MPX-24794S Serial USB Card

SPI Master, I2C Master, Counter, GPIO through USB Interfaces

Programmer’s Manual

Version 1.10

19F, No 94, Sec 1, Shintai 5th Road, Sijhih Dist.
New Taipei City 22102, Taiwan
Tel: +886-2-2696-3909
FAX: +886-2-2696-3911
email: tech@commell.com.tw
www: www.commell.com.tw
www.commell.com

Contents

Lists of Figures

List of Tables

Conventions

The numbers used in this manual.

Number	Description
Decimal	Decimal number will be noted just as normal numbers. For example, 3456.
Hexadecimal	Hexadecimal number will be noted in C-notation, the 0x prefix will be presented. For example, 0x3456.

Signs used in this manual.

Sign	Description
	Permanent damage. This sign indicates that permanent damage to the device and system might cause if not fully understood or followed. You should not start using the product before you have read this information.
	Information sign. This sign indicates that this information might be useful while you are using this product. This information might also help saving your time if you have read them.

Acronyms and Abbreviations

API	Application Programming Interface
AT24	Atmel AT24Cxxx I2C EEPROM
AT25	Atmel AT25xxx SPI EEPROM
CCP	Common Communication Protocol
CNTR	Counter
CNTR_EN	Counter Enable input pin
CNTR_CO	Counter Compare True output pin
CNTR_TO	Counter Terminal Count output pin
CSn#	Chip Select
EEPROM	Electrically Erasable Programmable Read-Only Memory
GPIO	General Purpose Input/Output
I2C Bus	Inter-Integrated Circuit Bus
I2CM	I2C Master
I2CmSCL	I2C master clock
I2CmSDA	I2C master data
LSB	Least Significant Byte
MHz	Megahertz (one million hertz)
MISO	Master In Slave Out
MOSI	Master Out Slave In
MSB	Most Significant Byte
PCIe	PCI Express
PWM	Pulse Width Module
SCLK	SPI Clock
SDK	Software Development Kit
SPI Bus	Serial Peripheral Interface Bus
SPIM	SPI Master
USB	Universal Serial Bus

1 Introduction

This MPX-24794S Programmer's Manual defines the communication protocol, protocol data format, and protocol functions. This manual is intended to provide all necessary technical information for MPX-24794S programmers.

MPX-24794S Card supports Microsoft Windows systems environment at this moment. No other operating systems environment are supported by the time this manual is written.

1.1 Common Communication Protocol

1.1.1 Introduction

A Common Communication Protocol (CCP) is defined in between MPX-24794S firmware and the host applications at the Microsoft Windows side.

There are two types of packets defined in CCP. They are Request Packets and Response Packets. Both Request Packets and Response Packets are formatted in Bulk transfer.

Request Packets are packets sent from USB host applications to the firmware operating in MPX-24749S through USB Bulk Out transfers. Response Packets are packets read from the MPX-24794S firmware by USB host applications through USB Bulk In transfers.

A Request Packet carries a task to be executed by the MPX-24794S firmware; a response packet in contrast is the executed result of its corresponding request packet. Normally, each response packet is generated by the MPX-24749S firmware depends on its corresponding request packet.

The following figure shows the control flow of Common Communication Protocol.

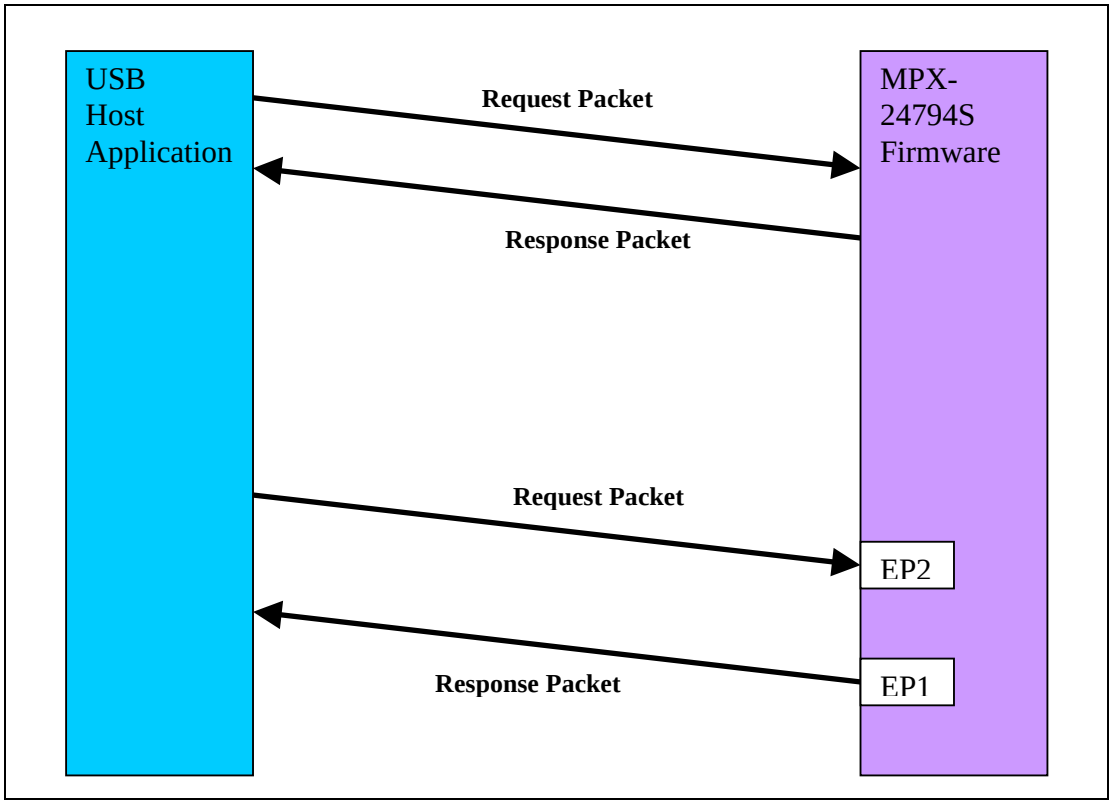

Figure 1 CCP Control Flow

1.1.2 Request Packet

A Request Packet is formatted in Bulk Out transfer data format. The maximum packet size is 64 bytes. A Request Packet is composed of a CCP header followed by DATA filed and ends with a 2's complement checksum. Request Packets are in different byte count. The host application can choose to send the Request Packet in 64 bytes or in just the size of a request type.

The following table shows the Request Packet data format.

OFFSET	NAME	DESCRIPTION
0	HEADER_MSB	0X55
1	HEADER_LSB	0XAA
2	VERSION	0X01
3	SIZEOF_DATA	The number of bytes residing in the DATA field.
4	COMMAND	The command category code.
5	ERROR	0X00 (Reserved for firmware use)
6	DATA_0 or COMMAND_0	Byte 0 of the data field or command_0 byte.
7	DATA_1 or	Byte 1 of the data filed or command_1 byte.

	COMMAND_1	
8	DATA_2 or COMMAND_2	Byte 2 of the data field or command_2 byte.
9	DATA_3 or COMMAND_3	Byte 3 of the data field or command_3 byte.
10 to N	DATA	The DATA field; up to 53 bytes.
N+1	CHECKSUM	2's complement checksum of all fields except this checksum field.

Table 1 Request Packet Format

1.1.3 Response Packet

A Response Packet is formatted in Bulk In transfer data format. The maximum packet size is 64 bytes. A Response Packet is composed of a CCP header followed by DATA filed and ends with a 2's complement checksum. Response Packets are in different byte count. The host application can choose to read the Response Packet in 64 bytes or in just the size of a request type.

The following table shows the Response Packet data format.

OFFSET	NAME	DESCRIPTION
0	HEADER_MSB	0X66
1	HEADER_LSB	0XBB
2	VERSION	0X01
3	SIZEOF_DATA	The number of bytes residing in the DATA field.
4	COMMAND	The corresponding command category code.
5	ERROR	The error code.
6 to N	DATA	The DATA field; up to 57 bytes.
N+1	CHECKSUM	2's complement checksum of all fields except this checksum field.

Table 2 Response Packet Data Format

1.2 Checksum Sample Code

The checksum byte used in CCP is 2's complement simple checksum. Here is an example in C of how the checksum is calculated.

```
/*
  Return 2's complement checksum.
*/
unsigned char ccpChecksum(unsigned char *buffer, int size)
{
```

```
    int checksum = 0;

    for(int i = 0; i < size; i++)
    {
        checksum += buffer[i];
    }

    return ((unsigned char)(0 - checksum);
}
```

2 CCP for MPX-24794S

This chapter defines the Request Packets and its Response Packets used in MPX-24749S Card. Please be noted that a request packet is a Bulk Out transfer, which towards to the Endpoint 2 of MPX-24794S Card. While, a response packet is a Bulk In transfer, which the USB host application reads it from the Endpoint 1 of the MPX-24794S Card.

2.1 SYSTEM Service

SYSTEM Service category code is 0X01. There are three sub-commands within this category.

```
#define CCP_CMND_SYSTEM_SERVICE  0X01
#define SYS_READ_REGISTERS  0X02
#define SYS_WRITE_REGISTERS  0X03
#define SYS_GET_VERSION  0X04
```

2.1.1 SYS_READ_REGISTERS (0X01:0X02)

OFFSET	BULK OUT (EP2)	BULK IN (EP1)
HEADER_MSB	0X55	0X66
HEADER_LSB	0XAA	0XBB
VERSION	0X01	0X01
SIZEOF_DATA	0X01	???
COMMAND	0X01	0X01
ERROR	0X00	Error code
DATA_0	0X02	Number of bytes read
DATA_1	number of registers to read	Byte_0
DATA_2	reg0_bank	Byte_1
DATA_3	reg0_index	Byte_2
DATA_4	reserved	...
DATA_5	reg1_bank	...
DATA_6	reg1_index	...
DATA_7	reserved	...
DATA_8	...	...
DATA_N	...	CHECKSUM
DATA_N+1	CHECKSUM	

Table 3 SYS_READ_REGISTERS

2.1.2 SYS_WRITE_REGISTERS (0X01:0X03)

OFFSET	BULK OUT (EP2)	BULK IN (EP1)
HEADER_MSB	0X55	0X66
HEADER_LSB	0XAA	0XBB
VERSION	0X01	0X01
SIZEOF_DATA	0X01	0X00
COMMAND	0X01	0X01
ERROR	0X00	Error code
DATA_0	0X03	CHECKSUM
DATA_1	number of registers to write	
DATA_2	reg0_bank	
DATA_3	reg0_index	
DATA_4	reg0_value	
DATA_5	reg1_bank	
DATA_6	reg1_index	
DATA_7	reg1_value	
DATA_8	...	
DATA_N	...	
DATA_N+1	CHECKSUM	

Table 4 SYS_WRITE_REGISTERS

2.1.3 SYS_GET_VERSION (0X01:0X04)

OFFSET	BULK OUT (EP2)	BULK IN (EP1)
HEADER_MSB	0X55	0X66
HEADER_LSB	0XAA	0XBB
VERSION	0X01	0X01
SIZEOF_DATA	0X00	0X02
COMMAND	0X01	0X01
ERROR	0X00	Error code
DATA_0	0X04	version_major
DATA_1	CHECKSUM	version_minor
DATA_2		CHECKSUM
DATA_3		

2.2 SPI Service

SPI Service provides some SPI service functions.

```
#define CCP_CMND_SPI_QUICK_READ  0X02
#define CCP_CMND_SPI_QUICK_WRITE  0X03
#define CCP_CMND_SPI_QUICK_INSTRUCT_READ  0X04
#define CCP_CMND_SPI_RESTART  0X05
#define CCP_CMND_SPI_READ  0X06
#define CCP_CMND_SPI_WRITE  0X07
#define CCP_CMND_SPI_INSTRUCT_READ  0X08
```

2.2.1 CCP_CMND_SPI_QUICK_READ (0X02)

OFFSET	BULK OUT (EP2)	BULK IN (EP1)
HEADER_MSB	0X55	0X66
HEADER_LSB	0XAA	0XBB
VERSION	0X01	0X01
SIZEOF_DATA	0X02	???
COMMAND	0X02	0X02
ERROR	0X00	Error code
DATA_0	CS (CS number)	Number of bytes read
DATA_1	number of bytes to read	Byte_0
DATA_2	CHECKSUM	Byte_1
DATA_3		Byte_2
DATA_4		...
DATA_5		...
DATA_6		...
DATA_7		...
DATA_8		...
DATA_N		CHECKSUM
DATA_N+1		

Table 5 CCP_CMND_SPI_QUICK_READ

2.2.2 CCP_CMND_SPI_QUICK_WRITE (0X03)

OFFSET	BULK OUT (EP2)	BULK IN (EP1)
HEADER_MSB	0X55	0X66
HEADER_LSB	0XAA	0XBB
VERSION	0X01	0X01
SIZEOF_DATA	???	0X01

COMMAND	0X03	0X03
ERROR	0X00	Error code
DATA_0	CS (CS number)	Number of bytes written
DATA_1	number of bytes to write	CHECKSUM
DATA_2	output_byte_0	
DATA_3	output_byte_1	
DATA_4	output_byte_2	
DATA_5	...	
DATA_6	...	
DATA_7	...	
DATA_8	...	
DATA_N	CHECKSUM	
DATA_N+1		

Table 6 CCP_CMND_SPI_QUICK_WRITE

2.2.3 CCP_CMND_SPI_QUICK_INSTRUCT_READ (0X04)

OFFSET	BULK OUT (EP2)	BULK IN (EP1)
HEADER_MSB	0X55	0X66
HEADER_LSB	0XAA	0XBB
VERSION	0X01	0X01
SIZEOF_DATA	???	???
COMMAND	0X04	0X04
ERROR	0X00	Error code
DATA_0	CS (CS number)	Number of bytes read
DATA_1	number of instruct bytes	Byte_0
DATA_2	number of bytes to read	Byte_1
DATA_3	instruct_byte_0	Byte_2
DATA_4	instruct_byte_1	...
DATA_5	instruct_byte_2	...
DATA_6	...	...
DATA_7	...	...
DATA_N	CHECKSUM	...
DATA_N+1		CHECKSUM

Table 7 CCP_CMND_SPI_QUICK_INSTRUCT_READ

2.2.4 CCP_CMND_SPI_RESTART (0X05)

OFFSET	BULK OUT (EP2)	BULK IN (EP1)

HEADER_MSB	0X55	0X66
HEADER_LSB	0XAA	0XBB
VERSION	0X01	0X01
SIZEOF_DATA	0X02	0X00
COMMAND	0X05	0X05
ERROR	0X00	Error code
DATA_0	SPI_MODE_0 SPI_MODE_1 SPI_MODE_2 SPI_MODE_3	CHECKSUM
DATA_1	SPI_LSB_FIRST SPI_MSB_FIRST	
DATA_2	CHECKSUM	
DATA_3		
DATA_4		
DATA_5		
DATA_6		
DATA_7		
DATA_8		
DATA_N		

Table 8 CCP_CMND_SPI_RESTART

2.2.5 CCP_CMND_SPI_READ (0X06)

OFFSET	BULK OUT (EP2)	BULK IN (EP1)
HEADER_MSB	0X55	0X66
HEADER_LSB	0XAA	0XBB
VERSION	0X01	0X01
SIZEOF_DATA	0X02	???
COMMAND	0X06	0X06
ERROR	0X00	Error code
DATA_0	CS (CS number)	Number of bytes read
DATA_1	SPI_WAIT_0/1/2/3	Byte_0
DATA_2	number of bytes to read	Byte_1
DATA_3	CHECKSUM	Byte_2
DATA_4		...
DATA_5		...
DATA_6		...
DATA_7		...
DATA_8		...
DATA_N		CHECKSUM

Table 9 CCP_CMND_SPI_READ

2.2.6 CCP_CMND_SPI_WRITE (0X07)

OFFSET	BULK OUT (EP2)	BULK IN (EP1)
HEADER_MSB	0X55	0X66
HEADER_LSB	0XAA	0XBB
VERSION	0X01	0X01
SIZEOF_DATA	0X02	0X01
COMMAND	0X07	0X07
ERROR	0X00	Error code
DATA_0	CS (CS number)	Number of bytes writtenh
DATA_1	SPI_WAIT_0/1/2/3	CHECKSUM
DATA_2	number of bytes to write	
DATA_3	output_byte_0	
DATA_4	output_byte_1	
DATA_5	output_byte_2	
DATA_6	...	
DATA_7	...	
DATA_8	...	
DATA_N	CHECKSUM	

Table 10 CCP_SPI_CMND_WRITE

2.2.7 CCP_CMND_SPI_INSTRUCT_READ (0X08)

OFFSET	BULK OUT (EP2)	BULK IN (EP1)
HEADER_MSB	0X55	0X66
HEADER_LSB	0XAA	0XBB
VERSION	0X01	0X01
SIZEOF_DATA	???	???
COMMAND	0X08	0X08
ERROR	0X00	Error code
DATA_0	CS (CS number)	Number of bytes read
DATA_1	SPI_WAIT_0/1/2/3	Byte_0
DATA_2	number of instruct bytes	Byte_1
DATA_3	number of bytes to read	Byte_2
DATA_4	instruct_byte_0	...
DATA_5	instruct_byte_1	...
DATA_6	...	...
DATA_N	...	CHECKSUM
DATA_N+1	CHECKSUM	

Table 11 CCP_CMND_SPI_INSTRUCT_READ

2.3 COUNTER Service

MPX-24794S provides COUNTER Service as defined here in this section.

```
#define CCP_CMND_CNTR_START_STOP  0X09
#define CCP_CMND_CNTR_CONFIGURE  0X0A
#define CCP_CMND_CNTR_WRITE_VALUES  0X0B
#define CCP_CMND_CNTR_READ_VALUES  0x0C
#define CCP_CMND_CNTR_QUICK_START  0X0D
#define CNTR_CONF_INTR_STATUS  0X00
#define CNTR_CONF_INTR_TYPE  0X01
#define CNTR_CONF_COMPARE_TYPE  0X02
#define CNTR_CONF_DATA_INVERT  0X03
#define CNTR_CONF_CLOCK_INPUT  0X04
#define CNTR_INTR_ENABLED  /* status */
#define CNTR_INTR_DISABLED /* status */
#define CNTR_INTR_COMPARE_TRUE  /* type */
#define CNTR_INTR_TERMINAL_COUNT /* type */
#define CNTR_COMPARE_LESS_THAN
#define CNTR_COMPARE_EQUAL
#define CNTR_CLK_VC1
#define CNTR_CLK_VC2
#define CNTR_CLK_VC3
#define CNTR_CLK_SYSCLKX2
#define CNTR_DATA_NORMAL
#define CNTR_DATA_INVERT
```

2.3.1 CCP_CMND_CNTR_START_STOP (0X09)

OFFSET	BULK OUT (EP2)	BULK IN (EP1)
HEADER_MSB	0X55	0X66
HEADER_LSB	0XAA	0XBB
VERSION	0X01	0X01
SIZEOF_DATA	0X01	0X00
COMMAND	0X09	0X09
ERROR	0X00	Error code
DATA_0	CNTR_START (0X80) CNTR_STOP (0X81)	CHECKSUM
DATA_1	CHECKSUM	
DATA_2		

Table 12 CCP_CMND_CNTR_START_STOP

2.3.2 CCP_CMND_CNTR_CONFIGURE (0X0A)

OFFSET	BULK OUT (EP2)	BULK IN (EP1)
HEADER_MSB	0X55	0X66
HEADER_LSB	0XAA	0XBB
VERSION	0X01	0X01
SIZEOF_DATA	0X01	0X00
COMMAND	0X09	0X09
ERROR	0X00	Error code
DATA_0	CNTR_CONF_INTR_STATUS CNTR_CONF_INTR_TYPE CNTR_CONF_COMPARE_TYPE CNTR_CONF_DATA_INVERT CNTR_CONF_CLOCK_INPUT	CHECKSUM
DATA_1	CNTR_INTR_XXXXX CNTR_COMPARE_XXXX CNTR_INVERT_XXXX CNTR_CLK_XXXX	
DATA_2	CHECKSUM	

Table 13 CCP_CMND_CNTR_CONFIGURE

2.3.3 CCP_CMND_CNTR_WRITE_VALUES (0X0B)

OFFSET	BULK OUT (EP2)	BULK IN (EP1)
HEADER_MSB	0X55	0X66
HEADER_LSB	0XAA	0XBB
VERSION	0X01	0X01
SIZEOF_DATA	0X04	0X01
COMMAND	0X0B	0X0B
ERROR	0X00	Error code
DATA_0	PERIOD_VALUE_MSB	number of bytes written
DATA_1	PERIOD_VALUE_LSB	CHECKSUM
DATA_2	COMPARE_VALUE_MSB	
DATA_3	COMPARE_VALUE_LSB	
DATA_4	CHECKSUM	

Table 14 CCP_CMND_CNTR_WRITE_VALUES

2.3.4 CCP_CMND_CNTR_READ_VALUES (0X0C)

OFFSET	BULK OUT (EP2)	BULK IN (EP1)
HEADER_MSB	0X55	0X66

HEADER_LSB	0XAA	0XBB
VERSION	0X01	0X01
SIZEOF_DATA	0X00	0X05
COMMAND	0X0C	0X0C
ERROR	0X00	Error code
DATA_0	CHECKSUM	number of bytes read
DATA_1		COMPARE_VALUE_MSB
DATA_2		COMAPRE_VALUE_LSB
DATA_3		COUNTER_VALUE_MSB
DATA_4		COUNTER_VALUE_LSB
DATA_5		CHECKSUM

Table 15 CCP_CMND_CNTR_READ_VALUES

2.3.5 CCP_CMND_CNTR_QUICK_START (0X0D)

OFFSET	BULK OUT (EP2)	BULK IN (EP1)
HEADER_MSB	0X55	0X66
HEADER_LSB	0XAA	0XBB
VERSION	0X01	0X01
SIZEOF_DATA	0X09	0X00
COMMAND	0X0D	0X0D
ERROR	0X00	Error code
DATA_0	CNTR_PERIOD_VALUE_MSB	CHECKSUM
DATA_1	CNTR_PERIOD_VALUE_LSB	
DATA_2	CNTR_COMPARE_VALUE_MSB	
DATA_3	CNTR_COMPARE_VALUE_LSB	
DATA_4	CNTR_INTR_ENABLED CNTR_INTR_DISABLED	
DATA_5	CNTR_INTR_COMPARE_TRUE CNTR_INTR_TERMINAL_COUNT	
DATA_6	CNTR_COMPARE_LESS_THAN CNTR_COMPARE_EQUAL	
DATA_7	CNTR_CLK_VC1 CNTR_CLK_VC2 CNTR_CLK_VC3 CNTR_CLK_SYSCLKX2	
DATA_8	CNTR_DATA_NORMAL CNTR_DATA_INVERT	
DATA_9	CHECKSUM	

Table 16 CCP_CMND_CNTR_QUICK_START

2.4 GPIO Service

GPIO Service category code is 0X0E. The MPX-24794S provides the following very useful GPIO sub-commands.

```
#define CCP_CMND_GPIO_COMMAND  0X0E

#define GPIO_READ_PRTxDR  0X10
#define GPIO_WRITE_PRTxDR  0X11
#define GPIO_READ_PRTxIE  0X20
#define GPIO_WRITE_PRTxIE  0x21
#define GPIO_READ_PRTxDMx  0X30
#define GPIO_WRITE_PRTxDMx  0X31
#define GPIO_READ_PRTxICx  0X40
#define GPIO_WRITE_PRTxICx  0X41

#define GPIO_PORT3  0X01
#define GPIO_PORT4  0X02
#define GPIO_PORT3_PORT4  (GPIO_PORT3 | GPIO_PORT4)
```

2.4.1 GPIO_READ_PRTxDR (0X0E:0X10)

OFFSET	BULK OUT (EP2)	BULK IN (EP1)
HEADER_MSB	0X55	0X66
HEADER_LSB	0XAA	0XBB
VERSION	0X01	0X01
SIZEOF_DATA	0X02	???
COMMAND	0X0E	0X0E
ERROR	0X00	Error code
DATA_0	0X10	Number of bytes read
DATA_1	GPIO_PORT3 GPIO_PORT4 GPIO_PORT3_PORT4	byte_0
DATA_2	CHECKSUM	byte_1 (if two ports)
DATA_3		CHECKSUM
DATA_4		
DATA_5		
DATA_6		
DATA_7		

Table 17 GPIO_READ_PRTxDR

2.4.2 GPIO_WRITE_PRTxDR (0X0E:0X11)

OFFSET	BULK OUT (EP2)	BULK IN (EP1)

HEADER_MSB	0X55	0X66
HEADER_LSB	0XAA	0XBB
VERSION	0X01	0X01
SIZEOF_DATA	???	0X01
COMMAND	0X0E	0X0E
ERROR	0X00	Error code
DATA_0	0X11	Number of bytes written
DATA_1	GPIO_PORT3 GPIO_PORT4 GPIO_PORT3_PORT4	CHECKSUM
DATA_2	output_byte_0	
DATA_3	output_byte_1 (if two ports)	
DATA_4	CHECKSUM	
DATA_5		
DATA_6		
DATA_7		

Table 18 GPIO_WRITE_PRTxDR

2.4.3 GPIO_READ_PRTxIE (0X0E:0X20)

OFFSET	BULK OUT (EP2)	BULK IN (EP1)
HEADER_MSB	0X55	0X66
HEADER_LSB	0XAA	0XBB
VERSION	0X01	0X01
SIZEOF_DATA	0X02	???
COMMAND	0X0E	0X0E
ERROR	0X00	Error code
DATA_0	0X20	Number of bytes read
DATA_1	GPIO_PORT3 GPIO_PORT4 GPIO_PORT3_PORT4	byte_0
DATA_2	CHECKSUM	byte_1 (if two ports)
DATA_3		CHECKSUM
DATA_4		
DATA_5		
DATA_6		
DATA_7		

Table 19 GPIO_READ_PRTxIE

2.4.4 GPIO_WRITE_PRTxIE (0X0E:0X21)

OFFSET	BULK OUT (EP2)	BULK IN (EP1)
HEADER_MSB	0X55	0X66
HEADER_LSB	0XAA	0XBB
VERSION	0X01	0X01
SIZEOF_DATA	???	0X01
COMMAND	0X0E	0X0E
ERROR	0X00	Error code
DATA_0	0X21	Number of bytes written
DATA_1	GPIO_PORT3 GPIO_PORT4 GPIO_PORT3_PORT4	CHECKSUM
DATA_2	output_byte_0	
DATA_3	output_byte_1 (if both)	
DATA_4	CHECKSUM	
DATA_5		
DATA_6		
DATA_7		

Table 20 GPIO_READ_PRTxIE

2.4.5 GPIO_READ_PRTxDMx (0X0E:0X30)

OFFSET	BULK OUT (EP2)	BULK IN (EP1)
HEADER_MSB	0X55	0X66
HEADER_LSB	0XAA	0XBB
VERSION	0X01	0X01
SIZEOF_DATA	0X02	???
COMMAND	0X0E	0X0E
ERROR	0X00	Error code
DATA_0	0X30	Number of bytes read
DATA_1	GPIO_PORT3 GPIO_PORT4 GPIO_PORT3_PORT4	PRT3DM0 / PRT4DM0
DATA_2	CHECKSUM	PRT3DM1 / PRT4DM1
DATA_3		PRT3DM2 / PRT4DM2
DATA_4		PRT4DM0 (if both)
DATA_5		PRT4DM1 (if both)
DATA_6		PRT4DM2 (if both)
DATA_7		CHECKSUM

Table 21 GPIO_READ_PRTxDMx

2.4.6 GPIO_WRITE_PRTxDMx (0X0E:0X31)

OFFSET	BULK OUT (EP2)	BULK IN (EP1)
HEADER_MSB	0X55	0X66
HEADER_LSB	0XAA	0XBB
VERSION	0X01	0X01
SIZEOF_DATA	???	0X01
COMMAND	0X0E	0X0E
ERROR	0X00	Error code
DATA_0	0X31	Number of bytes written
DATA_1	GPIO_PORT3 GPIO_PORT4 GPIO_PORT3_PORT4	CHECKSUM
DATA_2	PRT3DM0 / PRT4DM0	
DATA_3	PRT3DM1 / PRT4DM1	
DATA_4	PRT3DM2 / PRT4DM2	
DATA_5	PRT4DM0 (if both)	
DATA_6	PRT4DM1 (if both)	
DATA_7	PRT4DM2 (if both)	
DATA_8	CHECKSUM	

Table 22 GPIO_WRITE_PRTxDMx

2.4.7 GPIO_READ_PRTxICx (0X0E:0X40)

OFFSET	BULK OUT (EP2)	BULK IN (EP1)
HEADER_MSB	0X55	0X66
HEADER_LSB	0XAA	0XBB
VERSION	0X01	0X01
SIZEOF_DATA	0X02	???
COMMAND	0X0E	0X0E
ERROR	0X00	Error code
DATA_0	0X40	Number of bytes read
DATA_1	GPIO_PORT3 GPIO_PORT4 GPIO_PORT3_PORT4	PRT3IC0 / PRT4IC0
DATA_2	CHECKSUM	PRT3IC1 / PRT4IC1
DATA_3		PRT4IC0 (if both)
DATA_4		PRT4IC1 (if both)
DATA_5		CHECKSUM
DATA_6		
DATA_7		

Table 23 GPIO_READ_PRTxICx

2.4.8 GPIO_WRITE_PRTxICx (0X0E:0X41)

OFFSET	BULK OUT (EP2)	BULK IN (EP1)
HEADER_MSB	0X55	0X66
HEADER_LSB	0XAA	0XBB
VERSION	0X01	0X01
SIZEOF_DATA	???	0X01
COMMAND	0X0E	0X0E
ERROR	0X00	Error code
DATA_0	0X41	Number of bytes written
DATA_1	GPIO_PORT3 GPIO_PORT4 GPIO_PORT3_PORT4	CHECKSUM
DATA_2	PRT3IC0 / PRT4IC0	
DATA_3	PRT3IC1 / PRT4IC1	
DATA_4	PRT4IC0 (if both)	
DATA_5	PRT4IC1 (if both)	
DATA_6	CHECKSUM	
DATA_7		

Table 24 GPIO_WRITE_PRTxICx

2.5 I2C Service

This section defines I2C services provided by MPX-24749S firmware. These are I2C master services.

CCP_CMND_I2CM_LL_API is low-level I2C master APIs provided for those applications that want to control the I2C bus signaling all by themselves.

Users can get more detail technical information from Cypress I2C Master User Module manual.

```
#define CCP_CMND_I2CM_READ  0X11
#define CCP_CMND_I2CM_WRITE  0X12
#define CCP_CMND_I2CM_LL_API  0X13

#define I2C_MODE_COMPLETE_XFER  0X00
#define I2C_MODE_REPEAT_START  0X01
#define I2C_MODE_NO_STOP  0X02
#define I2CM_MODE_COMPLETE_XFER  I2C_MODE_COMPLETE_XFER
#define I2CM_MODE_REPEAT_START  I2C_MODE_REPEAT_START
#define I2CM_MODE_NO_STOP  I2C_MODE_NO_STOP

#define I2CM_LL_API_SEND_START  0X01
```

```
#define I2CM_LL_API_SEND_REPEAT_START  0X02
#define I2CM_LL_API_SEND_STOP  0X03
#define I2CM_LL_API_READ_SINGLE_BYTE  0X04
#define I2CM_LL_API_WRITE_SINGLE_BYTE  0X05

#define I2CM_LL_CONST_WRITE  0X00
#define I2CM_LL_CONST_READ  0x01
#define I2CM_LL_CONST_ACK_SLAVE  0X01
#define I2CM_LL_CONST_NAK_SLAVE  0X00
```

2.5.1 CCP_CMND_I2CM_READ (0X11)

OFFSET	BULK OUT (EP2)	BULK IN (EP1)
HEADER_MSB	0X55	0X66
HEADER_LSB	0XAA	0XBB
VERSION	0X01	0X01
SIZEOF_DATA	0X03	???
COMMAND	0X11	0X11
ERROR	0X00	Error code
DATA_0	I2C_MODE_COMPLETE_XFER I2C_MODE_REPEAT_START I2C_MODE_NO_STOP	Number of bytes read
DATA_1	7-bit slave address (MSB = 0; no shift)	read_byte_0
DATA_2	number of bytes to read	read_byte_1
DATA_3	CHECKSUM	read_byte_2
DATA_4		...
DATA_5		...
DATA_6		...
DATA_N		CHECKSUM

Table 25 CCP_CMND_I2CM_READ

2.5.2 CCP_CMND_I2CM_WRITE (0X12)

OFFSET	BULK OUT (EP2)	BULK IN (EP1)
HEADER_MSB	0X55	0X66
HEADER_LSB	0XAA	0XBB
VERSION	0X01	0X01
SIZEOF_DATA	???	0X01
COMMAND	0X12	0X11
ERROR	0X00	Error code
DATA_0	I2C_MODE_COMPLETE_XFER I2C_MODE_REPEAT_START	Number of bytes written

	I2C_MODE_NO_STOP	
DATA_1	7-bit slave address (MSB = 0; no shift)	CHECKSUM
DATA_2	number of bytes to write	
DATA_3	output_byte_0	
DATA_4	output_byte_1	
DATA_5	output_byte_2	
DATA_6	...	
...	...	
DATA_N	CHECKSUM	

Table 26 CCP_CMND_I2CM_WRITE

2.5.3 I2CM_LL_API_SEND_START (0X13:0X01)

This function ***I2Cm_fSendStart*** generates an I2C bus start condition, sends the address and R/W bit, and then returns the ACK result.

OFFSET	BULK OUT (EP2)	BULK IN (EP1)
HEADER_MSB	0X55	0X66
HEADER_LSB	0XAA	0XBB
VERSION	0X01	0X01
SIZEOF_DATA	0X03	0X01
COMMAND	0X13	0X11
ERROR	0X00	Error code
DATA_0	I2CM_LL_API_SEND_START	0X00: NAK Others: ACK
DATA_1	I2CM_LL_CONST_WRITE I2CM_LL_CONST_READ	CHECKSUM
DATA_2	7-bit slave address (MSB = 0; no shift)	
DATA_3	CHECKSUM	
DATA_4		
DATA_5		
DATA_6		
DATA_N		

Table 27 I2CM_LL_API_SEND_START

2.5.4 I2CM_LL_API_SEND_REPEAT_START (0X13:0X02)

This function ***I2Cm_fSendRepeatStart*** generates an I2C bus repeat start condition, sends the address and R/W bit, and then returns the ACK result.

OFFSET	BULK OUT (EP2)	BULK IN (EP1)
HEADER_MSB	0X55	0X66
HEADER_LSB	0XAA	0XBB
VERSION	0X01	0X01
SIZEOF_DATA	0X03	0X01
COMMAND	0X13	0X11
ERROR	0X00	Error code
DATA_0	I2CM_LL_API_SEND_REPEAT_START	0X00: NAK Others: ACK
DATA_1	I2CM_LL_CONST_WRITE I2CM_LL_CONST_READ	CHECKSUM
DATA_2	7-bit slave address (MSB = 0; no shift)	
DATA_3	CHECKSUM	
DATA_4		
DATA_5		
DATA_6		
DATA_N		

Table 28 I2CM_LL_API_SEND_REPEAT_START

2.5.5 I2CM_LL_API_SEND_STOP (0X13:0X03)

OFFSET	BULK OUT (EP2)	BULK IN (EP1)
HEADER_MSB	0X55	0X66
HEADER_LSB	0XAA	0XBB
VERSION	0X01	0X01
SIZEOF_DATA	0X01	0X00
COMMAND	0X13	0X13
ERROR	0X00	Error code
DATA_0	I2CM_LL_API_SEND_STOP	CHECKSUM
DATA_1	CHECKSUM	
DATA_2		
DATA_3		
DATA_4		
DATA_5		
DATA_6		
DATA_N		

Table 29 I2CM_LL_API_SEND_STOP

2.5.6 I2CM_LL_API_READ_SINGLE_BYTE (0X13:0X04)

OFFSET	BULK OUT (EP2)	BULK IN (EP1)
HEADER_MSB	0X55	0X66
HEADER_LSB	0XAA	0XBB
VERSION	0X01	0X01
SIZEOF_DATA	0X02	0X01
COMMAND	0X13	0X13
ERROR	0X00	Error code
DATA_0	I2CM_LL_API_READ_SINGLE_BYTE	read_byte
DATA_1	I2CM_LL_CONST_ACK_SLAVE I2CM_LL_CONST_NAK_SLAVE	CHECKSUM
DATA_2	CHECKSUM	
DATA_3		
DATA_4		
DATA_5		
DATA_6		
DATA_N		

Table 30 I2CM_LL_API_READ_SINGLE_BYTE

2.5.7 I2CM_LL_API_WRITE_SINGLE_BYTE (0X13:0X05)

OFFSET	BULK OUT (EP2)	BULK IN (EP1)
HEADER_MSB	0X55	0X66
HEADER_LSB	0XAA	0XBB
VERSION	0X01	0X01
SIZEOF_DATA	0X02	0X01
COMMAND	0X13	0X13
ERROR	0X00	Error code
DATA_0	I2CM_LL_API_WRITE_SINGLE_BYTE	0: NAK Others: ACK
DATA_1	output_byte	CHECKSUM
DATA_2	CHECKSUM	
DATA_3		
DATA_4		
DATA_5		
DATA_6		
DATA_N		

Table 31 I2CM_LL_API_WRITE_SINGLE_BYTE

3 Programming

Programmers have choice to use C/C++ library and/or .NET class from Cypress to implement USB host side applications. Please refer to the documentations in the product companion CD for detail information.

3.1 Sample Codes

Please refer to the sample codes in the product companion CD.

4 References

[1] PSoC Programmable System-on-Chip Technical Reference Manual, Document No. 001-14463 Rev. F, Cypress Semiconductors

[2] Serial Peripheral Interface (SPI) Master 2.10, Cypress Document Number 001-65239.

[3] I2C Master Datasheet v1.4, Cypress Document Number 001-13564 Rev. H.

[4] The I2C-Bus Specification, Version 2.1, January 2000, Philips Semiconductors.

[5] Microsoft MSDN Visual C/C++ Help

[6] Microsoft MSDN .NET Framework Help

[7] Microsoft Visual Studio 2010 Help