

libDaisy

Generated by Doxygen 1.8.13

Contents

1	libdaisy	1
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
2	Module Index	3
2.1	Modules	3
3	Class Index	5
3.1	Class List	5
4	File Index	7
4.1	File List	7
5	Module Documentation	9
5.1	LIBDAISY	9
5.1.1	Detailed Description	9
5.2	HUMAN_INTERFACE	10
5.2.1	Detailed Description	10
5.3	AUDIO	11
5.3.1	Detailed Description	11
5.3.2	Typedef Documentation	11
5.3.2.1	dsy_audio_callback	11
5.3.2.2	dsy_audio_mc_callback	11

5.3.3	Enumeration Type Documentation	11
5.3.3.1	anonymous enum	11
5.3.4	Function Documentation	12
5.3.4.1	dsy_audio_enter_bypass()	12
5.3.4.2	dsy_audio_exit_bypass()	12
5.3.4.3	dsy_audio_init()	12
5.3.4.4	dsy_audio_passthru()	12
5.3.4.5	dsy_audio_set_blocksize()	13
5.3.4.6	dsy_audio_set_callback()	13
5.3.4.7	dsy_audio_set_mc_callback()	13
5.3.4.8	dsy_audio_silence()	13
5.3.4.9	dsy_audio_start()	13
5.3.4.10	dsy_audio_stop()	13
5.4	CONTROLS	14
5.4.1	Detailed Description	15
5.4.2	Enumeration Type Documentation	15
5.4.2.1	Curve	15
5.4.2.2	Polarity	15
5.4.2.3	Pull	15
5.4.2.4	Type	16
5.4.3	Function Documentation	16
5.4.3.1	AnalogControl()	16
5.4.3.2	Debounce() [1/2]	16
5.4.3.3	Debounce() [2/2]	16
5.4.3.4	FallingEdge() [1/2]	16
5.4.3.5	FallingEdge() [2/2]	17
5.4.3.6	Increment()	17
5.4.3.7	Init() [1/5]	17
5.4.3.8	Init() [2/5]	17
5.4.3.9	Init() [3/5]	17

5.4.3.10	Init() [4/5]	18
5.4.3.11	Init() [5/5]	18
5.4.3.12	InitBipolarCv()	19
5.4.3.13	Parameter()	19
5.4.3.14	Pressed() [1/2]	19
5.4.3.15	Pressed() [2/2]	19
5.4.3.16	Process() [1/2]	20
5.4.3.17	Process() [2/2]	20
5.4.3.18	RisingEdge() [1/2]	20
5.4.3.19	RisingEdge() [2/2]	20
5.4.3.20	TimeHeldMs() [1/2]	20
5.4.3.21	TimeHeldMs() [2/2]	20
5.4.3.22	Value() [1/2]	21
5.4.3.23	Value() [2/2]	21
5.4.3.24	~AnalogControl()	21
5.4.3.25	~Parameter()	21
5.5	FEEDBACK	22
5.5.1	Detailed Description	22
5.5.2	Enumeration Type Documentation	22
5.5.2.1	Pins	22
5.5.3	Function Documentation	23
5.5.3.1	DrawPixel()	23
5.5.3.2	Fill()	23
5.5.3.3	Init() [1/3]	23
5.5.3.4	Init() [2/3]	24
5.5.3.5	Init() [3/3]	24
5.5.3.6	Set() [1/2]	24
5.5.3.7	Set() [2/2]	25
5.5.3.8	SetColor()	25
5.5.3.9	SetCursor()	25

5.5.3.10	Update() [1/3]	26
5.5.3.11	Update() [2/3]	26
5.5.3.12	Update() [3/3]	26
5.5.3.13	WriteChar()	26
5.5.3.14	WriteString()	27
5.6	EXTERNAL	28
5.6.1	Detailed Description	29
5.6.2	Enumeration Type Documentation	29
5.6.2.1	MidiInputMode	29
5.6.2.2	MidiMessageType	29
5.6.2.3	MidiOutputMode	29
5.6.3	Function Documentation	31
5.6.3.1	AsControlChange()	31
5.6.3.2	AsNoteOn()	31
5.6.3.3	HasEvents()	31
5.6.3.4	Init()	31
5.6.3.5	Listen()	32
5.6.3.6	Parse()	32
5.6.3.7	PopEvent()	32
5.6.3.8	StartReceive()	32
5.6.4	Variable Documentation	32
5.6.4.1	channel [1/3]	33
5.6.4.2	channel [2/3]	33
5.6.4.3	channel [3/3]	33
5.6.4.4	control_number	33
5.6.4.5	data	33
5.6.4.6	note	33
5.6.4.7	type	33
5.6.4.8	value	33
5.6.4.9	velocity	33

5.7	PERIPHERAL	34
5.8	SYSTEM	35
5.9	DEV	36
5.10	DAISY	37
5.11	UTILITY	38
5.12	USBD_CDC_IF	39
5.12.1	Detailed Description	39
5.13	USBD_CDC_IF_Exported_Defines	40
5.14	USBD_CDC_IF_Exported_Types	41
5.14.1	Detailed Description	41
5.14.2	Typedef Documentation	41
5.14.2.1	CDC_ReceiveCallback	41
5.15	USBD_CDC_IF_Exported_Macros	42
5.16	USBD_CDC_IF_Exported_Variables	43
5.16.1	Detailed Description	43
5.16.2	Variable Documentation	43
5.16.2.1	USBD_Interface_fops_FS	43
5.16.2.2	USBD_Interface_fops_HS	43
5.17	USBD_CDC_IF_Exported_FunctionsPrototype	44
5.17.1	Detailed Description	44
5.17.2	Function Documentation	44
5.17.2.1	CDC_Set_Rx_Callback_FS()	44
5.17.2.2	CDC_Transmit_FS()	44
5.17.2.3	CDC_Transmit_HS()	44
5.18	USBD_CONF	45
5.18.1	Detailed Description	45
5.19	USBD_CONF_Exported_Variables	46
5.20	USBD_CONF_Exported_Defines	47
5.20.1	Detailed Description	47
5.20.2	Macro Definition Documentation	47

5.20.2.1	DEVICE_FS	47
5.20.2.2	DEVICE_HS	47
5.20.2.3	USBD_DEBUG_LEVEL	47
5.20.2.4	USBD_LPM_ENABLED	48
5.20.2.5	USBD_MAX_NUM_CONFIGURATION	48
5.20.2.6	USBD_MAX_NUM_INTERFACES	48
5.20.2.7	USBD_MAX_STR_DESC_SIZ	48
5.20.2.8	USBD_SELF_POWERED	48
5.20.2.9	USBD_SUPPORT_USER_STRING	48
5.21	USBD_CONF_Exported_Macros	49
5.21.1	Detailed Description	49
5.21.2	Macro Definition Documentation	49
5.21.2.1	USBD_DbgLog	49
5.21.2.2	USBD_Delay	49
5.21.2.3	USBD_ErrLog	50
5.21.2.4	USBD_free	50
5.21.2.5	USBD_malloc	50
5.21.2.6	USBD_memcpy	50
5.21.2.7	USBD_memset	50
5.21.2.8	USBD_UsrLog	50
5.22	USBD_CONF_Exported_Types	51
5.23	USBD_CONF_Exported_FunctionsPrototype	52
5.24	USBD_DESC	53
5.24.1	Detailed Description	53
5.25	USBD_DESC_Exported_Constants	54
5.25.1	Detailed Description	54
5.25.2	Macro Definition Documentation	54
5.25.2.1	DEVICE_ID1	54
5.25.2.2	DEVICE_ID2	54
5.25.2.3	DEVICE_ID3	54

5.25.2.4 USB_SIZ_STRING_SERIAL	54
5.26 USBD_DESC_Exported_Defines	55
5.27 USBD_DESC_Exported_TypesDefinitions	56
5.28 USBD_DESC_Exported_Macros	57
5.29 USBD_DESC_Exported_Variables	58
5.29.1 Detailed Description	58
5.29.2 Variable Documentation	58
5.29.2.1 FS_Desc	58
5.29.2.2 HS_Desc	58
5.30 USBD_DESC_Exported_FunctionsPrototype	59
5.31 Externals	60
5.32 STM32_USB_OTG_DEVICE_LIBRARY	61
5.32.1 Detailed Description	61
5.33 USBD_OTG_DRIVER	62
5.33.1 Detailed Description	62
6 Class Documentation	63
6.1 daisy::AdcChannelConfig Struct Reference	63
6.1.1 Detailed Description	63
6.1.2 Member Enumeration Documentation	63
6.1.2.1 MuxPin	63
6.1.3 Member Function Documentation	64
6.1.3.1 InitMux()	64
6.1.3.2 InitSingle()	64
6.1.4 Member Data Documentation	65
6.1.4.1 mux_channels_	65
6.1.4.2 mux_pin_	65
6.1.4.3 pin_	65
6.2 daisy::AdcHandle Class Reference	65
6.2.1 Detailed Description	66
6.2.2 Member Enumeration Documentation	66

6.2.2.1	OverSampling	66
6.2.3	Member Function Documentation	66
6.2.3.1	Get()	66
6.2.3.2	GetFloat()	67
6.2.3.3	GetMux()	67
6.2.3.4	GetMuxFloat()	67
6.2.3.5	GetMuxPtr()	68
6.2.3.6	GetPtr()	68
6.2.3.7	Init()	69
6.2.3.8	Start()	69
6.2.3.9	Stop()	69
6.3	daisy::AnalogControl Class Reference	69
6.3.1	Detailed Description	70
6.4	codec_frame_t Struct Reference	70
6.4.1	Detailed Description	70
6.4.2	Member Data Documentation	70
6.4.2.1	l	70
6.4.2.2	r	71
6.5	color Struct Reference	71
6.5.1	Detailed Description	71
6.5.2	Member Data Documentation	71
6.5.2.1	blue	71
6.5.2.2	green	71
6.5.2.3	red	72
6.6	daisy::Color Class Reference	72
6.6.1	Detailed Description	72
6.6.2	Member Enumeration Documentation	72
6.6.2.1	PresetColor	72
6.6.3	Member Function Documentation	73
6.6.3.1	Blue()	73

6.6.3.2	Green()	73
6.6.3.3	Init() [1/2]	73
6.6.3.4	Init() [2/2]	73
6.6.3.5	Red()	74
6.7	daisy::ControlChangeEvent Struct Reference	74
6.7.1	Detailed Description	74
6.8	daisy::daisy_field Struct Reference	74
6.8.1	Detailed Description	75
6.8.2	Member Data Documentation	75
6.8.2.1	cvs	75
6.8.2.2	gate_in	75
6.8.2.3	gate_out	75
6.8.2.4	keyboard_sr	75
6.8.2.5	knobs	75
6.8.2.6	seed	76
6.8.2.7	switches	76
6.9	daisy::DaisyPatch Class Reference	76
6.9.1	Detailed Description	77
6.9.2	Member Enumeration Documentation	77
6.9.2.1	Ctrl	77
6.9.2.2	GateInput	77
6.9.3	Constructor & Destructor Documentation	77
6.9.3.1	DaisyPatch()	78
6.9.3.2	~DaisyPatch()	78
6.9.4	Member Function Documentation	78
6.9.4.1	AudioBlockSize()	78
6.9.4.2	AudioCallbackRate()	78
6.9.4.3	AudioSampleRate()	78
6.9.4.4	ChangeAudioCallback()	78
6.9.4.5	DebounceControls()	79

6.9.4.6	DelayMs()	79
6.9.4.7	DisplayControls()	79
6.9.4.8	GetCtrlValue()	79
6.9.4.9	Init()	79
6.9.4.10	SetAudioBlockSize()	80
6.9.4.11	StartAdc()	80
6.9.4.12	StartAudio()	80
6.9.4.13	UpdateAnalogControls()	80
6.9.5	Member Data Documentation	80
6.9.5.1	controls	80
6.9.5.2	display	81
6.9.5.3	encoder	81
6.9.5.4	gate_input	81
6.9.5.5	gate_output	81
6.9.5.6	midi	81
6.9.5.7	seed	81
6.10	daisy::DaisyPetal Class Reference	81
6.10.1	Detailed Description	82
6.10.2	Member Enumeration Documentation	83
6.10.2.1	FootswitchLed	83
6.10.2.2	Knob	83
6.10.2.3	RingLed	83
6.10.2.4	Sw	84
6.10.3	Constructor & Destructor Documentation	84
6.10.3.1	DaisyPetal()	84
6.10.3.2	~DaisyPetal()	84
6.10.4	Member Function Documentation	85
6.10.4.1	AudioBlockSize()	85
6.10.4.2	AudioCallbackRate()	85
6.10.4.3	AudioSampleRate()	85

6.10.4.4	ChangeAudioCallback()	85
6.10.4.5	ClearLeds()	85
6.10.4.6	DebounceControls()	85
6.10.4.7	DelayMs()	86
6.10.4.8	GetExpression()	86
6.10.4.9	GetKnobValue()	86
6.10.4.10	Init()	86
6.10.4.11	SetAudioBlockSize()	86
6.10.4.12	SetFootswitchLed()	87
6.10.4.13	SetRingLed()	87
6.10.4.14	StartAdc()	87
6.10.4.15	StartAudio()	88
6.10.4.16	UpdateAnalogControls()	88
6.10.4.17	UpdateLeds()	88
6.10.5	Member Data Documentation	88
6.10.5.1	encoder	88
6.10.5.2	expression	88
6.10.5.3	footswitch_led	88
6.10.5.4	knob	89
6.10.5.5	ring_led	89
6.10.5.6	seed	89
6.10.5.7	switches	89
6.11	daisy::DaisyPod Class Reference	89
6.11.1	Detailed Description	90
6.11.2	Member Enumeration Documentation	90
6.11.2.1	Knob	90
6.11.2.2	Sw	91
6.11.3	Member Function Documentation	91
6.11.3.1	AudioBlockSize()	91
6.11.3.2	AudioCallbackRate()	91

6.11.3.3	AudioSampleRate()	91
6.11.3.4	ChangeAudioCallback()	91
6.11.3.5	ClearLeds()	92
6.11.3.6	DebounceControls()	92
6.11.3.7	DelayMs()	92
6.11.3.8	GetKnobValue()	92
6.11.3.9	Init()	92
6.11.3.10	SetAudioBlockSize()	92
6.11.3.11	StartAdc()	93
6.11.3.12	StartAudio()	93
6.11.3.13	UpdateAnalogControls()	93
6.11.3.14	UpdateLeds()	93
6.11.4	Member Data Documentation	93
6.11.4.1	button1	93
6.11.4.2	button2	94
6.11.4.3	buttons	94
6.11.4.4	encoder	94
6.11.4.5	knob1	94
6.11.4.6	knob2	94
6.11.4.7	knobs	94
6.11.4.8	led1	94
6.11.4.9	led2	94
6.11.4.10	seed	95
6.12	daisy::DaisySeed Class Reference	95
6.12.1	Detailed Description	95
6.12.2	Member Function Documentation	95
6.12.2.1	AudioSampleRate()	96
6.12.2.2	Configure()	96
6.12.2.3	GetPin()	96
6.12.2.4	Init()	96

6.12.2.5	SetAudioBlockSize()	96
6.12.2.6	SetLed()	96
6.12.2.7	SetTestPoint()	97
6.12.2.8	StartAudio()	97
6.12.3	Member Data Documentation	97
6.12.3.1	adc	97
6.12.3.2	audio_handle	97
6.12.3.3	dac_handle	97
6.12.3.4	i2c1_handle	97
6.12.3.5	i2c2_handle	98
6.12.3.6	qspi_handle	98
6.12.3.7	sai_handle	98
6.12.3.8	sdram_handle	98
6.12.3.9	usb_handle	98
6.13	dsy_audio_handle Struct Reference	98
6.13.1	Detailed Description	99
6.13.2	Member Data Documentation	99
6.13.2.1	block_size	99
6.13.2.2	dev0_i2c	99
6.13.2.3	dev1_i2c	99
6.13.2.4	sai	99
6.14	dsy_dac_handle Struct Reference	99
6.14.1	Detailed Description	100
6.14.2	Member Data Documentation	100
6.14.2.1	bitdepth	100
6.14.2.2	mode	100
6.14.2.3	pin_config	100
6.15	dsy_gpio Struct Reference	100
6.15.1	Detailed Description	100
6.15.2	Member Data Documentation	101

6.15.2.1	mode	101
6.15.2.2	pin	101
6.15.2.3	pull	101
6.16	dsy_gpio_pin Struct Reference	101
6.16.1	Detailed Description	101
6.16.2	Member Data Documentation	101
6.16.2.1	pin	102
6.16.2.2	port	102
6.17	dsy_i2c_handle Struct Reference	102
6.17.1	Detailed Description	102
6.17.2	Member Data Documentation	102
6.17.2.1	periph	102
6.17.2.2	pin_config	102
6.17.2.3	speed	103
6.18	dsy_qspi_handle Struct Reference	103
6.18.1	Detailed Description	103
6.18.2	Member Data Documentation	103
6.18.2.1	device	103
6.18.2.2	mode	103
6.18.2.3	pin_config	104
6.19	dsy_sai_handle Struct Reference	104
6.19.1	Detailed Description	104
6.19.2	Member Data Documentation	104
6.19.2.1	a_direction	104
6.19.2.2	b_direction	104
6.19.2.3	bitdepth	105
6.19.2.4	device	105
6.19.2.5	init	105
6.19.2.6	sai1_pin_config	105
6.19.2.7	sai2_pin_config	105

6.19.2.8	samplerate	105
6.19.2.9	sync_config	105
6.20	DSY_SD_CardInfoTypeDef Struct Reference	106
6.20.1	Detailed Description	106
6.20.2	Member Data Documentation	106
6.20.2.1	BlockNbr	106
6.20.2.2	BlockSize	106
6.20.2.3	CardSpeed	106
6.20.2.4	CardType	107
6.20.2.5	CardVersion	107
6.20.2.6	Class	107
6.20.2.7	LogBlockNbr	107
6.20.2.8	LogBlockSize	107
6.20.2.9	RelCardAdd	107
6.21	dsy_sdram_handle Struct Reference	107
6.21.1	Detailed Description	108
6.21.2	Member Data Documentation	108
6.21.2.1	pin_config	108
6.21.2.2	state	108
6.22	dsy_sr_4021_handle Struct Reference	108
6.22.1	Detailed Description	109
6.22.2	Member Data Documentation	109
6.22.2.1	clk	109
6.22.2.2	cs	109
6.22.2.3	data	109
6.22.2.4	num_daisy chained	109
6.22.2.5	num_parallel	109
6.22.2.6	pin_config	109
6.22.2.7	states	110
6.23	daisy::Encoder Class Reference	110

6.23.1 Detailed Description	110
6.24 FontDef Struct Reference	110
6.24.1 Detailed Description	111
6.24.2 Member Data Documentation	111
6.24.2.1 data	111
6.24.2.2 FontHeight	111
6.24.2.3 FontWidth	111
6.25 daisy::GateIn Class Reference	111
6.25.1 Detailed Description	112
6.25.2 Constructor & Destructor Documentation	112
6.25.2.1 GateIn()	112
6.25.2.2 ~GateIn()	112
6.25.3 Member Function Documentation	112
6.25.3.1 Init()	112
6.25.3.2 Trig()	113
6.26 daisy::Led Class Reference	113
6.26.1 Detailed Description	113
6.27 daisy::MidiEvent Struct Reference	113
6.27.1 Detailed Description	114
6.28 daisy::MidiHandler Class Reference	114
6.28.1 Detailed Description	115
6.29 daisy::NoteOnEvent Struct Reference	115
6.29.1 Detailed Description	115
6.30 daisy::OledDisplay Class Reference	115
6.30.1 Detailed Description	116
6.31 daisy::Parameter Class Reference	116
6.31.1 Detailed Description	116
6.32 daisy::RgbLed Class Reference	117
6.32.1 Detailed Description	117
6.33 daisy::RingBuffer< T, size > Class Template Reference	117

6.33.1 Detailed Description	117
6.33.2 Member Function Documentation	118
6.33.2.1 capacity()	118
6.33.2.2 Flush()	118
6.33.2.3 ImmediateRead() [1/2]	118
6.33.2.4 ImmediateRead() [2/2]	118
6.33.2.5 Init()	119
6.33.2.6 Overwrite() [1/2]	119
6.33.2.7 Overwrite() [2/2]	119
6.33.2.8 Read()	119
6.33.2.9 readable()	120
6.33.2.10 Swallow()	120
6.33.2.11 writable()	120
6.33.2.12 Write()	120
6.34 daisy::RingBuffer< T, 0 > Class Template Reference	121
6.34.1 Detailed Description	121
6.34.2 Member Function Documentation	121
6.34.2.1 capacity()	121
6.34.2.2 Flush()	122
6.34.2.3 ImmediateRead() [1/2]	122
6.34.2.4 ImmediateRead() [2/2]	122
6.34.2.5 Init()	122
6.34.2.6 Overwrite() [1/2]	122
6.34.2.7 Overwrite() [2/2]	123
6.34.2.8 Read()	123
6.34.2.9 readable()	123
6.34.2.10 writable()	123
6.34.2.11 Write()	123
6.35 daisy::SdmmcHandler Class Reference	124
6.35.1 Detailed Description	124

6.35.2	Member Function Documentation	124
6.35.2.1	Init()	124
6.36	daisy::SdmmcHandlerInit Struct Reference	124
6.36.1	Detailed Description	125
6.36.2	Member Data Documentation	125
6.36.2.1	bitdepth	125
6.36.2.2	speed	125
6.37	ShiftRegister595 Class Reference	125
6.37.1	Detailed Description	126
6.37.2	Member Enumeration Documentation	126
6.37.2.1	Pins	126
6.37.3	Member Function Documentation	126
6.37.3.1	Init()	126
6.37.3.2	Set()	127
6.37.3.3	Write()	127
6.38	daisy::SpiHandle Class Reference	127
6.38.1	Detailed Description	127
6.38.2	Member Function Documentation	127
6.38.2.1	BlockingTransmit()	127
6.38.2.2	Init()	128
6.39	daisy::Switch Class Reference	128
6.39.1	Detailed Description	128
6.40	daisy::UartHandler Class Reference	129
6.40.1	Detailed Description	129
6.40.2	Member Function Documentation	129
6.40.2.1	CheckError()	129
6.40.2.2	FlushRx()	130
6.40.2.3	Init()	130
6.40.2.4	PollReceive()	130
6.40.2.5	PollTx()	130

6.40.2.6	PopRx()	131
6.40.2.7	Readable()	131
6.40.2.8	RxActive()	131
6.40.2.9	StartRx()	131
6.41	UsbHandle Class Reference	132
6.41.1	Detailed Description	132
6.41.2	Member Typedef Documentation	133
6.41.2.1	ReceiveCallback [1/2]	133
6.41.2.2	ReceiveCallback [2/2]	133
6.41.3	Member Enumeration Documentation	133
6.41.3.1	UsbPeriph [1/2]	133
6.41.3.2	UsbPeriph [2/2]	133
6.41.4	Member Function Documentation	134
6.41.4.1	Init() [1/2]	134
6.41.4.2	Init() [2/2]	134
6.41.4.3	SetReceiveCallback() [1/2]	134
6.41.4.4	SetReceiveCallback() [2/2]	135
6.41.4.5	TransmitExternal() [1/2]	135
6.41.4.6	TransmitExternal() [2/2]	135
6.41.4.7	TransmitInternal() [1/2]	136
6.41.4.8	TransmitInternal() [2/2]	136
6.42	WAV_FormatTypeDef Struct Reference	136
6.42.1	Detailed Description	137
6.42.2	Member Data Documentation	137
6.42.2.1	AudioFormat	137
6.42.2.2	BitPerSample	137
6.42.2.3	BlockAlign	137
6.42.2.4	ByteRate	137
6.42.2.5	ChunkId	137
6.42.2.6	FileFormat	138

6.42.2.7	FileSize	138
6.42.2.8	NbrChannels	138
6.42.2.9	SampleRate	138
6.42.2.10	SubChunk1ID	138
6.42.2.11	SubChunk1Size	138
6.42.2.12	SubChunk2ID	138
6.42.2.13	SubCHunk2Size	139
6.43	daisy::WavFileInfo Struct Reference	139
6.43.1	Detailed Description	139
6.43.2	Member Data Documentation	139
6.43.2.1	name	139
6.43.2.2	raw_data	139
6.44	daisy::WavPlayer Class Reference	140
6.44.1	Detailed Description	140
6.44.2	Member Function Documentation	140
6.44.2.1	Close()	140
6.44.2.2	GetCurrentFile()	140
6.44.2.3	GetLooping()	141
6.44.2.4	GetNumberFiles()	141
6.44.2.5	Init()	141
6.44.2.6	Open()	141
6.44.2.7	Prepare()	141
6.44.2.8	Restart()	142
6.44.2.9	SetLooping()	142
6.44.2.10	Stream()	142

7 File Documentation	143
7.1 src/daisy.h File Reference	143
7.1.1 Macro Definition Documentation	144
7.1.1.1 F2S16_SCALE	144
7.1.1.2 F2S24_SCALE	144
7.1.1.3 FBIPMAX	144
7.1.1.4 FBIPMIN	144
7.1.1.5 S162F_SCALE	145
7.1.1.6 S242F_SCALE	145
7.1.1.7 S24SIGN	145
7.1.2 Function Documentation	145
7.1.2.1 f2s16()	145
7.1.2.2 f2s24()	145
7.1.2.3 s162f()	145
7.1.2.4 s242f()	146
7.2 src/daisy_core.h File Reference	146
7.2.1 Macro Definition Documentation	147
7.2.1.1 DMA_BUFFER_MEM_SECTION	147
7.2.1.2 DSY_CORE_HW_H	147
7.2.1.3 DTCM_MEM_SECTION	147
7.2.2 Enumeration Type Documentation	147
7.2.2.1 dsy_gpio_port	147
7.2.3 Function Documentation	148
7.2.3.1 cube()	148
7.2.3.2 dsy_pin()	148
7.2.3.3 dsy_pin_cmp()	148
7.3 src/daisy_field.h File Reference	148
7.3.1 Detailed Description	150
7.3.2 Macro Definition Documentation	150
7.3.2.1 CV1_ADC_PIN	150

7.3.2.2	CV2_ADC_PIN	150
7.3.2.3	CV3_ADC_PIN	150
7.3.2.4	CV4_ADC_PIN	150
7.3.2.5	DSY_FIELD_BSP_H	150
7.3.2.6	GATE_IN_PIN	150
7.3.2.7	GATE_OUT_PIN	151
7.3.2.8	KB_SW_SR_CLK_PIN	151
7.3.2.9	KB_SW_SR_CS_PIN	151
7.3.2.10	KB_SW_SR_D1_PIN	151
7.3.2.11	KB_SW_SR_D2_PIN	151
7.3.2.12	LED_DRIVER_I2C	151
7.3.2.13	MIDI_IN_PIN	151
7.3.2.14	MIDI_OUT_PIN	151
7.3.2.15	MUX_ADC_PIN	152
7.3.2.16	MUX_SEL_0_PIN	152
7.3.2.17	MUX_SEL_1_PIN	152
7.3.2.18	MUX_SEL_2_PIN	152
7.3.2.19	SAMPLE_RATE	152
7.3.2.20	SW_1_PIN	152
7.3.2.21	SW_2_PIN	152
7.3.2.22	SW_3_PIN	152
7.3.3	Enumeration Type Documentation	152
7.3.3.1	anonymous enum	152
7.3.3.2	anonymous enum	153
7.3.3.3	anonymous enum	153
7.3.3.4	anonymous enum	153
7.3.4	Function Documentation	154
7.3.4.1	daisy_field_init()	154
7.4	src/daisy_patch.h File Reference	155
7.5	src/daisy_petal.h File Reference	156

7.5.1	Macro Definition Documentation	156
7.5.1.1	DSY_PETAL_H	156
7.6	src/daisy_pod.h File Reference	156
7.7	src/daisy_seed.h File Reference	156
7.8	src/dev_codec_ak4556.h File Reference	157
7.8.1	Detailed Description	157
7.8.2	Function Documentation	157
7.8.2.1	codec_ak4556_init()	157
7.9	src/dev_codec_pcm3060.h File Reference	157
7.9.1	Detailed Description	158
7.9.2	Function Documentation	158
7.9.2.1	codec_pcm3060_init()	158
7.10	src/dev_codec_wm8731.h File Reference	158
7.10.1	Detailed Description	158
7.10.2	Function Documentation	159
7.10.2.1	codec_wm8731_enter_bypass()	159
7.10.2.2	codec_wm8731_exit_bypass()	159
7.10.2.3	codec_wm8731_init()	159
7.11	src/dev_codec_wm8731_frame.h File Reference	160
7.11.1	Detailed Description	160
7.11.2	Typedef Documentation	160
7.11.2.1	sa_audio_callback	160
7.12	src/dev_flash_IS25LP064A.h File Reference	160
7.12.1	Detailed Description	163
7.12.2	Macro Definition Documentation	163
7.12.2.1	BLOCK_ERASE_32K_CMD	163
7.12.2.2	CLEAR_FLAG_STATUS_REG_CMD	163
7.12.2.3	DIE_ERASE_CMD	163
7.12.2.4	DUAL_IN_FAST_PROG_CMD	163
7.12.2.5	DUAL_INOUT_FAST_READ_4_BYTE_ADDR_CMD	163

7.12.2.6 DUAL_INOUT_FAST_READ_CMD	163
7.12.2.7 DUAL_INOUT_FAST_READ_DTR_CMD	164
7.12.2.8 DUAL_OUT_FAST_READ_4_BYTE_ADDR_CMD	164
7.12.2.9 DUAL_OUT_FAST_READ_CMD	164
7.12.2.10 DUAL_OUT_FAST_READ_DTR_CMD	164
7.12.2.11 ENTER_4_BYTE_ADDR_MODE_CMD	164
7.12.2.12 ENTER_QUAD_CMD	164
7.12.2.13 EXIT_4_BYTE_ADDR_MODE_CMD	164
7.12.2.14 EXIT_QUAD_CMD	164
7.12.2.15 EXT_DUAL_IN_FAST_PROG_CMD	165
7.12.2.16 EXT_QUAD_IN_FAST_PROG_CMD	165
7.12.2.17 FAST_READ_4_BYTE_ADDR_CMD	165
7.12.2.18 FAST_READ_CMD	165
7.12.2.19 FAST_READ_DTR_CMD	165
7.12.2.20 IS25LP064A_DIE_ERASE_MAX_TIME	165
7.12.2.21 IS25LP064A_DUMMY_CYCLES_READ	165
7.12.2.22 IS25LP064A_DUMMY_CYCLES_READ_DTR	165
7.12.2.23 IS25LP064A_DUMMY_CYCLES_READ_QUAD	166
7.12.2.24 IS25LP064A_DUMMY_CYCLES_READ_QUAD_DTR	166
7.12.2.25 IS25LP064A_EAR_HIGHEST_SE	166
7.12.2.26 IS25LP064A_EAR_LOWEST_SEG	166
7.12.2.27 IS25LP064A_EAR_SECOND_SEG	166
7.12.2.28 IS25LP064A_EAR_THIRD_SEG	166
7.12.2.29 IS25LP064A_EVCR_DTRP	166
7.12.2.30 IS25LP064A_EVCR_DUAL	166
7.12.2.31 IS25LP064A_EVCR_ODS	167
7.12.2.32 IS25LP064A_EVCR_QUAD	167
7.12.2.33 IS25LP064A_EVCR_RH	167
7.12.2.34 IS25LP064A_FLASH_SIZE	167
7.12.2.35 IS25LP064A_FSR_ERERR	167

7.12.2.36 IS25LP064A_FSR_ERSUS	167
7.12.2.37 IS25LP064A_FSR_NBADDR	167
7.12.2.38 IS25LP064A_FSR_PGERR	167
7.12.2.39 IS25LP064A_FSR_PGSUS	168
7.12.2.40 IS25LP064A_FSR_PRERR	168
7.12.2.41 IS25LP064A_FSR_READY	168
7.12.2.42 IS25LP064A_H	168
7.12.2.43 IS25LP064A_NVCR_DTRP	168
7.12.2.44 IS25LP064A_NVCR_DUAL	168
7.12.2.45 IS25LP064A_NVCR_NB_DUMMY	168
7.12.2.46 IS25LP064A_NVCR_NBADDR	168
7.12.2.47 IS25LP064A_NVCR_ODS	169
7.12.2.48 IS25LP064A_NVCR_QUAB	169
7.12.2.49 IS25LP064A_NVCR_RH	169
7.12.2.50 IS25LP064A_NVCR_SEGMENT	169
7.12.2.51 IS25LP064A_NVCR_XIP	169
7.12.2.52 IS25LP064A_PAGE_SIZE	169
7.12.2.53 IS25LP064A_SECTOR_ERASE_MAX_TIME	169
7.12.2.54 IS25LP064A_SECTOR_SIZE	169
7.12.2.55 IS25LP064A_SR_QE	170
7.12.2.56 IS25LP064A_SR_SRWREN	170
7.12.2.57 IS25LP064A_SR_WIP	170
7.12.2.58 IS25LP064A_SR_WREN	170
7.12.2.59 IS25LP064A_SUBSECTOR_ERASE_MAX_TIME	170
7.12.2.60 IS25LP064A_SUBSECTOR_SIZE	170
7.12.2.61 IS25LP064A_VCR_NB_DUMMY	170
7.12.2.62 IS25LP064A_VCR_WRAP	171
7.12.2.63 IS25LP064A_VCR_XIP	171
7.12.2.64 MULTIPLE_IO_READ_ID_CMD	171
7.12.2.65 PAGE_PROG_4_BYTE_ADDR_CMD	171

7.12.2.66 PAGE_PROG_CMD	171
7.12.2.67 PROG_ERASE_RESUME_CMD	171
7.12.2.68 PROG_ERASE_SUSPEND_CMD	171
7.12.2.69 PROG_OTP_ARRAY_CMD	171
7.12.2.70 QUAD_IN_FAST_PROG_4_BYTE_ADDR_CMD	172
7.12.2.71 QUAD_IN_FAST_PROG_CMD	172
7.12.2.72 QUAD_INOUT_FAST_READ_4_BYTE_ADDR_CMD	172
7.12.2.73 QUAD_INOUT_FAST_READ_CMD	172
7.12.2.74 QUAD_INOUT_FAST_READ_DTR_CMD	172
7.12.2.75 QUAD_OUT_FAST_READ_4_BYTE_ADDR_CMD	172
7.12.2.76 QUAD_OUT_FAST_READ_CMD	172
7.12.2.77 QUAD_OUT_FAST_READ_DTR_CMD	172
7.12.2.78 READ_4_BYTE_ADDR_CMD	173
7.12.2.79 READ_CMD	173
7.12.2.80 READ_ENHANCED_VOL_CFG_REG_CMD	173
7.12.2.81 READ_EXT_ADDR_REG_CMD	173
7.12.2.82 READ_FLAG_STATUS_REG_CMD	173
7.12.2.83 READ_ID_CMD	173
7.12.2.84 READ_ID_CMD2	173
7.12.2.85 READ_LOCK_REG_CMD	173
7.12.2.86 READ_NONVOL_CFG_REG_CMD	174
7.12.2.87 READ_OTP_ARRAY_CMD	174
7.12.2.88 READ_READ_PARAM_REG_CMD	174
7.12.2.89 READ_SERIAL_FLASH_DISCO_PARAM_CMD	174
7.12.2.90 READ_STATUS_REG_CMD	174
7.12.2.91 RESET_ENABLE_CMD	174
7.12.2.92 RESET_MEMORY_CMD	174
7.12.2.93 SECTOR_ERASE_4_BYTE_ADDR_CMD	174
7.12.2.94 SECTOR_ERASE_CMD	175
7.12.2.95 SUBSECTOR_ERASE_4_BYTE_ADDR_CMD	175

7.12.2.96 SUBSECTOR_ERASE_CMD	175
7.12.2.97 SUBSECTOR_ERASE_QPI_CMD	175
7.12.2.98 WRITE_DISABLE_CMD	175
7.12.2.99 WRITE_ENABLE_CMD	175
7.12.2.100WRITE_ENHANCED_VOL_CFG_REG_CMD	175
7.12.2.101WRITE_EXT_ADDR_REG_CMD	175
7.12.2.102WRITE_LOCK_REG_CMD	176
7.12.2.103WRITE_NONVOL_CFG_REG_CMD	176
7.12.2.104WRITE_READ_PARAM_REG_CMD	176
7.12.2.105WRITE_STATUS_REG_CMD	176
7.13 src/dev_flash_IS25LP080D.h File Reference	176
7.13.1 Detailed Description	178
7.13.2 Macro Definition Documentation	178
7.13.2.1 BLOCK_ERASE_32K_CMD	178
7.13.2.2 CLEAR_FLAG_STATUS_REG_CMD	179
7.13.2.3 DIE_ERASE_CMD	179
7.13.2.4 DUAL_IN_FAST_PROG_CMD	179
7.13.2.5 DUAL_INOUT_FAST_READ_4_BYTE_ADDR_CMD	179
7.13.2.6 DUAL_INOUT_FAST_READ_CMD	179
7.13.2.7 DUAL_INOUT_FAST_READ_DTR_CMD	179
7.13.2.8 DUAL_OUT_FAST_READ_4_BYTE_ADDR_CMD	179
7.13.2.9 DUAL_OUT_FAST_READ_CMD	179
7.13.2.10 DUAL_OUT_FAST_READ_DTR_CMD	180
7.13.2.11 ENTER_4_BYTE_ADDR_MODE_CMD	180
7.13.2.12 ENTER_QUAD_CMD	180
7.13.2.13 EXIT_4_BYTE_ADDR_MODE_CMD	180
7.13.2.14 EXIT_QUAD_CMD	180
7.13.2.15 EXT_DUAL_IN_FAST_PROG_CMD	180
7.13.2.16 EXT_QUAD_IN_FAST_PROG_CMD	180
7.13.2.17 FAST_READ_4_BYTE_ADDR_CMD	180

7.13.2.18 FAST_READ_CMD	181
7.13.2.19 FAST_READ_DTR_CMD	181
7.13.2.20 IS25LP080D_DIE_ERASE_MAX_TIME	181
7.13.2.21 IS25LP080D_DUMMY_CYCLES_READ	181
7.13.2.22 IS25LP080D_DUMMY_CYCLES_READ_DTR	181
7.13.2.23 IS25LP080D_DUMMY_CYCLES_READ_QUAD	181
7.13.2.24 IS25LP080D_DUMMY_CYCLES_READ_QUAD_DTR	181
7.13.2.25 IS25LP080D_EAR_HIGHEST_SE	181
7.13.2.26 IS25LP080D_EAR_LOWEST_SEG	182
7.13.2.27 IS25LP080D_EAR_SECOND_SEG	182
7.13.2.28 IS25LP080D_EAR_THIRD_SEG	182
7.13.2.29 IS25LP080D_EVCR_DTRP	182
7.13.2.30 IS25LP080D_EVCR_DUAL	182
7.13.2.31 IS25LP080D_EVCR_ODS	182
7.13.2.32 IS25LP080D_EVCR_QUAD	182
7.13.2.33 IS25LP080D_EVCR_RH	182
7.13.2.34 IS25LP080D_FLASH_SIZE	183
7.13.2.35 IS25LP080D_FSR_ERERR	183
7.13.2.36 IS25LP080D_FSR_ERSUS	183
7.13.2.37 IS25LP080D_FSR_NBADDR	183
7.13.2.38 IS25LP080D_FSR_PGERR	183
7.13.2.39 IS25LP080D_FSR_PGSUS	183
7.13.2.40 IS25LP080D_FSR_PRERR	183
7.13.2.41 IS25LP080D_FSR_READY	183
7.13.2.42 IS25LP080D_NVCR_DTRP	184
7.13.2.43 IS25LP080D_NVCR_DUAL	184
7.13.2.44 IS25LP080D_NVCR_NB_DUMMY	184
7.13.2.45 IS25LP080D_NVCR_NBADDR	184
7.13.2.46 IS25LP080D_NVCR_ODS	184
7.13.2.47 IS25LP080D_NVCR_QUAB	184

7.13.2.48 IS25LP080D_NVCR_RH	184
7.13.2.49 IS25LP080D_NVCR_SEGMENT	184
7.13.2.50 IS25LP080D_NVCR_XIP	185
7.13.2.51 IS25LP080D_PAGE_SIZE	185
7.13.2.52 IS25LP080D_SECTOR_ERASE_MAX_TIME	185
7.13.2.53 IS25LP080D_SECTOR_SIZE	185
7.13.2.54 IS25LP080D_SR_QE	185
7.13.2.55 IS25LP080D_SR_SRWREN	185
7.13.2.56 IS25LP080D_SR_WIP	185
7.13.2.57 IS25LP080D_SR_WREN	186
7.13.2.58 IS25LP080D_SUBSECTOR_ERASE_MAX_TIME	186
7.13.2.59 IS25LP080D_SUBSECTOR_SIZE	186
7.13.2.60 IS25LP080D_VCR_NB_DUMMY	186
7.13.2.61 IS25LP080D_VCR_WRAP	186
7.13.2.62 IS25LP080D_VCR_XIP	186
7.13.2.63 MULTIPLE_IO_READ_ID_CMD	186
7.13.2.64 PAGE_PROG_4_BYTE_ADDR_CMD	186
7.13.2.65 PAGE_PROG_CMD	187
7.13.2.66 PROG_ERASE_RESUME_CMD	187
7.13.2.67 PROG_ERASE_SUSPEND_CMD	187
7.13.2.68 PROG_OTP_ARRAY_CMD	187
7.13.2.69 QUAD_IN_FAST_PROG_4_BYTE_ADDR_CMD	187
7.13.2.70 QUAD_IN_FAST_PROG_CMD	187
7.13.2.71 QUAD_INOUT_FAST_READ_4_BYTE_ADDR_CMD	187
7.13.2.72 QUAD_INOUT_FAST_READ_CMD	187
7.13.2.73 QUAD_INOUT_FAST_READ_DTR_CMD	188
7.13.2.74 QUAD_OUT_FAST_READ_4_BYTE_ADDR_CMD	188
7.13.2.75 QUAD_OUT_FAST_READ_CMD	188
7.13.2.76 QUAD_OUT_FAST_READ_DTR_CMD	188
7.13.2.77 READ_4_BYTE_ADDR_CMD	188

7.13.2.78 READ_CMD	188
7.13.2.79 READ_ENHANCED_VOL_CFG_REG_CMD	188
7.13.2.80 READ_EXT_ADDR_REG_CMD	188
7.13.2.81 READ_FLAG_STATUS_REG_CMD	189
7.13.2.82 READ_ID_CMD	189
7.13.2.83 READ_ID_CMD2	189
7.13.2.84 READ_LOCK_REG_CMD	189
7.13.2.85 READ_NONVOL_CFG_REG_CMD	189
7.13.2.86 READ_OTP_ARRAY_CMD	189
7.13.2.87 READ_READ_PARAM_REG_CMD	189
7.13.2.88 READ_SERIAL_FLASH_DISCO_PARAM_CMD	189
7.13.2.89 READ_STATUS_REG_CMD	190
7.13.2.90 RESET_ENABLE_CMD	190
7.13.2.91 RESET_MEMORY_CMD	190
7.13.2.92 SECTOR_ERASE_4_BYTE_ADDR_CMD	190
7.13.2.93 SECTOR_ERASE_CMD	190
7.13.2.94 SUBSECTOR_ERASE_4_BYTE_ADDR_CMD	190
7.13.2.95 SUBSECTOR_ERASE_CMD	190
7.13.2.96 SUBSECTOR_ERASE_QPI_CMD	190
7.13.2.97 WRITE_DISABLE_CMD	191
7.13.2.98 WRITE_ENABLE_CMD	191
7.13.2.99 WRITE_ENHANCED_VOL_CFG_REG_CMD	191
7.13.2.100WRITE_EXT_ADDR_REG_CMD	191
7.13.2.101WRITE_LOCK_REG_CMD	191
7.13.2.102WRITE_NONVOL_CFG_REG_CMD	191
7.13.2.103WRITE_READ_PARAM_REG_CMD	191
7.13.2.104WRITE_STATUS_REG_CMD	191
7.14 src/dev_leddriver.h File Reference	192
7.14.1 Detailed Description	192
7.14.2 Macro Definition Documentation	192

7.14.2.1	DSY_LED_DRIVER_MAX_DRIVERS	192
7.14.2.2	SA_LED_DRIVER_H	193
7.14.3	Enumeration Type Documentation	193
7.14.3.1	anonymous enum	193
7.14.4	Function Documentation	193
7.14.4.1	dsy_led_driver_color_by_name()	193
7.14.4.2	dsy_led_driver_init()	194
7.14.4.3	dsy_led_driver_set_led()	194
7.14.4.4	dsy_led_driver_update()	194
7.15	src/dev_sdram.h File Reference	194
7.15.1	Macro Definition Documentation	195
7.15.1.1	DSY_SDRAM_BSS	195
7.15.1.2	DSY_SDRAM_DATA	195
7.15.1.3	RAM_AS4C16M16SA_H	195
7.15.2	Enumeration Type Documentation	196
7.15.2.1	anonymous enum	196
7.15.2.2	dsy_sdram_pin	196
7.15.2.3	dsy_sdram_state	196
7.15.3	Function Documentation	196
7.15.3.1	dsy_sdram_init()	197
7.16	src/dev_sr_4021.h File Reference	197
7.16.1	Detailed Description	197
7.16.2	Macro Definition Documentation	197
7.16.2.1	DEV_SR_4021_H	198
7.16.2.2	SR_4021_MAX_DAISSYCHAIN	198
7.16.2.3	SR_4021_MAX_PARALLEL	198
7.16.3	Enumeration Type Documentation	198
7.16.3.1	anonymous enum	198
7.16.4	Function Documentation	198
7.16.4.1	dsy_sr_4021_init()	198

7.16.4.2	<code>dsy_sr_4021_state()</code>	200
7.16.4.3	<code>dsy_sr_4021_update()</code>	200
7.17	<code>src/dev_sr_595.h</code> File Reference	200
7.17.1	Variable Documentation	201
7.17.1.1	<code>kMaxSr595DaisyChain</code>	201
7.18	<code>src/fatfs.h</code> File Reference	201
7.18.1	Detailed Description	201
7.18.2	Macro Definition Documentation	201
7.18.2.1	<code>__fatfs_H</code>	202
7.18.3	Function Documentation	202
7.18.3.1	<code>dsy_fatfs_init()</code>	202
7.18.4	Variable Documentation	202
7.18.4.1	<code>retSD</code>	202
7.18.4.2	<code>SDFatFS</code>	202
7.18.4.3	<code>SDFile</code>	202
7.18.4.4	<code>SDPath</code>	202
7.19	<code>src/ffconf.h</code> File Reference	203
7.19.1	Detailed Description	203
7.19.2	Macro Definition Documentation	204
7.19.2.1	<code>_CODE_PAGE</code>	204
7.19.2.2	<code>_FFCONF</code>	204
7.19.2.3	<code>_FS_EXFAT</code>	204
7.19.2.4	<code>_FS_LOCK</code>	204
7.19.2.5	<code>_FS_MINIMIZE</code>	205
7.19.2.6	<code>_FS_NOFSINFO</code>	205
7.19.2.7	<code>_FS_NORTC</code>	205
7.19.2.8	<code>_FS_READONLY</code>	205
7.19.2.9	<code>_FS_REENTRANT</code>	205
7.19.2.10	<code>_FS_RPATH</code>	205
7.19.2.11	<code>_FS_TIMEOUT</code>	206

7.19.2.12	_FS_TINY	206
7.19.2.13	_LFN_UNICODE	206
7.19.2.14	_MAX_LFN	206
7.19.2.15	_MAX_SS	206
7.19.2.16	_MIN_SS	206
7.19.2.17	_MULTI_PARTITION	207
7.19.2.18	_NORTC_MDAY	207
7.19.2.19	_NORTC_MON	207
7.19.2.20	_NORTC_YEAR	207
7.19.2.21	_STR_VOLUME_ID	207
7.19.2.22	_STRF_ENCODE	207
7.19.2.23	_SYNC_t	208
7.19.2.24	_USE_CHMOD	208
7.19.2.25	_USE_EXPAND	208
7.19.2.26	_USE_FASTSEEK	208
7.19.2.27	_USE_FIND	208
7.19.2.28	_USE_FORWARD	208
7.19.2.29	_USE_LABEL	209
7.19.2.30	_USE_LFN	209
7.19.2.31	_USE_MKFS	209
7.19.2.32	_USE_STRFUNC	209
7.19.2.33	_USE_TRIM	209
7.19.2.34	_VOLUME_STRS	209
7.19.2.35	_VOLUMES	210
7.19.2.36	ff_free	210
7.19.2.37	ff_malloc	210
7.20	src/hid_gatein.h File Reference	210
7.21	src/hid_wavplayer.h File Reference	210
7.21.1	Macro Definition Documentation	211
7.21.1.1	DSY_WAVPLAYER_H	211

7.21.1.2	WAV_FILENAME_MAX	211
7.22	src/per_adc.h File Reference	211
7.22.1	Macro Definition Documentation	211
7.22.1.1	DSY_ADC_H	211
7.22.1.2	DSY_ADC_MAX_CHANNELS	212
7.23	src/per_dac.h File Reference	212
7.23.1	Enumeration Type Documentation	212
7.23.1.1	dsy_dac_bitdepth	212
7.23.1.2	dsy_dac_channel	213
7.23.1.3	dsy_dac_mode	213
7.23.2	Function Documentation	213
7.23.2.1	dsy_dac_init()	213
7.23.2.2	dsy_dac_start()	214
7.23.2.3	dsy_dac_write()	214
7.24	src/per_gpio.h File Reference	214
7.24.1	Detailed Description	215
7.24.2	Enumeration Type Documentation	215
7.24.2.1	dsy_gpio_mode	215
7.24.2.2	dsy_gpio_pull	215
7.24.3	Function Documentation	216
7.24.3.1	dsy_gpio_deinit()	216
7.24.3.2	dsy_gpio_init()	216
7.24.3.3	dsy_gpio_read()	216
7.24.3.4	dsy_gpio_toggle()	217
7.24.3.5	dsy_gpio_write()	217
7.25	src/per_i2c.h File Reference	217
7.25.1	Macro Definition Documentation	218
7.25.1.1	DSY_I2C_H	218
7.25.2	Enumeration Type Documentation	218
7.25.2.1	dsy_i2c_periph	218

7.25.2.2	dsy_i2c_pin	218
7.25.2.3	dsy_i2c_speed	219
7.25.3	Function Documentation	219
7.25.3.1	dsy_i2c_init()	219
7.26	src/per_qspi.h File Reference	219
7.26.1	Macro Definition Documentation	220
7.26.1.1	DSY_MEMORY_ERROR	220
7.26.1.2	DSY_MEMORY_OK	220
7.26.1.3	DSY_QSPI	220
7.26.1.4	DSY_QSPI_BSS	221
7.26.1.5	DSY_QSPI_DATA	221
7.26.1.6	DSY_QSPI_TEXT	221
7.26.2	Enumeration Type Documentation	221
7.26.2.1	dsy_qspi_device	221
7.26.2.2	dsy_qspi_mode	222
7.26.2.3	dsy_qspi_pin	222
7.26.3	Function Documentation	222
7.26.3.1	dsy_qspi_deinit()	222
7.26.3.2	dsy_qspi_erase()	223
7.26.3.3	dsy_qspi_erasesector()	223
7.26.3.4	dsy_qspi_init()	223
7.26.3.5	dsy_qspi_write()	224
7.26.3.6	dsy_qspi_writepage()	224
7.27	src/per_sai.h File Reference	225
7.27.1	Enumeration Type Documentation	225
7.27.1.1	anonymous enum	225
7.27.1.2	dsy_audio_bitdepth	226
7.27.1.3	dsy_audio_device	226
7.27.1.4	dsy_audio_dir	226
7.27.1.5	dsy_audio_sai	227

7.27.1.6	dsy_audio_samplerate	227
7.27.1.7	dsy_audio_sync	227
7.27.1.8	dsy_sai_pin	228
7.27.2	Function Documentation	228
7.27.2.1	dsy_sai_init()	228
7.27.2.2	dsy_sai_init_from_handle()	228
7.28	src/per_sdmmc.h File Reference	230
7.28.1	Macro Definition Documentation	230
7.28.1.1	DSY_SD_ERROR	230
7.28.1.2	DSY_SD_OK	230
7.28.1.3	DSY_SDMMC_H	231
7.28.2	Enumeration Type Documentation	231
7.28.2.1	SdmmcBitWidth	231
7.28.2.2	SdmmcMode	231
7.28.2.3	SdmmcSpeed	231
7.29	src/per_spi.h File Reference	232
7.29.1	Enumeration Type Documentation	232
7.29.1.1	SpiPeriph	232
7.29.1.2	SpiPin	232
7.30	src/per_tim.h File Reference	233
7.30.1	Function Documentation	233
7.30.1.1	dsy_tim_delay_ms()	233
7.30.1.2	dsy_tim_delay_tick()	233
7.30.1.3	dsy_tim_delay_us()	234
7.30.1.4	dsy_tim_get_ms()	234
7.30.1.5	dsy_tim_get_tick()	234
7.30.1.6	dsy_tim_get_us()	234
7.30.1.7	dsy_tim_init()	235
7.30.1.8	dsy_tim_start()	235
7.31	src/per_uart.h File Reference	235

7.31.1	Macro Definition Documentation	235
7.31.1.1	DSY_UART_H	235
7.31.2	Variable Documentation	235
7.31.2.1	kUartMaxBufferSize	236
7.32	src/sys_dma.h File Reference	236
7.32.1	Function Documentation	236
7.32.1.1	dsy_dma_init()	236
7.33	src/sys_system.h File Reference	236
7.33.1	Detailed Description	236
7.33.2	Function Documentation	236
7.33.2.1	dsy_system_delay()	236
7.33.2.2	dsy_system_getnow()	237
7.33.2.3	dsy_system_init()	237
7.33.2.4	dsy_system_jumpto()	237
7.33.2.5	dsy_system_jumptoqspi()	237
7.34	src/usbd_cdc_if.h File Reference	238
7.34.1	Detailed Description	238
7.35	src/usbd_conf.h File Reference	239
7.35.1	Detailed Description	239
7.36	src/usbd_desc.h File Reference	240
7.36.1	Detailed Description	240
7.37	src/util_bsp_sd_diskio.h File Reference	240
7.37.1	Macro Definition Documentation	241
7.37.1.1	BSP_SD_CardInfo	241
7.37.1.2	DSY_BSP_SD_DISKIO_H	241
7.37.1.3	MSD_ERROR	242
7.37.1.4	MSD_ERROR_SD_NOT_PRESENT	242
7.37.1.5	MSD_OK	242
7.37.1.6	SD_DATATIMEOUT	242
7.37.1.7	SD_NOT_PRESENT	242

7.37.1.8	SD_PRESENT	242
7.37.1.9	SD_TRANSFER_BUSY	242
7.37.1.10	SD_TRANSFER_OK	242
7.37.2	Function Documentation	243
7.37.2.1	BSP_SD_AbortCallback()	243
7.37.2.2	BSP_SD_Erase()	243
7.37.2.3	BSP_SD_GetCardInfo()	243
7.37.2.4	BSP_SD_GetCardState()	244
7.37.2.5	BSP_SD_Init()	244
7.37.2.6	BSP_SD_IsDetected()	244
7.37.2.7	BSP_SD_ITConfig()	244
7.37.2.8	BSP_SD_ReadBlocks()	244
7.37.2.9	BSP_SD_ReadBlocks_DMA()	245
7.37.2.10	BSP_SD_ReadCpltCallback()	245
7.37.2.11	BSP_SD_WriteBlocks()	245
7.37.2.12	BSP_SD_WriteBlocks_DMA()	246
7.37.2.13	BSP_SD_WriteCpltCallback()	246
7.38	src/util_color.h File Reference	246
7.38.1	Macro Definition Documentation	247
7.38.1.1	DSY_COLOR_H	247
7.39	src/util_hal_map.h File Reference	247
7.39.1	Function Documentation	247
7.39.1.1	dsy_hal_map_get_i2c()	247
7.39.1.2	dsy_hal_map_get_pin()	248
7.39.1.3	dsy_hal_map_get_port()	248
7.39.2	Variable Documentation	248
7.39.2.1	hi2c1	248
7.39.2.2	hi2c2	249
7.39.2.3	hi2c3	249
7.39.2.4	hi2c4	249
7.40	src/util_ringbuffer.h File Reference	249
7.41	src/util_unique_id.h File Reference	249
7.41.1	Function Documentation	249
7.41.1.1	dsy_get_unique_id()	250
7.42	src/util_wav_format.h File Reference	250

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	9
HUMAN_INTERFACE	10
AUDIO	11
CONTROLS	14
FEEDBACK	22
EXTERNAL	28
PERIPHERAL	34
SYSTEM	35
DEV	36
DAISY	37
UTILITY	38
Externals	60
STM32_USB_OTG_DEVICE_LIBRARY	61
USBDCDCIF	39
USBDCDCIF_Exported_Defines	40
USBDCDCIF_Exported_Types	41
USBDCDCIF_Exported_Macros	42
USBDCDCIF_Exported_Variables	43
USBDCDCIF_Exported_FunctionsPrototype	44
USBDESC	53
USBDESC_Exported_Constants	54
USBDESC_Exported_Defines	55
USBDESC_Exported_TypesDefinitions	56
USBDESC_Exported_Macros	57
USBDESC_Exported_Variables	58
USBDESC_Exported_FunctionsPrototype	59
USB_OTG_DRIVER	62
USBCONF	45
USBCONF_Exported_Variables	46
USBCONF_Exported_Defines	47
USBCONF_Exported_Macros	49
USBCONF_Exported_Types	51
USBCONF_Exported_FunctionsPrototype	52

Chapter 3

Class Index

3.1 Class List

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

daisy::AdcChannelConfig	63
daisy::AdcHandle	65
daisy::AnalogControl	
Hardware Interface for control inputs	
Primarily designed for ADC input controls such as	
potentiometers, and control voltage.	
69	
codec_frame_t	70
color	71
daisy::Color	72
daisy::ControlChangeEvent	74
daisy::daisy_field	74
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	76
daisy::DaisyPetal	
Helpers and hardware definitions for daisy petal	81
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	89
daisy::DaisySeed	
This is the higher-level interface for the Daisy board.	
All basic peripheral configuration/initialization is setup here	95
dsy_audio_handle	98
dsy_dac_handle	99
dsy_gpio	100
dsy_gpio_pin	101
dsy_i2c_handle	102
dsy_qspi_handle	103
dsy_sai_handle	104
DSY_SD_CardInfoTypeDef	106
dsy_sdram_handle	107
dsy_sr_4021_handle	108
daisy::Encoder	
Generic Class for handling Quadrature Encoders	
Inspired/influenced by Mutable Instruments (pichenettes) Encoder classes	110

FontDef	110
daisy::GateIn	
Generic Class for handling gate inputs through GPIO	111
daisy::Led	
LED Class providing simple Software PWM ability, etc	
Eventually this will work with hardware PWM, and external LED Driver devices as well	113
daisy::MidiEvent	113
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	114
daisy::NoteOnEvent	115
daisy::OledDisplay	115
daisy::Parameter	116
daisy::RgbLed	117
daisy::RingBuffer< T, size >	117
daisy::RingBuffer< T, 0 >	121
daisy::SdmmcHandler	124
daisy::SdmmcHandlerInit	124
ShiftRegister595	
Device Driver for 8-bit shift register.	
CD74HC595 - 8-bit serial to parallel output shift	125
daisy::SpiHandle	127
daisy::Switch	128
daisy::UartHandler	129
UsbHandle	
Interface for initializing and using the USB Peripherals on the daisy	132
WAV_FormatTypeDef	136
daisy::WavFileInfo	139
daisy::WavPlayer	140

Chapter 4

File Index

4.1 File List

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

src/ daisy.h	143
src/ daisy_core.h	146
src/ daisy_field.h	
Hardware defines and helpers for daisy field platform	148
src/ daisy_patch.h	155
src/ daisy_petal.h	156
src/ daisy_pod.h	156
src/ daisy_seed.h	156
src/ dev_codec_ak4556.h	
Driver for the AK4556 Stereo Codec	157
src/ dev_codec_pcm3060.h	
Driver for the PCM3060 Codec	157
src/ dev_codec_wm8731.h	
Driver for the WM8731 Codec	158
src/ dev_codec_wm8731_frame.h	
WM8731 Codec framework	160
src/ dev_flash_IS25LP064A.h	
IS25LP08D Commands	160
src/ dev_flash_IS25LP080D.h	
IS25LP08D Commands	176
src/ dev_leddriver.h	
Device driver for PCA9685 16-channel 12-bit PWM generator	192
src/ dev_sdram.h	194
src/ dev_sr_4021.h	
Device driver for the CD4021. Bit-banged serial shift input	197
src/ dev_sr_595.h	200
src/ fatfs.h	
Fatfs support	201
src/ ffconf.h	203
src/ hid_audio.h	??
src/ hid_ctrl.h	??
src/ hid_encoder.h	??
src/ hid_gatein.h	210
src/ hid_led.h	??
src/ hid_midi.h	??

src/hid_oled_display.h	??
src/hid_parameter.h	??
src/hid_rgb_led.h	??
src/hid_switch.h	??
src/hid_usb.h	??
src/hid_wavplayer.h	210
src/per_adc.h	211
src/per_dac.h	212
src/per_gpio.h	214
src/per_i2c.h	217
src/per_qspi.h	219
src/per_sai.h	225
src/per_sdmmc.h	230
src/per_spi.h	232
src/per_tim.h	233
src/per_uart.h	235
src/stm32h7xx_hal_conf.h	??
src/sys_dma.h	236
src/sys_system.h	236
src/usbd_cdc_if.h	
: Header for usbd_cdc_if.c file	238
src/usbd_conf.h	
: Header for usbd_conf.c file	239
src/usbd_desc.h	
: Header for usbd_conf.c file	240
src/util_bsp_sd_diskio.h	240
src/util_color.h	246
src/util_hal_map.h	247
src/util_oled_fonts.h	??
src/util_ringbuffer.h	249
src/util_sd_diskio.h	??
src/util_unique_id.h	249
src/util_wav_format.h	250

Chapter 5

Module Documentation

5.1 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.
- [DEV](#)
Low level development. Led drivers, codecs, etc.
- [DAISY](#)
Daisy devices. Pod, seed, etc.
- [UTILITY](#)
General utilities. Ringbuffers, LED colors, OLED stuff, etc.

5.1.1 Detailed Description

The daisy library.

5.2 HUMAN_INTERFACE

Interface with the world.

Modules

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

5.2.1 Detailed Description

Interface with the world.

5.3 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)

5.3.1 Detailed Description

Embedded Audio Engine.

5.3.2 Typedef Documentation

5.3.2.1 [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 }

5.3.2.2 [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 }
```

5.3.3 Enumeration Type Documentation

5.3.3.1 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	&

5.3.4 Function Documentation

5.3.4.1 dsy_audio_enter_bypass()

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

If the device supports hardware bypass, enter that mode.

5.3.4.2 dsy_audio_exit_bypass()

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

If the device supports hardware bypass, exit that mode.

5.3.4.3 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.

5.3.4.4 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

5.3.4.5 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.

5.3.4.6 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.

5.3.4.7 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

5.3.4.8 dsy_audio_silence()

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

sets outputs to 0 without stopping the Audio Engine

5.3.4.9 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

5.3.4.10 dsy_audio_stop()

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

Stops transmitting/receiving audio on the specified audio block.

5.4 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

5.4.1 Detailed Description

Hardware Controls.

5.4.2 Enumeration Type Documentation

5.4.2.1 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.

5.4.2.2 Polarity

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

Specifies whether the pressed is HIGH or LOW.

Enumerator

POLARITY_NORMAL	&
POLARITY_INVERTED	&

5.4.2.3 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	&

5.4.2.4 Type

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

Specifies the expected behavior of the switch

Enumerator

TYPE_TOGGLE	&
TYPE_MOMENTARY	&

5.4.3 Function Documentation

5.4.3.1 AnalogControl()

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

Constructor

5.4.3.2 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.

5.4.3.3 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.

5.4.3.4 FallingEdge() [1/2]

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

Returns true if the encoder was just released.

5.4.3.5 FallingEdge() [2/2]

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

Returns

true if the button was just released

5.4.3.6 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.

5.4.3.7 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.

5.4.3.8 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.

5.4.3.9 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.

5.4.3.10 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 Debounce() 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

5.4.3.11 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 Debounce() function will be called. (used for timing).

5.4.3.12 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

5.4.3.13 Parameter()

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

Constructor

5.4.3.14 Pressed() [1/2]

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

Returns true while the encoder is held down.

5.4.3.15 Pressed() [2/2]

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

Returns

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

5.4.3.16 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.

5.4.3.17 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

5.4.3.18 RisingEdge() [1/2]

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

Returns true if the encoder was just pressed.

5.4.3.19 RisingEdge() [2/2]

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

Returns

true if a button was just pressed.

5.4.3.20 TimeHeldMs() [1/2]

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

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

5.4.3.21 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)

5.4.3.22 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.

5.4.3.23 Value() [2/2]

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

Returns the current stored value, without reprocessing

5.4.3.24 ~AnalogControl()

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

destructor

5.4.3.25 ~Parameter()

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

Destructor

5.5 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` ()

5.5.1 Detailed Description

Screens, leds, etc.

5.5.2 Enumeration Type Documentation

5.5.2.1 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

5.5.3 Function Documentation

5.5.3.1 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

5.5.3.2 Fill()

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

Fills the entire display with either on/off.

Parameters

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

5.5.3.3 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

5.5.3.4 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)

5.5.3.5 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 <code>OledDisplay::NUM_PINS</code> <code>dsy_gpio_pins</code>
----------------	--

5.5.3.6 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

5.5.3.7 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.
------------	--

5.5.3.8 SetColor()

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

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

Parameters

<i>c</i>	Color object to set.
----------	--------------------------------------

5.5.3.9 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

5.5.3.10 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)

5.5.3.11 Update() [2/3]

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

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

5.5.3.12 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.

5.5.3.13 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

&

5.5.3.14 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

&

5.6 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]

5.6.1 Detailed Description

External interface devices.

5.6.2 Enumeration Type Documentation

5.6.2.1 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	&

5.6.2.2 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	&

5.6.2.3 MidiOutputMode

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

Output mode

Enumerator

OUTPUT_MODE_NONE	&
OUTPUT_MODE_UART1	&
OUTPUT_MODE_USB_INT	&
OUTPUT_MODE_USB_EXT	&

5.6.3 Function Documentation

5.6.3.1 AsControlChange()

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

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

5.6.3.2 AsNoteOn()

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

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

5.6.3.3 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.

5.6.3.4 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

5.6.3.5 Listen()

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

Start listening

5.6.3.6 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>	&
-------------	---

5.6.3.7 PopEvent()

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

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

Returns

The event to be handled

5.6.3.8 StartReceive()

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

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

5.6.4 Variable Documentation

5.6.4.1 channel [1/3]

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

5.6.4.2 channel [2/3]

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

5.6.4.3 channel [3/3]

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

5.6.4.4 control_number

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

5.6.4.5 data

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

5.6.4.6 note

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

5.6.4.7 type

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

5.6.4.8 value

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

5.6.4.9 velocity

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

5.7 PERIPHERAL

Peripheral devices, not meant for human interaction.

Peripheral devices, not meant for human interaction.

5.8 SYSTEM

Deals with system. DMA, clocks, etc.

Deals with system. DMA, clocks, etc.

5.9 DEV

Low level development. Led drivers, codecs, etc.

Low level development. Led drivers, codecs, etc.

5.10 DAISY

Daisy devices. Pod, seed, etc.

Daisy devices. Pod, seed, etc.

5.11 UTILITY

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

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

5.12 USBD_CDC_IF

Usb VCP device module.

Modules

- [USB_D_CDC_IF_Exported_Defines](#)
Defines.
- [USB_D_CDC_IF_Exported_Types](#)
Types.
- [USB_D_CDC_IF_Exported_Macros](#)
Aliases.
- [USB_D_CDC_IF_Exported_Variables](#)
Public variables.
- [USB_D_CDC_IF_Exported_FunctionsPrototype](#)
Public functions declaration.

5.12.1 Detailed Description

Usb VCP device module.

5.13 USB_D_CDC_IF_Exported_Defines

Defines.

Defines.

5.14 USBD_CDC_IF_Exported_Types

Types.

Typedefs

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

5.14.1 Detailed Description

Types.

5.14.2 Typedef Documentation

5.14.2.1 CDC_ReceiveCallback

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

Parameters

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

5.15 USB_D_CDC_IF_Exported_Macros

Aliases.

Aliases.

5.16 USBD_CDC_IF_Exported_Variables

Public variables.

Variables

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

5.16.1 Detailed Description

Public variables.

5.16.2 Variable Documentation

5.16.2.1 USBInterface_fops_FS

USBDCDC_ItfTypeDef USBInterface_fops_FS

CDC Interface callback.

5.16.2.2 USBInterface_fops_HS

USBDCDC_ItfTypeDef USBInterface_fops_HS

CDC Interface callback.

5.17 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)

5.17.1 Detailed Description

Public functions declaration.

5.17.2 Function Documentation

5.17.2.1 CDC_Set_Rx_Callback_FS()

```
void CDC_Set_Rx_Callback_FS (
    CDC_ReceiveCallback cb )
```

&

5.17.2.2 CDC_Transmit_FS()

```
uint8_t CDC_Transmit_FS (
    uint8_t * Buf,
    uint16_t Len )
```

&

5.17.2.3 CDC_Transmit_HS()

```
uint8_t CDC_Transmit_HS (
    uint8_t * Buf,
    uint16_t Len )
```

&

5.18 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.

5.18.1 Detailed Description

Configuration file for Usb otg low level driver.

5.19 USBD_CONF_Exported_Variables

Public variables.

Public variables.

5.20 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`

5.20.1 Detailed Description

Defines for configuration of the Usb device.

5.20.2 Macro Definition Documentation

5.20.2.1 DEVICE_FS

```
#define DEVICE_FS 0
```

FS and HS identification

5.20.2.2 DEVICE_HS

```
#define DEVICE_HS 1
```

&

5.20.2.3 USBD_DEBUG_LEVEL

```
#define USBD_DEBUG_LEVEL 3U
```

&

5.20.2.4 USBD_LPM_ENABLED

```
#define USBD_LPM_ENABLED 0U
```

&

5.20.2.5 USBD_MAX_NUM_CONFIGURATION

```
#define USBD_MAX_NUM_CONFIGURATION 1U
```

&

5.20.2.6 USBD_MAX_NUM_INTERFACES

```
#define USBD_MAX_NUM_INTERFACES 1U
```

&

5.20.2.7 USBD_MAX_STR_DESC_SIZ

```
#define USBD_MAX_STR_DESC_SIZ 512U
```

&

5.20.2.8 USBD_SELF_POWERED

```
#define USBD_SELF_POWERED 1U
```

&

5.20.2.9 USBD_SUPPORT_USER_STRING

```
#define USBD_SUPPORT_USER_STRING 0U
```

&

5.21 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](#)(...)

5.21.1 Detailed Description

Aliases.

5.21.2 Macro Definition Documentation

5.21.2.1 USB_D_DbgLog

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

Value:

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

&

5.21.2.2 USB_D_Delay

```
#define USB_D_Delay HAL_Delay
```

Alias for delay.

5.21.2.3 USBD_ErrLog

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

Value:

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

&

5.21.2.4 USBD_free

```
#define USBD_free free
```

Alias for memory release.

5.21.2.5 USBD_malloc

```
#define USBD_malloc malloc
```

Alias for memory allocation.

5.21.2.6 USBD_memcpy

```
#define USBD_memcpy memcpy
```

Alias for memory copy.

5.21.2.7 USBD_memset

```
#define USBD_memset memset
```

Alias for memory set.

5.21.2.8 USBD_UsrLog

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

Value:

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

&

5.22 USBD_CONF_Exported_Types

Types.

Types.

5.23 USBD_CONF_Exported_FunctionsPrototype

Declaration of public functions for Usb device.

Declaration of public functions for Usb device.

5.24 USBD_DESC

Usb device descriptors module.

Modules

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

5.24.1 Detailed Description

Usb device descriptors module.

5.25 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`

5.25.1 Detailed Description

Constants.

5.25.2 Macro Definition Documentation

5.25.2.1 DEVICE_ID1

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

&

5.25.2.2 DEVICE_ID2

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

&

5.25.2.3 DEVICE_ID3

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

&

5.25.2.4 USB_SIZ_STRING_SERIAL

```
#define USB_SIZ_STRING_SERIAL 0x1A
```

&

5.26 USBD_DESC_Exported_Defines

Defines.

Defines.

5.27 USBD_DESC_Exported_TypesDefinitions

Types.

Types.

5.28 USBD_DESC_Exported_Macros

Aliases.

Aliases.

5.29 USBD_DESC_Exported_Variables

Public variables.

Variables

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

5.29.1 Detailed Description

Public variables.

5.29.2 Variable Documentation

5.29.2.1 FS_Desc

```
USB_DescriptorsTypeDef FS_Desc
```

Descriptor for the Usb device.

5.29.2.2 HS_Desc

```
USB_DescriptorsTypeDef HS_Desc
```

Descriptor for the Usb device.

5.30 USBD_DESC_Exported_FunctionsPrototype

Public functions declaration.

Public functions declaration.

5.31 Externals

5.32 STM32_USB_OTG_DEVICE_LIBRARY

For Usb device.

Modules

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

5.32.1 Detailed Description

For Usb device.

< Define to prevent recursive inclusion -----

5.33 USBD_OTG_DRIVER

Modules

- [USB_CONF](#)

Configuration file for Usb otg low level driver.

5.33.1 Detailed Description

Chapter 6

Class Documentation

6.1 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_pin` mux_pin_ [`MUX_SEL_LAST`]
- `uint8_t` mux_channels_

6.1.1 Detailed Description

Configuration Structure for a given channel

6.1.2 Member Enumeration Documentation

6.1.2.1 MuxPin

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

Which pin to use for multiplexing

Enumerator

MUX_SEL_0	&
MUX_SEL_1	&
MUX_SEL_2	&
MUX_SEL_LAST	&

6.1.3 Member Function Documentation

6.1.3.1 InitMux()

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

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

Parameters

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

6.1.3.2 InitSingle()

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

Initializes a single ADC pin as an ADC.

Parameters

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

6.1.4 Member Data Documentation

6.1.4.1 mux_channels_

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

6.1.4.2 mux_pin_

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

6.1.4.3 pin_

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

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

- [src/per_adc.h](#)

6.2 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)

6.2.1 Detailed Description

Handler for analog to digital conversion

6.2.2 Member Enumeration Documentation

6.2.2.1 OverSampling

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

Supported oversampling amounts

Enumerator

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

6.2.3 Member Function Documentation

6.2.3.1 Get()

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

Single channel getter

Parameters

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

Returns

Converted value

6.2.3.2 GetFloat()

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

Get floating point from single channel

Parameters

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

Returns

Floating point converted value

6.2.3.3 GetMux()

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

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

Parameters

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

Returns

data

6.2.3.4 GetMuxFloat()

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

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

Parameters

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

Returns

Floating point data

6.2.3.5 GetMuxPtr()

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

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

Parameters

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

Returns

Pointer to data

6.2.3.6 GetPtr()

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

Get pointer to a value from a single channel

Parameters

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

Returns

Pointer to converted value

6.2.3.7 Init()

```
void daisy::AdcHandle::Init (
    AdcChannelConfig * cfg,
    size_t num_channels,
    OverSampling ovs = OVS_32 )
```

Initializes the ADC with the pins passed in.

Parameters

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

6.2.3.8 Start()

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

Starts reading from the ADC

6.2.3.9 Stop()

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

Stops reading from the ADC

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

- [src/per_adc.h](#)

6.3 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

6.3.1 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

6.4 codec_frame_t Struct Reference

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

Public Attributes

- short `l`
- short `r`

6.4.1 Detailed Description

&

6.4.2 Member Data Documentation

6.4.2.1 `l`

```
short codec_frame_t::l
```

&

6.4.2.2 r

```
short codec_frame_t::r
```

&

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

- [src/dev_codec_wm8731_frame.h](#)

6.5 color Struct Reference

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

Public Attributes

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

6.5.1 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.

6.5.2 Member Data Documentation

6.5.2.1 blue

```
uint16_t color::blue
```

&

6.5.2.2 green

```
uint16_t color::green
```

&

6.5.2.3 red

```
uint16_t color::red
```

&

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

- [src/dev_leddriver.h](#)

6.6 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

6.6.1 Detailed Description

Class for handling simple colors

6.6.2 Member Enumeration Documentation

6.6.2.1 PresetColor

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

List of colors that have a preset RGB value

Enumerator

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

6.6.3 Member Function Documentation

6.6.3.1 Blue()

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

Returns the 0-1 value for Blue

6.6.3.2 Green()

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

Returns the 0-1 value for Green

6.6.3.3 Init() [1/2]

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

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

Parameters

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

6.6.3.4 Init() [2/2]

```
void daisy::Color::Init (
    float red,
```

```
float green,
float blue )
```

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

Parameters

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

6.6.3.5 Red()

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

Returns the 0-1 value for Red

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

- [src/util_color.h](#)

6.7 daisy::ControlChangeEvent Struct Reference

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

Public Attributes

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

6.7.1 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](#)

6.8 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]

6.8.1 Detailed Description

Struct containing hardware defines and daisy seed

6.8.2 Member Data Documentation

6.8.2.1 cvs

[AnalogControl](#) daisy::daisy_field::cvs[CV_LAST]

Array of cv ins

6.8.2.2 gate_in

[dsy_gpio](#) daisy::daisy_field::gate_in

Gate input.

6.8.2.3 gate_out

[dsy_gpio](#) daisy::daisy_field::gate_out

Gate output

6.8.2.4 keyboard_sr

[dsy_sr_4021_handle](#) daisy::daisy_field::keyboard_sr

Keyboard shift register

6.8.2.5 knobs

[AnalogControl](#) daisy::daisy_field::knobs[KNOB_LAST]

Array of hardware knobs

6.8.2.6 seed

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

Daisy seed

6.8.2.7 switches

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

Array of hardware switches

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

- [src/daisy_field.h](#)

6.9 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

6.9.1 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

6.9.2 Member Enumeration Documentation

6.9.2.1 Ctrl

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

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

6.9.2.2 GateInput

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

Daisy patch gate inputs

Enumerator

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

6.9.3 Constructor & Destructor Documentation

6.9.3.1 DaisyPatch()

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

Constructor

6.9.3.2 ~DaisyPatch()

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

Destructor

6.9.4 Member Function Documentation

6.9.4.1 AudioBlockSize()

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

Get block size

6.9.4.2 AudioCallbackRate()

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

Get callback rate

6.9.4.3 AudioSampleRate()

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

Get sample rate

6.9.4.4 ChangeAudioCallback()

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

Change to a different callback function.

Parameters

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

6.9.4.5 DebounceControls()

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

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

6.9.4.6 DelayMs()

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

Wait some ms before going on.

Parameters

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

6.9.4.7 DisplayControls()

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

Control the display

6.9.4.8 GetCtrlValue()

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

Get value for a partiular control

Parameters

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

6.9.4.9 Init()

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

Initializes the daisy seed, and patch hardware.

6.9.4.10 SetAudioBlockSize()

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

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

Parameters

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

6.9.4.11 StartAdc()

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

Start analog to digital conversion.

6.9.4.12 StartAudio()

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

Start audio output.

Parameters

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

6.9.4.13 UpdateAnalogControls()

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

Call at same rate as reading controls for good reads.

6.9.5 Member Data Documentation

6.9.5.1 controls

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

Array of controls

6.9.5.2 display

`OledDisplay` daisy::DaisyPatch::display

&

6.9.5.3 encoder

`Encoder` daisy::DaisyPatch::encoder

`Encoder` object

6.9.5.4 gate_input

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

Gate inputs

6.9.5.5 gate_output

`dsy_gpio` daisy::DaisyPatch::gate_output

&

6.9.5.6 midi

`MidiHandler` daisy::DaisyPatch::midi

Handles midi

6.9.5.7 seed

`DaisySeed` daisy::DaisyPatch::seed

Seed object

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

- src/[daisy_patch.h](#)

6.10 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]

6.10.1 Detailed Description

Helpers and hardware definitions for daisy petal.

6.10.2 Member Enumeration Documentation

6.10.2.1 FootswitchLed

enum `daisy::DaisyPetal::FootswitchLed`

footswitch leds

Enumerator

FOOTSWITCH_LED_1	&
FOOTSWITCH_LED_2	&
FOOTSWITCH_LED_3	&
FOOTSWITCH_LED_4	&
FOOTSWITCH_LED_LAST	&

6.10.2.2 Knob

enum `daisy::DaisyPetal::Knob`

Knobs

Enumerator

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

6.10.2.3 RingLed

enum `daisy::DaisyPetal::RingLed`

Leds in ringled

Enumerator

RING_LED_1	&
RING_LED_2	&

Enumerator

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

6.10.2.4 Sw

```
enum daisy::DaisyPetal::Sw
```

Switches

Enumerator

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

6.10.3 Constructor & Destructor Documentation

6.10.3.1 DaisyPetal()

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

Constructor

6.10.3.2 ~DaisyPetal()

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

Destructor

6.10.4 Member Function Documentation

6.10.4.1 AudioBlockSize()

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

Get audio block size

6.10.4.2 AudioCallbackRate()

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

Get callback rate

6.10.4.3 AudioSampleRate()

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

Device audio sample rate.

6.10.4.4 ChangeAudioCallback()

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

Change callback function

Parameters

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

6.10.4.5 ClearLeds()

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

Turn all leds off

6.10.4.6 DebounceControls()

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

Debounce inputs.

6.10.4.7 DelayMs()

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

Wait before moving on.

Parameters

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

6.10.4.8 GetExpression()

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

&

6.10.4.9 GetKnobValue()

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

Get value per knob.

Parameters

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

Returns

Floating point knob position.

6.10.4.10 Init()

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

Initialize daisy petal

6.10.4.11 SetAudioBlockSize()

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

Set size of audio blocks.

Parameters

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

6.10.4.12 SetFootswitchLed()

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

Set footswitch LED

Parameters

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

6.10.4.13 SetRingLed()

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

Set ring LED colors

Parameters

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

6.10.4.14 StartAdc()

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

Start analog to digital conversion.

6.10.4.15 StartAudio()

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

Start audio callback

Parameters

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

6.10.4.16 UpdateAnalogControls()

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

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

6.10.4.17 UpdateLeds()

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

Update Leds to values you had set.

6.10.5 Member Data Documentation

6.10.5.1 encoder

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

&

6.10.5.2 expression

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

&

6.10.5.3 footswitch_led

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

&

6.10.5.4 knob

```
AnalogControl daisy::DaisyPetal::knob[KNOB\_LAST]
```

&

6.10.5.5 ring_led

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

&

6.10.5.6 seed

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

&

6.10.5.7 switches

```
Switch daisy::DaisyPetal::switches[SW\_LAST]
```

< &

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

- src/[daisy_petal.h](#)

6.11 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

6.11.1 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

6.11.2 Member Enumeration Documentation

6.11.2.1 Knob

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

Knobs

Enumerator

KNOB_2	&
KNOB_LAST	&

6.11.2.2 Sw

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

Switches

Enumerator

BUTTON_2	&
BUTTON_LAST	&

6.11.3 Member Function Documentation

6.11.3.1 AudioBlockSize()

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

Get block size

6.11.3.2 AudioCallbackRate()

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

Get callback rate

6.11.3.3 AudioSampleRate()

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

Get sample rate

6.11.3.4 ChangeAudioCallback()

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

[Switch](#) callback functions

Parameters

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

6.11.3.5 ClearLeds()

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

Reset Leds

6.11.3.6 DebounceControls()

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

&

6.11.3.7 DelayMs()

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

Wait for a bit

Parameters

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

6.11.3.8 GetKnobValue()

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

&

6.11.3.9 Init()

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

Init related stuff.

6.11.3.10 SetAudioBlockSize()

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

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

Parameters

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

6.11.3.11 StartAdc()

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

Start analog to digital conversion.

6.11.3.12 StartAudio()

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

Start audio callback

Parameters

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

6.11.3.13 UpdateAnalogControls()

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

Call at same rate as analog reads for smooth reading.

6.11.3.14 UpdateLeds()

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

Update Leds to set colors

6.11.4 Member Data Documentation

6.11.4.1 button1

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

&

6.11.4.2 button2

`Switch` daisy::DaisyPod::button2

&

6.11.4.3 buttons

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

&

6.11.4.4 encoder

`Encoder` daisy::DaisyPod::encoder

&

6.11.4.5 knob1

`AnalogControl` daisy::DaisyPod::knob1

&

6.11.4.6 knob2

`AnalogControl` daisy::DaisyPod::knob2

&

6.11.4.7 knobs

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

&

6.11.4.8 led1

`RgbLed` daisy::DaisyPod::led1

&

6.11.4.9 led2

`RgbLed` daisy::DaisyPod::led2

&

6.11.4.10 seed

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

Public Members

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

- `src/daisy_pod.h`

6.12 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`

6.12.1 Detailed Description

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

6.12.2 Member Function Documentation

6.12.2.1 AudioSampleRate()

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

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

6.12.2.2 Configure()

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

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

6.12.2.3 GetPin()

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

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

6.12.2.4 Init()

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

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

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

6.12.2.5 SetAudioBlockSize()

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

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

6.12.2.6 SetLed()

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

Sets the state of the built in LED

6.12.2.7 SetTestPoint()

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

Sets the state of the test point near pin 10

6.12.2.8 StartAudio()

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

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

6.12.3 Member Data Documentation

6.12.3.1 adc

`AdcHandle` daisy::DaisySeed::adc

&

6.12.3.2 audio_handle

`dsy_audio_handle` daisy::DaisySeed::audio_handle

&

6.12.3.3 dac_handle

`dsy_dac_handle` daisy::DaisySeed::dac_handle

&

6.12.3.4 i2c1_handle

`dsy_i2c_handle` daisy::DaisySeed::i2c1_handle

&

6.12.3.5 i2c2_handle

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

&

6.12.3.6 qspi_handle

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

&

6.12.3.7 sai_handle

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

&

6.12.3.8 sdram_handle

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

&

6.12.3.9 usb_handle

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

&

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

- `src/daisy_seed.h`

6.13 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`

6.13.1 Detailed Description

Simple config struct that holds peripheral drivers.

6.13.2 Member Data Documentation

6.13.2.1 block_size

```
size_t dsy_audio_handle::block_size
```

&

6.13.2.2 dev0_i2c

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

&

6.13.2.3 dev1_i2c

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

&

6.13.2.4 sai

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

&

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

- src/hid_audio.h

6.14 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]

6.14.1 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.

6.14.2 Member Data Documentation

6.14.2.1 `bitdepth`

```
dsy_dac_bitdepth dsy_dac_handle::bitdepth
```

&

6.14.2.2 `mode`

```
dsy_dac_mode dsy_dac_handle::mode
```

&

6.14.2.3 `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`

6.15 `dsy_gpio` Struct Reference

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

Public Attributes

- `dsy_gpio_pin pin`
- `dsy_gpio_mode mode`
- `dsy_gpio_pull pull`

6.15.1 Detailed Description

Struct for holding the pin, and configuration

6.15.2 Member Data Documentation

6.15.2.1 mode

[dsy_gpio_mode](#) dsy_gpio::mode

&

6.15.2.2 pin

[dsy_gpio_pin](#) dsy_gpio::pin

&

6.15.2.3 pull

[dsy_gpio_pull](#) dsy_gpio::pull

&

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

- [src/per_gpio.h](#)

6.16 dsy_gpio_pin Struct Reference

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

Public Attributes

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

6.16.1 Detailed Description

Hardware define pins

6.16.2 Member Data Documentation

6.16.2.1 pin

```
uint8_t dsy_gpio_pin::pin
```

number 0-15

6.16.2.2 port

```
dsy_gpio_port dsy_gpio_pin::port
```

&

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

- [src/daisy_core.h](#)

6.17 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](#)

6.17.1 Detailed Description

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

6.17.2 Member Data Documentation

6.17.2.1 periph

```
dsy_i2c_periph dsy_i2c_handle::periph
```

&

6.17.2.2 pin_config

```
dsy_gpio_pin dsy_i2c_handle::pin_config[DSY\_I2C\_PIN\_LAST]
```

&

6.17.2.3 speed

`dsy_i2c_speed` `dsy_i2c_handle::speed`

&

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

- `src/per_i2c.h`

6.18 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`]

6.18.1 Detailed Description

Configuration structure for interfacing with QSPI Driver

6.18.2 Member Data Documentation

6.18.2.1 device

`dsy_qspi_device` `dsy_qspi_handle::device`

&

6.18.2.2 mode

`dsy_qspi_mode` `dsy_qspi_handle::mode`

&

6.18.2.3 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](#)

6.19 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]

6.19.1 Detailed Description

Configuration structure for SAI contains all above settings, and passes them to internal structure for hardware initialization.

6.19.2 Member Data Documentation

6.19.2.1 a_direction

```
dsy_audio_dir dsy_sai_handle::a_direction[DSY_SAI_LAST]
```

&

6.19.2.2 b_direction

```
dsy_audio_dir dsy_sai_handle::b_direction[DSY_SAI_LAST]
```

&

6.19.2.3 bitdepth

[dsy_audio_bitdepth](#) dsy_sai_handle::bitdepth[DSY_SAI_LAST]

&

6.19.2.4 device

[dsy_audio_device](#) dsy_sai_handle::device[DSY_SAI_LAST]

&

6.19.2.5 init

[dsy_audio_sai](#) dsy_sai_handle::init

&

6.19.2.6 sai1_pin_config

[dsy_gpio_pin](#) dsy_sai_handle::sai1_pin_config[DSY_SAI_PIN_LAST]

&

6.19.2.7 sai2_pin_config

[dsy_gpio_pin](#) dsy_sai_handle::sai2_pin_config[DSY_SAI_PIN_LAST]

&

6.19.2.8 samplerate

[dsy_audio_samplerate](#) dsy_sai_handle::samplerate[DSY_SAI_LAST]

&

6.19.2.9 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](#)

6.20 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](#)

6.20.1 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.

6.20.2 Member Data Documentation

6.20.2.1 BlockNbr

```
uint32_t DSY_SD_CardInfoTypeDef::BlockNbr
```

Specifies the Card Capacity in blocks

6.20.2.2 BlockSize

```
uint32_t DSY_SD_CardInfoTypeDef::BlockSize
```

Specifies one block size in bytes

6.20.2.3 CardSpeed

```
uint32_t DSY_SD_CardInfoTypeDef::CardSpeed
```

Specifies the card Speed

6.20.2.4 CardType

```
uint32_t DSY_SD_CardInfoTypeDef::CardType
```

Specifies the card Type

6.20.2.5 CardVersion

```
uint32_t DSY_SD_CardInfoTypeDef::CardVersion
```

Specifies the card version

6.20.2.6 Class

```
uint32_t DSY_SD_CardInfoTypeDef::Class
```

Specifies the class of the card class

6.20.2.7 LogBlockNbr

```
uint32_t DSY_SD_CardInfoTypeDef::LogBlockNbr
```

Specifies the Card logical Capacity in blocks

6.20.2.8 LogBlockSize

```
uint32_t DSY_SD_CardInfoTypeDef::LogBlockSize
```

Specifies logical block size in bytes

6.20.2.9 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](#)

6.21 dsy_sdram_handle Struct Reference

```
#include <dev_sdram.h>
```

Public Attributes

- [dsy_sdram_state](#) state
- [dsy_gpio_pin](#) pin_config [DSY_SDRAM_PIN_LAST]

6.21.1 Detailed Description

Configuration struct for passing to initialization

6.21.2 Member Data Documentation

6.21.2.1 pin_config

```
dsy_gpio_pin dsy_sdram_handle::pin_config[DSY_SDRAM_PIN_LAST]
```

&

6.21.2.2 state

```
dsy_sdram_state dsy_sdram_handle::state
```

&

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

- src/[dev_sdram.h](#)

6.22 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]

6.22.1 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_daisychained is the number of devices in a daisy-chain configuration

6.22.2 Member Data Documentation

6.22.2.1 clk

[dsy_gpio](#) dsy_sr_4021_handle::clk

clk pin

6.22.2.2 cs

[dsy_gpio](#) dsy_sr_4021_handle::cs

cs pin

6.22.2.3 data

[dsy_gpio](#) dsy_sr_4021_handle::data[2]

array of data pins

6.22.2.4 num_daisychained

uint8_t dsy_sr_4021_handle::num_daisychained

Number of devices daisy chained

6.22.2.5 num_parallel

uint8_t dsy_sr_4021_handle::num_parallel

number of devices connected

6.22.2.6 pin_config

[dsy_gpio_pin](#) dsy_sr_4021_handle::pin_config[DSY_SR_4021_PIN_LAST]

used to initialize the [dsy_gpio](#)

6.22.2.7 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](#)

6.23 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

6.23.1 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](#)

6.24 FontDef Struct Reference

```
#include <util_oled_fonts.h>
```

Public Attributes

- const uint8_t [FontWidth](#)
- uint8_t [FontHeight](#)
- const uint16_t * [data](#)

6.24.1 Detailed Description

Utility for displaying fonts on OLED displays
Migrated to work with libdaisy from stm32-ssd1306

Author

afiskon on github.Font struct

6.24.2 Member Data Documentation

6.24.2.1 data

```
const uint16_t* FontDef::data
```

Pointer to data font data array

6.24.2.2 FontHeight

```
uint8_t FontDef::FontHeight
```

Font height in pixels

6.24.2.3 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

6.25 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](#) ()

6.25.1 Detailed Description

Generic Class for handling gate inputs through GPIO.

Author

Stephen Hensley

Date

March 2020

6.25.2 Constructor & Destructor Documentation

6.25.2.1 GateIn()

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

[GateIn](#) Constructor

6.25.2.2 ~GateIn()

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

[GateIn](#)~ Destructor

6.25.3 Member Function Documentation

6.25.3.1 Init()

```
void daisy::GateIn::Init (
    dsy\_gpio\_pin * pin_cfg )
```

[Init](#) Initializes the gate input with specified hardware pin

6.25.3.2 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](#)

6.26 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](#) ()

6.26.1 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](#)

6.27 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]

6.27.1 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

6.28 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](#) ()

6.28.1 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

6.29 daisy::NoteOnEvent Struct Reference

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

Public Attributes

- int [channel](#)
- uint8_t [note](#)
- uint8_t [velocity](#)

6.29.1 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

6.30 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](#) ()

6.30.1 Detailed Description

Human Interface Driver for using an OLED Display (SSD1309) For all `bool on` arguments: true is on, false is off. Credit to Aleksander Alekseev (github.com/afiskon/stm32-ssd1306) on github for a great starting point. adapted for SSD1309 and H7 by shensley, 2020

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

- `src/hid_oled_display.h`

6.31 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](#) ()

6.31.1 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`

6.32 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](#) ()

6.32.1 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

6.33 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)

6.33.1 Detailed Description

```
template<typename T, size_t size>
class daisy::RingBuffer< T, size >
```

Utility Ring Buffer
imported from pichenettes/stmlib

6.33.2 Member Function Documentation

6.33.2.1 capacity()

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

Returns

The total size of the ring buffer

6.33.2.2 Flush()

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

Flushes unread elements from the ring buffer

6.33.2.3 ImmediateRead() [1/2]

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

Reads next element from ring buffer immediately

Returns

read value

6.33.2.4 ImmediateRead() [2/2]

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

Reads a number of elements into a buffer immediately

Parameters

<i>destination</i>	buffer to write to
<i>num_elements</i>	number of elements in buffer

6.33.2.5 Init()

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

Initializes the Ring Buffer

6.33.2.6 Overwrite() [1/2]

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

Writes the new element to the ring buffer, overwriting unread data if necessary.

Parameters

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

6.33.2.7 Overwrite() [2/2]

```
template<typename T, size_t size>
void daisy::RingBuffer< T, size >::Overwrite (
    const T * source,
    size_t num_elements ) [inline]
```

Overwrites a number of elements using the source buffer as input.

Parameters

<i>source</i>	Input buffer
<i>num_elements</i>	Number of elements in source

6.33.2.8 Read()

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

Reads the first available element from the ring buffer

Returns

read value

6.33.2.9 readable()

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

Returns

number of unread elements in ring buffer

6.33.2.10 Swallow()

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

Read enough samples to make it possible to read 1 sample.

Parameters

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

6.33.2.11 writable()

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

Returns

the number of samples that can be written to ring buffer without overwriting unread data.

6.33.2.12 Write()

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

Writes the value to the next available position in the ring buffer

Parameters

v	Value to write
-----	----------------

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

- src/[util_ringbuffer.h](#)

6.34 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)

6.34.1 Detailed Description

```
template<typename T>  
class daisy::RingBuffer< T, 0 >
```

Utility Ring Buffer imported from pichenettes/stmlib

6.34.2 Member Function Documentation

6.34.2.1 capacity()

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

Returns

0

6.34.2.2 Flush()

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

Flush the buffer

6.34.2.3 ImmediateRead() [1/2]

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

Returns

Read value

6.34.2.4 ImmediateRead() [2/2]

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

Parameters

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

6.34.2.5 Init()

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

Initialize ringbuffer

6.34.2.6 Overwrite() [1/2]

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

Parameters

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

6.34.2.7 Overwrite() [2/2]

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

Parameters

<i>source</i>	3
<i>num_elements</i>	&

6.34.2.8 Read()

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

Returns

Read value

6.34.2.9 readable()

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

Returns

0

6.34.2.10 writable()

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

Returns

0

6.34.2.11 Write()

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

Parameters

v	Value to write
---	----------------

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

- src/[util_ringbuffer.h](#)

6.35 daisy::SdmmcHandler Class Reference

```
#include <per_sdmmc.h>
```

Public Member Functions

- void [Init](#) ()

6.35.1 Detailed Description

Configuration for interfacing with SD cards. Currently only supports operation using FatFS filesystem

6.35.2 Member Function Documentation

6.35.2.1 Init()

```
void daisy::SdmmcHandler::Init ( )
```

Initializes the SD Card Interface For now all settings are fixed (See todo at top of section)

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

- src/[per_sdmmc.h](#)

6.36 daisy::SdmmcHandlerInit Struct Reference

```
#include <per_sdmmc.h>
```

Public Attributes

- [SdmmcBitWidth](#) bitdepth
- [SdmmcSpeed](#) speed

6.36.1 Detailed Description

Structure for setting the options above. Used to initialize [SdmmcHandler](#)

6.36.2 Member Data Documentation

6.36.2.1 bitdepth

[SdmmcBitWidth](#) daisy::SdmmcHandlerInit::bitdepth

&

6.36.2.2 speed

[SdmmcSpeed](#) daisy::SdmmcHandlerInit::speed

&

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

- [src/per_sdmmc.h](#)

6.37 ShiftRegister595 Class Reference

Device Driver for 8-bit shift register.

CD74HC595 - 8-bit serial to parallel output shift.

```
#include <dev_sr_595.h>
```

Public Types

- enum [Pins](#) { [PIN_LATCH](#), [PIN_CLK](#), [PIN_DATA](#), [NUM_PINS](#) }

Public Member Functions

- void [Init](#) ([dsy_gpio_pin](#) *pin_cfg, size_t num_daisy_chained=1)
- void [Set](#) (uint8_t idx, bool state)
- void [Write](#) ()

6.37.1 Detailed Description

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

Author

shensley

Date

May 2020

6.37.2 Member Enumeration Documentation

6.37.2.1 Pins

```
enum ShiftRegister595::Pins
```

The following pins correspond to the hardware connections to the 595.

Enumerator

PIN_CLK	LATCH corresponds to Pin 12 "RCLK"
PIN_DATA	CLK corresponds to Pin 11 "SRCLK"
NUM_PINS	DATA corresponds to Pin 14 "SER"

6.37.3 Member Function Documentation

6.37.3.1 Init()

```
void ShiftRegister595::Init (
    dsy_gpio_pin * pin_cfg,
    size_t num_daisy_chained = 1 )
```

Initializes the GPIO, and data for the ShiftRegister

Parameters

<i>pin_cfg</i>	is an array of dsy_gpio_pin corresponding the the Pins enum above.
<i>num_daisy_chained</i>	(default = 1) is the number of 595 devices daisy chained together.

6.37.3.2 Set()

```
void ShiftRegister595::Set (
    uint8_t idx,
    bool state )
```

Sets the state of the specified output.

Parameters

<i>idx</i>	The index starts with QA on the first device and ends with QH on the last device.
<i>state</i>	A true state will set the output HIGH, while a false state will set the output LOW.

6.37.3.3 Write()

```
void ShiftRegister595::Write ( )
```

Writes the states of shift register out to the connected devices.

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

- [src/dev_sr_595.h](#)

6.38 daisy::SpiHandle Class Reference

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

Public Member Functions

- void [Init](#) ()
- void [BlockingTransmit](#) (uint8_t *buff, size_t size)

6.38.1 Detailed Description

Handler for serial peripheral interface

6.38.2 Member Function Documentation

6.38.2.1 BlockingTransmit()

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

Blocking transmit

Parameters

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

6.38.2.2 Init()

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

Initializes handler

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

- [src/per_spi.h](#)

6.39 daisy::Switch Class Reference

```
#include <hid_switch.h>
```

Public Types

- enum [Type](#) { [TYPE_TOGGLE](#), [TYPE_MOMENTARY](#) }
- enum [Polarity](#) { [POLARITY_NORMAL](#), [POLARITY_INVERTED](#) }
- enum [Pull](#) { [PULL_UP](#), [PULL_DOWN](#), [PULL_NONE](#) }

Public Member Functions

- void [Init](#) ([dsy_gpio_pin](#) pin, float update_rate, [Type](#) t, [Polarity](#) pol, [Pull](#) pu)
- void [Init](#) ([dsy_gpio_pin](#) pin, float update_rate)
- void [Debounce](#) ()
- bool [RisingEdge](#) () const
- bool [FallingEdge](#) () const
- bool [Pressed](#) () const
- float [TimeHeldMs](#) () const

6.39.1 Detailed Description

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

Author

Stephen Hensley

Date

December 2019

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

- [src/hid_switch.h](#)

6.40 daisy::UartHandler Class Reference

```
#include <per_uart.h>
```

Public Member Functions

- void [Init](#) ()
- int [PollReceive](#) (uint8_t *buff, size_t size, uint32_t timeout)
- int [StartRx](#) (size_t size)
- bool [RxActive](#) ()
- int [FlushRx](#) ()
- int [PollTx](#) (uint8_t *buff, size_t size)
- uint8_t [PopRx](#) ()
- size_t [Readable](#) ()
- int [CheckError](#) ()

6.40.1 Detailed Description

Uart Peripheral

Author

shensley

Date

March 2020

6.40.2 Member Function Documentation

6.40.2.1 CheckError()

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

Returns

the result of HAL_UART_GetError() to the user.

6.40.2.2 FlushRx()

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

Flushes the Receive Queue

Returns

OK or ERROR

6.40.2.3 Init()

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

Initializes the UART Peripheral

6.40.2.4 PollReceive()

```
int daisy::UartHandler::PollReceive (
    uint8_t * buff,
    size_t size,
    uint32_t timeout )
```

Reads the amount of bytes in blocking mode with a 10ms timeout.

Parameters

<i>*buff</i>	Buffer to read to
<i>size</i>	Buff size
<i>timeout</i>	How long to timeout for (10ms?)

Returns

Data received

6.40.2.5 PollTx()

```
int daisy::UartHandler::PollTx (
    uint8_t * buff,
    size_t size )
```

Sends an amount of data in blocking mode.

Parameters

<i>*buff</i>	Buffer of data to send
<i>size</i>	Buffer size

Returns

OK or ERROR

6.40.2.6 PopRx()

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

Pops the oldest byte from the FIFO.

Returns

Popped byte

6.40.2.7 Readable()

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

Checks if there are any unread bytes in the FIFO

Returns

1 or 0 ??

6.40.2.8 RxActive()

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

Returns

whether Rx DMA is listening or not.

6.40.2.9 StartRx()

```
int daisy::UartHandler::StartRx (
    size_t size )
```

Starts a DMA Receive callback to fill a buffer of specified size. Data is populated into a FIFO queue, and can be queried with the functions below. Maximum Buffer size is defined above. If a value outside of the maximum is specified, the size will be set to the maximum.

Parameters

<i>size</i>	Queue size
-------------	------------

Returns

OK or ERROR

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

- [src/per_uart.h](#)

6.41 UsbHandle Class Reference

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

```
#include <hid_usb.h>
```

Public Types

- enum [UsbPeriph](#) {
 [FS_INTERNAL](#), [FS_EXTERNAL](#), [FS_BOTH](#), [FS_INTERNAL](#),
 [FS_EXTERNAL](#), [FS_BOTH](#) }
- enum [UsbPeriph](#) {
 [FS_INTERNAL](#), [FS_EXTERNAL](#), [FS_BOTH](#), [FS_INTERNAL](#),
 [FS_EXTERNAL](#), [FS_BOTH](#) }
- typedef void(* [ReceiveCallback](#)) (uint8_t *buff, uint32_t *len)
- typedef void(* [ReceiveCallback](#)) (uint8_t *buff, uint32_t *len)

Public Member Functions

- void [Init](#) ([UsbPeriph](#) dev)
- void [TransmitInternal](#) (uint8_t *buff, size_t size)
- void [TransmitExternal](#) (uint8_t *buff, size_t size)
- void [SetReceiveCallback](#) ([ReceiveCallback](#) cb)
- void [Init](#) ([UsbPeriph](#) dev)
- void [TransmitInternal](#) (uint8_t *buff, size_t size)
- void [TransmitExternal](#) (uint8_t *buff, size_t size)
- void [SetReceiveCallback](#) ([ReceiveCallback](#) cb)

6.41.1 Detailed Description

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

Author

Stephen Hensley

Date

December 2019

6.41.2 Member Typedef Documentation

6.41.2.1 ReceiveCallback [1/2]

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

Function called upon reception of a buffer

6.41.2.2 ReceiveCallback [2/2]

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

Function called upon reception of a buffer

6.41.3 Member Enumeration Documentation

6.41.3.1 UsbPeriph [1/2]

```
enum UsbHandle::UsbPeriph
```

Specified which of the two USB Peripherals to initialize.

Enumerator

FS_INTERNAL	Internal pin
FS_EXTERNAL	FS External D+ pin is Pin 38 (GPIO32). FS External D- pin is Pin 37 (GPIO31)
FS_BOTH	Both
FS_INTERNAL	Internal pin
FS_EXTERNAL	FS External D+ pin is Pin 38 (GPIO32). FS External D- pin is Pin 37 (GPIO31)
FS_BOTH	Both

6.41.3.2 UsbPeriph [2/2]

```
enum UsbHandle::UsbPeriph
```

Specified which of the two USB Peripherals to initialize.

Enumerator

FS_INTERNAL	Internal pin
-------------	--------------

Enumerator

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

6.41.4 Member Function Documentation

6.41.4.1 Init() [1/2]

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

Initializes the specified peripheral(s) as USB CDC Devices

Parameters

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

6.41.4.2 Init() [2/2]

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

Initializes the specified peripheral(s) as USB CDC Devices

Parameters

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

6.41.4.3 SetReceiveCallback() [1/2]

```
void UsbHandle::SetReceiveCallback (
    ReceiveCallback cb )
```

sets the callback to be called upon reception of new data

Parameters

<i>cb</i>	Function to serve as callback
-----------	-------------------------------

6.41.4.4 SetReceiveCallback() [2/2]

```
void UsbHandle::SetReceiveCallback (
    ReceiveCallback cb )
```

sets the callback to be called upon reception of new data

Parameters

<i>cb</i>	Function to serve as callback
-----------	-------------------------------

6.41.4.5 TransmitExternal() [1/2]

```
void UsbHandle::TransmitExternal (
    uint8_t * buff,
    size_t size )
```

Transmits a buffer of 'size' bytes from a USB port connected to the external USB Pins of the daisy seed.

Parameters

<i>buff</i>	Buffer to transmit
<i>size</i>	Buffer size

6.41.4.6 TransmitExternal() [2/2]

```
void UsbHandle::TransmitExternal (
    uint8_t * buff,
    size_t size )
```

Transmits a buffer of 'size' bytes from a USB port connected to the external USB Pins of the daisy seed.

Parameters

<i>buff</i>	Buffer to transmit
<i>size</i>	Buffer size

6.41.4.7 TransmitInternal() [1/2]

```
void UsbHandle::TransmitInternal (
    uint8_t * buff,
    size_t size )
```

Transmits a buffer of 'size' bytes from the on board USB FS port.

Parameters

<i>buff</i>	Buffer to transmit
<i>size</i>	Buffer size

6.41.4.8 TransmitInternal() [2/2]

```
void UsbHandle::TransmitInternal (
    uint8_t * buff,
    size_t size )
```

Transmits a buffer of 'size' bytes from the on board USB FS port.

Parameters

<i>buff</i>	Buffer to transmit
<i>size</i>	Buffer size

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

- `src/hid_usb.h`

6.42 WAV_FormatTypeDef Struct Reference

```
#include <util_wav_format.h>
```

Public Attributes

- uint32_t [ChunkId](#)
- uint32_t [FileSize](#)
- uint32_t [FileFormat](#)
- uint32_t [SubChunk1ID](#)
- uint32_t [SubChunk1Size](#)
- uint16_t [AudioFormat](#)

- `uint16_t` [NbrChannels](#)
- `uint32_t` [SampleRate](#)
- `uint32_t` [ByteRate](#)
- `uint16_t` [BlockAlign](#)
- `uint16_t` [BitPerSample](#)
- `uint32_t` [SubChunk2ID](#)
- `uint32_t` [SubCHunk2Size](#)

6.42.1 Detailed Description

Helper struct for handling the WAV file format

6.42.2 Member Data Documentation

6.42.2.1 AudioFormat

`uint16_t WAV_FormatTypeDef::AudioFormat`

&

6.42.2.2 BitPerSample

`uint16_t WAV_FormatTypeDef::BitPerSample`

&

6.42.2.3 BlockAlign

`uint16_t WAV_FormatTypeDef::BlockAlign`

&

6.42.2.4 ByteRate

`uint32_t WAV_FormatTypeDef::ByteRate`

&

6.42.2.5 ChunkId

`uint32_t WAV_FormatTypeDef::ChunkId`

&

6.42.2.6 FileFormat

```
uint32_t WAV_FormatTypeDef::FileFormat
```

&

6.42.2.7 FileSize

```
uint32_t WAV_FormatTypeDef::FileSize
```

&

6.42.2.8 NbrChannels

```
uint16_t WAV_FormatTypeDef::NbrChannels
```

&

6.42.2.9 SampleRate

```
uint32_t WAV_FormatTypeDef::SampleRate
```

&

6.42.2.10 SubChunk1ID

```
uint32_t WAV_FormatTypeDef::SubChunk1ID
```

&

6.42.2.11 SubChunk1Size

```
uint32_t WAV_FormatTypeDef::SubChunk1Size
```

&

6.42.2.12 SubChunk2ID

```
uint32_t WAV_FormatTypeDef::SubChunk2ID
```

&

6.42.2.13 SubCHunk2Size

```
uint32_t WAV_FormatTypeDef::SubChunk2Size
```

&

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

- [src/util_wav_format.h](#)

6.43 daisy::WavFileInfo Struct Reference

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

Public Attributes

- [WAV_FormatTypeDef raw_data](#)
- char [name](#) [256]

6.43.1 Detailed Description

Struct containing details of Wav File.

6.43.2 Member Data Documentation

6.43.2.1 name

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

Wav filename

6.43.2.2 raw_data

```
WAV\_FormatTypeDef daisy::WavFileInfo::raw_data
```

Raw wav data

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

- [src/hid_wavplayer.h](#)

6.44 daisy::WavPlayer Class Reference

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

Public Member Functions

- void [Init](#) ()
- int [Open](#) (size_t sel)
- int [Close](#) ()
- int16_t [Stream](#) ()
- void [Prepare](#) ()
- void [Restart](#) ()
- void [SetLooping](#) (bool loop)
- bool [GetLooping](#) () const
- size_t [GetNumberFiles](#) () const
- size_t [GetCurrentFile](#) () const

6.44.1 Detailed Description

Wav Player that will load .wav files from an SD Card, and then provide a method of accessing the samples with double-buffering.

6.44.2 Member Function Documentation

6.44.2.1 Close()

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

Closes whatever file is currently open.

Returns

&

6.44.2.2 GetCurrentFile()

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

Returns

currently selected file.

6.44.2.3 GetLooping()

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

Returns

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

6.44.2.4 GetNumberFiles()

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

Returns

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

6.44.2.5 Init()

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

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

6.44.2.6 Open()

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

Opens the file at index sel for reading.

Parameters

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

6.44.2.7 Prepare()

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

Collects buffer for playback when needed.

6.44.2.8 Restart()

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

Resets the playback position to the beginning of the file immediately

6.44.2.9 SetLooping()

```
void daisy::WavPlayer::SetLooping (
    bool loop ) [inline]
```

Sets whether or not the current file will repeat after completing playback.

Parameters

<i>loop</i>	To loop or not to loop.
-------------	-------------------------

6.44.2.10 Stream()

```
int16_t daisy::WavPlayer::Stream ( )
```

Returns

The next sample if playing, otherwise returns 0

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

- [src/hid_wavplayer.h](#)

Chapter 7

File Documentation

7.1 src/daisy.h File Reference

```
#include <stdint.h>
#include "daisy_core.h"
#include "sys_system.h"
#include "per_qspi.h"
#include "per_dac.h"
#include "per_gpio.h"
#include "per_i2c.h"
#include "per_sai.h"
#include "per_tim.h"
#include "dev_leddriver.h"
#include "dev_sdram.h"
#include "dev_sr_4021.h"
#include "hid_audio.h"
#include "util_unique_id.h"
#include "per_adc.h"
#include "per_uart.h"
#include "hid_midi.h"
#include "hid_encoder.h"
#include "hid_switch.h"
#include "hid_ctrl.h"
#include "hid_gatein.h"
#include "hid_parameter.h"
#include "hid_usb.h"
#include "per_sdmmc.h"
#include "per_spi.h"
#include "hid_oled_display.h"
#include "hid_wavplayer.h"
#include "hid_led.h"
#include "hid_rgb_led.h"
#include "dev_sr_595.h"
```

Macros

- `#define FBIPMAX 0.999985f`
- `#define FBIPMIN (-FBIPMAX)`

- `#define S162F_SCALE 3.0517578125e-05f`
- `#define F2S16_SCALE 32767.0f`
- `#define F2S24_SCALE 8388608.0f`
- `#define S242F_SCALE 1.192092896e-07f`
- `#define S24SIGN 0x800000`

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)`

7.1.1 Macro Definition Documentation

7.1.1.1 F2S16_SCALE

```
#define F2S16_SCALE 32767.0f
```

$(2^{**} 15) - 1$

7.1.1.2 F2S24_SCALE

```
#define F2S24_SCALE 8388608.0f
```

$2^{**} 23$

7.1.1.3 FBIPMAX

```
#define FBIPMAX 0.999985f
```

close to 1.0f-LSB at 16 bit

7.1.1.4 FBIPMIN

```
#define FBIPMIN (-FBIPMAX)
```

- $(1 - \text{LSB})$

7.1.1.5 S162F_SCALE

```
#define S162F_SCALE 3.0517578125e-05f
1 / (2** 15)
```

7.1.1.6 S242F_SCALE

```
#define S242F_SCALE 1.192092896e-07f
1 / (2 ** 23)
```

7.1.1.7 S24SIGN

```
#define S24SIGN 0x800000
2 ** 23
```

7.1.2 Function Documentation

7.1.2.1 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

7.1.2.2 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

7.1.2.3 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.

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

7.1.2.4 s242f()

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

$\# < 2^{**} 23$

$< 2^{**} 23$

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

7.2 src/daisy_core.h File Reference

```
#include <stdint.h>
#include <stdlib.h>
```

Classes

- struct [dsy_gpio_pin](#)

Macros

- #define [DSY_CORE_HW_H](#)
- #define [DMA_BUFFER_MEM_SECTION](#) __attribute__((section(".sram1_bss")))
- #define [DTCM_MEM_SECTION](#) __attribute__((section(".dtdcmram_bss")))

Enumerations

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

Functions

- FORCE_INLINE float [cube](#) (float x)
- FORCE_INLINE [dsy_gpio_pin](#) [dsy_pin](#) ([dsy_gpio_port](#) port, uint8_t pin)
- FORCE_INLINE uint8_t [dsy_pin_cmp](#) ([dsy_gpio_pin](#) *a, [dsy_gpio_pin](#) *b)

7.2.1 Macro Definition Documentation

7.2.1.1 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.

7.2.1.2 DSY_CORE_HW_H

```
#define DSY_CORE_HW_H
```

&

7.2.1.3 DTCM_MEM_SECTION

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

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

7.2.2 Enumeration Type Documentation

7.2.2.1 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	&

7.2.3 Function Documentation

7.2.3.1 cube()

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

Computes cube.

Parameters

x	Number to be cubed
---	--------------------

Returns

x^3

7.2.3.2 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

7.2.3.3 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

7.3 src/daisy_field.h File Reference

Hardware defines and helpers for daisy field platform.

```
#include "daisy_seed.h"
```

Classes

- struct [daisy::daisy_field](#)

Macros

- #define [DSY_FIELD_BSP_H](#)
- #define [SAMPLE_RATE](#) DSY_AUDIO_SAMPLE_RATE
- #define [SW_1_PIN](#) 29
- #define [SW_2_PIN](#) 28
- #define [SW_3_PIN](#) 27
- #define [GATE_OUT_PIN](#) 0
- #define [GATE_IN_PIN](#) 1
- #define [KB_SW_SR_CS_PIN](#) 8
- #define [KB_SW_SR_CLK_PIN](#) 9
- #define [KB_SW_SR_D1_PIN](#) 10
- #define [KB_SW_SR_D2_PIN](#) 11
- #define [MIDI_OUT_PIN](#) 14
- #define [MIDI_IN_PIN](#) 15
- #define [MUX_SEL_0_PIN](#) 21
- #define [MUX_SEL_1_PIN](#) 20
- #define [MUX_SEL_2_PIN](#) 19
- #define [MUX_ADC_PIN](#) 16
- #define [CV1_ADC_PIN](#) 17
- #define [CV2_ADC_PIN](#) 18
- #define [CV3_ADC_PIN](#) 23
- #define [CV4_ADC_PIN](#) 22
- #define [LED_DRIVER_I2C](#) i2c1_handle

Enumerations

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

Functions

- **FORCE_INLINE** void [daisy::daisy_field_init](#) ([daisy_field](#) *p)

7.3.1 Detailed Description

Hardware defines and helpers for daisy field platform.

7.3.2 Macro Definition Documentation

7.3.2.1 CV1_ADC_PIN

```
#define CV1_ADC_PIN 17
```

&

7.3.2.2 CV2_ADC_PIN

```
#define CV2_ADC_PIN 18
```

&

7.3.2.3 CV3_ADC_PIN

```
#define CV3_ADC_PIN 23
```

&

7.3.2.4 CV4_ADC_PIN

```
#define CV4_ADC_PIN 22
```

&

7.3.2.5 DSY_FIELD_BSP_H

```
#define DSY_FIELD_BSP_H
```

&

7.3.2.6 GATE_IN_PIN

```
#define GATE_IN_PIN 1
```

&

7.3.2.7 GATE_OUT_PIN

```
#define GATE_OUT_PIN 0
```

&

7.3.2.8 KB_SW_SR_CLK_PIN

```
#define KB_SW_SR_CLK_PIN 9
```

&

7.3.2.9 KB_SW_SR_CS_PIN

```
#define KB_SW_SR_CS_PIN 8
```

&

7.3.2.10 KB_SW_SR_D1_PIN

```
#define KB_SW_SR_D1_PIN 10
```

&

7.3.2.11 KB_SW_SR_D2_PIN

```
#define KB_SW_SR_D2_PIN 11
```

&

7.3.2.12 LED_DRIVER_I2C

```
#define LED_DRIVER_I2C i2c1_handle
```

&

7.3.2.13 MIDI_IN_PIN

```
#define MIDI_IN_PIN 15
```

&

7.3.2.14 MIDI_OUT_PIN

```
#define MIDI_OUT_PIN 14
```

&

7.3.2.15 MUX_ADC_PIN

```
#define MUX_ADC_PIN 16  
&
```

7.3.2.16 MUX_SEL_0_PIN

```
#define MUX_SEL_0_PIN 21  
&
```

7.3.2.17 MUX_SEL_1_PIN

```
#define MUX_SEL_1_PIN 20  
&
```

7.3.2.18 MUX_SEL_2_PIN

```
#define MUX_SEL_2_PIN 19  
&
```

7.3.2.19 SAMPLE_RATE

```
#define SAMPLE_RATE DSY_AUDIO_SAMPLE_RATE  
&
```

7.3.2.20 SW_1_PIN

```
#define SW_1_PIN 29  
&
```

7.3.2.21 SW_2_PIN

```
#define SW_2_PIN 28  
&
```

7.3.2.22 SW_3_PIN

```
#define SW_3_PIN 27  
&
```

7.3.3 Enumeration Type Documentation

7.3.3.1 anonymous enum

anonymous enum
enums for controls, etc.

Enumerator

SW_2	tactile switch
SW_1	tactile switch
SW_3	toggle
SW_LAST	&

7.3.3.2 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	&

7.3.3.3 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 &

7.3.3.4 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	&
LED_KNOB↵ _8	&
LED_SW_1	&
LED_SW_2	&
LED_LAST	&

7.3.4 Function Documentation

7.3.4.1 daisy_field_init()

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

Initializes daisy field

Parameters

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

< &

< &

< &

< &

< &

< &

< &

< &

< &

< &

< &

< &

< &

< &

< &

< &

< &

< &

7.4 src/daisy_patch.h File Reference

```
#include "daisy_seed.h"
```

Classes

- 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.*

7.5 src/daisy_petal.h File Reference

```
#include "daisy_seed.h"
```

Classes

- class [daisy::DaisyPetal](#)
Helpers and hardware definitions for daisy petal.

Macros

- #define [DSY_PETAL_H](#)

7.5.1 Macro Definition Documentation

7.5.1.1 DSY_PETAL_H

```
#define DSY_PETAL_H
```

&

7.6 src/daisy_pod.h File Reference

```
#include "daisy_seed.h"
```

Classes

- 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.*

7.7 src/daisy_seed.h File Reference

```
#include "daisy.h"
```

Classes

- class [daisy::DaisySeed](#)

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

7.8 src/dev_codec_ak4556.h File Reference

Driver for the AK4556 Stereo Codec.

```
#include "daisy_core.h"
```

Functions

- void [codec_ak4556_init](#) ([dsy_gpio_pin](#) reset_pin)

7.8.1 Detailed Description

Driver for the AK4556 Stereo Codec.

7.8.2 Function Documentation

7.8.2.1 codec_ak4556_init()

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

Resets the AK4556

Parameters

reset_pin	should be a dsy_gpio_pin that is connected to the RST pin of the AK4556
---------------------------	---

7.9 src/dev_codec_pcm3060.h File Reference

Driver for the PCM3060 Codec.

```
#include "per_i2c.h"
```

Functions

- void `codec_pcm3060_init` (`dsy_i2c_handle` *hi2c)

7.9.1 Detailed Description

Driver for the PCM3060 Codec.

7.9.2 Function Documentation

7.9.2.1 `codec_pcm3060_init()`

```
void codec_pcm3060_init (
    dsy_i2c_handle * hi2c )
```

Resets the PCM060

Parameters

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

7.10 `src/dev_codec_wm8731.h` File Reference

Driver for the WM8731 Codec.

```
#include <stddef.h>
#include "per_sai.h"
#include "per_i2c.h"
```

Functions

- 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)

7.10.1 Detailed Description

Driver for the WM8731 Codec.

7.10.2 Function Documentation

7.10.2.1 codec_wm8731_enter_bypass()

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

Put codec into bypass mode

Parameters

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

7.10.2.2 codec_wm8731_exit_bypass()

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

Take codec out of bypass mode

Parameters

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

7.10.2.3 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

7.11 src/dev_codec_wm8731_frame.h File Reference

WM8731 Codec framework.

```
#include <stddef.h>
```

Classes

- struct [codec_frame_t](#)

Typedefs

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

7.11.1 Detailed Description

WM8731 Codec framework.

7.11.2 Typedef Documentation

7.11.2.1 sa_audio_callback

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

&

7.12 src/dev_flash_IS25LP064A.h File Reference

IS25LP08D Commands.

Macros

- `#define IS25LP064A_H`
- `#define IS25LP064A_FLASH_SIZE 0x800000`
- `#define IS25LP064A_SECTOR_SIZE 0x10000`
- `#define IS25LP064A_SUBSECTOR_SIZE 0x1000`
- `#define IS25LP064A_PAGE_SIZE 0x100`
- `#define IS25LP064A_DUMMY_CYCLES_READ_QUAD 8`
- `#define IS25LP064A_DUMMY_CYCLES_READ 8`
- `#define IS25LP064A_DUMMY_CYCLES_READ_DTR 6`
- `#define IS25LP064A_DUMMY_CYCLES_READ_QUAD_DTR 6`
- `#define IS25LP064A_DIE_ERASE_MAX_TIME 460000`
- `#define IS25LP064A_SECTOR_ERASE_MAX_TIME 1000`
- `#define IS25LP064A_SUBSECTOR_ERASE_MAX_TIME 400`
- `#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)

7.12.1 Detailed Description

IS25LP08D Commands.

7.12.2 Macro Definition Documentation

7.12.2.1 BLOCK_ERASE_32K_CMD

```
#define BLOCK_ERASE_32K_CMD 0x52
```

&

7.12.2.2 CLEAR_FLAG_STATUS_REG_CMD

```
#define CLEAR_FLAG_STATUS_REG_CMD 0x50
```

&

7.12.2.3 DIE_ERASE_CMD

```
#define DIE_ERASE_CMD 0xC4
```

&

7.12.2.4 DUAL_IN_FAST_PROG_CMD

```
#define DUAL_IN_FAST_PROG_CMD 0xA2
```

&

7.12.2.5 DUAL_INOUT_FAST_READ_4_BYTE_ADDR_CMD

```
#define DUAL_INOUT_FAST_READ_4_BYTE_ADDR_CMD 0xBC
```

&

7.12.2.6 DUAL_INOUT_FAST_READ_CMD

```
#define DUAL_INOUT_FAST_READ_CMD 0xBB
```

&

7.12.2.7 DUAL_INOUT_FAST_READ_DTR_CMD

```
#define DUAL_INOUT_FAST_READ_DTR_CMD 0xBD
```

&

7.12.2.8 DUAL_OUT_FAST_READ_4_BYTE_ADDR_CMD

```
#define DUAL_OUT_FAST_READ_4_BYTE_ADDR_CMD 0x3C
```

&

7.12.2.9 DUAL_OUT_FAST_READ_CMD

```
#define DUAL_OUT_FAST_READ_CMD 0x3B
```

&

7.12.2.10 DUAL_OUT_FAST_READ_DTR_CMD

```
#define DUAL_OUT_FAST_READ_DTR_CMD 0x3D
```

&

7.12.2.11 ENTER_4_BYTE_ADDR_MODE_CMD

```
#define ENTER_4_BYTE_ADDR_MODE_CMD 0xB7
```

4-byte Address Mode Operations

7.12.2.12 ENTER_QUAD_CMD

```
#define ENTER_QUAD_CMD 0x35
```

Quad Operations

7.12.2.13 EXIT_4_BYTE_ADDR_MODE_CMD

```
#define EXIT_4_BYTE_ADDR_MODE_CMD 0xE9
```

&

7.12.2.14 EXIT_QUAD_CMD

```
#define EXIT_QUAD_CMD 0xF5
```

&

7.12.2.15 EXT_DUAL_IN_FAST_PROG_CMD

```
#define EXT_DUAL_IN_FAST_PROG_CMD 0xD2
```

&

7.12.2.16 EXT_QUAD_IN_FAST_PROG_CMD

```
#define EXT_QUAD_IN_FAST_PROG_CMD 0x38
```

&

7.12.2.17 FAST_READ_4_BYTE_ADDR_CMD

```
#define FAST_READ_4_BYTE_ADDR_CMD 0x0C
```

&

7.12.2.18 FAST_READ_CMD

```
#define FAST_READ_CMD 0x0B
```

&

7.12.2.19 FAST_READ_DTR_CMD

```
#define FAST_READ_DTR_CMD 0x0D
```

&

7.12.2.20 IS25LP064A_DIE_ERASE_MAX_TIME

```
#define IS25LP064A_DIE_ERASE_MAX_TIME 460000
```

&

7.12.2.21 IS25LP064A_DUMMY_CYCLES_READ

```
#define IS25LP064A_DUMMY_CYCLES_READ 8
```

&

7.12.2.22 IS25LP064A_DUMMY_CYCLES_READ_DTR

```
#define IS25LP064A_DUMMY_CYCLES_READ_DTR 6
```

&

7.12.2.23 IS25LP064A_DUMMY_CYCLES_READ_QUAD

```
#define IS25LP064A_DUMMY_CYCLES_READ_QUAD 8
```

&

7.12.2.24 IS25LP064A_DUMMY_CYCLES_READ_QUAD_DTR

```
#define IS25LP064A_DUMMY_CYCLES_READ_QUAD_DTR 6
```

&

7.12.2.25 IS25LP064A_EAR_HIGHEST_SE

```
#define IS25LP064A_EAR_HIGHEST_SE ((uint8_t)0x03)
```

Select the Highest 128Mb segment

7.12.2.26 IS25LP064A_EAR_LOWEST_SEG

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

Select the Lowest 128Mb segment (default)

7.12.2.27 IS25LP064A_EAR_SECOND_SEG

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

Select the Second 128Mb segment

7.12.2.28 IS25LP064A_EAR_THIRD_SEG

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

Select the Third 128Mb segment

7.12.2.29 IS25LP064A_EVCR_DTRP

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

Double transfer rate protocol

7.12.2.30 IS25LP064A_EVCR_DUAL

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

Dual I/O protocol

7.12.2.31 IS25LP064A_EVCR_ODS

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

Output driver strength

7.12.2.32 IS25LP064A_EVCR_QUAD

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

Quad I/O protocol

7.12.2.33 IS25LP064A_EVCR_RH

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

Reset/hold

7.12.2.34 IS25LP064A_FLASH_SIZE

```
#define IS25LP064A_FLASH_SIZE 0x800000
```

2 * 8 MBits => 1 * 1MBytes => 1MBytes

7.12.2.35 IS25LP064A_FSR_ERERR

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

Erase error

7.12.2.36 IS25LP064A_FSR_ERSUS

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

Erase operation suspended

7.12.2.37 IS25LP064A_FSR_NBADDR

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

3-bytes or 4-bytes addressing

7.12.2.38 IS25LP064A_FSR_PGERR

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

Program error

7.12.2.39 IS25LP064A_FSR_PGSUS

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

Program operation suspended

7.12.2.40 IS25LP064A_FSR_PRERR

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

Protection error

7.12.2.41 IS25LP064A_FSR_READY

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

Ready or command in progress

7.12.2.42 IS25LP064A_H

```
#define IS25LP064A_H
```

&

7.12.2.43 IS25LP064A_NVCR_DTRP

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

Double transfer rate protocol

7.12.2.44 IS25LP064A_NVCR_DUAL

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

Dual I/O protocol

7.12.2.45 IS25LP064A_NVCR_NB_DUMMY

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

Number of dummy clock cycles

7.12.2.46 IS25LP064A_NVCR_NBADDR

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

3-bytes or 4-bytes addressing

7.12.2.47 IS25LP064A_NVCR_ODS

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

Output driver strength

7.12.2.48 IS25LP064A_NVCR_QUAB

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

Quad I/O protocol

7.12.2.49 IS25LP064A_NVCR_RH

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

Reset/hold

7.12.2.50 IS25LP064A_NVCR_SEGMENT

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

Upper or lower 128Mb segment selected by default

7.12.2.51 IS25LP064A_NVCR_XIP

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

XIP mode at power-on reset

7.12.2.52 IS25LP064A_PAGE_SIZE

```
#define IS25LP064A_PAGE_SIZE 0x100
```

2 * 262144 pages of 256 bytes

7.12.2.53 IS25LP064A_SECTOR_ERASE_MAX_TIME

```
#define IS25LP064A_SECTOR_ERASE_MAX_TIME 1000
```

&

7.12.2.54 IS25LP064A_SECTOR_SIZE

```
#define IS25LP064A_SECTOR_SIZE 0x10000
```

2 * 1024 sectors of 64KBytes

7.12.2.55 IS25LP064A_SR_QE

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

&

7.12.2.56 IS25LP064A_SR_SRWREN

```
#define IS25LP064A_SR_SRWREN ((uint8_t)0x80)
```

Status register write enable/disable

7.12.2.57 IS25LP064A_SR_WIP

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

IS25LP08D Registers.

Write in progress

7.12.2.58 IS25LP064A_SR_WREN

```
#define IS25LP064A_SR_WREN ((uint8_t)0x02)
```

Write enable latch

7.12.2.59 IS25LP064A_SUBSECTOR_ERASE_MAX_TIME

```
#define IS25LP064A_SUBSECTOR_ERASE_MAX_TIME 400
```

&

7.12.2.60 IS25LP064A_SUBSECTOR_SIZE

```
#define IS25LP064A_SUBSECTOR_SIZE 0x1000
```

2 * 16384 subsectors of 4kBytes

7.12.2.61 IS25LP064A_VCR_NB_DUMMY

```
#define IS25LP064A_VCR_NB_DUMMY ((uint8_t)0xF0)
```

Number of dummy clock cycles

7.12.2.62 IS25LP064A_VCR_WRAP

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

Wrap

7.12.2.63 IS25LP064A_VCR_XIP

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

XIP

7.12.2.64 MULTIPLE_IO_READ_ID_CMD

```
#define MULTIPLE_IO_READ_ID_CMD 0xAF
```

&

7.12.2.65 PAGE_PROG_4_BYTE_ADDR_CMD

```
#define PAGE_PROG_4_BYTE_ADDR_CMD 0x12
```

&

7.12.2.66 PAGE_PROG_CMD

```
#define PAGE_PROG_CMD 0x02
```

Program Operations

7.12.2.67 PROG_ERASE_RESUME_CMD

```
#define PROG_ERASE_RESUME_CMD 0x7A
```

&

7.12.2.68 PROG_ERASE_SUSPEND_CMD

```
#define PROG_ERASE_SUSPEND_CMD 0x75
```

&

7.12.2.69 PROG_OTP_ARRAY_CMD

```
#define PROG_OTP_ARRAY_CMD 0x42
```

&

7.12.2.70 QUAD_IN_FAST_PROG_4_BYTE_ADDR_CMD

```
#define QUAD_IN_FAST_PROG_4_BYTE_ADDR_CMD 0x34
```

&

7.12.2.71 QUAD_IN_FAST_PROG_CMD

```
#define QUAD_IN_FAST_PROG_CMD 0x32
```

&

7.12.2.72 QUAD_INOUT_FAST_READ_4_BYTE_ADDR_CMD

```
#define QUAD_INOUT_FAST_READ_4_BYTE_ADDR_CMD 0xEC
```

&

7.12.2.73 QUAD_INOUT_FAST_READ_CMD

```
#define QUAD_INOUT_FAST_READ_CMD 0xEB
```

&

7.12.2.74 QUAD_INOUT_FAST_READ_DTR_CMD

```
#define QUAD_INOUT_FAST_READ_DTR_CMD 0xED
```

&

7.12.2.75 QUAD_OUT_FAST_READ_4_BYTE_ADDR_CMD

```
#define QUAD_OUT_FAST_READ_4_BYTE_ADDR_CMD 0x6C
```

&

7.12.2.76 QUAD_OUT_FAST_READ_CMD

```
#define QUAD_OUT_FAST_READ_CMD 0x6B
```

&

7.12.2.77 QUAD_OUT_FAST_READ_DTR_CMD

```
#define QUAD_OUT_FAST_READ_DTR_CMD 0x0D
```

&

7.12.2.78 READ_4_BYTE_ADDR_CMD

```
#define READ_4_BYTE_ADDR_CMD 0x13
```

&

7.12.2.79 READ_CMD

```
#define READ_CMD 0x03
```

Read Operations

7.12.2.80 READ_ENHANCED_VOL_CFG_REG_CMD

```
#define READ_ENHANCED_VOL_CFG_REG_CMD 0x81
```

&

7.12.2.81 READ_EXT_ADDR_REG_CMD

```
#define READ_EXT_ADDR_REG_CMD 0xC8
```

&

7.12.2.82 READ_FLAG_STATUS_REG_CMD

```
#define READ_FLAG_STATUS_REG_CMD 0x70
```

&

7.12.2.83 READ_ID_CMD

```
#define READ_ID_CMD 0x9E
```

Identification Operations

7.12.2.84 READ_ID_CMD2

```
#define READ_ID_CMD2 0x9F
```

&

7.12.2.85 READ_LOCK_REG_CMD

```
#define READ_LOCK_REG_CMD 0xE8
```

&

7.12.2.86 READ_NONVOL_CFG_REG_CMD

```
#define READ_NONVOL_CFG_REG_CMD 0xB5
```

&

7.12.2.87 READ_OTP_ARRAY_CMD

```
#define READ_OTP_ARRAY_CMD 0x4B
```

One-Time Programmable Operations**7.12.2.88 READ_READ_PARAM_REG_CMD**

```
#define READ_READ_PARAM_REG_CMD 0x61
```

&

7.12.2.89 READ_SERIAL_FLASH_DISCO_PARAM_CMD

```
#define READ_SERIAL_FLASH_DISCO_PARAM_CMD 0x5A
```

&

7.12.2.90 READ_STATUS_REG_CMD

```
#define READ_STATUS_REG_CMD 0x05
```

Register Operations**7.12.2.91 RESET_ENABLE_CMD**

```
#define RESET_ENABLE_CMD 0x66
```

Reset Operations**7.12.2.92 RESET_MEMORY_CMD**

```
#define RESET_MEMORY_CMD 0x99
```

&

7.12.2.93 SECTOR_ERASE_4_BYTE_ADDR_CMD

```
#define SECTOR_ERASE_4_BYTE_ADDR_CMD 0xDC
```

&

7.12.2.94 SECTOR_ERASE_CMD

```
#define SECTOR_ERASE_CMD 0xD8
```

&

7.12.2.95 SUBSECTOR_ERASE_4_BYTE_ADDR_CMD

```
#define SUBSECTOR_ERASE_4_BYTE_ADDR_CMD 0x21
```

&

7.12.2.96 SUBSECTOR_ERASE_CMD

```
#define SUBSECTOR_ERASE_CMD 0xd7
```

Erase Operations**7.12.2.97 SUBSECTOR_ERASE_QPI_CMD**

```
#define SUBSECTOR_ERASE_QPI_CMD 0x20
```

&

7.12.2.98 WRITE_DISABLE_CMD

```
#define WRITE_DISABLE_CMD 0x04
```

&

7.12.2.99 WRITE_ENABLE_CMD

```
#define WRITE_ENABLE_CMD 0x06
```

Write Operations**7.12.2.100 WRITE_ENHANCED_VOL_CFG_REG_CMD**

```
#define WRITE_ENHANCED_VOL_CFG_REG_CMD 0x85
```

&

7.12.2.101 WRITE_EXT_ADDR_REG_CMD

```
#define WRITE_EXT_ADDR_REG_CMD 0xC5
```

&

7.12.2.102 WRITE_LOCK_REG_CMD

```
#define WRITE_LOCK_REG_CMD 0xE5
```

&

7.12.2.103 WRITE_NONVOL_CFG_REG_CMD

```
#define WRITE_NONVOL_CFG_REG_CMD 0xB1
```

&

7.12.2.104 WRITE_READ_PARAM_REG_CMD

```
#define WRITE_READ_PARAM_REG_CMD 0xC0
```

&

7.12.2.105 WRITE_STATUS_REG_CMD

```
#define WRITE_STATUS_REG_CMD 0x01
```

&

7.13 src/dev_flash_IS25LP080D.h File Reference

IS25LP08D Commands.

Macros

- `#define IS25LP080D_FLASH_SIZE 0x100000`
- `#define IS25LP080D_SECTOR_SIZE 0x10000`
- `#define IS25LP080D_SUBSECTOR_SIZE 0x1000`
- `#define IS25LP080D_PAGE_SIZE 0x100`
- `#define IS25LP080D_DUMMY_CYCLES_READ_QUAD 8`
- `#define IS25LP080D_DUMMY_CYCLES_READ 8`
- `#define IS25LP080D_DUMMY_CYCLES_READ_DTR 6`
- `#define IS25LP080D_DUMMY_CYCLES_READ_QUAD_DTR 6`
- `#define IS25LP080D_DIE_ERASE_MAX_TIME 460000`
- `#define IS25LP080D_SECTOR_ERASE_MAX_TIME 1000`
- `#define IS25LP080D_SUBSECTOR_ERASE_MAX_TIME 400`
- `#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)`
IS25LP080D 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)`

7.13.1 Detailed Description

IS25LP080D Commands.

7.13.2 Macro Definition Documentation

7.13.2.1 BLOCK_ERASE_32K_CMD

```
#define BLOCK_ERASE_32K_CMD 0x52
```

&

7.13.2.2 CLEAR_FLAG_STATUS_REG_CMD

```
#define CLEAR_FLAG_STATUS_REG_CMD 0x50
```

&

7.13.2.3 DIE_ERASE_CMD

```
#define DIE_ERASE_CMD 0xC4
```

&

7.13.2.4 DUAL_IN_FAST_PROG_CMD

```
#define DUAL_IN_FAST_PROG_CMD 0xA2
```

&

7.13.2.5 DUAL_INOUT_FAST_READ_4_BYTE_ADDR_CMD

```
#define DUAL_INOUT_FAST_READ_4_BYTE_ADDR_CMD 0xBC
```

&

7.13.2.6 DUAL_INOUT_FAST_READ_CMD

```
#define DUAL_INOUT_FAST_READ_CMD 0xBB
```

&

7.13.2.7 DUAL_INOUT_FAST_READ_DTR_CMD

```
#define DUAL_INOUT_FAST_READ_DTR_CMD 0xBD
```

&

7.13.2.8 DUAL_OUT_FAST_READ_4_BYTE_ADDR_CMD

```
#define DUAL_OUT_FAST_READ_4_BYTE_ADDR_CMD 0x3C
```

&

7.13.2.9 DUAL_OUT_FAST_READ_CMD

```
#define DUAL_OUT_FAST_READ_CMD 0x3B
```

&

7.13.2.10 DUAL_OUT_FAST_READ_DTR_CMD

```
#define DUAL_OUT_FAST_READ_DTR_CMD 0x3D
```

&

7.13.2.11 ENTER_4_BYTE_ADDR_MODE_CMD

```
#define ENTER_4_BYTE_ADDR_MODE_CMD 0xB7
```

4-byte Address Mode Operations

7.13.2.12 ENTER_QUAD_CMD

```
#define ENTER_QUAD_CMD 0x35
```

Quad Operations

7.13.2.13 EXIT_4_BYTE_ADDR_MODE_CMD

```
#define EXIT_4_BYTE_ADDR_MODE_CMD 0xE9
```

&

7.13.2.14 EXIT_QUAD_CMD

```
#define EXIT_QUAD_CMD 0xF5
```

&

7.13.2.15 EXT_DUAL_IN_FAST_PROG_CMD

```
#define EXT_DUAL_IN_FAST_PROG_CMD 0xD2
```

&

7.13.2.16 EXT_QUAD_IN_FAST_PROG_CMD

```
#define EXT_QUAD_IN_FAST_PROG_CMD 0x38
```

&

7.13.2.17 FAST_READ_4_BYTE_ADDR_CMD

```
#define FAST_READ_4_BYTE_ADDR_CMD 0x0C
```

&

7.13.2.18 FAST_READ_CMD

```
#define FAST_READ_CMD 0x0B
```

&

7.13.2.19 FAST_READ_DTR_CMD

```
#define FAST_READ_DTR_CMD 0x0D
```

&

7.13.2.20 IS25LP080D_DIE_ERASE_MAX_TIME

```
#define IS25LP080D_DIE_ERASE_MAX_TIME 460000
```

&

7.13.2.21 IS25LP080D_DUMMY_CYCLES_READ

```
#define IS25LP080D_DUMMY_CYCLES_READ 8
```

&

7.13.2.22 IS25LP080D_DUMMY_CYCLES_READ_DTR

```
#define IS25LP080D_DUMMY_CYCLES_READ_DTR 6
```

&

7.13.2.23 IS25LP080D_DUMMY_CYCLES_READ_QUAD

```
#define IS25LP080D_DUMMY_CYCLES_READ_QUAD 8
```

&

7.13.2.24 IS25LP080D_DUMMY_CYCLES_READ_QUAD_DTR

```
#define IS25LP080D_DUMMY_CYCLES_READ_QUAD_DTR 6
```

&

7.13.2.25 IS25LP080D_EAR_HIGHEST_SE

```
#define IS25LP080D_EAR_HIGHEST_SE ((uint8_t)0x03)
```

Select the Highest 128Mb segment

7.13.2.26 IS25LP080D_EAR_LOWEST_SEG

```
#define IS25LP080D_EAR_LOWEST_SEG ((uint8_t)0x00)
```

Select the Lowest 128Mb segment (default)

7.13.2.27 IS25LP080D_EAR_SECOND_SEG

```
#define IS25LP080D_EAR_SECOND_SEG ((uint8_t)0x01)
```

Select the Second 128Mb segment

7.13.2.28 IS25LP080D_EAR_THIRD_SEG

```
#define IS25LP080D_EAR_THIRD_SEG ((uint8_t)0x02)
```

Select the Third 128Mb segment

7.13.2.29 IS25LP080D_EVCR_DTRP

```
#define IS25LP080D_EVCR_DTRP ((uint8_t)0x20)
```

Double transfer rate protocol

7.13.2.30 IS25LP080D_EVCR_DUAL

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

Dual I/O protocol

7.13.2.31 IS25LP080D_EVCR_ODS

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

Output driver strength

7.13.2.32 IS25LP080D_EVCR_QUAD

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

Quad I/O protocol

7.13.2.33 IS25LP080D_EVCR_RH

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

Reset/hold

7.13.2.34 IS25LP080D_FLASH_SIZE

```
#define IS25LP080D_FLASH_SIZE 0x100000
```

2 * 8 MBits => 1 * 1MBytes => 1MBytes

7.13.2.35 IS25LP080D_FSR_ERERR

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

Erase error

7.13.2.36 IS25LP080D_FSR_ERSUS

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

Erase operation suspended

7.13.2.37 IS25LP080D_FSR_NBADDR

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

3-bytes or 4-bytes addressing

7.13.2.38 IS25LP080D_FSR_PGERR

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

Program error

7.13.2.39 IS25LP080D_FSR_PGSUS

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

Program operation suspended

7.13.2.40 IS25LP080D_FSR_PRERR

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

Protection error

7.13.2.41 IS25LP080D_FSR_READY

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

Ready or command in progress

7.13.2.42 IS25LP080D_NVCR_DTRP

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

Double transfer rate protocol

7.13.2.43 IS25LP080D_NVCR_DUAL

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

Dual I/O protocol

7.13.2.44 IS25LP080D_NVCR_NB_DUMMY

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

Number of dummy clock cycles

7.13.2.45 IS25LP080D_NVCR_NBADDR

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

3-bytes or 4-bytes addressing

7.13.2.46 IS25LP080D_NVCR_ODS

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

Output driver strength

7.13.2.47 IS25LP080D_NVCR_QUAB

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

Quad I/O protocol

7.13.2.48 IS25LP080D_NVCR_RH

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

Reset/hold

7.13.2.49 IS25LP080D_NVCR_SEGMENT

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

Upper or lower 128Mb segment selected by default

7.13.2.50 IS25LP080D_NVCR_XIP

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

XIP mode at power-on reset

7.13.2.51 IS25LP080D_PAGE_SIZE

```
#define IS25LP080D_PAGE_SIZE 0x100
```

2 * 262144 pages of 256 bytes

7.13.2.52 IS25LP080D_SECTOR_ERASE_MAX_TIME

```
#define IS25LP080D_SECTOR_ERASE_MAX_TIME 1000
```

&

7.13.2.53 IS25LP080D_SECTOR_SIZE

```
#define IS25LP080D_SECTOR_SIZE 0x10000
```

2 * 1024 sectors of 64KBytes

7.13.2.54 IS25LP080D_SR_QE

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

&

7.13.2.55 IS25LP080D_SR_SRWREN

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

Status register write enable/disable

7.13.2.56 IS25LP080D_SR_WIP

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

IS25LP08D Registers.

Status Register Write in progress

7.13.2.57 IS25LP080D_SR_WREN

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

Write enable latch

7.13.2.58 IS25LP080D_SUBSECTOR_ERASE_MAX_TIME

```
#define IS25LP080D_SUBSECTOR_ERASE_MAX_TIME 400
```

&

7.13.2.59 IS25LP080D_SUBSECTOR_SIZE

```
#define IS25LP080D_SUBSECTOR_SIZE 0x1000
```

2 * 16384 subsectors of 4kBytes

7.13.2.60 IS25LP080D_VCR_NB_DUMMY

```
#define IS25LP080D_VCR_NB_DUMMY ((uint8_t)0xF0)
```

Number of dummy clock cycles

7.13.2.61 IS25LP080D_VCR_WRAP

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

Wrap

7.13.2.62 IS25LP080D_VCR_XIP

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

XIP

7.13.2.63 MULTIPLE_IO_READ_ID_CMD

```
#define MULTIPLE_IO_READ_ID_CMD 0xAF
```

&

7.13.2.64 PAGE_PROG_4_BYTE_ADDR_CMD

```
#define PAGE_PROG_4_BYTE_ADDR_CMD 0x12
```

&

7.13.2.65 PAGE_PROG_CMD

```
#define PAGE_PROG_CMD 0x02
```

Program Operations

7.13.2.66 PROG_ERASE_RESUME_CMD

```
#define PROG_ERASE_RESUME_CMD 0x7A
```

&

7.13.2.67 PROG_ERASE_SUSPEND_CMD

```
#define PROG_ERASE_SUSPEND_CMD 0x75
```

&

7.13.2.68 PROG_OTP_ARRAY_CMD

```
#define PROG_OTP_ARRAY_CMD 0x42
```

&

7.13.2.69 QUAD_IN_FAST_PROG_4_BYTE_ADDR_CMD

```
#define QUAD_IN_FAST_PROG_4_BYTE_ADDR_CMD 0x34
```

&

7.13.2.70 QUAD_IN_FAST_PROG_CMD

```
#define QUAD_IN_FAST_PROG_CMD 0x32
```

&

7.13.2.71 QUAD_INOUT_FAST_READ_4_BYTE_ADDR_CMD

```
#define QUAD_INOUT_FAST_READ_4_BYTE_ADDR_CMD 0xEC
```

&

7.13.2.72 QUAD_INOUT_FAST_READ_CMD

```
#define QUAD_INOUT_FAST_READ_CMD 0xEB
```

&

7.13.2.73 QUAD_INOUT_FAST_READ_DTR_CMD

```
#define QUAD_INOUT_FAST_READ_DTR_CMD 0xED
```

&

7.13.2.74 QUAD_OUT_FAST_READ_4_BYTE_ADDR_CMD

```
#define QUAD_OUT_FAST_READ_4_BYTE_ADDR_CMD 0x6C
```

&

7.13.2.75 QUAD_OUT_FAST_READ_CMD

```
#define QUAD_OUT_FAST_READ_CMD 0x6B
```

&

7.13.2.76 QUAD_OUT_FAST_READ_DTR_CMD

```
#define QUAD_OUT_FAST_READ_DTR_CMD 0x0D
```

&

7.13.2.77 READ_4_BYTE_ADDR_CMD

```
#define READ_4_BYTE_ADDR_CMD 0x13
```

&

7.13.2.78 READ_CMD

```
#define READ_CMD 0x03
```

Read Operations**7.13.2.79 READ_ENHANCED_VOL_CFG_REG_CMD**

```
#define READ_ENHANCED_VOL_CFG_REG_CMD 0x81
```

&

7.13.2.80 READ_EXT_ADDR_REG_CMD

```
#define READ_EXT_ADDR_REG_CMD 0xC8
```

&

7.13.2.81 READ_FLAG_STATUS_REG_CMD

```
#define READ_FLAG_STATUS_REG_CMD 0x70
```

&

7.13.2.82 READ_ID_CMD

```
#define READ_ID_CMD 0x9E
```

Identification Operations

7.13.2.83 READ_ID_CMD2

```
#define READ_ID_CMD2 0x9F
```

&

7.13.2.84 READ_LOCK_REG_CMD

```
#define READ_LOCK_REG_CMD 0xE8
```

&

7.13.2.85 READ_NONVOL_CFG_REG_CMD

```
#define READ_NONVOL_CFG_REG_CMD 0xB5
```

&

7.13.2.86 READ_OTP_ARRAY_CMD

```
#define READ_OTP_ARRAY_CMD 0x4B
```

One-Time Programmable Operations

7.13.2.87 READ_READ_PARAM_REG_CMD

```
#define READ_READ_PARAM_REG_CMD 0x61
```

&

7.13.2.88 READ_SERIAL_FLASH_DISCO_PARAM_CMD

```
#define READ_SERIAL_FLASH_DISCO_PARAM_CMD 0x5A
```

&

7.13.2.89 READ_STATUS_REG_CMD

```
#define READ_STATUS_REG_CMD 0x05
```

Register Operations

7.13.2.90 RESET_ENABLE_CMD

```
#define RESET_ENABLE_CMD 0x66
```

Reset Operations

7.13.2.91 RESET_MEMORY_CMD

```
#define RESET_MEMORY_CMD 0x99
```

&

7.13.2.92 SECTOR_ERASE_4_BYTE_ADDR_CMD

```
#define SECTOR_ERASE_4_BYTE_ADDR_CMD 0xDC
```

&

7.13.2.93 SECTOR_ERASE_CMD

```
#define SECTOR_ERASE_CMD 0xD8
```

&

7.13.2.94 SUBSECTOR_ERASE_4_BYTE_ADDR_CMD

```
#define SUBSECTOR_ERASE_4_BYTE_ADDR_CMD 0x21
```

&

7.13.2.95 SUBSECTOR_ERASE_CMD

```
#define SUBSECTOR_ERASE_CMD 0xd7
```

Erase Operations

7.13.2.96 SUBSECTOR_ERASE_QPI_CMD

```
#define SUBSECTOR_ERASE_QPI_CMD 0x20
```

&

7.13.2.97 WRITE_DISABLE_CMD

```
#define WRITE_DISABLE_CMD 0x04
```

&

7.13.2.98 WRITE_ENABLE_CMD

```
#define WRITE_ENABLE_CMD 0x06
```

Write Operations

7.13.2.99 WRITE_ENHANCED_VOL_CFG_REG_CMD

```
#define WRITE_ENHANCED_VOL_CFG_REG_CMD 0x85
```

&

7.13.2.100 WRITE_EXT_ADDR_REG_CMD

```
#define WRITE_EXT_ADDR_REG_CMD 0xC5
```

&

7.13.2.101 WRITE_LOCK_REG_CMD

```
#define WRITE_LOCK_REG_CMD 0xE5
```

&

7.13.2.102 WRITE_NONVOL_CFG_REG_CMD

```
#define WRITE_NONVOL_CFG_REG_CMD 0xB1
```

&

7.13.2.103 WRITE_READ_PARAM_REG_CMD

```
#define WRITE_READ_PARAM_REG_CMD 0xC0
```

&

7.13.2.104 WRITE_STATUS_REG_CMD

```
#define WRITE_STATUS_REG_CMD 0x01
```

&

7.14 src/dev_leddriver.h File Reference

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

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

Classes

- struct [color](#)

Macros

- #define [SA_LED_DRIVER_H](#)
- #define [DSY_LED_DRIVER_MAX_DRIVERS](#) 8

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)

7.14.1 Detailed Description

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

7.14.2 Macro Definition Documentation

7.14.2.1 DSY_LED_DRIVER_MAX_DRIVERS

```
#define DSY_LED_DRIVER_MAX_DRIVERS 8
```

Maximum number of drivers

7.14.2.2 SA_LED_DRIVER_H

```
#define SA_LED_DRIVER_H
```

&

7.14.3 Enumeration Type Documentation

7.14.3.1 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	&

7.14.4 Function Documentation

7.14.4.1 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
-------------	--------------

7.14.4.2 dsy_led_driver_init()

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

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

Parameters

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

7.14.4.3 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

7.14.4.4 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.

7.15 src/dev_sdram.h File Reference

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

Classes

- struct [dsy_sdram_handle](#)

Macros

- #define [RAM_AS4C16M16SA_H](#)
- #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)

7.15.1 Macro Definition Documentation

7.15.1.1 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;

7.15.1.2 DSY_SDRAM_DATA

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

Usage:

E.g. int DSY_SDRAM_DATA initialized_var = 1;

7.15.1.3 RAM_AS4C16M16SA_H

```
#define RAM_AS4C16M16SA_H
```

SDRAM for 32MB AS4C16M16SA (and 64MB equivalent). Thanks to whoever this awesome person is: http://main.lv/writeup/stm32f4_sdram_configuration.md The Init function is basically a copy and paste. He has references to timing, etc. RAM is configured at 100MHz (fastest possible on the MCU). To use these the .sram_data/_bss sections must be configured correctly in the LINKER SCRIPT. using BSS is advised for most things, since the DATA section must also fit in flash in order to be initialized. Data section init not properly set up, as SDRAM is not initialized until after startup code.&

7.15.2 Enumeration Type Documentation

7.15.2.1 anonymous enum

anonymous enum

Enumerator

DSY_SDRAM_OK	&
DSY_SDRAM_ERR	&

7.15.2.2 dsy_sdram_pin

enum `dsy_sdram_pin`

This is PH5 on Daisy

Enumerator

DSY_SDRAM_PIN_SDNWE	&
DSY_SDRAM_PIN_LAST	&

7.15.2.3 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	&

7.15.3 Function Documentation

7.15.3.1 dsy_sdram_init()

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

Initializes the SDRAM peripheral

7.16 src/dev_sr_4021.h File Reference

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

```
#include "per_gpio.h"
```

Classes

- struct [dsy_sr_4021_handle](#)

Macros

- #define [DEV_SR_4021_H](#)
- #define [SR_4021_MAX_PARALLEL](#) 2
- #define [SR_4021_MAX_DAISSYCHAIN](#) 1

Enumerations

- enum {
 [DSY_SR_4021_PIN_CS](#), [DSY_SR_4021_PIN_CLK](#), [DSY_SR_4021_PIN_DATA](#), [DSY_SR_4021_PIN_D<←](#)
 [ATA2](#),
 [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)

7.16.1 Detailed Description

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

7.16.2 Macro Definition Documentation

7.16.2.1 DEV_SR_4021_H

```
#define DEV_SR_4021_H
```

&

7.16.2.2 SR_4021_MAX_DAISYCHAIN

```
#define SR_4021_MAX_DAISYCHAIN 1
```

fixed maximum for daisy chained use

7.16.2.3 SR_4021_MAX_PARALLEL

```
#define SR_4021_MAX_PARALLEL 2
```

Fixed maximums for parallel/daisy chained use
These could be expanded, but haven't been tested beyond this

7.16.3 Enumeration Type Documentation

7.16.3.1 anonymous enum

```
anonymous enum
```

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

Enumerator

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

7.16.4 Function Documentation

7.16.4.1 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
-----------	----------------------

7.16.4.2 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

7.16.4.3 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
------------	------------------

7.17 src/dev_sr_595.h File Reference

```
#include "daisy_core.h"
#include "per_gpio.h"
```

Classes

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

Variables

- const size_t [kMaxSr595DaisyChain](#) = 16

7.17.1 Variable Documentation

7.17.1.1 kMaxSr595DaisyChain

```
const size_t kMaxSr595DaisyChain = 16
```

Maximum Number of chained devices Connect device's QH' pin to the next chips serial input

7.18 src/fatfs.h File Reference

fatfs support.

```
#include "ff.h"
#include "ff_gen_drv.h"
#include "util_sd_diskio.h"
```

Macros

- #define [__fatfs_H](#)

Functions

- void [dsy_fatfs_init](#) (void)

Variables

- uint8_t [retSD](#)
- char [SDPath](#) [4]
- FATFS [SDFatFS](#)
- FIL [SDFile](#)

7.18.1 Detailed Description

fatfs support.

7.18.2 Macro Definition Documentation

7.18.2.1 __fatfs_H

```
#define __fatfs_H
```

&

7.18.3 Function Documentation

7.18.3.1 dsy_fatfs_init()

```
void dsy_fatfs_init (  
    void )
```

&

7.18.4 Variable Documentation

7.18.4.1 retSD

```
uint8_t retSD
```

&

7.18.4.2 SDFatFS

```
FATFS SDFatFS
```

&

7.18.4.3 SDFile

```
FIL SDFile
```

&

7.18.4.4 SDPath

```
char SDPath[4]
```

&

7.19 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`

7.19.1 Detailed Description

Further fatfs support.

7.19.2 Macro Definition Documentation

7.19.2.1 _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)

7.19.2.2 _FFCONF

```
#define _FFCONF 68300
```

FatFs - Generic FAT file system module R0.12c (C)ChaN, 2017

Attention

© Copyright (c) 2019 STMicroelectronics. All rights reserved.

This software component is licensed by ST under Ultimate Liberty license SLA0044, the "License"; You may not use this file except in compliance with the License. You may obtain a copy of the License at: www.st.com/SLA0044↵
Revision ID

7.19.2.3 _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.

7.19.2.4 _FS_LOCK

```
#define _FS_LOCK 2
```

0:Disable or >=1:Enable The option _FS_LOCK switches file lock function to control duplicated file open / and illegal operation to open objects. This option must be 0 when _FS_READONLY / is 1. // 0: Disable file lock function. To avoid volume corruption, application program / should avoid illegal open, remove and rename to the open objects. / >0: Enable file lock function. The value defines how many files/sub-directories / can be opened simultaneously under file lock control. Note that the file / lock control is independent of re-entrancy.

7.19.2.5 _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.

7.19.2.6 _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.

7.19.2.7 _FS_NORTC

```
#define _FS_NORTC 0
```

&

7.19.2.8 _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.

7.19.2.9 _FS_REENTRANT

```
#define _FS_REENTRANT 0
```

0:Disable or 1:Enable

7.19.2.10 _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.

7.19.2.11 _FS_TIMEOUT

```
#define _FS_TIMEOUT 1000
```

Timeout period in unit of time ticks

7.19.2.12 _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.

7.19.2.13 _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.

7.19.2.14 _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.

7.19.2.15 _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.

7.19.2.16 _MIN_SS

```
#define _MIN_SS 512
```

512, 1024, 2048 or 4096

7.19.2.17 `_MULTI_PARTITION`

```
#define _MULTI_PARTITION 0
```

0:Single partition, 1:Multiple partition This option switches support of multi-partition on a physical drive. / By default (0), each logical drive number is bound to the same physical drive / number and only an FAT volume found on the physical drive will be mounted. / When multi-partition is enabled (1), each logical drive number can be bound to / arbitrary physical drive and partition listed in the VolToPart[]. Also `f_fdisk()` / `funciton` will be available.

7.19.2.18 `_NORTC_MDAY`

```
#define _NORTC_MDAY 4
```

&

7.19.2.19 `_NORTC_MON`

```
#define _NORTC_MON 6
```

&

7.19.2.20 `_NORTC_YEAR`

```
#define _NORTC_YEAR 2015
```

The option `_FS_NORTC` switches timestamp function. If the system does not have / any RTC function or valid timestamp is not needed, set `_FS_NORTC = 1` to disable / the timestamp function. All objects modified by FatFs will have a fixed timestamp / defined by `_NORTC_MON`, `_NORTC_MDAY` and `_NORTC_YEAR` in local time. / To enable timestamp function (`_FS_NORTC = 0`), `get_fattime()` function need to be / added to the project to get current time form real-time clock. `_NORTC_MON`, `_NORTC_MDAY` and `_NORTC_YEAR` have no effect. / These options have no effect at read-only configuration (`_FS_READONLY = 1`).

7.19.2.21 `_STR_VOLUME_ID`

```
#define _STR_VOLUME_ID 0
```

0:Use only 0-9 for drive ID, 1:Use strings for drive ID

7.19.2.22 `_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`.

7.19.2.23 `_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.

7.19.2.24 `_USE_CHMOD`

```
#define _USE_CHMOD 0
```

This option switches attribute manipulation functions, `f_chmod()` and `f_utime()`. / (0:Disable or 1:Enable) Also `_FS_READONLY` needs to be 0 to enable this option.

7.19.2.25 `_USE_EXPAND`

```
#define _USE_EXPAND 0
```

This option switches `f_expand` function. (0:Disable or 1:Enable)

7.19.2.26 `_USE_FASTSEEK`

```
#define _USE_FASTSEEK 1
```

This option switches fast seek feature. (0:Disable or 1:Enable)

7.19.2.27 `_USE_FIND`

```
#define _USE_FIND 0
```

This option switches filtered directory read functions, `f_findfirst()` and / `f_findnext()`. (0:Disable, 1:Enable 2:Enable with matching `altname[]` too)

7.19.2.28 `_USE_FORWARD`

```
#define _USE_FORWARD 0
```

This option switches `f_forward()` function. (0:Disable or 1:Enable)

7.19.2.29 _USE_LABEL

```
#define _USE_LABEL 0
```

This option switches volume label functions, `f_getlabel()` and `f_setlabel()`. / (0:Disable or 1:Enable)

7.19.2.30 _USE_LFN

```
#define _USE_LFN 1
```

0 to 3

7.19.2.31 _USE_MKFS

```
#define _USE_MKFS 1
```

This option switches `f_mkfs()` function. (0:Disable or 1:Enable)

7.19.2.32 _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.

7.19.2.33 _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.

7.19.2.34 _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.

7.19.2.35 `_VOLUMES`

```
#define _VOLUMES 1
```

Number of volumes (logical drives) to be used.

7.19.2.36 `ff_free`

```
#define ff_free free
```

define the `ff_malloc` `ff_free` macros as standard `malloc` `free`

7.19.2.37 `ff_malloc`

```
#define ff_malloc malloc
```

define the `ff_malloc` `ff_free` macros as standard `malloc` `free`

7.20 `src/hid_gatein.h` File Reference

```
#include "per_gpio.h"
```

Classes

- class [daisy::GateIn](#)
Generic Class for handling gate inputs through GPIO.

7.21 `src/hid_wavplayer.h` File Reference

```
#include "daisy_core.h"  
#include "util_wav_format.h"
```

Classes

- struct [daisy::WavFileInfo](#)
- class [daisy::WavPlayer](#)

Macros

- `#define` [DSY_WAVPLAYER_H](#)
- `#define` [WAV_FILENAME_MAX](#) 256

7.21.1 Macro Definition Documentation

7.21.1.1 DSY_WAVPLAYER_H

```
#define DSY_WAVPLAYER_H
```

Macro

7.21.1.2 WAV_FILENAME_MAX

```
#define WAV_FILENAME_MAX 256
```

Maximum LFN (set to same in FatFs ([ffconf.h](#)))

7.22 src/per_adc.h File Reference

```
#include <stdint.h>
#include <stdlib.h>
#include "daisy_core.h"
#include "per_gpio.h"
```

Classes

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

Macros

- #define [DSY_ADC_H](#)
- #define [DSY_ADC_MAX_CHANNELS](#) 14

7.22.1 Macro Definition Documentation

7.22.1.1 DSY_ADC_H

```
#define DSY_ADC_H
```

Macro

7.22.1.2 DSY_ADC_MAX_CHANNELS

```
#define DSY_ADC_MAX_CHANNELS 14
```

Maximum number of ADC channels

7.23 src/per_dac.h File Reference

```
#include "daisy_core.h"
```

Classes

- struct [dsy_dac_handle](#)

Enumerations

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

Functions

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

7.23.1 Enumeration Type Documentation

7.23.1.1 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	&

7.23.1.2 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	&

7.23.1.3 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

7.23.2 Function Documentation

7.23.2.1 dsy_dac_init()

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

Initializes the specified channel(s) of the DAC

Parameters

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

7.23.2.2 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
----------------	------------------

7.23.2.3 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

7.24 src/per_gpio.h File Reference

```
#include "daisy_core.h"
```

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)

7.24.1 Detailed Description

General Purpose IO driver

7.24.2 Enumeration Type Documentation

7.24.2.1 [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	&

7.24.2.2 [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	&

7.24.3 Function Documentation

7.24.3.1 dsy_gpio_deinit()

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

Deinitializes the gpio pin

Parameters

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

7.24.3.2 dsy_gpio_init()

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

Initializes the gpio with the settings configured.

Parameters

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

7.24.3.3 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

7.24.3.4 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
-----------	-------------

7.24.3.5 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

7.25 src/per_i2c.h File Reference

```
#include "daisy_core.h"
```

Classes

- struct [dsy_i2c_handle](#)

Macros

- #define [DSY_I2C_H](#)

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](#) }

Functions

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

7.25.1 Macro Definition Documentation

7.25.1.1 DSY_I2C_H

```
#define DSY_I2C_H
```

Macro

7.25.2 Enumeration Type Documentation

7.25.2.1 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	&

7.25.2.2 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	&

7.25.2.3 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	&

7.25.3 Function Documentation

7.25.3.1 dsy_i2c_init()

```
void dsy_i2c_init (
    dsy\_i2c\_handle * dsy_hi2c )
```

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

Parameters

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

7.26 src/per_qspi.h File Reference

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

Classes

- struct [dsy_qspi_handle](#)

Macros

- `#define DSY_QSPI`
- `#define DSY_MEMORY_OK ((uint32_t)0x00)`
- `#define DSY_MEMORY_ERROR ((uint32_t)0x01)`
- `#define DSY_QSPI_TEXT`
- `#define DSY_QSPI_DATA`
- `#define DSY_QSPI_BSS`

Enumerations

- `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_Q↵`
`SPI_DEVICE_LAST }`

Functions

- `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)`

7.26.1 Macro Definition Documentation

7.26.1.1 DSY_MEMORY_ERROR

```
#define DSY_MEMORY_ERROR ((uint32_t)0x01)
```

&

7.26.1.2 DSY_MEMORY_OK

```
#define DSY_MEMORY_OK ((uint32_t)0x00)
```

&

7.26.1.3 DSY_QSPI

```
#define DSY_QSPI
```

Macro

7.26.1.4 DSY_QSPI_BSS

```
#define DSY_QSPI_BSS
```

Value:

```
__attribute__((section( \
    ".qspiflash_bss")))
```

used for reading memory in memory_mapped mode.

7.26.1.5 DSY_QSPI_DATA

```
#define DSY_QSPI_DATA
```

Value:

```
__attribute__((section( \
    ".qspiflash_data")))
```

used for reading memory in memory_mapped mode.

7.26.1.6 DSY_QSPI_TEXT

```
#define DSY_QSPI_TEXT
```

Value:

```
__attribute__((section( \
    ".qspiflash_text")))
```

used for reading memory in memory_mapped mode.

7.26.2 Enumeration Type Documentation

7.26.2.1 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	&

7.26.2.2 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	&

7.26.2.3 dsy_qspi_pin

enum `dsy_qspi_pin`

Driver for QSPI peripheral to interface with external flash memory.

Currently supported QSPI Devices:

IS25LP080DList 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	&

7.26.3 Function Documentation

7.26.3.1 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

7.26.3.2 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

7.26.3.3 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

7.26.3.4 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

7.26.3.5 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

7.26.3.6 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

7.27 src/per_sai.h File Reference

```
#include "daisy_core.h"
```

Classes

- struct [dsy_sai_handle](#)

Enumerations

- 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 }

Functions

- 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)

7.27.1 Enumeration Type Documentation

7.27.1.1 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	&

7.27.1.2 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	&

7.27.1.3 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	&

7.27.1.4 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	&

7.27.1.5 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	&

7.27.1.6 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	&

7.27.1.7 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	&

7.27.1.8 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	&

7.27.2 Function Documentation

7.27.2.1 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>	&

7.27.2.2 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>	&
-------------	---

7.28 src/per_sdmmc.h File Reference

```
#include <stdint.h>
```

Classes

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

Macros

- #define [DSY_SDMMC_H](#)
- #define [DSY_SD_OK](#) 0
- #define [DSY_SD_ERROR](#) 1

Enumerations

- enum [daisy::SdmmcMode](#) { [daisy::SDMMC_MODE_FATFS](#) }
- enum [daisy::SdmmcBitWidth](#) { [daisy::SDMMC_BITS_1](#), [daisy::SDMMC_BITS_4](#) }
- enum [daisy::SdmmcSpeed](#) { [daisy::SDMMC_SPEED_400KHZ](#), [daisy::SDMMC_SPEED_12MHZ](#) }

7.28.1 Macro Definition Documentation

7.28.1.1 DSY_SD_ERROR

```
#define DSY_SD_ERROR 1
```

ERROR return

7.28.1.2 DSY_SD_OK

```
#define DSY_SD_OK 0
```

OK return

7.28.1.3 DSY_SDMMC_H

```
#define DSY_SDMMC_H
```

macro

7.28.2 Enumeration Type Documentation

7.28.2.1 SdmmcBitWidth

```
enum daisy::SdmmcBitWidth
```

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

Enumerator

SDMMC_BITS↔ _1	&
SDMMC_BITS↔ _4	&

7.28.2.2 SdmmcMode

```
enum daisy::SdmmcMode
```

Operating Mode. Currently only FatFS is supported.

Enumerator

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

7.28.2.3 SdmmcSpeed

```
enum daisy::SdmmcSpeed
```

Sets the desired clock speed of the SD card bus.
Initialization is always done at or below 400kHz, and then the user speed is set.

Enumerator

SDMMC_SPEED_400KHZ	&
SDMMC_SPEED_12MHZ	&

7.29 src/per_spi.h File Reference

```
#include "daisy_core.h"
```

Classes

- class [daisy::SpiHandle](#)

Enumerations

- 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](#) }

7.29.1 Enumeration Type Documentation

7.29.1.1 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

7.29.1.2 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

7.30 src/per_tim.h File Reference

```
#include <stdint.h>
```

Functions

- 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)

7.30.1 Function Documentation

7.30.1.1 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
------------	------------------

7.30.1.2 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
------------	-----------------

7.30.1.3 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
------------	------------------

7.30.1.4 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.

7.30.1.5 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

7.30.1.6 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.

7.30.1.7 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.

7.30.1.8 dsy_tim_start()

```
void dsy_tim_start ( )
```

Starts the timer ticking.

7.31 src/per_uart.h File Reference

```
#include "daisy_core.h"
```

Classes

- class [daisy::UartHandler](#)

Macros

- #define [DSY_UART_H](#)

Variables

- const size_t [daisy::kUartMaxBufferSize](#) = 32

7.31.1 Macro Definition Documentation

7.31.1.1 DSY_UART_H

```
#define DSY_UART_H
```

macro

7.31.2 Variable Documentation

7.31.2.1 kUartMaxBufferSize

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

Maximum Queue buffer size

7.32 src/sys_dma.h File Reference

Functions

- void [dsy_dma_init](#) (void)

7.32.1 Function Documentation

7.32.1.1 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)

7.33 src/sys_system.h File Reference

```
#include <stdint.h>
```

Functions

- 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)

7.33.1 Detailed Description

Low level System Configuration

7.33.2 Function Documentation

7.33.2.1 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
-----------------	---------------------

7.33.2.2 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()

7.33.2.3 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.

7.33.2.4 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
-------------	--------------------

7.33.2.5 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.

7.34 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_ItfTypeDef [USBDC_Interface_fops_FS](#)
- USBDCDC_ItfTypeDef [USBDC_Interface_fops_HS](#)

7.34.1 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

7.35 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\(...\)](#)

7.35.1 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

7.36 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](#)

7.36.1 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

7.37 src/util_bsp_sd_diskio.h File Reference

```
#include <stdint.h>
```


Classes

- struct [DSY_SD_CardInfoTypeDef](#)

Macros

- `#define DSY_BSP_SD_DISKIO_H`
- `#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)`

Functions

- `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)`

7.37.1 Macro Definition Documentation

7.37.1.1 BSP_SD_CardInfo

```
#define BSP_SD_CardInfo DSY_SD_CardInfoTypeDef
```

&

7.37.1.2 DSY_BSP_SD_DISKIO_H

```
#define DSY_BSP_SD_DISKIO_H
```

&

7.37.1.3 MSD_ERROR

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

&

7.37.1.4 MSD_ERROR_SD_NOT_PRESENT

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

&

7.37.1.5 MSD_OK

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

&

7.37.1.6 SD_DATATIMEOUT

```
#define SD_DATATIMEOUT ((uint32_t)100000000)
```

&

7.37.1.7 SD_NOT_PRESENT

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

&

7.37.1.8 SD_PRESENT

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

&

7.37.1.9 SD_TRANSFER_BUSY

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

&

7.37.1.10 SD_TRANSFER_OK

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

&

7.37.2 Function Documentation

7.37.2.1 BSP_SD_AbortCallback()

```
void BSP_SD_AbortCallback (
    void )
```

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

Abort the callback

7.37.2.2 BSP_SD_Erase()

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

Erase a section of memory

Parameters

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

Returns

card state, ERROR, etc.

7.37.2.3 BSP_SD_GetCardInfo()

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

Parameters

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

Parameters

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

7.37.2.4 BSP_SD_GetCardState()

```
uint8_t BSP_SD_GetCardState (
    void )
```

Returns

card state, ERROR, etc.

7.37.2.5 BSP_SD_Init()

```
uint8_t BSP_SD_Init (
    void )
```

Returns

card state, ERROR, etc.

7.37.2.6 BSP_SD_IsDetected()

```
uint8_t BSP_SD_IsDetected (
    void )
```

Returns

Is card detected

7.37.2.7 BSP_SD_ITConfig()

```
uint8_t BSP_SD_ITConfig (
    void )
```

Returns

card state, ERROR, etc.

7.37.2.8 BSP_SD_ReadBlocks()

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

Parameters

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

Returns

OK ERROR, etc.

7.37.2.9 BSP_SD_ReadBlocks_DMA()

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

No timeout

Parameters

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

Returns

card state, ERROR, etc.

7.37.2.10 BSP_SD_ReadCpltCallback()

```
void BSP_SD_ReadCpltCallback (
    void )
```

Write complete callback

7.37.2.11 BSP_SD_WriteBlocks()

```
uint8_t BSP_SD_WriteBlocks (
    uint32_t * pData,
    uint32_t WriteAddr,
    uint32_t NumOfBlocks,
    uint32_t Timeout )
```

Parameters

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

Returns

card state, ERROR, etc.

7.37.2.12 BSP_SD_WriteBlocks_DMA()

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

No timeout

Parameters

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

Returns

card state, ERROR, etc.

7.37.2.13 BSP_SD_WriteCpltCallback()

```
void BSP_SD_WriteCpltCallback (
    void )
```

Read complete callback

7.38 src/util_color.h File Reference

```
#include <stdint.h>
```

Classes

- class [daisy::Color](#)

Macros

- `#define` [DSY_COLOR_H](#)

7.38.1 Macro Definition Documentation

7.38.1.1 DSY_COLOR_H

```
#define DSY_COLOR_H
```

I'd like for it to be easy and not `Color::PresetColor::Foo`- There's no way to change a color once its been created (without unintuitively reiniting it).

7.39 src/util_hal_map.h File Reference

```
#include "stm32h7xx_hal.h"
#include "daisy_core.h"
#include "per_i2c.h"
```

Functions

- `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)

Variables

- `I2C_HandleTypeDef` [hi2c1](#)
- `I2C_HandleTypeDef` [hi2c2](#)
- `I2C_HandleTypeDef` [hi2c3](#)
- `I2C_HandleTypeDef` [hi2c4](#)

7.39.1 Function Documentation

7.39.1.1 dsy_hal_map_get_i2c()

```
I2C_HandleTypeDef* dsy_hal_map_get_i2c (
    dsy\_i2c\_handle * p )
```

Parameters

<i>*p</i>	dsy_i2c_handle to get
-----------	---------------------------------------

Returns

The I2C_HandleTypeDef for the given **p*

7.39.1.2 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.

7.39.1.3 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.

7.39.2 Variable Documentation**7.39.2.1 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...

7.39.2.2 hi2c2

```
I2C_HandleTypeDef hi2c2
```

externs of HAL handles...

7.39.2.3 hi2c3

```
I2C_HandleTypeDef hi2c3
```

externs of HAL handles...

7.39.2.4 hi2c4

```
I2C_HandleTypeDef hi2c4
```

externs of HAL handles...

7.40 src/util_ringbuffer.h File Reference

```
#include <algorithm>
```

Classes

- class [daisy::RingBuffer< T, size >](#)
- class [daisy::RingBuffer< T, 0 >](#)

7.41 src/util_unique_id.h File Reference

```
#include "daisy_core.h"
```

Functions

- void [dsy_get_unique_id](#) (uint32_t *w0, uint32_t *w1, uint32_t *w2)

7.41.1 Function Documentation

7.41.1.1 dsy_get_unique_id()

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

Returns 96-bit Unique ID of the MCU

Author

shensley

Date

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

Parameters

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

7.42 src/util_wav_format.h File Reference

```
#include <stdint.h>
```

Classes

- struct [WAV_FormatTypeDef](#)

Index

- [_CODE_PAGE](#)
 - [ffconf.h, 204](#)
- [_FFCONF](#)
 - [ffconf.h, 204](#)
- [_FS_EXFAT](#)
 - [ffconf.h, 204](#)
- [_FS_LOCK](#)
 - [ffconf.h, 204](#)
- [_FS_MINIMIZE](#)
 - [ffconf.h, 204](#)
- [_FS_NOFSINFO](#)
 - [ffconf.h, 205](#)
- [_FS_NORTC](#)
 - [ffconf.h, 205](#)
- [_FS_READONLY](#)
 - [ffconf.h, 205](#)
- [_FS_REENTRANT](#)
 - [ffconf.h, 205](#)
- [_FS_RPATH](#)
 - [ffconf.h, 205](#)
- [_FS_TIMEOUT](#)
 - [ffconf.h, 205](#)
- [_FS_TINY](#)
 - [ffconf.h, 206](#)
- [_LFN_UNICODE](#)
 - [ffconf.h, 206](#)
- [_MAX_LFN](#)
 - [ffconf.h, 206](#)
- [_MAX_SS](#)
 - [ffconf.h, 206](#)
- [_MIN_SS](#)
 - [ffconf.h, 206](#)
- [_MULTI_PARTITION](#)
 - [ffconf.h, 206](#)
- [_NORTC_MDAY](#)
 - [ffconf.h, 207](#)
- [_NORTC_MON](#)
 - [ffconf.h, 207](#)
- [_NORTC_YEAR](#)
 - [ffconf.h, 207](#)
- [_STRF_ENCODE](#)
 - [ffconf.h, 207](#)
- [_STR_VOLUME_ID](#)
 - [ffconf.h, 207](#)
- [_SYNC_t](#)
 - [ffconf.h, 207](#)
- [_USE_CHMOD](#)
 - [ffconf.h, 208](#)
- [_USE_EXPAND](#)
 - [ffconf.h, 208](#)
- [_USE_FASTSEEK](#)
 - [ffconf.h, 208](#)
- [_USE_FIND](#)
 - [ffconf.h, 208](#)
- [_USE_FORWARD](#)
 - [ffconf.h, 208](#)
- [_USE_LABEL](#)
 - [ffconf.h, 208](#)
- [_USE_LFN](#)
 - [ffconf.h, 209](#)
- [_USE_MKFS](#)
 - [ffconf.h, 209](#)
- [_USE_STRFUNC](#)
 - [ffconf.h, 209](#)
- [_USE_TRIM](#)
 - [ffconf.h, 209](#)
- [_VOLUMES](#)
 - [ffconf.h, 209](#)
- [_VOLUME_STRS](#)
 - [ffconf.h, 209](#)
- [__fatfs_H](#)
 - [fatfs.h, 201](#)
- [~AnalogControl](#)
 - [CONTROLS, 21](#)
- [~DaisyPatch](#)
 - [daisy::DaisyPatch, 78](#)
- [~DaisyPetal](#)
 - [daisy::DaisyPetal, 84](#)
- [~GateIn](#)
 - [daisy::GateIn, 112](#)
- [~Parameter](#)
 - [CONTROLS, 21](#)
- [a_direction](#)
 - [dsy_sai_handle, 104](#)
- [AUDIO, 11](#)
 - [dsy_audio_callback, 11](#)
 - [dsy_audio_enter_bypass, 12](#)
 - [dsy_audio_exit_bypass, 12](#)
 - [dsy_audio_init, 12](#)
 - [dsy_audio_mc_callback, 11](#)
 - [dsy_audio_passthru, 12](#)
 - [dsy_audio_set_blocksize, 12](#)
 - [dsy_audio_set_callback, 13](#)
 - [dsy_audio_set_mc_callback, 13](#)
 - [dsy_audio_silence, 13](#)
 - [dsy_audio_start, 13](#)
 - [dsy_audio_stop, 13](#)
- [adc](#)

- daisy::DaisySeed, 97
- AnalogControl
 - CONTROLS, 16
- AsControlChange
 - EXTERNAL, 31
- AsNoteOn
 - EXTERNAL, 31
- audio_handle
 - daisy::DaisySeed, 97
- AudioBlockSize
 - daisy::DaisyPatch, 78
 - daisy::DaisyPetal, 85
 - daisy::DaisyPod, 91
- AudioCallbackRate
 - daisy::DaisyPatch, 78
 - daisy::DaisyPetal, 85
 - daisy::DaisyPod, 91
- AudioFormat
 - WAV_FormatTypeDef, 137
- AudioSampleRate
 - daisy::DaisyPatch, 78
 - daisy::DaisyPetal, 85
 - daisy::DaisyPod, 91
 - daisy::DaisySeed, 95
- b_direction
 - dsy_sai_handle, 104
- BLOCK_ERASE_32K_CMD
 - dev_flash_IS25LP064A.h, 163
 - dev_flash_IS25LP080D.h, 178
- BSP_SD_AbortCallback
 - util_bsp_sd_diskio.h, 243
- BSP_SD_CardInfo
 - util_bsp_sd_diskio.h, 241
- BSP_SD_Erase
 - util_bsp_sd_diskio.h, 243
- BSP_SD_GetCardInfo
 - util_bsp_sd_diskio.h, 243
- BSP_SD_GetCardState
 - util_bsp_sd_diskio.h, 243
- BSP_SD_ITConfig
 - util_bsp_sd_diskio.h, 244
- BSP_SD_Init
 - util_bsp_sd_diskio.h, 244
- BSP_SD_IsDetected
 - util_bsp_sd_diskio.h, 244
- BSP_SD_ReadBlocks
 - util_bsp_sd_diskio.h, 244
- BSP_SD_ReadBlocks_DMA
 - util_bsp_sd_diskio.h, 245
- BSP_SD_ReadCpltCallback
 - util_bsp_sd_diskio.h, 245
- BSP_SD_WriteBlocks
 - util_bsp_sd_diskio.h, 245
- BSP_SD_WriteBlocks_DMA
 - util_bsp_sd_diskio.h, 246
- BSP_SD_WriteCpltCallback
 - util_bsp_sd_diskio.h, 246
- BitPerSample
 - WAV_FormatTypeDef, 137
- bitdepth
 - daisy::SdmmcHandlerInit, 125
 - dsy_dac_handle, 100
 - dsy_sai_handle, 104
- block_size
 - dsy_audio_handle, 99
- BlockAlign
 - WAV_FormatTypeDef, 137
- BlockNbr
 - DSY_SD_CardInfoTypeDef, 106
- BlockSize
 - DSY_SD_CardInfoTypeDef, 106
- BlockingTransmit
 - daisy::SpiHandle, 127
- Blue
 - daisy::Color, 73
- blue
 - color, 71
- button1
 - daisy::DaisyPod, 93
- button2
 - daisy::DaisyPod, 93
- buttons
 - daisy::DaisyPod, 94
- ByteRate
 - WAV_FormatTypeDef, 137
- CDC_ReceiveCallback
 - USBD_CDC_IF_Exported_Types, 41
- CDC_Set_Rx_Callback_FS
 - USBD_CDC_IF_Exported_FunctionsPrototype, 44
- CDC_Transmit_FS
 - USBD_CDC_IF_Exported_FunctionsPrototype, 44
- CDC_Transmit_HS
 - USBD_CDC_IF_Exported_FunctionsPrototype, 44
- CLEAR_FLAG_STATUS_REG_CMD
 - dev_flash_IS25LP064A.h, 163
 - dev_flash_IS25LP080D.h, 178
- CONTROLS, 14
 - ~AnalogControl, 21
 - ~Parameter, 21
 - AnalogControl, 16
 - Curve, 15
 - Debounce, 16
 - FallingEdge, 16
 - Increment, 17
 - Init, 17, 18
 - InitBipolarCv, 19
 - Parameter, 19
 - Polarity, 15
 - Pressed, 19
 - Process, 19, 20
 - Pull, 15
 - RisingEdge, 20
 - TimeHeldMs, 20
 - Type, 16
 - Value, 20, 21
- CV1_ADC_PIN

- daisy_field.h, [150](#)
- CV2_ADC_PIN
 - daisy_field.h, [150](#)
- CV3_ADC_PIN
 - daisy_field.h, [150](#)
- CV4_ADC_PIN
 - daisy_field.h, [150](#)
- capacity
 - daisy::RingBuffer, [118](#)
 - daisy::RingBuffer< T, 0 >, [121](#)
- CardSpeed
 - DSY_SD_CardInfoTypeDef, [106](#)
- CardType
 - DSY_SD_CardInfoTypeDef, [106](#)
- CardVersion
 - DSY_SD_CardInfoTypeDef, [107](#)
- ChangeAudioCallback
 - daisy::DaisyPatch, [78](#)
 - daisy::DaisyPetal, [85](#)
 - daisy::DaisyPod, [91](#)
- channel
 - EXTERNAL, [32](#), [33](#)
- CheckError
 - daisy::UartHandler, [129](#)
- ChunkId
 - WAV_FormatTypeDef, [137](#)
- Class
 - DSY_SD_CardInfoTypeDef, [107](#)
- ClearLeds
 - daisy::DaisyPetal, [85](#)
 - daisy::DaisyPod, [92](#)
- clk
 - dsy_sr_4021_handle, [109](#)
- Close
 - daisy::WavPlayer, [140](#)
- codec_ak4556_init
 - dev_codec_ak4556.h, [157](#)
- codec_frame_t, [70](#)
 - l, [70](#)
 - r, [70](#)
- codec_pcm3060_init
 - dev_codec_pcm3060.h, [158](#)
- codec_wm8731_enter_bypass
 - dev_codec_wm8731.h, [159](#)
- codec_wm8731_exit_bypass
 - dev_codec_wm8731.h, [159](#)
- codec_wm8731_init
 - dev_codec_wm8731.h, [159](#)
- color, [71](#)
 - blue, [71](#)
 - green, [71](#)
 - red, [71](#)
- Configure
 - daisy::DaisySeed, [96](#)
- control_number
 - EXTERNAL, [33](#)
- controls
 - daisy::DaisyPatch, [80](#)
- cs
 - dsy_sr_4021_handle, [109](#)
- Ctrl
 - daisy::DaisyPatch, [77](#)
- cube
 - daisy_core.h, [148](#)
- Curve
 - CONTROLS, [15](#)
- cvs
 - daisy::daisy_field, [75](#)
- DAISY, [37](#)
- DEV_SR_4021_H
 - dev_sr_4021.h, [197](#)
- DEVICE_FS
 - USBD_CONF_Exported_Defines, [47](#)
- DEVICE_HS
 - USBD_CONF_Exported_Defines, [47](#)
- DEVICE_ID1
 - USBD_DESC_Exported_Constants, [54](#)
- DEVICE_ID2
 - USBD_DESC_Exported_Constants, [54](#)
- DEVICE_ID3
 - USBD_DESC_Exported_Constants, [54](#)
- DEV, [36](#)
- DIE_ERASE_CMD
 - dev_flash_IS25LP064A.h, [163](#)
 - dev_flash_IS25LP080D.h, [179](#)
- DMA_BUFFER_MEM_SECTION
 - daisy_core.h, [147](#)
- DSY_ADC_MAX_CHANNELS
 - per_adc.h, [211](#)
- DSY_ADC_H
 - per_adc.h, [211](#)
- DSY_BSP_SD_DISKIO_H
 - util_bsp_sd_diskio.h, [241](#)
- DSY_COLOR_H
 - util_color.h, [247](#)
- DSY_CORE_HW_H
 - daisy_core.h, [147](#)
- DSY_FIELD_BSP_H
 - daisy_field.h, [150](#)
- DSY_I2C_H
 - per_i2c.h, [218](#)
- DSY_LED_DRIVER_MAX_DRIVERS
 - dev_leddriver.h, [192](#)
- DSY_MEMORY_ERROR
 - per_qspi.h, [220](#)
- DSY_MEMORY_OK
 - per_qspi.h, [220](#)
- DSY_PETAL_H
 - daisy_petal.h, [156](#)
- DSY_QSPI_BSS
 - per_qspi.h, [220](#)
- DSY_QSPI_DATA
 - per_qspi.h, [221](#)
- DSY_QSPI_TEXT
 - per_qspi.h, [221](#)
- DSY_QSPI

- per_qspi.h, 220
- DSY_SD_CardInfoTypeDef, 106
 - BlockNbr, 106
 - BlockSize, 106
 - CardSpeed, 106
 - CardType, 106
 - CardVersion, 107
 - Class, 107
 - LogBlockNbr, 107
 - LogBlockSize, 107
 - RelCardAdd, 107
- DSY_SD_ERROR
 - per_sdmmc.h, 230
- DSY_SD_OK
 - per_sdmmc.h, 230
- DSY_SDMMC_H
 - per_sdmmc.h, 230
- DSY_SDRAM_BSS
 - dev_sdram.h, 195
- DSY_SDRAM_DATA
 - dev_sdram.h, 195
- DSY_UART_H
 - per_uart.h, 235
- DSY_WAVPLAYER_H
 - hid_wavplayer.h, 211
- DTCM_MEM_SECTION
 - daisy_core.h, 147
- DUAL_IN_FAST_PROG_CMD
 - dev_flash_IS25LP064A.h, 163
 - dev_flash_IS25LP080D.h, 179
- DUAL_INOUT_FAST_READ_4_BYTE_ADDR_CMD
 - dev_flash_IS25LP064A.h, 163
 - dev_flash_IS25LP080D.h, 179
- DUAL_INOUT_FAST_READ_CMD
 - dev_flash_IS25LP064A.h, 163
 - dev_flash_IS25LP080D.h, 179
- DUAL_INOUT_FAST_READ_DTR_CMD
 - dev_flash_IS25LP064A.h, 163
 - dev_flash_IS25LP080D.h, 179
- DUAL_OUT_FAST_READ_4_BYTE_ADDR_CMD
 - dev_flash_IS25LP064A.h, 164
 - dev_flash_IS25LP080D.h, 179
- DUAL_OUT_FAST_READ_CMD
 - dev_flash_IS25LP064A.h, 164
 - dev_flash_IS25LP080D.h, 179
- DUAL_OUT_FAST_READ_DTR_CMD
 - dev_flash_IS25LP064A.h, 164
 - dev_flash_IS25LP080D.h, 179
- dac_handle
 - daisy::DaisySeed, 97
- daisy.h
 - F2S16_SCALE, 144
 - F2S24_SCALE, 144
 - f2s16, 145
 - f2s24, 145
 - FBIPMAX, 144
 - FBIPMIN, 144
 - S162F_SCALE, 144
 - s162f, 145
 - S242F_SCALE, 145
 - s242f, 146
 - S24SIGN, 145
- daisy::AdcChannelConfig, 63
 - InitMux, 64
 - InitSingle, 64
 - mux_channels_, 65
 - mux_pin_, 65
 - MuxPin, 63
 - pin_, 65
- daisy::AdcHandle, 65
 - Get, 66
 - GetFloat, 67
 - GetMux, 67
 - GetMuxFloat, 67
 - GetMuxPtr, 68
 - GetPtr, 68
 - Init, 68
 - OverSampling, 66
 - Start, 69
 - Stop, 69
- daisy::AnalogControl, 69
- daisy::Color, 72
 - Blue, 73
 - Green, 73
 - Init, 73
 - PresetColor, 72
 - Red, 74
- daisy::ControlChangeEvent, 74
- daisy::DaisyPatch, 76
 - ~DaisyPatch, 78
 - AudioBlockSize, 78
 - AudioCallbackRate, 78
 - AudioSampleRate, 78
 - ChangeAudioCallback, 78
 - controls, 80
 - Ctrl, 77
 - DaisyPatch, 77
 - DebounceControls, 79
 - DelayMs, 79
 - display, 80
 - DisplayControls, 79
 - encoder, 81
 - gate_input, 81
 - gate_output, 81
 - GateInput, 77
 - GetCtrlValue, 79
 - Init, 79
 - midi, 81
 - seed, 81
 - SetAudioBlockSize, 79
 - StartAdc, 80
 - StartAudio, 80
 - UpdateAnalogControls, 80
- daisy::DaisyPetal, 81
 - ~DaisyPetal, 84
 - AudioBlockSize, 85

- AudioCallbackRate, [85](#)
- AudioSampleRate, [85](#)
- ChangeAudioCallback, [85](#)
- ClearLeds, [85](#)
- DaisyPetal, [84](#)
- DebounceControls, [85](#)
- DelayMs, [85](#)
- encoder, [88](#)
- expression, [88](#)
- footswitch_led, [88](#)
- FootswitchLed, [83](#)
- GetExpression, [86](#)
- GetKnobValue, [86](#)
- Init, [86](#)
- Knob, [83](#)
- knob, [88](#)
- ring_led, [89](#)
- RingLed, [83](#)
- seed, [89](#)
- SetAudioBlockSize, [86](#)
- SetFootswitchLed, [87](#)
- SetRingLed, [87](#)
- StartAdc, [87](#)
- StartAudio, [87](#)
- Sw, [84](#)
- switches, [89](#)
- UpdateAnalogControls, [88](#)
- UpdateLeds, [88](#)
- daisy::DaisyPod, [89](#)
 - AudioBlockSize, [91](#)
 - AudioCallbackRate, [91](#)
 - AudioSampleRate, [91](#)
 - button1, [93](#)
 - button2, [93](#)
 - buttons, [94](#)
 - ChangeAudioCallback, [91](#)
 - ClearLeds, [92](#)
 - DebounceControls, [92](#)
 - DelayMs, [92](#)
 - encoder, [94](#)
 - GetKnobValue, [92](#)
 - Init, [92](#)
 - Knob, [90](#)
 - knob1, [94](#)
 - knob2, [94](#)
 - knobs, [94](#)
 - led1, [94](#)
 - led2, [94](#)
 - seed, [94](#)
 - SetAudioBlockSize, [92](#)
 - StartAdc, [93](#)
 - StartAudio, [93](#)
 - Sw, [91](#)
 - UpdateAnalogControls, [93](#)
 - UpdateLeds, [93](#)
- daisy::DaisySeed, [95](#)
 - adc, [97](#)
 - audio_handle, [97](#)
 - AudioSampleRate, [95](#)
 - Configure, [96](#)
 - dac_handle, [97](#)
 - GetPin, [96](#)
 - i2c1_handle, [97](#)
 - i2c2_handle, [97](#)
 - Init, [96](#)
 - qspi_handle, [98](#)
 - sai_handle, [98](#)
 - sdram_handle, [98](#)
 - SetAudioBlockSize, [96](#)
 - SetLed, [96](#)
 - SetTestPoint, [96](#)
 - StartAudio, [97](#)
 - usb_handle, [98](#)
- daisy::Encoder, [110](#)
- daisy::GateIn, [111](#)
 - ~GateIn, [112](#)
 - GateIn, [112](#)
 - Init, [112](#)
 - Trig, [112](#)
- daisy::Led, [113](#)
- daisy::MidiEvent, [113](#)
- daisy::MidiHandler, [114](#)
- daisy::NoteOnEvent, [115](#)
- daisy::OledDisplay, [115](#)
- daisy::Parameter, [116](#)
- daisy::RgbLed, [117](#)
- daisy::RingBuffer
 - capacity, [118](#)
 - Flush, [118](#)
 - ImmediateRead, [118](#)
 - Init, [119](#)
 - Overwrite, [119](#)
 - Read, [119](#)
 - readable, [119](#)
 - Swallow, [120](#)
 - writable, [120](#)
 - Write, [120](#)
- daisy::RingBuffer< T, 0 >, [121](#)
 - capacity, [121](#)
 - Flush, [121](#)
 - ImmediateRead, [122](#)
 - Init, [122](#)
 - Overwrite, [122](#), [123](#)
 - Read, [123](#)
 - readable, [123](#)
 - writable, [123](#)
 - Write, [123](#)
- daisy::RingBuffer< T, size >, [117](#)
- daisy::SdmmcHandler, [124](#)
 - Init, [124](#)
- daisy::SdmmcHandlerInit, [124](#)
 - bitdepth, [125](#)
 - speed, [125](#)
- daisy::SpiHandle, [127](#)
 - BlockingTransmit, [127](#)
 - Init, [128](#)

- daisy::Switch, 128
- daisy::UartHandler, 129
 - CheckError, 129
 - FlushRx, 129
 - Init, 130
 - PollReceive, 130
 - PollTx, 130
 - PopRx, 131
 - Readable, 131
 - RxActive, 131
 - StartRx, 131
- daisy::WavFileInfo, 139
 - name, 139
 - raw_data, 139
- daisy::WavPlayer, 140
 - Close, 140
 - GetCurrentFile, 140
 - GetLooping, 140
 - GetNumberFiles, 141
 - Init, 141
 - Open, 141
 - Prepare, 141
 - Restart, 141
 - SetLooping, 142
 - Stream, 142
- daisy::daisy_field, 74
 - cvs, 75
 - gate_in, 75
 - gate_out, 75
 - keyboard_sr, 75
 - knobs, 75
 - seed, 75
 - switches, 76
- daisy_core.h
 - cube, 148
 - DMA_BUFFER_MEM_SECTION, 147
 - DSY_CORE_HW_H, 147
 - DTCM_MEM_SECTION, 147
 - dsy_gpio_port, 147
 - dsy_pin, 148
 - dsy_pin_cmp, 148
- daisy_field.h
 - CV1_ADC_PIN, 150
 - CV2_ADC_PIN, 150
 - CV3_ADC_PIN, 150
 - CV4_ADC_PIN, 150
 - DSY_FIELD_BSP_H, 150
 - daisy_field_init, 154
 - GATE_IN_PIN, 150
 - GATE_OUT_PIN, 150
 - KB_SW_SR_CLK_PIN, 151
 - KB_SW_SR_CS_PIN, 151
 - KB_SW_SR_D1_PIN, 151
 - KB_SW_SR_D2_PIN, 151
 - LED_DRIVER_I2C, 151
 - MIDI_IN_PIN, 151
 - MIDI_OUT_PIN, 151
 - MUX_ADC_PIN, 151
 - MUX_SEL_0_PIN, 152
 - MUX_SEL_1_PIN, 152
 - MUX_SEL_2_PIN, 152
 - SAMPLE_RATE, 152
 - SW_1_PIN, 152
 - SW_2_PIN, 152
 - SW_3_PIN, 152
- daisy_field_init
 - daisy_field.h, 154
- daisy_petal.h
 - DSY_PETAL_H, 156
- DaisyPatch
 - daisy::DaisyPatch, 77
- DaisyPetal
 - daisy::DaisyPetal, 84
- data
 - dsy_sr_4021_handle, 109
 - EXTERNAL, 33
 - FontDef, 111
- Debounce
 - CONTROLS, 16
- DebounceControls
 - daisy::DaisyPatch, 79
 - daisy::DaisyPetal, 85
 - daisy::DaisyPod, 92
- DelayMs
 - daisy::DaisyPatch, 79
 - daisy::DaisyPetal, 85
 - daisy::DaisyPod, 92
- dev0_i2c
 - dsy_audio_handle, 99
- dev1_i2c
 - dsy_audio_handle, 99
- dev_codec_ak4556.h
 - codec_ak4556_init, 157
- dev_codec_pcm3060.h
 - codec_pcm3060_init, 158
- dev_codec_wm8731.h
 - codec_wm8731_enter_bypass, 159
 - codec_wm8731_exit_bypass, 159
 - codec_wm8731_init, 159
- dev_codec_wm8731_frame.h
 - sa_audio_callback, 160
- dev_flash_IS25LP064A.h
 - BLOCK_ERASE_32K_CMD, 163
 - CLEAR_FLAG_STATUS_REG_CMD, 163
 - DIE_ERASE_CMD, 163
 - DUAL_IN_FAST_PROG_CMD, 163
 - DUAL_INOUT_FAST_READ_4_BYTE_ADDR_CMD, 163
 - DUAL_INOUT_FAST_READ_CMD, 163
 - DUAL_INOUT_FAST_READ_DTR_CMD, 163
 - DUAL_OUT_FAST_READ_4_BYTE_ADDR_CMD, 164
 - DUAL_OUT_FAST_READ_CMD, 164
 - DUAL_OUT_FAST_READ_DTR_CMD, 164
 - ENTER_4_BYTE_ADDR_MODE_CMD, 164
 - ENTER_QUAD_CMD, 164

EXIT_4_BYTE_ADDR_MODE_CMD, 164
 EXIT_QUAD_CMD, 164
 EXT_DUAL_IN_FAST_PROG_CMD, 164
 EXT_QUAD_IN_FAST_PROG_CMD, 165
 FAST_READ_4_BYTE_ADDR_CMD, 165
 FAST_READ_CMD, 165
 FAST_READ_DTR_CMD, 165
 IS25LP064A_DIE_ERASE_MAX_TIME, 165
 IS25LP064A_DUMMY_CYCLES_READ_DTR, 165
 IS25LP064A_DUMMY_CYCLES_READ_QUAD↵
 DTR, 166
 IS25LP064A_DUMMY_CYCLES_READ_QUAD, 165
 IS25LP064A_DUMMY_CYCLES_READ, 165
 IS25LP064A_EAR_HIGHEST_SE, 166
 IS25LP064A_EAR_LOWEST_SEG, 166
 IS25LP064A_EAR_SECOND_SEG, 166
 IS25LP064A_EAR_THIRD_SEG, 166
 IS25LP064A_EVCR_DTRP, 166
 IS25LP064A_EVCR_DUAL, 166
 IS25LP064A_EVCR_ODS, 166
 IS25LP064A_EVCR_QUAD, 167
 IS25LP064A_EVCR_RH, 167
 IS25LP064A_FLASH_SIZE, 167
 IS25LP064A_FSR_ERERR, 167
 IS25LP064A_FSR_ERSUS, 167
 IS25LP064A_FSR_NBADDR, 167
 IS25LP064A_FSR_PGERR, 167
 IS25LP064A_FSR_PGSUS, 167
 IS25LP064A_FSR_PRERR, 168
 IS25LP064A_FSR_READY, 168
 IS25LP064A_NVCR_DTRP, 168
 IS25LP064A_NVCR_DUAL, 168
 IS25LP064A_NVCR_NB_DUMMY, 168
 IS25LP064A_NVCR_NBADDR, 168
 IS25LP064A_NVCR_ODS, 168
 IS25LP064A_NVCR_QUAB, 169
 IS25LP064A_NVCR_RH, 169
 IS25LP064A_NVCR_SEGMENT, 169
 IS25LP064A_NVCR_XIP, 169
 IS25LP064A_PAGE_SIZE, 169
 IS25LP064A_SECTOR_ERASE_MAX_TIME, 169
 IS25LP064A_SECTOR_SIZE, 169
 IS25LP064A_SR_QE, 169
 IS25LP064A_SR_SRWREN, 170
 IS25LP064A_SR_WIP, 170
 IS25LP064A_SR_WREN, 170
 IS25LP064A_SUBSECTOR_ERASE_MAX_TIME, 170
 IS25LP064A_SUBSECTOR_SIZE, 170
 IS25LP064A_VCR_NB_DUMMY, 170
 IS25LP064A_VCR_WRAP, 170
 IS25LP064A_VCR_XIP, 171
 IS25LP064A_H, 168
 MULTIPLE_IO_READ_ID_CMD, 171
 PAGE_PROG_4_BYTE_ADDR_CMD, 171
 PAGE_PROG_CMD, 171
 PROG_ERASE_RESUME_CMD, 171
 PROG_ERASE_SUSPEND_CMD, 171
 PROG_OTP_ARRAY_CMD, 171
 QUAD_IN_FAST_PROG_4_BYTE_ADDR_CMD, 171
 QUAD_IN_FAST_PROG_CMD, 172
 QUAD_INOUT_FAST_READ_4_BYTE_ADDR↵
 CMD, 172
 QUAD_INOUT_FAST_READ_CMD, 172
 QUAD_INOUT_FAST_READ_DTR_CMD, 172
 QUAD_OUT_FAST_READ_4_BYTE_ADDR↵
 MD, 172
 QUAD_OUT_FAST_READ_CMD, 172
 QUAD_OUT_FAST_READ_DTR_CMD, 172
 READ_4_BYTE_ADDR_CMD, 172
 READ_CMD, 173
 READ_ENHANCED_VOL_CFG_REG_CMD, 173
 READ_EXT_ADDR_REG_CMD, 173
 READ_FLAG_STATUS_REG_CMD, 173
 READ_ID_CMD2, 173
 READ_ID_CMD, 173
 READ_LOCK_REG_CMD, 173
 READ_NONVOL_CFG_REG_CMD, 173
 READ_OTP_ARRAY_CMD, 174
 READ_READ_PARAM_REG_CMD, 174
 READ_SERIAL_FLASH_DISCO_PARAM_CMD, 174
 READ_STATUS_REG_CMD, 174
 RESET_ENABLE_CMD, 174
 RESET_MEMORY_CMD, 174
 SECTOR_ERASE_4_BYTE_ADDR_CMD, 174
 SECTOR_ERASE_CMD, 174
 SUBSECTOR_ERASE_4_BYTE_ADDR_CMD, 175
 SUBSECTOR_ERASE_CMD, 175
 SUBSECTOR_ERASE_QPI_CMD, 175
 WRITE_DISABLE_CMD, 175
 WRITE_ENABLE_CMD, 175
 WRITE_ENHANCED_VOL_CFG_REG_CMD, 175
 WRITE_EXT_ADDR_REG_CMD, 175
 WRITE_LOCK_REG_CMD, 175
 WRITE_NONVOL_CFG_REG_CMD, 176
 WRITE_READ_PARAM_REG_CMD, 176
 WRITE_STATUS_REG_CMD, 176
 dev_flash_IS25LP080D.h
 BLOCK_ERASE_32K_CMD, 178
 CLEAR_FLAG_STATUS_REG_CMD, 178
 DIE_ERASE_CMD, 179
 DUAL_IN_FAST_PROG_CMD, 179
 DUAL_INOUT_FAST_READ_4_BYTE_ADDR↵
 CMD, 179
 DUAL_INOUT_FAST_READ_CMD, 179
 DUAL_INOUT_FAST_READ_DTR_CMD, 179
 DUAL_OUT_FAST_READ_4_BYTE_ADDR↵
 MD, 179
 DUAL_OUT_FAST_READ_CMD, 179
 DUAL_OUT_FAST_READ_DTR_CMD, 179
 ENTER_4_BYTE_ADDR_MODE_CMD, 180

ENTER_QUAD_CMD, 180
 EXIT_4_BYTE_ADDR_MODE_CMD, 180
 EXIT_QUAD_CMD, 180
 EXT_DUAL_IN_FAST_PROG_CMD, 180
 EXT_QUAD_IN_FAST_PROG_CMD, 180
 FAST_READ_4_BYTE_ADDR_CMD, 180
 FAST_READ_CMD, 180
 FAST_READ_DTR_CMD, 181
 IS25LP080D_DIE_ERASE_MAX_TIME, 181
 IS25LP080D_DUMMY_CYCLES_READ_DTR, 181
 IS25LP080D_DUMMY_CYCLES_READ_QUAD↔_DTR, 181
 IS25LP080D_DUMMY_CYCLES_READ_QUAD, 181
 IS25LP080D_DUMMY_CYCLES_READ, 181
 IS25LP080D_EAR_HIGHEST_SE, 181
 IS25LP080D_EAR_LOWEST_SEG, 181
 IS25LP080D_EAR_SECOND_SEG, 182
 IS25LP080D_EAR_THIRD_SEG, 182
 IS25LP080D_EVCR_DTRP, 182
 IS25LP080D_EVCR_DUAL, 182
 IS25LP080D_EVCR_ODS, 182
 IS25LP080D_EVCR_QUAD, 182
 IS25LP080D_EVCR_RH, 182
 IS25LP080D_FLASH_SIZE, 182
 IS25LP080D_FSR_ERERR, 183
 IS25LP080D_FSR_ERSUS, 183
 IS25LP080D_FSR_NBADDR, 183
 IS25LP080D_FSR_PGERR, 183
 IS25LP080D_FSR_PGSUS, 183
 IS25LP080D_FSR_PRERR, 183
 IS25LP080D_FSR_READY, 183
 IS25LP080D_NVCR_DTRP, 183
 IS25LP080D_NVCR_DUAL, 184
 IS25LP080D_NVCR_NB_DUMMY, 184
 IS25LP080D_NVCR_NBADDR, 184
 IS25LP080D_NVCR_ODS, 184
 IS25LP080D_NVCR_QUAB, 184
 IS25LP080D_NVCR_RH, 184
 IS25LP080D_NVCR_SEGMENT, 184
 IS25LP080D_NVCR_XIP, 184
 IS25LP080D_PAGE_SIZE, 185
 IS25LP080D_SECTOR_ERASE_MAX_TIME, 185
 IS25LP080D_SECTOR_SIZE, 185
 IS25LP080D_SR_QE, 185
 IS25LP080D_SR_SRWREN, 185
 IS25LP080D_SR_WIP, 185
 IS25LP080D_SR_WREN, 185
 IS25LP080D_SUBSECTOR_ERASE_MAX_TIME, 186
 IS25LP080D_SUBSECTOR_SIZE, 186
 IS25LP080D_VCR_NB_DUMMY, 186
 IS25LP080D_VCR_WRAP, 186
 IS25LP080D_VCR_XIP, 186
 MULTIPLE_IO_READ_ID_CMD, 186
 PAGE_PROG_4_BYTE_ADDR_CMD, 186
 PAGE_PROG_CMD, 186
 PROG_ERASE_RESUME_CMD, 187
 PROG_ERASE_SUSPEND_CMD, 187
 PROG_OTP_ARRAY_CMD, 187
 QUAD_IN_FAST_PROG_4_BYTE_ADDR_CMD, 187
 QUAD_IN_FAST_PROG_CMD, 187
 QUAD_INOUT_FAST_READ_4_BYTE_ADDR↔_CMD, 187
 QUAD_INOUT_FAST_READ_CMD, 187
 QUAD_INOUT_FAST_READ_DTR_CMD, 187
 QUAD_OUT_FAST_READ_4_BYTE_ADDR↔_MD, 188
 QUAD_OUT_FAST_READ_CMD, 188
 QUAD_OUT_FAST_READ_DTR_CMD, 188
 READ_4_BYTE_ADDR_CMD, 188
 READ_CMD, 188
 READ_ENHANCED_VOL_CFG_REG_CMD, 188
 READ_EXT_ADDR_REG_CMD, 188
 READ_FLAG_STATUS_REG_CMD, 188
 READ_ID_CMD2, 189
 READ_ID_CMD, 189
 READ_LOCK_REG_CMD, 189
 READ_NONVOL_CFG_REG_CMD, 189
 READ_OTP_ARRAY_CMD, 189
 READ_READ_PARAM_REG_CMD, 189
 READ_SERIAL_FLASH_DISCO_PARAM_CMD, 189
 READ_STATUS_REG_CMD, 189
 RESET_ENABLE_CMD, 190
 RESET_MEMORY_CMD, 190
 SECTOR_ERASE_4_BYTE_ADDR_CMD, 190
 SECTOR_ERASE_CMD, 190
 SUBSECTOR_ERASE_4_BYTE_ADDR_CMD, 190
 SUBSECTOR_ERASE_CMD, 190
 SUBSECTOR_ERASE_QPI_CMD, 190
 WRITE_DISABLE_CMD, 190
 WRITE_ENABLE_CMD, 191
 WRITE_ENHANCED_VOL_CFG_REG_CMD, 191
 WRITE_EXT_ADDR_REG_CMD, 191
 WRITE_LOCK_REG_CMD, 191
 WRITE_NONVOL_CFG_REG_CMD, 191
 WRITE_READ_PARAM_REG_CMD, 191
 WRITE_STATUS_REG_CMD, 191
 dev_leddriver.h
 DSY_LED_DRIVER_MAX_DRIVERS, 192
 dsy_led_driver_color_by_name, 193
 dsy_led_driver_init, 193
 dsy_led_driver_set_led, 194
 dsy_led_driver_update, 194
 SA_LED_DRIVER_H, 192
 dev_sdram.h
 DSY_SDRAM_BSS, 195
 DSY_SDRAM_DATA, 195
 dsy_sdram_init, 196
 dsy_sdram_pin, 196
 dsy_sdram_state, 196
 RAM_AS4C16M16SA_H, 195

- dev_sr_4021.h
 - DEV_SR_4021_H, [197](#)
 - dsy_sr_4021_init, [198](#)
 - dsy_sr_4021_state, [200](#)
 - dsy_sr_4021_update, [200](#)
 - SR_4021_MAX_DAISYCHAIN, [198](#)
 - SR_4021_MAX_PARALLEL, [198](#)
- dev_sr_595.h
 - kMaxSr595DaisyChain, [201](#)
- device
 - dsy_qspi_handle, [103](#)
 - dsy_sai_handle, [105](#)
- display
 - daisy::DaisyPatch, [80](#)
- DisplayControls
 - daisy::DaisyPatch, [79](#)
- DrawPixel
 - FEEDBACK, [23](#)
- dsy_audio_bitdepth
 - per_sai.h, [226](#)
- dsy_audio_callback
 - AUDIO, [11](#)
- dsy_audio_device
 - per_sai.h, [226](#)
- dsy_audio_dir
 - per_sai.h, [226](#)
- dsy_audio_enter_bypass
 - AUDIO, [12](#)
- dsy_audio_exit_bypass
 - AUDIO, [12](#)
- dsy_audio_handle, [98](#)
 - block_size, [99](#)
 - dev0_i2c, [99](#)
 - dev1_i2c, [99](#)
 - sai, [99](#)
- dsy_audio_init
 - AUDIO, [12](#)
- dsy_audio_mc_callback
 - AUDIO, [11](#)
- dsy_audio_passthru
 - AUDIO, [12](#)
- dsy_audio_sai
 - per_sai.h, [227](#)
- dsy_audio_samplerate
 - per_sai.h, [227](#)
- dsy_audio_set_blocksize
 - AUDIO, [12](#)
- dsy_audio_set_callback
 - AUDIO, [13](#)
- dsy_audio_set_mc_callback
 - AUDIO, [13](#)
- dsy_audio_silence
 - AUDIO, [13](#)
- dsy_audio_start
 - AUDIO, [13](#)
- dsy_audio_stop
 - AUDIO, [13](#)
- dsy_audio_sync
 - per_sai.h, [227](#)
- dsy_dac_bitdepth
 - per_dac.h, [212](#)
- dsy_dac_channel
 - per_dac.h, [212](#)
- dsy_dac_handle, [99](#)
 - bitdepth, [100](#)
 - mode, [100](#)
 - pin_config, [100](#)
- dsy_dac_init
 - per_dac.h, [213](#)
- dsy_dac_mode
 - per_dac.h, [213](#)
- dsy_dac_start
 - per_dac.h, [213](#)
- dsy_dac_write
 - per_dac.h, [214](#)
- dsy_dma_init
 - sys_dma.h, [236](#)
- dsy_fatfs_init
 - fatfs.h, [202](#)
- dsy_get_unique_id
 - util_unique_id.h, [249](#)
- dsy_gpio, [100](#)
 - mode, [101](#)
 - pin, [101](#)
 - pull, [101](#)
- dsy_gpio_deinit
 - per_gpio.h, [216](#)
- dsy_gpio_init
 - per_gpio.h, [216](#)
- dsy_gpio_mode
 - per_gpio.h, [215](#)
- dsy_gpio_pin, [101](#)
 - pin, [101](#)
 - port, [102](#)
- dsy_gpio_port
 - daisy_core.h, [147](#)
- dsy_gpio_pull
 - per_gpio.h, [215](#)
- dsy_gpio_read
 - per_gpio.h, [216](#)
- dsy_gpio_toggle
 - per_gpio.h, [216](#)
- dsy_gpio_write
 - per_gpio.h, [217](#)
- dsy_hal_map_get_i2c
 - util_hal_map.h, [247](#)
- dsy_hal_map_get_pin
 - util_hal_map.h, [248](#)
- dsy_hal_map_get_port
 - util_hal_map.h, [248](#)
- dsy_i2c_handle, [102](#)
 - periph, [102](#)
 - pin_config, [102](#)
 - speed, [102](#)
- dsy_i2c_init
 - per_i2c.h, [219](#)

- dsy_i2c_periph
 - per_i2c.h, 218
- dsy_i2c_pin
 - per_i2c.h, 218
- dsy_i2c_speed
 - per_i2c.h, 219
- dsy_led_driver_color_by_name
 - dev_leddriver.h, 193
- dsy_led_driver_init
 - dev_leddriver.h, 193
- dsy_led_driver_set_led
 - dev_leddriver.h, 194
- dsy_led_driver_update
 - dev_leddriver.h, 194
- dsy_pin
 - daisy_core.h, 148
- dsy_pin_cmp
 - daisy_core.h, 148
- dsy_qspi_deinit
 - per_qspi.h, 222
- dsy_qspi_device
 - per_qspi.h, 221
- dsy_qspi_erase
 - per_qspi.h, 223
- dsy_qspi_erasesector
 - per_qspi.h, 223
- dsy_qspi_handle, 103
 - device, 103
 - mode, 103
 - pin_config, 103
- dsy_qspi_init
 - per_qspi.h, 223
- dsy_qspi_mode
 - per_qspi.h, 222
- dsy_qspi_pin
 - per_qspi.h, 222
- dsy_qspi_write
 - per_qspi.h, 224
- dsy_qspi_writepage
 - per_qspi.h, 224
- dsy_sai_handle, 104
 - a_direction, 104
 - b_direction, 104
 - bitdepth, 104
 - device, 105
 - init, 105
 - sai1_pin_config, 105
 - sai2_pin_config, 105
 - samplerate, 105
 - sync_config, 105
- dsy_sai_init
 - per_sai.h, 228
- dsy_sai_init_from_handle
 - per_sai.h, 228
- dsy_sai_pin
 - per_sai.h, 227
- dsy_sdram_handle, 107
 - pin_config, 108
 - state, 108
- dsy_sdram_init
 - dev_sdram.h, 196
- dsy_sdram_pin
 - dev_sdram.h, 196
- dsy_sdram_state
 - dev_sdram.h, 196
- dsy_sr_4021_handle, 108
 - clk, 109
 - cs, 109
 - data, 109
 - num_daisychained, 109
 - num_parallel, 109
 - pin_config, 109
 - states, 109
- dsy_sr_4021_init
 - dev_sr_4021.h, 198
- dsy_sr_4021_state
 - dev_sr_4021.h, 200
- dsy_sr_4021_update
 - dev_sr_4021.h, 200
- dsy_system_delay
 - sys_system.h, 236
- dsy_system_getnow
 - sys_system.h, 237
- dsy_system_init
 - sys_system.h, 237
- dsy_system_jumpto
 - sys_system.h, 237
- dsy_system_jumptoqspi
 - sys_system.h, 237
- dsy_tim_delay_ms
 - per_tim.h, 233
- dsy_tim_delay_tick
 - per_tim.h, 233
- dsy_tim_delay_us
 - per_tim.h, 233
- dsy_tim_get_ms
 - per_tim.h, 234
- dsy_tim_get_tick
 - per_tim.h, 234
- dsy_tim_get_us
 - per_tim.h, 234
- dsy_tim_init
 - per_tim.h, 234
- dsy_tim_start
 - per_tim.h, 235
- ENTER_4_BYTE_ADDR_MODE_CMD
 - dev_flash_IS25LP064A.h, 164
 - dev_flash_IS25LP080D.h, 180
- ENTER_QUAD_CMD
 - dev_flash_IS25LP064A.h, 164
 - dev_flash_IS25LP080D.h, 180
- EXIT_4_BYTE_ADDR_MODE_CMD
 - dev_flash_IS25LP064A.h, 164
 - dev_flash_IS25LP080D.h, 180
- EXIT_QUAD_CMD
 - dev_flash_IS25LP064A.h, 164

- dev_flash_IS25LP080D.h, [180](#)
- EXT_DUAL_IN_FAST_PROG_CMD
 - dev_flash_IS25LP064A.h, [164](#)
 - dev_flash_IS25LP080D.h, [180](#)
- EXT_QUAD_IN_FAST_PROG_CMD
 - dev_flash_IS25LP064A.h, [165](#)
 - dev_flash_IS25LP080D.h, [180](#)
- EXTERNAL, [28](#)
 - AsControlChange, [31](#)
 - AsNoteOn, [31](#)
 - channel, [32](#), [33](#)
 - control_number, [33](#)
 - data, [33](#)
 - HasEvents, [31](#)
 - Init, [31](#)
 - Listen, [32](#)
 - MidiInputMode, [29](#)
 - MidiMessageType, [29](#)
 - MidiOutputMode, [29](#)
 - note, [33](#)
 - Parse, [32](#)
 - PopEvent, [32](#)
 - StartReceive, [32](#)
 - type, [33](#)
 - value, [33](#)
 - velocity, [33](#)
- encoder
 - daisy::DaisyPatch, [81](#)
 - daisy::DaisyPetal, [88](#)
 - daisy::DaisyPod, [94](#)
- expression
 - daisy::DaisyPetal, [88](#)
- Externals, [60](#)
- F2S16_SCALE
 - daisy.h, [144](#)
- F2S24_SCALE
 - daisy.h, [144](#)
- f2s16
 - daisy.h, [145](#)
- f2s24
 - daisy.h, [145](#)
- FAST_READ_4_BYTE_ADDR_CMD
 - dev_flash_IS25LP064A.h, [165](#)
 - dev_flash_IS25LP080D.h, [180](#)
- FAST_READ_CMD
 - dev_flash_IS25LP064A.h, [165](#)
 - dev_flash_IS25LP080D.h, [180](#)
- FAST_READ_DTR_CMD
 - dev_flash_IS25LP064A.h, [165](#)
 - dev_flash_IS25LP080D.h, [181](#)
- FBIPMAX
 - daisy.h, [144](#)
- FBIPMIN
 - daisy.h, [144](#)
- FEEDBACK, [22](#)
 - DrawPixel, [23](#)
 - Fill, [23](#)
 - Init, [23](#), [24](#)
 - Pins, [22](#)
 - Set, [24](#), [25](#)
 - SetColor, [25](#)
 - SetCursor, [25](#)
 - Update, [26](#)
 - WriteChar, [26](#)
 - WriteString, [26](#)
- FS_Desc
 - USBD_DESC_Exported_Variables, [58](#)
- FallingEdge
 - CONTROLS, [16](#)
- fatfs.h
 - __fatfs_H, [201](#)
 - dsy_fatfs_init, [202](#)
 - retSD, [202](#)
 - SDFatFS, [202](#)
 - SDFFile, [202](#)
 - SDPath, [202](#)
- ff_free
 - ffconf.h, [210](#)
- ff_malloc
 - ffconf.h, [210](#)
- ffconf.h
 - _CODE_PAGE, [204](#)
 - _FFCONF, [204](#)
 - _FS_EXFAT, [204](#)
 - _FS_LOCK, [204](#)
 - _FS_MINIMIZE, [204](#)
 - _FS_NOFSINFO, [205](#)
 - _FS_NORTC, [205](#)
 - _FS_READONLY, [205](#)
 - _FS_REENTRANT, [205](#)
 - _FS_RPATH, [205](#)
 - _FS_TIMEOUT, [205](#)
 - _FS_TINY, [206](#)
 - _LFN_UNICODE, [206](#)
 - _MAX_LFN, [206](#)
 - _MAX_SS, [206](#)
 - _MIN_SS, [206](#)
 - _MULTI_PARTITION, [206](#)
 - _NORTC_MDAY, [207](#)
 - _NORTC_MON, [207](#)
 - _NORTC_YEAR, [207](#)
 - _STRF_ENCODE, [207](#)
 - _STR_VOLUME_ID, [207](#)
 - _SYNC_t, [207](#)
 - _USE_CHMOD, [208](#)
 - _USE_EXPAND, [208](#)
 - _USE_FASTSEEK, [208](#)
 - _USE_FIND, [208](#)
 - _USE_FORWARD, [208](#)
 - _USE_LABEL, [208](#)
 - _USE_LFN, [209](#)
 - _USE_MKFS, [209](#)
 - _USE_STRFUNC, [209](#)
 - _USE_TRIM, [209](#)
 - _VOLUMES, [209](#)
 - _VOLUME_STRS, [209](#)

- ff_free, 210
- ff_malloc, 210
- FileFormat
 - WAV_FormatTypeDef, 137
- FileSize
 - WAV_FormatTypeDef, 138
- Fill
 - FEEDBACK, 23
- Flush
 - daisy::RingBuffer, 118
 - daisy::RingBuffer< T, 0 >, 121
- FlushRx
 - daisy::UartHandler, 129
- FontDef, 110
 - data, 111
 - FontHeight, 111
 - FontWidth, 111
- FontHeight
 - FontDef, 111
- FontWidth
 - FontDef, 111
- footswitch_led
 - daisy::DaisyPetal, 88
- FootswitchLed
 - daisy::DaisyPetal, 83
- GATE_IN_PIN
 - daisy_field.h, 150
- GATE_OUT_PIN
 - daisy_field.h, 150
- gate_in
 - daisy::daisy_field, 75
- gate_input
 - daisy::DaisyPatch, 81
- gate_out
 - daisy::daisy_field, 75
- gate_output
 - daisy::DaisyPatch, 81
- Gateln
 - daisy::Gateln, 112
- GatelnInput
 - daisy::DaisyPatch, 77
- Get
 - daisy::AdcHandle, 66
- GetCtrlValue
 - daisy::DaisyPatch, 79
- GetCurrentFile
 - daisy::WavPlayer, 140
- GetExpression
 - daisy::DaisyPetal, 86
- GetFloat
 - daisy::AdcHandle, 67
- GetKnobValue
 - daisy::DaisyPetal, 86
 - daisy::DaisyPod, 92
- GetLooping
 - daisy::WavPlayer, 140
- GetMux
 - daisy::AdcHandle, 67
- GetMuxFloat
 - daisy::AdcHandle, 67
- GetMuxPtr
 - daisy::AdcHandle, 68
- GetNumberFiles
 - daisy::WavPlayer, 141
- GetPin
 - daisy::DaisySeed, 96
- GetPtr
 - daisy::AdcHandle, 68
- Green
 - daisy::Color, 73
- green
 - color, 71
- HS_Desc
 - USBD_DESC_Exported_Variables, 58
- HUMAN_INTERFACE, 10
- HasEvents
 - EXTERNAL, 31
- hi2c1
 - util_hal_map.h, 248
- hi2c2
 - util_hal_map.h, 248
- hi2c3
 - util_hal_map.h, 249
- hi2c4
 - util_hal_map.h, 249
- hid_wavplayer.h
 - DSY_WAVPLAYER_H, 211
 - WAV_FILENAME_MAX, 211
- i2c1_handle
 - daisy::DaisySeed, 97
- i2c2_handle
 - daisy::DaisySeed, 97
- IS25LP064A_DIE_ERASE_MAX_TIME
 - dev_flash_IS25LP064A.h, 165
- IS25LP064A_DUMMY_CYCLES_READ_DTR
 - dev_flash_IS25LP064A.h, 165
- IS25LP064A_DUMMY_CYCLES_READ_QUAD_DTR
 - dev_flash_IS25LP064A.h, 166
- IS25LP064A_DUMMY_CYCLES_READ_QUAD
 - dev_flash_IS25LP064A.h, 165
- IS25LP064A_DUMMY_CYCLES_READ
 - dev_flash_IS25LP064A.h, 165
- IS25LP064A_EAR_HIGHEST_SE
 - dev_flash_IS25LP064A.h, 166
- IS25LP064A_EAR_LOWEST_SEG
 - dev_flash_IS25LP064A.h, 166
- IS25LP064A_EAR_SECOND_SEG
 - dev_flash_IS25LP064A.h, 166
- IS25LP064A_EAR_THIRD_SEG
 - dev_flash_IS25LP064A.h, 166
- IS25LP064A_EVCR_DTRP
 - dev_flash_IS25LP064A.h, 166
- IS25LP064A_EVCR_DUAL
 - dev_flash_IS25LP064A.h, 166
- IS25LP064A_EVCR_ODS

dev_flash_IS25LP064A.h, [166](#)
IS25LP064A_EVCR_QUAD
dev_flash_IS25LP064A.h, [167](#)
IS25LP064A_EVCR_RH
dev_flash_IS25LP064A.h, [167](#)
IS25LP064A_FLASH_SIZE
dev_flash_IS25LP064A.h, [167](#)
IS25LP064A_FSR_ERERR
dev_flash_IS25LP064A.h, [167](#)
IS25LP064A_FSR_ERSUS
dev_flash_IS25LP064A.h, [167](#)
IS25LP064A_FSR_NBADDR
dev_flash_IS25LP064A.h, [167](#)
IS25LP064A_FSR_PGERR
dev_flash_IS25LP064A.h, [167](#)
IS25LP064A_FSR_PGSUS
dev_flash_IS25LP064A.h, [167](#)
IS25LP064A_FSR_PRERR
dev_flash_IS25LP064A.h, [168](#)
IS25LP064A_FSR_READY
dev_flash_IS25LP064A.h, [168](#)
IS25LP064A_NVCR_DTRP
dev_flash_IS25LP064A.h, [168](#)
IS25LP064A_NVCR_DUAL
dev_flash_IS25LP064A.h, [168](#)
IS25LP064A_NVCR_NB_DUMMY
dev_flash_IS25LP064A.h, [168](#)
IS25LP064A_NVCR_NBADDR
dev_flash_IS25LP064A.h, [168](#)
IS25LP064A_NVCR_ODS
dev_flash_IS25LP064A.h, [168](#)
IS25LP064A_NVCR_QUAB
dev_flash_IS25LP064A.h, [169](#)
IS25LP064A_NVCR_RH
dev_flash_IS25LP064A.h, [169](#)
IS25LP064A_NVCR_SEGMENT
dev_flash_IS25LP064A.h, [169](#)
IS25LP064A_NVCR_XIP
dev_flash_IS25LP064A.h, [169](#)
IS25LP064A_PAGE_SIZE
dev_flash_IS25LP064A.h, [169](#)
IS25LP064A_SECTOR_ERASE_MAX_TIME
dev_flash_IS25LP064A.h, [169](#)
IS25LP064A_SECTOR_SIZE
dev_flash_IS25LP064A.h, [169](#)
IS25LP064A_SR_QE
dev_flash_IS25LP064A.h, [169](#)
IS25LP064A_SR_SRWREN
dev_flash_IS25LP064A.h, [170](#)
IS25LP064A_SR_WIP
dev_flash_IS25LP064A.h, [170](#)
IS25LP064A_SR_WREN
dev_flash_IS25LP064A.h, [170](#)
IS25LP064A_SUBSECTOR_ERASE_MAX_TIME
dev_flash_IS25LP064A.h, [170](#)
IS25LP064A_SUBSECTOR_SIZE
dev_flash_IS25LP064A.h, [170](#)
IS25LP064A_VCR_NB_DUMMY
dev_flash_IS25LP064A.h, [170](#)
IS25LP064A_VCR_WRAP
dev_flash_IS25LP064A.h, [170](#)
IS25LP064A_VCR_XIP
dev_flash_IS25LP064A.h, [171](#)
IS25LP064A_H
dev_flash_IS25LP064A.h, [168](#)
IS25LP080D_DIE_ERASE_MAX_TIME
dev_flash_IS25LP080D.h, [181](#)
IS25LP080D_DUMMY_CYCLES_READ_DTR
dev_flash_IS25LP080D.h, [181](#)
IS25LP080D_DUMMY_CYCLES_READ_QUAD_DTR
dev_flash_IS25LP080D.h, [181](#)
IS25LP080D_DUMMY_CYCLES_READ_QUAD
dev_flash_IS25LP080D.h, [181](#)
IS25LP080D_DUMMY_CYCLES_READ
dev_flash_IS25LP080D.h, [181](#)
IS25LP080D_EAR_HIGHEST_SE
dev_flash_IS25LP080D.h, [181](#)
IS25LP080D_EAR_LOWEST_SEG
dev_flash_IS25LP080D.h, [181](#)
IS25LP080D_EAR_SECOND_SEG
dev_flash_IS25LP080D.h, [182](#)
IS25LP080D_EAR_THIRD_SEG
dev_flash_IS25LP080D.h, [182](#)
IS25LP080D_EVCR_DTRP
dev_flash_IS25LP080D.h, [182](#)
IS25LP080D_EVCR_DUAL
dev_flash_IS25LP080D.h, [182](#)
IS25LP080D_EVCR_ODS
dev_flash_IS25LP080D.h, [182](#)
IS25LP080D_EVCR_QUAD
dev_flash_IS25LP080D.h, [182](#)
IS25LP080D_EVCR_RH
dev_flash_IS25LP080D.h, [182](#)
IS25LP080D_FLASH_SIZE
dev_flash_IS25LP080D.h, [182](#)
IS25LP080D_FSR_ERERR
dev_flash_IS25LP080D.h, [183](#)
IS25LP080D_FSR_ERSUS
dev_flash_IS25LP080D.h, [183](#)
IS25LP080D_FSR_NBADDR
dev_flash_IS25LP080D.h, [183](#)
IS25LP080D_FSR_PGERR
dev_flash_IS25LP080D.h, [183](#)
IS25LP080D_FSR_PGSUS
dev_flash_IS25LP080D.h, [183](#)
IS25LP080D_FSR_PRERR
dev_flash_IS25LP080D.h, [183](#)
IS25LP080D_FSR_READY
dev_flash_IS25LP080D.h, [183](#)
IS25LP080D_NVCR_DTRP
dev_flash_IS25LP080D.h, [183](#)
IS25LP080D_NVCR_DUAL
dev_flash_IS25LP080D.h, [184](#)
IS25LP080D_NVCR_NB_DUMMY
dev_flash_IS25LP080D.h, [184](#)
IS25LP080D_NVCR_NBADDR

- dev_flash_IS25LP080D.h, [184](#)
- IS25LP080D_NVCR_ODS
 - dev_flash_IS25LP080D.h, [184](#)
- IS25LP080D_NVCR_QUAB
 - dev_flash_IS25LP080D.h, [184](#)
- IS25LP080D_NVCR_RH
 - dev_flash_IS25LP080D.h, [184](#)
- IS25LP080D_NVCR_SEGMENT
 - dev_flash_IS25LP080D.h, [184](#)
- IS25LP080D_NVCR_XIP
 - dev_flash_IS25LP080D.h, [184](#)
- IS25LP080D_PAGE_SIZE
 - dev_flash_IS25LP080D.h, [185](#)
- IS25LP080D_SECTOR_ERASE_MAX_TIME
 - dev_flash_IS25LP080D.h, [185](#)
- IS25LP080D_SECTOR_SIZE
 - dev_flash_IS25LP080D.h, [185](#)
- IS25LP080D_SR_QE
 - dev_flash_IS25LP080D.h, [185](#)
- IS25LP080D_SR_SRWREN
 - dev_flash_IS25LP080D.h, [185](#)
- IS25LP080D_SR_WIP
 - dev_flash_IS25LP080D.h, [185](#)
- IS25LP080D_SR_WREN
 - dev_flash_IS25LP080D.h, [185](#)
- IS25LP080D_SUBSECTOR_ERASE_MAX_TIME
 - dev_flash_IS25LP080D.h, [186](#)
- IS25LP080D_SUBSECTOR_SIZE
 - dev_flash_IS25LP080D.h, [186](#)
- IS25LP080D_VCR_NB_DUMMY
 - dev_flash_IS25LP080D.h, [186](#)
- IS25LP080D_VCR_WRAP
 - dev_flash_IS25LP080D.h, [186](#)
- IS25LP080D_VCR_XIP
 - dev_flash_IS25LP080D.h, [186](#)
- ImmediateRead
 - daisy::RingBuffer, [118](#)
 - daisy::RingBuffer< T, 0 >, [122](#)
- Increment
 - CONTROLS, [17](#)
- Init
 - CONTROLS, [17](#), [18](#)
 - daisy::AdcHandle, [68](#)
 - daisy::Color, [73](#)
 - daisy::DaisyPatch, [79](#)
 - daisy::DaisyPetal, [86](#)
 - daisy::DaisyPod, [92](#)
 - daisy::DaisySeed, [96](#)
 - daisy::GateIn, [112](#)
 - daisy::RingBuffer, [119](#)
 - daisy::RingBuffer< T, 0 >, [122](#)
 - daisy::SdmmcHandler, [124](#)
 - daisy::SpiHandle, [128](#)
 - daisy::UartHandler, [130](#)
 - daisy::WavPlayer, [141](#)
 - EXTERNAL, [31](#)
 - FEEDBACK, [23](#), [24](#)
 - ShiftRegister595, [126](#)
 - UsbHandle, [134](#)
- init
 - dsy_sai_handle, [105](#)
- InitBipolarCv
 - CONTROLS, [19](#)
- InitMux
 - daisy::AdcChannelConfig, [64](#)
- InitSingle
 - daisy::AdcChannelConfig, [64](#)
- KB_SW_SR_CLK_PIN
 - daisy_field.h, [151](#)
- KB_SW_SR_CS_PIN
 - daisy_field.h, [151](#)
- KB_SW_SR_D1_PIN
 - daisy_field.h, [151](#)
- KB_SW_SR_D2_PIN
 - daisy_field.h, [151](#)
- kMaxSr595DaisyChain
 - dev_sr_595.h, [201](#)
- kUartMaxBufferSize
 - per_uart.h, [235](#)
- keyboard_sr
 - daisy::daisy_field, [75](#)
- Knob
 - daisy::DaisyPetal, [83](#)
 - daisy::DaisyPod, [90](#)
- knob
 - daisy::DaisyPetal, [88](#)
- knob1
 - daisy::DaisyPod, [94](#)
- knob2
 - daisy::DaisyPod, [94](#)
- knobs
 - daisy::DaisyPod, [94](#)
 - daisy::daisy_field, [75](#)
- I
 - codec_frame_t, [70](#)
- LED_DRIVER_I2C
 - daisy_field.h, [151](#)
- LIBDAISY, [9](#)
- led1
 - daisy::DaisyPod, [94](#)
- led2
 - daisy::DaisyPod, [94](#)
- Listen
 - EXTERNAL, [32](#)
- LogBlockNbr
 - DSY_SD_CardInfoTypeDef, [107](#)
- LogBlockSize
 - DSY_SD_CardInfoTypeDef, [107](#)
- MIDI_IN_PIN
 - daisy_field.h, [151](#)
- MIDI_OUT_PIN
 - daisy_field.h, [151](#)
- MSD_ERROR_SD_NOT_PRESENT
 - util_bsp_sd_diskio.h, [242](#)

- MSD_ERROR
 - util_bsp_sd_diskio.h, [241](#)
- MSD_OK
 - util_bsp_sd_diskio.h, [242](#)
- MULTIPLE_IO_READ_ID_CMD
 - dev_flash_IS25LP064A.h, [171](#)
 - dev_flash_IS25LP080D.h, [186](#)
- MUX_ADC_PIN
 - daisy_field.h, [151](#)
- MUX_SEL_0_PIN
 - daisy_field.h, [152](#)
- MUX_SEL_1_PIN
 - daisy_field.h, [152](#)
- MUX_SEL_2_PIN
 - daisy_field.h, [152](#)
- midi
 - daisy::DaisyPatch, [81](#)
- MidiInputMode
 - EXTERNAL, [29](#)
- MidiMessageType
 - EXTERNAL, [29](#)
- MidiOutputMode
 - EXTERNAL, [29](#)
- mode
 - dsy_dac_handle, [100](#)
 - dsy_gpio, [101](#)
 - dsy_qspi_handle, [103](#)
- mux_channels_
 - daisy::AdcChannelConfig, [65](#)
- mux_pin_
 - daisy::AdcChannelConfig, [65](#)
- MuxPin
 - daisy::AdcChannelConfig, [63](#)
- name
 - daisy::WavFileInfo, [139](#)
- NbrChannels
 - WAV_FormatTypeDef, [138](#)
- note
 - EXTERNAL, [33](#)
- num_daisy chained
 - dsy_sr_4021_handle, [109](#)
- num_parallel
 - dsy_sr_4021_handle, [109](#)
- Open
 - daisy::WavPlayer, [141](#)
- OverSampling
 - daisy::AdcHandle, [66](#)
- Overwrite
 - daisy::RingBuffer, [119](#)
 - daisy::RingBuffer< T, 0 >, [122](#), [123](#)
- PAGE_PROG_4_BYTE_ADDR_CMD
 - dev_flash_IS25LP064A.h, [171](#)
 - dev_flash_IS25LP080D.h, [186](#)
- PAGE_PROG_CMD
 - dev_flash_IS25LP064A.h, [171](#)
 - dev_flash_IS25LP080D.h, [186](#)
- PERIPHERAL, [34](#)
- PROG_ERASE_RESUME_CMD
 - dev_flash_IS25LP064A.h, [171](#)
 - dev_flash_IS25LP080D.h, [187](#)
- PROG_ERASE_SUSPEND_CMD
 - dev_flash_IS25LP064A.h, [171](#)
 - dev_flash_IS25LP080D.h, [187](#)
- PROG_OTP_ARRAY_CMD
 - dev_flash_IS25LP064A.h, [171](#)
 - dev_flash_IS25LP080D.h, [187](#)
- Parameter
 - CONTROLS, [19](#)
- Parse
 - EXTERNAL, [32](#)
- per_adc.h
 - DSY_ADC_MAX_CHANNELS, [211](#)
 - DSY_ADC_H, [211](#)
- per_dac.h
 - dsy_dac_bitdepth, [212](#)
 - dsy_dac_channel, [212](#)
 - dsy_dac_init, [213](#)
 - dsy_dac_mode, [213](#)
 - dsy_dac_start, [213](#)
 - dsy_dac_write, [214](#)
- per_gpio.h
 - dsy_gpio_deinit, [216](#)
 - dsy_gpio_init, [216](#)
 - dsy_gpio_mode, [215](#)
 - dsy_gpio_pull, [215](#)
 - dsy_gpio_read, [216](#)
 - dsy_gpio_toggle, [216](#)
 - dsy_gpio_write, [217](#)
- per_i2c.h
 - DSY_I2C_H, [218](#)
 - dsy_i2c_init, [219](#)
 - dsy_i2c_periph, [218](#)
 - dsy_i2c_pin, [218](#)
 - dsy_i2c_speed, [219](#)
- per_qspi.h
 - DSY_MEMORY_ERROR, [220](#)
 - DSY_MEMORY_OK, [220](#)
 - DSY_QSPI_BSS, [220](#)
 - DSY_QSPI_DATA, [221](#)
 - DSY_QSPI_TEXT, [221](#)
 - DSY_QSPI, [220](#)
 - dsy_qspi_deinit, [222](#)
 - dsy_qspi_device, [221](#)
 - dsy_qspi_erase, [223](#)
 - dsy_qspi_erasesector, [223](#)
 - dsy_qspi_init, [223](#)
 - dsy_qspi_mode, [222](#)
 - dsy_qspi_pin, [222](#)
 - dsy_qspi_write, [224](#)
 - dsy_qspi_writepage, [224](#)
- per_sai.h
 - dsy_audio_bitdepth, [226](#)
 - dsy_audio_device, [226](#)
 - dsy_audio_dir, [226](#)

- dsy_audio_sai, [227](#)
- dsy_audio_samplerate, [227](#)
- dsy_audio_sync, [227](#)
- dsy_sai_init, [228](#)
- dsy_sai_init_from_handle, [228](#)
- dsy_sai_pin, [227](#)
- per_sdmmc.h
 - DSY_SD_ERROR, [230](#)
 - DSY_SD_OK, [230](#)
 - DSY_SDMMC_H, [230](#)
 - SdmmcBitWidth, [231](#)
 - SdmmcMode, [231](#)
 - SdmmcSpeed, [231](#)
- per_spi.h
 - SpiPeriph, [232](#)
 - SpiPin, [232](#)
- per_tim.h
 - dsy_tim_delay_ms, [233](#)
 - dsy_tim_delay_tick, [233](#)
 - dsy_tim_delay_us, [233](#)
 - dsy_tim_get_ms, [234](#)
 - dsy_tim_get_tick, [234](#)
 - dsy_tim_get_us, [234](#)
 - dsy_tim_init, [234](#)
 - dsy_tim_start, [235](#)
- per_uart.h
 - DSY_UART_H, [235](#)
 - kUartMaxBufferSize, [235](#)
- periph
 - dsy_i2c_handle, [102](#)
- pin
 - dsy_gpio, [101](#)
 - dsy_gpio_pin, [101](#)
- pin_
 - daisy::AdcChannelConfig, [65](#)
- pin_config
 - dsy_dac_handle, [100](#)
 - dsy_i2c_handle, [102](#)
 - dsy_qspi_handle, [103](#)
 - dsy_sdram_handle, [108](#)
 - dsy_sr_4021_handle, [109](#)
- Pins
 - FEEDBACK, [22](#)
 - ShiftRegister595, [126](#)
- Polarity
 - CONTROLS, [15](#)
- PollReceive
 - daisy::UartHandler, [130](#)
- PollTx
 - daisy::UartHandler, [130](#)
- PopEvent
 - EXTERNAL, [32](#)
- PopRx
 - daisy::UartHandler, [131](#)
- port
 - dsy_gpio_pin, [102](#)
- Prepare
 - daisy::WavPlayer, [141](#)
- PresetColor
 - daisy::Color, [72](#)
- Pressed
 - CONTROLS, [19](#)
- Process
 - CONTROLS, [19](#), [20](#)
- Pull
 - CONTROLS, [15](#)
- pull
 - dsy_gpio, [101](#)
- QUAD_IN_FAST_PROG_4_BYTE_ADDR_CMD
 - dev_flash_IS25LP064A.h, [171](#)
 - dev_flash_IS25LP080D.h, [187](#)
- QUAD_IN_FAST_PROG_CMD
 - dev_flash_IS25LP064A.h, [172](#)
 - dev_flash_IS25LP080D.h, [187](#)
- QUAD_INOUT_FAST_READ_4_BYTE_ADDR_CMD
 - dev_flash_IS25LP064A.h, [172](#)
 - dev_flash_IS25LP080D.h, [187](#)
- QUAD_INOUT_FAST_READ_CMD
 - dev_flash_IS25LP064A.h, [172](#)
 - dev_flash_IS25LP080D.h, [187](#)
- QUAD_INOUT_FAST_READ_DTR_CMD
 - dev_flash_IS25LP064A.h, [172](#)
 - dev_flash_IS25LP080D.h, [187](#)
- QUAD_OUT_FAST_READ_4_BYTE_ADDR_CMD
 - dev_flash_IS25LP064A.h, [172](#)
 - dev_flash_IS25LP080D.h, [188](#)
- QUAD_OUT_FAST_READ_CMD
 - dev_flash_IS25LP064A.h, [172](#)
 - dev_flash_IS25LP080D.h, [188](#)
- QUAD_OUT_FAST_READ_DTR_CMD
 - dev_flash_IS25LP064A.h, [172](#)
 - dev_flash_IS25LP080D.h, [188](#)
- qspi_handle
 - daisy::DaisySeed, [98](#)
- r
 - codec_frame_t, [70](#)
- RAM_AS4C16M16SA_H
 - dev_sdram.h, [195](#)
- READ_4_BYTE_ADDR_CMD
 - dev_flash_IS25LP064A.h, [172](#)
 - dev_flash_IS25LP080D.h, [188](#)
- READ_CMD
 - dev_flash_IS25LP064A.h, [173](#)
 - dev_flash_IS25LP080D.h, [188](#)
- READ_ENHANCED_VOL_CFG_REG_CMD
 - dev_flash_IS25LP064A.h, [173](#)
 - dev_flash_IS25LP080D.h, [188](#)
- READ_EXT_ADDR_REG_CMD
 - dev_flash_IS25LP064A.h, [173](#)
 - dev_flash_IS25LP080D.h, [188](#)
- READ_FLAG_STATUS_REG_CMD
 - dev_flash_IS25LP064A.h, [173](#)
 - dev_flash_IS25LP080D.h, [188](#)
- READ_ID_CMD2
 - dev_flash_IS25LP064A.h, [173](#)

- dev_flash_IS25LP080D.h, [189](#)
- READ_ID_CMD
 - dev_flash_IS25LP064A.h, [173](#)
 - dev_flash_IS25LP080D.h, [189](#)
- READ_LOCK_REG_CMD
 - dev_flash_IS25LP064A.h, [173](#)
 - dev_flash_IS25LP080D.h, [189](#)
- READ_NONVOL_CFG_REG_CMD
 - dev_flash_IS25LP064A.h, [173](#)
 - dev_flash_IS25LP080D.h, [189](#)
- READ_OTP_ARRAY_CMD
 - dev_flash_IS25LP064A.h, [174](#)
 - dev_flash_IS25LP080D.h, [189](#)
- READ_READ_PARAM_REG_CMD
 - dev_flash_IS25LP064A.h, [174](#)
 - dev_flash_IS25LP080D.h, [189](#)
- READ_SERIAL_FLASH_DISCO_PARAM_CMD
 - dev_flash_IS25LP064A.h, [174](#)
 - dev_flash_IS25LP080D.h, [189](#)
- READ_STATUS_REG_CMD
 - dev_flash_IS25LP064A.h, [174](#)
 - dev_flash_IS25LP080D.h, [189](#)
- RESET_ENABLE_CMD
 - dev_flash_IS25LP064A.h, [174](#)
 - dev_flash_IS25LP080D.h, [190](#)
- RESET_MEMORY_CMD
 - dev_flash_IS25LP064A.h, [174](#)
 - dev_flash_IS25LP080D.h, [190](#)
- raw_data
 - daisy::WavFileInfo, [139](#)
- Read
 - daisy::RingBuffer, [119](#)
 - daisy::RingBuffer< T, 0 >, [123](#)
- Readable
 - daisy::UartHandler, [131](#)
- readable
 - daisy::RingBuffer, [119](#)
 - daisy::RingBuffer< T, 0 >, [123](#)
- ReceiveCallback
 - UsbHandle, [133](#)
- Red
 - daisy::Color, [74](#)
- red
 - color, [71](#)
- RelCardAdd
 - DSY_SD_CardInfoTypeDef, [107](#)
- Restart
 - daisy::WavPlayer, [141](#)
- retSD
 - fatfs.h, [202](#)
- ring_led
 - daisy::DaisyPetal, [89](#)
- RingLed
 - daisy::DaisyPetal, [83](#)
- RisingEdge
 - CONTROLS, [20](#)
- RxActive
 - daisy::UartHandler, [131](#)
- S162F_SCALE
 - daisy.h, [144](#)
- s162f
 - daisy.h, [145](#)
- S242F_SCALE
 - daisy.h, [145](#)
- s242f
 - daisy.h, [146](#)
- S24SIGN
 - daisy.h, [145](#)
- SA_LED_DRIVER_H
 - dev_leddriver.h, [192](#)
- SAMPLE_RATE
 - daisy_field.h, [152](#)
- SD_DATATIMEOUT
 - util_bsp_sd_diskio.h, [242](#)
- SD_NOT_PRESENT
 - util_bsp_sd_diskio.h, [242](#)
- SD_PRESENT
 - util_bsp_sd_diskio.h, [242](#)
- SD_TRANSFER_BUSY
 - util_bsp_sd_diskio.h, [242](#)
- SD_TRANSFER_OK
 - util_bsp_sd_diskio.h, [242](#)
- SDFatFS
 - fatfs.h, [202](#)
- SDFile
 - fatfs.h, [202](#)
- SDPath
 - fatfs.h, [202](#)
- SECTOR_ERASE_4_BYTE_ADDR_CMD
 - dev_flash_IS25LP064A.h, [174](#)
 - dev_flash_IS25LP080D.h, [190](#)
- SECTOR_ERASE_CMD
 - dev_flash_IS25LP064A.h, [174](#)
 - dev_flash_IS25LP080D.h, [190](#)
- SR_4021_MAX_DAISSYCHAIN
 - dev_sr_4021.h, [198](#)
- SR_4021_MAX_PARALLEL
 - dev_sr_4021.h, [198](#)
- STM32_USB_OTG_DEVICE_LIBRARY, [61](#)
- SUBSECTOR_ERASE_4_BYTE_ADDR_CMD
 - dev_flash_IS25LP064A.h, [175](#)
 - dev_flash_IS25LP080D.h, [190](#)
- SUBSECTOR_ERASE_CMD
 - dev_flash_IS25LP064A.h, [175](#)
 - dev_flash_IS25LP080D.h, [190](#)
- SUBSECTOR_ERASE_QPI_CMD
 - dev_flash_IS25LP064A.h, [175](#)
 - dev_flash_IS25LP080D.h, [190](#)
- SW_1_PIN
 - daisy_field.h, [152](#)
- SW_2_PIN
 - daisy_field.h, [152](#)
- SW_3_PIN
 - daisy_field.h, [152](#)
- SYSTEM, [35](#)
- sa_audio_callback

- dev_codec_wm8731_frame.h, 160
- sai
 - dsy_audio_handle, 99
- sai1_pin_config
 - dsy_sai_handle, 105
- sai2_pin_config
 - dsy_sai_handle, 105
- sai_handle
 - daisy::DaisySeed, 98
- SampleRate
 - WAV_FormatTypeDef, 138
- samplerate
 - dsy_sai_handle, 105
- SdmmcBitWidth
 - per_sdmmc.h, 231
- SdmmcMode
 - per_sdmmc.h, 231
- SdmmcSpeed
 - per_sdmmc.h, 231
- sdram_handle
 - daisy::DaisySeed, 98
- seed
 - daisy::DaisyPatch, 81
 - daisy::DaisyPetal, 89
 - daisy::DaisyPod, 94
 - daisy::daisy_field, 75
- Set
 - FEEDBACK, 24, 25
 - ShiftRegister595, 127
- SetAudioBlockSize
 - daisy::DaisyPatch, 79
 - daisy::DaisyPetal, 86
 - daisy::DaisyPod, 92
 - daisy::DaisySeed, 96
- SetColor
 - FEEDBACK, 25
- SetCursor
 - FEEDBACK, 25
- SetFootswitchLed
 - daisy::DaisyPetal, 87
- SetLed
 - daisy::DaisySeed, 96
- SetLooping
 - daisy::WavPlayer, 142
- SetReceiveCallback
 - UsbHandle, 134, 135
- SetRingLed
 - daisy::DaisyPetal, 87
- SetTestPoint
 - daisy::DaisySeed, 96
- ShiftRegister595, 125
 - Init, 126
 - Pins, 126
 - Set, 127
 - Write, 127
- speed
 - daisy::SdmmcHandlerInit, 125
 - dsy_i2c_handle, 102
- SpiPeriph
 - per_spi.h, 232
- SpiPin
 - per_spi.h, 232
- src/daisy.h, 143
- src/daisy_core.h, 146
- src/daisy_field.h, 148
- src/daisy_patch.h, 155
- src/daisy_petal.h, 156
- src/daisy_pod.h, 156
- src/daisy_seed.h, 156
- src/dev_codec_ak4556.h, 157
- src/dev_codec_pcm3060.h, 157
- src/dev_codec_wm8731.h, 158
- src/dev_codec_wm8731_frame.h, 160
- src/dev_flash_IS25LP064A.h, 160
- src/dev_flash_IS25LP080D.h, 176
- src/dev_leddriver.h, 192
- src/dev_sdram.h, 194
- src/dev_sr_4021.h, 197
- src/dev_sr_595.h, 200
- src/fatfs.h, 201
- src/ffconf.h, 203
- src/hid_gatein.h, 210
- src/hid_wavplayer.h, 210
- src/per_adc.h, 211
- src/per_dac.h, 212
- src/per_gpio.h, 214
- src/per_i2c.h, 217
- src/per_qspi.h, 219
- src/per_sai.h, 225
- src/per_sdmmc.h, 230
- src/per_spi.h, 232
- src/per_tim.h, 233
- src/per_uart.h, 235
- src/sys_dma.h, 236
- src/sys_system.h, 236
- src/usbd_cdc_if.h, 238
- src/usbd_conf.h, 239
- src/usbd_desc.h, 240
- src/util_bsp_sd_diskio.h, 240
- src/util_color.h, 246
- src/util_hal_map.h, 247
- src/util_ringbuffer.h, 249
- src/util_unique_id.h, 249
- src/util_wav_format.h, 250
- Start
 - daisy::AdcHandle, 69
- StartAdc
 - daisy::DaisyPatch, 80
 - daisy::DaisyPetal, 87
 - daisy::DaisyPod, 93
- StartAudio
 - daisy::DaisyPatch, 80
 - daisy::DaisyPetal, 87
 - daisy::DaisyPod, 93
 - daisy::DaisySeed, 97
- StartReceive

- EXTERNAL, 32
- StartRx
 - daisy::UartHandler, 131
- state
 - dsy_sdram_handle, 108
- states
 - dsy_sr_4021_handle, 109
- Stop
 - daisy::AdcHandle, 69
- Stream
 - daisy::WavPlayer, 142
- SubChunk2Size
 - WAV_FormatTypeDef, 138
- SubChunk1ID
 - WAV_FormatTypeDef, 138
- SubChunk1Size
 - WAV_FormatTypeDef, 138
- SubChunk2ID
 - WAV_FormatTypeDef, 138
- Sw
 - daisy::DaisyPetal, 84
 - daisy::DaisyPod, 91
- Swallow
 - daisy::RingBuffer, 120
- switches
 - daisy::DaisyPetal, 89
 - daisy::daisy_field, 76
- sync_config
 - dsy_sai_handle, 105
- sys_dma.h
 - dsy_dma_init, 236
- sys_system.h
 - dsy_system_delay, 236
 - dsy_system_getnow, 237
 - dsy_system_init, 237
 - dsy_system_jumpto, 237
 - dsy_system_jumptoqspi, 237
- TimeHeldMs
 - CONTROLS, 20
- TransmitExternal
 - UsbHandle, 135
- TransmitInternal
 - UsbHandle, 136
- Trig
 - daisy::Gateln, 112
- Type
 - CONTROLS, 16
- type
 - EXTERNAL, 33
- USB_SIZ_STRING_SERIAL
 - USB_DESC_Exported_Constants, 54
- USB_CDC_IF_Exported_Defines, 40
- USB_CDC_IF_Exported_FunctionsPrototype, 44
 - CDC_Set_Rx_Callback_FS, 44
 - CDC_Transmit_FS, 44
 - CDC_Transmit_HS, 44
- USB_CDC_IF_Exported_Macros, 42
- USB_CDC_IF_Exported_Types, 41
 - CDC_ReceiveCallback, 41
- USB_CDC_IF_Exported_Variables, 43
 - USB_Interface_fops_FS, 43
 - USB_Interface_fops_HS, 43
- USB_CDC_IF, 39
- USB_CONF_Exported_Defines, 47
 - DEVICE_FS, 47
 - DEVICE_HS, 47
 - USB_DEBUG_LEVEL, 47
 - USB_LPM_ENABLED, 47
 - USB_MAX_NUM_CONFIGURATION, 48
 - USB_MAX_NUM_INTERFACES, 48
 - USB_MAX_STR_DESC_SIZ, 48
 - USB_SELF_POWERED, 48
 - USB_SUPPORT_USER_STRING, 48
- USB_CONF_Exported_FunctionsPrototype, 52
- USB_CONF_Exported_Macros, 49
 - USB_DbgLog, 49
 - USB_Delay, 49
 - USB_ErrLog, 49
 - USB_UsrLog, 50
 - USB_free, 50
 - USB_malloc, 50
 - USB_memcpy, 50
 - USB_memset, 50
- USB_CONF_Exported_Types, 51
- USB_CONF_Exported_Variables, 46
- USB_CONF, 45
- USB_DEBUG_LEVEL
 - USB_CONF_Exported_Defines, 47
- USB_DESC_Exported_Constants, 54
 - DEVICE_ID1, 54
 - DEVICE_ID2, 54
 - DEVICE_ID3, 54
 - USB_SIZ_STRING_SERIAL, 54
- USB_DESC_Exported_Defines, 55
- USB_DESC_Exported_FunctionsPrototype, 59
- USB_DESC_Exported_Macros, 57
- USB_DESC_Exported_TypesDefinitions, 56
- USB_DESC_Exported_Variables, 58
 - FS_Desc, 58
 - HS_Desc, 58
- USB_DESC, 53
- USB_DbgLog
 - USB_CONF_Exported_Macros, 49
- USB_Delay
 - USB_CONF_Exported_Macros, 49
- USB_ErrLog
 - USB_CONF_Exported_Macros, 49
- USB_Interface_fops_FS
 - USB_CDC_IF_Exported_Variables, 43
- USB_Interface_fops_HS
 - USB_CDC_IF_Exported_Variables, 43
- USB_LPM_ENABLED
 - USB_CONF_Exported_Defines, 47
- USB_MAX_NUM_CONFIGURATION
 - USB_CONF_Exported_Defines, 48

- USBD_MAX_NUM_INTERFACES
 - USBD_CONF_Exported_Defines, 48
- USBD_MAX_STR_DESC_SIZ
 - USBD_CONF_Exported_Defines, 48
- USBD_OTG_DRIVER, 62
- USBD_SELF_POWERED
 - USBD_CONF_Exported_Defines, 48
- USBD_SUPPORT_USER_STRING
 - USBD_CONF_Exported_Defines, 48
- USBD_UsrLog
 - USBD_CONF_Exported_Macros, 50
- USBD_free
 - USBD_CONF_Exported_Macros, 50
- USBD_malloc
 - USBD_CONF_Exported_Macros, 50
- USBD_memcpy
 - USBD_CONF_Exported_Macros, 50
- USBD_memset
 - USBD_CONF_Exported_Macros, 50
- UTILITY, 38
- Update
 - FEEDBACK, 26
- UpdateAnalogControls
 - daisy::DaisyPatch, 80
 - daisy::DaisyPetal, 88
 - daisy::DaisyPod, 93
- UpdateLeds
 - daisy::DaisyPetal, 88
 - daisy::DaisyPod, 93
- usb_handle
 - daisy::DaisySeed, 98
- UsbHandle, 132
 - Init, 134
 - ReceiveCallback, 133
 - SetReceiveCallback, 134, 135
 - TransmitExternal, 135
 - TransmitInternal, 136
 - UsbPeriph, 133
- UsbPeriph
 - UsbHandle, 133
- util_bsp_sd_diskio.h
 - BSP_SD_AbortCallback, 243
 - BSP_SD_CardInfo, 241
 - BSP_SD_Erase, 243
 - BSP_SD_GetCardInfo, 243
 - BSP_SD_GetCardState, 243
 - BSP_SD_ITConfig, 244
 - BSP_SD_Init, 244
 - BSP_SD_IsDetected, 244
 - BSP_SD_ReadBlocks, 244
 - BSP_SD_ReadBlocks_DMA, 245
 - BSP_SD_ReadCpltCallback, 245
 - BSP_SD_WriteBlocks, 245
 - BSP_SD_WriteBlocks_DMA, 246
 - BSP_SD_WriteCpltCallback, 246
 - DSY_BSP_SD_DISKIO_H, 241
 - MSD_ERROR_SD_NOT_PRESENT, 242
 - MSD_ERROR, 241
 - MSD_OK, 242
 - SD_DATATIMEOUT, 242
 - SD_NOT_PRESENT, 242
 - SD_PRESENT, 242
 - SD_TRANSFER_BUSY, 242
 - SD_TRANSFER_OK, 242
- util_color.h
 - DSY_COLOR_H, 247
- util_hal_map.h
 - dsy_hal_map_get_i2c, 247
 - dsy_hal_map_get_pin, 248
 - dsy_hal_map_get_port, 248
 - hi2c1, 248
 - hi2c2, 248
 - hi2c3, 249
 - hi2c4, 249
- util_unique_id.h
 - dsy_get_unique_id, 249
- Value
 - CONTROLS, 20, 21
- value
 - EXTERNAL, 33
- velocity
 - EXTERNAL, 33
- WAV_FILENAME_MAX
 - hid_wavplayer.h, 211
- WAV_FormatTypeDef, 136
 - AudioFormat, 137
 - BitPerSample, 137
 - BlockAlign, 137
 - ByteRate, 137
 - ChunkId, 137
 - FileFormat, 137
 - FileSize, 138
 - NbrChannels, 138
 - SampleRate, 138
 - SubCHunk2Size, 138
 - SubChunk1ID, 138
 - SubChunk1Size, 138
 - SubChunk2ID, 138
- WRITE_DISABLE_CMD
 - dev_flash_IS25LP064A.h, 175
 - dev_flash_IS25LP080D.h, 190
- WRITE_ENABLE_CMD
 - dev_flash_IS25LP064A.h, 175
 - dev_flash_IS25LP080D.h, 191
- WRITE_ENHANCED_VOL_CFG_REG_CMD
 - dev_flash_IS25LP064A.h, 175
 - dev_flash_IS25LP080D.h, 191
- WRITE_EXT_ADDR_REG_CMD
 - dev_flash_IS25LP064A.h, 175
 - dev_flash_IS25LP080D.h, 191
- WRITE_LOCK_REG_CMD
 - dev_flash_IS25LP064A.h, 175
 - dev_flash_IS25LP080D.h, 191
- WRITE_NONVOL_CFG_REG_CMD
 - dev_flash_IS25LP064A.h, 176

- dev_flash_IS25LP080D.h, [191](#)
- WRITE_READ_PARAM_REG_CMD
 - dev_flash_IS25LP064A.h, [176](#)
 - dev_flash_IS25LP080D.h, [191](#)
- WRITE_STATUS_REG_CMD
 - dev_flash_IS25LP064A.h, [176](#)
 - dev_flash_IS25LP080D.h, [191](#)
- writable
 - daisy::RingBuffer, [120](#)
 - daisy::RingBuffer< T, 0 >, [123](#)
- Write
 - daisy::RingBuffer, [120](#)
 - daisy::RingBuffer< T, 0 >, [123](#)
 - ShiftRegister595, [127](#)
- WriteChar
 - FEEDBACK, [26](#)
- WriteString
 - FEEDBACK, [26](#)