SPI library for AVR uC

Version : 1.0

Generated by Doxygen 1.7.4

Tue Jul 26 2011 21:20:35

Contents

1 Todo List

File spi.h Find a way to use multiple peripherals (currently only support one slave)

Functions for slave mode

2 File Documentation

2.1 spi.h File Reference

Public SPI functions for AVR uC.

```
#include <avr/io.h>
#include <stdbool.h>
#include "utils/utils.h"
```

Defines

Bit order, Mode and clock divider

- #define spi_MSB 0

  Most Significant Bit mode.
- #define spi_LSB !spi_MSB

  Least Significant Bit mode.
- #define spi_MODE0 0

Mode 0 CPOL=0, CPHA=0.

- #define spi_MODE1 1

  Mode 1 CPOL=0, CPHA=1.
- #define spi_MODE2 2

  Mode 2 CPOL=1, CPHA=0.
- #define spi_MODE3 3

  Mode 3 CPOL=1, CPHA=1.
- #define spi_CLOCK_2 2

  Clock/2.
- #define spi_CLOCK_4 4

  Clock/4.
- #define spi_CLOCK_8 8

  Clock/8.
- #define spi_CLOCK_16 16

  Clock/8.
- #define spi_CLOCK_32 32

  Clock/32.
- #define spi_CLOCK_64 64

  Clock/64.
- #define spi_CLOCK_128 128

  Clock/128.

Functions

- void spi_MasterInit (uint8_t data_order, uint8_t mode, uint8_t clock_rate, uint8_t interrupt_enable)

  Initialize the SPI registers and the SlaveSelect pin.
- uint8_t spi_MasterTransmit (uint8_t data)

  Send a byte and return the received byte.
- void spi_SlaveSelect (uint8_t action)

  Enable or disable the SlaveSelect pin.

2.1.1 Detailed Description

Public SPI functions for AVR uC.

2.1.2 License

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: Redistributions of source code

2.1.3 Specifications

Low-level functions to interface a peripheral with an AVR via the SPI protocol

2.1.3.1 Language

- C (c99)

2.1.3.2 Target

- ATmega48
- ATmega88
- ATmega168 (Tested)
- ATmega328 (Tested)

2.1.4 More informations

2.1.5 Datasheet

-ATmega168, section 19

- http://www.atmel.com/dyn/resources/prod_documents/doc8271.pdf

Author

Patrice Nadeau
email : patricen@telwarwick.net
www : http://nadeaup.homeip.net:8080/

Note

The following SPI port and pins are defined at compile time depending of the AVR used

- **spi_PORT**
- **spi_MOSI**
- **spi_SCK**
- **spi_SS**

Todo

Find a way to use multiple peripherals (currently only support one slave)
Functions for slave mode

2.1.6 Function Documentation

2.1.6.1 void spi_MasterInit (uint8_t *data_order*, uint8_t *mode*, uint8_t *clock_rate*, uint8_t *interrupt_enable*)

Initialize the SPI registers and the SlaveSelect pin.

Parameters

in	*data_order*	Possible values : • spi_MSB • spi_LSB
in	*mode*	Possible values : • spi_MODE0 • spi_MODE1 • spi_MODE2 • spi_MODE3

in	*clock_rate*	Possible values : • spi_CLOCK_2 • spi_CLOCK_4 • spi_CLOCK_8 • spi_CLOCK_16 • spi_CLOCK_32 • spi_CLOCK_64 • spi_CLOCK_128
in	*interrupt_enable*	Possible values : • true • false

Example

Activate the SPI interface in MSB mode0 with a clock/16 and interrupt disabled

```
spi_MasterInit(spi_MSB, spi_MODE0, spi_CLOCK_16, false);
```

2.1.6.2 uint8_t spi_MasterTransmit (uint8_t *data*)

Send a byte and return the received byte.

Parameters

in	*data*	A byte to transmit

Returns

The value received (if any)

Example

Transmit a value and read the result

```
uint8_t value;
spi_SlaveSelect(true);
value = spi_MasterTransmit(23);
spi_SlaveSelect(false);
```

2.1.6.3 void spi_SlaveSelect (uint8_t *action*)

Enable or disable the SlaveSelect pin.

Parameters

in	*action*	Possible values : • true • false

Example

Transmit a value and read the result

```
uint8_t value;
spi_SlaveSelect(true);
value = spi_MasterTransmit(23);
spi_SlaveSelect(false);
```

Note

Active low

Index