Application Note

AN2417/D
Rev. 0, 04/2003

MC9328MX1/MXL PCMCIA and Compact Flash Interface

By Angus Lai

1 Introduction

This document describes the implementation of PCMCIA / Compact Flash interface on MC9328MX1 Application Development System (ADS). PCMCIA / Compact Flash interface is connected to Chip Select 5 of External Interface Module (EIM). The hardware connection between PCMCIA / Compact Flash interface and the DragonBall MX Processor are described. Programming example to access Compact Flash Memory Card are given.

1.1 PCMCIA Overview

The PCMCIA specification is developed by Personal Computer Memory Card International Association. It defines a standard 68 pin hardware interface for small removable 16 bit PC card. A wide variety of hardware applications can be implemented using PC Cards. Typical application include memory device, modems, ethernet controller etc. PC Card can be operated in two distinct modes:

1.1.1 Memory Mode

The memory only interface support memory cards, but does not include signals which support I/O Cards. It is the default interface whenever a card is inserted into a socket, as well as the default interface for host to access the PC Card‘s Card Information Structure.

1.1.2 I/O Mode

The IO interface permits operation of either memory cards, I/O cards or multifunction cards containing combinations of both memory and I/O devices.

Besides, the 26 bit address space of PC Cards are divided into attribute memory space and common memory space by the REG# signal. Common memory space is where the storage memory of the memory card lies on. It is referred by logic high of the REG# signal. Attribute memory is a separately access section of card memory, and generally is used to access the Card Information Structure and Function Configuration Registers. Only even locations are accessible within attribute memory space.

1.2 Compact Flash Overview

Compact Flash is a PCMCIA compatible small removable 16 bit storage device introduced in 1994 by SanDisk Corporation. Compact Flash cards are designed with flash technology, a non-volatile storage solution that does not require a battery to retain data indefinitely. CF I/O cards are available as modems, Ethernet, serial, BlueTooth wireless, 802.11b WiFi LAN, etc. Compared to a 68-pin PCMCIA PC Card, a CF card has 50 pins and 11 bit address space, but still conforms to PCMCIA specifications.

Besides memory mode and I/O mode, Compact Flash can also operate in True IDE mode for easy interface to hardware platforms containing IDE controllers.

Compact Flash is electrically compatible with PCMCIA specification.

1.3 Hardware Interface

This section describes the hardware interface between the PCMCIA / Compact Flash interface and the MC9328MX1 ADS.

The PCMCIA / Compact Flash interface should be connected to MC9328MX1/MXL processor through Chip Select 5 of EIM. The DTACK feature of CS5 allow PC Card / Compact Flash to use WAIT# signal to extend normal access timing. As DTACK feature is available in Chip Select 5 only, the PCMCIA / Compact Flash interface should not be connect to Chip Select other than CS5 unless programmable wait state is used instead of DTACK.

The DragonBall MX processor cannot generate PCMCIA / CF bus cycle directly. For this reason, glue logic is added between the PCMCIA / CF interface and the EIM interface to convert the signal from EIM to the necessary PCMCIA / Compact Flash control signal. The glue logic also generate the right timing of the control signal between the processor and the Compact Flash device, ensuring setup time and hold time requirement of both devices are met.

The DragonBall MX processor will generate memory cycle that do not contain separated I/O control signal. However the Compact Flash standard specify that separate read and write control signals be used to distinguish between Compact Flash memory and I/O cycle. For this reason, address line A23 is used to break the read and write control signal into memory and I/O cycle.

The address bus and data bus of Compact Flash interface is isolated from the processor address bus and data bus by buffers. This is necessary to ensure the PCMCIA / Compact Flash device will not deteriorate the bus during hot insertion / removal.

Compact Flash interface can work in three separate mode

1. Memory mode
2. I/O mode
3. True IDE mode

Only Memory mode and I/O mode is support as this two operating mode is supported by all compact flash device and is electrically compatible with PCMCIA

1.4 Compact Flash Interface Block Diagram

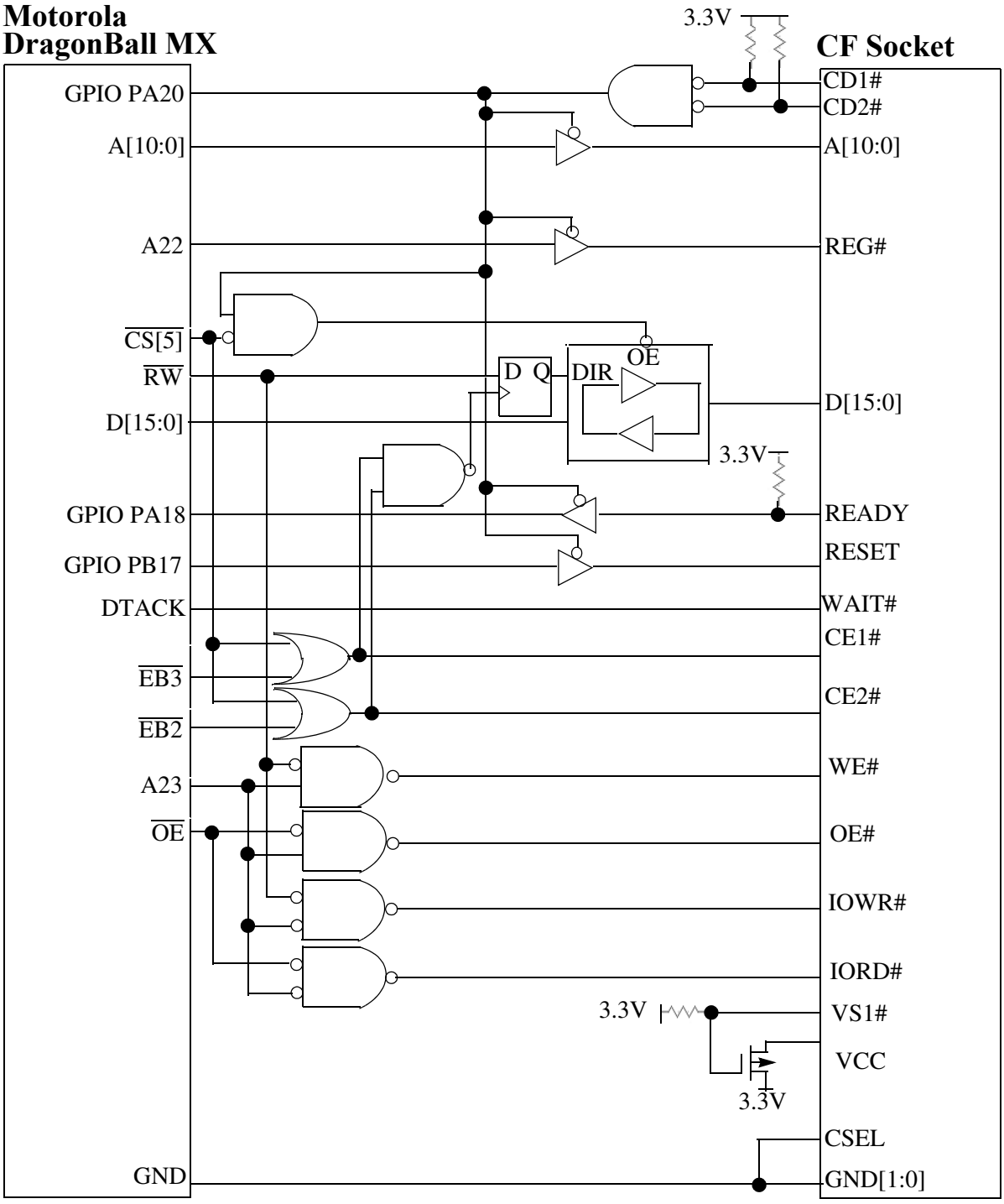
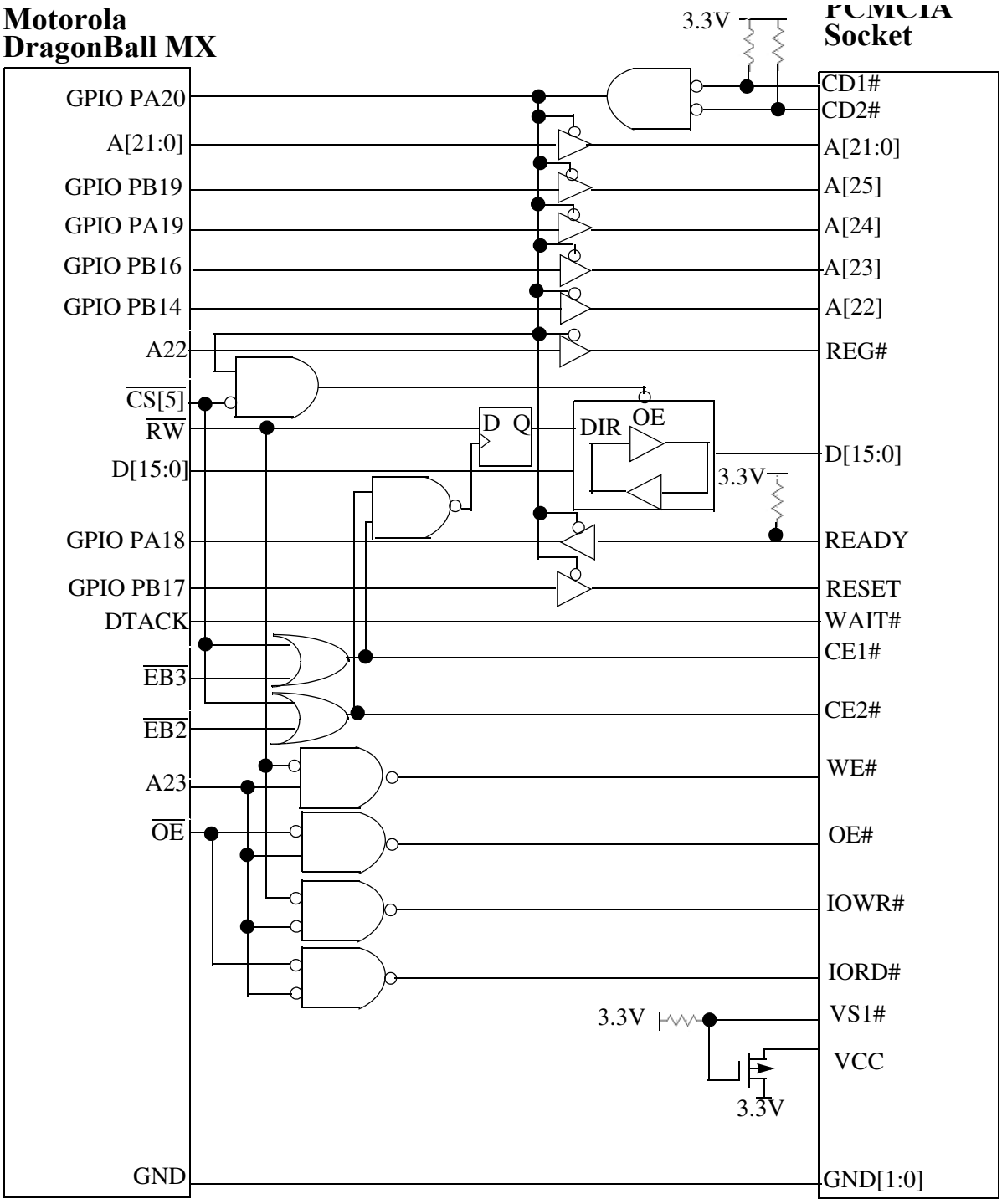


Figure 1.

NOTE:

This diagram only show the logic level block diagram. Timing consideration is needed in the exact implementation and is discussed in section 4 timing characteristics

1.5 PCMCIA Interface Block Diagram



NOTE:

This diagram only show the logic level block diagram. Timing consideration is needed in the exact implementation and is discussed in section 4 timing characteristics

1.6 Signal Description (Compact Flash)

1.6.1 Power Signal

The voltage supply circuit is implemented by a PMOS FET transistor. VS1# is connected to the gate of the PMOS FET with pull up resistor. If the Card support 3.3 V voltage, the VS1# pin is internally grounded inside the CF card, and VCC will have 3.3V voltage supply. 5V voltage supply is not supported.

Table 1.

Signal Name	Total Number of Pin	Direction of Signal	Function	Connection to MX1
VCC	1	DC in	Supply Voltage	Connect to Power Supply Circuit
GND	4	Ground	Ground	Connect to Ground
VS1# VS2#	2	Output	Voltage Sense	Vs1 Connect to power supply circuit Vs2 not connected

1.6.2 Address Bus

Compact Flash address bus width is 11 bit. The address bus is connected to DragonBall MX Processor lower 11 bit address bus through a unidirectional buffer.

The REG# signal is used to separate attribute memory address space from common memory address space. When REG# signal is asserted, the address refers to attribute memory address space. Since Compact Flash need only 11 address pins and CS5 can support up to 24 address pin, the REG# signal is implemented by A22.

Table 2.

Signal Name	Total Number of Pin	Direction of Signal	Function	Connection to MX1
A0 - A10	11	Input	Address Bus	Connection to MC9328MX1/MXL address bus A[10:0] through buffer
REG#	4	Input	Use to Distinguish between attribute memory and common memory	Connect to MC9328MX1/MXL A22 through buffer

1.6.3 Data Bus

The data bus is connected to DragonBall MX databus through bi-directional buffer. The buffer is enable only when access cycle is active (CS5 is asserted). $\overline{RW}$ signal from CS5 indicate whether it is a read or write access cycle. This signal is latched by a D flipflop and connected to buffer direction control.

Table 3.

Signal Name	Total Number of Pin	Direction of Signal	Function	Connection to MX1
D0 - D15	16	Bi Directional	Data Bus	Connection to MC9328MX1/MXL databus bus through buffer

1.6.4 Control Signal

OE# WE# IORD# IOWR#

Output Enable signal OE# is generated by host to indicate the access cycle is a read cycle, and this signal is used in memory mode. In I/O mode, separate signal IORD# is used to indicate a read cycle. Similarly, WE# is used in memory mode and IOWR# for I/O mode to indicate a write access cycle. Since DragonBall MX do not have separate output enable pin for I/O access cycle, address pin A23 is used to select whether the access cycle is an memory access or I/O access.

The Compact Flash specification define minimum setup time and hold time for the OE#, WE#, IORD# and IOWR# signal that may not met by CS5. For this reason, glue logic is necessary to produce the delay to meet the setup and hold time requirement which is discussed under the section Timing Characteristic.

CE1# CE2#

CE1# and CE2# are used to select the card and to indicate the card whether a byte or word operation is being performed. CE2# access the odd byte of word and CE1# access the even or odd byte of the word depends on A0. A multiplexing scheme allow 16 bit or 8 bit data to be transferred in the access cycle and is shown in the following table.

Table 4.

Addressing Mode	CE1#	CE2#	A0	D15:D8	D7:D0
No access	1	1	X	High-Z	High-Z
8/16 bit (even byte)	0	1	0	High-Z	Even Byte
8 bit (odd byte)	0	1	1	High-Z	Odd Byte
16 bit (odd byte only)	1	0	X	Odd Byte	High-Z
16 bit (even & odd byte)	0	0	X	Odd Byte	Even Byte

In EIM, $\overline{EB3}$ signal control the enable of D7:D0 and $\overline{EB2}$ signal control the enable of D15:D8. Hence CE1# signal is the logic OR output of $\overline{EB3}$ and $\overline{CS5}$ while CE2# signal is the logic OR output of $\overline{EB2}$ and $\overline{CS5}$.

READY (IREQ)

The ready signal is set high when Compact Flash is ready to accept a new data transfer operation and held low when the card is busy. This signal is specially useful at power up and reset by the reset pin. At power up and reset, the RDY/BSY signal is held low until Compact Flash has complete its reset function. No access to card should be made during this time.

If the Compact Flash is configured to work in I/O mode, the signal is used as interrupt request. This line is strobed low to generate a pulse mode interrupt or held low for a level mode interrupt. The mode of interrupt can be select in the Configuration Option Register in attribute memory. When the Interrupt function is used, the GPIO PA18 should be set as interrupt accordingly.

RESET

The reset pulse is generated by setting GPIO PB17 high, running a delay loop for a minimum duration of 10us and setting GPIO PB 17 low. This reset is necessary at power up after 1ms of power supply.

WAIT

The WAIT signal from compact flash is connect to DTACK of CS5.

The duration of the access cycle is controlled by either the wait signal or the setting of programmable wait state. If wait signal is used to control the access cycle length, DTACK should be enabled and select as Compact Flash wait function. To enable DTACK, WSC bits (EIM_CS5H[45:40]) should be set as 0x3F. Also bit 63 of Chip Select 5 Upper Control Register (EIM_CS5H[63]) should be set to configure DTACK to support Compact Flash wait signal.

Note that if DTACK is enabled, the Compact Flash Card must assert the wait signal within 1024 system clock cycle. Otherwise bus error will occur. Also only 16 bit access is can be used when DTACK is enabled and CS5 is configured as 16 bit access. The EIM will not separate 32 bit access into two 16 bit access when DTACK is enabled.

The duration of the access cycle can also be controlled by programmable wait state, that is the configuration in the WSC bit (EIM_CS5H[45:40]). In this case, the WAIT signal is ignored and the access cycle length is fixed by the WSC bit.

Table 5.

Signal Name	Total Number of Pin	Direction of Signal	Function	Connection to MX1
OE#, WE#, IORD#, IOWR#	4	Input	In memory mode, OE is used to read data and WE used to write data to Compact Flash In I/O mode, IORD and IOWR is used instead of OE and WR	Output from MC9328MX1/MXL processor OE and WE signal connect to glue logic
CE1#, CE2#	2	Input	Use to select the card and to indicate whether a byte or word operation is being performed.	CS[5], EB3 and EB2 is connected to glue logic, which output CE1# and CE2# signals
READY (IREQ in I/O mode)	1	Output	Indicate CF card is ready to accept new data transfer	Connect to GPIO PA18 through buffer
RESET	1	Input	Perform a hardware reset to compact flash card, usually asserted only at power up	Connect to GPIO PB17 through buffer
WAIT	1	Output	Signal the host to delay completion of a memory or I/O cycle that is in progress	Connect to DTACK through buffer

1.6.5 Card Detection

The CD1# and CD2# pins are connected to ground on the Compact Flash Card internally. They are used by host to determine Compact Flash Card is fully inserted in the socket.

This two pin is input to a OR gate with pull up resistor. The output of OR gate is connected to GPIO PA20. Card insertion can be detected whenever there is a high to low transaction in GPIO PA20. The power up and reset process can then proceed.

The output is also used to enable the buffer between the DragonBall processor and the Compact Flash interface.

Table 6.

Signal Name	Total Number of Pin	Direction of Signal	Function	Connection to MX1
CD1#, CD2#	2	Output	Used for card insert detection	Connection to MC9328MX1/MXL GPIO PA20 through OR gate

1.6.6 Signals Not Implemented

BVD1 BVD2

These signals are used as battery detect in PCMCIA in memory mode, and are not necessary and supported in Compact Flash specification

In I/O mode, BVD1 is used as Status change signal STSCHG which will assert low to assert host change in the RDY/BSY and write protect states. Same function can be found in Pin Replacement Register of attribute memory.

BVD2 will be used as speaker output signal SPKR for binary audio output from the card in I/O mode.

These signal are not implemented as they have insignificant application for Compact Flash and for the reason of limited number GPIO available.

INPACK

The input acknowledgement signal is used in I/O mode only to indicate the address on address bus is referring to I/O read cycle. This signal is originally used for enabling input data buffer between Compact Flash and the processor. This signal is not implemented as it is not related to the functionality of the Compact Flash interface.

CSEL

The CSEL signal is used for configuring device as master or slave in TrueIDE mode. As True IDE mode is not supported, this signal is not implemented

2 Compact Flash Timing Characteristics

2.1 Timing Requirement for Read Cycle

As EIM do not distinguish between memory read cycle and I/O read cycle, the interface timing is reference from the read cycle timing requirement in both memory mode and I/O mode. The following timing parameters work for both memory mode and I/O mode read cycle and is the reference design timing parameter determined from the host side.

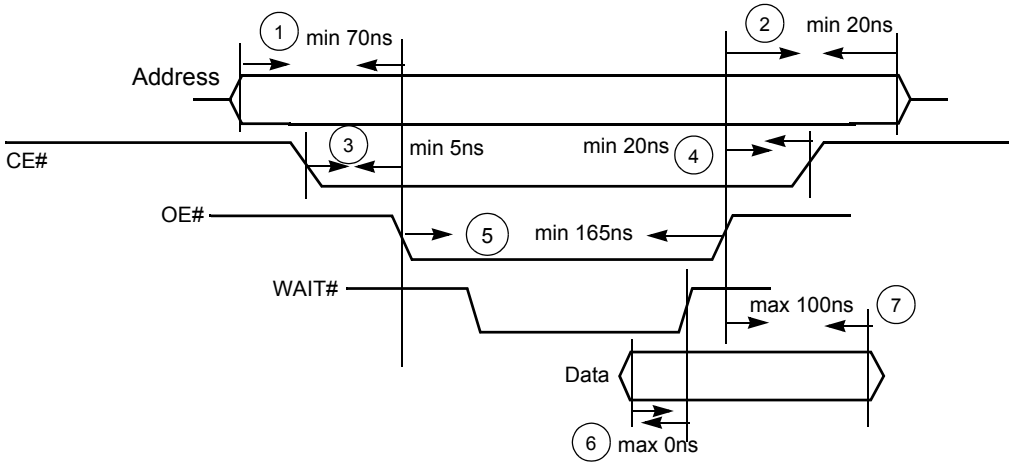


Figure 2.

Table 7.

Timing Item	Item	Symbol	Memory Mode Requirement	I/O Mode Requirement	Reference Design Timing Requirement / Comment
1	Address Setup Time	tsu(A)	min 30ns	min 70ns	min 70ns achieve by glue logic
2	Address Hold Time	th(A)	min 20ns	min 20ns	min 20ns timing match by CS5
3	CE Setup before OE	tsu(CE)	min 0ns	min 5ns	min 5ns achieve by glue logic
4	CE Hold following OE	th(CE)	min 20ns	min 5ns	min 20ns achieve by glue logic
5	IORD Width Time	tw(IORD)		min 165ns	min 165ns signal width programmable by WSC bit in CS5 register/ determined by wait signal
6	Data Setup Time for Wait Release	tv(WT)	min 0ns	min 0ns	timing match by CS5
7	Data Hold following OE	tdis(OE)	max 100ns	min 0ns	buffer is used to ensure enough hold time

2.2 Timing Requirement for Write Cycle

As EIM do not distinguish between memory write cycle and I/O read cycle, the interface timing is reference from the write cycle timing requirement in both memory mode and I/O mode. The following timing parameters work for both memory mode and I/O mode write cycle and is the reference design timing parameter determined from the host side

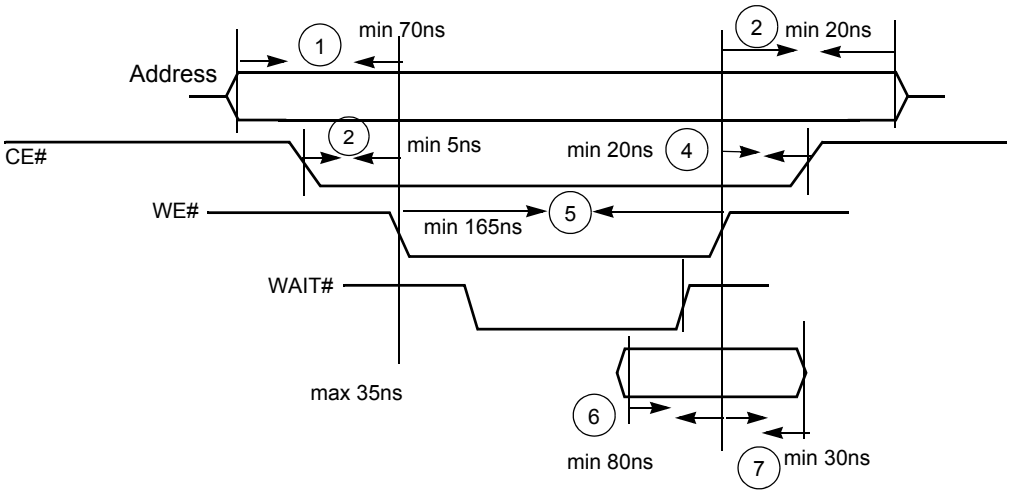


Figure 3.

Table 8.

Timing Item	Item	Symbol	Memory Mode Requirement	I/O Mode Requirement	Reference Design Timing Requirement / Comment
1	Address Setup Time	tsu(A)	min 30ns	min 70ns	min 70ns achieve by glue logic
2	Address Hold Time	th(A)	min 20ns	min 20ns	min 20ns timing match by CS5
3	CE Setup before WE	tsu(CE)	min 0ns	min 5ns	min 5ns achieve by glue logic
4	CE Hold following WE	th(CE)	min 20ns	min 5ns	min 20ns achieve by glue logic
5	IOWR Width Time	tw(IOWR)		min 165ns	min 165ns signal width programmable by WSC bit in CS5 register/ determined by wait signal
6	Data Setup before WE	tv(WT)	min 80ns	min 60ns	min 80ns timing match by CS5
7	Data Hold following WE	tdis(WE)	min 30ns	min 30ns	min 30ns buffer used to ensure enough hold time

2.3 Glue Logic Block Diagram for Setup and Hold Time

NOTE:

Delay 1 should be long enough to ensure enough holdtime for CE signal. (Refer to timing diagram below) Large propagation delay logic gate (HC series) is used to implemented the delay. The required delay can also implement using flipflop, with EIM burst clock as clock input to flipflop. The EIM burst clock can be configured as a standalone clock source by EIM configuration register.

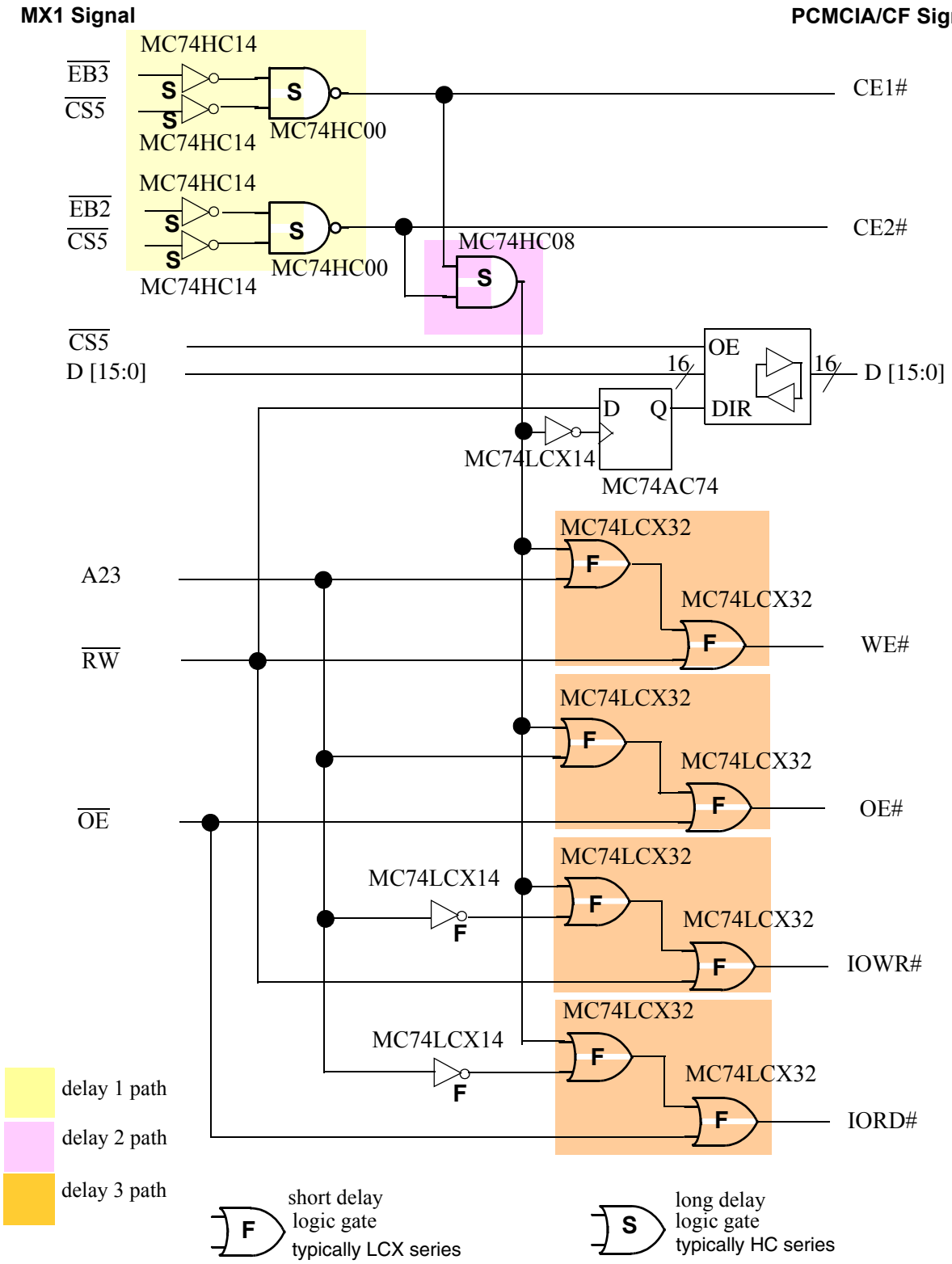


Figure 4.

2.4 Glue Logic Output Timing Diagram - Read Cycle

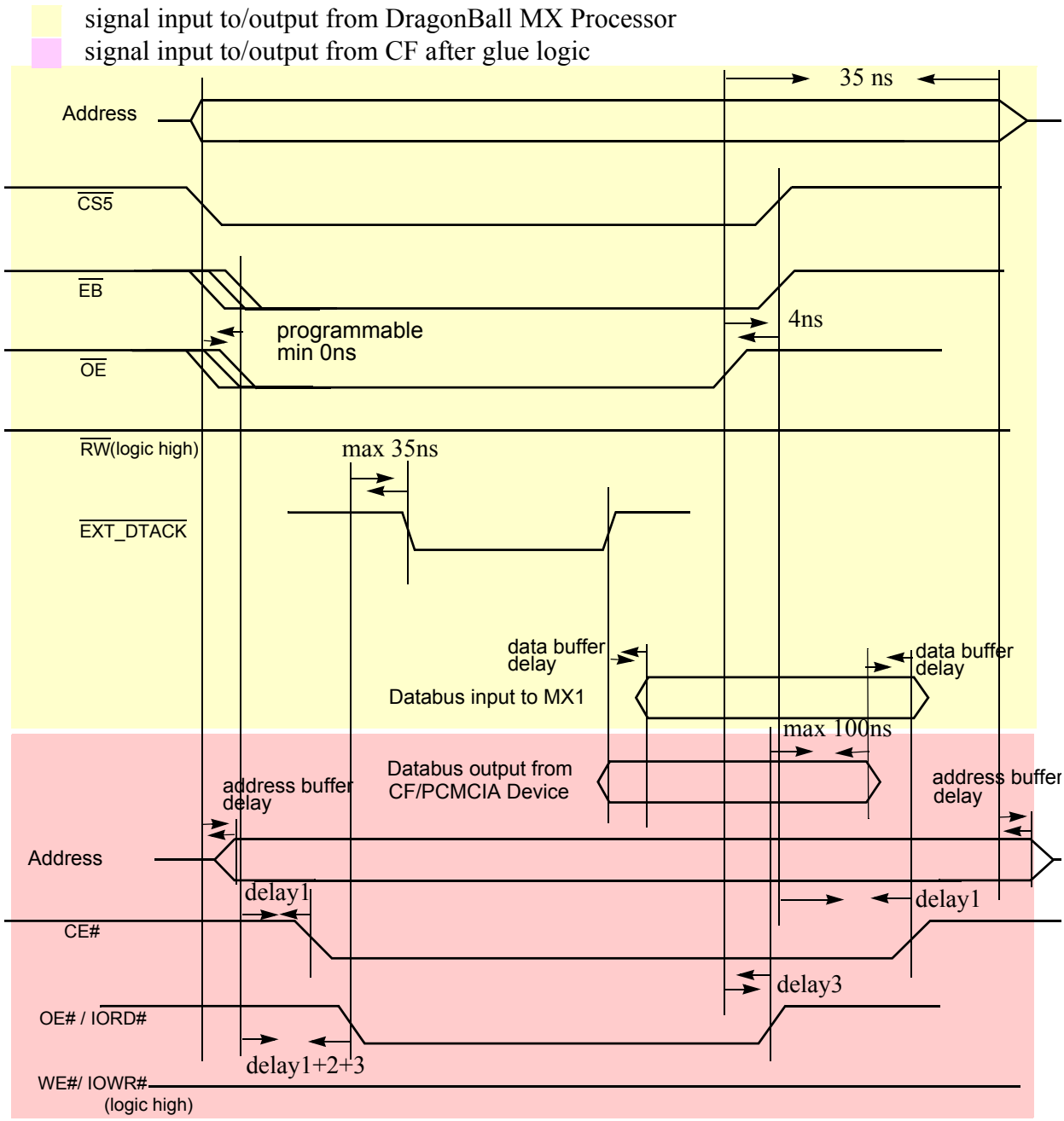


Figure 5.

Table 9.

Setup / hold time requirement	delay path	required delay time
CE setup before OE	delay 1+2+3 - delay 1	> 5ns
CE hold following OE	delay 1 - delay 3 + 4ns	> 20ns

Table 9. (Continued)

Setup / hold time requirement	delay path	required delay time
Address setup time	delay 1+2+3 + OE/EB programmable setup time (by OEA bit in CS5L) - address buffer delay	> 70ns

2.5 Glue Logic Output Timing Diagram - Write Cycle

signal input to/output from DragonBall MX Processor
signal input to/output from CF after glue logic

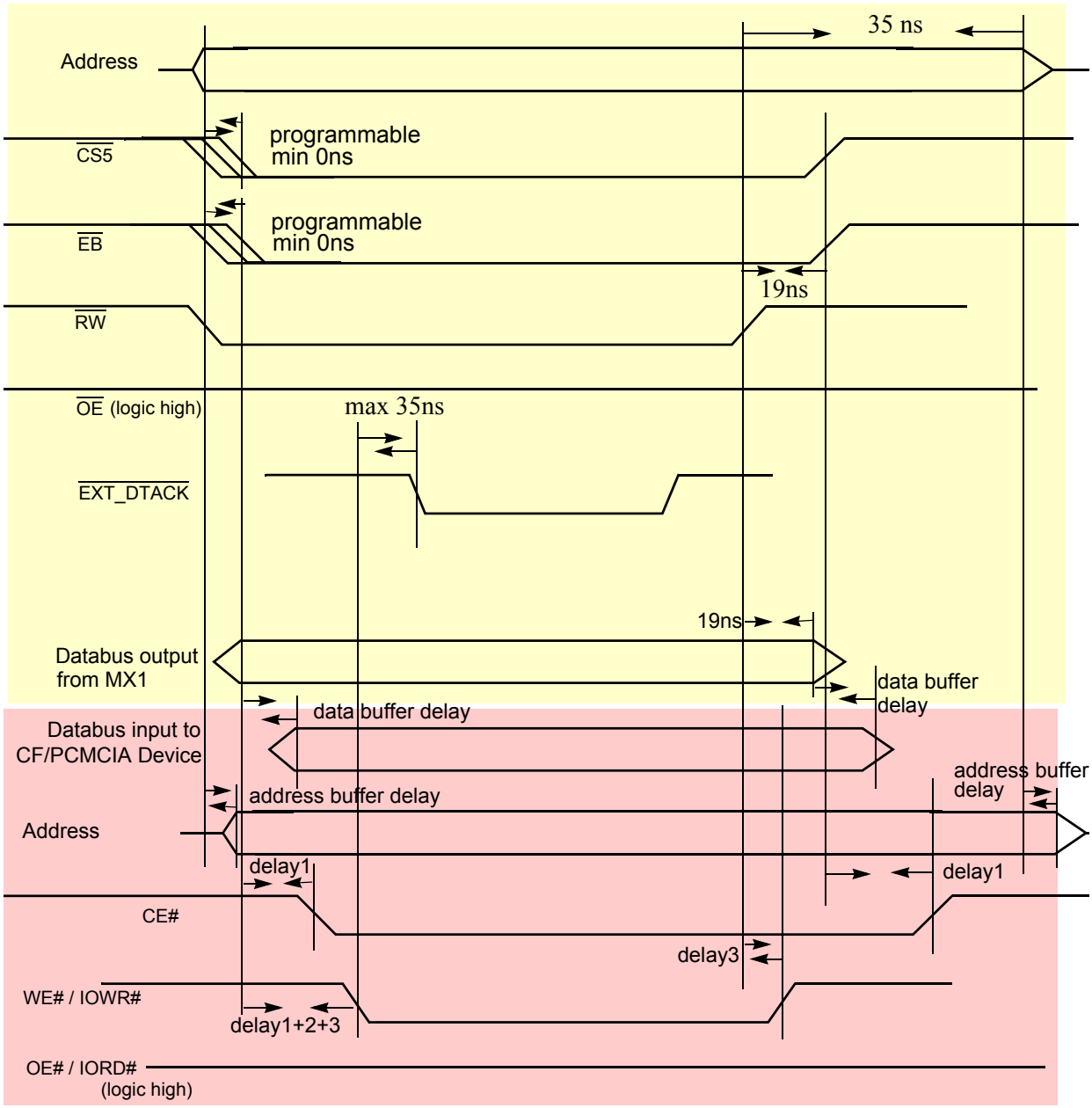


Table 10.

Setup / hold time requirement	delay path	required delay time
CE setup before WE	delay 1+2+3 - delay 1	> 5ns
CE hold following WE	delay 1 - delay 3 + 19ns	> 20ns
Address setup time	CS/EB programmable setup time + delay 1+2+3 - address buffer delay	> 70ns

3 Programming Example

Compact Flash Memory Card use ATA command to perform the read write operation on memory block. Each block of memory have a capacity of 512 bytes. Before access to Compact Flash memory card, various initialization is required.

Initialization

1. EIM Configuration
2. GPIO Configuration
3. Card Detect and Power Up

After initialization, the following test is carried on Compact Flash memory card.

Memory Mode Access

1. Read Card Information Structure
2. Write to a memory block (30H)
3. Read back from memory block (20H)
4. Issue Identify Drive AT command (ECH)

I/O Mode Access

1. Write to a memory block (30H)
2. Read back from memory block (20H)
3. Issue Identify Drive AT command (ECH)

The model of Compact Flash memory card used in testing:

Sandisk 32MB Compact Flash Memory Card

3.1 Memory Address Space

Compact Flash device can be configured as working in memory mode or I/O mode. Address pin A23 is connected for memory mode or I/O mode selection. In memory mode, the address space is divided into attribute memory space and common memory space. REG# signal = 0 will set the address on attribute memory space and REG# signal = 1 will set the address on common memory space. Address pin A22 is connected to REG# signal for attribute memory or common memory selection.

For I/O mode access, the REG# signal should be clear. The address space decoding is listed in the following table.

Table 11.

Address	Memory space
0x 00 0XXX	Memory Mode Access Attribute Memory Space
0x 40 0XXX	Memory Mode Access Common Memory Space
0x 80 0XXX	I/O Mode Access

The lower 11 bit address is the address of the compact flash address bus.

3.2 EIM Configuration

Chip Select 5 of External Interface Module will be connected to Compact Flash, configuration of CS5 is needed before access to compact flash device.

1. Set CSEN to enable Chip Select 5 (EIM_CS5L[0])
2. Configure DSZ to set the Data Bus Size as 16 bit on D0:D15 (EIM_CS5L[10:8])
3. Configure OEA for OE setup time (EIM_CS5L[31:28])
4. Configure EBC (EIM_CS5L[11]), WEA (EIM_CS5L[23:20]) for EB timing
5. Configure WSC for the number of wait states to access compact flash, or enable the DTACK function to support Compact Flash WAIT signal (EIM_CS5H[45:40])
6. Set EIM_CS5H[63] to enable Compact Flash wait support of DTACK signal (when DTACK is enabled)

NOTE:

Setting programmable wait state for access cycle length control is preferred when the card is inserted into the card. The access cycle length for accessing attribute memory is fixed. After reading the Card Information Structure (CIS), EIM can be configured to enable DTACK for wait signal support if wait signal is used as indicated in CIS.

```
*(P_U32) EIM_CS5H = 0x00000F00; // programmable wait state used, WSC = 0x0F
*(P_U32) EIM_CS5L = 0x00000501; // EBC = 0, assert EB for read and write cycle
// DSZ = 5, Data Bus configured as 16 bit on D[15:0]
// CSEN = 1, Enable CS5
```

3.3 GPIO Configuration

3.3.1 GPIO Configuration Procedure

The procedures to configure Port A/B/C/D pin i as GPIO data output are:

- Set bit i of Port A/B/C/D GPIO Data Direction Register (DDIR) to configure as output pin.
- Set bit i of Port A/B/C/D GPIO In Use Register (GIUS)
- The value of data register is selected as an output

If i < 16, set bits [2*i +1] and [2*i] of Output Configuration Register1 (OCR1) as b11

If i >=16, set bits [2*i-32 +1] and [2*i-32] of Output Configuration Register2 (OCR2) as b11

The procedures to configure Port A/B/C/D pin i as GPIO data input are:

- Clear that bit i of Port A/B/C/D GPIO Data Direction Register (DDIR) to configure as input GPIO pin.
- Set bit i of Port A/B/C/D GPIO In Use Register (GIUS)
- If internal pull high is required, set bit i of Port A/B/C/D Pull up Enable Register, Otherwise, bit i should be clear

3.3.2 GPIO Setting Requirement

The following table shows the GPIO signals required in the Compact Flash interface

Table 12.

GPIO	Connection to Compact Flash	GPIO Direction
PA18	READY	Input
PA20	Card Detect Logic Output	Input
PB17	RESET	Output

3.3.3 GPIO Port A Setting

```
*(P_U32) PTA_GIUS = 0x00140000; // Set bit 18 and 20 of PTA_GIUS
*(P_U32) PTA_DDIR = ~(0x00140000); // Clear bit 18 and 20 of PTA_GIUS
*(P_U32) PTA_PUEN = 0x00000000;
```

3.3.4 GPIO Port B Setting

```
*(P_U32) PTB_GIUS =0x00020000; // Set bit 17 of PTB_GIUS
*(P_U32) PTB_DDIR =0x00020000; // Set bit 17 of PTB_GIUS
*(P_U32) PTB_OCR2 = 0x0000000C;// Set bit 2 and 3 of PTB_OCR2
```

3.4 Card Detect and Power Up

When Compact Flash Card is inserted in the socket, GPIO PA 20 will change from 1 to 0. Card insertion can be detected by a negative edge interrupt or polling PA20 for low.

```
*(P_U32) PTB_DR &=~(0x1<<17); //Reset Pin =0
if (    ~(   (*(P_U32) PTA_SSR) | 0xFFEFFFFF)    )
        printf("waiting for card insertion\n");
while (    ~( (*(P_U32) PTA_SSR)|0xFFEFFFFF)){//poll for Card Detect,Quit loop when PA20=1
}
printf("Card inserted\n");
```

After 1 ms of card insertion, a reset pulse should be sent to the card. with a minimum duration of 10us.

```
delay(1500); //delay 1ms after Card Inserted
*(P_U32) PTB_DR |=  0x1<<17; // Set Reset Pin High
delay(15); // Pulse Width at least 1us
*(P_U32) PTB_DR &=~(0x1<<17); //Set Reset Pin Low
```

NOTE:

The delay time may vary depends on how the delay function is implemented and the CPU clock. In above example, system clock is running at 48MHz and implementation of delay loop is:

```
void delay(U16 delaytime)
{
U16 i=0;
for (i=0; i<delaytime; i++) {
}
return;
}
```

After sending the reset pulse, the program should poll for ready signal. When the ready signal is high, the card is ready for access.

```
if ( ~(  (*(P_U32) PTA_SSR) | 0xFFFBFFFF)  ) { //poll for Card Ready, Quit loop when PA18 = 1
printf("Card Initialization ... Busy\n");
}
while ( ~((*(P_U32) PTB_SSR)|0xFFFFBFFF) ) {//poll for Card Ready, Quit loop when PA18 = 1
}
printf("Card is Ready\n\n");
```

The following diagram shows the power up sequence:

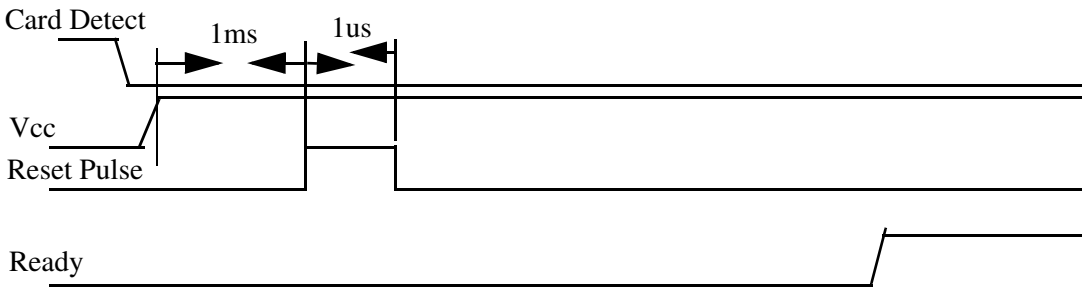


Figure 6.

3.5 Memory Mode Test Program

3.5.1 Read Card Information Structure

Card Information Structure (CIS) contains the information of the Compact Flash device. It is located in even address of attribute memory address 000H - 168H. In address range 034H to 05AH the manufacturer string is stored. The following program code will read the CIS, store it in a variable array and print the manufacturer string to screen.

```
U8 CISdata[180];
for (i=0;i<180;i++){ //copy CIS to CISdata[]
        CISdata[i]=*(P_U8) (CS5_BASE_ADDR+i*2);// CIS can be access in even address only
}
printf("Manufacturer Information Stored in CIS :\n");
for (i=26; i<46 ; i++) {
        if (CISdata[i]!='\0')
        printf("%c", CISdata[i]);
        else printf("\n");
}
```

The following string will be printed to screen.

SunDisk

SDP

5/3 0.6

Other data of CIS in the variable array can be compared with the information on SanDisk Compact Flash Memory Card Product Manual to verify the CIS is read successfully.

3.5.2 Write to Memory Block

Compact Flash memory are NAND flash memory that need ATA Command to perform read write operation. Block access is required for read write access to Compact Flash memory card. The memory space used should be common memory space. The capacity of each block is 512 bytes. To write a 512 bytes of data to a block of memory, The following sequence is required.

1. Write the number of block of data to be access to the sector count register
2. Write the LBA Address to the card
3. Issue 30H command

When this command is accepted, the Compact Flash Memory Card set BSY, then sets DRQ and clear BSY, then wait for the host to fill the sector buffer with the data to be written. DRQ will be clear when 512 bytes of data is written.

Example Program to write 512 byte of data to a memory block at LBA Address 0x0000001

```
U8 tempU8;
U16 i;
//***Set sector count***
*(P_U8) (CS5_BASE_ADDR+0x400002) = 0x1;//sector count =1
//***Set the LBA address of memory block to be written***
*(P_U8) (CS5_BASE_ADDR+0x400003) = 0x1; //LBA [7:0] =1
*(P_U8) (CS5_BASE_ADDR+0x400004) = 0x0; //LBA [15:8:] =0
*(P_U8) (CS5_BASE_ADDR+0x400005) = 0x0; //LBA [23:16:] =0
*(P_U8) (CS5_BASE_ADDR+0x400006) = 0xE0; //LBA [27:24]=0 (lower 4 bit of register)
//Issue write command
*(P_U8) (CS5_BASE_ADDR+0x400007) = 0x30;//issue 30H command for sector write
//Poll for busy bit
tempU8 = *(P_U8) (CS5_BASE_ADDR+0x400007);
while ( tempU8&0x80) {//poll for busy bit (bit 7 of register), quit loop when busy bit =0
        tempU8 = *(P_U8) (CS5_BASE_ADDR+0x400007);
}
//Write data to data buffer until DRQ is clear
for  (i=0;(tempU8&0x08)==0x08;i++) {
        *(P_U16) (CS5_BASE_ADDR+0x400008)=(0xAA00+i);// write 2 byte of data to data buffer
        tempU8=*(P_U8)  (CS5_BASE_ADDR+0x400007);// poll for DRQ (bit 3 of register)
}
printf("Finish Writing\n");
```

3.5.3 Read from Memory Block

To read back from a memory block, the following sequence is required:

1, Write the number of block of data to be access to the sector count register

2, Write the LBA Address to the card

3, Issue 20H command

When this command is accepted, the Compact Flash Memory Card set BSY, then sets DRQ, put the sector of data in buffer and clear BSY, then wait for the host to read the sector data from buffer. DRQ will be clear when 512 bytes of data is read.

Example Program to read 512 byte of data to a memory block at LBA Address 0x0000001

```
U8 tempU8;
U16 i, U16data[256];
//Set sector count
```

```
*(P_U8) (CS5_BASE_ADDR+0x400002) = 0x1;//sector count =1
//Set the LBA address of memory block to be read
*(P_U8) (CS5_BASE_ADDR+0x400003) = 0x1; //LBA [7:0] =1
*(P_U8) (CS5_BASE_ADDR+0x400004) = 0x0; //LBA [15:8] =0
*(P_U8) (CS5_BASE_ADDR+0x400005) = 0x0; //LBA [23:16] =0
*(P_U8) (CS5_BASE_ADDR+0x400006) = 0xE0; //LBA [27:24] =0 (lower 4 bit of register)
//Issue read command
*(P_U8) (CS5_BASE_ADDR+0x400007) = 0x20;//issue 20H command for sector read
//Poll for busy bit
tempU8 = *(P_U8) (CS5_BASE_ADDR+0x400007);
while ( tempU8&0x80) {//poll for busy bit (bit 7 of register), quit loop when busy bit =0
        tempU8 = *(P_U8) (CS5_BASE_ADDR+0x400007);
}
//Read data from buffer into U16data[] until DRQ is clear
for  (i=0;(tempU8&0x08)==0x08;i++) {
        U16data[i]=*(P_U16) (CS5_BASE_ADDR+0x400008);// read 2 byte of data from buffer
        tempU8=*(P_U8) (CS5_BASE_ADDR+0x400007);// poll for DRQ (bit 3 of register)
}
printf("Finish Reading\n");
```

3.5.4 Identify Drive Command

The identify drive command will put information about the compact flash memory card into card buffer. The host can get this information in the same way as read command:

1, Issue 0xEC command

When this command is accepted, the Compact Flash Memory Card set BSY, then sets DRQ, put the information data in buffer and clear BSY, then wait for the host to read the information data from buffer. DRQ will be clear when 512 bytes of data is read.

Example Program to issue identify drive command:

```
U8 tempU8;
U16 i, U16data[256];
//Issue identify drive command
*(P_U8) (CS5_BASE_ADDR+0x400007) = 0xEC; // issue EC command for Identify Drive
//Poll for busy bit
tempU8 = *(P_U8) (CS5_BASE_ADDR+0x400007);
while ( tempU8&0x80) {//poll for busy bit (bit 7 of register), quit loop when busy bit =0
        tempU8 = *(P_U8) (CS5_BASE_ADDR+0x400007);
}
//Read data from buffer into U16data[] until DRQ is clear
for  (i=0;(tempU8&0x08)==0x08;i++) {
        U16data[i]=*(P_U16) (CS5_BASE_ADDR+0x400008);// read 2 byte of data from buffer
        tempU8=*(P_U8) (CS5_BASE_ADDR+0x400007);// poll for DRQ (bit 3 of register)
}
printf("Finish Reading\n");
```

For information read back from identify drive command, Word Address 27-46 is the model number in ASCII (Left Justified) Big Endian Byte Order in Word (32 bit). The following example code will print the model on screen.

```
for (i=27; i<47; i++) { //Print the CF Model string from Drive Identify Data
        printf("%c%c", U16data[i*2+1], U16data[i*2] );
}
        printf("\n");
```

The following model name is found on the testing Sandisk Compact Flash by identify drive command:

Sandisk SDCFB-32

The definition of the information get from identify drive command is listed in Sandisk Compact Flash Memory Card Product Manual.

3.6 I/O Mode Test Program

3.6.1 Configure the Card to Work in I/O Mode

Compact Flash memory card can be configure as I/O mode in the configuration option register.The configuration option register is a 8 bit register located in attribute memory space address 0x200. To configure the Compact Flash as I/O mode, the lower 6 bit of the configuration option register should be set as 0x01

// configure compact flash device to I/O mode

(P_U8) (CS5_BASE_ADDR+0x200) = ((P_U8) (CS5_BASE_ADDR+0x200) &0xC0) +0x1;

3.6.2 Write to Memory Block

The procedure for writing a block of memory in I/O mode is the same as in memory mode. The only different is that correct address space should be used. Procedure for writing to a block of memory:

1, Write the number of block of data to be access to the sector count register

2, Write the LBA Address to the card

3, Issue 30H command

Example Program to write 512 byte of data to a memory block at LBA Address 0x0000001

```
U8 tempU8;
U16 i;
//Set sector count
*(P_U8) (CS5_BASE_ADDR+0x800002) = 0x1;//sector count =1
//Set the LBA address of memory block to be written
*(P_U8) (CS5_BASE_ADDR+0x800003) = 0x1; //LBA [7:0] =1
*(P_U8) (CS5_BASE_ADDR+0x800004) = 0x0; //LBA [15:8] =0
*(P_U8) (CS5_BASE_ADDR+0x800005) = 0x0; //LBA [23:16] =0
*(P_U8) (CS5_BASE_ADDR+0x800006) = 0xE0; //LBA [27:24]=0 (lower 4 bit of register)
//Issue write command
*(P_U8) (CS5_BASE_ADDR+0x800007) = 0x30;//issue 30H command for sector write
//Poll for busy bit
tempU8 = *(P_U8) (CS5_BASE_ADDR+0x800007);
while ( tempU8&0x80) {//poll for busy bit (bit 7 of register), quit loop when busy bit =0
        tempU8 = *(P_U8) (CS5_BASE_ADDR+0x800007);
}

//Write data to data buffer until DRQ is clear
```

```
for  (i=0;(tempU8&0x08)==0x08;i++) {
        *(P_U16) (CS5_BASE_ADDR+0x800008)=(0xAA00+i);// write 2 byte of data to data buffer
        tempU8=*(P_U8)  (CS5_BASE_ADDR+0x800007);// poll for DRQ (bit 3 of register)
}
printf("Finish Writing\n");
```

3.6.3 Read from Memory Block

The procedure for reading a block of memory in I/O mode is the same as in memory mode. The only different is that correct address space should be used. Procedure for writing to a block of memory:

1, Write the number of block of data to be access to the sector count register

2, Write the LBA Address to the card

3, Issue 20H command

Example Program to read 512 byte of data to a memory block at LBA Address 0x0000001

```
U8 tempU8;
U16 i, U16data[256];
//Set sector count
*(P_U8) (CS5_BASE_ADDR+0x800002) = 0x1;//sector count =1
//Set the LBA address of memory block to be read
*(P_U8) (CS5_BASE_ADDR+0x800003) = 0x1; //LBA [7:0] =1
*(P_U8) (CS5_BASE_ADDR+0x800004) = 0x0; //LBA [15:8] =0
*(P_U8) (CS5_BASE_ADDR+0x800005) = 0x0; //LBA [23:16] =0
*(P_U8) (CS5_BASE_ADDR+0x800006) = 0xE0; //LBA [27:24]=0 (lower 4 bit of register)
//Issue read command
*(P_U8) (CS5_BASE_ADDR+0x800007) = 0x20;//issue 20H command for sector read
//Poll for busy bit
tempU8 = *(P_U8) (CS5_BASE_ADDR+0x800007);
while ( tempU8&0x80) { //poll for busy bit (bit 7 of register), quit loop when busy bit =0
        tempU8 = *(P_U8) (CS5_BASE_ADDR+0x800007);
}
```

```
//Read data from buffer into U16data[] until DRQ is clear
for  (i=0;(tempU8&0x08)==0x08;i++) {
```

```
        U16data[i]=*(P_U16) (CS5_BASE_ADDR+0x800008);// read 2 byte of data from buffer
        tempU8=*(P_U8) (CS5_BASE_ADDR+0x800007);// poll for DRQ (bit 3 of register)
}
printf("Finish Reading\n");
```

3.6.4 Identify Drive Command

The procedure for issuing identify drive command in I/O mode is the same as in memory mode. The only different is that correct address space should be used.

Example Program to issue identify drive command:

```
U8 tempU8;
U16 i, U16data[256];
//Issue identify drive command
*(P_U8) (CS5_BASE_ADDR+0x800007) = 0xEC; // issue EC command for Identify Drive
//Poll for busy bit
tempU8 = *(P_U8) (CS5_BASE_ADDR+0x800007);
while ( tempU8&0x80) { //poll for busy bit (bit 7 of register), quit loop when busy bit =0
        tempU8 = *(P_U8) (CS5_BASE_ADDR+0x800007);
}
//Read data from buffer into U16data[] until DRQ is clear
for  (i=0;(tempU8&0x08)==0x08;i++) {
        U16data[i]=*(P_U16) (CS5_BASE_ADDR+0x800008);// read 2 byte of data from buffer
        tempU8=*(P_U8) (CS5_BASE_ADDR+0x800007);// poll for DRQ (bit 3 of register)
}
printf("Finish Reading\n");
```

NOTES

HOW TO REACH US:

USA/EUROPE/LOCATIONS NOT LISTED:

Motorola Literature Distribution
P.O. Box 5405, Denver, Colorado 80217
1-800-521-6274 or 480-768-2130

JAPAN:

Motorola Japan Ltd.; SPS, Technical Information Center,
3-20-1, Minami-Azabu, Minato-ku, Tokyo 106-8573, Japan
81-3-3440-3569

ASIA/PACIFIC:

Motorola Semiconductors H.K. Ltd.; Silicon Harbour Centre,
2 Dai King Street, Tai Po Industrial Estate, Tai Po, N.T.,
Hong Kong
852-26668334

HOME PAGE:

http://motorola.com/semiconductors

AN2417/D

Engineering Draft / Preliminary