

Little-Wire

Generated by Doxygen 1.8.3.1

Thu May 2 2013 01:43:07

Contents

1	Introduction	1
2	Module Index	3
2.1	Modules	3
3	Module Documentation	5
3.1	General	5
3.1.1	Detailed Description	5
3.1.2	Function Documentation	5
3.1.2.1	customMessage	5
3.1.2.2	littleWire_connect	6
3.1.2.3	littleWire_error	6
3.1.2.4	littleWire_errorName	6
3.1.2.5	readFirmwareVersion	6
3.2	GPIO	8
3.2.1	Detailed Description	8
3.2.2	Function Documentation	8
3.2.2.1	digitalRead	8
3.2.2.2	digitalWrite	8
3.2.2.3	internalPullup	8
3.2.2.4	pinMode	9
3.3	ADC	10
3.3.1	Detailed Description	10
3.3.2	Function Documentation	10
3.3.2.1	analog_init	10
3.3.2.2	analogRead	10
3.4	PWM	11
3.4.1	Detailed Description	11
3.4.2	Function Documentation	11
3.4.2.1	pwm_init	11
3.4.2.2	pwm_stop	11
3.4.2.3	pwm_updateCompare	11

3.4.2.4	pwm_updatePrescaler	12
3.5	SPI	13
3.5.1	Detailed Description	13
3.5.2	Function Documentation	13
3.5.2.1	debugSpi	13
3.5.2.2	spi_init	13
3.5.2.3	spi_sendMessage	13
3.5.2.4	spi_updateDelay	14
3.6	I2C	15
3.6.1	Detailed Description	15
3.6.2	Function Documentation	15
3.6.2.1	i2c_init	15
3.6.2.2	i2c_read	15
3.6.2.3	i2c_start	15
3.6.2.4	i2c_updateDelay	16
3.6.2.5	i2c_write	16
3.7	Onewire	17
3.7.1	Detailed Description	17
3.7.2	Function Documentation	17
3.7.2.1	onewire_firstAddress	17
3.7.2.2	onewire_nextAddress	17
3.7.2.3	onewire_readBit	17
3.7.2.4	onewire_readByte	18
3.7.2.5	onewire_resetPulse	18
3.7.2.6	onewire_sendBit	18
3.7.2.7	onewire_writeByte	18
3.8	SOFT_PWM	20
3.8.1	Detailed Description	20
3.8.2	Function Documentation	20
3.8.2.1	softPWM_state	20
3.8.2.2	softPWM_write	20
3.9	Servo	21
3.9.1	Detailed Description	21
3.9.2	Function Documentation	21
3.9.2.1	servo_init	21
3.9.2.2	servo_updateLocation	21

Chapter 1

Introduction

Chapter 2

Module Index

2.1 Modules

Here is a list of all modules:

General	5
GPIO	8
ADC	10
PWM	11
SPI	13
I2C	15
Onewire	17
SOFT_PWM	20
Servo	21

Chapter 3

Module Documentation

3.1 General

General library functions.

Typedefs

- typedef usb_dev_handle **littleWire**

Functions

- littleWire * [littleWire_connect](#) ()
- unsigned char [readFirmwareVersion](#) (littleWire *lwHandle)
- int [customMessage](#) (littleWire *lwHandle, unsigned char *receiveBuffer, unsigned char command, unsigned char d1, unsigned char d2, unsigned char d3, unsigned char d4)
- int [littleWire_error](#) ()
- char * [littleWire_errorName](#) ()

3.1.1 Detailed Description

General library functions.

3.1.2 Function Documentation

3.1.2.1 int customMessage (littleWire * *lwHandle*, unsigned char * *receiveBuffer*, unsigned char *command*, unsigned char *d1*, unsigned char *d2*, unsigned char *d3*, unsigned char *d4*)

Sends a custom message to the device.

Useful when developing new features in the firmware.

Parameters

<i>receiveBuffer</i>	Returned data buffer
<i>command</i>	Firmware command
<i>d1</i>	data[0] for the command
<i>d2</i>	data[1] for the command
<i>d3</i>	data[2] for the command
<i>d4</i>	data[3] for the command

Returns

status

Definition at line 273 of file littleWire.c.

3.1.2.2 littleWire* littleWire_connect ()

Tries to connect to the device.

Parameters

(none)	
--------	--

Returns

littleWire pointer for healthy connection, NULL for a failed trial.

Definition at line 60 of file littleWire.c.

3.1.2.3 int littleWire_error ()

Returns the numeric value of the status of the last communication attempt

Parameters

(none)	
--------	--

Returns

Numeric value of the status of the last communication attempt

Definition at line 427 of file littleWire.c.

3.1.2.4 char* littleWire_errorName ()

Returns the string version of the last communication attempt status if there was an error

Parameters

(none)	
--------	--

Returns

String version of the last communication attempt status if there was an error

Definition at line 432 of file littleWire.c.

3.1.2.5 unsigned char readFirmwareVersion (littleWire * lwHandle)

Reads the firmware version of the Little Wire

Format: 0xXY => X: Primary version Y: Minor version

Parameters

(none)	
--------	--

Returns

Firmware version

Definition at line 70 of file littleWire.c.

3.2 GPIO

GPIO library functions with Arduino-like syntax.

Functions

- void [digitalWrite](#) (littleWire *lwHandle, unsigned char pin, unsigned char state)
- void [pinMode](#) (littleWire *lwHandle, unsigned char pin, unsigned char mode)
- unsigned char [digitalRead](#) (littleWire *lwHandle, unsigned char pin)
- void [internalPullup](#) (littleWire *lwHandle, unsigned char pin, unsigned char state)

3.2.1 Detailed Description

GPIO library functions with Arduino-like syntax.

3.2.2 Function Documentation

3.2.2.1 unsigned char digitalRead (littleWire * lwHandle, unsigned char pin)

Read pin value

Parameters

<i>lwHandle</i>	littleWire device pointer
<i>pin</i>	Pin name (PIN1 , PIN2 , PIN3 or PIN4)

Returns

Pin state (**HIGH** or **LOW**)

Definition at line 95 of file littleWire.c.

3.2.2.2 void digitalWrite (littleWire * lwHandle, unsigned char pin, unsigned char state)

Set pin value

Parameters

<i>lwHandle</i>	littleWire device pointer
<i>pin</i>	Pin name (PIN1 , PIN2 , PIN3 or PIN4)
<i>state</i>	Pin state (HIGH or LOW)

Returns

(none)

Definition at line 77 of file littleWire.c.

3.2.2.3 void internalPullup (littleWire * lwHandle, unsigned char pin, unsigned char state)

Sets the state of the internal pullup resistor.

Call this function after you assign the pin as an input.

Parameters

<i>lwHandle</i>	littleWire device pointer
<i>pin</i>	Pin name (PIN1 , PIN2 , PIN3 or PIN4) state (ENABLE or DISABLE)

Returns

(none)

Definition at line 102 of file littleWire.c.

3.2.2.4 void pinMode (littleWire * *lwHandle*, unsigned char *pin*, unsigned char *mode*)

Set pin as input/output

Parameters

<i>lwHandle</i>	littleWire device pointer
<i>pin</i>	Pin name (PIN1 , PIN2 , PIN3 or PIN4)
<i>mode</i>	Mode of pin (INPUT or OUTPUT)

Returns

(none)

Definition at line 86 of file littleWire.c.

3.3 ADC

Analog to digital converter functions.

Functions

- void [analog_init](#) (littleWire *lwHandle, unsigned char voltageRef)
- unsigned int [analogRead](#) (littleWire *lwHandle, unsigned char channel)

3.3.1 Detailed Description

Analog to digital converter functions.

3.3.2 Function Documentation

3.3.2.1 void analog_init (littleWire * *lwHandle*, unsigned char *voltageRef*)

Initialize the analog module. VREF_VCC is the standard voltage coming from the USB plug

Others are the Attiny's internal voltage references.

Parameters

<i>lwHandle</i>	littleWire device pointer
<i>voltageRef</i>	(VREF_VCC , VREF_110mV or VREF_2560mV)

Returns

(none)

Definition at line 111 of file littleWire.c.

3.3.2.2 unsigned int analogRead (littleWire * *lwHandle*, unsigned char *channel*)

Read analog voltage. Analog voltage reading from ADC_PIN3 isn't advised (it is a bit noisy) but supported. Use it at your own risk.

For more about internal temperature sensor, look at the Attiny85 datasheet.

Parameters

<i>lwHandle</i>	littleWire device pointer
<i>channel</i>	Source of ADC reading (ADC_PIN2 , ADC_PIN3 or ADC_TEMP_SENS)

Returns

10 bit ADC result

Definition at line 116 of file littleWire.c.

3.4 PWM

Pulse width modulation functions.

Functions

- void [pwm_init](#) (littleWire *lwHandle)
- void [pwm_stop](#) (littleWire *lwHandle)
- void [pwm_updateCompare](#) (littleWire *lwHandle, unsigned char channelA, unsigned char channelB)
- void [pwm_updatePrescaler](#) (littleWire *lwHandle, unsigned int value)

3.4.1 Detailed Description

Pulse width modulation functions.

3.4.2 Function Documentation

3.4.2.1 void pwm_init (littleWire * lwHandle)

Initialize the PWM module on the Little-Wire

Parameters

<i>lwHandle</i>	littleWire device pointer
-----------------	---------------------------

Returns

(none)

Definition at line 123 of file littleWire.c.

3.4.2.2 void pwm_stop (littleWire * lwHandle)

Stop the PWM module on the Little-Wire

Parameters

<i>lwHandle</i>	littleWire device pointer
-----------------	---------------------------

Returns

(none)

Definition at line 128 of file littleWire.c.

3.4.2.3 void pwm_updateCompare (littleWire * lwHandle, unsigned char channelA, unsigned char channelB)

Update the compare values of the PWM output pins. Resolution is 8 bit.

Parameters

<i>lwHandle</i>	littleWire device pointer
<i>channelA</i>	Compare value of PWMA pin
<i>channelB</i>	Compare value of PWMB pin

Returns

(none)

Definition at line 133 of file littleWire.c.

3.4.2.4 void pwm.updatePrescaler (littleWire * *lwHandle*, unsigned int *value*)

Update the prescaler of the PWM module. Adjust this value according to your need for speed in PWM output. Default is 1024. Lower prescale means higher frequency PWM output.

Parameters

<i>lwHandle</i>	littleWire device pointer
<i>value</i>	Presecaler value (1024 , 256 , 64 , 8 or 1)

Returns

(none)

Definition at line 138 of file littleWire.c.

3.5 SPI

Serial peripheral interface functions.

Functions

- void [spi_init](#) (littleWire *lwHandle)
- void [spi_sendMessage](#) (littleWire *lwHandle, unsigned char *sendBuffer, unsigned char *inputBuffer, unsigned char length, unsigned char mode)
- unsigned char [debugSpi](#) (littleWire *lwHandle, unsigned char message)
- void [spi_updateDelay](#) (littleWire *lwHandle, unsigned int duration)

3.5.1 Detailed Description

Serial peripheral interface functions.

3.5.2 Function Documentation

3.5.2.1 unsigned char debugSpi (littleWire * *lwHandle*, unsigned char *message*)

Send one byte SPI message over MOSI pin. Slightly slower than the actual one.

There isn't any chip select control involved. Useful for debug console app

Parameters

<i>lwHandle</i>	littleWire device pointer
<i>message</i>	Message to send

Returns

Received SPI message

Definition at line 176 of file littleWire.c.

3.5.2.2 void spi_init (littleWire * *lwHandle*)

Initialize the SPI module on the Little-Wire

Parameters

<i>lwHandle</i>	littleWire device pointer
-----------------	---------------------------

Returns

(none)

Definition at line 160 of file littleWire.c.

3.5.2.3 void spi_sendMessage (littleWire * *lwHandle*, unsigned char * *sendBuffer*, unsigned char * *inputBuffer*, unsigned char *length*, unsigned char *mode*)

Send SPI message(s). SPI Mode is 0.

Parameters

<i>lwHandle</i>	littleWire device pointer
<i>sendBuffer</i>	Message array to send
<i>inputBuffer</i>	Returned answer message
<i>length</i>	Message length - maximum 4
<i>mode</i>	AUTO_CS or MANUAL_CS

Returns

(none)

Definition at line 165 of file littleWire.c.

3.5.2.4 void spi_updateDelay (littleWire * lwHandle, unsigned int duration)

Change the SPI message frequency by adjusting delay duration. By default, Little-Wire sends the SPI messages with maximum speed.

If your hardware can't catch up with the speed, increase the duration value to lower the SPI speed.

Parameters

<i>lwHandle</i>	littleWire device pointer
<i>duration</i>	Amount of delay.

Returns

(none)

Definition at line 183 of file littleWire.c.

3.6 I2C

Inter IC communication functions.

Functions

- void [i2c_init](#) (littleWire *lwHandle)
- unsigned char [i2c_start](#) (littleWire *lwHandle, unsigned char address7bit, unsigned char direction)
- void [i2c_write](#) (littleWire *lwHandle, unsigned char *sendBuffer, unsigned char length, unsigned char end-WithStop)
- void [i2c_read](#) (littleWire *lwHandle, unsigned char *readBuffer, unsigned char length, unsigned char end-WithStop)
- void [i2c_updateDelay](#) (littleWire *lwHandle, unsigned int duration)

3.6.1 Detailed Description

Inter IC communication functions.

3.6.2 Function Documentation

3.6.2.1 void i2c_init (littleWire * lwHandle)

Initialize the I2C module on the Little-Wire

Returns

(none)

Definition at line 188 of file littleWire.c.

3.6.2.2 void i2c_read (littleWire * lwHandle, unsigned char * readBuffer, unsigned char length, unsigned char endWithStop)

Read byte(s) over i2c bus

Parameters

<i>lwHandle</i>	littleWire device pointer
<i>readBuffer</i>	Returned message array
<i>length</i>	Message length -> Max = 8
<i>endWithStop</i>	Should we send a STOP condition after this buffer? (END_WITH_STOP or NO_STOP)

Returns

(none)

Definition at line 209 of file littleWire.c.

3.6.2.3 unsigned char i2c_start (littleWire * lwHandle, unsigned char address7bit, unsigned char direction)

Start the i2c communication

Parameters

<i>lwHandle</i>	littleWire device pointer
<i>address</i>	7 bit slave address.
<i>direction</i>	(READ or WRITE)

Returns

1 if received ACK

Definition at line 193 of file littleWire.c.

3.6.2.4 void i2c.updateDelay (littleWire * *lwHandle*, unsigned int *duration*)

Update i2c signal delay amount. Tune if neccessary to fit your requirements.

Parameters

<i>lwHandle</i>	littleWire device pointer
<i>duration</i>	Delay amount

Returns

(none)

Definition at line 224 of file littleWire.c.

3.6.2.5 void i2c.write (littleWire * *lwHandle*, unsigned char * *sendBuffer*, unsigned char *length*, unsigned char *endWithStop*)

Send byte(s) over i2c bus

Parameters

<i>lwHandle</i>	littleWire device pointer
<i>sendBuffer</i>	Message array to send
<i>length</i>	Message length -> Max = 4
<i>endWithStop</i>	Should we send a STOP condition after this buffer? (END_WITH_STOP or NO_STOP)

Returns

(none)

Definition at line 204 of file littleWire.c.

3.7 Onewire

Onewire functions.

Functions

- void [onewire_sendBit](#) (littleWire *lwHandle, unsigned char bitValue)
- void [onewire_writeByte](#) (littleWire *lwHandle, unsigned char messageToSend)
- unsigned char [onewire_readByte](#) (littleWire *lwHandle)
- unsigned char [onewire_readBit](#) (littleWire *lwHandle)
- unsigned char [onewire_resetPulse](#) (littleWire *lwHandle)
- int [onewire_firstAddress](#) (littleWire *lwHandle)
- int [onewire_nextAddress](#) (littleWire *lwHandle)

3.7.1 Detailed Description

Onewire functions.

3.7.2 Function Documentation

3.7.2.1 int onewire_firstAddress (littleWire * lwHandle)

Start searching for device address on the onewire bus.

Read the 8 byte address from **ROM_NO** array

Parameters

<i>lwHandle</i>	littleWire device pointer
-----------------	---------------------------

Returns

Nonzero if any device found

Definition at line 415 of file littleWire.c.

3.7.2.2 int onewire_nextAddress (littleWire * lwHandle)

Try to find the next adress on the onewire bus.

Read the 8 byte address from **ROM_NO** array

Parameters

<i>lwHandle</i>	littleWire device pointer
-----------------	---------------------------

Returns

Nonzero if any new device found

Definition at line 296 of file littleWire.c.

3.7.2.3 unsigned char onewire_readBit (littleWire * lwHandle)

Read a single bit over onewire bus

Parameters

<i>lwHandle</i>	littleWire device pointer
-----------------	---------------------------

Returns

Read bit (**1** or **0**)

Definition at line 248 of file littleWire.c.

3.7.2.4 unsigned char onewire_readByte (littleWire * *lwHandle*)

Read a byte over onewire bus.

Parameters

<i>lwHandle</i>	littleWire device pointer
-----------------	---------------------------

Returns

Read byte

Definition at line 240 of file littleWire.c.

3.7.2.5 unsigned char onewire_resetPulse (littleWire * *lwHandle*)

Send a reset pulse over onewire bus

Parameters

<i>lwHandle</i>	littleWire device pointer
-----------------	---------------------------

Returns

Nonzero if any device presents on the bus

Definition at line 255 of file littleWire.c.

3.7.2.6 void onewire_sendBit (littleWire * *lwHandle*, unsigned char *bitValue*)

Send a single bit over onewire bus.

Parameters

<i>lwHandle</i>	littleWire device pointer
<i>bitValue</i>	1 or 0

Returns

(none)

Definition at line 229 of file littleWire.c.

3.7.2.7 void onewire_writeByte (littleWire * *lwHandle*, unsigned char *messageToSend*)

Send a byte over onewire bus.

Parameters

<i>lwHandle</i>	littleWire device pointer
<i>messageToSend</i>	Message to send

Returns

(none)

Definition at line 234 of file littleWire.c.

3.8 SOFT_PWM

Software PWM functions. Designed to be used with RGB LEDs.

Functions

- void [softPWM_state](#) (littleWire *lwHandle, unsigned char state)
- void [softPWM_write](#) (littleWire *lwHandle, unsigned char ch1, unsigned char ch2, unsigned char ch3)

3.8.1 Detailed Description

Software PWM functions. Designed to be used with RGB LEDs.

3.8.2 Function Documentation

3.8.2.1 void [softPWM_state](#) (littleWire * *lwHandle*, unsigned char *state*)

Sets the state of the softPWM module

Parameters

<i>lwHandle</i>	littleWire device pointer
<i>state</i>	State of the softPWM module (ENABLE or DISABLE)

Returns

(none)

Definition at line 263 of file littleWire.c.

3.8.2.2 void [softPWM_write](#) (littleWire * *lwHandle*, unsigned char *ch1*, unsigned char *ch2*, unsigned char *ch3*)

Updates the values of softPWM modules

Parameters

<i>lwHandle</i>	littleWire device pointer
<i>ch1</i>	Value of channel 1 - PIN4
<i>ch2</i>	Value of channel 2 - PIN1
<i>ch3</i>	Value of channel 3 - PIN2

Returns

(none)

Definition at line 268 of file littleWire.c.

3.9 Servo

Servo functions. Higher level access to PWM module.

Functions

- void [servo_init](#) (littleWire *lwHandle)
- void [servo_updateLocation](#) (littleWire *lwHandle, unsigned char locationChannelA, unsigned char locationChannelB)

3.9.1 Detailed Description

Servo functions. Higher level access to PWM module.

3.9.2 Function Documentation

3.9.2.1 void servo_init (littleWire * lwHandle)

Initialize the PWM module on the Little-Wire with the Servo special settings.

Parameters

<i>lwHandle</i>	littleWire device pointer
-----------------	---------------------------

Returns

(none)

Definition at line 39 of file littleWire_servo.c.

3.9.2.2 void servo_updateLocation (littleWire * lwHandle, unsigned char locationChannelA, unsigned char locationChannelB)

Update servo locations

Parameters

<i>lwHandle</i>	littleWire device pointer
<i>locationChannel-A</i>	Location of servo connected to channel A (in degrees)
<i>locationChannel-B</i>	Location of servo connected to channel B (in degrees)

Returns

(none)

Definition at line 52 of file littleWire_servo.c.

Index

- ADC, [10](#)
 - [analog_init](#), [10](#)
 - [analogRead](#), [10](#)
- [analog_init](#)
 - ADC, [10](#)
- [analogRead](#)
 - ADC, [10](#)
- [customMessage](#)
 - General, [5](#)
- [debugSpi](#)
 - SPI, [13](#)
- [digitalRead](#)
 - GPIO, [8](#)
- [digitalWrite](#)
 - GPIO, [8](#)
- GPIO, [8](#)
 - [digitalRead](#), [8](#)
 - [digitalWrite](#), [8](#)
 - [internalPullup](#), [8](#)
 - [pinMode](#), [9](#)
- General, [5](#)
 - [customMessage](#), [5](#)
 - [littleWire_connect](#), [6](#)
 - [littleWire_error](#), [6](#)
 - [littleWire_errorName](#), [6](#)
 - [readFirmwareVersion](#), [6](#)
- I2C, [15](#)
 - [i2c_init](#), [15](#)
 - [i2c_read](#), [15](#)
 - [i2c_start](#), [15](#)
 - [i2c_updateDelay](#), [16](#)
 - [i2c_write](#), [16](#)
- [i2c_init](#)
 - I2C, [15](#)
- [i2c_read](#)
 - I2C, [15](#)
- [i2c_start](#)
 - I2C, [15](#)
- [i2c_updateDelay](#)
 - I2C, [16](#)
- [i2c_write](#)
 - I2C, [16](#)
- [internalPullup](#)
 - GPIO, [8](#)
- [littleWire_connect](#)
 - General, [6](#)
- [littleWire_error](#)
 - General, [6](#)
- [littleWire_errorName](#)
 - General, [6](#)
- Onewire, [17](#)
 - [onewire_firstAddress](#), [17](#)
 - [onewire_nextAddress](#), [17](#)
 - [onewire_readBit](#), [17](#)
 - [onewire_readByte](#), [18](#)
 - [onewire_resetPulse](#), [18](#)
 - [onewire_sendBit](#), [18](#)
 - [onewire_writeByte](#), [18](#)
- [onewire_firstAddress](#)
 - Onewire, [17](#)
- [onewire_nextAddress](#)
 - Onewire, [17](#)
- [onewire_readBit](#)
 - Onewire, [17](#)
- [onewire_readByte](#)
 - Onewire, [18](#)
- [onewire_resetPulse](#)
 - Onewire, [18](#)
- [onewire_sendBit](#)
 - Onewire, [18](#)
- [onewire_writeByte](#)
 - Onewire, [18](#)
- PWM, [11](#)
 - [pwm_init](#), [11](#)
 - [pwm_stop](#), [11](#)
 - [pwm_updateCompare](#), [11](#)
 - [pwm_updatePrescaler](#), [12](#)
- [pinMode](#)
 - GPIO, [9](#)
- [pwm_init](#)
 - PWM, [11](#)
- [pwm_stop](#)
 - PWM, [11](#)
- [pwm_updateCompare](#)
 - PWM, [11](#)
- [pwm_updatePrescaler](#)
 - PWM, [12](#)
- [readFirmwareVersion](#)
 - General, [6](#)
- SOFT_PWM, [20](#)
 - [softPWM_state](#), [20](#)
 - [softPWM_write](#), [20](#)

- SPI, [13](#)
 - debugSpi, [13](#)
 - spi_init, [13](#)
 - spi_sendMessage, [13](#)
 - spi_updateDelay, [14](#)
- Servo, [21](#)
 - servo_init, [21](#)
 - servo_updateLocation, [21](#)
- servo_init
 - Servo, [21](#)
- servo_updateLocation
 - Servo, [21](#)
- softPWM_state
 - SOFT_PWM, [20](#)
- softPWM_write
 - SOFT_PWM, [20](#)
- spi_init
 - SPI, [13](#)
- spi_sendMessage
 - SPI, [13](#)
- spi_updateDelay
 - SPI, [14](#)