Fairisle Port Controller
Design and Ideas

Mark Hayter and Richard Black

10th March 1994

The Fairisle Port Controller is designed to be a flexible card providing the input buffering for the Fairisle switch and to allow for investigations of queueing and network management. There are three versions of the hardware. The original version, FPC1, was used for initial testing of the hardware and software for the project, only five were built and they gained a number of patches including connector changes. The FPC2 forms the basis of the network built for the Fairisle project itself, and only requires patches if it has its processor upgraded or if it runs the Xi6 design (see later). The FPC3 is built as part of the BT supported Fairisle follow-on project and contains the updated hardware to aid experimental work.

1 The Port Controller

The Fairisle Switch is based around a simple switch fabric with all the processing being done in the port controllers. The port controller consists of two parts. The first is a processing unit based around an ARM RISC processor. The second section consists of buffer memory and a DMA engine. Figure 1 shows an overview of the port controller, the lower half forming the processor section, and the upper the network section. The transceiver section, containing the encoders and drivers for the line, is a plug in card on the first two versions of the port controller, and is included on the main board in version three.

The processing section of the port controller has an ARM processor and runs the Wanda microkernel. The current boards can have either 1 or 4 Mbytes of dynamic memory and 128, 256 or 512k bytes of ROM. The card includes an Acorn podule bus to allow the addition of other network interfaces. In addition to the standard ARM version of the Wanda kernel there is a hand crafted interrupt routine to deal with the queueing of cells. Cells are only buffered on the input to the fabric - the processor has no control of the fabric output. The port controller includes an identity PROM, which is used to give each port controller a unique id.

The network section of the card contains 128k bytes of static memory, buffer fifos and the xilinx control chip. The static memory is used for cell buffering, and resides in the memory mapped I/O space of the processor. The xilinx arbitrates access to this memory between the processor, transmission to the fabric and reception from the network. The fifos are between the port controller and the transmission system to decouple the different transfer speeds.

The port controller is a double hight extended depth eurocard which plugs into the Fairisle Backplane[1]. The backplane links port controllers with the fabric card and provides power.

The main control hardware is contained on a xilinx FPGA. This is programmed by the processor at kernel initialisation time. Over the period of the project many versions of the xilinx hardware have been produced and the hardware/software balance has been investigated. There are three

[1] There are clock problems with version 1 backplanes which requires a patch to the card or reduced operation speed.

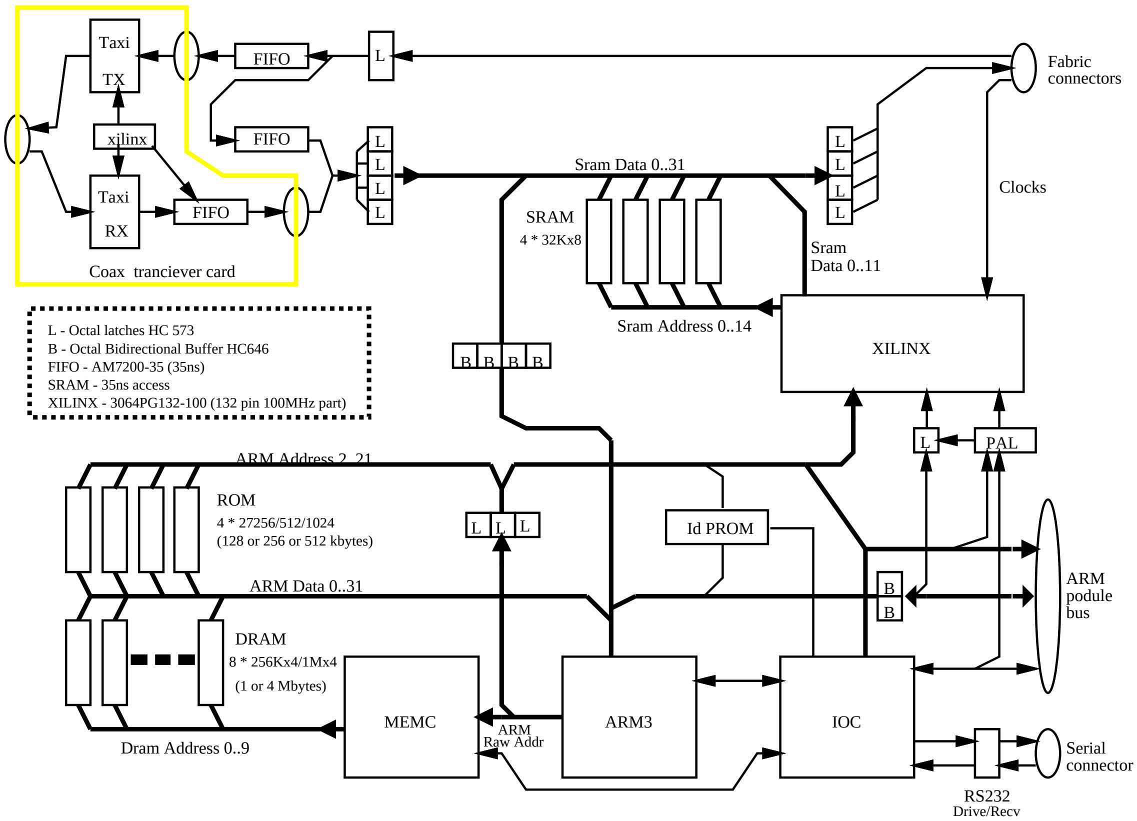

Figure 1: Port controller layout

main versions of the xilinx design in general use. For the FPC2 the two designs are known as Xi3 and Xi6. For the FPC3 the design is Xi5. These three are described in separate documents.

2 The cell buffer memory

Cells on the port controller consist of a 4 byte header and a 48 byte payload. In addition a two byte routeing tag must be prepended to cells injected into the switch fabric. The input buffer on the port controller consists of 128k of fast SRAM, organised as 32k 32 bit words. This is logically divided into 2048 cell buffers, each of 16 words. The cell data occupies words 2 to 13 (inclusive) of a cell buffer, use of the other words depends on the version of the control xilinx used. Internally when data is moved as 32 bit words a little-endian format is used, ie low numbered bytes appear in the low bits.

The switch routeing information consists of two bytes. In all cases the least significant bit is the “active” bit and is used to indicate the start of a valid cell. The first byte is used to select the switch output to be used, and is described in the Fabric design document. The second byte, known as the “portc route byte” is used by the device on the output of the switch to perform internal routeing. The format used by the port controller is shown in table 1, the documentation for the DAN ATM devices and CPU node should be consulted for details for other devices. Note that since the routeing bytes are discarded on the way through the fabric and port controller the spare bits cannot be used to send information.

Bit	Name	Use
0	Active	Always set for a cell
1	Loop	set for loop-back, clear to transmit to net
2	SMcast	(Xi5 only) set to enable serial multicast if bit 1 is clear
3-7	Spare	Unused

Table 1: FPC Port Controller Route byte

3 The xilinx DMA controller

The xilinx chip, which controls the network part of the card, consists of five main sections: the processor interface, the SRAM interface, reception from the network, injection into the fabric and transmission to the network. The general design of each of these sections is outlined below, more specific information appears in the individual design documents.

3.1 SRAM interface

The xilinx is the only device which drives the SRAM, arbitrating between its DMA use and the processor. The arbiter controls a multiplexor which drives the SRAM address bus and enables of SRAM data bus drivers. The address used in DMA operations is generated from registers internal to the xilinx chip.

Part of the SRAM data bus is used directly by the xilinx, to load its internal registers and output status information. The width of data bus to the xilinx depends on the part used, for a 3042 data lines 0..11 and 30..31 are available, a 3064 also has lines 12..15 connected, the 3090 has all the data lines connected. When a cell header is read the VCI will be on the bottom 16 bits. If the SAR1A or FAS protocols are used then the start and end bits will be on bits 30 and 31.

3.2 Injection to the fabric

The xilinx will start sending cells into the fabric if there are buffers in the transmit queue and the processor has issued a RamRdGo command since reset or a fabric NACK. Transmission continues until the queue is empty, or a cell is NACKed by the fabric or destination port controller. If transmission stops because the queue becomes empty then it will restart as soon as a new item is put in the queue – strobing of the RamRdGo line is only required after NACK or xilinx reset.

3.3 Fabric output to network

Data from the fabric is moved under control of the xilinx "fab2fifo" section, into one (or both) of the two fifos. The cell arriving from the fabric has had the fabric routeing information removed but still contains the port controller route byte. The route byte is shown in table refpcrou. The cell Active bit is always a 1, and is used to indicate the start of the full cell in the same way as in the fabric. The loopback bit indicates whether the cell is destined for transmission or for looping back into the port controller. Using the Xi5 design insertion is possible into both fifos by setting the SMcast bit in the route byte.

Bytes arriving from the fabric are latched externally to provide a cycle delay for decoding the routeing information. On recognition of a cell the loopback bit is used to indicate which fifo should be used for the duration of this cell. Bytes are then strobed into the fifo for the number of bytes in a cell. The fifo contains 256[2] nine bit words. The ninth bit is used as a start of cell marker and is

[2] 512 on FPC3 due to the use of synchronous fifos.

set as the portc route byte is strobed into the fifo. This byte will be removed by the transmission card and replaced by the appropriate start of cell marker for the transmission medium. The data associated with this byte is ignored by the transmission system.

The Half Full signal from the selected fifo is used to generate the Ack signal to the fabric. Thus as soon as the fifo becomes half full back pressure is applied. Since the transmitting port controller only senses the Ack signal for a single byte time it is possible for almost an entire cell to be put in the fifo past its half full point.

3.4 Reception from network or loopback fifo

The reception system expects to read from a fifo, a reception card must therefore either use a fifo or have the same interface as a fifo. The start of cell control sequence will have been replaced by a byte with the SOC bit set. The data in this byte is currently not defined and the byte is discarded. Data transmitted through the loopback fifo will already be in this format.

If either fifo is seen to contain data then it is read from, if both contain data then they are read alternately. When a SOC bit is seen from one of the fifos it is selected. Bytes are read from the selected fifo and assembled into words in the latches. When a word has been assembled it is written to the SRAM. If a new SOC is seen part way through reading a cell then the first cell is discarded and the new one read into the same cell buffer, a status bit is set to indicate this happening.

If there are no buffers available for reception then only the receive fifo is read, and any cells found will be read out of the fifo and discarded. This ensures that the fifo does not fill, so cells are dropped safely as they emerge from the fifo rather than in a uncontrolled way on the way into the fifo. The loopback fifo will apply back pressure across the fabric when it fills so it is safe not to discard cells from it. When cells are discarded a status bit is set.

This section of the chip caused many of the problems found in the implementation. It has been reimplemented many times, to improve the operation speed of the xilinx. The problems are caused by the asynchronous nature of the problem – the line speed is slower than the port controller clock speed, so care has to be taken with the fifo going empty. In both FPC1 and FPC2 the fifos used required pulses with a 25:75 duty cycle and generated the empty signal very late in the read. To avoid some of these problems the FPC3 uses clocked fifos, which take a free running clock and an enable signal. Despite these reimplementations the overall design of this section did not change between versions.

3.5 The processor interface

The processor control signals are decoded and I/O accesses to the xilinx space (0x3180000 - 0x31FFFFF) are detected. The use of I/O space causes the access cycle to be asynchronous, allowing the xilinx to arbitrate the memory access. The xilinx decodes addresses in this range to allow processor access to internal registers and the buffer SRAM Addresses 0x3180000 and 0x31C0000 are ARM immediate constants and so it is advantageous to place things at these addresses. The results of performing a byte write are undefined, but it usually results in all 4 bytes in the addressed word being set to the value given.

Since an asynchronous memory access is used it will take at least 2 cycles of the ARM REF8M 8MHz clock to perform the access. There are two designs of the ARM interface section "newarm.hdl" and "slowarm.hdl". The slow version goes to extreme lengths to ensure the synchronisation between the ARM REF8M clock and the fabric clock, the faster version makes the assumption that $REF8M < \frac{1}{2}fabClk$ and therefore needs less synchronisation. The slow interface takes a minimum of three REF8M cycles, and the faster a minimum of two.

4 Status and Control

The port controller interrupt and control lines for all devices are managed by the I/O controller (IOC). This provides two eight bit wide conventional interrupt (IRQ) mask registers, an eight bit wide fast interrupt (FIQ) mask register and a six bit wide control register.

Bit	Name	use
0-1	I2C[0-1]	I^2C bus to podule connector
2	RamRdGobar	Strobe low to start xilinx sending cells (Xi2 bits)
3	FifoReset TransReset	High to reset the Fifos and transmission subsystem low to enable (FPC3 reset transmission subsystem only)
4	Led1	Low to turn on red CPU controlled LED
5	Led2	Low to turn on yellow[3] CPU controlled LED

Table 2: IOC Control lines

The IOC control lines are used to control aspects of the xilinx chip, as detailed in table 2. The control register lines have pull up resistors on them so on reset, when they are undriven, they will go high. Thus a card reset will cause the fifos to be reset, and the strobe line to be deasserted. The top two bits of the IOC control register must always be set when the register is written.

The FIQ interrupt register which is described in table 3 contains three bits which are used by the Xilinx design and other general signals.

bit	Name	Use
0	FH[0]	Xilinx signal Xifiq[0]
1	FH[1]	Xilinx signal Xifiq[1]
2	FLbar	Xilinx signal Xifiq[2]
3	C[3]	State of fifo reset signal
4	C[4]	State of LED1 (red)
5	C[5]	State of LED2 (yellow)
6	IL0	Podule FIQ line
7	-	Always set

Table 3: IOC FIQ Status register

The IOC general purpose interrupt registers contain a mixture of external signals of which only some are meaningful on the port controller and some internal signals relating to the timer and console function also provided by the IOC. Interrupt status register A is described in table 4 and register B in table 5. The FIQ uses the "Always set" IRQ bit to post interrupts to the Wanda kernel. Note that the podule FIQ line also appears in the IRQ register B should it be necessary to service a podule device which generates FIQ interrupts without complicating the xilinx FIQ software.

5 Programming the xilinx chip on the port controller

The xilinx gate array on the port controller is configured in peripheral mode (M0=+5V M1=gnd M2=+5V) for easy programming under software control. There are three status lines which come from the xilinx chip to the IOC. These are INIT, HDC and RDY/BUSYbar which appear in IOC

Bit	name	use
0	IL6bar	Connected to a link. See section 10.
1	IL7bar	Connected to a link. See section 10.
2	IFbar	FPC1,2: Connected to a link. See section 10. FPC3: Xilinx IRQ line. See Xi5 documentation.
3	IR	FPC1,2: Connected to a link. See section 10. FPC3: Xilinx IRQ line. See Xi5 documentation.
4	POR	Set at power on, may be explicitly cleared
5	TM0	Set on timer 0 reload
6	TM1	Set on timer 1 reload
7	-	Always set

Table 4: IOC Interrupt status register A

Bit	name	use
0	PFIQbar	Podule FIQ
1	XiInitbar	Xilinx Init line
2	XiHdc	Xilinx high during configuration
3	XiRdy	Xilinx ready
4	LIRQ IL4 IL4	FPC1: Transputer link IRQ) FPC2: IL4bar pin hard wired high FPC3: Xilinx IRQ line. See Xi5 documentation.
5	PIRQ	Podule bus IRQ
6	STx	KART Tx Data empty
7	SRx	KART Rx Data full

Table 5: IOC Interrupt status register B

status register B[4] as noted in table 5.

In addition to the data there are three control lines which affect the xilinx chip. These control lines are manipulated by the processor through a simple PAL. These lines are the WriteStobe (WRTbar) active low line, a low to high transition on which causes the current value on the data lines to be read in by the chip. The other lines are the reprogram line which can be used to cause an already programmed chip to be wiped, and the reset line which is connected to global reset on the chip. In addition the PAL controls the enable line of a buffer between the IOC data bus lines and the xilinx data lines.

On power on reset or board reset the PAL asserts and holds the reset signal and deasserts all other signals. This ensures there are no board conflicts and prevents the xilinx chip from operating until the software chooses to bring it out of the reset state.

The PAL operations are controlled by the address bus only, appearing in the IOC space at device 1, see table 6, and are activated by a write to the given address. For FPC1 the pal equations may be found in */usr/groups/fairisle/designs/portc/revisions/palrev3.{pld, map}*. For FPC2 the pal equations may be found in directory */homes/mdh/portc/fpc2pal*.

The first thing that the software does is to being the chip out of reset and wait for the INIT signal to go inactive. This indicates that the xilinx chip has completed its internal clearing operation. (If the chip has currently got a configuration in it then this signal will not have gone active and the chip will be running with the old configuration).

[4]These interrupts are not enabled so do not appear in the request register

Name	Assert	Deassert
WriteData	0x3210000	0x3210004
DataEnable	0x3210008	0x321000C
Reprogram	0x3210010	0x3210014
Reset	0x3210018	0x321001C

Table 6: PAL access addresses

Then the HDC (High During Configuration) signal is tested. This signal is not valid during reset or initialisation phases but otherwise indicates if the chip is ready to receive a configuration. This signal will be high at this point iff there is no configuration in the chip. If this signal is low then the chip must have its current data removed, this is performed by asserting the reprogram line for a short period and waiting for the INIT line to go inactive. The xilinx allows for the HDC pin to become a user I/O pin after configuration but this is not permitted on the port controller. This line has a pull down resistor so can be left unused in the xilinx design.

In either case the chip is now ready to be programmed. This is performed by loading the configuration data bytewise, with the BUSY line being checked to ensure handshaking. Bytes are written using four operations. This is not quite right but it goes something like this:

1. The Data connect line is asserted. This links the xilinx data lines with the IOC data lines. The IOC data lines are isolated from the processor data lines except when the processor issues an address in the I/O space. Thus opcode fetches do not reach the xilinx data pins.
2. The Write line is asserted; the value written being the data byte required to be sent to the chip. At the end of this cycle the data is floating on the linked I/O - xilinx bus.
3. The write line is deasserted causing the xilinx to latch the data. Again the byte written to the address is the data byte for the chip. This ensures that the IOC buffers enforce rather than destroy the required data floating on the capacitance of the I/O bus from the previous cycle.
4. The data connect line is deasserted, decoupling the busses and permitting the processor to read the IOC register and wait for the handshaking signal.

It should be noted that these operations are extremely sensitive not only to delay but also to potential accesses to the I/O space. For this reason these operations should be performed with interrupts disabled. Once all the data has been written the HDC line should be checked for inactivity to ensure that the chip has accepted the data and assumed the configuration. After configuration strobing the reset line will reset the xilinx device, clearing all the internal flipflops. If this is done the fifos should also be reset using the IOC control line (see table 2).

Once the xilinx has been programmed the WriteData line may be used by the processor to control the xilinx, and the XiRdy line may be used by the xilinx to generate a processor IRQ interrupt. Currently neither of these are used.

6 The network interface

The FPC1 and FPC2 have a parallel network interface. This is normally used to connect a daughter board on FPC2 to interface to the transmission medium, but a Universal Fairisle Connector may also be used. The interface is byte wide, and is asynchronous. The connection has two parts, one for transmission and one for reception, as detailed in table 7. A separate connection to the daughter board joins to two 50R coax connectors which are mounted on the main PCB for stability.

Signal Name	Transmit Direction	Receive Direction	Use
Data[0-7]	Out	In	Bytewide data lines
SOC	Out	In	Set for start of cell byte
Readbar	In	Out	Strobe low to read next byte
Emptybar	Out	In	Clear if there is no data
FifoReset	N/C	Out	Reset the fifos

Table 7: Port Controller network interface

Data is extracted a byte at a time on the transmit side by strobing the Readbar line low. The start of a cell is marked by having the SOC bit set, the data which accompanies this is undefined. Following the SOC are the 52 bytes of the cell. If there is no data available then the Emptybar line will go low, and reads are undefined. Emptybar goes low as the last valid byte is being read. The transmission system should read this stream of bytes and convert them into the form required by the link, to maintain compatibility with the emerging link standard it may inset the HEC byte after the first four bytes have been sent[5]. The reception system should convert the form on the link back into the SOC and bytes form, which it should supply bytewise when the read request is strobed on the receive side. The port controller expects to receive in the same form as it sent, so the simplest link may be provided by cross connecting the transmit and receive connections of two port controllers using ribbon cable; experiments suggest that the ribbon cable should be less than 40cm. Alternatively a serial link can be used by plugging in a TAXI transmission card which provides a link based on twin coax that can run a few hundred Metres. Another possibility is for a parallel connection to be made to the interface card of a host machine. The timing details for the reads and empty signal are as for a Am7200-35 CMOS fifo, with a 45ns read cycle, 35ns read access and empty going low 30ns after the read cycle of the last byte is initiated. This may be modified to the faster -25 fifo.

The FPC3 has the TAXI transmission system on the main PCB. No other transmission interface can be used. There are problems with some of the batches of synchronous fifos used on the FPC3 transmission system. Fifos with date stamp of 9230 are not affected. Fifos with a date stamp of 9318, 9338 or 9343 require 22pF capacitors on the clock lines of the side of the fifo which is driven by a TAXI chip.

7 The Podule Bus

The podule connector that is on the port controller conforms to the standard podule specification "Acorn A Series Podules" with two exceptions to standard use. In Acorn machines writes are done to the high 16 bits and reads from the low 16 bits. On the port controller both reads and writes are done using the low 16 bits. The second change is in the select line that is provided to the podule backplane. Podule backplanes normally use A21 to enable the selection of the Memc Podule Strobe signals; the Simple Podule select enable being generated by IOC. On any machine with an IO controller other than IOC on the motherboard (ie. the xilinx chip on the port controller), the select signal supplied to the backplane in place of A21 must exclude the range of IO addresses used by the motherboard. This is performed in the port controller PAL.

[5]see the "Fairisle/Yes/Maybe protocol" description for more details.

8 The Identity PROM

The port controllers (excepting FPC1) have a 32 byte PROM. This is used to give each port controller a unique Id. The PROM is connected on IO select line 5, and provides data on the top 8 bits of the data bus. The format of the data is given in table 8.

byte	Name	Use
0	Revision	PCB revision number
1-2	Card	Card serial number (1-LSB, 2-MSB)
3	Flags	PROM flag bits (see table 9)
4-9	Uid	48 bit Uid for card
10-23	-	Reserved, set to 0
24-31	Secret	Security tag, or 0

Table 8: Id PROM data format

The PROM used has unprogrammed value of zeros, so reserved and unused fields should be set to zero. The flag bits indicate what is in this PROM, as shown in table 9.

bit	Name	Use
0	Uid	Set if PROM contains valid UID field
1	Secret	Set if PROM contains valid secret field
2-7	Reserved	Reserved, set to zero

Table 9: Id flag byte data format

9 The interrupt handler

The interrupt handler, or FIQ, is a hand crafted assembler routine which is responsible for forwarding the cells through the port controller. This is a low latency interrupt, which runs asynchronously to the Wanda networking management code - that is it is not disabled even during a Wanda critical section. In order to further improve the latency of FIQ interrupt response the ARM provides six banked registers which are only accessible in FIQ mode and which may be used both to preserve state from one FIQ to the next and to avoid having to save registers before use.

The IOC provides the interrupt request and masking facility for the FIQ. The eight possible interrupt sources are detailed in table 3. Normally only the three interrupts generated by the xilinx chip are used. All interrupts may be individually masked.

The meanings of the interrupts and the actions of the interrupt software are highly dependent on the xilinx hardware in use, and are described in the appropriate document.

9.1 Potential loading

The port controller's operating speeds are 100Mbits/sec on the network interface and 20MBytes/sec[6] on the fabric interface. There is one frame pulse every 64 clock periods. A cell could arrive on the transmission interface every 4.24us. A frame pulse occurs every 3.2us[7]. Additionally NACKs could occur on 60% of cells at saturation.

[6] Using the original backplane 18MBytes/sec is the maximum reliable fabric rate

[7] 3.55us at 18MHz

9.2 Arm CPU Timings

The ARM3 operates on a 25MHz[8] FCLK and a MCLK configurable as in table 11. The ARM3 contains a 4K cache and when executing or fetching data from the cache is able to complete such a cycle in one FCLK. [9] The ARM cache is write through and so on a write to memory or an access to an I/O device an external cycle is required. The ARM synchronises to MCLK performs the cycle and resynchronises to FCLK. The synchronising takes between 0 and 1 FCLK and between 1 and 2 MCLKs. The resynchronising takes between 0 and 1 FCLK. All external cycles require one MCLK "latent" thus an external DRAM write cycle takes 2 MCLKs, an access to the IOC 3 MCLKs and an access to the Xilinx chip with the "slowarm" interface (currently used for all designs) takes 4 MCLKs. Based on an 8.33MHz MCLK on average a DRAM write cycle, an IOC access and a Xilinx access take 460ns, 580ns and 700ns respectively. These are a long time compared with an FCLK period of 40ns.

The ARM600 which is used on some FPC2 cards and the ARM610 used on the FPC3 runs at 25 MHz FCLK[10], but (due to a bug on the chip) requires a special relationship between MCLK and FCLK, requiring use of 8.33MHz MCLK. These newer processors are faster at synchronising between the clocks and contain a write buffer which enables the CPU to continue to run from its internal clock while writing to external devices.

10 Links on FPC2

The port controller revision 2 PCB has a number of configuration links. Table 10 lists them, and the usual settings. Many of the links cause spare signals to be pulled up or down, these may be useful for experiments or modifications. The Xilinx link enables the pin out of a 3064 to differ from the 3042 in the position of the RxFifoRbar signal, which requires a fast path output. 3042 designs may only use the B11 (south) setting of this link, 3064s may use either. The Osc Join link enables use of a single oscillator module for the CPU and memory clocks.

link	Usual State	Use
1	Right	ROM size. Right for 27512/271024, left for 27256
2	On	IL6bar. Readable through IOC, IL6bar is low if link on.
3	On	IL7bar. Readable through IOC, IL7bar is low if link on.
4	North	Xilinx link. Selects RxFifoRbar on A11 (North) or B11 (south).
5	On	Osc join. If on joins OSC48 and OSC60 output.
6	On	Clock Ph1 source. On to enable backplane clock.
7	On	Clock Ph2 source. On to enable backplane clock.
8	On	IFbar. Spare FIQ interrupt request, on to disable (==high)
9	On	IR. Spare IRQ interrupt request, on to disable (==low)
10	On	Spare inverter. MBUF pin 8 (out on 12). On to pull high.
11	On	Spare Ph1 latch. TXLATCH pin 10 (out on 15). On to pull low

Table 10: Links on FPC2

Links 2 and 3 are provided for the Port Controller to discover the speed of its 8 MHz reference clock. They provide a 2 bit number from which the mapping is described in table 11.

[8] There is one 30MHz and four 28MHz parts.

[9] A noteworthy exception is the swap operation which always reads from the memory even if present in the cache

[10] This may be reduced

Link3	Link2	Speed of REF8M the 8 MHz reference clock.
Off	Off	reserved.
Off	On	12MHz
On	Off	10MHz
On	On	8.333 MHz

Table 11: Configuring REF8M

11 Links on the FPC3

The links on the FPC3 are numbered to be consistent with those on the FPC2, though there are fewer links. The links are shown in table 12. Links 2 and 3 provide the REF8M mapping as described above.

link	Usual State	Use
1	27512	ROM size. 27512/271024, or 27256
2	On	IL6bar. Readable through IOC, IL6bar is low if link on.
3	On	IL7bar. Readable through IOC, IL7bar is low if link on.
10	On	Spare inverter. MBUF pin 8 (out on 12). On to pull high.
11	On	Spare Ph1 latch. TXLATCH pin 10 (out on 15). On to pull low
12	On	Link TAXI and CPU clocks to same OSC module. On to link

Table 12: Links on FPC3

12 Patches for FPC2 and Xi6

To run the Xi6 design on an FPC2 the xilinx chip needs to be connected to two additional data bits. This is a compatible upgrade which uses two previously unconnected pins on the xilinx chip, and so it is still possible to use the older Xi3 design. Pin 2 on the xilinx (also known as A2) should be connected to data bit 26. This is best achieved by soldering to the via on the back of the board near where that signal attaches to SRAM[3] pin 13. Pin 3 on the xilinx (also known as A3) should be connected to data bit 27. This is best achieved by soldering to the via on the back of the board near where that signal attaches to WR[3] pin 16.