SPI Flash Commands

These are commands that are not (yet) explicitly exposed in the C# and C++ library (they are constants define / enum: SPI_Command in te_api.h).

SPI Flash command	Description
#define SPI_WREN 0x06	Set Write Enable Latch
#define SPI_WRDI 0x04	Reset Write Enable Latch
#define SPI_RDSR1 0x05	Read Status Register 1
#define SPI_RDSR2 0x35	Read Status Register 2
#define SPI_WRSR 0x01	Write Status Register
#define SPI_READ 0x03	Read data from memory
#define SPI_FAST_READ 0x0b	Similar to the READ command, but possibly uses a faster clock
#define SPI_WRITE 0x02	Write data to memory array
#define SPI_SE 0xD8	Erase one sector in memory
#define SPI_BE 0xC7	Erase all memory
#define SPI_DP 0xb9	Write Enable Command
#define SPI_RES 0xab	Read Electonic Signature
#define SPI_RDID 0x9F	reads the ID of the SPI Flash

The SPI Command can be dispatched through

- a particular SW API function: TE_USB_FX2_SendCommand(..., Command, …) where
- "command_array" is a byte array that contains both
  - CMD_FLASH_WRITE_COMMAND (call the spi_command() firmware function in the TE USB FX2 microcontroller) and
  - SPI Flash Commands (multiple SPI Flash Commands could be dispatched through spi_command() firmware called before).

In particular, SPI Flash Commands could be used:

- to reads the ID of the SPI Flash (# Define SPI_RDID 0x9F)
- to unlock the SPI Flash.

First case example (reads the ID of the SPI Flash)
SW host computer: reads the ID of the SPI Flash This is pseudocode close to the real one. The real code need indirection for SPI command (if written in C#).

Pseudocode, The real code need indirection for SPI command (if written in C#)

```
byte[] Command = new byte[64];
byte[] Reply = new byte[64];

Command[0] = (byte) FX2_Commands.CMD_FX2_FLASH_WRITE_COMMAND;
Command[1] = (byte) 1; //Numeber of SPI commands used by spi_command(): putcSPI(SPI_RDID);
Command[2] = (byte) 3; //Number of SPI bytes as reply: mid = 0x20 did = 0x20 uid = 0x16
Command[3] = (byte) 0x9F; //(byte)SPI_Commands.CMD_SPI_RDID; // SPI_RDID 0x9F â‰¡ get ID command
Command[4] = (byte)0;
Command[5] = (byte)0;
Command[6] = (byte)0;

/*
TE_USB_FX2_SendCommand (Command[0] = (byte) FX2_Commands.CMD_FX2_FLASH_WRITE_COMMAND) calls "case
CMD_FLASH_WRITE_COMMAND"

"case CMD_FLASH_WRITE_COMMAND" calls spi_command(EP1OUTBUF[1], &EP1OUTBUF[3], EP1OUTBUF[2], &EP1INBUF[1])
with EP1OUTBUF[3]= SPI_RDID SPI Flash Command

EP1OUTBUF = Command
Reply = EP1INBUF
*/

if (TE_USB_FX2_SendCommand(..., Command, CmdLength, Reply, ReplyLength, 5000) == true)
{
        LogTextLine += "SPI Flash IDCODE " + "uid = 0x" + Reply[1].ToString("x") + "mid = 0x "+ Reply[2].
ToString("x")+ " did = 0x" + Reply[3].ToString("x") + "\r\n";
}
```

C# real code; indirection for SPI commands

```
byte[] Command1 = new byte[64];
byte[] Reply1 = new byte[64];
int CmdLength1 = 4;
int ReplyLength1 = 64;
byte[] Command2 = new byte[64];
byte[] Reply2 = new byte[64];
int CmdLength2 = 64;
int ReplyLength2 = 64;
//To use the firmware function spi_command() you need to use a indirection
Command2[0] = (byte)FX2_Commands.CMD_FX2_FLASH_WRITE_COMMAND;
Command2[1] = (byte)0x01; //Numeber of SPI commands used by spi_command(): putcSPI
(SPI_RDID);
Command2[2] = (byte)0x03; //Number of SPI bytes as reply: mid = 0x20 did = 0x20 uid = 0x16
Command2[3] = (byte)0x9F; //(byte)SPI_Commands.CMD_SPI_RDID; // SPI_RDID 0x9F â‰¡ get ID command
Command1[0] = Command2[0];
Command1[1] = Command2[1];
Command1[2] = Command2[2];
Command1[3] = Command2[3];
Command1[4] = (byte)0;
Command1[5] = (byte)0;
Command1[6] = (byte)0;
if (TE_USB_FX2_SendCommand(..., Command1, CmdLength1, Reply1, ReplyLength1, 5000) == true)
{
        LogTextLine += "SPI Flash IDCODE " + "uid = 0x" + Reply1[1].ToString("x") + "mid = 0x "+ Reply1[2].
ToString("x")+ " did = 0x" + Reply1[3].ToString("x") + "\r\n";
}
```

FW running on USB FX2 microcontroller

This is a piece of real code (FW running on USB FX2 microcontroller)

te_api.c, lines 207-211

EP1INBUF: read Reply[] from USB FX2 microcontroller to host computer

EP1OUTBUF: write Command[] from host computer to USB FX2 microcontroller

Lines 207-211 of te_api.c

```
case CMD_FLASH_WRITE_COMMAND:
        EP1INBUF[0] = 0x55;
        //void spi_command(BYTE CmdLen, unsigned char *CmdData, BYTE RdLen, unsigned char *RdData)
        spi_command(EP1OUTBUF[1], &EP1OUTBUF[3], EP1OUTBUF[2], &EP1INBUF[1]);
        new_data = 1;
        break;

/*
Command[0] = CMD_FLASH_WRITE_COMMAND; used by TE_USB_FX2_SendCommand () to call "case
CMD_FLASH_WRITE_COMMAND" and then spi_command()

EP1OUTBUF[1] = CmdLen = Command[1]= CmdLength = 1; // used by spi_command(), MD_SPI_RDID = 0x9F is a single
byte
EP1OUTBUF[2] = RdLen = Command[2]= ReplyLength = 3; // used by spi_command(), SPI Flash ID should be 3 byte
EP1OUTBUF[3] = Command[3] = CMD_SPI_RDID = 0x9F; //used by spi_command()
Â
Reply[0] = EP1INBUF[0] = 0x55;
Reply[1] = EP1INBUF[1] = 0x20; // for example
Reply[2] = EP1INBUF[2] = 0x20; // for example
Reply[3] = EP1INBUF[3] = 0x16; // for example
*/
```

Flash	Manufacturer ID	Memory Type	Capacity
M25P32	20h - Micron	20h	16h
W25Q64FV	EFh - Winbond	40h	17h

Flash IDCODEs

FW running on USB FX2 microcontroller

This is a piece of real code

spi.c, lines 63-89

Lines 63-89 of spi.c

```
void spi_command(BYTE CmdLen, unsigned char *CmdData, BYTE RdLen, unsigned char *RdData)
{
        volatile unsigned char spi_count, rd_buff;// pr_address;
        OED = 0x73; // 0b01110011;
        FPGA_POWER = 0; //power off fpga
        FLASH_ENABLE; //assert chip select
        //Write command
        spi_count = CmdLen;
        if (spi_count > 64) spi_count = 64;
        while (spi_count > 0)
        {
                putcSPI(*CmdData); //send read command
                CmdData++;
                spi_count = spi_count - 1;
        }

        //Read response
        spi_count = RdLen;
        if (spi_count > 64) spi_count = 64;
        while (spi_count > 0)
        {
                rd_buff = getcSPI();
                *RdData = rd_buff;
                RdData++;
                spi_count = spi_count - 1;
        }
        FLASH_DISABLE;
}
```