PmodSF™
Library Reference Manual

Revision: March 1, 2012

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

Introduction

The PmodSF library is a library with functions that are use to program a PmodSF through SPI. PmodSF is available with either a 16Mbit Serial Flash or a 128Mbit Serial Flash. The 16Mbit Serial flash uses Numonyx M25P16 and the 128Mbit Serial Flash uses Numonyx M25P128. Some basic features are as follow:

- The maximum clock rate is 50MHz for M25P16 and 40MHz for M25P128.
- The highest address is 0x1FFFFF for M25P16 and 0xFFFFFF for M25P128.
- A sector starts from 0xnn0000 to 0xnnFFFF. (nn starts from 00 to the highest address allowed)
- Both chips have an 8 bit status register. The bits on the status register are shown in Table1.

Table 1

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)

For further information please refer to the datasheet for the specific chip.

Description of calls

Utility calls

BOOL WriteEnable(spiHIF * spiH)
Parameters
spiH - Pointer to spi port handle

Return Values
Returns true if success, false otherwise.

Description
This function enables the device to write or erase by setting the WEL bit on the status register. This funciton needs to be, and is, called at the beginning of each write and erase function.

BYTE ReadSR(spiHIF * spiH)
Parameters
spiH - Pointer to spi port handle

Return Values
Returns status register

Description
This function returns the status register.

BOOL WaitForWrite(spiHIF * spiH)
Parameters
spiH - Pointer to spi port handle

Return Values
Returns true if success, false otherwise.

Description
This function waits for memory to finish with a Write or Erase cycle by polling the WIP bit on the status register. Due to the nature of the device, This function is called at the beginning of each function to prevent the function from being ignored.

Procedure calls

int GetDeviceInfo(spiHIF * spiH, int* maxAddress)

Parameters

spiH - Pointer to spi port handle
maxAddress - Pointer to store maximum address for the device

Return Values

Returns Chip Model
- 0x16 for M25P16
- 0x128 for M25P128
- 0 for anything else

Description

This function returns the model and the highest address of the chip. Ideally this function should be called before any function, just so the user has a clear idea of which chip is on the board.

BOOL NormalRead(spiHIF * spiH, BYTE* readData, int address, DWORD byteNum)

Parameters

spiH - Pointer to spi port handle
address - A 3 byte address to read from
readData - Pointer to an array to store data read
byteNum - Number of byte desire to read

Return Values

Returns true if success, false otherwise.

Description

This function reads byteNum of data from the address specified at a normal speed. The address is automatically incremented to the next higher address after each byte of data is shifted out. When the highest address is reached, it continues to read from 0x000000.

BOOL FastRead(spiHIF * spiH, BYTE* readData, int address, DWORD byteNum)

Parameters

spiH - Pointer to spi port handle
address - A 3 byte address to read from
readData - Pointer to an array to store data read
byteNum - Number of byte desire to read

Return Values

Returns true if success, false otherwise.

Description

This function reads byteNum of data from the address specified at a higher speed. The address is automatically incremented to the next higher address after each byte of data is shifted out. When the highest address is reached, it continues to read from 0x000000.

BOOL WritePage(spiHIF * spiH, BYTE* writeData, int address, DWORD byteNum)

Parameters

spiH	- Pointer to spi port handle
address	- A 3 byte address to write to
writeData	- Pointer to an array of data to write
byteNum	- Number of byte desire to write

Return Values

Returns true if success, false otherwise.

Description

This function writes ByteNum of data to the address specified. The address is automatically incremented. If the 8 least significant address bits are not all zero, when the highest address is reached, the data that goes beyond the end of the current page are programmed from the start address of the same page. Only the least 256 bytes are guaranteed to be program within the page.

BOOL WriteProtect(spiHIF * spiH, BYTE areaNum)

Parameters

spiH	- Pointer to spi port handle
areaNum	- sector area number to protect against write/erase

Return Values

Returns true if success, false otherwise.

Description

This function protects certain sectors against write/erase. The corresponding areaNum for each sector are shown in Table 2.

Table 2

ChipModel*	**0x16**	**0x128**
areaNum	**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

* ChipModel can be found with GetDeviceInfo();

BOOL SectorErase(spiHIF * spiH, int Address)
Parameters
spiH - Pointer to spi port handle
address - A 3 byte address of the sector to be erased

Return Values
Returns true if success, false otherwise.

Description
This function erases the sector that the address is in.

BOOL BulkErase(spiHIF * spiH)
Parameters
spiH - Pointer to spi port handle

Return Values
Returns true if success, false otherwise.

Description
This function erases the entire memory. It is executed only if none of the sectors are protected. This function may take a while to execute.

Calls only for chip M25P16

BOOL Sleep(spiHIF * spiH)
Parameters
spiH - Pointer to spi port handle

Return Values
Returns true if success, false otherwise.

Description
This function puts the device in low consumption mode (1uA). All commands recieved after this should be ignored, unless Wake() is called.

BOOL Wake(spiHIF * spiH)
Parameters
spiH - Pointer to spi port handle

Return Values
Returns true if success, false otherwise.

Description
This function releases the device from low consumption mode.