chipKIT™ SFSPI Library Reference Manual

Revision: December 14, 2011

1300 NE Henley Court, Suite 3
Pullman, WA 99163
(509) 334 6306 Voice | (509) 334 6300 Fax

Introduction

The Digilent SFSPI library provides a programming interface for an SPI serial flash ROM. This document provides an overview of the operation of this library and describes the functions that make up its programming interface.

Overview

The targeted serial flash memory uses either ST Microelectronics M25P16 serial flash ROM integrated circuit or M25P128 serial flash ROM integrated circuit. The highest memory address is 0x1FFFFF for M25P16 and 0xFFFFFF for M25P128. The memory is accessed through SPI interface. The memory must be erased before new data can be written.

Library Operation

Library Interface

The header file SFSPI.h defines the interfaces to the SFSPI library. The library is accessed via the methods and constants defined for the SFSPI object class. To instantiate a SFSPI object; simply include the library and instantiate a SFSPI object (e.g. mySF, or whatever name you want).

SFSPI Initialization

The SFSPI is accessed by the SPI interface. Before making calls to any other library functions, SFSPI.begin() must be called in order to setup SPI.

SFSPI Library Functions

Initialization Functions

void begin (void)

Parameters:
None

This function turns on SPI. This function must be called before any other functions in the library are called.

Procedure Functions

void getDeviceID (void)

Parameters:
None

This function returns the device ID.

uint8_t normalRead(int address)

Parameters:

address	- A 3 byte address to read from

This function reads a byte of data from the address specified.

uint8_t fastRead(int address)

Parameters:

address	- A 3 byte address to read from

This function reads a byte of data from the address specified at a faster speed

void writePage (uint8_t writeData, int address)

Parameters:

address	- A 3 byte address to write to
writeData	- A byte of data to write

This function writes a byte of data to the address specified.

void sectorErase (int address)

Parameters:
Address - A 3 byte address in the sector to be erased

This function erases the sector that the address is in. A sector starts from 0xnn0000 to 0xnnFFFF

(nn starts from 00 to the highest address allowed).

void bulkErase (void)

Parameters:
None

This function erases the entire memory.

void writeSR(uint8_t setReg)

Parameters:
setReg - the value to set the register with

This function writes to the status register.

The structure of the status register is shown as the table below:

B0							B7
Status Register Write Protect (SRWD)	0	0	Block Protect Bits 2 (BP2)	Block Protect Bits 1 (BP1)	Block Protect Bits 0 (BP0)	Write Enable Latch (WEL)	Write In Progress (WIP)

It can be use to protect certain sectors against write/erase with the corresponding values shown in the table below:

Device ID*	**0x16**	**0x128**
setReg	**Protected Area**	**Protected Area**
0x00	None	None
0x04	Sector 31	Sector 63
0x08	30 and 31	62 to 63
0x0C	28 to 31	60 to 63
0x10	24 to 31	56 to 63
0x14	16 to 31	48 to 63
0x18	0 to 31	32 to 63
0x1C	0 to 31	0-63