

IP-CYCLONE II- PIO

INDUSTRY PACK

ALTERA

HARDWARE REFERENCE MANUAL

Revision 1.3

August 2008

This Document shall not be duplicated, nor its contents used for any purpose, unless express permission has been granted in advance.

ALPHI TECHNOLOGY CORP.

1898 E. SOUTHERN Avenue

Tempe, AZ 85282 USA

Tel : (480) 838 - 2428

Fax: (480) 838 - 4477

TABLE OF CONTENTS

1	Introduction:	4
1.1	Functional description:	4
1.2	BASIC BLOCK DIAGRAM:	4
1.3	ELECTRICAL BLOCK DIAGRAM:	5
2	I/O BLOCK DIAGRAM	6
3	ADDRESS MAP:	7
3.1	IDSPACE:	7
3.2	IOSPACE:	8
3.2.1	MAX3128A- EP2Cxx dual addressing registers	8
3.2.2	IDEPLD EP2Cxx identification part	9
3.2.2.1	EPREV MAX3128A revision control	9
3.2.2.2	ICR Initialization Control Register	10
3.2.2.3	ISR Init Status Register	11
3.2.3	EP2Cxx ALTERA PROGRAMMING	11
3.3	INTSPACE	12
3.3.1	IVRx Interrupt Vector Register	12
3.4	MEMORY SPACE	12
3.4.1	DPR Timing:	12
3.5	SRAM interconnection with EP2Cxx	13
3.6	EP2Cxx Memory Request Lines:	14
3.7	IP bus interconnection with EP2Cxx	15
4	EP2Cxx I/O ASSIGNMENT	16
5	CYCLONE I/O LINES	16
5.1.1	FST3245 SWITCHES Outputs	17
5.2	Driver	18
5.2.1	DE[23..0] Outputs	18
5.2.2	DO[23..0] Outputs	19
5.2.3	nSLO[23..0] Outputs	20
5.3	RECEIVER	21
5.3.1	RI[23..0] Inputs	21
5.3.2	nRE[23..0] Outputs	22
5.3.3	TE[23..0] Outputs	23
5.3.4	I/O Pin	24
6	EP2Cxx IO Control line ASSIGNMENT	25
7	JUMPERS AND CONNECTORS LOCATION	25
7.1	Jumper's or connector's description:	25
7.1.1	J3 description	26
7.1.2	J4 IP strobe generation	26
8	Connectors Description:	26

8.1	IP bus interface P1.....	27
8.2	I/O Port P2	28
9	CONFIGURATION DEVICES: SERIAL EEPROM.....	28
10	PSA Passive serial asynchronous programming	29
11	EXTERNAL CLOCK FOR EP2Cxx	29
12	Timing between MAX3128A AND EP2Cxx	30
13	Additional timing	31
Figure 1.1: IP- Cyclone II-PIO BLOCK DIAGRAM		5
Figure 2.1: RS-485 Block DIAGRAM		6
Table 3-1 IDSEL0 SPACE byte content		7
Table 3-1 IOSPACE MAP		9
Table 3-2 SRAM Address and Data to EP2Cxx connection		14
Table 3-3 SRAM Data to EP2Cxx connection		14
Table 7-4 Jumper Description		25
Table 7-5 J3 Description		26
Table 8-6 Connectors Description		26
Table 8-1 P1 IP BUS connector		27
Table 8-1 P2 I/O connector		28

History :

Rev 1.0:

Release

Rev 1.2 :

Change page 27,28 Jumpers description

Correct page 32 Configuration Devices

Rev 1.3 :

Change from LVDS reference to RS-485

Add device used on board page 4

Correct Basic Block Diagram

Change pin location for RI16,RI17,RI18,RI19,RI22,RI23

Add Jumpers Location

1 Introduction:

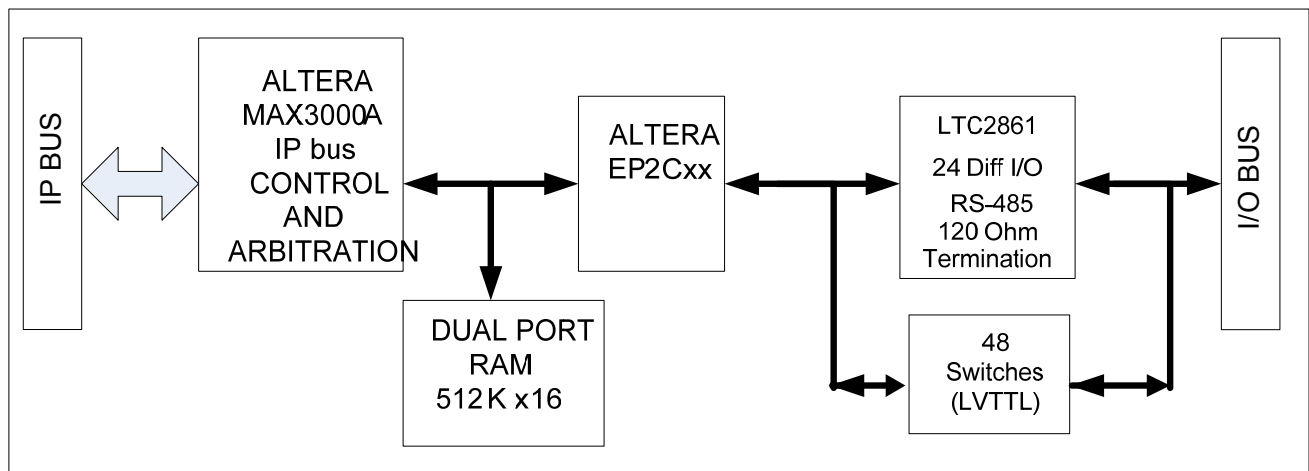
1.1 Functional description:

The IP-CYCLONE II-PIO module is populated with a Cyclone II EP2C20F484 C7 embedded programmable FPGA. Another PLD from Altera (MAX3128A) is used to provide all the timings and interface between the IPBUS, Cyclone II EP2Cxx chip and the 1 Mbyte of Dual Access Memory.

Key Features are:

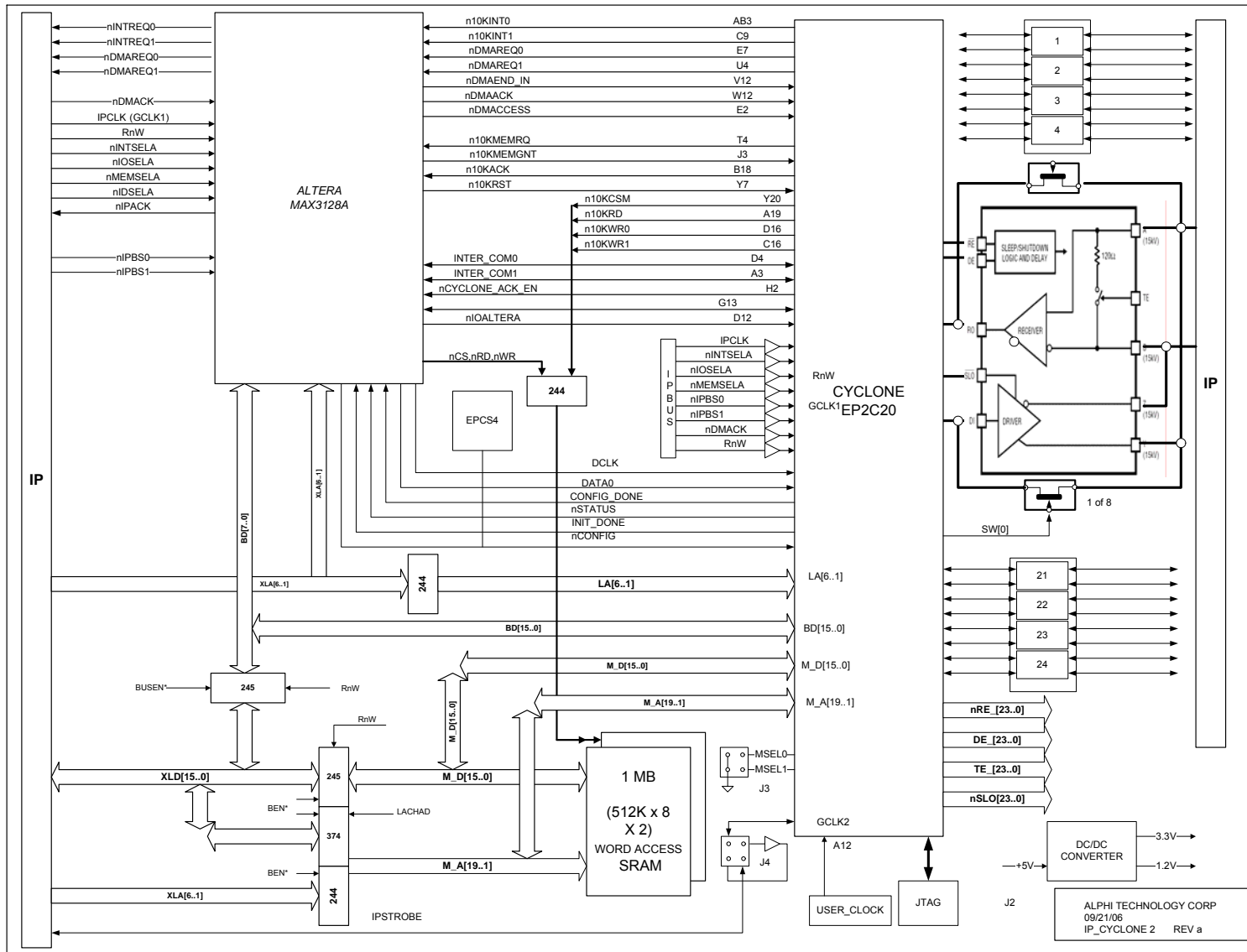
- Up to 24 RS-485 driver/receiver with disable option
- 48 TTL I/O in groups of 8
- Each line can be selected as an I/O line within a group
- Up to 1Mbyte of Dual Ported SRAM.
- 8 or 32 MHz IP clock.
- 2 interrupt lines and 2 DMA request lines.
- IPBus Configuration.
- EEPROM Configuration using EEPROM: EPCS4.
- JTAG configuration
- External oscillator
- A **MAX3128A** chip is used for all timing related to the IPBUS, DPR, FLEXEP2Cxx.
- A CYCLONE2 EP2Cxx fpga

1.2 BASIC BLOCK DIAGRAM:



The IP-CYCLONE II-PIO is divided into 5 blocks:

- 1- **A factory-programmed MAX3128A FPGA** is used for the control of the IP bus and arbitration of the Dual Port Ram.
- 2- **The Dual Port Ram** is a 512k of X16 SRAM that can be written to and read from by the IP bus and the EP2Cxx.
- 3- **The Cyclone II EP2Cxx** is programmable by the user.
- 4- **24 RS-485 driver/receiver** with programmable termination resistance and disable.
- 5- **6 Programmable switches** for TTL I/O by-passing the RS-485.



1.3 ELECTRICAL BLOCK DIAGRAM:

Figure 1.1: IP- Cyclone II-PIO BLOCK DIAGRAM

2 I/O BLOCK DIAGRAM

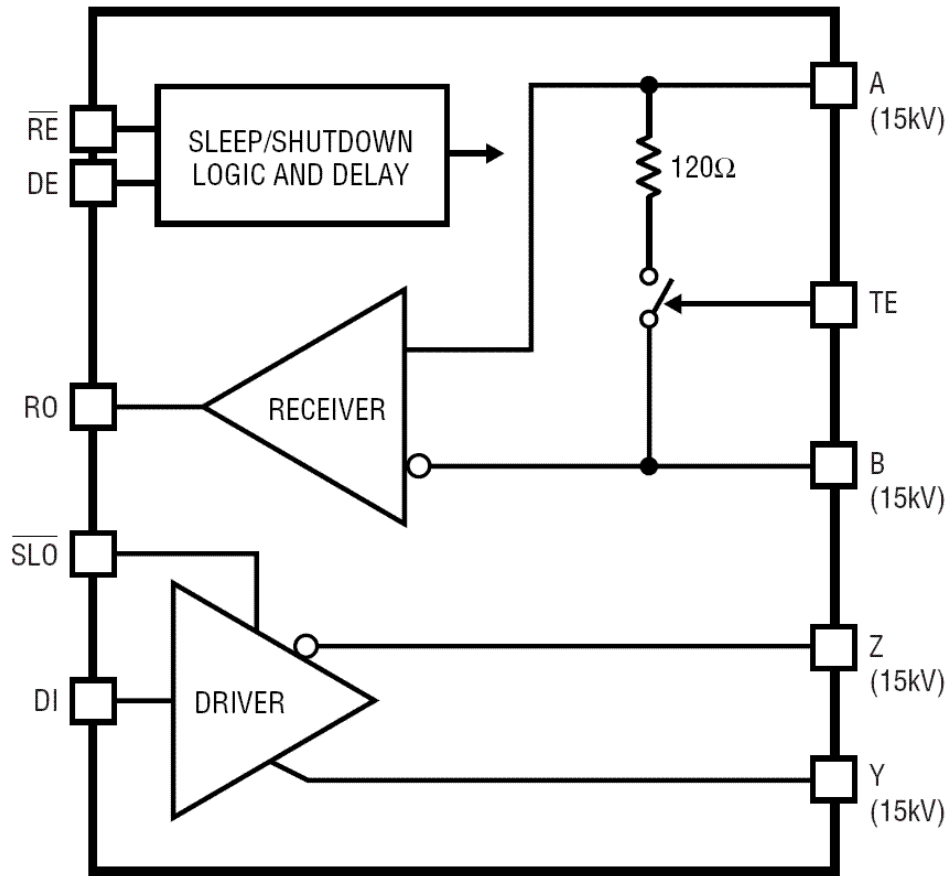


Figure 2.1: RS-485 Block DIAGRAM

3 ADDRESS MAP:

The IP-CYCLONE II-PIO module uses the three available spaces defined in the Industry Pack specifications.

3.1 IDSPACE:

Up to 32 bytes of registered data provides information about the module to the user. The lower address contains data related to the type of module, revision, etc.

Only ODD addresses are valid in byte read mode.

ID space address	Description	Value
\$01	ASCII "I"	\$49
\$03	ASCII "P"	\$50
\$05	ASCII "A"	\$41
\$07	ASCII "H"	\$48
\$09	Manufacturer identification	\$11
\$0B	Module type	\$25
\$0D	Revision module	\$0A
\$0F	Reserved	\$00
\$11	Driver ID, low byte	\$00
\$13	Driver ID, high byte	\$00
\$15	Number of bytes used	\$0A
\$17	CRC	
\$19-\$3F	User space	

Table 3-1 IDSEL0 SPACE byte content

I/O	NAME	REGISTER	TYPE	R/W
\$00	PLD Identification	IDEPLD	Byte	R
\$02	Revision Identification	EPREV	Byte	R
\$04	Init control register	ICR	Byte	R/W
\$06	Init status register	ISR	Byte	R
\$08	EP2Cxx Programming	PROG CYCLONE	Byte	W
\$78	Interrupt Vector Register #0	IVR0-1	Byte	R/W

Table 3-1 IOSPACE MAP

3.2.2 IDEPLD EP2Cxx identification part

Address: IOSPACE + \$00

Actual content: \$02

A read at this location will identify which Altera Cyclone II EP2Cxx FPGA is on the board.

Value	Part #
\$02	EP2C20
\$04	EP2C50

3.2.2.1 EPREV MAX3128A revision control

Address: IOSPACE + \$02

Actual content: \$80

A read at this location will identify the revision of the MAX3128A.

The Bit #7 (when set to a "1") is used to identify if the MAX3128A controller has control of the local bus to program the EP2Cxx. At the end of the programming the signal CONFIG_DONE resets this bit. However it is possible to switch from programming mode to user mode.

A write operation at the I/O base address \$0 (an address that the user can allocate as a read only ID register) into the Altera EP2Cxx and will set the bit #7 to "1" 'switching to MAX3128A programming mode.

Writing \$1 at the I/O base address + \$2 which corresponds to the EPREV register returns the EP2Cxx to user control.

Value	Revision
\$00	Rev. A
\$01	Rev. B
\$02	Rev. C

3.2.2.2 ICR Initialization Control Register

Address: IOSPACE + \$04

BD03	BD02	BD01	BD00
-10K_RST	-	nCONFIG	nCONFIG_EN

BD07	BD06	BD05	BD04
DEV_CLRn	DEV_OE	DEV_CLRn_EN	DEV_OE_EN

Bit 0: nCONFIG_EN:

This bit enables a tri-state buffer that controls the nCONFIG line of the EP2Cxx. During power-on tri-state mode is enabled

Bit 1: nCONFIG:

This bit is used to control the nCONFIG line of the EP2Cxx, and has a 10kΩ pull-up resistor. To set it to "0" after programming you must first disable nCONFIG_EN so that the 10kΩ pull-up resistor take over to maintain the nCONFIG line high then set the nCONFIG bit to "0" else a negative pulse will disable the functionality of the board.

Bit 2: Not used

Bit 3: 10K_RST

When this bit is set to "1" the 10K_RST line is active low.
The ipreset also pull this line low when active.

Bit 4: DEV_OE_EN:

This bit enables a tri-state buffer that controls the DEV_OE line of the EP2Cxx

Bit 5: DEV_CLR_EN:

This bit enables a tri-state buffer that controls the DEV_CLRn line of the EP2Cxx

Bit 6: DEV_OE:

This bit can be used to control the DEV_OE line of the EP2Cxx

Bit 7: DEV_CLRn:

This bit can be used to control the DEV_CLRn line of the EP2Cxx

Note: DEV_OE AND DEV_CLRn will have to be enabled before compilation.

3.2.2.3 ISR Init Status Register

Address: IOSPACE + \$06

This Register provides the status of EP2Cxx programming lines

BD03	BD02	BD01	BD00
SHIFT_STAT	CONF_DONE	nSTATUS	nCONFIG
BD07	BD06	BD05	BD04
Msel1	Msel0	INIT_DONE	VCC

Bit 1: nCONFIG:

This bit when set indicates that the EP2Cxx Configuration started. Upon Reset this bit is set high by a pull-up resistor to allow multiple possibilities of programming, else this line is low in theory.

Bit 1: nSTATUS

Goes high 30-40µ after nCONFIG is enabled, returns to a low status if it detects an error

Bit 2: CONF_DONE:

This bit indicated that the EP2CXX has been programmed correctly, this bit will not enable if any errors were encountered while programming or if the chip is not properly programmed. Errors will generally be easier detected by monitoring nSTATUS than simply checking CONF_DONE

Bit 3: THIS BIT IS “1” during the transfer of serialized data to the CYCLONE.

Wait until back to “0” to send another byte.

Bit 4: VCC

Bit 5: INIT_DONE

/~INIT_DONE, goes low at the end of the initialization temporarily and then goes high after around 40µ

Bit 7, 6: MSEL[1..0]

MSEL[1..0]	Description
00	JTAG or ACTIVE SERIAL (EEPROM)
01	Passive SERIAL ASYNCHRONOUS (Host programming through IPBUS .

3.2.3 EP2Cxx ALTERA PROGRAMMING

Address: IOSPACE + \$08

The EP2Cxx can be programmed in **PSA (parallel serial asynchronous mode)** by writing at the address IOSPACE + \$8. Upon receiving a byte, the bits are sent by a state machine one at a time with the corresponding clock. The lower bit is sent first. The DCLK is the IPCLK.

Each byte is transferred in ~ 1 μ S at 32 MHZ and ~ 4 μ S at 8MHZ.
Bit # 3 (SHIFT_STAT) from the ISR register provides a status of the transfer. It is "1" while the serialization is active.
An example of PSA programming sequence is provided at the end of this manual.

3.3 INTSPACE

When the EP2Cxx has an interrupt pending the carrier module can read the IVR register that has been programmed by the carrier. The ATC_EP2Cxx supports the two interrupts. Interrupt vector is transferred from the EP2Cxx through a buffer to the IPbus. Upon receiving an interrupt cycle (INTSELA) an Interrupt vector register is provided. Each interrupt is routed directly to the Ipbus through the MAX3128A.

3.3.1 IVRx Interrupt Vector Register

Address: IOSPACE + \$78

This eight bit register located at address IOSPACE + \$78 can be read and written by the host. The upper 6 bits of the vector are provided from this register when the host performs an INTSPACE cycle. Bit # 0 is set to "0" for interrupt # 0 and Bit # 1 is set to "0" for interrupt # 1.

3.4 MEMORY SPACE

Up to 1Mbytes of SRAM is mapped into the Memory space. The SRAM can be set to be dual-access SRAM by the IP interface using the Memory space or by the EP2Cxx.

The MAX3128A chip provides arbitration between the DPR accessed by the I/O connector through the EP2Cxx and the IP bus.

An example module to have the EP2Cxx accessing the Dual Port Ram is provided with timing diagram. Address, Data lines and controls lines need to be connected to I/O lines for the module to perform an access to the DPR. The MAX3128A provides arbitration and control lines for the DPR. Some interconnection lines are provided between the two Altera for further application...
Memory space above \$100000 has been decoded to give the user possibilities to have internal memory into the cyclone. IP acknowledge is normally provided by the MAX3128A, however the CYLONE can be programmed to disable the MAX 3128A's acknowledges and provide its own. Refer to example provided for this possibility.

3.4.1 DPR Timing:

Figure 1 below shows the arbitration timing between an IP BUS access to the DPR and an EP2Cxx access to the DPR. The IP BUS gets the DPR first, and then the EP2Cxx, the following IP BUS DPR access has to wait until the EP2Cxx has finished the cycle. All the arbitration is made by the MAX3128A with IPCLK as reference.

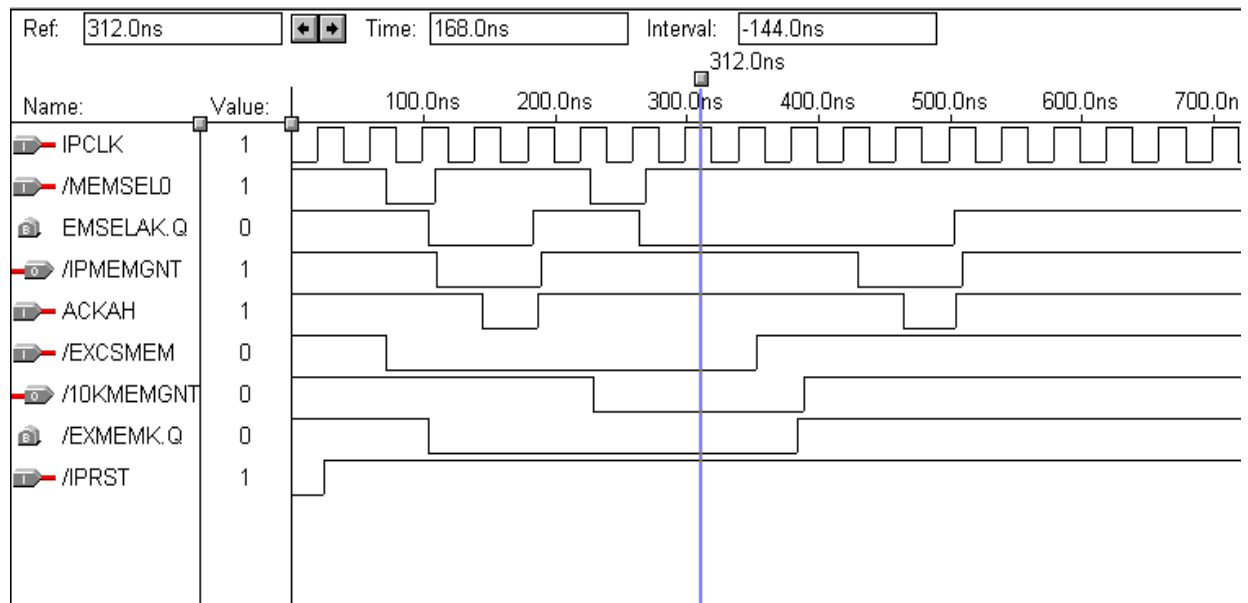


Fig 1. Sample IPBUS/EP2Cxx access arbitration

3.5 SRAM interconnection with EP2Cxx

A separation is provided between the IP bus connected to the EP2Cxx and the M_d bus connected to the dual ported memory providing faster access. The IP bus can access concurrent registers located within the EP2Cxx while the EP2Cxx is accessing the Dual shared memory.

SRAM ADDRESS	EP2Cxx PIN	Description
M_A[1]	PIN_G18	Lower Address Line
M_A[2]	PIN_R22	
M_A[3]	PIN_U19	
M_A[4]	PIN_D21	
M_A[5]	PIN_E20	
M_A[6]	PIN_G20	
M_A[7]	PIN_D22	
M_A[8]	PIN_U20	
M_A[9]	PIN_V20	
M_A[10]	PIN_W22	
M_A[11]	PIN_R17	
M_A[12]	PIN_F20	
M_A[13]	PIN_T18	
M_A[14]	PIN_W21	
M_A[15]	PIN_U18	
M_A[16]	PIN_V21	
M_A[17]	PIN_R21	
M_A[18]	PIN_N21	
M_A[19]	PIN_G17	Higher Address Line

Table 3-2 SRAM Address and Data to EP2Cxx connection

SRAM DATA	EP2Cxx PIN	Description
M_D[0]	PIN_L18	Lower Data Line
M_D[1]	PIN_K22	
M_D[2]	PIN_C17	
M_D[3]	PIN_J17	
M_D[4]	PIN_F21	
M_D[5]	PIN_L19	
M_D[6]	PIN_K21	
M_D[7]	PIN_K20	
M_D[8]	PIN_J18	
M_D[9]	PIN_H17	
M_D[10]	PIN_C19	
M_D[11]	PIN_D19	
M_D[12]	PIN_E21	
M_D[13]	PIN_C22	
M_D[14]	PIN_E19	
M_D[15]	PIN_C18	Higher Data Line

Table 3-3 SRAM Data to EP2Cxx connection**3.6 EP2Cxx Memory Request Lines:**

Signal Name	EP2Cxx PIN	Description	Type
n10KMEMGNT	PIN_M2	EP2Cxx receive Access Grant to the DPR	Input
n10KMEMRQ	PIN_AB6	EP2Cxx Memory Request. Must stay low until end of cycle. Re-sync with IPCLK (GCLK1) before arbitration.	Output
n10KACK	PIN_B18	EP2Cxx Unused Acknowledge, See Note.	Output
n10KCSM	PIN_Y20	EP2Cxx Chip Select Memory.	Output
n10KRD	PIN_A19	EP2Cxx Read signal for both DPR.	Output
n10KWR0	PIN_D16	EP2Cxx Write signal for DPR0.	Output
n10KWR1	PIN_C16	EP2Cxx Write signal for DPR1.	Output
nIOALTERA	PIN_D12	Enable for memory above \$100000 or I/O space +\$7E-\$7F	Input
IP_CYCLE[2..0]	PIN_B17, D14, E15	Representation of the IP_cycle in progress	Input
IP_WRITE	PIN_R18	Active high during write cycle. Signal is terminated when IPACK is generated.	Input

IP_CYCLE [2..0]	Type	Decoded by MAX3128A
0.0.0	None	Y
0.0.1	ID_cycle	Y
0.1.0	INT_cycle	Y
0.1.1	IO_cycle	Y
1.0.0	MEM_cycle	Y
1.0.1	DMA_IO_0_cycle	N
1.1.0	DMA_IO_1_cycle	N
1.1.1	DMA_MEM_cycle	N

Note: The normal source of acknowledge (IPACK) for the Ipbus is the MAX3128A, however, by enabling the line “ncyclone_ack_en” (low) the CYCLONE2 can be the source of the acknowledge instead.

Length of the signal n10kack should be one clock (Gclk1).

3.7 IP bus interconnection with EP2Cxx

Signal Name	EP2Cxx PIN	Description	Type
nDMAREQ0	PIN_E7	IP DMA REQ0	Output
nDMAREQ1	PIN_U4	IP DMA REQ2	Output
nINTREQ0	PIN_AB3	IP INT. REQ 0	Output
nINTREQ1	PIN_C9	IP INT. REQ 1	Output
nBS0	PIN_E12	DATA STROBE 0	Input
nBS1	PIN_B12	DATA STROBE 1	Input
RnW	PIN_A18	IP BUS READ/WRITE	Input
nRESET	PIN_D15	IP BUS RESET	Input
GCLK1	PIN_M1	IP BUS CLOCK 8/32MHZ	Input
A[1]	PIN_T21	IP BUS ADDRESS LINE	Input
A[2]	PIN_T22	IP BUS ADDRESS LINE	Input
A[3]	PIN_AA16	IP BUS ADDRESS LINE	Input
A[4]	PIN_A8	IP BUS ADDRESS LINE	Input
A[5]	PIN_D11	IP BUS ADDRESS LINE	Input
A[6]	PIN_E14	IP BUS ADDRESS LINE	Input
nDMAACK	PIN_W12	IP DMA ACKNOWLEDGE	Input
nDMAEND	PIN_V12	IP DMA END	Input
nDMAACCES	PIN_E2	DMA in process	Input
		Interrupt Space Ipbus chip	Input
n10K_INTSELA	PIN_F4	select	
n10K_IOSELA	PIN_F3	IO Space Ipbus chip select	Input
n10K_MEMSELA	PIN_E4	Memory Space Ipbus chip select	Input

10K_STROBE	PIN_V4	IP bus STROBE	Input or output depending on J3
------------	--------	---------------	---------------------------------

Signal Name	EP2Cxx PIN	Description	Type
BD[0]	PIN_J22	IP data 0	Input/output
BD[1]	PIN_J20		Input/output
BD[2]	PIN_B19		Input/output
BD[3]	PIN_G21		Input/output
BD[4]	PIN_H18		Input/output
BD[5]	PIN_H19		Input/output
BD[6]	PIN_J21		Input/output
BD[7]	PIN_J19		Input/output
BD[8]	PIN_G22		Input/output
BD[9]	PIN_F22		Input/output
BD[10]	PIN_C21		Input/output
BD[11]	PIN_E18		Input/output
BD[12]	PIN_E22		Input/output
BD[13]	PIN_D20		Input/output
BD[14]	PIN_C20		Input/output
BD[15]	PIN_B20	IP data 15	Input/output

4 EP2Cxx I/O ASSIGNMENT

CLOCKS:

Signal Name	EP2Cxx PIN	Description	Type
GCLK1(IPCLK)	M1	IP BUS CLOCK (32 OR 8 MHz)	Input
GCLK2(OSC_IN)	A12	User Clock if Populated on IC9 location	Input
CLKUSER	D4	IP BUS CLOCK (32 OR 8 MHz)	Input

5 CYCLONE I/O LINES

The module IP_CYCLONE 2 HAS 24 20Mbps RS485 transceivers with integrated switch able termination.

Also a set of switches connect the:

- Cyclone output pin driver (input transmitter LTC2861) to the I/O line (positive output transmitter).
- Cyclone input receiver pin (output receiver LTC2861) to the I/O line (negative output transmitter).

With the correct programming of the Cyclone Pin, the user has up to 48 LVTTTL lines configured in groups of eight. Each line within a group can be an Input or an output.

5.1.1 FST3245 SWITCHES Outputs SW[5..0]

These outputs are fed into 6 switches that allow direct connection of the Cyclone II outputs and inputs to the I/O connector. Each switch allows connecting 8 signals. Since this is a switch, it doesn't need to be programmed to set a direction, however it adds neither buffering nor delay or noise. The Cyclone II outputs are LVTTTL.

There is no provision on the board to prevent the Cyclone II to enable the switch at the same time as the RS-485 transceivers. Good judgment needs to be used to prevent a line to be driven both by the cyclone and the transceiver. The switch are enables when SW[5..0] are set to "0"

Signal Name	EP2Cxx PIN	Type	Signals enabled
SW[0]	AB11	Output	DO0 with IO_1 RI0 with IO_2 DO1 with IO_3 RI1 with IO_4 DO2 with IO_5 RI2 with IO_6 DO3 with IO_7 RI3 with IO_8
SW [1]	AB7	Output	DO4 with IO_9 RI4 with IO_10 DO5 with IO_11 RI5 with IO_12 DO6 with IO_13 RI6 with IO_14 DO7 with IO_15 RI7 with IO_16
SW [2]	M19	Output	DO8 with IO_17 RI8 with IO_18 DO9 with IO_19 RI9 with IO_20 DO10 with IO_21 RI10 with IO_22 DO11 with IO_23 RI11 with IO_24
SW [3]	Y9	Output	DO12 with IO_25 RI12 with IO_26 DO13 with IO_27 RI13 with IO_28 DO14 with IO_29 RI14 with IO_30 DO15 with IO_31 RI15 with IO_32
SW [4]	B7	Output	DO16 with IO_33 RI16 with IO_34 DO17 with IO_35 RI17 with IO_36 DO18 with IO_37 RI18 with IO_38 DO19 with IO_39 RI19 with IO_40
SW [5]	U10	Output	DO20 with IO_41 RI20 with IO_42 DO21 with IO_43 RI21 with IO_44 DO22 with IO_45 RI22 with IO_46 DO23 with IO_47 RI23 with IO_48

5.2 Driver

5.2.1 DE[23..0] Outputs

Each DEx line is controlled within the EP2Cxx by the user. An example is to have a:

- WORD DE[15..0]
- BYTE DE[23..16]

The Enable lines control the Output Enable Transmitter of the Differential Driver LTC2861. Each Bit controls one LTC2861 Transmitter.

Writing "0" Disables the Differential Driver and puts it in high impedance.

Writing "1" Enables the Differential Driver.

Pin attribution for these lines is described later.

Upon reset a 4.7 K Ω pull-down resistors keeps the line low. The CYCLONE pins are tri-state upon power-on.

These outputs allow selectively enabling or disabling the corresponding RS-485 output. A "1" enables the transmitter, while a "0" puts its outputs in high impedance.

Signal Name	EP2Cxx PIN	Type
DE[0]	PIN_Y2	Output
DE[1]	PIN_AB19	Output
DE[2]	PIN_Y16	Output
DE[3]	PIN_A9	Output
DE[4]	PIN_F15	Output
DE[5]	PIN_A7	Output
DE[6]	PIN_Y17	Output
DE[7]	PIN_F14	Output
DE[8]	PIN_N22	Output
DE[9]	PIN_F13	Output
DE[10]	PIN_W16	Output
DE[11]	PIN_C14	Output
DE[12]	PIN_B9	Output
DE[13]	PIN_A10	Output
DE[14]	PIN_R20	Output
DE[15]	PIN_A17	Output
DE[16]	PIN_B8	Output
DE[17]	PIN_W14	Output
DE[18]	PIN_Y19	Output
DE[19]	PIN_U15	Output
DE[20]	PIN_AA18	Output
DE[21]	PIN_AB17	Output
DE[22]	PIN_V14	Output
DE[23]	PIN_U21	Output

5.2.2 DO[23..0] Outputs

Cyclone pins connected to the transmitter input pin of the RS485 transceiver.

Signal Name	EP2Cxx PIN	Type
DO[0]	PIN_R5	Output
DO[1]	PIN_V2	Output
DO[2]	PIN_U1	Output
DO[3]	PIN_U2	Output
DO[4]	PIN_V1	Output
DO[5]	PIN_T5	Output
DO[6]	PIN_R6	Output
DO[7]	PIN_M5	Output
DO[8]	PIN_N2	Output
DO[9]	PIN_P1	Output
DO[10]	PIN_J2	Output
DO[11]	PIN_H1	Output
DO[12]	PIN_J1	Output
DO[13]	PIN_R2	Output
DO[14]	PIN_N1	Output
DO[15]	PIN_N4	Output
DO[16]	PIN_T1	Output
DO[17]	PIN_T3	Output
DO[18]	PIN_P2	Output
DO[19]	PIN_P3	Output
DO[20]	PIN_N6	Output
DO[21]	PIN_P5	Output
DO[22]	PIN_T2	Output
DO[23]	PIN_N3	Output

5.2.3 nSLO[23..0] Outputs

These outputs allow one to selectively enable or disable the Driver Slew Rate Control. A low input will force the driver into a reduced slew rate mode.

Signal Name	EP2Cxx PIN	Type
nSLO[0]	PIN_AA6	Output
nSLO[1]	PIN_U8	Output
nSLO[2]	PIN_AB8	Output
nSLO[3]	PIN_AB9	Output
nSLO[4]	PIN_W9	Output
nSLO[5]	PIN_AA7	Output
nSLO[6]	PIN_W15	Output
nSLO[7]	PIN_P17	Output
nSLO[8]	PIN_U22	Output
nSLO[9]	PIN_W8	Output
nSLO[10]	PIN_R1	Output
nSLO[11]	PIN_P6	Output
nSLO[12]	PIN_AA8	Output
nSLO[13]	PIN_V11	Output
nSLO[14]	PIN_AB18	Output
nSLO[15]	PIN_AA9	Output
nSLO[16]	PIN_Y18	Output
nSLO[17]	PIN_AB20	Output
nSLO[18]	PIN_Y21	Output
nSLO[19]	PIN_V9	Output
nSLO[20]	PIN_Y22	Output
nSLO[21]	PIN_V8	Output
nSLO[22]	PIN_AA20	Output
nSLO[23]	PIN_AA19	Output

5.3 RECEIVER

5.3.1 RI[23..0] Inputs

Cyclone pins connected to the receiver output pin of the RS485 transceiver.

Signal Name	EP2Cxx PIN	Type
RI[0]	PIN_W5	Input
RI[1]	PIN_G5	Input
RI[2]	PIN_E1	Input
RI[3]	PIN_W4	Input
RI[4]	PIN_F2	Input
RI[5]	PIN_Y6	Input
RI[6]	PIN_H5	Input
RI[7]	PIN_D1	Input
RI[8]	PIN_F9	Input
RI[9]	PIN_AA5	Input
RI[10]	PIN_D3	Input
RI[11]	PIN_B6	Input
RI[12]	PIN_C2	Input
RI[13]	PIN_Y4	Input
RI[14]	PIN_G3	Input
RI[15]	PIN_D5	Input
RI[16]	PIN_A15	Input
RI[17]	PIN_A16	Input
RI[18]	PIN_A20	Input
RI[19]	PIN_AA12	Input
RI[20]	PIN_J4	Input
RI[21]	PIN_D7	Input
RI[22]	PIN_AA14	Input
RI[23]	PIN_AB14	Input

5.3.2 nRE[23..0] Outputs

These outputs allow selectively enabling or disabling the corresponding RS-485 input. A “0” enables the receiver, while a “1” puts its output in high impedance.

Signal Name	EP2Cxx PIN	Type
nRE[0]	PIN_Y3	Output
nRE[1]	PIN_B4	Output
nRE[2]	PIN_F8	Output
nRE[3]	PIN_AB4	Output
nRE[4]	PIN_E8	Output
nRE[5]	PIN_H4	Output
nRE[6]	PIN_D6	Output
nRE[7]	PIN_AB5	Output
nRE[8]	PIN_H3	Output
nRE[9]	PIN_F1	Output
nRE[10]	PIN_D2	Output
nRE[11]	PIN_A4	Output
nRE[12]	PIN_H6	Output
nRE[13]	PIN_C7	Output
nRE[14]	PIN_B5	Output
nRE[15]	PIN_AA4	Output
nRE[16]	PIN_A5	Output
nRE[17]	PIN_E3	Output
nRE[18]	PIN_W2	Output
nRE[19]	PIN_C1	Output
nRE[20]	PIN_G6	Output
nRE[21]	PIN_Y1	Output
nRE[22]	PIN_A6	Output
nRE[23]	PIN_W1	Output

5.3.3 TE[23..0] Outputs

These outputs allow selectively enabling or disabling Internal Termination Resistance Enable. A high input will connect a termination resistor (120Ω typical) between pins (+) and (-).

Signal Name	EP2Cxx PIN	Type
TE[0]	PIN_V22	Output
TE[1]	PIN_W11	Output
TE[2]	PIN_R19	Output
TE[3]	PIN_A13	Output
TE[4]	PIN_V15	Output
TE[5]	PIN_P18	Output
TE[6]	PIN_F10	Output
TE[7]	PIN_E11	Output
TE[8]	PIN_D9	Output
TE[9]	PIN_AA17	Output
TE[10]	PIN_U14	Output
TE[11]	PIN_Y14	Output
TE[12]	PIN_A14	Output
TE[13]	PIN_B10	Output
TE[14]	PIN_B11	Output
TE[15]	PIN_AA10	Output
TE[16]	PIN_B14	Output
TE[17]	PIN_M18	Output
TE[18]	PIN_C10	Output
TE[19]	PIN_B13	Output
TE[20]	PIN_A11	Output
TE[21]	PIN_M6	Output
TE[22]	PIN_D8	Output
TE[23]	PIN_E9	Output

5.3.4 I/O Pin

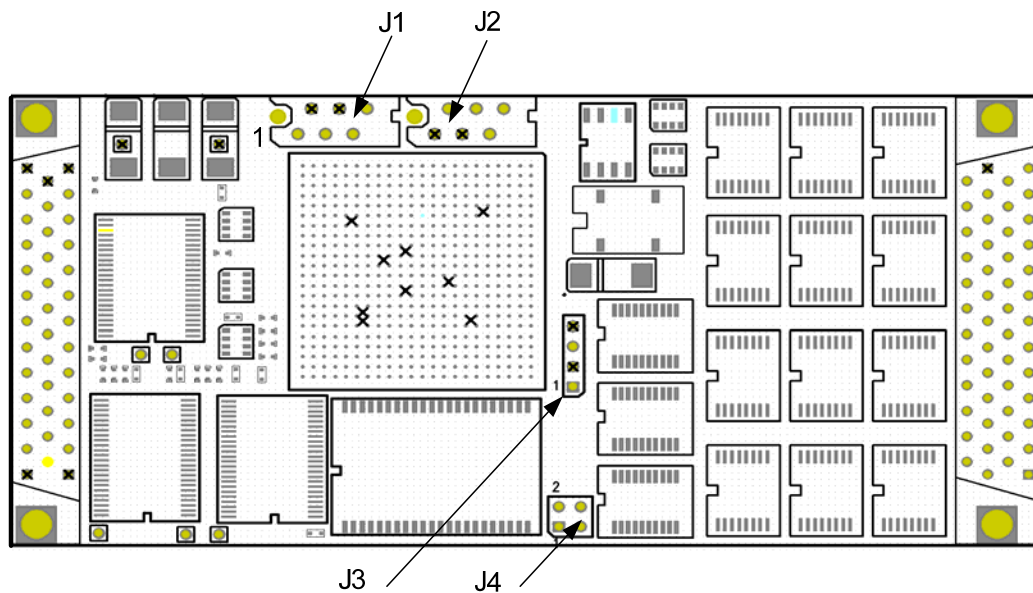
NAME	CYCLONE PIN	I/O name	IP connector
DO[0]	PIN_R5	IO_1	Pin 1
RI[0]	PIN_W5	IO_2	Pin 2
DO[1]	PIN_V2	IO_3	Pin 3
RI[1]	PIN_G5	IO_4	Pin 4
DO[2]	PIN_U1	IO_5	Pin 5
RI[2]	PIN_E1	IO_6	Pin 6
DO[3]	PIN_U2	IO_7	Pin 7
RI[3]	PIN_W4	IO_8	Pin 8
DO[4]	PIN_V1	IO_9	Pin 9
RI[4]	PIN_F2	IO_10	Pin 10
DO[5]	PIN_T5	IO_11	Pin 11
RI[5]	PIN_Y6	IO_12	Pin 12
DO[6]	PIN_R6	IO_13	Pin 13
RI[6]	PIN_H5	IO_14	Pin 14
DO[7]	PIN_M5	IO_15	Pin 15
RI[7]	PIN_D1	IO_16	Pin 16
DO[8]	PIN_N2	IO_17	Pin 17
RI[8]	PIN_F9	IO_18	Pin 18
DO[9]	PIN_P1	IO_19	Pin 19
RI[9]	PIN_AA5	IO_20	Pin 20
DO[10]	PIN_J2	IO_21	Pin 21
RI[10]	PIN_D3	IO_22	Pin 22
DO[11]	PIN_H1	IO_23	Pin 23
RI[11]	PIN_B6	IO_24	Pin 24
DO[12]	PIN_J1	IO_25	Pin 25
RI[12]	PIN_C2	IO_26	Pin 26
DO[13]	PIN_R2	IO_27	Pin 27
RI[13]	PIN_Y4	IO_28	Pin 28
DO[14]	PIN_N1	IO_29	Pin 29
RI[14]	PIN_G3	IO_30	Pin 30
DO[15]	PIN_N4	IO_31	Pin 31
RI[15]	PIN_D5	IO_32	Pin 32
DO[16]	PIN_T1	IO_33	Pin 33
RI[16]	PIN_A15	IO_34	Pin 34
DO[17]	PIN_T3	IO_35	Pin 35
RI[17]	PIN_A16	IO_36	Pin 36
DO[18]	PIN_P2	IO_37	Pin 37
RI[18]	PIN_A20	IO_38	Pin 38
DO[19]	PIN_P3	IO_39	Pin 39
RI[19]	PIN_AA12	IO_40	Pin 40
DO[20]	PIN_N6	IO_41	Pin 41
RI[20]	PIN_J4	IO_42	Pin 42
DO[21]	PIN_P5	IO_43	Pin 43
RI[21]	PIN_D7	IO_44	Pin 44
DO[22]	PIN_T2	IO_45	Pin 45
RI[22]	PIN_AA14	IO_46	Pin 46
DO[23]	PIN_N3	IO_47	Pin 47
RI[23]	PIN_AB14	IO_48	Pin 48

6 EP2Cxx IO Control line ASSIGNMENT

Spare pins Between MAX3128A and EP2Cxx for custom use:

Signal Name	EP2Cxx PIN	Description	Type
INTER_COM[1]	PIN_A3	Possible expansion Image of IPACK provide for test	Bidir
nIOALTERA	D12	Active low when IP_memory access above \$100000 (A20 = "1") or lospace + \$7E-\$7F	Input
n10K_RST	PIN_Y7	Active LOW when IP_Reset is low or when bit #3 (loSPACE + \$04) is set to "1".	Input

7 JUMPERS AND CONNECTORS LOCATION



7.1 Jumper's or connector's description:

Jumpers	Description
J1	MAX3128A PROGRAMMATION connector
J2	EP2Cxx JTAG connector
J3	EP2Cxx Programming Mode selection : MSEL0 and MSEL1
J4	CYCLONE STROBE SELECTION (INPUT/OUTPUT)

Table 7-4 Jumper Description

7.1.1 J3 description

Jumpers	Pin	Description
J3	1-2 ON 3-4 ON	MSEL0 & MSEL1 are Pulled Low (Grounded) EP2Cxx Configuration Data source is JTAG
J3	1-2 OFF 3-4 ON	MSEL1 (GND) MSEL0 (VCC) EP2Cxx Configuration Data source IP Bus for PASSIVE SERIAL MODE configuration
J3	1-2 ON 3-4 ON	MSEL1 (GND) MSEL0 (GND) EP2Cxx Configuration IS ACTIVE SERIAL MODE (SOURCE IS EPCS4)

Table 7-5 J3 Description

7.1.2 J4 IP strobe generation

Jumpers	Pin	Description
J4	1-3 ON 2-4 ON	Cyclone generates IPSTROBE
J4	1-2 ON 3-4 ON	Cyclone receives IPSTROBE (default)

Table 7-7 J3 Description

8 Connectors Description:

CONNECTOR	DESCRIPTION
P1	IP Bus
P2	I/O Bus

Table 8-6 Connectors Description

8.1 IP bus interface P1

		P1		
Pin 1	GND		Pin 26	GND
Pin 2	+5V		Pin 27	+5V
Pin 3	IPRESET*		Pin 28	IPRW*
Pin 4	XLD00		Pin 29	IDSEL0*
Pin 5	XLD01		Pin 30	DMAREQ0*
Pin 6	XLD02		Pin 31	MEMSEL0*
Pin 7	XLD03		Pin 32	DMAREQ1*
Pin 8	XLD04		Pin 33	INTESEL0*
Pin 9	XLD05		Pin 34	DMACK*
Pin 10	XLD06		Pin 35	IOSEL0*
Pin 11	XLD07		Pin 36	
Pin 12	XLD08		Pin 37	XLA1
Pin 13	XLD09		Pin 38	DMAEND*
Pin 14	XLD10		Pin 39	XLA02
Pin 15	XLD11		Pin 40	ERROR*
Pin 16	XLD12		Pin 41	XLA03
Pin 17	XLD13		Pin 42	INTREQ0*
Pin 18	XLD14		Pin 43	XLA04
Pin 19	XLD15		Pin 44	INTREQ1*
Pin 20	IPBS0*		Pin 45	XLA05
Pin 21	IPBS1*		Pin 46	Strobe*
Pin 22			Pin 47	XLA06
Pin 23			Pin 48	IPACK*
Pin 24	+5V		Pin 49	+5V
Pin 25	GND		Pin 50	GND

Table 8-1 P1 IP BUS connector

8.2 I/O Port P2

PIN	Name	PIN	Name
Pin 1	IO_1	Pin 26	IO_26
Pin 2	IO_2	Pin 27	IO_27
Pin 3	IO_3	Pin 28	IO_28
Pin 4	IO_4	Pin 29	IO_29
Pin 5	IO_5	Pin 30	IO_30
Pin 6	IO_6	Pin 31	IO_31
Pin 7	IO_7	Pin 32	IO_32
Pin 8	IO_8	Pin 33	IO_33
Pin 9	IO_9	Pin 34	IO_34
Pin 10	IO_10	Pin 35	IO_35
Pin 11	IO_11	Pin 36	IO_36
Pin 12	IO_12	Pin 37	IO_37
Pin 13	IO_13	Pin 38	IO_38
Pin 14	IO_14	Pin 39	IO_39
Pin 15	IO_15	Pin 40	IO_40
Pin 16	IO_16	Pin 41	IO_41
Pin 17	IO_17	Pin 42	IO_42
Pin 18	IO_18	Pin 43	IO_43
Pin 19	IO_19	Pin 44	IO_44
Pin 20	IO_20	Pin 45	IO_45
Pin 21	IO_21	Pin 46	IO_46
Pin 22	IO_22	Pin 47	IO_47
Pin 23	IO_23	Pin 48	IO_48
Pin 24	IO_24	Pin 49	
Pin 25	IO_25	Pin 50	GND

Table 8-1 P2 I/O connector

9 CONFIGURATION DEVICES: SERIAL EEPROM

The EP2Cxx can be programmed using Configuration device serial EEPROM: EPCS4.

10 PSA Passive serial asynchronous programming

Download sequence

1. First set (J3) jumpers to have MSEL0 (VCC) and MSEL1 (GND).
2. Start with either IP bus reset or power-up reset.
3. The module should be under EP control.
 - Verify that EP_CONTROL line is "1" by reading Iospace +\$2 location. Bit #7 should be set to "1" (EP_CONTROL = "1"). Other bits are used for revision control of the Max3128A.
 - With actual revision you should read \$80.
 - Else Write \$0 at I/O base address + \$00 and verify that the module switches to EP control allowing access to the programming registers.
4. The ICR initialization register located at I/O base address + \$4 should read "00".
The ISR initialization register located at I/O base address + \$6 should read "73".
5. Enable the tri-state buffer used to control the nCONFIG line(pull externally by a 10kΩ).
 - Write \$01 to the ICR register (I/O + \$4).
 - Read \$70 from the ISR register (I/O + \$6).
 - This line is normally LOW because it is controlled by the bit #2 of the ICR register. Setting the tri buffer enable pull down the nCONFIG line to low. Should be low for 8 uS min.
6. Now set the nCONFIG line to a "1" to start the programming process by writing \$3 at the ICR address (I/O + \$4).
7. nSTATUS line should come to a "1" after around 30 μS.
8. Read the ISR (I/O + \$6) should read "73", Initialization status register , wait until nSTATUS is high and verify that:
 - bit #0 = 1 nCONFIG set to "1"
 - bit #1 = 1 nSTATUS set to "1"
9. Write the first byte to the Initialization Register located at base I/O address + \$8.
The byte is then serialized and sent to the cyclone using the IPCLK (GCLK1) .
It takes less than 1 μS at 32 MHZ clock and ~ 4 μS at 8 MHZ clock. The lower bit is sent first.
During the transfer the bit #3 of ISR is high, you must wait for it to go low before you continue to write

During the Serialization transfer the bit #3 of the ISR is High. Wait until it goes back Low to continue.
10. Repeat step 9 until all bytes have been downloaded.
11. Read the ISR and verify that CONF_DONE (bit #2) is high
12. Release the nCONFIG. Line by first disable the tri-state buffer than set bit #1 to "0".
13. Write a \$1 at the I/O base address + \$02 to switch the EP2Cxx under user control.
14. The EPLD should have release the control of the CYCLONE, you should be able now to read the ID and revision registers of the EP2Cxx and also accessed all the other registers.

11 EXTERNAL CLOCK FOR EP2Cxx

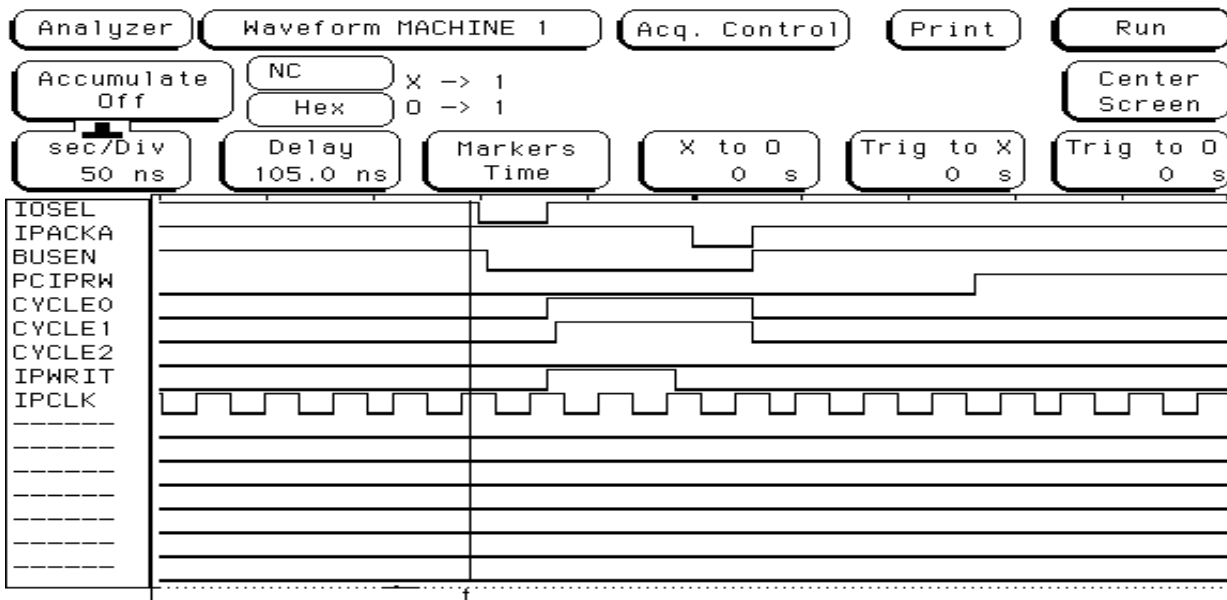
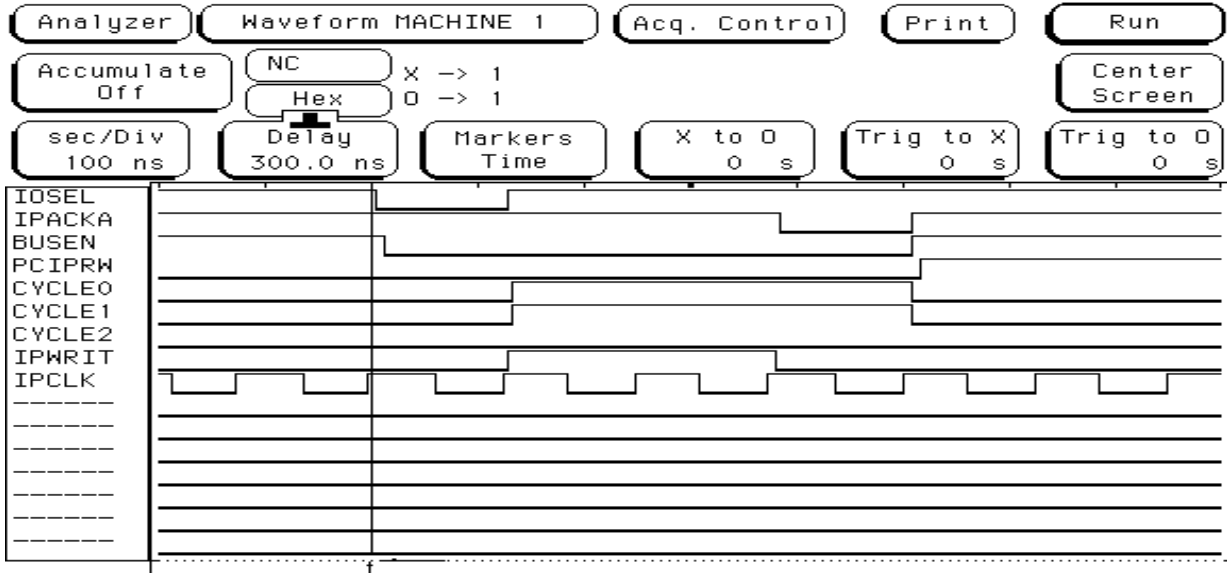
1. External oscillator.
 - Must assign pin A12 to GCLK2.
2. Internal IP-CLK.
 - Must assign pin M1 to GCLK1.

12 Timing between MAX3128A AND EP2Cxx

When the IP BUS accesses the I/O space register inside the CYCLONE, the MAX3128A provide the following signals.

- CYCLE [2...0] Define the type of cycle
- IP_Write Set to "1" during a write access. This signal is disabled when the MAX3128A sends the IPACK signal to the IP bus. The two following timing tables shows the signals at 8MHZ and 32 MHz.

Note: Older versions were keeping the signal active until the end of the cycle. When used as a Clock Enable it was latching data at every clock. PowerPC host pipelines address as soon they receive a DTACK on the VME bus. A write to the I/O has the effect to write at two consecutive locations.



13 Additional timing

Some additional arbitration timing between IP BUS and EP2Cxx

EXAMPLE # 1:

Arbitration between an IP bus memory access (read) and an EP2Cxx memory access (read).

On the diagram, MEMRQ is 10KMEMRQ; MEMGNT is 10KMEMGNT; MEMACK is 10KMEMACK.

The IP bus has the control first of the DPR then the EP2Cxx.

10KMEMRQ must stay low until the end of the cycle.

10KMEMGNT is removed two clocks later due to the 10KMEMRQ from the EP2Cxx which gets resynchronized with the IPCLK (8 MHz)

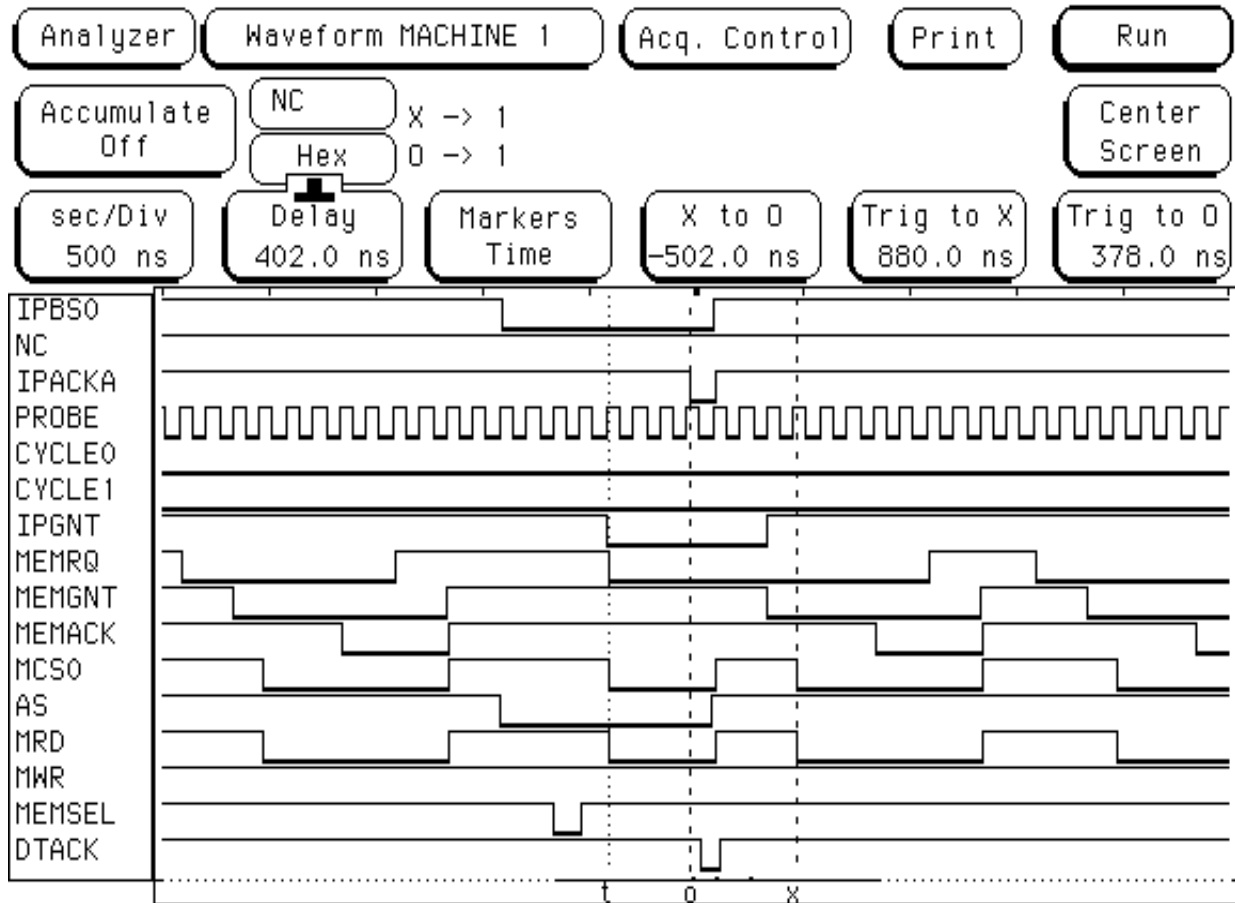
Notes:

10KMEMACK is generated but not used into arbitration.

The MAX3128A control the buffer that enables the memory control lines.

All these tests use a state machine implemented as an example into the EP2Cxx with a 10KMEMRQ generated by the GCLK1 clock divided by 8.

The timing access of the DPR matches its requirement for a 70 ns memory access. Using GCLK1 at 8 MHz should be no problem



EXAMPLE # 2:

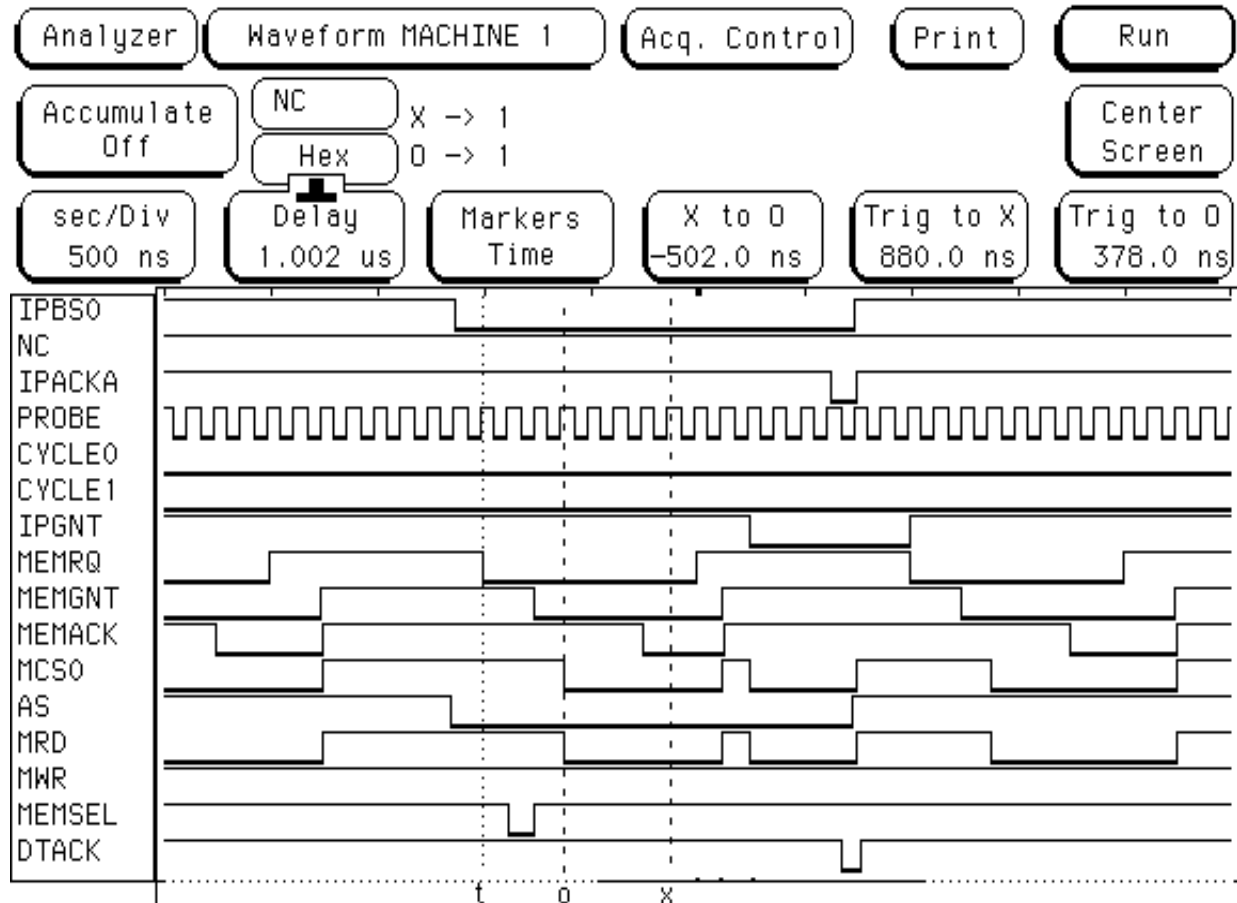
Arbitration between an IP bus memory access (read) and an EP2Cxx memory access (read).

MEMRQ = 10KMEMRQ

MEMGNT= 10KMEMGNT

MEMACK =10KMEMACK.

Same as above except that the EP2Cxx gets the bus before the IP bus.



EXAMPLE # 3:

Arbitration between an IP bus memory access (read) and an EP2Cxx memory access (read).

MEMRQ = 10KMEMRQ

MEMGNT= 10KMEMGNT

MEMACK =10KMEMACK.

This example is the same as example # 1 but the IP bus gets first the control of the DPR, and then comes the EP2Cxx. A write is performed by the IP bus.

