

DE10-Lite Arduino IP

Version 2.1
2/14/2021 8:37:00 AM

Table of Contents

File Index	1
File Documentation.....	1
HAL/inc/DE10_Lite_Arduino_Driver.h.....	1
Index	8

File Index

File List

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

HAL/inc/DE10_Lite_Arduino_Driver.h 1

File Documentation

HAL/inc/DE10_Lite_Arduino_Driver.h File Reference

```
#include <altera_avalon_i2c.h>
#include <stdint.h>
```

Macros

- **#define ANALOG_VOLTAGE_REFERENCE** 2.5f
Voltage discreet reference to floating point value factor.
- **#define ACCELEROMETER_ADDRESS** 0x53
Accelerometer I2C address.
- **#define D0** 0
- **#define D1** 1
- **#define D2** 2
- **#define D3** 3
- **#define D4** 4
- **#define D5** 5
- **#define D6** 6
- **#define D7** 7
- **#define D8** 8
- **#define D9** 9
- **#define D10** 10
- **#define D11** 11
- **#define D12** 12
- **#define D13** 13
- **#define D14** 14
- **#define D15** 15
- **#define A0** 0
- **#define A1** 1
- **#define A2** 2
- **#define A3** 3
- **#define A4** 4
- **#define A5** 5

- `#define PIN_INPUT 0`
- `#define PIN_OUTPUT 1`

- `#define SIGNAL_LOW 0`
- `#define SIGNAL_HIGH 1`

Functions

- `void arduino_uart_enable (uint8_t value)`
Toggle Arduino pin D0 and D1 as UART.

- `void arduino_spi_enable (uint8_t value)`
Toggle Arduino pin D10-D13 as SPI.

- `void arduino_pin_mode (uint8_t pin, uint8_t mode)`
Set Arduino pin to output or input.

- `void arduino_digital_write (uint8_t pin, uint8_t value)`
Write to Arduino pin.

- `uint8_t arduino_digital_read (uint8_t pin)`
Read Arduino pin.

- `void arduino_analog_init (uint32_t sequencer_base, uint32_t sample_store_base)`
Start ADC in continuous mode.

- `uint32_t arduino_analog_read (uint8_t pin)`
Read analog pin, DE10-Lite accepts up to 5.0v as input. Downscaled internally to max 2.5v. See voltage read function below for info on how to use values.

- `float arduino_analog_read_voltage (uint8_t pin)`
Read analog pin and convert to voltage level.

- `uint8_t accelerometer_open_dev (void)`
Open connection to accelerometer.

- `uint8_t accelerometer_init (void)`
Initialize accelerometer. It must have been opened first.

- `uint8_t accelerometer_read (uint32_t *accelerometer_x, uint32_t *accelerometer_y, uint32_t *accelerometer_z)`
*Read current accelerometer values. It must have been opened and initialized first. This function provides backwards compatibility with version 2.0 and earlier. Consider using **accelerometer_receive()** instead.*

- `uint8_t accelerometer_receive (int16_t *accelerometer_x, int16_t *accelerometer_y, int16_t *accelerometer_z)`
Receive current accelerometer values. It must have been opened and initialized first.

- **ALT_AVALON_I2C_STATUS_CODE accelerometer_status** (void)
Read accelerometer status code from last request. Useful to determine why a request failed.
- **uint8_t i2c_open_dev** (void)
Open connection to I2C.
- **void i2c_set_slave_address** (uint32_t slaveAddress)
Set the address the I2C device should be using. It must have been opened first.
- **uint8_t i2c_transmit** (uint8_t numBytes, uint8_t *data)
Send a byte buffer to the I2C device.
- **uint8_t i2c_receive** (uint8_t numBytes, uint8_t *data)
Receive a byte buffer from the I2C device.
- **ALT_AVALON_I2C_STATUS_CODE i2c_status** (void)
Read I2C status code from last request. Useful to determine why a request failed.
- **void i2c_set_normal_mode** (void)
Set I2C to use normal mode - 100 Kbps.
- **void i2c_set_fast_mode** (void)
Set I2C to use fast mode - 400 Kbps.
- **int spi_command** (uint32_t write_length, const uint8_t *wdata, uint32_t read_length, uint8_t *read_data)
This function writes a data buffer of arbitrary length to the mosi port, and then reads back an arbitrary amount of data from the miso port.
- **void spi_tx** (uint8_t data)
Fast function to send a single byte of data over SPI.

Detailed Description

Interface defining functions for accessing Arduino devices

Author

Linus Eriksson
Jesper Dahlbäck

Version

2.1

Date

2017-2021

Copyright

AGSTU AB

Macro Definition Documentation

#define A0 0

Analog pin indicies

#define D0 0

Digital pin indicies

#define PIN_INPUT 0

Pin modes

#define SIGNAL_LOW 0

Signal modes

Function Documentation

uint8_t accelerometer_init (void)

Initialize accelerometer. It must have been opened first.

Returns

1 if successful, otherwise 0.

uint8_t accelerometer_open_dev (void)

Open connection to accelerometer.

Returns

1 if successful, otherwise 0.

**uint8_t accelerometer_read (uint32_t * *accelerometer_x*, uint32_t * *accelerometer_y*,
uint32_t * *accelerometer_z*)**

Read current accelerometer values. It must have been opened and initialized first. This function provides backwards compatibility with version 2.0 and earlier. Consider using **accelerometer_receive()** instead.

Parameters

<i>accelerometer_x</i>	Pointer to unsigned 32-bit integer used for storing x value.
<i>accelerometer_y</i>	Pointer to unsigned 32-bit integer used for storing y value.
<i>accelerometer_z</i>	Pointer to unsigned 32-bit integer used for storing z value.

Returns

1 if successful, otherwise 0.

uint8_t accelerometer_receive (int16_t * *accelerometer_x*, int16_t * *accelerometer_y*, int16_t * *accelerometer_z*)

Receive current accelerometer values. It must have been opened and initialized first.

Parameters

<i>accelerometer_x</i>	Pointer to 16-bit integer used for storing x value.
<i>accelerometer_y</i>	Pointer to 16-bit integer used for storing y value.
<i>accelerometer_z</i>	Pointer to 16-bit integer used for storing z value.

Returns

1 if successful, otherwise 0.

ALT_AVALON_I2C_STATUS_CODE accelerometer_status (void)

Read accelerometer status code from last request. Useful to determine why a request failed.

Returns

Altera specific status code, see altera_avalon_i2c.h for more information.

void arduino_analog_init (uint32_t *sequencer_base*, uint32_t *sample_store_base*)

Start ADC in continuous mode.

Parameters

<i>sequencer_base</i>	Base address for sequencer, usually found in system.h.
<i>sample_store_base</i>	Base address for sample store, usually found in system.h.

uint32_t arduino_analog_read (uint8_t *pin*)

Read analog pin, DE10-Lite accepts up to 5.0v as input. Downscaled internally to max 2.5v. See voltage read function below for info on how to use values.

Parameters

<i>pin</i>	Index of pin to read from, will be shifted internally.
------------	--

Returns

Discreet voltage reference value.

float arduino_analog_read_voltage (uint8_t *pin*)

Read analog pin and convert to voltage level.

Parameters

<i>pin</i>	Index of pin to read from, will be shifted internally.
------------	--

Returns

Floating point voltage value.

uint8_t arduino_digital_read (uint8_t *pin*)

Read Arduino pin.

Parameters

<i>pin</i>	Index of pin to read from, will be bitshifted internally.
------------	---

Returns

current 1-bit value on pin.

void arduino_digital_write (uint8_t *pin*, uint8_t *value*)

Write to Arduino pin.

Parameters

<i>pin</i>	Index of pin to write to, will be bitshifted internally.
<i>value</i>	1-bit value to write to the pin.

void arduino_pin_mode (uint8_t *pin*, uint8_t *mode*)

Set Arduino pin to output or input.

Parameters

<i>pin</i>	Index of pin to manipulate, will be bitshifted internally.
<i>mode</i>	If LSB is 1, pin will be set to output mode. For readability author suggest using the defines PIN_OUTPUT and PIN_INPUT.

void arduino_spi_enable (uint8_t *value*)

Toggle Arduino pin D10-D13 as SPI.

Parameters

<i>value</i>	if LSB is 1, SPI is enabled, otherwise disabled.
--------------	--

void arduino_uart_enable (uint8_t *value*)

Toggle Arduino pin D0 and D1 as UART.

Parameters

<i>value</i>	if LSB is 1, UART is enabled, otherwise disabled .
--------------	--

uint8_t i2c_open_dev (void)

Open connection to I2C.

Returns

1 if successful, otherwise 0.

uint8_t i2c_receive (uint8_t *numBytes*, uint8_t * *data*)

Receive a byte buffer from the I2C device.

Parameters

<i>numBytes</i>	Number of bytes in buffer.
<i>data</i>	Pointer to first byte in buffer.

void i2c_set_slave_address (uint32_t *slaveAddress*)

Set the address the I2C device should be using. It must have been opened first.

Parameters

<i>slaveAddress</i>	Address to set.
---------------------	-----------------

ALT_AVALON_I2C_STATUS_CODE i2c_status (void)

Read I2C status code from last request. Useful to determine why a request failed.

Returns

Altera specific status code, see altera_avalon_i2c.h for more information.

uint8_t i2c_transmit (uint8_t *numBytes*, uint8_t * *data*)

Send a byte buffer to the I2C device.

Parameters

<i>numBytes</i>	Number of bytes in buffer.
<i>data</i>	Pointer to first byte in buffer.

int spi_command (uint32_t *write_length*, const uint8_t * *wdata*, uint32_t *read_length*, uint8_t * *read_data*)

This function writes a data buffer of arbitrary length to the mosi port, and then reads back an arbitrary amount of data from the miso port.

Parameters

<i>write_length</i>	Number of bytes in <i>write_data</i> buffer.
<i>wdata</i>	Pointer to first byte in buffer to write.
<i>read_length</i>	Number of bytes in <i>read_data</i> buffer.
<i>read_data</i>	Pointer to first byte in buffer to read.

Returns

Number of bytes read into *read_data* buffer.

void spi_tx (uint8_t *data*)

Fast function to send a single byte of data over SPI.

Parameters

<i>data</i>	Data to send.
-------------	---------------

Index

A0

DE10_Lite_Arduino_Driver.h, 4

accelerometer_init

DE10_Lite_Arduino_Driver.h, 4

accelerometer_open_dev

DE10_Lite_Arduino_Driver.h, 4

accelerometer_read

DE10_Lite_Arduino_Driver.h, 4

accelerometer_receive

DE10_Lite_Arduino_Driver.h, 5

accelerometer_status

DE10_Lite_Arduino_Driver.h, 5

arduino_analog_init

DE10_Lite_Arduino_Driver.h, 5

arduino_analog_read

DE10_Lite_Arduino_Driver.h, 5

arduino_analog_read_voltage

DE10_Lite_Arduino_Driver.h, 5

arduino_digital_read

DE10_Lite_Arduino_Driver.h, 6

arduino_digital_write

DE10_Lite_Arduino_Driver.h, 6

arduino_pin_mode

DE10_Lite_Arduino_Driver.h, 6

arduino_spi_enable

DE10_Lite_Arduino_Driver.h, 6

arduino_uart_enable

DE10_Lite_Arduino_Driver.h, 6

D0

DE10_Lite_Arduino_Driver.h, 4

DE10_Lite_Arduino_Driver.h

A0, 4

accelerometer_init, 4

accelerometer_open_dev, 4

accelerometer_read, 4

accelerometer_receive, 5

accelerometer_status, 5

arduino_analog_init, 5

arduino_analog_read, 5

arduino_analog_read_voltage, 5

arduino_digital_read, 6

arduino_digital_write, 6

arduino_pin_mode, 6

arduino_spi_enable, 6

arduino_uart_enable, 6

D0, 4

i2c_open_dev, 6

i2c_receive, 7

i2c_set_slave_address, 7

i2c_status, 7

i2c_transmit, 7

PIN_INPUT, 4

SIGNAL_LOW, 4

spi_command, 7

spi_tx, 7

HAL/inc/DE10_Lite_Arduino_Driver.h, 1

i2c_open_dev

DE10_Lite_Arduino_Driver.h, 6

i2c_receive

DE10_Lite_Arduino_Driver.h, 7

i2c_set_slave_address

DE10_Lite_Arduino_Driver.h, 7

i2c_status

DE10_Lite_Arduino_Driver.h, 7

i2c_transmit

DE10_Lite_Arduino_Driver.h, 7

PIN_INPUT

DE10_Lite_Arduino_Driver.h, 4

SIGNAL_LOW

DE10_Lite_Arduino_Driver.h, 4

spi_command

DE10_Lite_Arduino_Driver.h, 7

spi_tx

DE10_Lite_Arduino_Driver.h, 7