pull
  - OTHER, [60](#)

- dsy\_gpio\_read
  - OTHER, [61](#)
- dsy\_gpio\_toggle
  - OTHER, [61](#)
- dsy\_gpio\_write
  - OTHER, [61](#)
- dsy\_hal\_map\_get\_i2c
  - UTILITY, [139](#)
- dsy\_hal\_map\_get\_pin
  - UTILITY, [140](#)
- dsy\_hal\_map\_get\_port
  - UTILITY, [140](#)
- dsy\_i2c\_handle, [188](#)
  - periph, [188](#)
  - pin\_config, [188](#)
  - speed, [188](#)
- dsy\_i2c\_init
  - SERIAL, [44](#)
- dsy\_i2c\_periph
  - SERIAL, [40](#)
- dsy\_i2c\_pin
  - SERIAL, [41](#)
- dsy\_i2c\_speed
  - SERIAL, [41](#)
- dsy\_led\_driver\_color\_by\_name
  - LED, [101](#)
- dsy\_led\_driver\_init
  - LED, [101](#)
- dsy\_led\_driver\_set\_led
  - LED, [101](#)
- dsy\_led\_driver\_update
  - LED, [102](#)
- dsy\_pin
  - UTILITY, [140](#)
- dsy\_pin\_cmp
  - UTILITY, [140](#)
- dsy\_qspi\_deinit
  - SERIAL, [44](#)
- dsy\_qspi\_device
  - SERIAL, [41](#)
- dsy\_qspi\_erase
  - SERIAL, [45](#)
- dsy\_qspi\_erasesector
  - SERIAL, [45](#)
- dsy\_qspi\_handle, [189](#)
  - device, [189](#)
  - mode, [189](#)
  - pin\_config, [189](#)
- dsy\_qspi\_init
  - SERIAL, [45](#)
- dsy\_qspi\_mode
  - SERIAL, [42](#)
- dsy\_qspi\_pin
  - SERIAL, [42](#)
- dsy\_qspi\_write
  - SERIAL, [46](#)
- dsy\_qspi\_writepage
  - SERIAL, [46](#)
- dsy\_sai\_handle, [190](#)
  - a\_direction, [190](#)
  - b\_direction, [190](#)
  - bitdepth, [190](#)
  - device, [191](#)
  - init, [191](#)
  - sai1\_pin\_config, [191](#)
  - sai2\_pin\_config, [191](#)
  - samplerate, [191](#)
  - sync\_config, [191](#)
- dsy\_sai\_init
  - SERIAL, [47](#)
- dsy\_sai\_init\_from\_handle
  - SERIAL, [47](#)
- dsy\_sai\_pin
  - SERIAL, [43](#)
- dsy\_sdram\_handle, [193](#)
  - pin\_config, [194](#)
  - state, [194](#)
- dsy\_sdram\_init
  - SDRAM, [104](#)
- dsy\_sdram\_pin
  - SDRAM, [104](#)
- dsy\_sdram\_state
  - SDRAM, [104](#)
- dsy\_sr\_4021\_handle, [194](#)
  - clk, [195](#)
  - cs, [195](#)
  - data, [195](#)
  - num\_daisy chained, [195](#)
  - num\_parallel, [195](#)
  - pin\_config, [195](#)
  - states, [195](#)
- dsy\_sr\_4021\_init
  - SHIFTREGISTER, [68](#)
- dsy\_sr\_4021\_state
  - SHIFTREGISTER, [68](#)
- dsy\_sr\_4021\_update
  - SHIFTREGISTER, [68](#)
- dsy\_system\_delay
  - SYSTEM, [64](#)
- dsy\_system\_getnow
  - SYSTEM, [64](#)
- dsy\_system\_init
  - SYSTEM, [65](#)
- dsy\_system\_jumpton
  - SYSTEM, [65](#)
- dsy\_system\_jumptonqspi
  - SYSTEM, [65](#)
- dsy\_tim\_delay\_ms
  - OTHER, [62](#)
- dsy\_tim\_delay\_tick
  - OTHER, [62](#)
- dsy\_tim\_delay\_us
  - OTHER, [62](#)
- dsy\_tim\_get\_ms
  - OTHER, [62](#)
- dsy\_tim\_get\_tick



- OTHER, [63](#)
- dsy\_tim\_get\_us
  - OTHER, [63](#)
- dsy\_tim\_init
  - OTHER, [63](#)
- dsy\_tim\_start
  - OTHER, [63](#)
- ENTER\_4\_BYTE\_ADDR\_MODE\_CMD
  - FLASH, [75](#)
- ENTER\_QUAD\_CMD
  - FLASH, [75](#)
- EXIT\_4\_BYTE\_ADDR\_MODE\_CMD
  - FLASH, [75](#)
- EXIT\_QUAD\_CMD
  - FLASH, [76](#)
- EXT\_DUAL\_IN\_FAST\_PROG\_CMD
  - FLASH, [76](#)
- EXT\_QUAD\_IN\_FAST\_PROG\_CMD
  - FLASH, [76](#)
- EXTERNAL, [30](#)
  - AsControlChange, [33](#)
  - AsNoteOn, [33](#)
  - channel, [34](#), [35](#)
  - control\_number, [35](#)
  - data, [35](#)
  - HasEvents, [33](#)
  - Init, [33](#)
  - Listen, [34](#)
  - MidiInputMode, [31](#)
  - MidiMessageType, [31](#)
  - MidiOutputMode, [31](#)
  - note, [35](#)
  - Parse, [34](#)
  - PopEvent, [34](#)
  - StartReceive, [34](#)
  - type, [35](#)
  - value, [35](#)
  - velocity, [35](#)
- encoder
  - BOARDS, [125](#), [126](#)
- expression
  - BOARDS, [126](#)
- Externals, [169](#)
- f2s16
  - BOARDS, [117](#)
- f2s24
  - BOARDS, [117](#)
- FAST\_READ\_4\_BYTE\_ADDR\_CMD
  - FLASH, [76](#)
- FAST\_READ\_CMD
  - FLASH, [77](#)
- FAST\_READ\_DTR\_CMD
  - FLASH, [77](#)
- FEEDBACK, [24](#)
  - DrawPixel, [25](#)
  - Fill, [25](#)
  - Init, [25](#), [26](#)
- Pins, [24](#)
- Set, [26](#), [27](#)
- SetColor, [27](#)
- SetCursor, [27](#)
- Update, [28](#)
- WriteChar, [28](#)
- WriteString, [28](#)
- FLASH, [69](#)
  - BLOCK\_ERASE\_32K\_CMD, [72](#)
  - CLEAR\_FLAG\_STATUS\_REG\_CMD, [73](#)
  - DIE\_ERASE\_CMD, [73](#)
  - DUAL\_IN\_FAST\_PROG\_CMD, [73](#)
  - DUAL\_INOUT\_FAST\_READ\_4\_BYTE\_ADDR\_CMD, [73](#)
  - DUAL\_INOUT\_FAST\_READ\_CMD, [74](#)
  - DUAL\_INOUT\_FAST\_READ\_DTR\_CMD, [74](#)
  - DUAL\_OUT\_FAST\_READ\_4\_BYTE\_ADDR\_CMD, [74](#)
  - DUAL\_OUT\_FAST\_READ\_CMD, [74](#)
  - DUAL\_OUT\_FAST\_READ\_DTR\_CMD, [75](#)
  - ENTER\_4\_BYTE\_ADDR\_MODE\_CMD, [75](#)
  - ENTER\_QUAD\_CMD, [75](#)
  - EXIT\_4\_BYTE\_ADDR\_MODE\_CMD, [75](#)
  - EXIT\_QUAD\_CMD, [76](#)
  - EXT\_DUAL\_IN\_FAST\_PROG\_CMD, [76](#)
  - EXT\_QUAD\_IN\_FAST\_PROG\_CMD, [76](#)
  - FAST\_READ\_4\_BYTE\_ADDR\_CMD, [76](#)
  - FAST\_READ\_CMD, [77](#)
  - FAST\_READ\_DTR\_CMD, [77](#)
  - IS25LP064A\_EAR\_HIGHEST\_SEG, [77](#)
  - IS25LP064A\_EAR\_LOWEST\_SEG, [77](#)
  - IS25LP064A\_EAR\_SECOND\_SEG, [77](#)
  - IS25LP064A\_EAR\_THIRD\_SEG, [77](#)
  - IS25LP064A\_EVCR\_DTRP, [78](#)
  - IS25LP064A\_EVCR\_DUAL, [78](#)
  - IS25LP064A\_EVCR\_ODS, [78](#)
  - IS25LP064A\_EVCR\_QUAD, [78](#)
  - IS25LP064A\_EVCR\_RH, [78](#)
  - IS25LP064A\_FSR\_ERERR, [78](#)
  - IS25LP064A\_FSR\_ERSUS, [78](#)
  - IS25LP064A\_FSR\_NBADDR, [78](#)
  - IS25LP064A\_FSR\_PGERR, [79](#)
  - IS25LP064A\_FSR\_PGSUS, [79](#)
  - IS25LP064A\_FSR\_PRERR, [79](#)
  - IS25LP064A\_FSR\_READY, [79](#)
  - IS25LP064A\_NVCR\_DTRP, [79](#)
  - IS25LP064A\_NVCR\_DUAL, [79](#)
  - IS25LP064A\_NVCR\_NB\_DUMMY, [79](#)
  - IS25LP064A\_NVCR\_NBADDR, [79](#)
  - IS25LP064A\_NVCR\_ODS, [80](#)
  - IS25LP064A\_NVCR\_QUAB, [80](#)
  - IS25LP064A\_NVCR\_RH, [80](#)
  - IS25LP064A\_NVCR\_SEGMENT, [80](#)
  - IS25LP064A\_NVCR\_XIP, [80](#)
  - IS25LP064A\_SR\_QE, [80](#)
  - IS25LP064A\_SR\_SRWREN, [80](#)
  - IS25LP064A\_SR\_WIP, [80](#)
  - IS25LP064A\_SR\_WREN, [81](#)

- IS25LP064A\_VCR\_NB\_DUMMY, 81
- IS25LP064A\_VCR\_WRAP, 81
- IS25LP064A\_VCR\_XIP, 81
- IS25LP080D\_EAR\_HIGHEST\_SE, 81
- IS25LP080D\_EAR\_LOWEST\_SEG, 81
- IS25LP080D\_EAR\_SECOND\_SEG, 81
- IS25LP080D\_EAR\_THIRD\_SEG, 82
- IS25LP080D\_EVCR\_DTRP, 82
- IS25LP080D\_EVCR\_DUAL, 82
- IS25LP080D\_EVCR\_ODS, 82
- IS25LP080D\_EVCR\_QUAD, 82
- IS25LP080D\_EVCR\_RH, 82
- IS25LP080D\_FSR\_ERERR, 82
- IS25LP080D\_FSR\_ERSUS, 82
- IS25LP080D\_FSR\_NBADDR, 83
- IS25LP080D\_FSR\_PGERR, 83
- IS25LP080D\_FSR\_PGSUS, 83
- IS25LP080D\_FSR\_PRERR, 83
- IS25LP080D\_FSR\_READY, 83
- IS25LP080D\_NVCR\_DTRP, 83
- IS25LP080D\_NVCR\_DUAL, 83
- IS25LP080D\_NVCR\_NB\_DUMMY, 83
- IS25LP080D\_NVCR\_NBADDR, 84
- IS25LP080D\_NVCR\_ODS, 84
- IS25LP080D\_NVCR\_QUAB, 84
- IS25LP080D\_NVCR\_RH, 84
- IS25LP080D\_NVCR\_SEGMENT, 84
- IS25LP080D\_NVCR\_XIP, 84
- IS25LP080D\_SR\_QE, 84
- IS25LP080D\_SR\_SRWREN, 84
- IS25LP080D\_SR\_WIP, 85
- IS25LP080D\_SR\_WREN, 85
- IS25LP080D\_VCR\_NB\_DUMMY, 85
- IS25LP080D\_VCR\_WRAP, 85
- IS25LP080D\_VCR\_XIP, 85
- MULTIPLE\_IO\_READ\_ID\_CMD, 85
- PAGE\_PROG\_4\_BYTE\_ADDR\_CMD, 86
- PAGE\_PROG\_CMD, 86
- PROG\_ERASE\_RESUME\_CMD, 86
- PROG\_ERASE\_SUSPEND\_CMD, 86
- PROG\_OTP\_ARRAY\_CMD, 87
- QUAD\_IN\_FAST\_PROG\_4\_BYTE\_ADDR\_CMD, 87
- QUAD\_IN\_FAST\_PROG\_CMD, 87
- QUAD\_INOUT\_FAST\_READ\_4\_BYTE\_ADDR\_CMD, 87
- QUAD\_INOUT\_FAST\_READ\_CMD, 88
- QUAD\_INOUT\_FAST\_READ\_DTR\_CMD, 88
- QUAD\_OUT\_FAST\_READ\_4\_BYTE\_ADDR\_CMD, 88
- QUAD\_OUT\_FAST\_READ\_CMD, 88
- QUAD\_OUT\_FAST\_READ\_DTR\_CMD, 89
- READ\_4\_BYTE\_ADDR\_CMD, 89
- READ\_CMD, 89
- READ\_ENHANCED\_VOL\_CFG\_REG\_CMD, 89
- READ\_EXT\_ADDR\_REG\_CMD, 90
- READ\_FLAG\_STATUS\_REG\_CMD, 90
- READ\_ID\_CMD2, 90
- READ\_ID\_CMD, 90
- READ\_LOCK\_REG\_CMD, 91
- READ\_NONVOL\_CFG\_REG\_CMD, 91
- READ\_OTP\_ARRAY\_CMD, 91
- READ\_READ\_PARAM\_REG\_CMD, 91
- READ\_SERIAL\_FLASH\_DISCO\_PARAM\_CMD, 92
- READ\_STATUS\_REG\_CMD, 92
- RESET\_ENABLE\_CMD, 92
- RESET\_MEMORY\_CMD, 92
- SECTOR\_ERASE\_4\_BYTE\_ADDR\_CMD, 93
- SECTOR\_ERASE\_CMD, 93
- SUBSECTOR\_ERASE\_4\_BYTE\_ADDR\_CMD, 93
- SUBSECTOR\_ERASE\_CMD, 93
- SUBSECTOR\_ERASE\_QPI\_CMD, 94
- WRITE\_DISABLE\_CMD, 94
- WRITE\_ENABLE\_CMD, 94
- WRITE\_ENHANCED\_VOL\_CFG\_REG\_CMD, 94
- WRITE\_EXT\_ADDR\_REG\_CMD, 95
- WRITE\_LOCK\_REG\_CMD, 95
- WRITE\_NONVOL\_CFG\_REG\_CMD, 95
- WRITE\_READ\_PARAM\_REG\_CMD, 95
- WRITE\_STATUS\_REG\_CMD, 96
- FS\_Desc
  - USB\_D\_DESC\_Exported\_Variables, 167
- FallingEdge
  - CONTROLS, 18
- ff\_free
  - ffconf.h, 226
- ff\_malloc
  - ffconf.h, 226
- ffconf.h
  - \_CODE\_PAGE, 220
  - \_FFCONF, 220
  - \_FS\_EXFAT, 220
  - \_FS\_LOCK, 221
  - \_FS\_MINIMIZE, 221
  - \_FS\_NOFSINFO, 221
  - \_FS\_NORTC, 221
  - \_FS\_READONLY, 221
  - \_FS\_REENTRANT, 221
  - \_FS\_RPATH, 222
  - \_FS\_TIMEOUT, 222
  - \_FS\_TINY, 222
  - \_LFN\_UNICODE, 222
  - \_MAX\_LFN, 222
  - \_MAX\_SS, 222
  - \_MIN\_SS, 223
  - \_MULTI\_PARTITION, 223
  - \_NORTC\_MDAY, 223
  - \_NORTC\_MON, 223
  - \_NORTC\_YEAR, 223
  - \_STRF\_ENCODE, 224
  - \_STR\_VOLUME\_ID, 223
  - \_SYNC\_t, 224
  - \_USE\_CHMOD, 224
  - \_USE\_EXPAND, 224
  - \_USE\_FASTSEEK, 224

- [\\_USE\\_FIND](#), [224](#)
- [\\_USE\\_FORWARD](#), [225](#)
- [\\_USE\\_LABEL](#), [225](#)
- [\\_USE\\_LFN](#), [225](#)
- [\\_USE\\_MKFS](#), [225](#)
- [\\_USE\\_STRFUNC](#), [225](#)
- [\\_USE\\_TRIM](#), [225](#)
- [\\_VOLUMES](#), [226](#)
- [\\_VOLUME\\_STRS](#), [225](#)
- [ff\\_free](#), [226](#)
- [ff\\_malloc](#), [226](#)
- FileFormat
  - [WAV\\_FormatTypeDef](#), [214](#)
- FileSize
  - [WAV\\_FormatTypeDef](#), [214](#)
- Fill
  - [FEEDBACK](#), [25](#)
- Flush
  - [UTILITY](#), [140](#), [141](#)
- FlushRx
  - [SERIAL](#), [47](#)
- Font\_11x18
  - [UTILITY](#), [146](#)
- Font\_16x26
  - [UTILITY](#), [147](#)
- Font\_6x8
  - [UTILITY](#), [147](#)
- Font\_7x10
  - [UTILITY](#), [147](#)
- FontDef, [196](#)
  - [data](#), [197](#)
  - [FontHeight](#), [197](#)
  - [FontWidth](#), [197](#)
- FontHeight
  - [FontDef](#), [197](#)
- FontWidth
  - [FontDef](#), [197](#)
- footswitch\_led
  - [BOARDS](#), [126](#)
- FootswitchLed
  - [BOARDS](#), [110](#)
- gate\_in
  - [BOARDS](#), [126](#)
- gate\_input
  - [BOARDS](#), [126](#)
- gate\_out
  - [BOARDS](#), [126](#)
- gate\_output
  - [BOARDS](#), [126](#)
- Gateln
  - [daisy::Gateln](#), [198](#)
- Gatelnput
  - [BOARDS](#), [110](#)
- Get
  - [ANALOG\\_DIGITAL\\_CONVERSION](#), [54](#)
- GetCtrlValue
  - [BOARDS](#), [118](#)
- GetCurrentFile
  - [daisy::WavPlayer](#), [216](#)
- GetExpression
  - [BOARDS](#), [118](#)
- GetFloat
  - [ANALOG\\_DIGITAL\\_CONVERSION](#), [55](#)
- GetKnobValue
  - [BOARDS](#), [118](#)
- GetLooping
  - [daisy::WavPlayer](#), [217](#)
- GetMux
  - [ANALOG\\_DIGITAL\\_CONVERSION](#), [55](#)
- GetMuxFloat
  - [ANALOG\\_DIGITAL\\_CONVERSION](#), [55](#)
- GetMuxPtr
  - [ANALOG\\_DIGITAL\\_CONVERSION](#), [56](#)
- GetNumberFiles
  - [daisy::WavPlayer](#), [217](#)
- GetPin
  - [BOARDS](#), [119](#)
- GetPtr
  - [ANALOG\\_DIGITAL\\_CONVERSION](#), [56](#)
- Green
  - [UTILITY](#), [141](#)
- green
  - [color](#), [178](#)
- HS\_Desc
  - [USB\\_DESCRIPTOR\\_Exported\\_Variables](#), [167](#)
- HUMAN\_INTERFACE, [12](#)
- HasEvents
  - [EXTERNAL](#), [33](#)
- hi2c1
  - [UTILITY](#), [147](#)
- hi2c2
  - [UTILITY](#), [147](#)
- hi2c3
  - [UTILITY](#), [147](#)
- hi2c4
  - [UTILITY](#), [147](#)
- hid\_wavplayer.h
  - [DSY\\_WAVPLAYER\\_H](#), [227](#)
  - [WAV\\_FILENAME\\_MAX](#), [227](#)
- i2c1\_handle
  - [BOARDS](#), [126](#)
- i2c2\_handle
  - [BOARDS](#), [127](#)
- IS25LP064A\_EAR\_HIGHEST\_SE
  - [FLASH](#), [77](#)
- IS25LP064A\_EAR\_LOWEST\_SEG
  - [FLASH](#), [77](#)
- IS25LP064A\_EAR\_SECOND\_SEG
  - [FLASH](#), [77](#)
- IS25LP064A\_EAR\_THIRD\_SEG
  - [FLASH](#), [77](#)
- IS25LP064A\_EVCR\_DTRP
  - [FLASH](#), [78](#)
- IS25LP064A\_EVCR\_DUAL
  - [FLASH](#), [78](#)

IS25LP064A\_EVCR\_ODS  
FLASH, [78](#)

IS25LP064A\_EVCR\_QUAD  
FLASH, [78](#)

IS25LP064A\_EVCR\_RH  
FLASH, [78](#)

IS25LP064A\_FSR\_ERERR  
FLASH, [78](#)

IS25LP064A\_FSR\_ERSUS  
FLASH, [78](#)

IS25LP064A\_FSR\_NBADDR  
FLASH, [78](#)

IS25LP064A\_FSR\_PGERR  
FLASH, [79](#)

IS25LP064A\_FSR\_PGSUS  
FLASH, [79](#)

IS25LP064A\_FSR\_PRERR  
FLASH, [79](#)

IS25LP064A\_FSR\_READY  
FLASH, [79](#)

IS25LP064A\_NVCR\_DTRP  
FLASH, [79](#)

IS25LP064A\_NVCR\_DUAL  
FLASH, [79](#)

IS25LP064A\_NVCR\_NB\_DUMMY  
FLASH, [79](#)

IS25LP064A\_NVCR\_NBADDR  
FLASH, [79](#)

IS25LP064A\_NVCR\_ODS  
FLASH, [80](#)

IS25LP064A\_NVCR\_QUAB  
FLASH, [80](#)

IS25LP064A\_NVCR\_RH  
FLASH, [80](#)

IS25LP064A\_NVCR\_SEGMENT  
FLASH, [80](#)

IS25LP064A\_NVCR\_XIP  
FLASH, [80](#)

IS25LP064A\_SR\_QE  
FLASH, [80](#)

IS25LP064A\_SR\_SRWREN  
FLASH, [80](#)

IS25LP064A\_SR\_WIP  
FLASH, [80](#)

IS25LP064A\_SR\_WREN  
FLASH, [81](#)

IS25LP064A\_VCR\_NB\_DUMMY  
FLASH, [81](#)

IS25LP064A\_VCR\_WRAP  
FLASH, [81](#)

IS25LP064A\_VCR\_XIP  
FLASH, [81](#)

IS25LP080D\_EAR\_HIGHEST\_SE  
FLASH, [81](#)

IS25LP080D\_EAR\_LOWEST\_SEG  
FLASH, [81](#)

IS25LP080D\_EAR\_SECOND\_SEG  
FLASH, [81](#)

IS25LP080D\_EAR\_THIRD\_SEG  
FLASH, [82](#)

IS25LP080D\_EVCR\_DTRP  
FLASH, [82](#)

IS25LP080D\_EVCR\_DUAL  
FLASH, [82](#)

IS25LP080D\_EVCR\_ODS  
FLASH, [82](#)

IS25LP080D\_EVCR\_QUAD  
FLASH, [82](#)

IS25LP080D\_EVCR\_RH  
FLASH, [82](#)

IS25LP080D\_FSR\_ERERR  
FLASH, [82](#)

IS25LP080D\_FSR\_ERSUS  
FLASH, [82](#)

IS25LP080D\_FSR\_NBADDR  
FLASH, [83](#)

IS25LP080D\_FSR\_PGERR  
FLASH, [83](#)

IS25LP080D\_FSR\_PGSUS  
FLASH, [83](#)

IS25LP080D\_FSR\_PRERR  
FLASH, [83](#)

IS25LP080D\_FSR\_READY  
FLASH, [83](#)

IS25LP080D\_NVCR\_DTRP  
FLASH, [83](#)

IS25LP080D\_NVCR\_DUAL  
FLASH, [83](#)

IS25LP080D\_NVCR\_NB\_DUMMY  
FLASH, [83](#)

IS25LP080D\_NVCR\_NBADDR  
FLASH, [84](#)

IS25LP080D\_NVCR\_ODS  
FLASH, [84](#)

IS25LP080D\_NVCR\_QUAB  
FLASH, [84](#)

IS25LP080D\_NVCR\_RH  
FLASH, [84](#)

IS25LP080D\_NVCR\_SEGMENT  
FLASH, [84](#)

IS25LP080D\_NVCR\_XIP  
FLASH, [84](#)

IS25LP080D\_SR\_QE  
FLASH, [84](#)

IS25LP080D\_SR\_SRWREN  
FLASH, [84](#)

IS25LP080D\_SR\_WIP  
FLASH, [85](#)

IS25LP080D\_SR\_WREN  
FLASH, [85](#)

IS25LP080D\_VCR\_NB\_DUMMY  
FLASH, [85](#)

IS25LP080D\_VCR\_WRAP  
FLASH, [85](#)

IS25LP080D\_VCR\_XIP  
FLASH, [85](#)

- ImmediateRead
  - UTILITY, [141](#), [142](#)
- Increment
  - CONTROLS, [19](#)
- Init
  - ANALOG\_DIGITAL\_CONVERSION, [56](#)
  - BOARDS, [119](#)
  - CONTROLS, [19](#), [20](#)
  - daisy::GateIn, [198](#)
  - daisy::WavPlayer, [217](#)
  - EXTERNAL, [33](#)
  - FEEDBACK, [25](#), [26](#)
  - SERIAL, [48](#)
  - ShiftRegister595, [205](#)
  - UTILITY, [142](#), [143](#)
  - UsbHandle, [210](#)
- init
  - dsy\_sai\_handle, [191](#)
- InitBipolarCv
  - CONTROLS, [21](#)
- InitMux
  - ANALOG\_DIGITAL\_CONVERSION, [57](#)
- InitSingle
  - ANALOG\_DIGITAL\_CONVERSION, [57](#)
- kUartMaxBufferSize
  - SERIAL, [50](#)
- keyboard\_sr
  - BOARDS, [127](#)
- Knob
  - BOARDS, [110](#), [111](#)
- knob
  - BOARDS, [127](#)
- knob1
  - BOARDS, [127](#)
- knob2
  - BOARDS, [127](#)
- knobs
  - BOARDS, [127](#)
- I
  - codec\_frame\_t, [177](#)
- LED, [100](#)
  - dsy\_led\_driver\_color\_by\_name, [101](#)
  - dsy\_led\_driver\_init, [101](#)
  - dsy\_led\_driver\_set\_led, [101](#)
  - dsy\_led\_driver\_update, [102](#)
- LIBDAISY, [11](#)
- led1
  - BOARDS, [127](#)
- led2
  - BOARDS, [128](#)
- Listen
  - EXTERNAL, [34](#)
- LogBlockNbr
  - DSY\_SD\_CardInfoTypeDef, [193](#)
- LogBlockSize
  - DSY\_SD\_CardInfoTypeDef, [193](#)
- MSD\_ERROR\_SD\_NOT\_PRESENT
  - UTILITY, [132](#)
- MSD\_ERROR
  - UTILITY, [132](#)
- MSD\_OK
  - UTILITY, [132](#)
- MULTIPLE\_IO\_READ\_ID\_CMD
  - FLASH, [85](#)
- midi
  - BOARDS, [128](#)
- MidiInputMode
  - EXTERNAL, [31](#)
- MidiMessageType
  - EXTERNAL, [31](#)
- MidiOutputMode
  - EXTERNAL, [31](#)
- mode
  - dsy\_dac\_handle, [186](#)
  - dsy\_gpio, [187](#)
  - dsy\_qspi\_handle, [189](#)
- mux\_channels\_
  - ANALOG\_DIGITAL\_CONVERSION, [58](#)
- mux\_pin\_
  - ANALOG\_DIGITAL\_CONVERSION, [58](#)
- MuxPin
  - ANALOG\_DIGITAL\_CONVERSION, [53](#)
- name
  - daisy::WavFileInfo, [215](#)
- NbrChannels
  - WAV\_FormatTypeDef, [214](#)
- note
  - EXTERNAL, [35](#)
- num\_daisy chained
  - dsy\_sr\_4021\_handle, [195](#)
- num\_parallel
  - dsy\_sr\_4021\_handle, [195](#)
- OTHER, [59](#)
  - dsy\_gpio\_deinit, [60](#)
  - dsy\_gpio\_init, [60](#)
  - dsy\_gpio\_mode, [59](#)
  - dsy\_gpio\_pull, [60](#)
  - dsy\_gpio\_read, [61](#)
  - dsy\_gpio\_toggle, [61](#)
  - dsy\_gpio\_write, [61](#)
  - dsy\_tim\_delay\_ms, [62](#)
  - dsy\_tim\_delay\_tick, [62](#)
  - dsy\_tim\_delay\_us, [62](#)
  - dsy\_tim\_get\_ms, [62](#)
  - dsy\_tim\_get\_tick, [63](#)
  - dsy\_tim\_get\_us, [63](#)
  - dsy\_tim\_init, [63](#)
  - dsy\_tim\_start, [63](#)
- Open
  - daisy::WavPlayer, [217](#)
- OverSampling
  - ANALOG\_DIGITAL\_CONVERSION, [53](#)
- Overwrite

- UTILITY, [143](#), [144](#)
- PAGE\_PROG\_4\_BYTE\_ADDR\_CMD
  - FLASH, [86](#)
- PAGE\_PROG\_CMD
  - FLASH, [86](#)
- PERIPHERAL, [36](#)
- PROG\_ERASE\_RESUME\_CMD
  - FLASH, [86](#)
- PROG\_ERASE\_SUSPEND\_CMD
  - FLASH, [86](#)
- PROG\_OTP\_ARRAY\_CMD
  - FLASH, [87](#)
- Parameter
  - CONTROLS, [21](#)
- Parse
  - EXTERNAL, [34](#)
- periph
  - dsy\_i2c\_handle, [188](#)
- pin
  - dsy\_gpio, [187](#)
  - dsy\_gpio\_pin, [187](#)
- pin\_
  - ANALOG\_DIGITAL\_CONVERSION, [58](#)
- pin\_config
  - dsy\_dac\_handle, [186](#)
  - dsy\_i2c\_handle, [188](#)
  - dsy\_qspi\_handle, [189](#)
  - dsy\_sdram\_handle, [194](#)
  - dsy\_sr\_4021\_handle, [195](#)
- Pins
  - FEEDBACK, [24](#)
  - ShiftRegister595, [205](#)
- Polarity
  - CONTROLS, [17](#)
- PollReceive
  - SERIAL, [48](#)
- PollTx
  - SERIAL, [48](#)
- PopEvent
  - EXTERNAL, [34](#)
- PopRx
  - SERIAL, [49](#)
- port
  - dsy\_gpio\_pin, [188](#)
- Prepare
  - daisy::WavPlayer, [218](#)
- PresetColor
  - UTILITY, [134](#)
- Pressed
  - CONTROLS, [21](#)
- Process
  - CONTROLS, [21](#), [22](#)
- Pull
  - CONTROLS, [17](#)
- pull
  - dsy\_gpio, [187](#)
- QUAD\_IN\_FAST\_PROG\_4\_BYTE\_ADDR\_CMD
  - FLASH, [87](#)
- QUAD\_IN\_FAST\_PROG\_CMD
  - FLASH, [87](#)
- QUAD\_INOUT\_FAST\_READ\_4\_BYTE\_ADDR\_CMD
  - FLASH, [87](#)
- QUAD\_INOUT\_FAST\_READ\_CMD
  - FLASH, [88](#)
- QUAD\_INOUT\_FAST\_READ\_DTR\_CMD
  - FLASH, [88](#)
- QUAD\_OUT\_FAST\_READ\_4\_BYTE\_ADDR\_CMD
  - FLASH, [88](#)
- QUAD\_OUT\_FAST\_READ\_CMD
  - FLASH, [88](#)
- QUAD\_OUT\_FAST\_READ\_DTR\_CMD
  - FLASH, [89](#)
- qspi\_handle
  - BOARDS, [128](#)
- r
  - codec\_frame\_t, [177](#)
- READ\_4\_BYTE\_ADDR\_CMD
  - FLASH, [89](#)
- READ\_CMD
  - FLASH, [89](#)
- READ\_ENHANCED\_VOL\_CFG\_REG\_CMD
  - FLASH, [89](#)
- READ\_EXT\_ADDR\_REG\_CMD
  - FLASH, [90](#)
- READ\_FLAG\_STATUS\_REG\_CMD
  - FLASH, [90](#)
- READ\_ID\_CMD2
  - FLASH, [90](#)
- READ\_ID\_CMD
  - FLASH, [90](#)
- READ\_LOCK\_REG\_CMD
  - FLASH, [91](#)
- READ\_NONVOL\_CFG\_REG\_CMD
  - FLASH, [91](#)
- READ\_OTP\_ARRAY\_CMD
  - FLASH, [91](#)
- READ\_READ\_PARAM\_REG\_CMD
  - FLASH, [91](#)
- READ\_SERIAL\_FLASH\_DISCO\_PARAM\_CMD
  - FLASH, [92](#)
- READ\_STATUS\_REG\_CMD
  - FLASH, [92](#)
- RESET\_ENABLE\_CMD
  - FLASH, [92](#)
- RESET\_MEMORY\_CMD
  - FLASH, [92](#)
- raw\_data
  - daisy::WavFileInfo, [215](#)
- Read
  - UTILITY, [144](#)
- Readable
  - SERIAL, [49](#)
- readable
  - UTILITY, [144](#), [145](#)
- ReceiveCallback

- UsbHandle, 209
- Red
  - UTILITY, 145
- red
  - color, 178
- RelCardAdd
  - DSY\_SD\_CardInfoTypeDef, 193
- Restart
  - daisy::WavPlayer, 218
- ring\_led
  - BOARDS, 128
- RingLed
  - BOARDS, 111
- RisingEdge
  - CONTROLS, 22
- RxActive
  - SERIAL, 49
- s162f
  - BOARDS, 119
- s242f
  - BOARDS, 120
- SD\_DATATIMEOUT
  - UTILITY, 132
- SD\_NOT\_PRESENT
  - UTILITY, 133
- SD\_PRESENT
  - UTILITY, 133
- SD\_TRANSFER\_BUSY
  - UTILITY, 133
- SD\_TRANSFER\_OK
  - UTILITY, 133
- SDRAM, 103
  - DSY\_SDRAM\_BSS, 103
  - DSY\_SDRAM\_DATA, 103
  - dsy\_sdram\_init, 104
  - dsy\_sdram\_pin, 104
  - dsy\_sdram\_state, 104
- SECTOR\_ERASE\_4\_BYTE\_ADDR\_CMD
  - FLASH, 93
- SECTOR\_ERASE\_CMD
  - FLASH, 93
- SERIAL, 37
  - BlockingTransmit, 44
  - CheckError, 44
  - dsy\_audio\_bitdepth, 39
  - dsy\_audio\_device, 39
  - dsy\_audio\_dir, 39
  - dsy\_audio\_sai, 39
  - dsy\_audio\_samplerate, 40
  - dsy\_audio\_sync, 40
  - dsy\_i2c\_init, 44
  - dsy\_i2c\_periph, 40
  - dsy\_i2c\_pin, 41
  - dsy\_i2c\_speed, 41
  - dsy\_qspi\_deinit, 44
  - dsy\_qspi\_device, 41
  - dsy\_qspi\_erase, 45
  - dsy\_qspi\_erasector, 45
  - dsy\_qspi\_init, 45
  - dsy\_qspi\_mode, 42
  - dsy\_qspi\_pin, 42
  - dsy\_qspi\_write, 46
  - dsy\_qspi\_writepage, 46
  - dsy\_sai\_init, 47
  - dsy\_sai\_init\_from\_handle, 47
  - dsy\_sai\_pin, 43
  - FlushRx, 47
  - Init, 48
  - kUartMaxBufferSize, 50
  - PollReceive, 48
  - PollTx, 48
  - PopRx, 49
  - Readable, 49
  - RxActive, 49
  - SpiPeriph, 43
  - SpiPin, 43
  - StartRx, 49
- SHIFTREGISTER, 67
  - dsy\_sr\_4021\_init, 68
  - dsy\_sr\_4021\_state, 68
  - dsy\_sr\_4021\_update, 68
- STM32\_USB\_OTG\_DEVICE\_LIBRARY, 170
- SUBSECTOR\_ERASE\_4\_BYTE\_ADDR\_CMD
  - FLASH, 93
- SUBSECTOR\_ERASE\_CMD
  - FLASH, 93
- SUBSECTOR\_ERASE\_QPI\_CMD
  - FLASH, 94
- SYSTEM, 64
  - dsy\_dma\_init, 64
  - dsy\_system\_delay, 64
  - dsy\_system\_getnow, 64
  - dsy\_system\_init, 65
  - dsy\_system\_jumpton, 65
  - dsy\_system\_jumptonqspi, 65
- sa\_audio\_callback
  - CODEC, 97
- sai
  - dsy\_audio\_handle, 185
- sai1\_pin\_config
  - dsy\_sai\_handle, 191
- sai2\_pin\_config
  - dsy\_sai\_handle, 191
- sai\_handle
  - BOARDS, 128
- SampleRate
  - WAV\_FormatTypeDef, 214
- samplerate
  - dsy\_sai\_handle, 191
- sdram\_handle
  - BOARDS, 128
- seed
  - BOARDS, 128, 129
- Set
  - FEEDBACK, 26, 27
  - ShiftRegister595, 206



- SetAudioBlockSize
  - BOARDS, [120](#), [121](#)
- SetColor
  - FEEDBACK, [27](#)
- SetCursor
  - FEEDBACK, [27](#)
- SetFootswitchLed
  - BOARDS, [121](#)
- SetLed
  - BOARDS, [121](#)
- SetLooping
  - daisy::WavPlayer, [218](#)
- SetReceiveCallback
  - UsbHandle, [211](#)
- SetRingLed
  - BOARDS, [121](#)
- SetTestPoint
  - BOARDS, [122](#)
- ShiftRegister595, [204](#)
  - Init, [205](#)
  - Pins, [205](#)
  - Set, [206](#)
  - Write, [206](#)
- speed
  - dsy\_i2c\_handle, [188](#)
- SpiPeriph
  - SERIAL, [43](#)
- SpiPin
  - SERIAL, [43](#)
- src/ffconf.h, [219](#)
- src/hid\_gatein.h, [226](#)
- src/hid\_wavplayer.h, [227](#)
- src/usbd\_cdc\_if.h, [227](#)
- src/usbd\_conf.h, [228](#)
- src/usbd\_desc.h, [229](#)
- Start
  - ANALOG\_DIGITAL\_CONVERSION, [58](#)
- StartAdc
  - BOARDS, [122](#)
- StartAudio
  - BOARDS, [122](#), [123](#)
- StartReceive
  - EXTERNAL, [34](#)
- StartRx
  - SERIAL, [49](#)
- state
  - dsy\_sdram\_handle, [194](#)
- states
  - dsy\_sr\_4021\_handle, [195](#)
- Stop
  - ANALOG\_DIGITAL\_CONVERSION, [58](#)
- Stream
  - daisy::WavPlayer, [218](#)
- SubChunk2Size
  - WAV\_FormatTypeDef, [215](#)
- SubChunk1ID
  - WAV\_FormatTypeDef, [214](#)
- SubChunk1Size
  - WAV\_FormatTypeDef, [214](#)
- SubChunk2ID
  - WAV\_FormatTypeDef, [214](#)
- Sw
  - BOARDS, [111](#), [112](#)
- Swallow
  - UTILITY, [145](#)
- switches
  - BOARDS, [129](#)
- sync\_config
  - dsy\_sai\_handle, [191](#)
- TimeHeldMs
  - CONTROLS, [22](#)
- TransmitExternal
  - UsbHandle, [211](#), [212](#)
- TransmitInternal
  - UsbHandle, [212](#)
- Trig
  - daisy::GateIn, [198](#)
- Type
  - CONTROLS, [18](#)
- type
  - EXTERNAL, [35](#)
- USB\_SIZ\_STRING\_SERIAL
  - USBD\_DESC\_Exported\_Constants, [163](#)
- USBD\_CDC\_IF\_Exported\_Defines, [149](#)
- USBD\_CDC\_IF\_Exported\_FunctionsPrototype, [153](#)
  - CDC\_Set\_Rx\_Callback\_FS, [153](#)
  - CDC\_Transmit\_FS, [153](#)
  - CDC\_Transmit\_HS, [153](#)
- USBD\_CDC\_IF\_Exported\_Macros, [151](#)
- USBD\_CDC\_IF\_Exported\_Types, [150](#)
  - CDC\_ReceiveCallback, [150](#)
- USBD\_CDC\_IF\_Exported\_Variables, [152](#)
  - USBD\_Interface\_fops\_FS, [152](#)
  - USBD\_Interface\_fops\_HS, [152](#)
- USBD\_CDC\_IF, [148](#)
- USBD\_CONF\_Exported\_Defines, [156](#)
  - DEVICE\_FS, [156](#)
  - DEVICE\_HS, [156](#)
  - USBD\_DEBUG\_LEVEL, [156](#)
  - USBD\_LPM\_ENABLED, [156](#)
  - USBD\_MAX\_NUM\_CONFIGURATION, [157](#)
  - USBD\_MAX\_NUM\_INTERFACES, [157](#)
  - USBD\_MAX\_STR\_DESC\_SIZ, [157](#)
  - USBD\_SELF\_POWERED, [157](#)
  - USBD\_SUPPORT\_USER\_STRING, [157](#)
- USBD\_CONF\_Exported\_FunctionsPrototype, [161](#)
- USBD\_CONF\_Exported\_Macros, [158](#)
  - USBD\_DbgLog, [158](#)
  - USBD\_Delay, [158](#)
  - USBD\_ErrLog, [158](#)
  - USBD\_UsrLog, [159](#)
  - USBD\_free, [159](#)
  - USBD\_malloc, [159](#)
  - USBD\_memcpy, [159](#)
  - USBD\_memset, [159](#)



- USBD\_CONF\_Exported\_Types, 160
- USBD\_CONF\_Exported\_Variables, 155
- USBD\_CONF, 154
- USBD\_DEBUG\_LEVEL
  - USBD\_CONF\_Exported\_Defines, 156
- USBD\_DESC\_Exported\_Constants, 163
  - DEVICE\_ID1, 163
  - DEVICE\_ID2, 163
  - DEVICE\_ID3, 163
  - USB\_SIZ\_STRING\_SERIAL, 163
- USBD\_DESC\_Exported\_Defines, 164
- USBD\_DESC\_Exported\_FunctionsPrototype, 168
- USBD\_DESC\_Exported\_Macros, 166
- USBD\_DESC\_Exported\_TypesDefinitions, 165
- USBD\_DESC\_Exported\_Variables, 167
  - FS\_Desc, 167
  - HS\_Desc, 167
- USBD\_DESC, 162
- USBD\_DbgLog
  - USBD\_CONF\_Exported\_Macros, 158
- USBD\_Delay
  - USBD\_CONF\_Exported\_Macros, 158
- USBD\_ErrLog
  - USBD\_CONF\_Exported\_Macros, 158
- USBD\_Interface\_fops\_FS
  - USBD\_CDC\_IF\_Exported\_Variables, 152
- USBD\_Interface\_fops\_HS
  - USBD\_CDC\_IF\_Exported\_Variables, 152
- USBD\_LPM\_ENABLED
  - USBD\_CONF\_Exported\_Defines, 156
- USBD\_MAX\_NUM\_CONFIGURATION
  - USBD\_CONF\_Exported\_Defines, 157
- USBD\_MAX\_NUM\_INTERFACES
  - USBD\_CONF\_Exported\_Defines, 157
- USBD\_MAX\_STR\_DESC\_SIZ
  - USBD\_CONF\_Exported\_Defines, 157
- USBD\_OTG\_DRIVER, 171
- USBD\_SELF\_POWERED
  - USBD\_CONF\_Exported\_Defines, 157
- USBD\_SUPPORT\_USER\_STRING
  - USBD\_CONF\_Exported\_Defines, 157
- USBD\_UsrLog
  - USBD\_CONF\_Exported\_Macros, 159
- USBD\_free
  - USBD\_CONF\_Exported\_Macros, 159
- USBD\_malloc
  - USBD\_CONF\_Exported\_Macros, 159
- USBD\_memcpy
  - USBD\_CONF\_Exported\_Macros, 159
- USBD\_memset
  - USBD\_CONF\_Exported\_Macros, 159
- UTILITY, 130
  - BSP\_SD\_AbortCallback, 134
  - BSP\_SD\_CardInfo, 132
  - BSP\_SD\_Erase, 134
  - BSP\_SD\_GetCardInfo, 135
  - BSP\_SD\_GetCardState, 135
  - BSP\_SD\_ITConfig, 136
  - BSP\_SD\_Init, 135
  - BSP\_SD\_IsDetected, 135
  - BSP\_SD\_ReadBlocks, 136
  - BSP\_SD\_ReadBlocks\_DMA, 136
  - BSP\_SD\_ReadCpltCallback, 137
  - BSP\_SD\_WriteBlocks, 137
  - BSP\_SD\_WriteBlocks\_DMA, 137
  - BSP\_SD\_WriteCpltCallback, 138
  - Blue, 134
  - capacity, 138
  - cube, 138
  - DMA\_BUFFER\_MEM\_SECTION, 132
  - DTCM\_MEM\_SECTION, 132
  - dsy\_get\_unique\_id, 139
  - dsy\_gpio\_port, 133
  - dsy\_hal\_map\_get\_i2c, 139
  - dsy\_hal\_map\_get\_pin, 140
  - dsy\_hal\_map\_get\_port, 140
  - dsy\_pin, 140
  - dsy\_pin\_cmp, 140
  - Flush, 140, 141
  - Font\_11x18, 146
  - Font\_16x26, 147
  - Font\_6x8, 147
  - Font\_7x10, 147
  - Green, 141
  - hi2c1, 147
  - hi2c2, 147
  - hi2c3, 147
  - hi2c4, 147
  - ImmediateRead, 141, 142
  - Init, 142, 143
  - MSD\_ERROR\_SD\_NOT\_PRESENT, 132
  - MSD\_ERROR, 132
  - MSD\_OK, 132
  - Overwrite, 143, 144
  - PresetColor, 134
  - Read, 144
  - readable, 144, 145
  - Red, 145
  - SD\_DATATIMEOUT, 132
  - SD\_NOT\_PRESENT, 133
  - SD\_PRESENT, 133
  - SD\_TRANSFER\_BUSY, 133
  - SD\_TRANSFER\_OK, 133
  - Swallow, 145
  - writable, 145, 146
  - Write, 146
- Update
  - FEEDBACK, 28
- UpdateAnalogControls
  - BOARDS, 123
- UpdateLeds
  - BOARDS, 124
- usb\_handle
  - BOARDS, 129
- UsbHandle, 208
  - Init, 210

- ReceiveCallback, [209](#)
- SetReceiveCallback, [211](#)
- TransmitExternal, [211](#), [212](#)
- TransmitInternal, [212](#)
- UsbPeriph, [209](#), [210](#)
- UsbPeriph
  - UsbHandle, [209](#), [210](#)
- Value
  - CONTROLS, [22](#), [23](#)
- value
  - EXTERNAL, [35](#)
- velocity
  - EXTERNAL, [35](#)
- WAV\_FILENAME\_MAX
  - hid\_wavplayer.h, [227](#)
- WAV\_FormatTypeDef, [213](#)
  - AudioFormat, [213](#)
  - BitPerSample, [213](#)
  - BlockAlign, [213](#)
  - ByteRate, [213](#)
  - ChunkId, [214](#)
  - FileFormat, [214](#)
  - FileSize, [214](#)
  - NbrChannels, [214](#)
  - SampleRate, [214](#)
  - SubChunk2Size, [215](#)
  - SubChunk1ID, [214](#)
  - SubChunk1Size, [214](#)
  - SubChunk2ID, [214](#)
- WRITE\_DISABLE\_CMD
  - FLASH, [94](#)
- WRITE\_ENABLE\_CMD
  - FLASH, [94](#)
- WRITE\_ENHANCED\_VOL\_CFG\_REG\_CMD
  - FLASH, [94](#)
- WRITE\_EXT\_ADDR\_REG\_CMD
  - FLASH, [95](#)
- WRITE\_LOCK\_REG\_CMD
  - FLASH, [95](#)
- WRITE\_NONVOL\_CFG\_REG\_CMD
  - FLASH, [95](#)
- WRITE\_READ\_PARAM\_REG\_CMD
  - FLASH, [95](#)
- WRITE\_STATUS\_REG\_CMD
  - FLASH, [96](#)
- writable
  - UTILITY, [145](#), [146](#)
- Write
  - ShiftRegister595, [206](#)
  - UTILITY, [146](#)
- WriteChar
  - FEEDBACK, [28](#)
- WriteString
  - FEEDBACK, [28](#)