XC6200 Field Programmable Gate Arrays

Table Of Contents

XC6200
Field Programmable Gate Arrays

April 24, 1997 (Version 1.10)

Product Description

Features

- High-Performance Sea-Of-Gates FPGA
  - Thousands of configurable cells
  - Fine-grain architecture, abundant registers, gates and routing resources
  - Extremely high gate count for structured logic or datapath designs
  - High-speed SRAM control store
  - 220 MHz flip-flop toggle rates
- Extremely Flexible Cell Architecture
  - Over 50 distinct logic functions per cell
  - One register and gate/multiplexer possible for *every* cell
- Advanced Processor Compatible Architecture
  - Xilinx FastMAP™ processor interface
  - Direct processor read/write access to *all* internal registers in user design with no logic overhead
  - All user registers and SRAM control store memory mapped onto processor address space
  - Programmable data bus width (8, 16 or 32-bits)
  - Easily interfaced to most microcontrollers and microprocessors
- Advanced Dynamic Reconfiguration Capability
  - High-speed reconfiguration via parallel CPU interface
  - Full or partial reconfiguration/context switching possible
  - Unlimited reprogrammability
  - Ideal for custom computing applications
- Flexible Pin Configuration
  - All User I/Os programmable as in, out, bidirect, three-state or open drain.
  - Configurable pull-up/down resistors
  - CMOS or TTL logic levels
- Flexible Interconnect Architecture
  - Low-delay FastLANE™ hierarchical routing scheme gives large number of fast ‘Longlines’
  - Any cell can be connected to any other
  - Suited to both structured synchronous data path type designs or irregular random logic
  - Completely flexible clocks and asynchronous clears for registers
  - 4 Global low-skew signals
- Testability
  - Pre-tested, high-volume, standard part
  - JTAG capability with library macrocells
- Sophisticated CAD Tools
  - Implement designs using familiar tools like Viewlogic and Synopsys
  - Dedicated XACT*step* Series 6000 back-end tools
  - Use PC or Unix workstation platforms
  - Fully automatic mapping, placement and routing
  - Interactive Physical Editor for design optimization
  - Large Xilinx parts library for schematic capture
  - VHDL synthesis

Table 1: The XC6200 Family of Field Programmable Gate Arrays

Device	XC6209†	XC6216	XC6236†	XC6264
Typical Gate Count Range	9000-13000	16000-24000	36000-55000	64000-100000
Number of Cells	2304	4096	9216	16384
Number of Registers	2304	4096	9216	16384
Number of IOBs	192	256	384	512
Cell Rows x Columns	48x48	64x64	96x96	128x128

† = Planned Product

Description

The XC6200 family is a new type of high performance FPGA from Xilinx.

XC6200 is a family of fine-grain, sea-of-gates FPGAs. These devices are designed to operate in close co-operation with a microprocessor or microcontroller to provide an implementation of functions normally placed on an ASIC. These include interfaces to external hardware and peripherals, glue logic and custom coprocessors, including bit-level and systolic operations unsuited to standard processors.

The XC6200 can provide extremely high gate counts for data path or regular array type designs. In these cases the actual gate count may turn out to be a factor of two or more greater than those given in Table 1.

An XC6200 part is composed of a large array of simple, configurable cells. Each basic cell contains a computation unit capable of simultaneously implementing one of a set of logic level functions *and* a routing area through which inter-cell communication can take place. The structure is simple, symmetrical, hierarchical and regular, allowing novice users to quickly make efficient use of the resources available.

The nearest-neighbor interconnect of the underlying cells is supplemented with wires of length 4 cells, 16 cells and Chip-Length, which provide low delay paths for longer connections. In addition there are four global input signals which provide a low skew distribution path for critical high fan-out nets such as clocks and initialization signals.

XC6200 parts are configured by an integral, highly stable six-transistor SRAM control store. This allows XC6200 parts to be quickly reconfigured an unlimited number of times. The SRAM control store can be mapped into the address space of a host processor and additional support logic is provided to allow rapid reconfiguration of all or part of the device. In addition, the outputs of function units within the device can be read by a processor through the FastMAP™. Processors can read or write registers within logic implemented on the device. Data transfers can be 8, 16 or 32 bits wide, even when register bits are distributed over a column of cells. These capabilities allow XC6200 FPGAs to support virtual hardware in which circuits running on the FPGA can be saved (`swapped out') to allow the FPGA resources to be assigned to a different task, then restored (`swapped in') at a later time with the same internal state in their registers. Sections of the device can be reconfigured without disturbing circuits running in other sections. Thus an XC6200 in a coprocessor application can be shared by several processes running on the host computer.

Design entry and proving may be carried out with Xilinx software products using industry standard schematic capture, synthesis and simulation packages such as Viewlogic, Mentor Graphics and Synopsys. A comprehensive library of parts, ranging from simple gate primitives to complex macro-functions, exists to make this an easy task.

Below the top level design tools, the XC6200 product family is supported by XACT*step* Series 6000. This contains tools ranging from simple symbolic editors for high-efficiency user designs to sophisticated cell-compilation tools. These tools help to ensure the design captured is laid out efficiently with no user intervention. Node delays can then be back-annotated to the front-end logic simulator for design proving. The tools allow for manual intervention in the layout process if desired. Incremental design is also supported: if a design is laid out and subsequently changed, only the modified block has to be re-laid out.

The functions available within each cell provide a good target for logic synthesis programs. The simple cell architecture allows arbitrary user logic designs to be mapped onto a number of cells, rather than having to split the design up into medium-complexity mini-functions for mapping to a larger configurable logic block. Because each cell can be configured as a register, designs containing far more registers than would be possible with a larger configurable block are achievable.

Architecture

Logical and Physical Organization

The XC6200 architecture may be viewed as a hierarchy. At the lowest level of the hierarchy lies a large array of simple cells (Figure 1). This is the 'sea of gates'. Each cell is individually programmable to implement a D-type register and a logic function such as a multiplexer or gate. Any cell may also be configured to implement a purely combinatorial function, with no register. This is illustrated in Figure 7.

First generation fine-grain architectures implemented only nearest-neighbor interconnection and had no hierarchical routing (Figure 1). XC6200 is a second generation fine-grain architecture, employing a hierarchical cellular array structure. Neighbor connected cells are grouped into blocks of 4x4 cells (Figure 2) which themselves form a cellular array, communicating with neighboring 4x4 cell blocks. A 4x4 array of these 4x4 blocks forms a 16x16 block (Figure 3). In the XC6216 part, a 4x4 array of these 16x16 blocks forms the central 64x64 cell array which is then surrounded by I/O pads (Figure 4).

Each level of hierarchy (unit cells, 4x4 cell blocks, 16x16 cell blocks, 64x64, etc.) has its own associated routing resources. Basic cells can route across themselves to connect to their nearest neighbors and thus provide wires

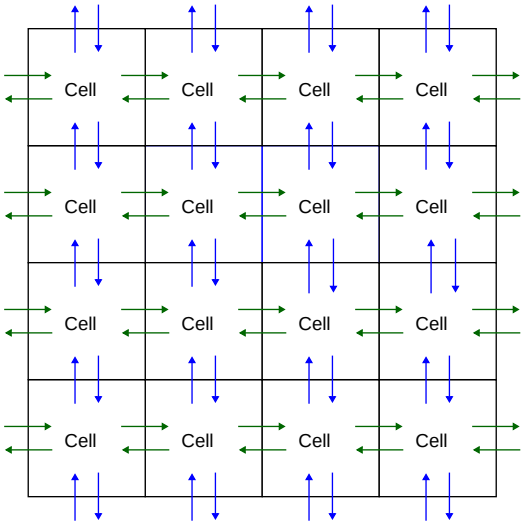


Figure 1. Nearest-Neighbor Interconnect Array Structure

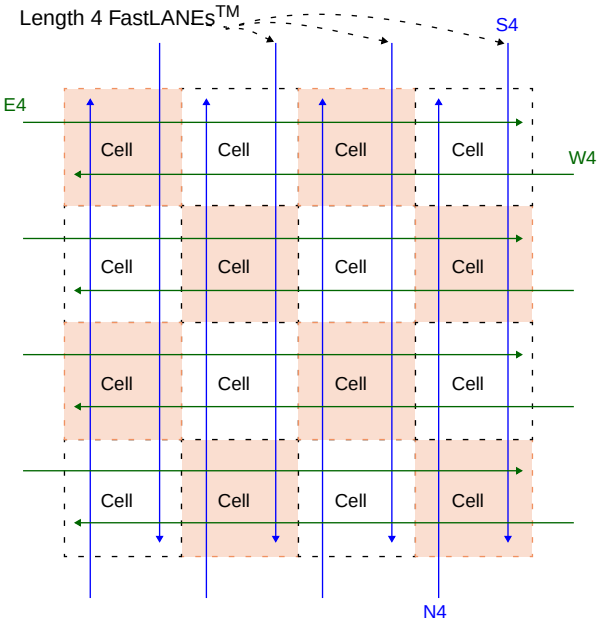


Figure 2. XC6200 4x4 Cell Block

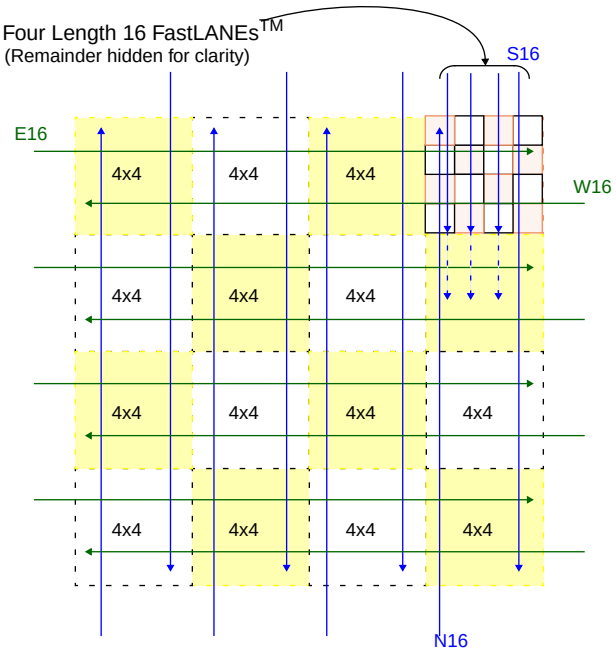


Figure 3. XC6200 16x16 Cell Block

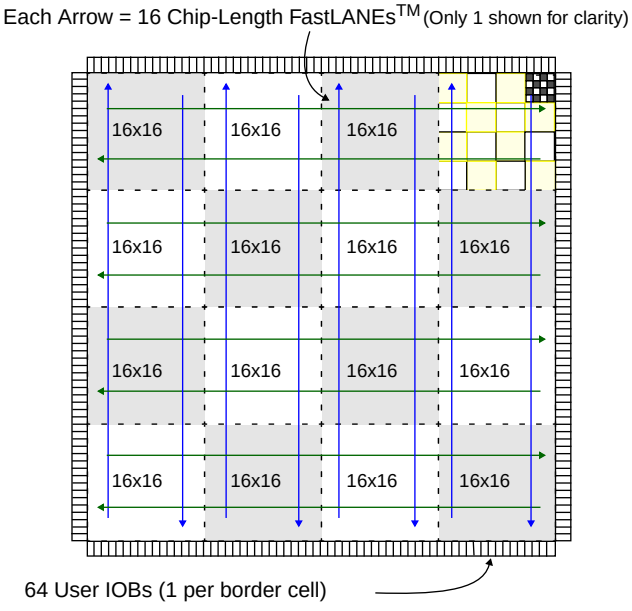


Figure 4. XC6216 Device

of length 1 cell. Note that cells used for interconnect in this manner can still be used to provide a logic function. Wires of length four cells are provided to allow 4x4 cell blocks to route across themselves without using unit cell resources. Similarly 16x16 cell blocks provide additional wires of length 16 cells and the 64x64 array provides Chip-Length wires. Larger XC6200 products extend this process to 256x256 cell blocks and so on, scaling by a factor of 4 at each hierarchical level as required. Intermediate array sizes (e.g. 96x96) are created by adding more 16x16 blocks. Switches at the edge of the blocks provide for connections between the various levels of interconnect at the same position in the array (e.g. connecting length 4 wires to neighbor wires).

The longer wires provided at each hierarchical level are termed 'FastLANEs^{TM}'. it is convenient to visualize the structure in three dimensions with routing at each hierarchical level being conceptually above that in lower hierarchical levels, with the cellular array as the base layer. The length-4 FastLANEs^{TM} are driven by special routing multiplexers within the cells at 4x4 block boundaries. All routing wires are directional. They are always labeled according to the signal travel direction. For example, S4 is a length-4 FastLANE^{TM} heading from North to South. In Figures 2, 3 and 4 each individual cell has a length 4, 16 and Chip-Length FastLANE^{TM} above it. However only a small number are shown for clarity.

The benefit of the additional wiring resources provided at each level of the hierarchy is that wiring delays in the XC6200 architecture scale logarithmically with distance in cell units rather than linearly as is the case with the first generation neighbor interconnect architectures. Since 4x4 cell block boundaries lie on unit cell boundaries, the switching function provided at 4x4 cell boundaries is a superset of that provided at unit cell boundaries; i.e. it provides for neighbor interconnect between the adjacent cells as well as additional switching options using the length 4 wires. Similarly, the switching unit on 16x16 cell block boundaries provides a superset of the permutations available from that on the 4x4 cell block boundaries. Further switching units are also provided on the 64x64 cell boundaries to provide the Chip-Length FastLANEs^{TM}.

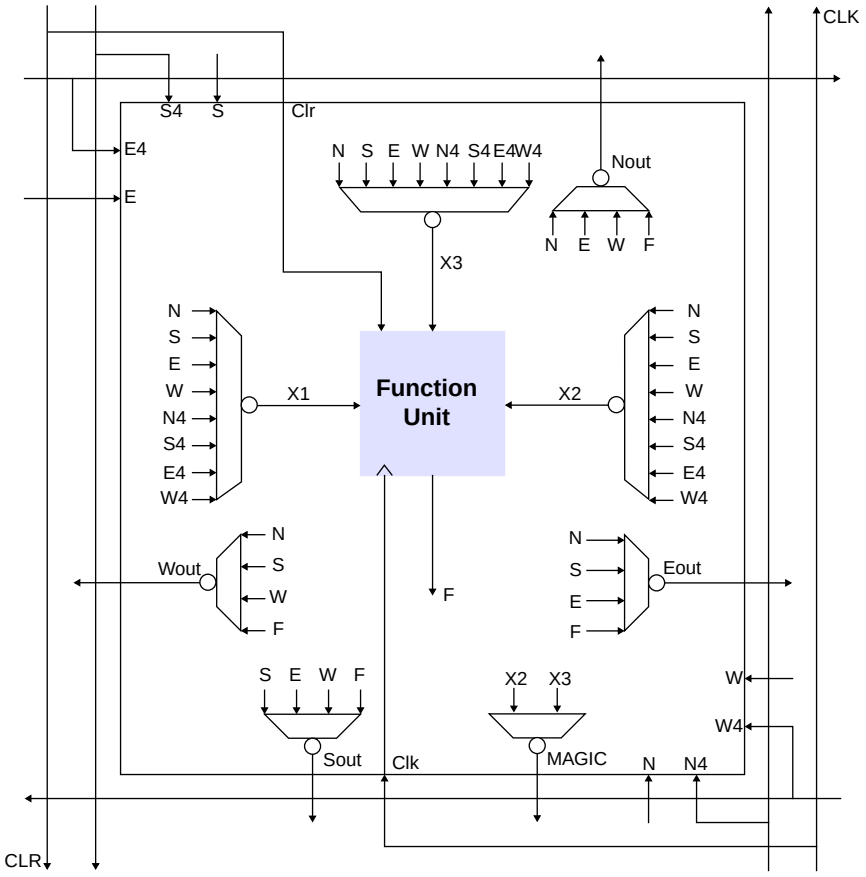


Figure 5. XC6200 Basic Cell

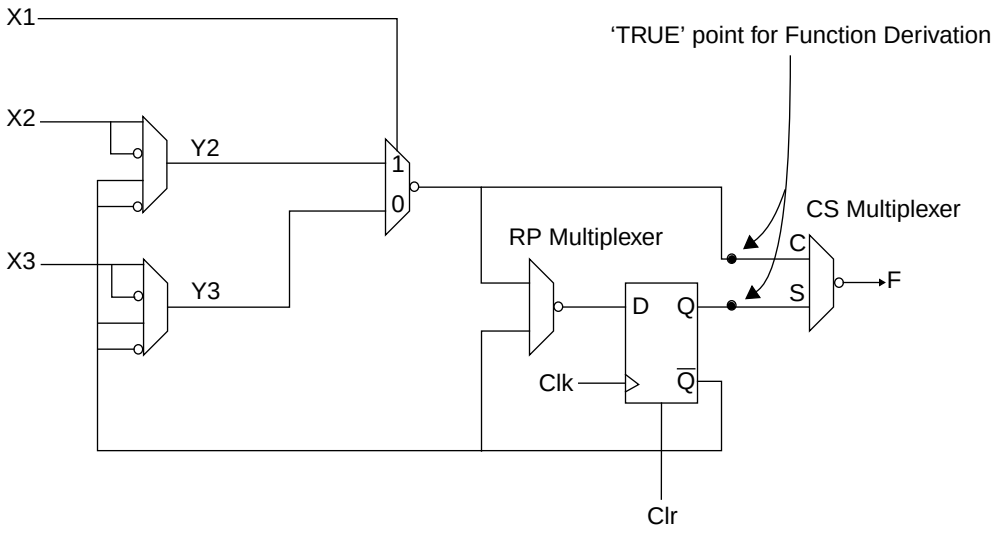


Figure 6. XC6200 Function Unit

Additional Routing Resources

Magic Wires

The majority of interconnections are routed using the nearest-neighbor and FastLANEs™ described above. Each cell has a further output (labeled 'Magic') which provides an additional routing resource. A cell's Magic output is not always available for routing. Its availability is dependent on the logic function implemented inside the cell. More information on the physical nature of the Magic wires is given in the section "Function Unit" on page 6.

Each cell's Magic output is routed to two distinct 4x4 block boundary switches. The Magic wire can be driven by N, S, E or W from adjacent cells or from the N4, S4, E4 or W4 FastLANEs™ passing over the cell. This makes it particularly useful for corner-turning (all other routing resources are straight).

The Magic wires are illustrated in Figure 8.

Global Wires

The XC6200 architecture permits registers within a user design to be clocked by different clocks and cleared by different asynchronous clears. Clocks and Clears may be provided by any user I/O pin or generated from user logic internally. In line with good synchronous digital design practices, it is recommended that a single global Clock and Clear are used. This minimizes the likelihood of timing problems and gives more reliable simulations.

Four Global wires (G1, G2, GClk and GClr) are provided for low skew, low delay signals. These wires are intended for global Clock and Clear or other high fan-out signals and are distributed throughout the array in a low skew pattern. A global signal can reach the clock and clear inputs of any cell on the array passing through very few routing switches. The four Globals are very similar. It would be possible to use GClk as a global Clear signal, however for minimum delay, it is recommended that GClk be used for global clocks and GClr for global clears. GClk and GClr can reach the inputs of any register in the array, passing through only a single routing switch. G1 and G2 may be used for secondary global clocks or clears. G1 and G2 have a slightly larger delay than GClk and GClr.

Function Unit

Figure 5 shows the basic XC6200 cell in detail. The inputs from neighboring cells are labeled N, S, E, W and those from length 4 wires N4, S4, E4, W4 according to their signal direction. Additional inputs include Clock and Asynchronous Clear for the Function Unit D-type register. The output from the cell function unit, which implements the gates and registers required by the user's design, is labeled F. The Magic output is used for routing as described earlier. The multiplexers within the cell are controlled by bits within the configuration memory. As can be seen from Figure 5, the basic cells in the array have inputs from the length 4 wires associated with 4x4 cell blocks as well as their nearest neighbor cells. The function unit design allows the cells to efficiently support D-type registers with Asynchronous Clear and 2:1 multiplexers, as well as all Boolean functions of two variables (*A* and *B*) chosen from the inputs to the cell (N,S,E,W,N4,S4,E4,W4) (Table 2). Figure 7 shows the schematic representations of the basic cell functions pos-

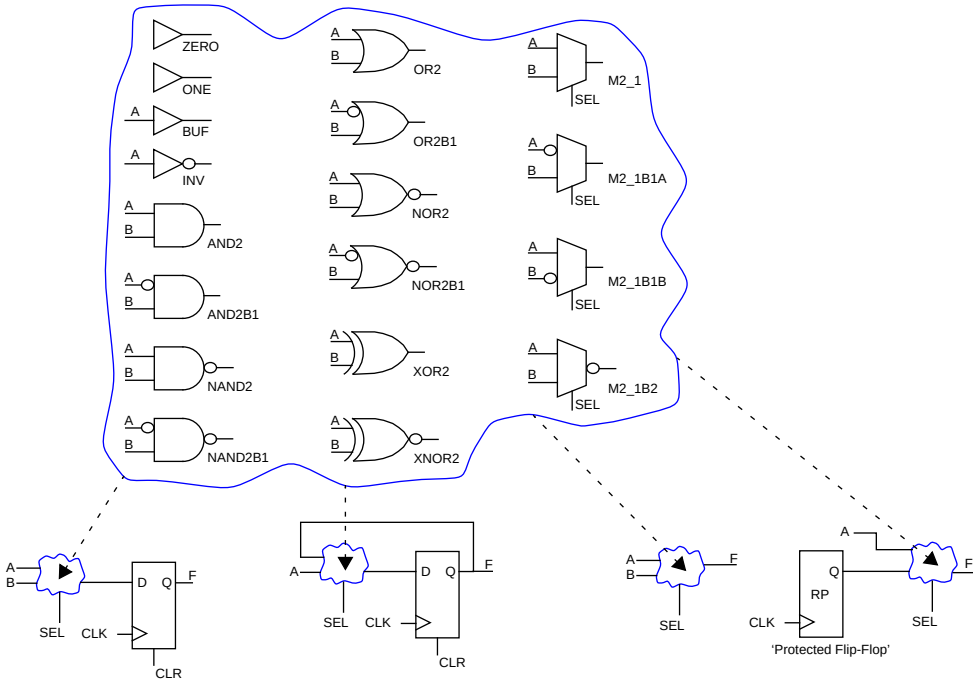


Figure 7. Cell Logic Functions

sible. The Magic routing output can only be used if the signal to be routed can be placed on X2 or X3.

Figure 6 shows the implementation of the XC6200 function unit. The design uses the fact that any function of two Boolean variables can be computed by a 2:1 multiplexer if suitable values chosen from the input variables and their complements are placed on its inputs. The Y2 and Y3 multiplexers provide for this conditional inversion of the inputs. The CS multiplexer selects a combinatorial or sequential output. The RP multiplexer allows the contents of the register to be 'protected'. If register protection is enabled then only the programming interface can write to the register. It does not change when the X inputs to the function unit change, **even if it is clocked or cleared**. This feature is useful in designs containing control registers which are only to be written by an external microprocessor. The control inputs of all the multiplexers, except the one switched by X1, come from configuration memory bits.

Cell Logic Functions

Each cell can be configured as any two-input gate function, any flavour of 2:1 multiplexer, constant 0 or 1, single input functions (buffer or inverter) or any of these in addition to a D-type register. This is illustrated in Figure 7. The gate names given correspond to standard Xilinx library part names for these primitives. Although three inputs are shown entering the combinatorial 'cloud', dual and single input functions are also possible. e.g. inverter + register or register alone. The buffer symbol is available in the CAD libraries. There is no requirement for the designer to buffer signals with this architecture. This is because signals are regularly buffered by routing multiplexers. Symmetrical functions are also possible but not shown in Figure 7. e.g. $\overline{A}.B$ (AND2B1) is shown but $A.\overline{B}$ (AND2B2) is not. This is because *A* and *B* are assigned to user signals by the logic mapping software to provide the required function. Thus a multiplexer with inversion on the SEL input is unnecessary because the mapping software can simply swap the signal assignments for *A* and *B*.

The sources of the X1, X2 and X3 input multiplexers are set automatically by CAD software during the logic mapping phase. Table 2 shows the assignments for all the cell multiplexers to compute the various logic gate functions. A NAND2B1 is equivalent to an OR2B1 with the inputs swapped and a NOR2B1 is equivalent to an AND2B1 with the inputs swapped therefore these gates are not listed in Table 2. The C and S signals are taken as the 'true points' for the gate mappings in this table.

If the register within a cell is not used in the design then a special 'fast' version of most gates can be configured, using the register to provide a constant 1 or 0. For example a fast AND gate (*A.B*) can be configured by setting the register to 0 during configuration and assigning $\overline{Q}$ to Y3. *A* is routed to X1 and *B* to X2. $\overline{X2}$ is assigned to Y2. When *A* changes to 0, Y3 is selected and F is forced Low as soon as the X1-controlled multiplexer

switches. In the normal AND gate, there would be an additional delay as *A* propagated through the Y3 multiplexer. Fast or normal gates may be specified by the designer but for optimal layout density this is best left to the logic mapping software.

The multiplexer functions have a straightforward mapping with fixed assignments to X1,X2 and X3, with Y2 and Y3 providing input inversions as required.

Routing Switches

As described earlier, each cell within a 4x4 block is able to drive its output to its nearest neighbors to the N,S,E and W. In addition to this, cells at 4x4 block boundaries are also able to drive their outputs onto length-4 FastLANEs™. Special switch units are provided around each 4x4 block boundary to facilitate these connections. This is illustrated in Figure 8. These switches also allow higher levels of hierarchical routing (e.g. length-16 and Chip-Length FastLANEs™) to be connected to length-4 FastLANEs™.

Figure 8 also shows the connections for each cell's Magic output. Each Magic output is routed to two destinations for increased routing flexibility. The two connections are labeled M and MA. The Magic wires allow cell outputs to jump to the edge of the 4x4 block and hence onto FastLANEs™ or into the next 4x4 block. They are also a particularly efficient way of making large busses turn corners.

N,S,E and W switches are similar, however the N switches contain additional multiplexers to drive the register Clock lines. The contents of the boundary switches are shown in Figures 9 to 12. These multiplexers

Table 2: Function Derivation

Function	X1	X2	X3	Y2	Y3	RP	CS	Q
0	A	A	A	X2	$\overline{X3}$	X	C	X
1	A	A	A	$\overline{X2}$	X3	X	C	X
BUF (Fast)	A	X	X	Q	$\overline{Q}$	Q	C	0
BUF	X	A	A	$\overline{X2}$	$\overline{X3}$	X	C	X
INV (Fast)	A	X	X	$\overline{Q}$	Q	Q	C	0
INV	X	A	A	X2	X3	X	C	X
A.B (Fast)	A	B	X	$\overline{X2}$	$\overline{Q}$	Q	C	0
A.B	A	B	A	$\overline{X2}$	$\overline{X3}$	X	C	X
$\overline{A}$.B (Fast)	A	X	B	$\overline{Q}$	$\overline{X3}$	Q	C	0
$\overline{A}$.B	A	A	B	X2	$\overline{X3}$	X	C	X
$\overline{A.B}$ (Fast)	A	B	X	X2	Q	Q	C	0
$\overline{A.B}$	A	B	A	X2	X3	X	C	X
A+B (Fast)	A	X	B	Q	$\overline{X3}$	Q	C	0
A+B	A	A	B	$\overline{X2}$	$\overline{X3}$	X	C	X
$\overline{A}$+B (Fast)	A	B	X	$\overline{X2}$	Q	Q	C	0
$\overline{A}$+B	A	B	A	$\overline{X2}$	X3	X	C	X
$\overline{A+B}$ (Fast)	A	X	B	$\overline{Q}$	X3	Q	C	0
$\overline{A+B}$	A	A	B	X2	X3	X	C	X
A⊕B	A	B	B	X2	$\overline{X3}$	X	C	X
$\overline{A⊕B}$	A	B	B	$\overline{X2}$	X3	X	C	X
M2_1	SEL	A	B	$\overline{X2}$	$\overline{X3}$	X	C	X
M2_1B1A	SEL	A	B	X2	$\overline{X3}$	X	C	X
M2_1B1B	SEL	A	B	$\overline{X2}$	X3	X	C	X
M2_1B2	SEL	A	B	X2	X3	X	C	X

invert some inputs. This is not shown in the figures. See the "Programming" section for details.

The multiplexers driving the NOut, SOut, EOut and WOut lines are actually implemented within the cell adjacent to the switch. These multiplexers take the place of the neighbor multiplexers found in the basic cell (see Figure 5). Boundary cells contain additional RAM bits to control the larger multiplexers. An additional output is available from these multiplexers. This output reflects the output that would have come from the cell's neighbor multiplexer had it been a basic non-boundary cell. To distinguish this from the output of the boundary switch (NOut, SOut, EOut or WOut), it is suffixed with a 'C' (Cell). e.g. NC for an Nswitch. NC is one of NIn, E, W, or F depending on the least-significant two bits of the NOut multiplexer select lines. Hence NC is identical to NOut if NOut is one of F, NIn, E or W. If NOut is one of N4In, N16, PS4 or MN then NC is one of F, NIn, E or W depending on which signal is routed to NOut. The 'C' signal is one of the upper four inputs to the 8:1 multiplexers shown in Figures 9 to 12, the actual value being selected by the two least-significant multiplexer select lines. Similar 'C' signals are generated in the Sswitch, Eswitch, and WSwitch.

The S4 input to the NOut multiplexer in the Nswitch is actually the S4 *input* to the adjacent Sswitch in the 4x4 block immediately to the North of this block. This should not be confused with the S4Out signal from that block's Sswitch. This is also true of some of the other inputs to the multiplexers in other boundary switches. To avoid confusion, these inputs are prefixed with the letter 'P' (for

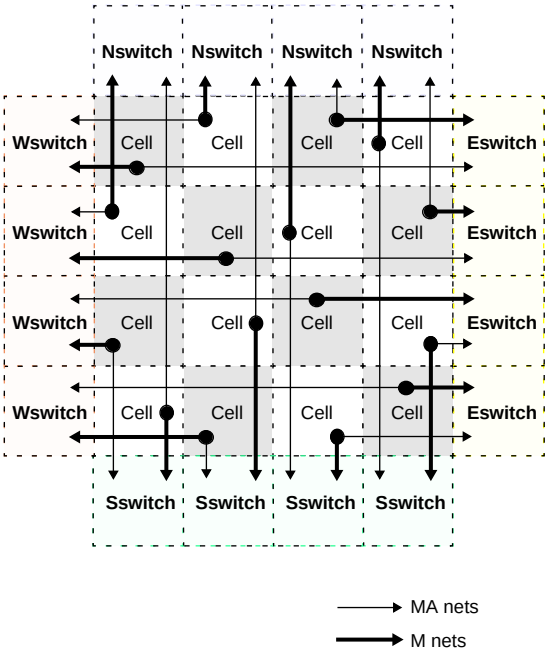


Figure 8. Routing Switches at 4x4 Block Boundary

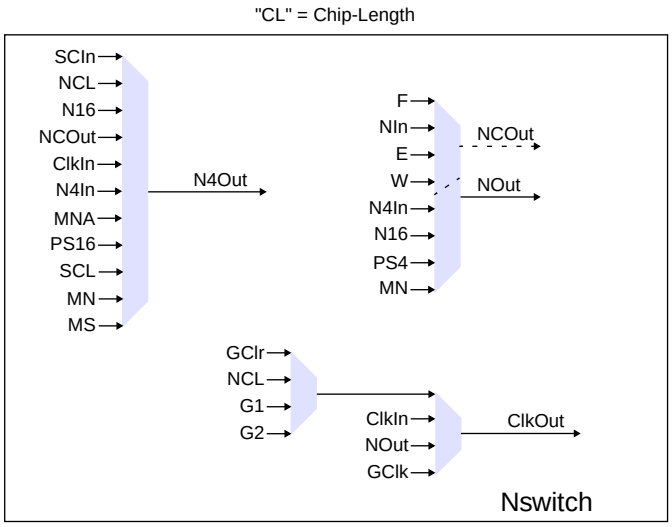


Figure 9. Contents of Nswitch

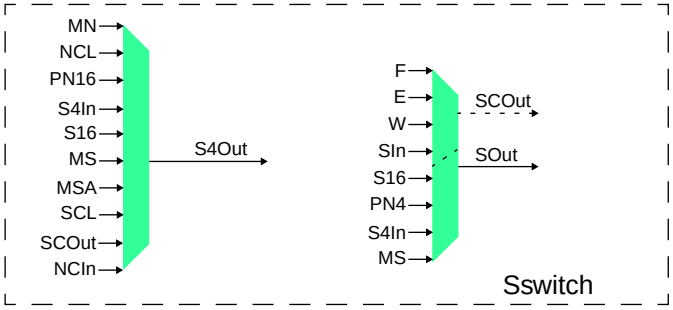


Figure 10. Contents of Sswitch

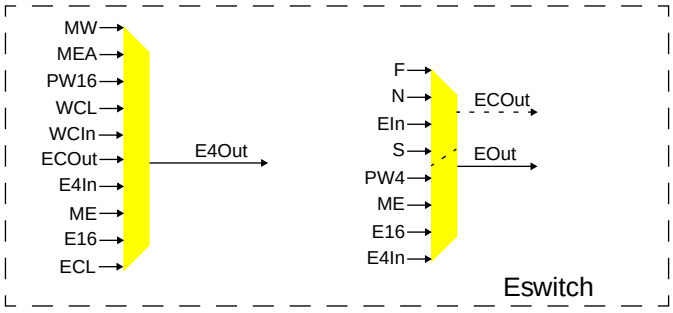


Figure 11. Contents of Eswitch

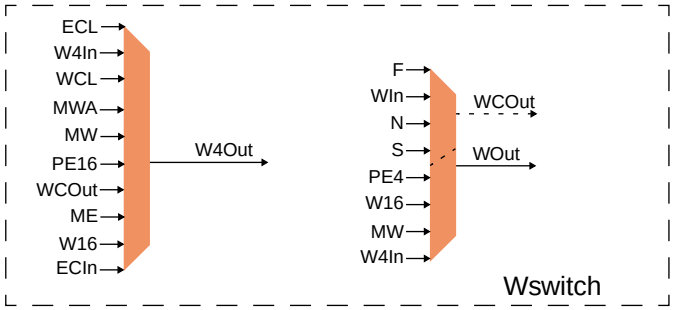


Figure 12. Contents of Wswitch

Previous). e.g. PS4. This feature allows FastLANEs™ to perform U-turns.

Clock Distribution

As described previously, register clock inputs may be driven from any source but it is recommended that the GClk signal is used. GClk also has the advantage that it can be stopped by writing to the Device Configuration Register (see Table 23). The Global wires enter the part through dedicated input pins and are distributed in a special low-skew 'H' pattern (Figure 13). Each vertically aligned (South to North) group of four cells within a 4x4 block is clocked by its own clock source. This is driven from a multiplexer in the Nswitch immediately to the South of the group of cells. The connections for this multiplexer are shown in Figure 9. ClkOut drives the Clk inputs to each of the four cells in the group. As can be seen from Figure 9, the register clock for each group of four cells can be driven by ClkIn, NOut, GClk, GClr, G1, G2 or NCL (N Chip-Length). ClkIn is the ClkOut from the 4x4 block to the South, allowing vertical daisy-chaining of clock signals. NOut is the N output from the cell associated with the Nswitch. This can be used to provide local user-generated or gated clock signals if required. GClk is the Global Clock signal direct from the device GClk input. Clearly this signal only has to pass through one 4:1 multiplexer whereas GClr, G1 and G2 have to pass through two. This is one reason why there is less delay on GClk.

It is also possible to route North Chip-Length wires onto the Clock lines. This allows up to 64 (for a XC6216) locally used clocks to be provided which can still run the entire length of the chip with minimal skew. These local clock signals may be generated internally (e.g. by dividing a faster clock) or sourced directly from the device programmable I/O pins.

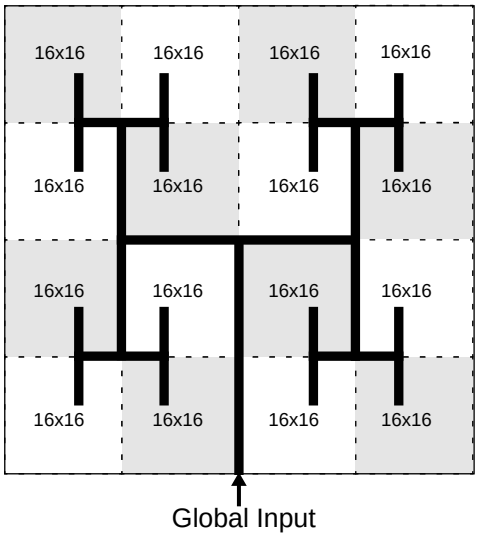


Figure 13. Low Skew 'H' Distribution Of Global Signals (XC6216)

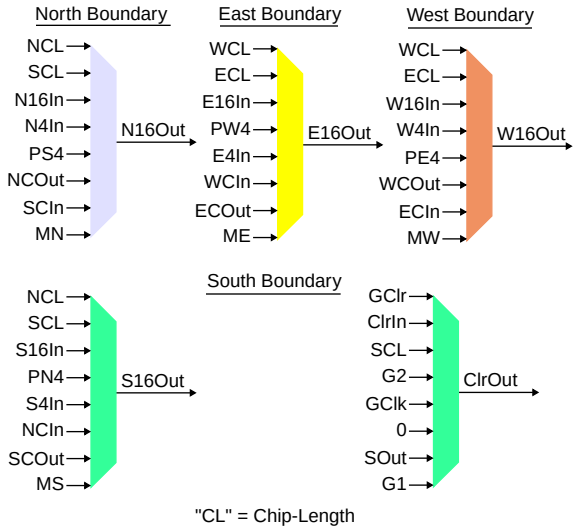


Figure 14. Additional Switches at 16x16 Boundaries

Where a fast clock is required by only a small fraction of the logic on the device it may be preferable to employ user interconnect resources rather than a Global or Chip-Length signal. This is because limiting fast clock distribution to the area of the device where it is required reduces power consumption.

Clear Distribution

Register Clear inputs are routed in a similar manner to Clock inputs. In this case vertical groups of 16 cells, within a 16x16 block, share a common Clear. Clear lines run in a Southerly direction and are sourced from the Sswitch unit of 4x4 blocks which also lie on a 16x16 boundary. All of the boundary switches at 16x16 boundaries contain additional switching multiplexers. These are illustrated in Figure 14. These multiplexers invert some inputs. This is not shown in the figures. See the "Programming" section for details.

ClrOut drives the Clr inputs to each of the sixteen cells in the group. The S and SCL connections allow the output of a cell to provide a user-generated local Clear signal.

I/O Architecture

User-configurable Input/Output Blocks (IOBs) provide the interface between external package pins and the internal logic.

One IOB is provided for every cell position around the array border. IOBs are connected to fixed pad locations. There are more IOBs than available pads, hence some IOBs are 'padless'. However it is still possible to route signals from padless IOBs to device pins.

Figure 15 is a simplified diagram of an IOB and its associated IO pad. The IOB is located at the array border and the pad is located close to its device pin. The pad may be located some distance from its associated IOB. The

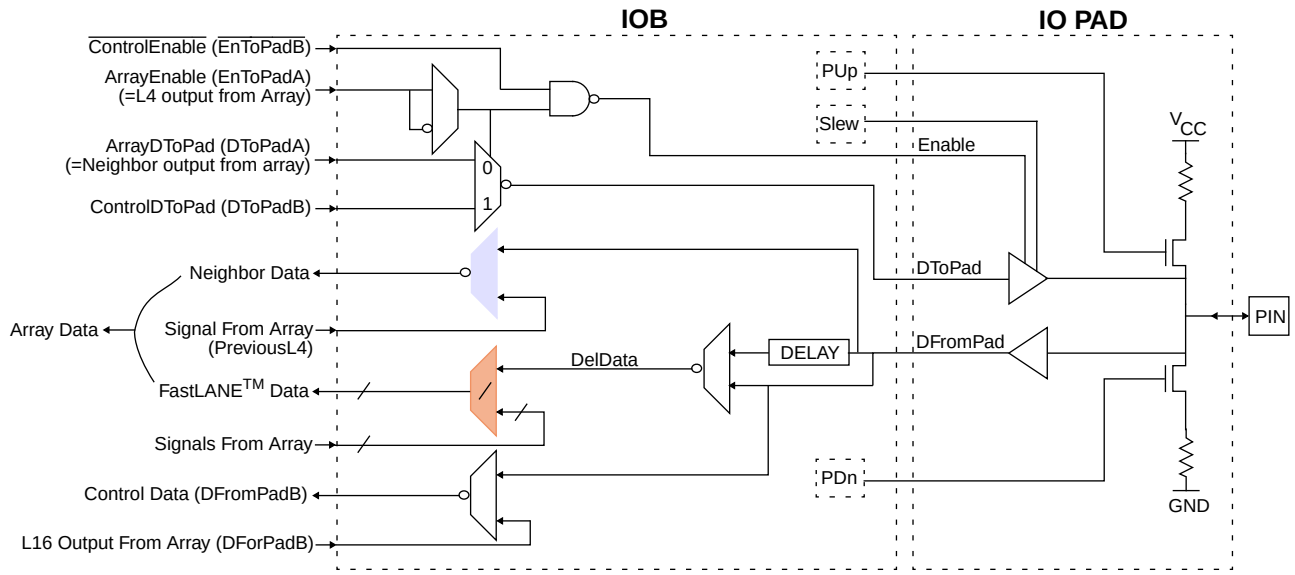


Figure 15. Input/Output Architecture

mapping of IOBs to device pins is given in the pinout tables starting on page 48.

The XC6200 IOB architecture incorporates a novel and very powerful feature: every IOB has the capability of routing either an array signal or a control logic signal to/from the device pin. Every signal, including all the control signals (e.g. $\overline{CS}$, $Rd\overline{Wr}$, *Address Bus*, *Data Bus*, etc.), passes through an IOB. This means that all the control signals can be routed into the logic array for use in user designs. Similarly, user logic can control the XC6200 internal control circuitry. For example a user signal could be used to drive the internal $\overline{CS}$ signal rather than the $\overline{CS}$ pin.

As an example of the power of this feature, an XC6200 design could include an address decoder which decoded microprocessor read/write cycles and produced appropriately retimed signals for all the parts on a board *including itself*, thereby removing the need for address decoding PALs or discrete logic.

Each IOB has an array data input and a control data input, labeled *ArrayDToPad* and *ControlDToPad* in Figure 15. Associated with these inputs are two enable signals - *ArrayEnable* and *ControlEnable*. These signals control whether the pad associated with this IOB is in the input or output mode. Each IOB also supplies *ArrayData* and *ControlData* when acting as an input.

The 'Control' signals are routed to the internal XC6200 control circuitry. If control signals are not required all the time then these IOBs can be used to route other user signals into the array. For example if only eight data bus bits were continuously required, the remaining twenty-four IOBs associated with the data bus could be used to route user signals to/from the array. *ControlEnable* comes either from the internal XC6200 control circuitry if there is a bidirectional control signal or output signal on that IOB, or it is tied inactive.

The 'Control' signals are also referred to as 'B' signals in this data sheet. *ControlDToPad* = *DToPadB*, *ControlEnable* = *EnToPadB* and *Control Data* = *DFromPadB*. The L16 output from the array, which can be routed onto *Control Data*, is also referred to as *DForPadB*.

The *DFromPadB* output is unconnected in IOBs which have an output-only 'B' signal, such as $\overline{SECE}$. IOBs which have an input-only 'B' signal, such as $\overline{CS}$, have the *DToPadB* input tied Low and the $\overline{EnToPadB}$ input tied High. IOBs which have no associated 'B' signal also have

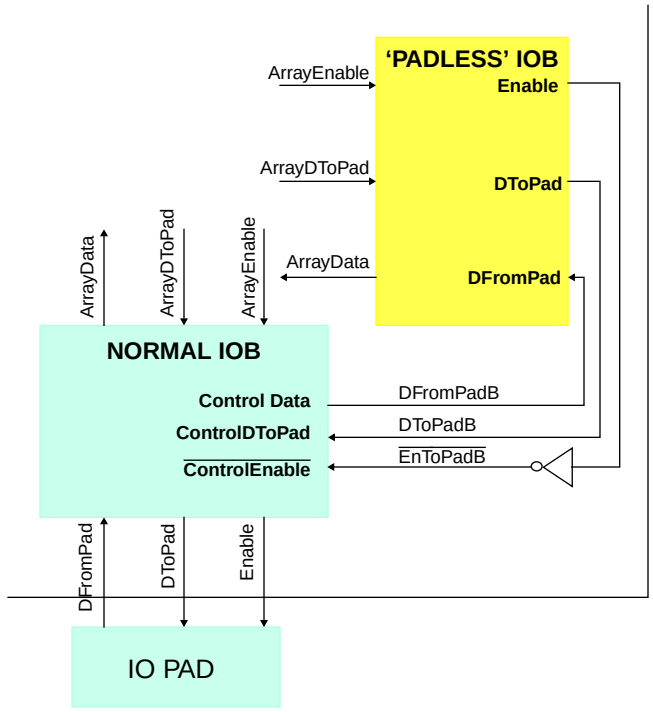


Figure 16. 'Padless' IOB Configuration

DToPadB tied Low, $\overline{EnToPadB}$ tied High and *DFromPadB* unconnected.

There are less real control signals than IOBs, hence the three signals, *DFromPadB*, *DToPadB* and $\overline{EnToPadB}$, on some IOBs are not connected to the XC6200 control logic. Some of these spare 'B' signals are used to route data to and from the padless IOBs mentioned above. The 'B' signals on the padless IOB are not used. This is illustrated in Figure 16. This arrangement allows data to be routed in or out of the chip via an IOB which has no associated IOPAD. The padless IOBs and their padded partner IOBs are detailed in the pinout tables starting on page 48. For example, in a XC6216 IOB W0 is padless and is partnered with IOB S12, which has a pad.

The *ArrayEnable*, *Array Data* and *Control Data* multiplexers are controlled by configuration RAM bits. A fixed delay may be optionally applied to *Array Data* inputs. This allows the input data hold time specification to be removed.

The *ArrayEnable* and *ArrayDToPad* signals can be configured to constant 0 or 1 values within the logic array. The constant values are particularly useful for the enable signal when the pin is to function as an input or output rather than a bidirectional pin. Constant values on the data signal and a computed value on the enable signal produce open drain pull-up (*DToPad*=1) or pull-down (*DToPad*=0) pins.

Table 3: Connections Between IOBs And Built-In XC6200 Control Logic

B Signal Type	Example	***$\overline{EnToPadB}$***	***DToPadB***	***DForPadB***	***DFromPadB***
Input Only	$\overline{CS}$	1	0	L16 Output From Array	Drives XC6200 Control Logic $\overline{CS}$ Input
Output Only	$\overline{SECE}$	Driven By XC6200 Control Logic	*SECE* Out From XC6200 Control Logic	L16 Output From Array	Not Connected
Bidirectional	Data Bus	Driven By XC6200 Control Logic	$\overline{\text{Data-Bus}}$ Out From XC6200 Control Logic	L16 Output From Array	Drives XC6200 Internal FastMAP™ Data Bus Inputs
From Padless IOB	South IOB12	$\overline{\text{Enable}}$ Output From W0 IOB	DToPad Output From W0 IOB	L16 Output From Array	Drives W0 IOB DFromPad Input
None	North IOB30	1	0	L16 Output From Array	Not Connected

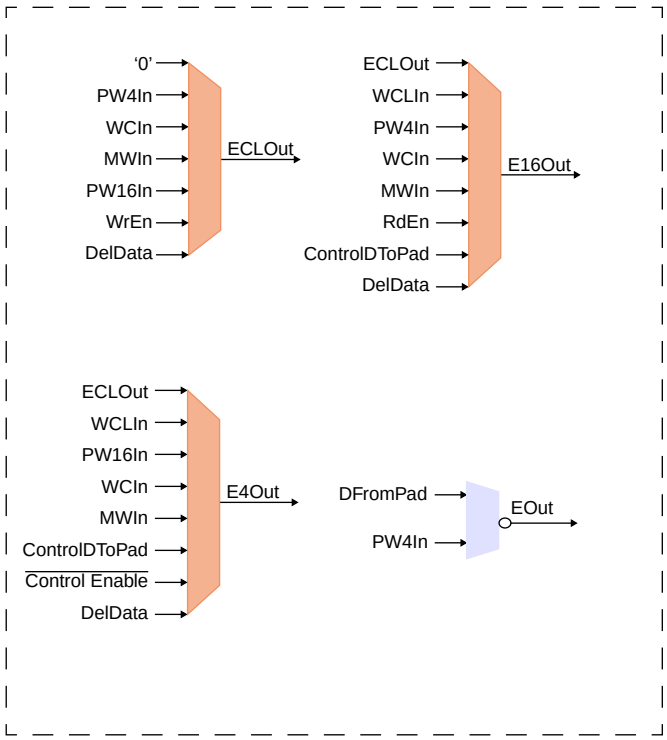


Figure 17. Array Data Sources In West IOBs

Pull-Up, Pull-Down And Slew

Three configuration RAM bits within each IOB control the programmable aspects of its IO pad. These RAM bits have no effect for padless IOBs. 'PUp' and 'PDn' enable the pull-up and pull-down resistors. The resistors may be used to tie floating logic inputs to a known value. 'Slew' slows the output transition time to reduce supply noise and ground-bounce. The default condition is pull-up off, pull-down off and slew on.

During reset, all the output drivers are disabled and the pull-up resistors are enabled. The pull-up and pull-down RAM control bits have no effect. After a reset the output drivers remain in this state. For the output drivers to be enabled, the global $\overline{OE}$ signal must be asserted (Low) and a valid configuration must be present in the device ID register. The ID register is usually the last thing to be written during configuration and acts as a check that the programming interface is operating correctly. More details of this are given in the 'Programming' section. The $\overline{OE}$ signal provides a quick way of disabling all the output drivers and may be activated at any time. Only when $\overline{OE}$ is active and there is a valid ID pattern in the ID register, do the pull-up and pull-down RAM control bits determine the IO-pad resistor configuration. When $\overline{OE}$=1 or the ID pattern is not valid, the pull-ups default to on and the pull-downs to off. The only exception to this default occurs in the Leakage Test Mode (see "Serial Interface State Machine", State 1) where all the pull-up/down resistors are disabled.

Border Routing

The array signals to and from the IOBs are generally just the signals which would have passed between two cells in the array. The *ArrayDToPad* signal in Figure 15 is actually the neighbor output from the border cell associated with the IOB. The *Array Enable* signal is the length-4 FastLAN-E™ output from the same cell.

The *Array Data* multiplexer in Figure 15 is actually a collection of multiplexers which source the neighbor, length-4, length-16 and Chip-Length wires into the array. South IOBs (IOBs at the South edge of the array) also source the local clock signals into the array. North IOBs source the local clear signals. The actual signals which can be routed onto the IOB *Array Data* outputs are detailed in Tables 14 and 19. This is illustrated for a West IOB in Figure 17. Inversions are not shown. See "Programming" section for details. These multiplexers also allow a number of other internal control signals to be routed into the array: *WrEn* and *RdEn* are signals which are active during state register accesses. *RegData* (available on West Chip-Length FastLANEs™) is the state register output value for this row during a state access. The timing of these signals is described under "Timing" on page 17. Note that in order to provide a minimal delay signal path into the core array, the neighbor data output from the IOB cannot select the delayed version of *DFromPad*. Only the un-delayed *DFromPad* and the Previous Length-4 Input can be routed onto the neighbor data output. Therefore the neighbor data output is unaffected by the value of the configuration memory which controls the DelData multiplexer in Figure 15.

The length-4 and length-16 routing multiplexers at the array border also expect some inputs which are not available. For example at the West edge, MEIn, ECIn, PE4In and PE16In are non-existent. These inputs are tied to ground or V_{CC}, thereby providing an abundant source of constant zeros and ones at the array border. These can be used to provide constant values to drive the *ArrayEnable* inputs to IOBs.

GClk, $\overline{OE}$ And $\overline{Reset}$ Routing

The connections from the GClk I/O pad differ from all the other I/O signals. This is necessary because GClk is used to clock all the built-in FPGA control logic.

In common with the $\overline{Reset}$ and $\overline{OE}$ pads, the GClk pad cannot be enabled as an output pad. However, the main difference between these and all the other control signals is that they are routed to the FPGA control logic directly from the pad and **not** from the IOB. This means that a user circuit cannot modify them before they reach the control logic. This is illustrated for GClk in Figure 18.

GClk to the logic array *is* supplied from the IOB. It is this signal which passes through the enable circuit controlled by a single bit in the device control register. Thus it is still

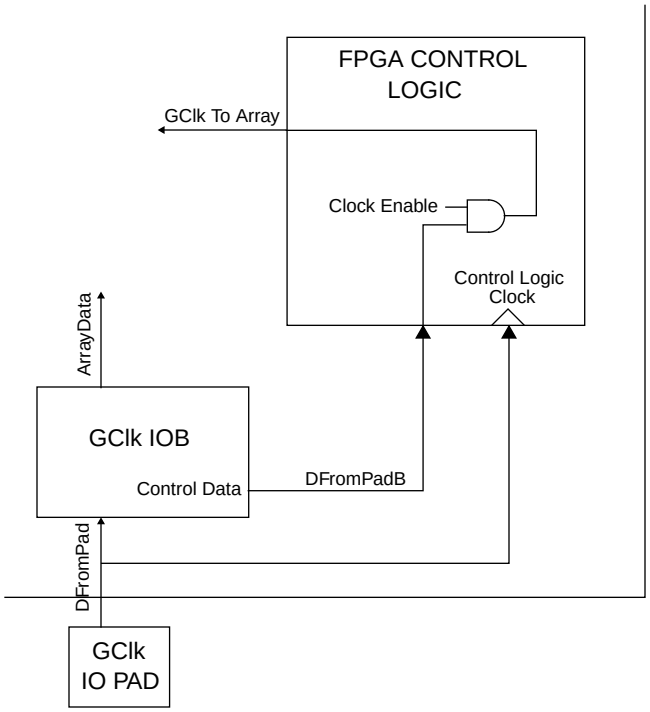


Figure 18. GClk Routing

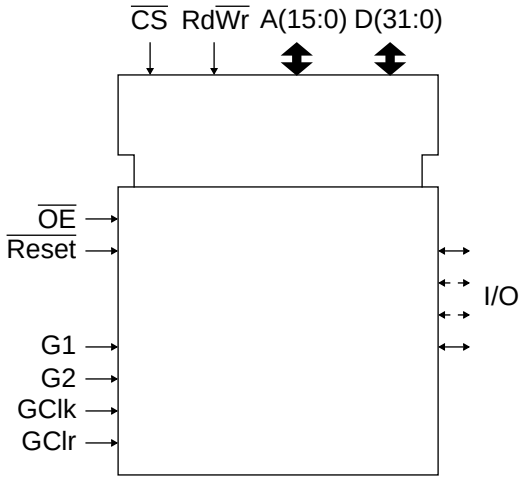


Figure 19. XC6216 Logic Symbol

possible to route any signal onto the array GClk net using the L16 output from the array (see Figure 15). However this GClk can only be safely stopped and started (without glitches) if the DFromPadB signal is the DFromPad signal from the GClk I/O pad.

Designing with XC6200

Designing of XC6200 into systems may be partitioned into three distinct activities.

1. Board design with XC6200

An XC6200 part may be used on a board design as a microprocessor peripheral part, as an ASIC-type device or as both. In the first instance the XC6200 part has conventional SRAM data, address and control signals as illustrated in Figure 19. In other cases it may only require the user defined I/O signals of an ASIC. Packaging information for the part is given in section "Packaging" on page 36. The number of user I/O signals depends on the exact package used.

Several XC6200 devices may be tiled together on a board to form a larger array. The regular array structure of XC6200 makes this particularly easy. East I/Os on one chip would connect to West I/Os on the adjacent chip. North I/Os would connect to South I/Os on the adjacent chip and so on until the required array size was reached. It may be required to use the control signal pins, such as the Data and Address busses, on every part in the array. The control signals use every second IOB, leaving evenly distributed IOBs in between which can be used to interconnect the XC6200s. See the device pin-out tables, starting on page 48 for details.

The configuration RAM bits in the IOBs allow for a number of different programmable options to make interfacing to other ICs easier. This is more fully defined in the "I/O Architecture" section.

2. Logic design with XC6200

This can be approached as an ASIC type design using the function and routing architecture defined in the previous sections. An example design flow is illustrated in Figure 20. The design may be carried out in a variety of different ways. Hardware description languages such as VHDL may be used with the synthesized design targeted to the XC6200 architecture. Alternatively, schematic capture, using the extensive Xilinx Unified Library, with commonly used front end design tools (e.g. ViewLogic PROcapture/ViewDraw) may be used. These tools produce an EDIF netlist which is subsequently passed to the underlying XC6200 place and route software. This automatically maps the user's design to the XC6200 architecture in an efficient way and provides individual node delays which can be passed back to the high level simulation tools such as Viewlogic PROsim/ViewSim for accurate simulation. Simulation may be carried out prior to placement to check the logical correctness of the design using nominal delays. The place and route software also has optimization capability to carry out tasks such as redundant gate removal. A binary configuration file which

can be written to the XC6200 device via the programming interface is also produced automatically. The underlying CAD software is highly integrated with the high level CAD tools, providing user-friendly pull-down menus and dialog boxes to carry out all tasks.

These methods allow designers with little or no knowledge of the XC6200 architecture to quickly produce large and complex designs. Some designers may wish to carry out detailed hand placement and routing to produce ultra-optimized very high-speed/small area sections in their designs. Others may wish to generate large regular structures such as systolic arrays or perform floor-planning for extra efficiency. For these cases, a sophisticated physical editor is available which allows designers to graphically modify the automatic placement of gates/registers into cells and modify the routing as much as required. Alternatively this software may simply be used to see how the automatic placement and routing software has optimized a design. If a modification is subsequently made then only the modified part of the design needs to be re-laid out. This incremental design process gives a very rapid change cycle during debugging.

All the design tasks may be carried out on PC or Unix workstation platforms.

As an example, the simple accumulator circuit of Figure 21 is mapped onto the XC6200 architecture. Figure 22 shows the resulting layout as displayed by the Physical Editor, running under Microsoft Windows in this case. The Physical Editor tools are also available for Unix workstations. The boundaries of basic cells within the array are denoted by the squares, with larger rectangles representing the switch units on 4 cell boundaries. The wiring resources used by the design mapped onto the array are indicated by the darker black lines. When a cell function unit is used by the design it is annotated with the instance names of the mapped primitives. The primary inputs and outputs are not shown in this example. The rectangles around the edge represent IOBs and their pads.

The inputs and outputs to the function unit are connected to the edges of the cell box. The small squares within each cell represent the input ports (X3, X2, X1) and the output port (F).

The Physical Editor allows cells to be selected and moved. The inter-cell routing rubber-bands and adapts automatically to the new placement. The routing may also be manually modified if desired.

Full details of using the software are contained in the online help. An interactive software demonstration is also available.

3. Software design with XC6200

This is the design of a program for the host processor which interacts with a design running on the XC6200. Here various registers within the XC6200 design appear as locations within the processor's memory map. In addition the configuration memory of the device appears

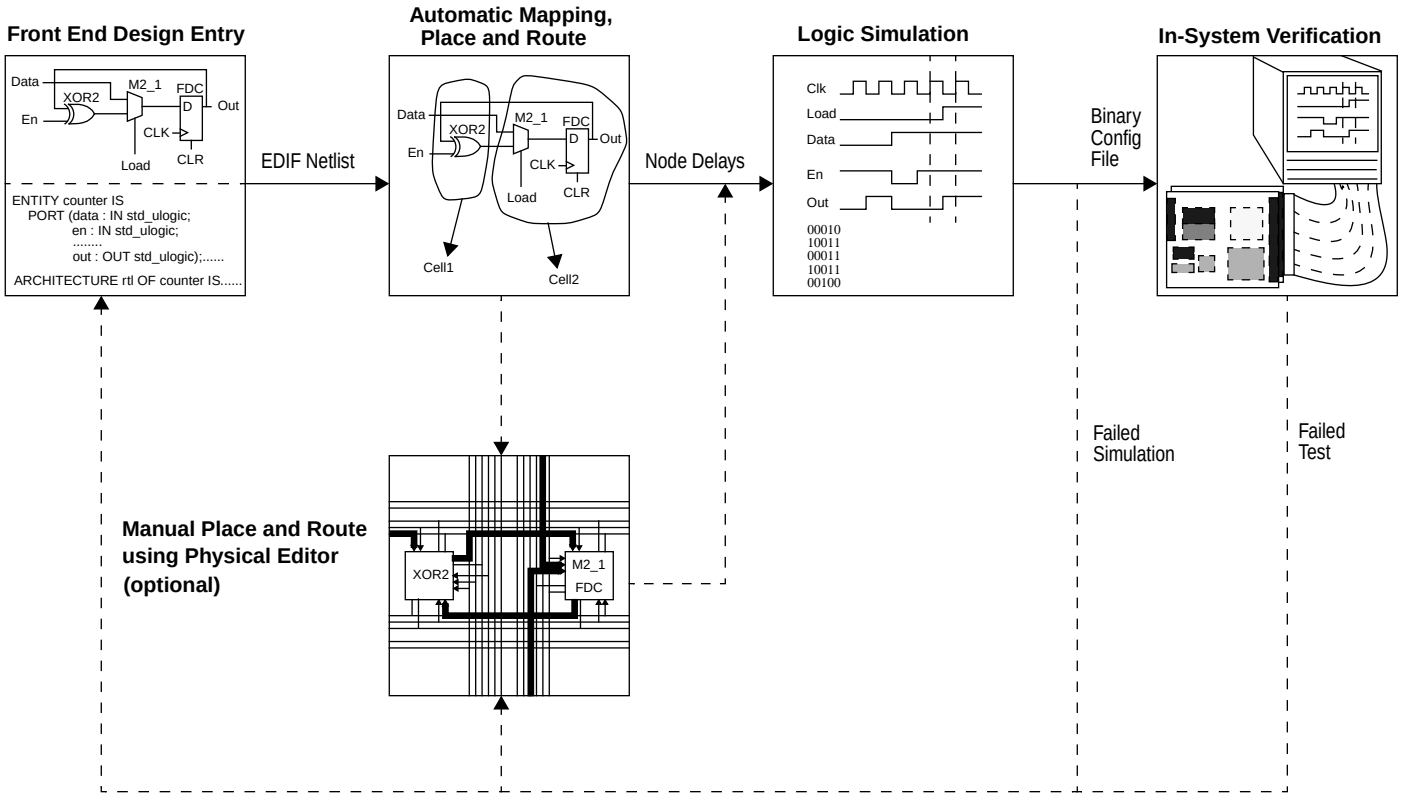


Figure 20. XC6200 Logic Design Flow

within the memory map and portions of the device can be reconfigured as required. Predefined device drivers and an efficient run-time library are available to make optimal use of XC6200's high speed reconfiguration capabilities with minimal development time.

Register Access

XC6200 supports direct accesses from the processor to nodes within the user's circuit: the output of any cell's function unit can be read and the flip-flop within any cell can be written. During state reads a number of cell outputs are routed onto the CPU data bus. The signal which is actually read is either C or S in Figure 6, depending on whether the combinatorial or sequential output is selected. See Table 11 on page 21 for details of signal inversions. These inversions are cancelled out by the readback circuit so that the true value of C or S is read.

These accesses are carried out through the control store interface and involve no additional wiring within the user's design. The CPU interface signals involved in addressing the cell state can be routed into the configurable array so that user circuits can detect that an access has been made and take appropriate action: for example, calculate a new value for an output register or process a value placed in an input register.

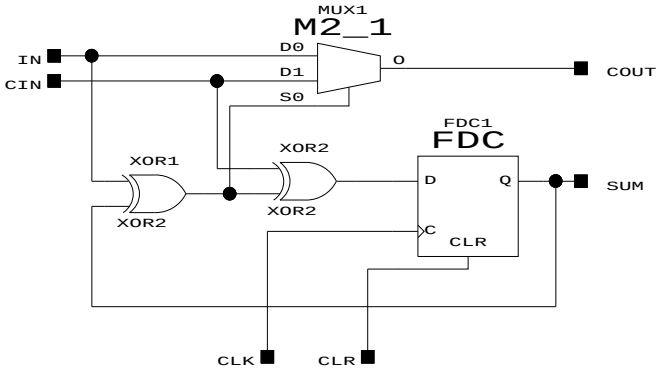


Figure 21. Accumulator Schematic

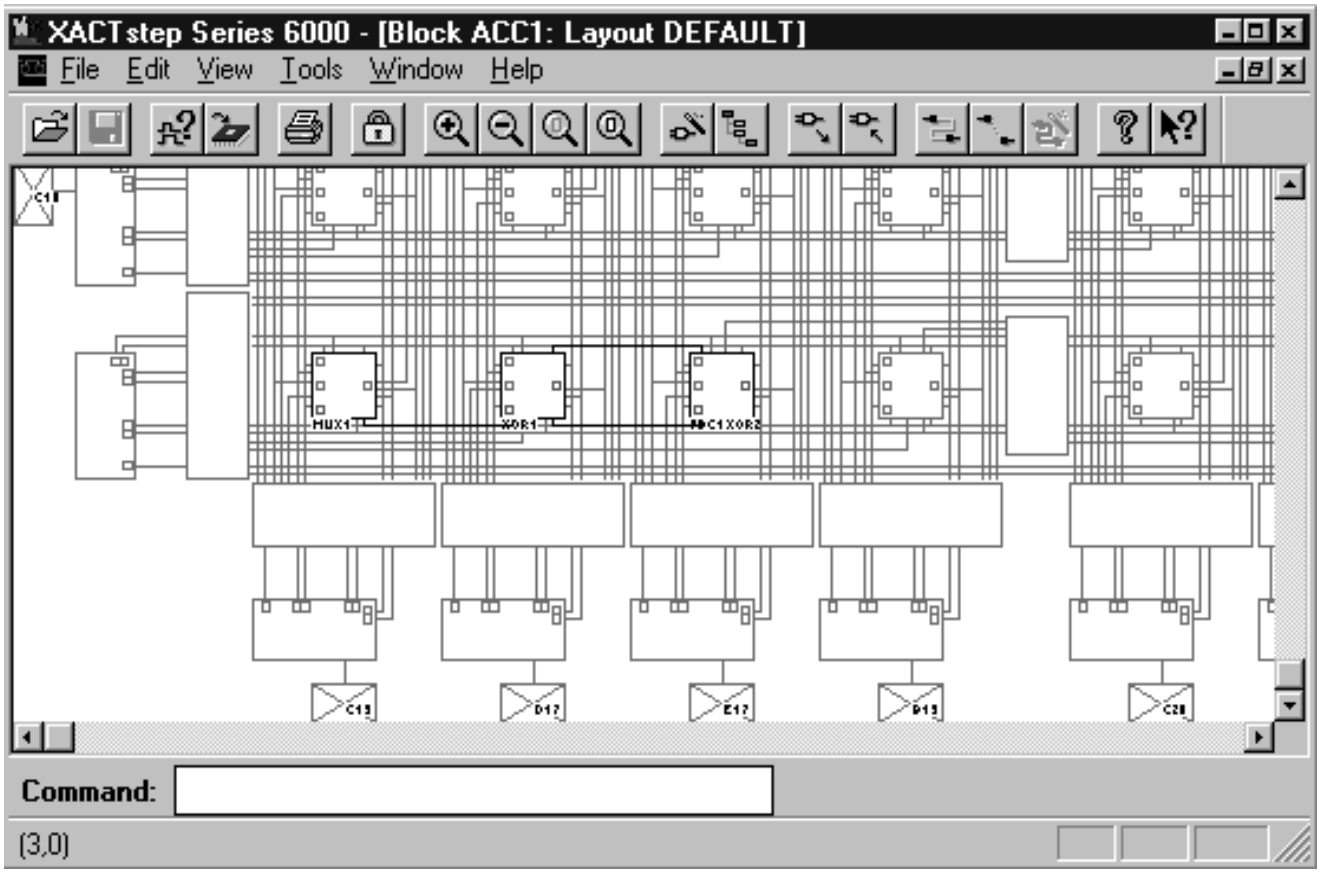


Figure 22. Accumulator Physical Editor View

In many applications this access to internal nodes is the main path through which data is transferred to the processor and in some coprocessor type applications it may be the only external I/O method: user programmable I/O pads may not be required at all.

To allow high bandwidth transfers between the processor and internal nodes it is necessary to be able to transfer a complete processor data word of up to 32 bits in one memory cycle. For this reason state access bits within XC6200 are mapped into a separate region of the device address space from configuration bits so that all the bits in a word contain state access bits. Figure 23 is a block diagram of the XC6216 part, showing the row and column address decoders. Figure 24 shows the mapping of this area of the address space: there are 64 I/O signals from each column of cells and a 6-bit column address selects a particular column of cells to access. This row and column addressing scheme puts a constraint on the placement of registers within the user's design which are to be accessed word-wide: they must be on the same column of cells within the array.

Timing

Externally, timing for state accesses is the same as for reads and writes to configuration SRAM. This is illustrated in Figures 30 and 31 and described in the section "Parallel CPU Interface" on page 19. Internally, state accesses differ from configuration SRAM accesses slightly in that writes to registers are synchronous. When the processor writes data to registers, the data is clocked in on a rising edge of the register's clock. The FastMAP™ interface presents the CPU data on the register's 'D' input and expects the register to clock it in. The internal timing relationships are shown in Figure 32 on page 46. Data is present on the 'D' input from just after the first falling GClk edge of the Write Cycle to the second falling clock edge. If GClk is routed to the register then the data is clocked in at time t_2 in Figure 32. If GClk is not being used to clock the register then the designer must ensure that the register's clock has a rising edge during the period the data is present.

During state reads there is no requirement for the registers to be clocked. Data is read from the output of the cells and sampled by the FastMAP™ interface at time t_2 in Figure 33. Thus the data should be stable before and after this edge of GClk to guarantee a correct read.

The internal timing relationships illustrated in Figures 32 and 33 are the same regardless of whether the cycle is stretched by holding $\overline{CS}$ low, as shown in the extended cycles in Figures 30 and 31.

Map Register

XC6200 provides a mechanism for mapping all the possible cell outputs from a column onto the 8,16 or 32-bit external data bus, selecting only those cells which implement bits of the register to be accessed. Without this unit the processor would have to implement a complex sequence of shift and mask operations to discard those bits corresponding to cells not within the register, or the user would have to constrain the layout so that the register bits were in adjacent cells. The mechanism provided by XC6200 takes the form of a **Map Register**, one bit for each row I/O signal from the array. This

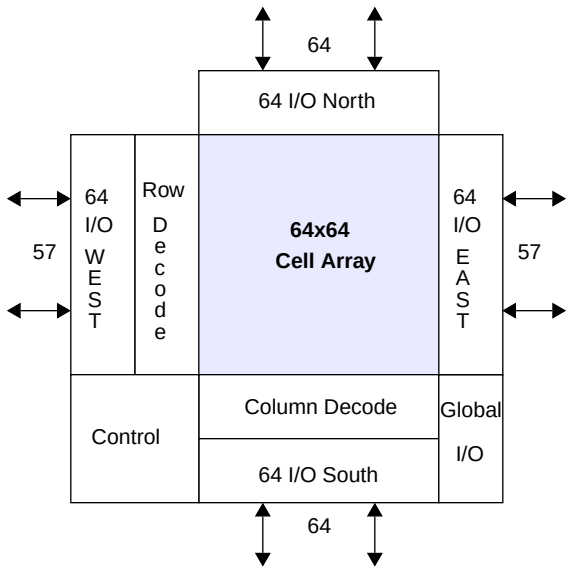


Figure 23. XC6216 Block Diagram

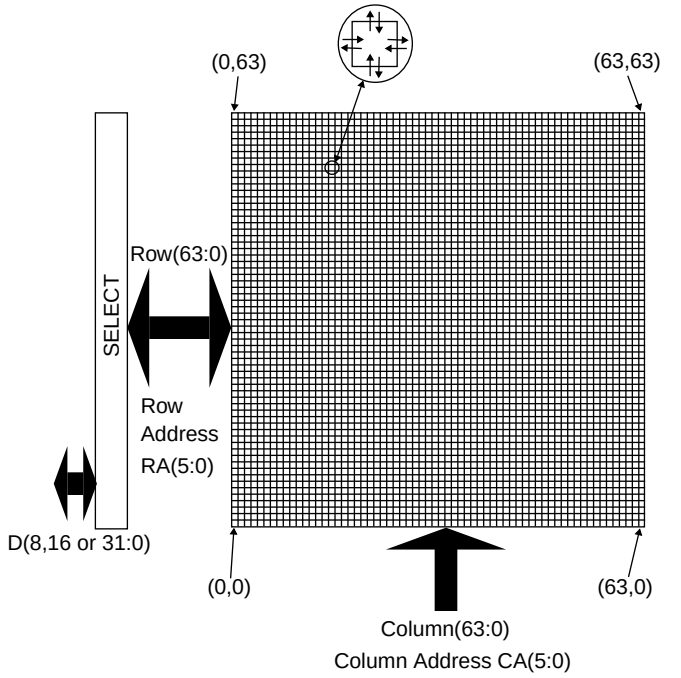


Figure 24. Memory Mapped I/O

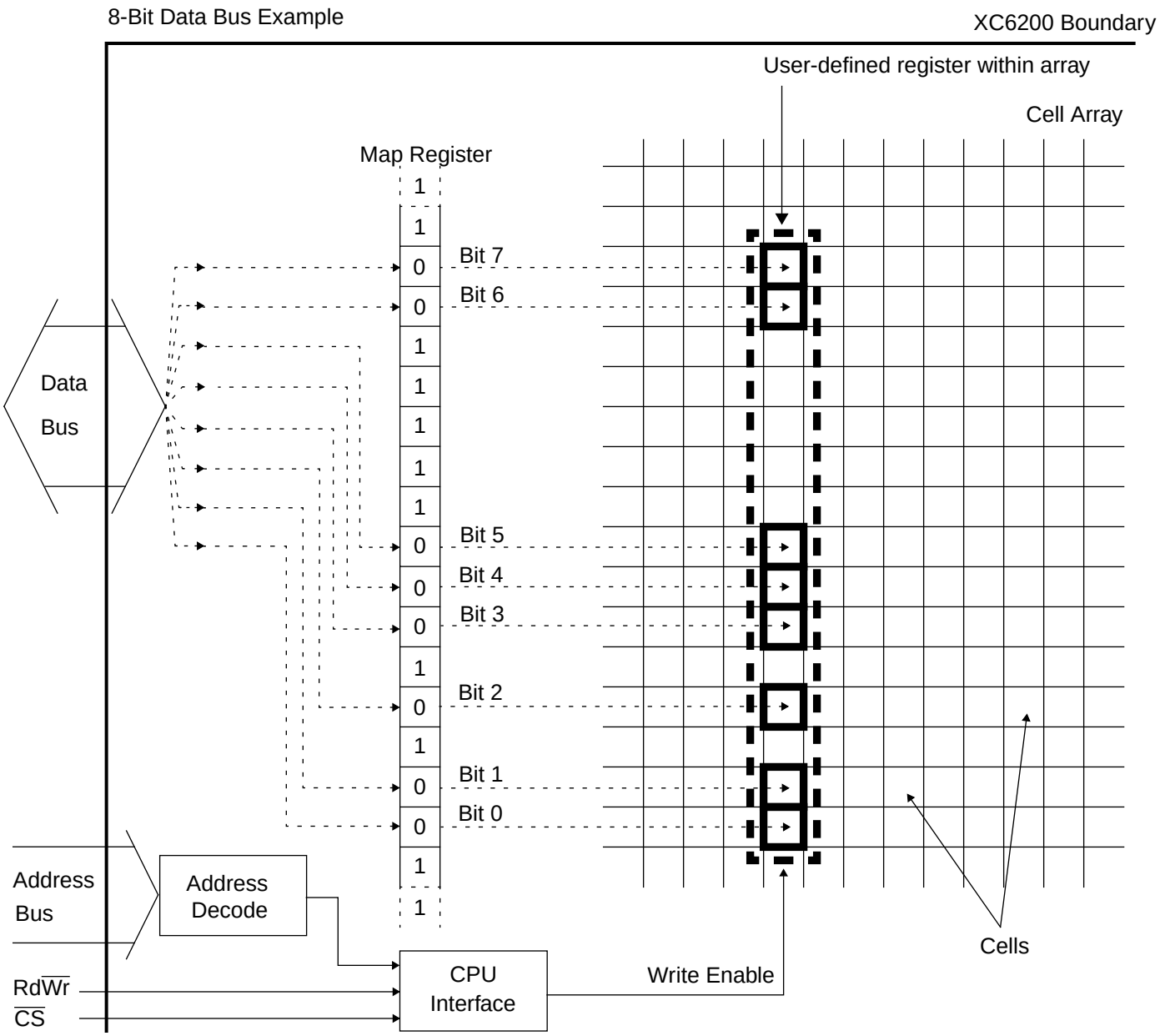


Figure 25. Internal Register Access

Map Register can be read and written through the control store interface and is set up prior to state accesses. A logic 0 in the Map Register indicates that the cell in the corresponding row is part of the register to be accessed. The unit maps rows from the cell array onto external data lines starting with the least significant bit: thus the first row with a 0 in the Map Register connects to external data bus bit 0, the second row with a 0 in the Map Register to data bus bit 1 and so on.

This technique puts a further constraint on the user's layout: the cells implementing the bits of the register must be ordered so that less significant bits occur below more significant bits. However, there are no constraints about the relative separations of the cells. In practice these two placement constraints: cells occurring in the same column and in order vertically are easy to meet in datapath type designs.

Normally, the Map Register is set once to indicate the placement of the user I/O register which is then accessed many times. Therefore the two write operations required with a 32-bit bus to set up the Map Register represent a small overhead. In data path type designs where several registers are required, for example two input operand registers and a result register, it is easy to ensure that the corresponding bits of the registers occur on the same row but different columns of the array so that the same Map Register value can be used with different column addresses to access the various registers.

If more 0s exist in the Map Register than there are valid data bus bits then a form of wildcarding occurs during writes. The data bus bits are allocated to the rows of the array with a 0 in their Map Register bit. Once all of the data bus bits have been allocated, Bit 0 of the data bus is allocated to the next row whose Map Register bit is a 0, Bit 1 of the data bus to the next row and so on. This feature means that an entire column of state registers can be written with a single 8-bit write. For example, if the Map Register contains all 0s and the CPU writes FFh to a particular column. All the state registers in that column are written with a 1. The default state of the Map Register is all 0s.

During reads, if there are more 0 bits in the Map Register than data bus bits, the first rows with 0 bits are mapped onto the bus.

If there are fewer 0s in the Map Register than data bus bits, the upper data bus bits, which are not mapped, are undefined during CPU reads and ignored during CPU writes.

An example of Map Register operation is shown in Figure 25. The position of the user-defined register within the cell array is defined by the 0s in the Map Register. Similar registers could be defined for every column in the array if desired.

There is a delay of T_{MPST} (see page 43) after a write to the Map Register before the change takes effect. No state accesses should be carried out during this time. There are important notes on the register clocking requirements for reliable register reading and writing in the "Parallel CPU Interface" section.

Mask Register

A mask unit controlled by a 32-bit register is placed between the external data bus and the internal data connections. When the external data bus is 8 or 16 bits wide only the bottom 8 or 16 bits of this register are significant. A logic '1' in a bit of this register indicates that the corresponding bit of the internal data bus is *not* relevant. Bit locations corresponding to 1s in the Mask Register retain their values when written. On a write operation the corresponding bit line is not enabled and the state information for that bit is not changed. When the device is reset the Mask Register contains all logic 0s corresponding to all data bus bits valid.

During CPU reads, valid register bits which are disabled are read as '0'. Invalid bits (bits which do not physically exist for the register being read) may be read as '0' or '1'.

The Mask Register does not affect state register accesses. In this case the Map Register can be used to prevent certain bits being modified.

The Mask Register is also ignored during reads and writes to XC6200 Control Registers. These are memory locations which control various XC6200 functions and are defined in "Address Mode 11 - Programming Control Registers" on page 25

Programming

The binary data for configuring XC6200, generated by CAD software from the textual description of a user design, must be downloaded into the part itself. This may be performed in several ways. Generally the fastest and most efficient way is by writing directly to the control store, mapped into the address space of a host processor. If a microprocessor or other parallel data source is not available then the serial programming interface may be used.

Parallel CPU Interface

XC6200 has a full parallel CPU interface, referred to as 'FastMAP^{TM}'. This makes all the configuration SRAM and logic cells appear as conventional memory mapped SRAM. The FastMAP^{TM} interface is based on Chip Select ($\overline{CS}$) and Read/Write ($Rd\overline{Wr}$) control signals. The $\overline{CS}$ signal can be used to address a single part within an array of devices and allows data to be read or written. Timing for these signals is illustrated in Figures 30 and 31. These figures show that the programming interface is synchronous. The GClk input is used to sample all the interface signals. GClk is also used when accessing user registers as illustrated in Figure 25. This is an important point, as only registers clocked directly by GClk can be reliably read or written using this method. This is because the value written by the CPU is presented to the inputs of the cell registers just before t_2 in Figure 30 and held there until GClk goes Low again. Thus it is essential that the register receives a rising clock edge at t_2. This can be guaranteed if GClk is used to clock the registers. If another signal is used then it must have a rising edge at t_2 for FastMAP^{TM} register writes to work. For reliable register reads, the register contents must be stable between t_1 and t_2 in Figure 31. This is more fully illustrated in Figure 33.

Figure 31 shows two separate read cycles - a normal cycle immediately followed by an extended cycle. In the normal read cycle $\overline{CS}$ is sampled Low on the first rising GClk edge (t_1) and High on the next (t_2). The data bus is then driven until the next rising GClk edge (t_3). In cases where this is not long enough, the read cycle can be extended by keeping $\overline{CS}$ asserted beyond t_2. This is equivalent to adding wait states. In this case the data bus is driven until $\overline{CS}$ is deasserted. $\overline{CS}$ should not be allowed to go High and Low again. This would cause another cycle to begin. $\overline{CS}$ is sampled on every rising GClk edge. Other CPU interface signals such as $Rd\overline{Wr}$

and the Address Bus are only sampled on the first GClk edge of the cycle (t_1 for the first cycle and t_3 for the second in the figure examples).

Extended write cycles are also possible, however these are functionally no different to normal write cycles, the data and address busses still being sampled on the first rising GClk edge of the cycle (t_3 in Figure 30).

$\overline{CS}$ must always be sampled as a '1' before the next cycle can begin. In Figure 31 the extended read cycle starts immediately after the normal read cycle at time t_3. A write cycle could not start until the *next* rising GClk edge as the data from the read cycle is still on the data bus.

The SRAM programming interface is supplemented by additional hardware resources designed to minimize the number of processor cycles required for reconfiguration. These resources are initially inactive after a reset so the device looks like an SRAM. Before any data can be read from the device using the FastMAP™, the Device ID Registers must all be correctly written. This is descibed under "Device Identification Register" on page 26.

The control store layout is designed to minimize the overhead of computations required for dynamic access while maintaining adequate density to minimize the external storage required for device configurations. When an external processor is used to configure the device it may be convenient to use a compressed format of the configuration information.

A feature of the XC6200 architecture is that a rectangular area of cells specified as a hierarchical block within a user's design corresponds directly with a rectangular area within the configuration memory of the XC6200 device. This means that a block within the user's design can be dynamically replaced with another block by the host processor, reconfiguring only the corresponding area of the control store. The binary data for both blocks can be precalculated from the cellular design and the actual replacement can be carried out very rapidly using block transfer operations.

The format of the address bus to the XC6216 device is shown in Table 4. Larger XC6200 devices have proportionally more bits allocated to row and column addresses.

Mode(1:0)	Column(5:0)	Column Offset<1:0>	Row(5:0)
15:14	13:8	7:6	5:0

Table 4: Address Bus Format (XC6209 and XC6216)

Mode(1:0)	Column(6:0)	Column Offset<1:0>	Row(6:0)
17:16	15:9	8:7	6:0

Table 5: Address Bus Format (XC6236 and XC6264)

All the configuration memory can be accessed as 8-bit bytes. When a 16-bit transfer occurs Address<0> is irrelevant. When a 32-bit transfer occurs Address<1:0> is irrelevant. Data Bus bits <7:0> are written to the address with Address<1:0>=00, bits <15:8> are written to the address with Address<1:0> = 01, etc. The Address Mode bits are used to determine which area of the control store is to be accessed according to Table 6.

Mode1	Mode0	Area Selected
0	0	Cell Configuration and State
0	1	East/West Switch or IOB
1	0	North/South Switch or IOB
1	1	Device Control Registers

Table 6: Address Mode Selection

Understanding The Configuration Bits

The full memory map is given in Tables 26 and 27. From these tables it is possible to work out the address for any bytes of configuration or cell state register in the FPGA. The address/data pairs are normally calculated automatically by XACT*step* Series 6000 and written to a .cal file.

From Tables 7 to 21 it is possible to work out what data needs to be written to the above addresses to change the configuration of any routing multiplexer or cell in the FPGA. These tables are split into subsections - Cells, East/West Switches and North/South Switches. Within each section, the bytes which control the switches are defined first. This table is then followed by a group of tables which define the coding of the bits within these bytes.

This sequence of tables details the coding for the various multiplexer select lines. These are always referred to as 'Sel'. The columns refer to the output of the appropriate multiplexer when Sel is at a particular value. As an example, in Table 7, Cell Routing Register Byte 00, Bits [7:6] correspond to the North Neighbor Multiplexer Sel[1:0] in Table 8. When Sel[1:0] = 10, the North output of the cell is routed from the East Neighbor input.

As another example, in Table 13, the EOut multiplexer in the West IOB is controlled by Bit4 of Byte 1. This corre-

sponds to the single Sel bit in the West IOB block of Table 14. Thus when Bit4=0, EOut=$\overline{PW4}$. When Bit4=1, EOut=$\overline{DFromPad}$.

Note that most of the routing multiplexers invert their outputs to reduce propagation delays. This has not been shown in earlier figures.

Address Mode 00 - Cell Mode

In Mode 00 the 6-bit row and column addresses are effectively a Cartesian coordinate pointer to a particular cell. (0,0) is the cell in the South-West corner of the array. Once a particular cell has been pin-pointed, the 2-bit Column Offset determines which cell configuration RAM byte is accessed. Each cell has 3 separate 8-bit configuration bytes and 1 single-bit state register. The state register is the cell register shown in Figure 6. The three cell configuration bytes are described below.

In Mode 00 bytes read from and written to the control store have the format shown in Table 7. Bit 7 is the msb. Column Offset = 00 addresses the neighbor routing multiplexers select lines. Thus a single byte controls all the neighbor routing multiplexers within a basic cell. Bytes 01 and 10 control the remaining cell routing. See Figures 5 and 6 for the connections to these multiplexers.

Column Offset<1:0>	DATA BIT							
	7	6	5	4	3	2	1	0
00	North		East		West		South	
01	CS	X1[2:0]			X2[1:0]		X3[1:0]	
10	M	RP	Y2[1:0]		Y3[1:0]		X3[2]	X2[2]

Table 7: Cell Routing

CS is the Combinatorial/Sequential multiplexer select line. M is the Magic multiplexer select line and RP is the Register Protect bit for the cell.

Column Offset = 11 is used for state accesses. In this case the Data Bus bit values are determined by the Map Register. The Row bits of the address bus are ignored.

The multiplexer which selects between Y2 and Y3 in Figure 6 and the RP multiplexer also invert. Thus the inputs to the RP multiplexer are actually C and $\overline{Q}$.

Sel[1:0]	North	South	East	West
00	$\overline{F}$	$\overline{F}$	$\overline{F}$	$\overline{F}$
01	$\overline{N}$	$\overline{E}$	$\overline{N}$	$\overline{W}$
10	$\overline{E}$	$\overline{W}$	$\overline{E}$	$\overline{N}$
11	$\overline{W}$	$\overline{S}$	$\overline{S}$	$\overline{S}$

Table 8: Neighbor Multiplexer Selection

Sel[2:0]	X1	X2	X3
000	$\overline{S}$	$\overline{S}$	$\overline{S}$
001	$\overline{E}$	$\overline{W}$	$\overline{E}$
010	$\overline{W}$	$\overline{E}$	$\overline{W}$
011	$\overline{N}$	$\overline{N}$	$\overline{N}$
100	$\overline{W4}$	$\overline{W4}$	$\overline{W4}$
101	$\overline{S4}$	$\overline{E4}$	$\overline{S4}$
110	$\overline{E4}$	$\overline{S4}$	$\overline{E4}$
111	$\overline{N4}$	$\overline{N4}$	$\overline{N4}$

Table 9: X Multiplexer Selection

Sel[1:0]	Y2	Y3
00	X2	X3
01	$\overline{Q}$	$\overline{X3}$
10	$\overline{X2}$	Q
11	Q	$\overline{Q}$

Table 10: Y Multiplexer Selection

Sel	CS	M	RP
0	$\overline{Q}$	$\overline{X3}$	NOT PROTECTED (D)
1	$\overline{C}$	$\overline{X2}$	PROTECTED (Q)

Table 11: Remaining Configurable Cell Multiplexers

Address Mode 01 - East/West Switches And IOBs

Mode 01 addresses the switches which control the East and West FastLANEs™ and the IOBs along the East and West edges. The upper Column address bits pin-point a particular 4x4 block. Column[1:0] then selects which edge of the 4x4 block.

Column[1:0]	Switches
00	West edge of 4x4 block
01	West IOB (Col[5:2] = 0000)
10	East IOB (Col[5:2] = 1111)
11	East edge of 4x4 block

Table 12: East/West Column Decoding

The IOBs are only addressed if Column[5:2] = 0000 or 1111, in a XC6216.

The Row address bits select an individual row within the array. For example 000000 selects the row at the South edge of the array and 111111 selects the row at the North edge of the array, in a XC6216.

Having pin-pointed a particular switch group, the Column Offset selects the exact switch required. For length-4 and 16 FastLANE™ switches within the array, Column Offset = 00. If an IOB is selected then the Column Offset selects a particular register within the IOB.

East/West IOB Configuration

Column Offset <1:0>		DATA BIT							
		7	6	5	4	3	2	1	0
East IOB	00	W4[2:0]			Del	W	W16[2:0]		
	01	WCL[2:0]			En	DfPB	$\overline{Slew}$	PDn	PUp
West IOB	00	PUp	PDn	$\overline{Slew}$	DfPB	En	ECL[2:0]		
	01	E16[2:0]			E	Del	E4[2:0]		

Table 13: East/West IOB Configuration Registers

See Figure 15 for details of the IOB architecture. 'Del' allows a fixed delay to be added to the DFromPad signal. 'E' and 'W' are the neighbor outputs into the array. 'E4', 'W4', etc are the FastLANE™ outputs into the array. 'DfPB' is the select line for the Control Data (DFromPadB) output from the IOB. 'En' allows the Array Enable signal to be inverted. PUp = 1 enables the pad pull-up resistor. PDn = 1 enables the pad pull-down resistor. $\overline{Slew}$ = 0 causes the output driver to slew its output. See "Pull-Up, Pull-Down And Slew" on page 13.

The coding for the individual multiplexers is as follows:

West IOB				
Sel	Del	En	E	DfPB
0	Delay	W4In	$\overline{PW4}$	$\overline{DFromPad}$
1	No Delay	$\overline{W4In}$	$\overline{DFromPad}$	$\overline{W16}$

West IOB			
Sel[2:0]	E4 Mux	E16 Mux	ECL Mux
000	ECLOut		0
001	WCLIn		PW4
010	PW16	PW4	WCIn
011	WCIn		MW
100	MW		PW16
101	DToPadB	$\overline{RdEn}$	$\overline{WrEn}$
110	EnToPadB	DToPadB	-
111	DelData		

East IOB				
Sel	Del	En	W	DfPB
0	Delay	$\overline{E4In}$	$\overline{PE4}$	$\overline{DFromPad}$
1	No Delay	E4In	$\overline{DFromPad}$	$\overline{E16}$

East IOB			
Sel[2:0]	W4 Mux	W16 Mux	WCL Mux
000	WCLOut		0
001	ECLIn		PE4
010	PE16	PE4	ECIn
011	ECIn		ME
100	ME		PE16
101	DToPadB	$\overline{RdEn}$	$\overline{WrEn}$
110	EnToPadB	DToPadB	RegData
111	DelData		

Table 14: East/West IOB Configuration Coding

$\overline{RdEn}$ and *$\overline{WrEn}$* are signals which are active during reads and writes to state registers in the row corresponding to the IOB. They can be routed back into user designs to detect CPU reads and writes of state registers. The CAD libraries provide special 'CBUF' primitives to access this feature. These signals go active after the first rising clock edge of a CPU cycle and go inactive after the second falling clock edge. *RegData* is the value being written to or read from a state register during a state access. This is illustrated in Figures 32 and 33 on page 46. The FastMAP™ uses a current sensing technique to read *RegData* internally, therefore if the GClk frequency is high, the voltage on *RegData* may not actually appear to change during the Read Cycle. For this reason *RegData* should only be used during Read Cycles if GClk is running at a frequency < TBAns.

East/West Switch Configuration

Column Offsets of 00 select array routing switches at 4x4 or 16x16 boundaries:

	DATA BIT							
	7	6	5	4	3	2	1	0
East	E16[2:0]			E[2]	E4[3:0]			
West	W4[3:0]				W[2]	W16[2:0]		

Table 15: East/West Routing Configuration Registers

The length-16 FastLANE™ bits are only of relevance at 16x16 boundaries. The E[2] and W[2] bits are the MSBs controlling the cell neighbor output. The LSBs are the normal neighbor selection bits in the cell routing register. See "Routing Switches" on page 8.

Sel[2:0]	E Mux	W Mux	E16 Mux	W16 Mux
000	$\overline{F}$	$\overline{F}$	ECL	WCL
001	$\overline{N}$	$\overline{W}$	WCIn	PE4
010	$\overline{E}$	$\overline{N}$	E4In	W4In
011	$\overline{S}$	$\overline{S}$	PW4	WCOut
100	$\overline{PW4}$	$\overline{PE4}$	ECOut	ECIn
101	$\overline{ME}$	$\overline{W16}$	WCL	ECL
110	$\overline{E16}$	$\overline{MW}$	E16	W16
111	$\overline{E4In}$	$\overline{W4In}$	ME	MW

Sel[3:0]	E4 Mux	W4 Mux	Sel[3:0]	E4 Mux	W4 Mux
0000	MW	$\overline{ECL}$	1000	ECOut	-
0001	MEA	W4In	1001	E4In	WCOut
0010	W16*	$\overline{WCL}$	1010	ME	-
0011	WCL	MWA	1011	E16	ME
0100	$\overline{WCIn}$	-	1100	$\overline{ECL}$	-
0101	-	MW	1101	-	W16
0110	-	-	1110	-	-
0111	-	E16*	1111	-	ECIn

* E4Out at East edge of 16x16 uses PW16. W4Out at West edge of 16x16 uses PE16.

Table 16: East/West Routing Configuration Coding

At the edge of the array some inputs clearly do not make sense. For example there can be no PE4 input to a W Multiplexer at the West edge of the array as there are no cells to the West of this multiplexer. In these cases the signals are tied to ground. The only exceptions are the Previous16 inputs to the length-4 multiplexers, which are tied to V_{CC}. Thus for an E4 multiplexer at the extreme Eastern edge of the array, MW=0, PW16=1 and WCIn=0.

Address Mode 10 - North/South Switches And IOBs

Mode 10 addresses the switches which control the North and South FastLANEs™ and the IOBs along the North and South edges. Row[5:2] pin-points a particular 4x4 block. Row[1:0] then selects which edge of the 4x4 block.

Row[1:0]	Switches
00	South edge of 4x4 block
01	South IOB (Row[5:2] = 0000)
10	North IOB (Row[5:2] = 1111)
11	North edge of 4x4 block

Table 17: North/South Row Decoding

The IOBs are only addressed if Row[5:2] = 0000 or 1111.

Column[5:0] selects an individual column within the array. For example 000000 selects the column at the West edge of the array and 111111 selects the column at the East edge of the array.

Having pin-pointed a particular switch group, the Column Offset selects the exact switch required. If an IOB is selected then the Column Offset selects a particular register within the IOB.

North/South IOB Configuration

Column Offset <1:0>		DATA BIT							
		7	6	5	4	3	2	1	0
North IOB	00	Del	En	DfPB	S	S16[2:0]			-
	01	SCL[2:0]			-	S4[0]	-	-	-
	10	PUp	PDn	$\overline{Slew}$	S4[2:1]		Clr[2:0]		
South IOB	00	-	-	Clk[1:0]		N	DfPB	En	Del
	01	N16[2:0]			Clk[2]	N4[0]	$\overline{Slew}$	PDn	PUp
	10	-	-	-	N4[2:1]		NCL[2:0]		

Table 18: North/South IOB Configuration Registers

The coding for the individual multiplexers is shown below:

North IOB				
Sel	Del	En	S	DfPB
0	Delay	$\overline{N4In}$	$\overline{PN4}$	$\overline{DFromPad}$
1	No Delay	N4In	$\overline{DFromPad}$	$\overline{N16}$

North IOB				
Sel[2:0]	S4 Mux	S16 Mux	SCL Mux	Clr Mux
000	SCLOut		0	GClr
001	NCLIn		PN4	GClk
010	PN16	PN4	NCIn	G1
011	NCIn		MN	G2
100	MN		PN16	SOut
101	DToPadB	$\overline{RegWord}$	$\overline{RegWord}$	0
110	EnToPadB	DToPadB	EnToPadB	DelData
111	DelData			SCLOut

South IOB				
Sel	Del	En	N	DfPB
0	Delay	$\overline{S4In}$	$\overline{PS4}$	$\overline{DFromPad}$
1	No Delay	S4In	$\overline{DFromPad}$	$\overline{S16}$

South IOB				
Sel[2:0]	N4 Mux	N16 Mux	NCL Mux	Clk Mux
000	NCLOut		0	NCLOut
001	SCLIn		PS4	NOut
010	PS16	PS4	SCIn	0
011	SCIn		MS	G1
100	MS		PS16	G2
101	DToPadB	$\overline{RegWord}$	$\overline{RegWord}$	GClr
110	EnToPadB	DToPadB	EnToPadB	GClk
111	DelData			

Table 19: North/South IOB Configuration Coding

RegWord is the Word line which is asserted when state registers in the column corresponding to the IOB are read or written. This may be routed back into user designs as a means of detecting CPU reads and writes of state registers. This has many applications. For example it could be used to implement a 'Wait-Signal' semaphore system where a register must be read before it can be re-written. *RegWord* is a pulse which starts after the first falling clock edge of a CPU cycle and ends after the second falling clock edge. This is illustrated in Figures 32 and 33 on page 46. The CAD libraries provide special 'CBUF' primitives to access this feature.

North/South Switch Configuration

Column Offset <1:0>		DATA BIT							
		7	6	5	4	3	2	1	0
North Switch	00	-	-	-	-	-	PrimaryClk[1:0]		N[2]
	01	-	-	-	N16[2:0]			N4[1:0]	
	10	-	-	-	-	-	-	N4[3:2]	
	11	-	-	-	-	SecClkA[1:0]		SecClkB[1:0]	
South Switch	00	S[2]	S16[2:0]			-	-	-	-
	01	S4[3:2]		-	-	-	-	-	-
	10	S4[1:0]		Clr[2:0]		-	-	-	-

Table 20: North/South Routing Configuration Registers

The clock multiplexer in each North Switch is split into two sections (see Figure 9 on page 9). The multiplexer which drives ClkOut is referred to as the primary clock multiplexer (PrimaryClk). The multiplexer whose output drives an input of the primary clock multiplexer is referred to as the secondary clock multiplexer (SecClk). Column Offset = 11 in the North Switch addresses the control lines for the secondary clock multiplexers. There is a separate secondary clock multiplexer for each column of cells in a 4x4 block. They are written to in pairs with Column[1:0] determining which pair is addressed. If Column[1:0] = 00 then SecClkA = Column1 (within 4x4 block), SecClkB = Column0. If Column[1:0] = 11 then SecClkA = Column2, SecClkB = Column3. Column[1:0] = 01 and 10 are illegal. Row[1:0] must be set to 11.

The length-16 FastLANE™ and Clr bits are only of relevance at 16x16 boundaries, otherwise they are non-existent. The N[2] and S[2] bits are the MSBs controlling the cell neighbor output. The LSBs are the normal neighbor selection bits in the cell routing register. See "Routing Switches" on page 8.

Sel[2:0]	N Mux	S Mux	N16 Mux	S16 Mux	Clr Mux
000	$\overline{F}$	$\overline{F}$	NCL	SCL	GClr
001	$\overline{N}$	$\overline{E}$	SCL	NCL	ClrIn
010	$\overline{E}$	$\overline{W}$	N16	S16	SCL

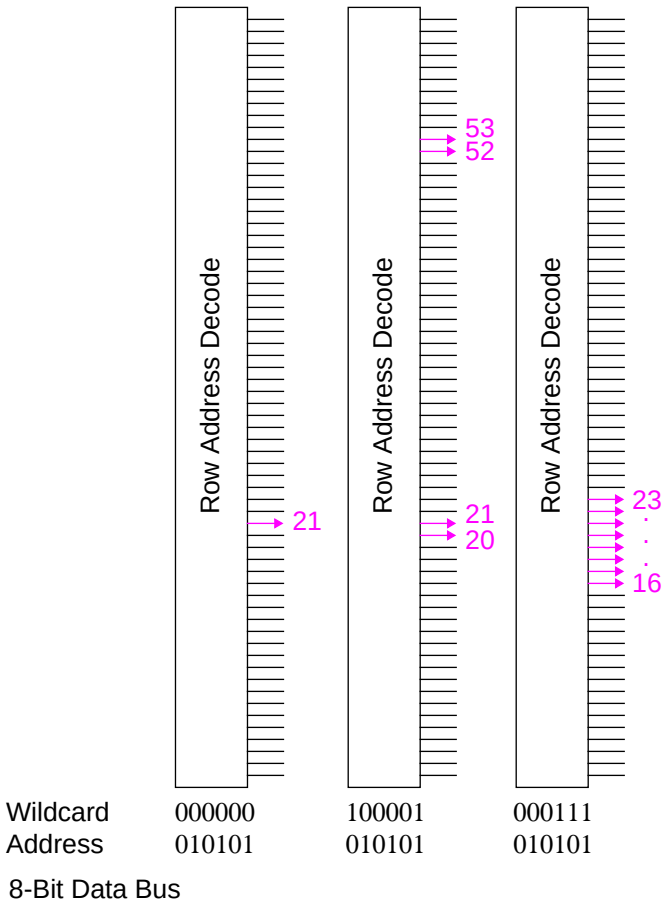


Figure 26. Row Wildcard Register

Sel[2:0]	N Mux	S Mux	N16 Mux	S16 Mux	Clr Mux
011	$\overline{W}$	$\overline{S}$	N4In	S4In	G2
100	$\overline{N4In}$	$\overline{S16}$	PS4	PN4	GClk
101	$\overline{N16}$	$\overline{PN4}$	NCOut	SCOut	0
110	$\overline{PS4}$	$\overline{S4In}$	SCIn	NCIn	S
111	$\overline{MN}$	$\overline{MS}$	MN	MS	G1

Sel[3:0]	N4 Mux	S4 Mux	Sel[3:0]	N4 Mux	S4 Mux
0000	SCIn	MN	1000	$\overline{ClkIn}$	$\overline{SCOut}$
0001	NCL	NCL	1001	$\overline{N4In}$	-
0010	N16	N16*	1010	-	-
0011	NCOut	S4In	1011	$\overline{MNA}$	-
0100	-	S16	1100	S16*	$\overline{NCIn}$
0101	-	MS	1101	SCL	-
0110	-	MSA	1110	MN	-
0111	-	SCL	1111	MS	-

* S4Out at South edge of 16x16 uses PN16. N4Out at North edge of 16x16 uses PS16.

Sel[1:0]	PrimaryClk	SecClk
00	SecClk	GClr
01	ClkIn	NCL
10	$\overline{NOut}$	G1
11	GClk	G2

Table 21: North/South Routing Configuration Coding

Address Mode 11 - Programming Control Registers

Gross features of the microprocessor interface are controlled by a number of Control Registers. We have already come across some of these in the form of the Map and Mask Registers. All the Programming Control Registers are mapped into the region of the device address space with the mode bits set to 11.

Wildcard Registers

The FastMAP™ contains additional hardware subsystems which can significantly reduce the processor overhead involved in configuring the device. These units are only active on write cycles. They are also inactive during writes to all Programming Control Registers.

The row address decoder is supplemented with a Wildcard Register which can be written through the FastMAP™. This register has one bit for each bit in the row address. During write cycles, logic one bits in the Wildcard Register indicate that the corresponding bit in the address is to be taken as ‘don’t-care’: that is the address decoder matches addresses independent of this bit. When the device is reset this register is initialized to zero so all address bits are treated as significant. Figure 26 shows some examples of the use of this unit assuming an 8-bit external data bus and a XC6216 device, so the cell array is eight words high. Outputs from the row address decoder enable bit line circuitry for the appropriate word.

The Wildcard Register allows many cell configuration memories within the same column of cells to be written simultaneously with the same data. This is used during device testing to allow regular patterns to be loaded efficiently into the control memory, but is more generally useful, especially with regular bit sliced designs, because it allows many cells to be changed simultaneously. For example, a 16-bit 2:1 multiplexer could be built using cell routing multiplexers and switched between sources using a single control store access.

Similarly, the column address decoder has a Wildcard Register which allows several cells on the same row to be written with the same configuration. The column address decoder drives the word lines to enable particular columns of RAM cells. In this case the number of columns which can be written simultaneously is internally limited to 32: that is, at most five don't care bits can be set. However, to guarantee that cells which are not being written do not overwrite each other, **there is a limit on the number of columns which should be written simultaneously**.

Part	Maximum Number Of Column Wildcards
XC6209	4
XC6216	4
XC6236	2
XC6264	2

Table 22: Maximum Number Of Column Wildcards

The row and column Wildcard Registers can be used simultaneously to rapidly configure regular structures onto the device.

The row Wildcard Register is ignored in state access mode, as the row decoding is controlled by the Map Register here. The column Wildcard Register may still be used to write to several banks of registers simultaneously.

The mask unit, described on page 19, simplifies changing areas of the control store within a single word unit: for example, changing the source of one multiplexer within a cell without affecting the others. Consider changing the source for a cell's North multiplexer without this unit: the following operations are required:

1. Read control store at appropriate address.
2. Mask out bits corresponding to North register.
3. Get new value for north register bits 6 and 7. Make sure other bits are 0.
4. OR new value with value from stage 2.
5. Write back.

Using the mask unit the following steps suffice (using 8-bit transfers):

1. Write Mask Register with binary 00111111.
2. Write new value to control store at appropriate address.

The mask and wildcard units can be used together to perform complex operations such as changing the source of the West multiplexer on every second cell at offsets between 0 and 15.

Device Configuration Register

This register controls global device functions and modes. The control functions of the bits are given in Table 23. 'Config Speed' sets the baud rate of the serial programming interface. 'Bus Width' allows selection of external data bus width between 8,16 and 32 bits. The '$\overline{\text{TTL}}$/CMOS' bit globally controls the input logic threshold levels for all the I/Os. The default value of '0' causes TTL input thresholds to be used. '1' selects CMOS thresholds. 'Clock Enable' allows the user to stop the GClk signal to the cell array. The clock to the control circuitry is not stopped so that CPU cycles, etc. may continue. The default state is Clock Enable = 0, disabling the array GClk. This bit may be modified at any time. Internal circuitry makes sure that the clock is started and stopped in a safe, glitch free manner. Changes to Clock Enable take effect immediately after the write cycle to the Configuration Register.

Bit:	7	6	5	4	3	2	1	0
Function:	-	Clock Enable	-	$\overline{\text{TTL}}$/CMOS	**Bus Width**		**Config Speed**	

Table 23: Device Configuration Register

Config. Reg [3:2]	Data Bus Width
00	**8**
01	**16**
10	**32**
11	**Illegal**

Table 24: Data Bus Width Coding

Config. Reg [1:0]	Config Speed
00	**GClk/16**
01	**GClk/8**
10	**GClk/4**
11	**GClk/2**

Table 25: Configuration Speed Coding

Device Identification Register

The ID register is a 16-byte store which must be written with the correct pattern before the device outputs can be enabled. This serves as a check that the programming interface is operating correctly before allowing potentially damaging outputs to be driven. Each byte must be written with the ASCII code for the letters shown in Table 26. The only exception is the last byte. This must be written with a number which distinguishes this part from others within the XC6000 group. It is not an ASCII coded number.

Details of the appropriate ID number for different family members are given in Table 28.

Until the ID register is correctly programmed, all the IO pad output drivers are disabled and the pull-up resistors are all enabled. Once a valid ID pattern has been written, the state of the IO pads depends on the $\overline{OE}$ signal. If $\overline{OE}$ is High then the output drivers remain disabled with their pull-ups on. When $\overline{OE}$ is Low individual outputs may be enabled and pulled up depending on their individual control signals.

The ID register also provides a means of simultaneously disabling all the device outputs under software control. If any valid ID byte is changed then the outputs are all disabled. The outputs are all re-enabled as soon as the ID byte is written with the correct value.

The internal *ConfigOK* signal is available to user designs to determine when a valid pattern is present.

Programming Control Register Memory Map

Table 26 lists the address for all the control registers for the XC6209 and XC6216. Each register may be written in 8-bit bytes. If 16 or 32 data bits are available then less writes are required. For example eight write cycles (to C010, C011,.....,C017) are required to change every bit of the Map Register with an 8-bit bus. Only two write cycles (to C010 and C014) are required with a 32-bit bus.

If only some bits are to be changed within a particular register then only the appropriate bytes need be written.

XC6236 and XC6264 have 18-bit address busses. The Control Register memory map for these parts is given in Table 27.

The full address decoding for the XC6200 family is summarized in Tables 29 to 32.

A[15:0]	Register	A[15:0]	Register
C000	Device Config	C031	ID (Byte1) (='i')
C004	Row Wildcard	C032	ID (Byte2) ('l')
C005	Column Wildcard	C033	ID (Byte3) ('i')
C008	Mask (Byte0)	C034	ID (Byte4) ('n')
C009	Mask (Byte1)	C035	ID (Byte5) ('x')
C00A	Mask (Byte2)	C036	ID (Byte6) (=' ')
C00B	Mask (Byte3)	C037	ID (Byte7) ('X')
C010	Map (Byte0)	C038	ID (Byte8) ('C')
C011	Map (Byte1)	C039	ID (Byte9) ('6')
C012	Map (Byte2)	C03A	ID (Byte10) ('0')
C013	Map (Byte3)	C03B	ID (Byte11) ('0')
C014	Map (Byte4)	C03C	ID (Byte12) ('0')
C015	Map (Byte5)	C03D	ID (Byte13) (=' ')
C016	Map (Byte6)*	C03E	ID (Byte14) (=' ')
C017	Map (Byte7)*	**C03F**	**ID (Byte15) (=ID #)**
C030	ID (Byte0) ('X')		

Table 26: Control Register Memory Map For XC6209 and XC6216

* Reserved locations in XC6209

A[17:0]	Register	A[17:0]	Register
30000	Device Config	3001D	Map (Byte13)*
30004	Row Wildcard	3001E	Map (Byte14)*
30005	Column Wildcard	3001F	Map (Byte15)*
30008	Mask (Byte0)	30030	ID (Byte0) (='X')
30009	Mask (Byte1)	30031	ID (Byte1) ('i')
3000A	Mask (Byte2)	30032	ID (Byte2) ('l')
3000B	Mask (Byte3)	30033	ID (Byte3) ('i')
30010	Map (Byte0)	30034	ID (Byte4) ('n')
30011	Map (Byte1)	30035	ID (Byte5) ('x')
30012	Map (Byte2)	30036	ID (Byte6) (=' ')
30013	Map (Byte3)	30037	ID (Byte7) ('X')
30014	Map (Byte4)	30038	ID (Byte8) ('C')
30015	Map (Byte5)	30039	ID (Byte9) ('6')
30016	Map (Byte6)	3003A	ID (Byte10) ('0')
30017	Map (Byte7)	3003B	ID (Byte11) ('0')
30018	Map (Byte8)	3003C	ID (Byte12) ('0')
30019	Map (Byte9)	3003D	ID (Byte13) (=' ')
3001A	Map (Byte10)	3003E	ID (Byte14) (=' ')
3001B	Map (Byte11)	**3003F**	**ID (Byte15) (=ID #)**
3001C	Map (Byte12)*		

Table 27: Control Register Memory Map For XC6236 and XC6264

* Reserved locations in XC6236

Device	ID Number
XC6209	**2**
XC6216	**1**
XC6236	**4**
XC6264	**3**

Table 28: XC6200 Family ID Numbers

Address Bus	Decode	
A[15:14] (Mode[1:0])	00	Cells
	01	East/West Switch or IOB
	10	North/South Switch or IOB
	11	Control Registers
A[13:8] (Column[5:0])	Cell Mode - Cell column	
	East/West Mode - Column[5:2] = 4x4 block number Column[1:0] decoded as:	
	00	West Switch
	01	West IOB (Column[5:2]=0000)
	10	East IOB (Column[5:2]=1011)
	11	East Switch
	North/South Mode - Switch column	
A[7:6] (Column Offset[1:0])	Cell Mode	
	00	Neighbor Routing
	01	Function Input Routing
	10	Function
	11	State Access (Cell Registers)
	East/West Switch Mode	
	00	E/W Switch or E/W IOB Reg 0
	01	E/W IOB Reg 1
	North/South Switch Mode	
	00	N/S Switch or N/S IOB Reg 0
	01	N/S Switch or N/S IOB Reg 1
	10	N/S Switch or N/S IOB Reg 2
	11	Secondary Clock Mux (Column[1:0] = 00 or 11) && (Row[1:0]=11)
A[5:0] (Row[5:0])	Cell Mode - Cell row	
	East/West Mode - Switch row	
	North/South Mode - Row[5:2] = 4x4 block number Row[1:0] decoded as:	
	00	South Switch
	01	South IOB (Row[5:2]=0000)
	10	North IOB (Row[5:2]=1011)
	11	North Switch

Table 29: XC6209 Memory Map

Address Bus	Decode	
A[15:14] (Mode[1:0])	00	Cells
	01	East/West Switch or IOB
	10	North/South Switch or IOB
	11	Control Registers
A[13:8] (Column[5:0])	Cell Mode - Cell column	
	East/West Mode - Column[5:2] = 4x4 block number Column[1:0] decoded as:	
	00	West Switch
	01	West IOB (Column[5:2]=0000)
	10	East IOB (Column[5:2]=1111)
	11	East Switch
	North/South Mode - Switch column	
A[7:6] (Column Offset[1:0])	Cell Mode	
	00	Neighbor Routing
	01	Function Input Routing
	10	Function
	11	State Access (Cell Registers)
	East/West Switch Mode	
	00	E/W Switch or E/W IOB Reg 0
	01	E/W IOB Reg 1
	North/South Switch Mode	
	00	N/S Switch or N/S IOB Reg 0
	01	N/S Switch or N/S IOB Reg 1
	10	N/S Switch or N/S IOB Reg 2
	11	Secondary Clock Mux (Column[1:0] = 00 or 11) && (Row[1:0]=11)
A[5:0] (Row[5:0])	Cell Mode - Cell row	
	East/West Mode - Switch row	
	North/South Mode - Row[5:2] = 4x4 block number Row[1:0] decoded as:	
	00	South Switch
	01	South IOB (Row[5:2]=0000)
	10	North IOB (Row[5:2]=1111)
	11	North Switch

Table 30: XC6216 Memory Map

Address Bus	Decode	
A[17:16] (Mode[1:0])	00	Cells
	01	East/West Switch or IOB
	10	North/South Switch or IOB
	11	Control Registers
A[15:9] (Column[6:0])	Cell Mode - Cell column	
	East/West Mode - Column[6:2] = 4x4 block number Column[1:0] decoded as:	
	00	West Switch
	01	West IOB (Column[6:2]=00000)
	10	East IOB (Column[6:2]=10111)
	11	East Switch
	North/South Mode - Switch column	
A[8:7] (Column Offset[1:0])	Cell Mode	
	00	Neighbor Routing
	01	Function Input Routing
	10	Function
	11	State Access (Cell Registers)
	East/West Switch Mode	
	00	E/W Switch or E/W IOB Reg 0
	01	E/W IOB Reg 1
	North/South Switch Mode	
	00	N/S Switch or N/S IOB Reg 0
	01	N/S Switch or N/S IOB Reg 1
	10	N/S Switch or N/S IOB Reg 2
	11	Secondary Clock Mux (Column[1:0] = 00 or 11) && (Row[1:0]=11)
A[6:0] (Row[6:0])	Cell Mode - Cell row	
	East/West Mode - Switch row	
	North/South Mode - Row[6:2] = 4x4 block number Row[1:0] decoded as:	
	00	South Switch
	01	South IOB (Row[6:2]=00000)
	10	North IOB (Row[6:2]=10111)
	11	North Switch

Table 31: XC6236 Memory Map

Address Bus	Decode	
A[17:16] (Mode[1:0])	00	Cells
	01	East/West Switch or IOB
	10	North/South Switch or IOB
	11	Control Registers
A[15:9] (Column[6:0])	Cell Mode - Cell column	
	East/West Mode - Column[6:2] = 4x4 block number Column[1:0] decoded as:	
	00	West Switch
	01	West IOB (Column[6:2]=00000)
	10	East IOB (Column[6:2]=11111)
	11	East Switch
	North/South Mode - Switch column	
A[8:7] (Column Offset[1:0])	Cell Mode	
	00	Neighbor Routing
	01	Function Input Routing
	10	Function
	11	State Access (Cell Registers)
	East/West Switch Mode	
	00	E/W Switch or E/W IOB Reg 0
	01	E/W IOB Reg 1
	North/South Switch Mode	
	00	N/S Switch or N/S IOB Reg 0
	01	N/S Switch or N/S IOB Reg 1
	10	N/S Switch or N/S IOB Reg 2
	11	Secondary Clock Mux (Column[1:0] = 00 or 11) && (Row[1:0]=11)
A[6:0] (Row[6:0])	Cell Mode - Cell row	
	East/West Mode - Switch row	
	North/South Mode - Row[6:2] = 4x4 block number Row[1:0] decoded as:	
	00	South Switch
	01	South IOB (Row[6:2]=00000)
	10	North IOB (Row[6:2]=11111)
	11	North Switch

Table 32: XC6264 Memory Map

Serial Programming Interface

All the memory mapped locations in XC6200 may be written in parallel or serial mode. All the operations which can be carried out with the FastMAP™ parallel interface may also be done serially. The serial interface gives random access to all the XC6200 memory locations. It is designed to operate with any Xilinx serial PROM. A single serial PROM may be used to configure several FPGAs. In this case one of the FPGAs acts as a 'Master' and the others as 'Slaves'. The Master controls the serial PROM and the Slaves. This is illustrated in Figure 27.

The serial PROM interface consists of 6 dedicated I/O pins:

$\overline{Serial}$	Input which controls transitions between states in serial mode state machine. 0 => serial mode, 1 => parallel mode
Wait	Input which controls transitions between states in serial mode state machine. 0 => continue loading, 1 => pause until *Wait* deasserted
SEReset	Output from Master FPGA which resets serial PROM address counter.
$\overline{SECE}$	Output from Master FPGA which enables serial PROM output.
SEClk	Output from Master FPGA which clocks serial PROM and slave FPGAs. *SEData* is clocked into the FPGAs on the rising edge of *SEClk*.
SEData	Serial data input to FPGA. This is sampled in the FPGA by *SEClk* and retimed by the FPGA's own GClk.

In a multi-FPGA configuration a user I/O also has to be available to provide the *Wait* input to the next device in the chain.

On Reset each FPGA examines its $\overline{Serial}$ and *Wait* inputs. Any FPGA which sees both these signals Low at this time assumes it is the master and drives *SEReset*, $\overline{SECE}$ and *SEClk*. All User I/Os are held in a high-impedance state (with pull-up) until a valid configuration is loaded. In the Figure 27 example the User I/O's are pulled High on Reset, hence the *Wait* input to the Slaves is High and they configure as Slaves. A valid configuration is assumed when the device ID register is loaded with the correct ID. Programmable I/Os can only be enabled when this is present.

Serial data is loaded in address/data pairs. Once an address/data pair has been shifted into the FPGA, the data word is parallel written to the corresponding address inside the FPGA, just as though a parallel CPU write had occurred. This means it is possible to do all the things which can be accomplished with the FastMAP™. e.g. use of the Mask Register, writes to cell state registers, etc.

The write operation is pipelined so there need be no interruption in the serial data stream. The first address/data pair must be preceded by a Synchronization Byte = 1111_1110. There are no start/stop bits, checksums or error check/correction bits. A full 8-bit Synchronization byte is not actually required by the FPGA. Three or more ones followed by a single zero is interpreted as a valid synchronization pattern.

The address and data are shifted in MSB first. The address is always 16-bits. The data word is initially 8-bits but may be increased to 16 or 32 bits by loading the Device Configuration Register with the appropriate code. The bits are shifted in on the rising edge of *SEClk*. The *SEClk* rate may also be increased by writing the appropri-

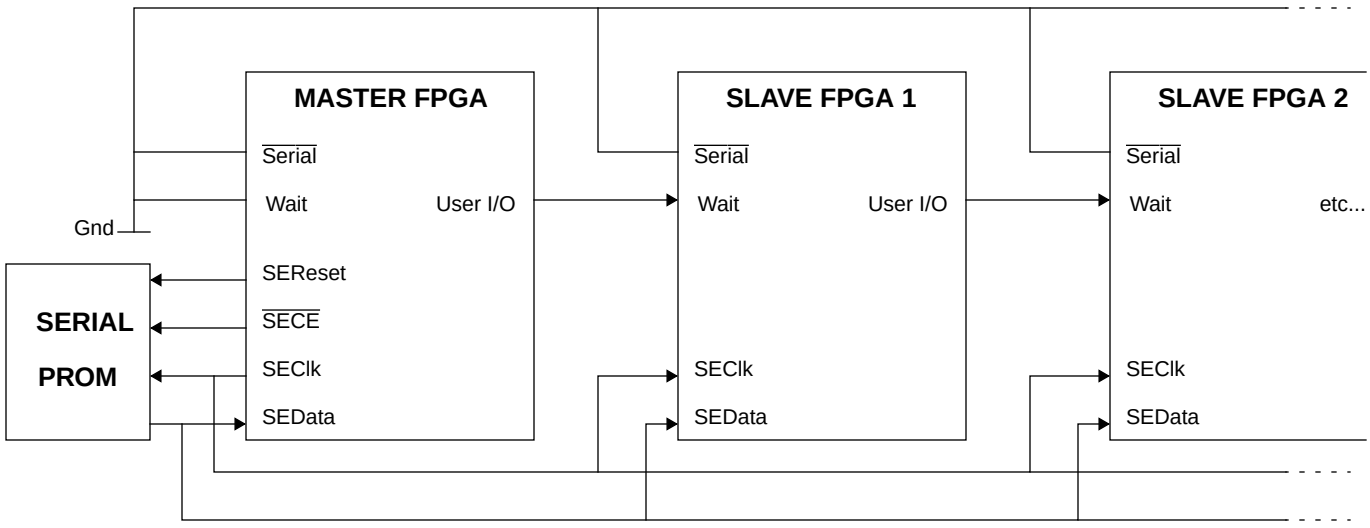


Figure 27. Master-Slave Serial Configuration

ate code to the Device Configuration Register. Initially *SECIk* is 1/16 GClk frequency. It can also be set to 1/8, 1/4 or 1/2 GClk.

An example is shown in Figure 28. Data1 is loaded into Addr1 after the address lsb has been shifted in. In this example the first write was to the Device Configuration Register and the data bus width was changed from 8 to 32 bits. Data word 2 starts immediately after Addr1 has been shifted in. Due to the new data bus width, 32 data bits are shifted in. If the width had not been changed data word 2 would also have been 8 bits. Data continues to be loaded until $\overline{Serial}$ goes High or *Wait* goes High.

Serial Interface State Machine

The serial interface is controlled by a state machine. This is a synchronous state machine with the state transitions occurring on rising edges of GClk. The transitions between states are mainly controlled by the $\overline{Serial}$ and *Wait* inputs. These signals are internally retimed, so it is the value sampled on the rising GClk edge **prior** to the edge causing the state transition which is used to determine the next state. A state diagram is shown in Figure 29. Transitions which show the state machine remaining in a state are not shown. The behavior is described below:

Reset

Master = False - i.e. initially this FPGA is not the Master.

Device remains in this state for T_{MRR} after $\overline{Reset}$ has been deasserted. The 4th rising GClk edge after T_{MRR} causes the transition to State 0. It is also this 4th clock edge which samples $\overline{Serial}$ and *Wait* to determine which state to enter after State 0.

Goto State 0

State 0 : Initial State - Determine if this is the Master FPGA.

$\overline{SECE}$ inactive, *SECIk* Low.

if ($\overline{Serial}$ = 1) then
    goto State 1 (Parallel)
else if (*Wait* = 1) then
    goto State 4 (Wait)
  else
    goto State 2 (Master Initialize)

State 1 : Parallel - Ready to load parallel data via CPU interface.

If Master, $\overline{SECE}$ inactive, *SECIk* Low.

If *Wait*=1 in this state, and the outputs are disabled (either because $\overline{OE}$=1 or the ID registers have not yet been written) , all the FPGA I/Os enter Leakage Test mode. This mode is for

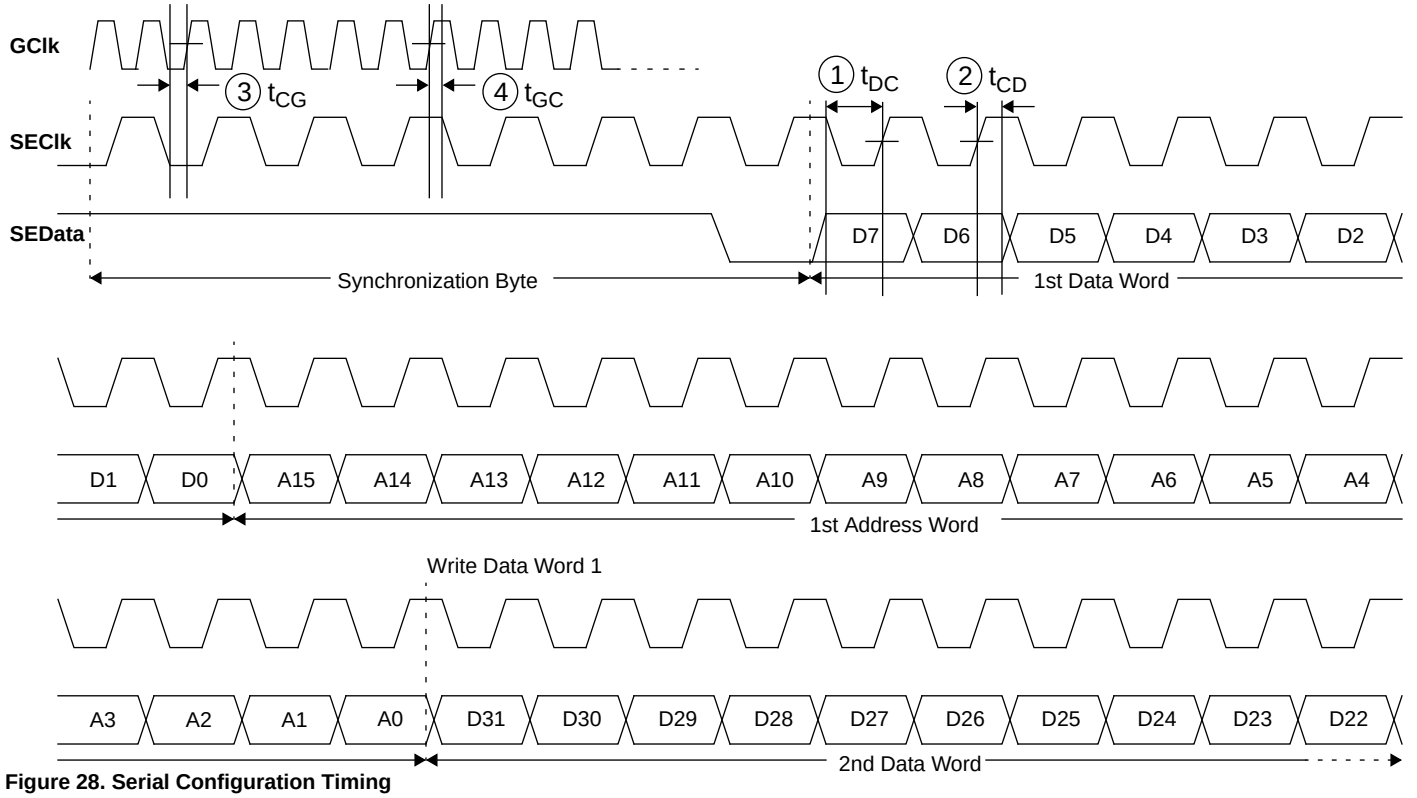


Figure 28. Serial Configuration Timing

factory testing and should not be used. All pull-up's and pull-down's are disabled in this mode.

if ($\overline{Serial}$ = 1) then
    goto State 1 (Remain in this state)
else if (*Wait* = 1) then
    goto State 4 (Wait)
  else if (Master) then
    goto State 5 ($\overline{SECE}$ Delay - $\overline{Serial}$ = 0 & *Wait* = 0)
  else
    goto State 6 (Sync. Wait - $\overline{Serial}$ = 0 & *Wait* = 0)

State 2 : Master Initialize - FPGA assumes it is the Master and resets Serial PROM.

$\overline{SECE}$ inactive, *SEClk* Low.

Master = True.

Assert SEReset.

Wait 16 GClk cycles (Note - this must be greater than the minimum reset time specified in the Xilinx databook Serial PROM section (T_{HOE})).

Deassert SEReset.

if (*Wait* = 1) then
    goto State 4 (Wait)
else
    goto State 5 ($\overline{SECE}$ Delay)

State 3 : Load Data - Shift in serial data and load.

If Master then $\overline{SECE}$ active, *SEClk* toggles.

if (Data_Loaded = 0) then (Data_Loaded is asserted at end of each addr/data pair)
    goto State 3 (Wait until complete addr/data pair loaded)
else if ($\overline{Serial}$ = 1) then
    goto State 1 (Parallel)
  else if ($\overline{Serial}$ = 0 & *Wait* = 1) then
    goto State 4 (Wait)

State 4 : Wait - Still in serial mode but idle.

If Master then $\overline{SECE}$ active, *SEClk* toggles.

if ($\overline{Serial}$ = 1) then
    goto State 1 (Parallel)
else if (*Wait* = 1) then
    goto State 4 (Remain in this state until Wait deasserted)
  else
    goto State 6 (Sync. Wait)

State 5 : $\overline{SECE}$ Delay - Assert $\overline{SECE}$ and wait for serial PROM to output data (Master only). *SEClk* Low.

Assert $\overline{SECE}$

Wait 9 GClk cycles (Note - this must be greater than the minimum $\overline{CE}$ to Data time specified in the Xilinx databook (T_{CE}))

goto State 6 (Sync. Wait)

State 6 : Sync. Wait - Wait for valid synchronization byte to be shifted in.

If Master then $\overline{SECE}$ active, *SEClk* toggles.

The FPGA actually looks for the pattern '1110' so it doesn't matter if the first few bits are missed.

if ($\overline{Serial}$ = 1) then (Can escape out of this state by returning to parallel mode)
    goto State 1 (Parallel)
else if (Valid_Sync_Pattern_In) then
    goto State 3 (Load Data)
  else
    goto State 6 (Wait for valid sync byte)

The Master toggles *SEClk* during States 3, 4 and 6, otherwise it is held Low. $\overline{SECE}$ is driven Low by the Master in States 3, 4, 5 and 6.

Typical Serial Load Sequence For Chain Of FPGAs

Central to the operation of serial loads is XC6200's ability to control the values seen on 'dedicated' I/O pins, such as *Wait*, by simple writes to IOB addresses. The following sequence is just one possible sequence. XC6200 allows the serial interface to write to any memory mapped RAM location, therefore it is completely flexible.

1) All FPGAs are reset and enter State 0.
2) FPGAs interrogate $\overline{Serial}$ and *Wait* inputs to see if they are the Master.
3) Master FPGA enters State 2, Slaves State 4.
4) Master resets serial PROM. Slaves wait in State 4.
5) Master enters State 5 and enables serial PROM.
6) Master enters State 6 and waits for a Sync pattern.
7) Master enters State 3 and starts loading data. The last data word to be loaded modifies the Master's

Wait IOB so that *Wait* goes High. This write also modifies the configuration to deassert the signal driving the *Wait* input of the next FPGA in the chain, causing the first Slave to enter State 6.

Note that if *SEClk* is set to GClk/2 rate then one more word is always loaded after the write which causes the transition from State 3 to 4. In this case the second last data word asserts the Master's *Wait* input and the last data word modifies the configuration to deassert the signal driving the *Wait* input of the next FPGA in the chain, causing the first Slave to enter State 6.

8) Master enters State 4 but still drives *SEClk* and $\overline{SECE}$. Slave waits for Sync pattern.

9) Slave receives Sync pattern, enters State 3 and starts loading data.

10) First Slave completes loading data. The last data word modifies the Slave's own $\overline{Serial}$ IOB so it enters State 1 after the next word. The last data word also causes the *Wait* input of the next Slave in the chain to be de-asserted. Again, if *SEClk*=GClk/2, this is performed with two separate writes.

11) First Slave enters State 1 and is ready to receive parallel CPU interface cycles. Next Slave enters

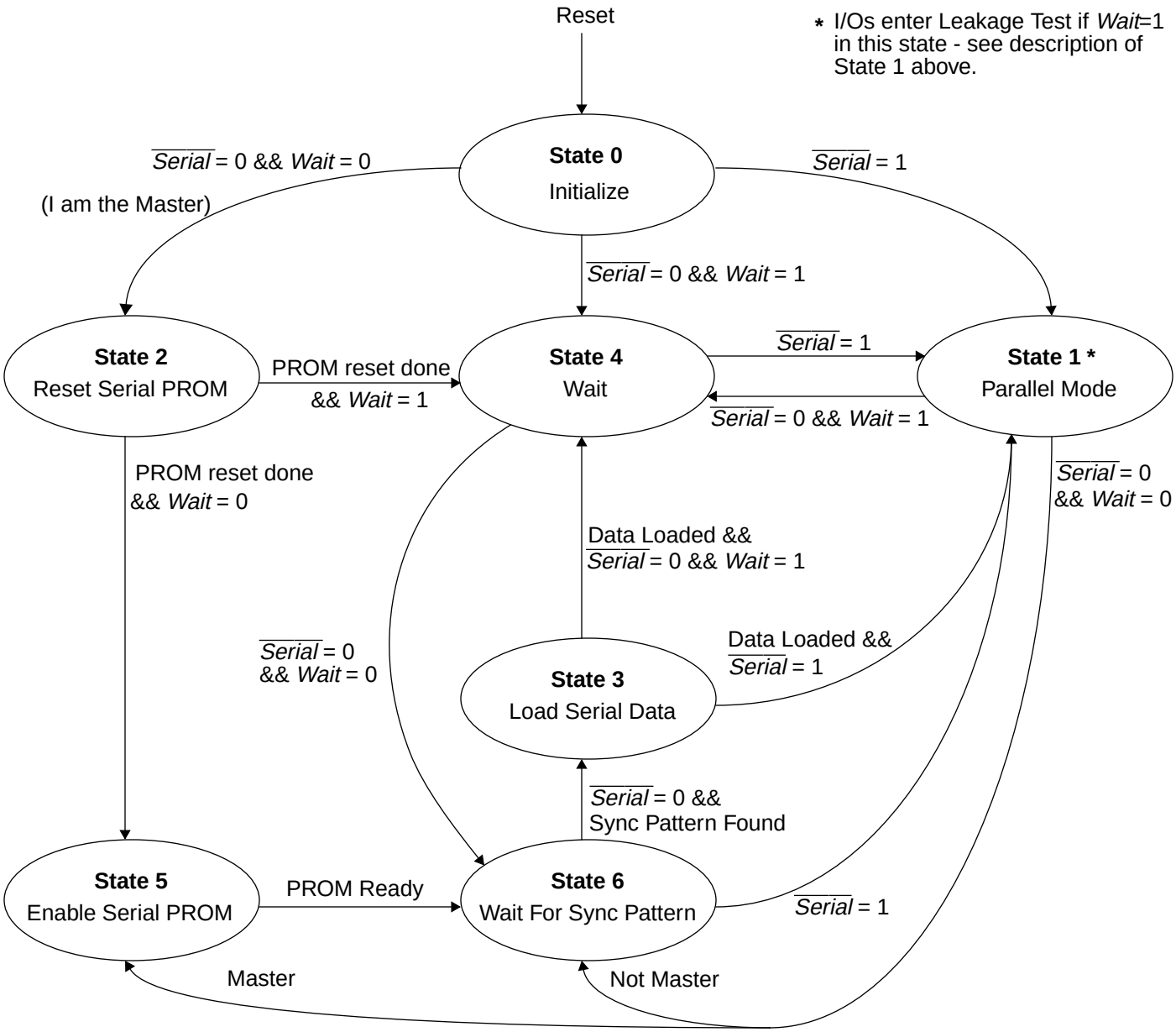


Figure 29. Serial Interface State Diagram

State 6 and waits for Sync pattern. The above sequence from step 8 is repeated until the last FPGA in the chain is reached.

12) The last FPGA could simply return to parallel mode (State 1) and leave the Master in State 4. If the Master is also to be returned to parallel mode, the last Slave must cause the Master's *Serial* input to go High. There are a number of ways of doing this. One way is for one of the last Slave's I/Os to be routed back to the Master and used to control the Master's *Serial* IOB. The last write to the last Slave would deassert this signal, causing the Master's *Serial* IOB to go High.

13) Master enters State 1, stops *SECIk* and deasserts *SECE*.

Reset And Initialization

When the XC6200 is powered up or after a reset, all configuration memory is cleared and the cell state registers are cleared. XC6200 does not require the *Reset* pin to be active during or after power-up to initialize itself. To avoid potential high current random configurations, the power-up reset is carried out automatically. The automatic power-up initialization takes T_{PRR} (see page 40). All the XC6200 I/O pads are disabled during this time and it is impossible to access the device. The XC6200 may be re-initialized at any time by asserting the *Reset* input for a minimum of T_{WMR}. This acts as a signal to the chip to initialize itself. This initialization occurs **after** *Reset* has been deasserted. Therefore there is a T_{MRR} reset recovery time when no device accesses are possible. The 5th rising GClk edge after this reset recovery period brings the state machine (on page 34) into State 1, 2 or 4, depending on whether Serial or Parallel Mode is selected. It is not possible to start parallel write cycles until State 1 is entered.

XC6200 devices initialize to the following configuration:

1) All control registers are cleared to 0. This disables all row and column wildcarding. GClk to the cell array is disabled. Inputs have TTL logic thresholds. The data bus width is 8 bits. The Map Register contains all 0s, therefore the eight data bus bits are applied to the entire column of state registers addressed. The serial configuration speed is GClk/16 (the slowest possible).

2) Cell function units are configured as registers with register protect off. Registers are cleared to ‘0’. The register clock inputs are held at a constant value to reduce power consumption. Thus the output of all function units is a constant 0.

3) All neighbor multiplexers select the function unit output so all cell neighbor outputs are constant.

4) Cell X1, X2 and X3 multiplexers select the South input. With the exception of the top row of cells, this is a cell neighbor output, and hence constant. In the top row of cells this signal comes from the North IOB and is connected to PN4In. This is the N4 output from the top cell in the array which is also a constant.

5) Cell magic outputs are sourced from X3 and hence are constant. Magic signals at the edge of the array are grounded.

6) Array N4 multiplexers select SC which is a cell output and hence constant. The N4 multiplexer in the South IOB selects NCL which is 0.

7) Array S4 multiplexers select MN which is a magic signal and hence constant. The S4 multiplexer in the North IOB selects SCL which is 0.

8) Array E4 multiplexers select MW which is a magic signal and hence constant. The E4 multiplexer in the West IOB selects ECL which is 0.

9) Array W4 multiplexers and the W4 multiplexer in the East IOB select WCL which is 0.

10) Chip-Length wires from the IOBs all select ‘0’.

11) N16 multiplexers in the array and the South IOB select NCL which is 0.

12) S16 multiplexers in the array and the North IOB select SCL which is 0.

13) E16 multiplexers in the array and the West IOB select ECL which is 0.

14) W16 multiplexers in the array and the East IOB select WCL which is 0.

15) Clear multiplexers in the array and North IOB select GClr. GClr is asserted by initialize logic independent of the GClr pin.

16) Clock multiplexers in the array and South IOBs select NCL = 0. This means the state register clocks are constant.

17) Enable, PUp and PDwn signals to the I/O pads are not acted upon until the device identification register has a valid pattern, so initial values are irrelevant.

18) All I/O pads function as inputs. Pull-up resistors on pads are active until the device identification register is correctly loaded so undriven control inputs are pulled High.

Packaging

This section contains pin identification for the XC6200 family. Devices are available in small and large packages. The small packages are useful where board area is at a premium and the design can make use of the wireless I/O FastMAP™ CPU interface to determine the state of internal nodes. The large package options give a very high user programmable I/O count where this is a prime requirement. The available packaging options for the XC6200 family are summarized in Tables 33 to 34. Please confirm availability with Xilinx.

Package	Pins	Signal Pins	Max Data Bus Pins	Unshared User I/O
PC	84	68	16	22
HT	144	120	16	72
BG	225	198	32	134
HQ	240	199	32	137
PG	299	242	32	180
HQ	304	242	32	180

Table 33: XC6216 Package Options

Package	Pins	Signal Pins	Max Data Bus Pins	Unshared User I/O
HQ	240	199	32	135
PG	299	242	32	178
HQ	304	242	32	178
PG	411	359	32	295
BG	432	360	32	296

Table 34: XC6264 Package Options

Signal pins are all the non-supply pins which drive into the array or control circuitry. Some of these pins are shared between control signals and user I/O. The un-shared user I/Os do not share a pin with a control signal. The number of user I/Os available is somewhere between the number of signal pins and the number of un-shared I/Os, depending on how many of the FPGA control signals are actually required. For example, if only an 8-bit data bus and no serial interface were required, the number of user I/Os would go up by 24+6=30 in a PGA299 package.

Precise pin-out information for all package types is given in the Device Pin-Outs section starting on page 48.

Pin Descriptions

The pins are labeled as follows:

V_{CC} - Supply

Connections to the nominal +5V supply. All must be connected.

GND - Supply

Connections to ground. All must be connected.

$\overline{CS}$ - Input

Chip Select enables the programming circuitry and initiates address decoding. When *$\overline{CS}$* is Low data can be read from or written to the control memory. This signal is intended to be used in conjunction with address decoding circuitry to select one part within a larger array for programming.

D<d:0> - Bidirectional

(d+1)-bit bidirectional data bus. Used for device configuration and direct cell register access.

A<a:0> - Input

Address bus for CPU access of internal registers and configuration memory. ‘a’ varies between family members.

Rd$\overline{Wr}$ - Input

When *$\overline{CS}$* is Low this signal determines whether data is read from or written to the control memory. If Rd*$\overline{Wr}$* is High then a read cycle takes place. If *Rd$\overline{Wr}$* is Low then a write cycle takes place.

GClk, GClr, G1, G2 - Inputs

Global signals. GClk should be used for global user clocks, GClr for global user clears and G1 and G2 for other global, low-skew signals. The GClk pin is **always** configured as an input and **cannot** be used as a fully flexible User I/O like the majority of other control signals.

All the usual IOB input functionality is available except it is not possible to route a user signal onto the GClk Control Signal using DfPB (Figure 15 and Table 18) as the GClk is routed to the built-in control logic directly from the pad. This differs from the GClk signal supplied to the user array, which comes from the GClk IOB and can have a user signal routed onto it in the usual way. See Figure 18 for details.

$\overline{Reset}$ - Input

When *$\overline{Reset}$* is driven Low the programming registers (mask unit and address wildcard unit) are re-initialized, resulting in the XC6200 appearing as a conventional SRAM. The control store of the cell array is initialized into a low power consumption configuration. All programmable output pad enable signals are forced inactive. All the IO-pad pull-up resistors are also enabled. As with GClk, this pin is **always** configured as an input and **cannot** be used as a fully flexible User I/O like the majority of other control signals.

$\overline{OE}$ - Input

When this signal is High the outputs of all programmable I/O pads are forced into a high impedance state (independent of the contents of the control store). All the IO-pad pull-up resistors are also enabled. As with GClk, this pin is **always** configured as an input and **cannot** be used as a fully flexible User I/O like the majority of other control signals.

$\overline{Serial}$ - Input

Input which controls transitions between states in serial state machine. *$\overline{Serial}$* **must** be Low initially to enter Serial mode or High to enter Parallel mode. After configuration, *$\overline{Serial}$* may be driven internally from the logic array to leave the IOB free for use as a general purpose I/O.

Wait - Input

Input which controls transitions between states in serial state machine. In Parallel mode *Wait* **must** be Low whilst initially configuring. This avoids putting the I/Os in Leakage Test mode (see "Serial Interface State Machine" on page 32). After configuration, *Wait* may be driven internally from the logic array to leave the IOB free for use as a general purpose I/O.

0 => continue loading, 1 => pause until Wait deasserted

SEReset - Output

Output from Master FPGA which resets serial PROM address counter.

$\overline{SECE}$ - Output

Output from Master FPGA which enables serial PROM output.

SEClk - Bidirectional

Output from Master FPGA which clocks serial PROM and slave FPGAs. On Slave FPGAs this is an input which is used to sample *SEData*.

SEData - Input

Serial data input to FPGA. This is sampled in the FPGA by *SEClk* and retimed by the FPGA's own GClk.

ConfigOK - Internal

Signal is active (High) when a valid pattern is present in the ID register and inactive when the pattern is invalid. ConfigOK cannot be directly routed onto the S31 pin as the Control Enable for S31 is permanently disabled. However it is available as 'DToPadB' in Table 19 and can be routed onto the N4 or N16 output from the S31 IOB.

ShiftDOut - Internal

MSB of input shift chain during serial loads. Control Enable for this signal is permanently disabled, so it can only be observed by routing through another IOB as with *ConfigOK*.

N_x - Bidirectional

North I/Os. Connections to I/O Blocks on the north of the array.

S_x - Bidirectional

South I/Os. Connections to I/O Blocks on the south of the array.

E_x - Bidirectional

East I/Os. Connections to I/O Blocks on the east of the array.

W_x - Bidirectional

West I/Os. Connections to I/O Blocks on the west of the array

---

Electrical Parameters

The XC6200 series is fabricated in triple-metal n-well CMOS process.

As with all CMOS devices, care must be exercised when handling, since inputs can be damaged by static discharge, although standard circuit design procedures have been used to minimize this risk.

Internal to the XC6200 devices, V_{CC} and ground are distributed on grids, which ensure minimal voltage drops between supply pads and internal circuitry. Power to the I/O cells is distributed in separate Vcc and ground rings.

All V_{CC} and GND pins must be connected to their respective PC-board plane. At least one low-inductance decoupling capacitor is recommended for each V_{CC} pin. A typical value is 0.1 uF. It must be placed very close to the V_{CC} lead, with a very short connection to the ground plane.

The power-supply current of an XC6200 device can vary between a few tens to several hundreds of milliamps, depending on the implemented design and the input activity. Power consumption is almost exclusively dynamic, and caused by the switching activity of the internal logic and the device outputs. For each internal node, power consumption is proportional to the switching frequency of that node.

The user must consider power consumption, package thermal characteristics, and ambient temperature to calculate the maximum chip (junction) temperature. Above $85^{o}C$, the internal delays increase by 0.35% per degree. See the Packages and Thermal Characteristics section of the Xilinx Data Book.

Simultaneous switching of many outputs can cause ground-bounce effects. The Product Technical Information section of the Xilinx Data Book discusses this.

Notice: The information contained in this data sheet pertains to products just entering production. These specifications are subject to change without notice. Verify with your local Xilinx sales office that you have the latest data sheet before finalizing a design. All the parameters given in the following tables are for the **XC6216 only**.

Absolute Maximum Ratings

Symbol	Parameter	Value	Units
V_{CC}	Supply voltage with respect to GND	-0.5 to 7.0	V
V_{IN}	DC Input voltage with respect to GND	-0.5 to V_{CC}+0.5	V
V_{TS}	Voltage applied to 3-state output with respect to GND	-0.5 to V_{CC}+0.5	V
T_{STG}	Storage temperature	-65 to +150	°C
T_{SOL}	Maximum soldering temperature (10s @ 1/16 in. = 1.5 mm)	+260	°C
I_{RINPU}	Pad pull-up (when selected) @V_{IN}=0V, V_{CC}=5V	TBA	mA
I_{RINPD}	Pad pull-down (when selected) @V_{IN}=5V, V_{CC}=5V	TBA	mA
		Preliminary	

Warning: *Stresses beyond those listed under Absolute Maximum Ratings may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those listed under Recommended Operating Conditions is not implied. Exposure to Absolute Maximum Ratings conditions for extended periods of time may affect device reliability.*

Recommended Operating Conditions

Symbol	Parameter		Min	Max	Units
V_{CC}	Supply voltage relative to GND	Commercial T_A = 0° C to 85° C junction	4.75	5.25	V
	Supply voltage relative to GND	Industrial T_A = -40° C to 100° C junction	4.50	5.50	V
V_{ILT}	Low-level input voltage	- TTL configuration	0	0.80	V
V_{IHT}	High-level input voltage	- TTL configuration	2.0	V_{CC}	V
V_{ILC}	Low-level input voltage	- CMOS configuration	0	20%	V_{CC}
V_{IHC}	High-level input voltage	- CMOS configuration	70%	100%	V_{CC}
T_{IN}	Input signal transition time		-	250	ns
			Preliminary		

DC Characteristics Over Recommended Operating Conditions

Symbol	Parameter	Test Conditions	Min	Max	Units
V_{OH}	High-level output voltage	I_{OH} = -8.0 mA V_{CC} = Min	3.86	-	V
V_{OL}	Low-level output voltage	I_{OL} = 8mA V_{CC} = Max	-	0.4	V
I_{IL}	Input leakage current	V_{CC} = Max V_{IN} = GND or V_{CC}	-1	1	µA
C_{IN}	Input capacitance for Input and I/O pins	V_{IN} = GND	-	15	pF
I_{CC}(2)	Quiescent Supply Current	V_{IN} = V_{CC} or GND V_{CC} = 5 V	-	12	mA
			Preliminary		

Notes: 1. Sample tested.
2. Measured with no output loads, no active input pull-up resistors and all package pins at V_{CC} or GND.

Power-up/Reset Timing Parameters

	Speed Grade:	**-2**			
Symbol	**Parameter**	**Min**	**Typ**	**Max**	**Units**
T_{WMR}	Master Reset input Low pulse width	5	-	-	ns
T_{MRR}	Recovery time after Master Reset deasserted	460	-	-	ns
T_{PRR}	Recovery time after power up(1)	-	-	1.5	µs
		Preliminary			

Notes: 1. Power up is defined as the time when V_{CC} has reached a stable level within the Min/Max limits specified in the "Recommended Operating Conditions" table.

AC Characteristics

Global Buffer Switching Characteristic Guidelines

Speed Grade:		-2		
Symbol	**Parameter**	**Min**	**Max**	**Units**
T_{PGClk}	From pad through **GClk** buffer to any register clock	-	10	ns
T_{PG}	From pad through **G1,G2,GClr** buffers to any register clock	-	10	ns
T_{PClr}	From pad through **global** buffers to any register clear	-	12	ns
T_{PCkS}	Skew between any pair of register clocks using the same **global**	-	0.9	ns
T_{PClS}	Skew between any pair of register clears using the same **global**	-	0.9	ns
		Preliminary		

Notes: 1. Typical loading values are used.

Guaranteed Input and Output Parameters (Pin-to-Pin)

All values listed below are tested directly and guaranteed over all the operating conditions. The same parameters can also be derived indirectly from the IOB and Global Buffer specifications. The delay calculator software uses this indirect method. When there is a discrepancy between these two methods, the directly tested values listed below should be used and the derived values should be ignored.

Speed Grade:		-2				
Symbol	**Parameter**	**Best I/O[5]**		**Worst I/O**		**Units**
		Min	**Max**	**Min**	**Max**	
T_{ICKOF}	Global Clock (GClk) to Output (fast)	-	16	-	19	ns
T_{ICKO}	Global Clock (GClk) to Output (slew limited)	-	17	-	20	ns
T_{PSUF}	Input Set-up Time (fast)	-	-	4	-	ns
T_{PSU}	Input Set-up Time with delay	-	-	10	-	ns
T_{PHF}	Input Hold Time (fast)	-	-	6	-	ns
T_{PH}	Input Hold Time with delay	-	-	0	-	ns
T_{OEHZ}	$\overline{OE}$ deasserted (High) to Pad begin hi-Z (slew rate independent)	-	TBA	-	TBA	ns
T_{OEONF}	$\overline{OE}$ asserted (Low) to Pad active and valid (fast)	-	TBA	-	TBA	ns
T_{OEONS}	$\overline{OE}$ asserted (Low) to Pad active and valid (slew rate limited)	-	TBA	-	TBA	ns
		Preliminary				

Notes: 1. All appropriate ac specifications tested using +/-8mA test load.
2. These parameters are tested directly and guaranteed over the operating conditions.
3. As the parameters vary between I/Os, values are given for best and worst I/Os. The parameters for other I/Os are somewhere between these two extremes. The delay calculator software calculates the correct value for each I/O used.
4. All parameters assume the cell register is the closest one to the IOB.
5. The difference between Best I/O and Worst I/O is caused by differing IOB to Pad delays.

IOB Switching Characteristic Guidelines

Speed Grade:		-2				
Symbol	**Parameter**	**Best I/O**		**Worst I/O**		**Units**
		Min	**Max**	**Min**	**Max**	**Units**
	INPUT					
T_{PID}	Pad to Neighbor data (NO-DELAY)	-	TBA	-	5	ns
T_{PID4}	Pad to L4 FastLANE™ (MID-DELAY)	-	TBA	-	6	ns
T_{PDID4}	Pad to L4 FastLANE™ with delay	-	TBA	-	14	ns
	OUTPUT					
T_{OPF}	Neighbor data to Output (fast)	-	TBA	-	6.5	ns
T_{OPS}	Neighbor data to Output (slew rate limited)	-	TBA	-	8	ns
T_{TSHZ}	3-state to Pad begin hi-Z (slew rate independent)	-	TBA	-	4	ns
T_{TSONF}	3-state to Pad active and valid (fast)	-	TBA	-	5	ns
T_{TSONS}	3-state to Pad active and valid (slew rate limited)	-	TBA	-	7	ns
		Preliminary				

Notes: 1. As the parameters vary between I/Os, values are given for best and worst I/Os. The parameters for other I/Os are somewhere between these two extremes. The delay calculator software calculates the correct value for each I/O used.
2. Typical loading values are used.

Cell Switching Characteristic Guidelines

Speed Grade:		-2		
Symbol	**Parameter**	**Min**	**Max**	**Units**
T_{ILO1}	X1 change to Function Output(1)	-	2	ns
T_{ILO23}	X2/X3 change to Function Output(2)	-	2.5	ns
T_{ICK1}	Internal Register Set-Up Time @ X1(1)	3	-	ns
T_{ICK23}	Internal Register Set-Up Time @ X2/X3(2)	3	-	ns
T_{IHCK1}	Internal Register Hold Time @ X1(1)	0	-	ns
T_{IHCK23}	Internal Register Hold Time @ X2/3(2)	0	-	ns
T_{CH}	Clock High Time(3)	2	-	ns
T_{CL}	Clock Low Time(3)	2	-	ns
T_{CLW}	Clear Pulse Width(3)	1	-	ns
T_{CKO}	Clock to Function Output	-	2.5	ns
T_{CKLO}	Clock to Function Output via X2/X3 feedback multiplexers	-	3.5	ns
		Preliminary		

Notes: 1. Data input measured at input to X1 routing multiplexer. Clock input measured at register.
2. Data input measured at input to X2/X3 routing multiplexers. Clock input measured at register.
3. Measured at the actual register in the cell.
4. Typical loading values are used.

Internal Routing Delays

Speed Grade:		-2		
Symbol	**Parameter**	**Min**	**Max**	**Units**
T_{NN}	Route Neighbor In to Neighbor Out	-	1	ns
T_{Magic}	Route X2/X3 to Magic Out	-	1.5	ns
T_{L4}	Length-4 FastLANE™ delay	-	1.5	ns
T_{L16}	Length-16 FastLANE™ delay	-	2	ns
T_{CL64}	Chip-Length (64) FastLANE™ delay	-	3	ns
		Preliminary		

Notes: 1. Delays vary depending on direction. Typical figures are given here. The delay calculator software calculates the correct delay for each direction.
2. Typical loading values are used.

CPU Interface Timing

Speed Grade:			-2		
Symbol		**Parameter**	**Min**	**Max**	**Units**
1	$T_{su\overline{CS}}$	$\overline{CS}$ set up before Clock(1)	8	-	ns
2	$T_{h\overline{CS}}$	$\overline{CS}$ hold after Clock(1)	0	-	ns
3	$T_{suRd\overline{Wr}}$	Rd$\overline{Wr}$ set up before Clock	3	-	ns
4	$T_{hRd\overline{Wr}}$	Rd$\overline{Wr}$ hold after Clock	0	-	ns
5	T_{suA}	Address Bus set up before Clock	2	-	ns
6	T_{hA}	Address Bus hold after Clock	0	-	ns
7	T_{suD}	Data Bus set up before Clock	3	-	ns
8	T_{hD}	Data Bus hold after Clock	0	-	ns
9	$T_{WC(ram)}$	Configuration SRAM Write cycle time(2)	30	-	ns
10	$T_{RC(ram)}$	Configuration SRAM Read cycle time(2)	40	-	ns
11	$T_{WC(reg)}$	State Write cycle time(2,3)	30	-	ns
12	$T_{RC(reg)}$	State Read cycle time(2,3)	56	-	ns
13	T_{CKD}	Clock to Valid Data	-	16	ns
14	T_{CKDZ}	Clock to Data high impedance(4)	-	18	ns
15	$T_{\overline{CS}DZ}$	$\overline{CS}$ to Data high impedance(4)	-	18.5	ns
16	T_{MPST}	Map Register Settling Time(5)	-	140	ns
			Preliminary		

Notes: 1. $\overline{CS}$ must be correctly sampled Low at the start of the cycle (t_1) and sampled High at the end of the cycle (t_2). Other signals only require to be correctly sampled at t_1.
2. The minimum time for a read or write cycle is two CPU clock periods, although the cycles shown do not start and finish at the start of a clock period. A 50% GClk duty cycle is assumed.
3. The cycle time for state accesses differs from the time for configuration SRAM accesses.
4. Data is removed from the bus T_{CKDZ} after t_3 unless $\overline{CS}$ is still asserted at this time. In this case, data is removed from the bus asynchronously $T_{\overline{CS}DZ}$ after $\overline{CS}$ goes High.
5. After a modification to any of the Map Register bits, it is not safe to perform register writes or reads for T_{MPST}. This is measured from the first falling clock edge of the Map Register Write Cycle to the **second** rising clock edge of the next register access (t_2 in Figures 30 and 31).

Serial Interface Timing

		Speed Grade:	-2		
Symbol		Description	Min	Max	Units
1	T_{DC}	SEData setup before SEClk rising	3	-	ns
2	T_{CD}	SEData hold after SEClk rising	0	-	ns
3	T_{CG}	SEClk setup before GClk rising(2)	5	-	ns
4	T_{GC}	SEClk hold after GClk rising(2)	0	-	ns
5	T_{CKSE}	GClk rising to SEClkOut(1)	12	-	ns
6	$T_{su\overline{Serial}}$	Serial setup before GClk rising	3	-	ns
7	$T_{h\overline{Serial}}$	Serial hold after GClk rising	0	-	ns
8	T_{suWait}	Wait setup before GClk rising	4	-	ns
9	T_{hWait}	Wait hold after GClk rising	0	-	ns
			Preliminary		

Notes: 1. Only for Master device.
2. Only for Slave device.

Configuration SRAM Access Timing Diagrams

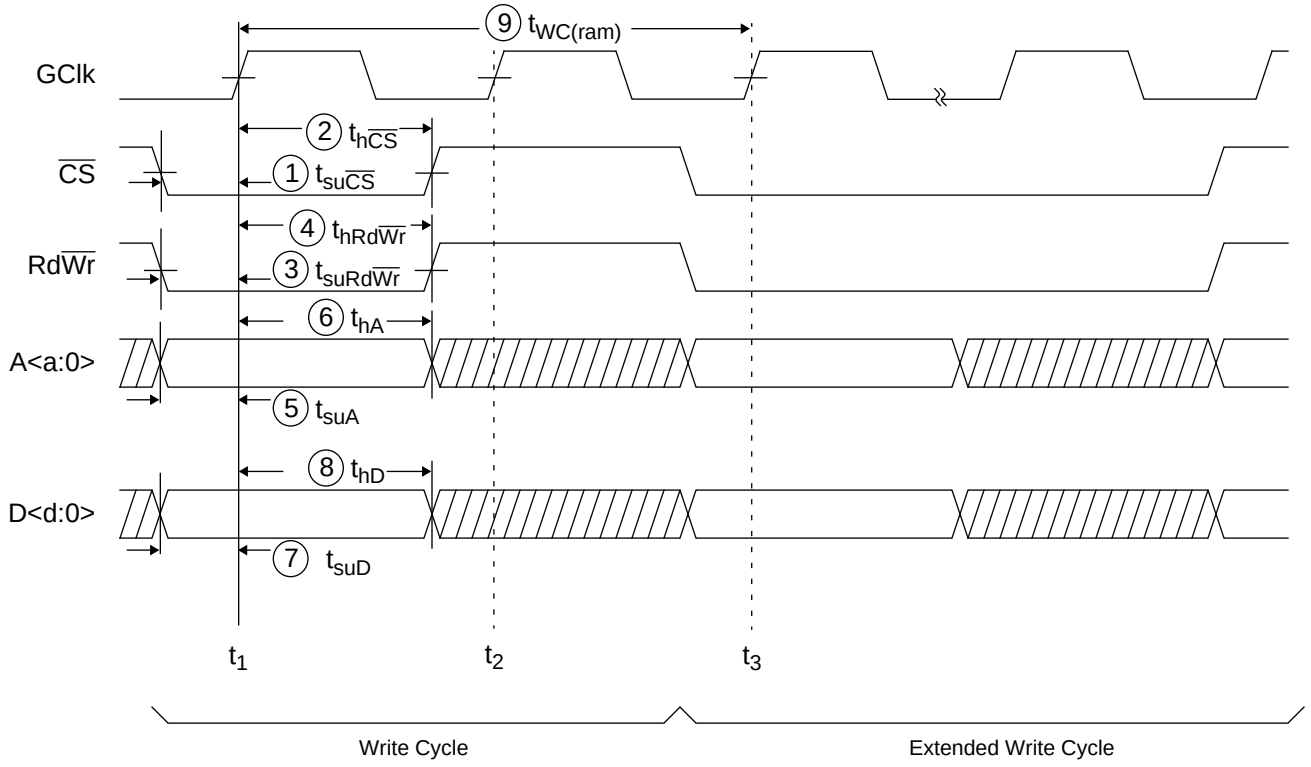


Figure 30. Configuration Memory Write Cycles

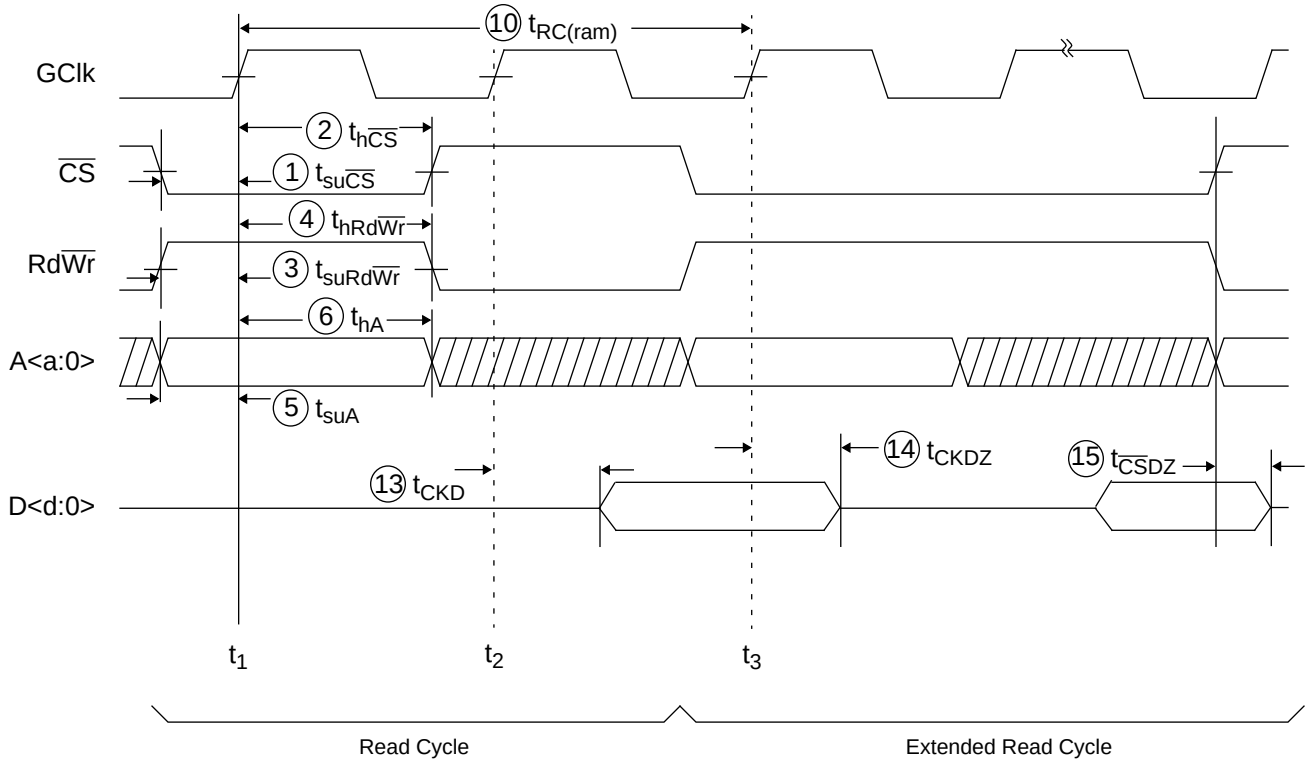


Figure 31. Configuration Memory Read Cycles

State Access Timing Diagrams

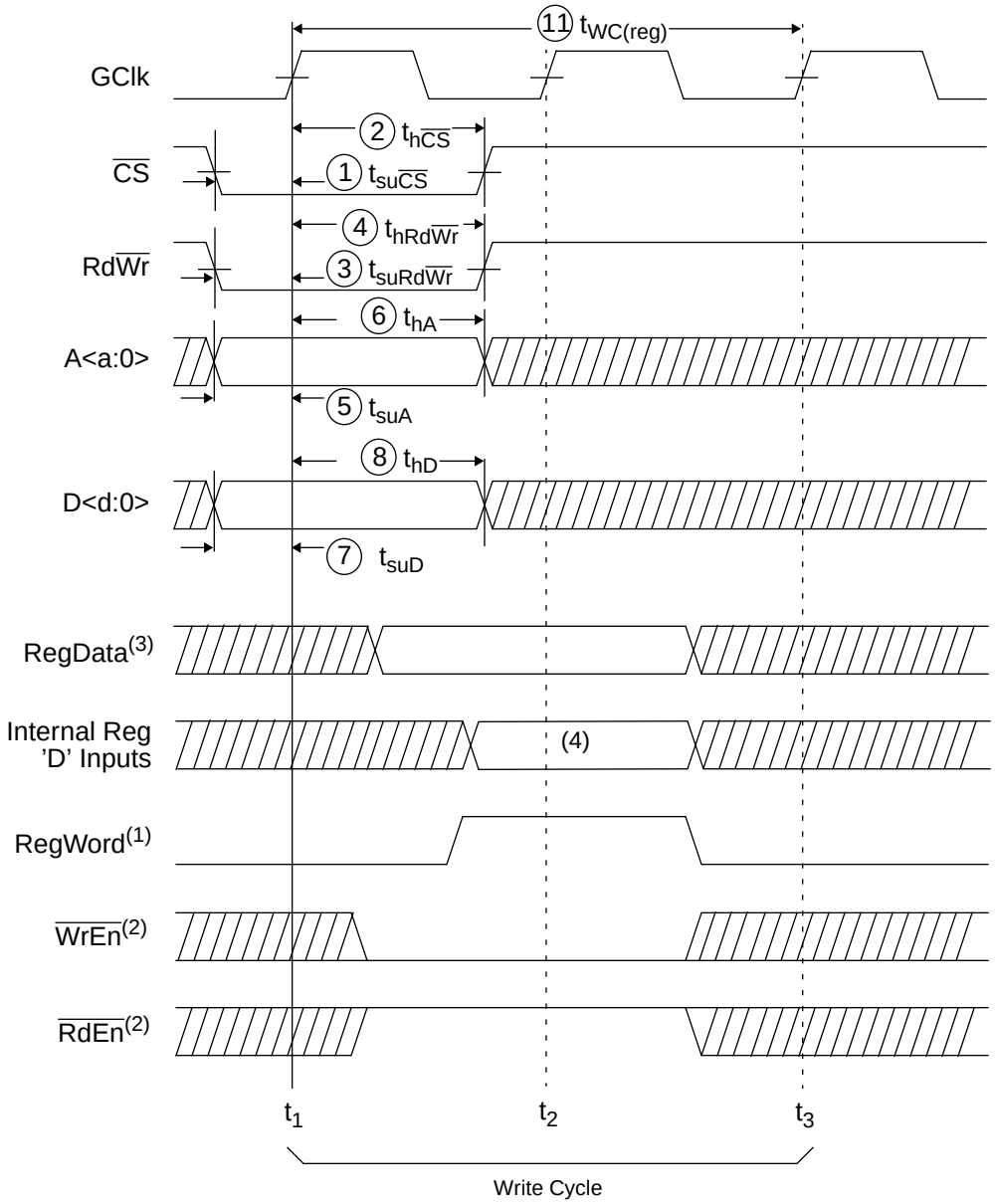


Figure 32. Register Write Cycle

Notes:

1. *RegWord* is an internal signal which can be accessed via the routing multiplexers in the North and South IOBs. See Table 19, "North/South IOB Configuration Coding," on page 24. Each column in the array has its own *RegWord* signal. It is activated during CPU reads and writes to any registers in the column. See the description following Table 19 for more details.
2. *WrEn* and *RdEn* are internal signals which can be accessed via the routing multiplexers in the East and West IOBs. See Table 14, "East/West IOB Configuration Coding," on page 22. Outside the Cycle they are normally Low but may change during accesses to other registers or configuration SRAM. See the description following Table 14 for more details.
3. *RegData* is an internal signal which can be accessed via routing multiplexers in the East IOBs. Outside the Cycle *RegData* is normally High but may change during accesses to other registers or configuration SRAM. See the description following Table 14 for more details.
4. The FastMAP™ places the CPU data to be written on the register inputs at this time. The designer must ensure the registers receive a rising clock edge around time t_2 to write the data. If GClk is routed to the register clock inputs then this timing is guaranteed. See "Timing" on page 17. for more details.

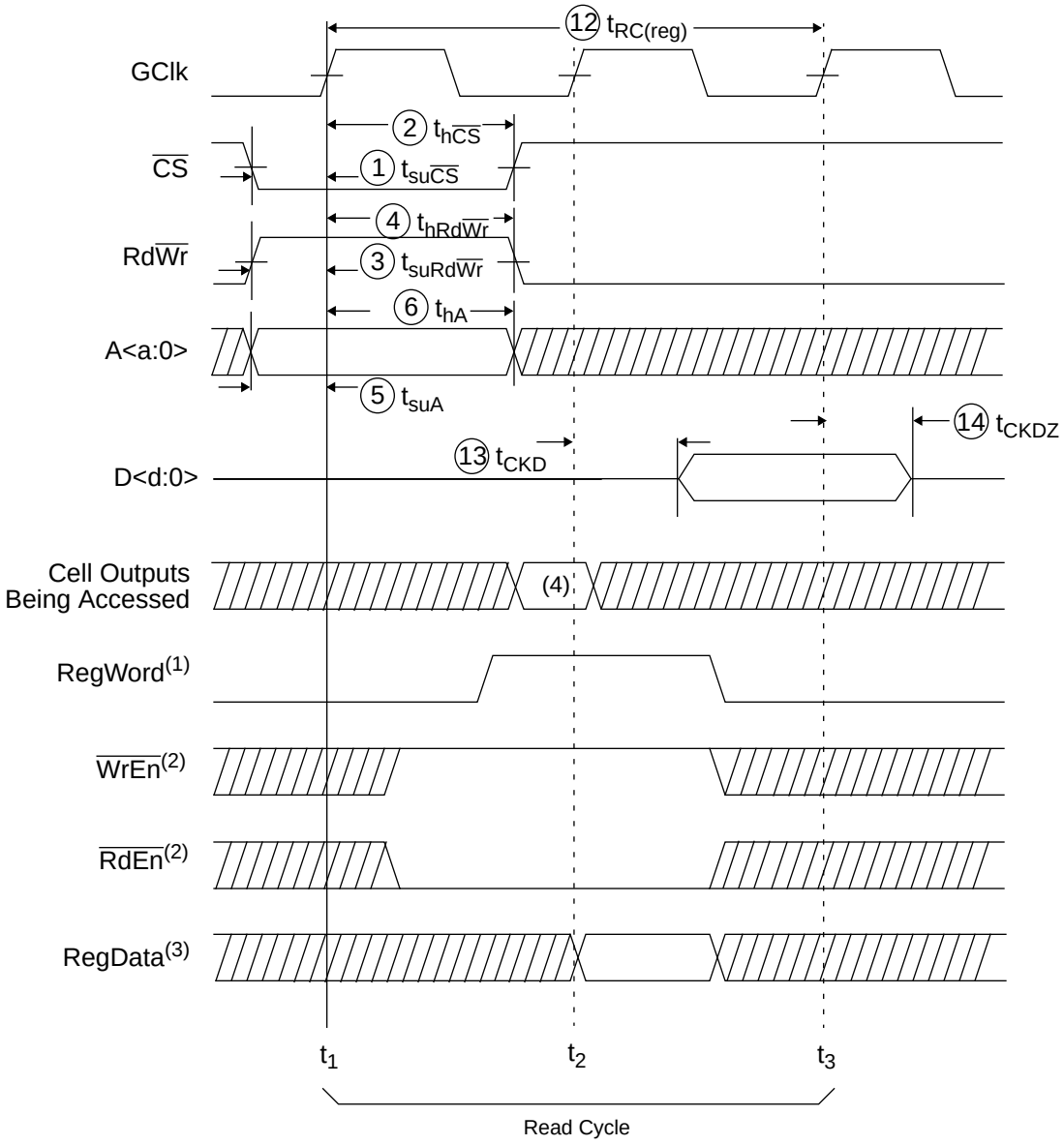


Figure 33. Register Read Cycle

Notes:

1. *RegWord* is an internal signal which can be accessed via the routing multiplexers in the North and South IOBs. See Table 19, “North/South IOB Configuration Coding,” on page 24. Each column in the array has its own *RegWord* signal. It is activated during CPU reads and writes to any registers in the column. See the description following Table 19 for more details.
2. *WrEn* and *RdEn* are internal signals which can be accessed via the routing multiplexers in the East and West IOBs. See Table 14, “East/West IOB Configuration Coding,” on page 22. Outside the Cycle they are normally Low but may change during accesses to other registers or configuration SRAM. See the description following Table 14 for more details.
3. *RegData* is an internal signal which can be accessed via routing multiplexers in the East IOBs. This signal can only be used during Read Cycles if a very slow GClk is being used. Outside the Cycle *RegData* is normally High but may change during accesses to other registers or configuration SRAM. See the description following Table 14 for more details.
4. Data is read from the cell outputs and sampled at time t_2. Therefore the cell output must be stable before and after t_2 to ensure a valid read occurs.

XC6216 Pinouts - West Side

Pin Description	PC84	HT144	BG225	HQ240	PG299	HQ304
$D0/W_1$ (2)	P32	P36	P2	P60	C18	P229
GND	**P31**	**P35**	**H10**	**P59**	**A19**	**P230**
W_{14}	-	P34	N3	P58	A20	P231
NC	-	-	P1 (1)	P57 (1)	C17 (1)	P232 (1)
$D1/W_3$	P30	P33	K6	P56	D16	P233
W_{16}	-	P32	N2	P55	E15	P234
$D2/W_5$	P29	P31	M3	P54	B18	P235
W_{18}	-	-	N1	P53	B17	P236
$D3/W_7$	P28	P30	L4	P52	C16	P237
NC	-	-	-	-	-	P254 (1)
NC	-	-	-	-	D15 (1)	P238 (1)
NC	-	-	-	-	A18 (1)	P239 (1)
W_{20}	-	-	M2	P51	E14	P240
$D4/W_9$	P27	P29	K5	P50	C15	P241
W_{22}	-	-	M1	P49	B16	P242
W_{24}	-	-	L3	P48	D14	P243
W_{26}	-	-	K4	P47	A17	P244
$D5/W_{11}$	P26	P28	L2	P46	C14	P245
NC	-	-	-	-	E13 (1)	P246 (1)
NC	-	-	-	-	B15 (1)	P247 (1)
GND	**-**	**P27**	**H9**	**P45**	**A15**	**P248**
W_{28}	-	P26	L1	P44	D13	P249
W_{30}	-	P25	J6	P43	B14	P250
W_{32}	-	-	K3	P42	C13	P251
$D6/W_{13}$	P25	P24	K2	P41	A14	P252
V_{CC}	**-**	**-**	**-**	**P40**	**A16**	**P253**
$D16/W_{33}$	-	P23	K1	P39	B13	P255
W_{34}	-	-	J5	P38	E12	P256
GND	**-**	**-**	**-**	**P37**	**E16**	**-**
NC	-	-	-	-	D12 (1)	P257 (1)
NC	-	-	-	-	C12 (1)	P258 (1)
NC	-	-	-	-	A13 (1)	P259 (1)
NC	-	-	-		B12 (1)	P260 (1)
$D7/W_{15}$	P24	P22	J4	P36	A12	P261
$D17/W_{35}$	-	-	J3	P35	D11	P262
W_{36}	-	-	J1	P34	E11	P263
$D18/W_{37}$	-	P21	J2	P33	C11	P264
$D8/W_{17}$	P23	P20	H5	P32	B11	P265
W_{38}	-	P19	H4	P31	B10	P266

Note: 1. Pin not connected.
2. Pins with a dual function have the 'Control' signal shown first. See section "I/O Architecture" on page 10 for details.

XC6216 Pinouts - West Side

Pin Description	PC84	HT144	BG225	HQ240	PG299	HQ304
V_{CC}	**P22**	**P18**	**H1**	**P30**	**A11**	**P267**
GND	**P21**	**P17**	**H2**	**P29**	**A10**	**P268**
D19/W_{39}	-	P16	H3	P28	C10	P269
W_{40}	-	-	-	-	D10	P270
D9/W_{19}	P20	P15	G5	P27	A9	P271
D20/W_{41}	-	P14	G1	P26	E10	P272
W_{42}	-	-	-	-	B9	P273
D21/W_{43}	-	-	G2	P25	C9	P274
D10/W_{21}	P19	P13	G3	P24	A8	P275
W_{44}	-	-	-	-	B8	P276
W_{46}	-	-	-	-	D9	P277
D22/W_{45}	-	-	G4	P23	A7	P278
GND	**-**	**-**	**-**	**P22**	**-**	**-**
W_{48}	-	-	F1	P21	E9	P279
W_{50}	-	-	F2	P20	C8	P280
V_{CC}	**-**	**-**	**-**	**P19**	**A6**	**P282**
D24/W_{49}	-	P12	F3	P18	B7	P283
D11/W_{23}	P18	P11	F4	P17	C7	P284
D23/W_{47}	-	P10	E1	P16	D8	P285
D25/W_{51}	-	P9	F5	P15	B6	P286
GND	**-**	**P8**	**G7**	**P14**	**A5**	**P287**
W_{52}	-	-	-	-	B5	P288
W_{54}	-	-	-	-	E8	P289
W_{56}	-	-	E2	P13	C6	P290
D12/W_{25}	P17	P7	E3	P12	D7	P291
W_{58}	-	-	D1	P11	A4	P292
D26/W_{53}	-	-	E4	P10	C5	P293
D27/W_{55}	-	-	G6	P9	B4	P294
D13/W_{27}	P16	P6	D2	P8	E7	P295
W_{60}	-	-	-	-	D6	P296
W_{62}	-	-	-	-	A3	P297
NC	-	-	-	-	-	P281(1)
D28/W_{57}	-	-	C1	P7	C4	P298
D14/W_{29}	P15	P5	D3	P6	D5	P299
D29/W_{59}	-	-	E5	P5	E6	P300
D15/W_{31}	P14	P4	C2	P4	B3	P301
D30/W_{61}	-	P3	B1	P3	B2	P302
D31/W_{63}	P13	P2	D4	P2	D4	P303
GND	**P12**	**P1**	**A1**	**P1**	**B1**	**P304**

Note: 1. Pin not connected.

XC6216 Pinouts - South Side

Pin Description	PC84	HT144	BG225	HQ240	PG299	HQ304
V_{CC}	**P33**	**P37**	**R1**	**P61**	**B20**	**P228**
Rd$\overline{Wr}$/S_1	P34	P38	M4	P62	D17	P227
$\overline{CS}$/S_3	P35	P39	R2	P63	B19	P226
W_{12}/S_0	-	P40	P3	P64	C19	P225
$\overline{OE}$/S_5	P36	P41	L5	P65	F16	P224
W_{10}/S_2	-	P42	N4	P66	E17	P223
$\overline{Reset}$/S_7	P37	P43	R3	P67	D18	P222
W_8/S_4	-	P44	P4	P68	C20	P221
NC	-	-	-	-	-	P205(1)
W_6/S_6	-	-	K7	P69	F17	P220
W_4/S_8	-	-	M5	P70	G16	P219
W_2/S_{10}	-	-	R4	P71	D19	P218
W_0/S_{12}	-	-	N5	P72	E18	P217
S_{14}	-	-	P5	P73	D20	P216
S_{16}	-	-	L6	P74	G17	P215
S_{18}	-	-	-	-	F18	P214
S_{20}	-	-	-	-	H16	P213
S_{22}	-	-	-	-	E19	P212
S_{24}	-	-	-	-	F19	P211
GND	**-**	**P45**	**K8**	**P75**	**E20**	**P210**
S_{26}	-	P46	R5	P76	H17	P209
S_{33}	-	P47	M6	P77	G18	P208
$\overline{Serial}$/S_9	P38	P48	N6	P78	G19	P207
Wait/S_{11}	P39	P49	P6	P79	H18	P206
V_{CC}	**-**	**-**	**-**	**P80**	**F20**	**P204**
S_{28}	-	-	R6	P81	J16	P203
S_{35}	-	-	M7	P82	G20	P202
GND	**-**	**-**	**-**	**P83**	**-**	**-**
S_{30}	-	-	-	-	J17	P201
S_{32}	-	-	-	-	H19	P200
S_{34}	-	-	-	-	H20	P199
S_{36}	-	-	-	-	J18	P198
GClk/S_{13}	P40	P50	N7	P84	J19	P197
S_{38}	-	-	P7	P85	K16	P196
GClr/S_{15}	P41	P51	R7	P86	J20	P195
S_{37}	-	-	L7	P87	K17	P194
S_{40}	-	P52	N8	P88	K18	P193
S_{39}	-	P53	P8	P89	K19	P192
V_{CC}	**P42**	**P54**	**R8**	**P90**	**L20**	**P191**

Note: 1. Pin not connected.

XC6216 Pinouts - South Side

Pin Description	PC84	HT144	BG225	HQ240	PG299	HQ304
GND	**P43**	**P55**	**M8**	**P91**	**K20**	**P190**
G1/S_{17}	P44	P56	L8	P92	L19	P189
G2/S_{19}	P45	P57	P9	P93	L18	P188
S_{42}	-	-	R9	P94	L16	P187
S_{41}	-	-	N9	P95	L17	P186
S_{44}	-	-	-	-	M20	P185
S_{43}	-	-	-	-	M19	P184
S_{46}	-	-	-	-	N20	P183
S_{48}	-	-	-	-	M18	P182
E_0/S_{50}	-	P58	M9	P96	M17	P181
E_2/S_{52}	-	P59	L9	P97	M16	P180
GND	**-**	**-**	**-**	**P98**	**-**	**-**
E_4/S_{54}	-	-	R10	P99	N19	P179
S_{45}	-	-	P10	P100	P20	P178
V_{CC}	**-**	**-**	**-**	**P101**	**T20**	**P177**
ShiftDOut/S_{21}	P46	P60	N10	P102	N18	P175
SEData/S_{23}	P47	P61	K9	P103	P19	P174
E_6/S_{56}	-	P62	R11	P104	N17	P173
S_{47}	-	P63	P11	P105	R19	P172
GND	**-**	**P64**	**J7**	**P106**	**R20**	**P171**
NC	-	-	-	-	N16(1)	P170(1)
NC	-	-	-	-	P18(1)	P169(1)
E_8/S_{58}	-	-	M10	P107	U20	P168
S_{49}	-	-	N11	P108	P17	P167
S_{51}	-	-	R12	P109	T19	P166
S_{53}	-	-	L10	P110	R18	P165
S_{55}	-	-	-	-	P16	P164
S_{57}	-	-	-	-	V20	P163
E_{10}/S_{60}	-	-	P12	P111	R17	P162
E_{12}/S_{62}	-	-	M11	P112	T18	P161
NC	-	-	-	-	-	P176(1)
$\overline{SECE}/S_{25}$	P48	P65	R13	P113	U19	P160
SEReset/S_{27}	P49	P66	N12	P114	V19	P159
S_{59}	-	P67	P13	P115	R16	P158
S_{61}	-	P68	K10	P116	T17	P157
SEClk/S_{29}	P50	P69	R14	P117	U18	P156
ConfigOK/S_{31}	P51	P70	N13	P118	X20	P155
GND	**P52**	**P71**	**H8**	**P119**	**W20**	**P154**
S_{63}	P53	P72	P14	P120	V18	P153

Note: 1. Pin not connected.

XC6216 Pinouts - East Side

Pin Description	PC84	HT144	BG225	HQ240	PG299	HQ304
V_{CC}	**P54**	**P73**	**R15**	**P121**	**X19,T16**	**P152**
E_{14}	-	P74	M12	P122	U17	P151
A0/E_1	P55	P75	P15	P123	W19	P150
E_{16}	-	P76	N14	P124	W18	P149
A1/E_3	P56	P77	L11	P125	T15	P148
E_{18}	-	-	M13	P126	U16	P147
A2/E_5	P57	P78	N15	P127	V17	P146
E_{20}	-	-	M14	P128	X18	P145
NC	-	-	-	-	U15(1)	P144(1)
NC	-	-	-	-	T14(1)	P143(1)
NC	-	-	-	-	-	P128(1)
A3/E_7	P58	P79	J10	P129	W17	P142
E_{22}	-	-	L12	P130	V16	P141
E_{24}	-	-	M15	P131	X17	P140
E_{26}	-	-	L13	P132	U14	P139
A4/E_9	P59	P80	L14	P133	V15	P138
E_{28}	-	-	K11	P134	T13	P137
NC	-	-	-	-	W16(1)	P136(1)
NC	-	-	-	-	W15(1)	P135(1)
GND	**-**	**P81**	**J8**	**P135**	**X16**	**P134**
E_{30}	-	-	L15	P136	U13	P133
E_{32}	-	-	K12	P137	V14	P132
A5/E_{11}	P60	P82	K13	P138	W14	P131
A16/E_{33}(2)	-	P83	K14	P139	V13	P130
V_{CC}	**-**	**-**	**-**	**P140**	**X15**	**P129**
E_{34}	-	P84	K15	P141	T12	P127
A17/E_{35}(2)	-	P85	J12	P142	X14	P126
GND	**-**	**-**	**J9**	**P143**	**-**	**-**
NC	-	-	-	-	U12(1)	P125(1)
NC	-	-	-	-	W13(1)	P124(1)
NC	-	-	-	-	X13(1)	P123(1)
NC	-	-	-	-	V12(1)	P122(1)
A6/E_{13}	P61	P86	J13	P144	W12	P121
E_{36}	-	-	J14	P145	T11	P120
E_{37}	-	-	J15	P146	X12	P119
E_{38}	-	P87	J11	P147	U11	P118
A7/E_{15}	P62	P88	H13	P148	V11	P117
E_{39}	-	P89	H14	P149	W11	P116
V_{CC}	**P63**	**P90**	**H15**	**P150**	**X10**	**P115**

Note: 1. Pin not connected.
2. A16/A17 only listed for designers wishing to be upwardly compatible with XC6264.

XC6216 Pinouts - East Side

Pin Description	PC84	HT144	BG225	HQ240	PG299	HQ304
GND	**P64**	**P91**	**H6**	**P151**	**X11**	**P114**
$A8/E_{17}$	P65	P92	H12	P152	W10	P113
E_{40}	-	P93	H11	P153	V10	P112
$A9/E_{19}$	P66	P94	G14	P154	T10	P111
E_{41}	-	P95	G15	P155	U10	P110
E_{42}	-	-	G13	P156	X9	P109
E_{43}	-	-	G12	P157	W9	P108
NC	-	-	-	-	X8(1)	P107(1)
E_{44}	-	-	-	-	V9	P106
E_{46}	-	-	-	-	U9	P105
E_{48}	-	-	-	-	T9	P104
GND	**-**	**-**	**-**	**P158**	**T5**	**-**
$A10/E_{21}$	P67	P96	G11	P159	W8	P103
E_{50}	-	P97	F15	P160	X7	P102
V_{CC}	**-**	**-**	**-**	**P161**	**X5**	**P101**
E_{45}	-	P98	F14	P162	V8	P99
E_{52}	-	-	F13	P163	W7	P98
$A11/E_{23}$	P68	P99	G10	P164	U8	P97
E_{47}	-	-	E15	P165	W6	P96
GND	**-**	**P100**	**H7**	**P166**	**X6**	**P95**
E_{54}	-	-	-	-	T8	P94
E_{56}	-	-	-	-	V7	P93
E_{58}	-	-	E14	P167	X4	P92
E_{49}	-	-	F12	P168	U7	P91
$A12/E_{25}$	P69	P101	E13	P169	W5	P90
E_{51}	-	-	D15	P170	V6	P89
E_{53}	-	-	F11	P171	T7	P88
E_{55}	-	-	D14	P172	X3	P87
NC	-	-	-	-	-	P100(1)
$A13/E_{27}$	P70	P102	E12	P173	U6	P86
E_{57}	-	-	C15	P174	V5	P85
E_{60}	-	-	-	-	W4	P84
E_{62}	-	-	-	-	W3	P83
$A14/E_{29}$	P71	P103	D13	P175	T6	P82
E_{59}	-	P104	C14	P176	U5	P81
$A15/E_{31}$	P72	P105	F10	P177	V4	P80
E_{61}	-	P106	B15	P178	X1	P79
E_{63}	P73	P107	C13	P179	V3	P78
V_{CC}	**P74**	**P108**	**B14**	**P180**	**W1**	**P77**

Note: 1. Pin not connected.

XC6216 Pinouts - North Side

Pin Description	PC84	HT144	BG225	HQ240	PG299	HQ304
V_{CC}	**P11**	**P144**	**B2**	**P240**	**A2,E5**	**P1**
N_1	P10	P143	C3	P239	C3	P2
N_3	P9	P142	A2	P238	D3	P3
N_0	-	P141	F6	P237	E4	P4
N_2	-	P140	B3	P236	F5	P5
N_4	-	-	C4	P235	C2	P6
N_6	-	-	A3	P234	D2	P7
N_8	-	-	-	-	E3	P8
N_{10}	-	-	-	-	F4	P9
NC	-	-	-	-	-	P11(1)
NC	-	-	-	-	-	P24(1)
N_5	P8	P139	D5	P233	C1	P10
N_7	P7	P138	B4	P232	G5	P12
N_{12}	-	-	E6	P231	F3	P13
N_{14}	-	-	A4	P230	E2	P14
N_{16}	-	-	C5	P229	G4	P15
N_{18}	-	-	D6	P228	D1	P16
N_{20}	-	-	-	-	G3	P17
N_{22}	-	-	-	-	H5	P18
GND	**-**	**P137**	**F8**	**P227**	**F1**	**P19**
N_{24}	-	P136	B5	P226	F2	P20
N_{26}	-	P135	A5	P225	H4	P21
N_{33}	-	-	F7	P224	G2	P22
N_{28}	-	-	C6	P223	H3	P23
V_{CC}	**-**	**-**	**-**	**P222**	**E1**	**P25**
N_{30}	-	-	-	-	G1	P26
N_{32}	-	-	-	-	H2	P27
N_{34}	-	-	-	-	J5	P28
N_{36}	-	-	-	-	J4	P29
N_{35}	P6	P134	B6	P221	J3	P30
N_9	P5	P133	A6	P220	H1	P31
GND	**-**	**-**	**G9**	**P219**	**-**	**-**
N_{11}	-	-	E7	P218	J2	P32
N_{38}	-	-	D7	P217	J1	P33
N_{37}	-	P132	C7	P216	K4	P34
N_{40}	-	P131	A7	P215	K5	P35
N_{39}	P4	P130	B7	P214	K3	P36
N_{13}	P3	P129	E8	P213	K2	P37
V_{CC}	**P2**	**P128**	**D8**	**P212**	**K1**	**P38**

Note: 1. Pin not connected.

XC6216 Pinouts - North Side

Pin Description	PC84	HT144	BG225	HQ240	PG299	HQ304
GND	**P1**	**P127**	**A8**	**P211**	**L1**	**P39**
N_{15}	P84	P126	B8	P210	L2	P40
N_{17}	P83	P125	C8	P209	L3	P41
N_{19}	-	P124	E9	P208	L4	P42
N_{42}	-	P123	A9	P207	M1	P43
N_{41}	-	-	B9	P206	L5	P44
N_{44}	-	-	C9	P205	M2	P45
GND	**-**	**-**	**-**	**P204**	**-**	**-**
N_{43}	P82	P122	D9	P203	M3	P46
N_{21}	P81	P121	A10	P202	N1	P47
N_{46}	-	-	-	-	N2	P48
N_{48}	-	-	-	-	M4	P49
N_{50}	-	-	-	-	P1	P50
N_{52}	-	-	-	-	M5	P51
V_{CC}	**-**	**-**	**-**	**P201**	**R1**	**P52**
N_{23}	-	-	B10	P200	N3	P54
N_{54}	-	-	C10	P199	P2	P55
N_{45}	-	P120	D10	P198	P3	P56
N_{56}	-	P119	A11	P197	N4	P57
GND	**-**	**P118**	**G8**	**P196**	**T1**	**P58**
NC	-	-	-	-	R2(1)	P59(1)
N_{47}	-	-	-	P195	T2	P60
N_{58}	-	-	E10	P194	N5	P61
N_{49}	-	-	B11	P193	R3	P62
N_{51}	-	-	C11	P192	P4	P63
N_{53}	-	P117	A12	P191	U1	P64
N_{55}	-	-	D11	P190	T3	P65
N_{57}	-	-	F9	P189	U2	P66
N_{60}	-	-	-	-	P5	P67
N_{62}	-	-	-	-	R4	P68
NC	-	-	-	-	-	P53(1)
N_{59}	P80	P116	B12	P188	V1	P69
N_{25}	P79	P115	A13	P187	U3	P70
N_{27}	-	P114	C12	P186	T4	P71
N_{61}	-	P113	E11	P185	R5	P72
N_{63}	P78	P112	B13	P184	V2	P73
N_{29}	P77	P111	A14	P183	W2	P74
GND	**P76**	**P110**	**D12**	**P182**	**X2**	**P75**
N_{31}	P75	P109	A15	P181	U4	P76

Note: 1. Pin not connected.

XC6264 Pinouts - West Side

Pin Description	HQ240	PG299	HQ304	PG411	BG432
$D0/W_1$ (1)	P60	C18	P229	E35	AH28
GND	**P59**	**A19**	**P230**	**GND***	**GND***
W_{14}	P58	A20	P231	A39	AH29
W_{64}	P57	C17	P232	B36	AJ30
$D1/W_3$	P56	D16	P233	H32	AH30
W_{16}	P55	E15	P234	G31	AG28
$D2/W_5$	P54	B18	P235	E33	AH31
W_{65}	-	-	-	B38	AG29
W_{66}	-	-	-	C35	AG30
W_{67}	-	-	-	-	-
GND	**-**	**-**	**-**	**GND***	**GND***
W_{18}	P53	B17	P236	F32	AF28
$D3/W_7$	P52	C16	P237	A35	AG31
W_{68}	-	-	-	B34	AF29
W_{69}	-	-	-	C33	AF30
W_{70}	-	D15	P238	A33	AE28
W_{71}	-	A18	P239	B32	AF31
V_{CC}	**-**	**-**	**-**	**V_{CC}***	**V_{CC}***
GND	**-**	**-**	**-**	**GND***	**GND***
W_{20}	P51	E14	P240	E31	AE29
$D4/W_9$	P50	C15	P241	C31	AE30
W_{22}	P49	B16	P242	F30	AD28
W_{24}	P48	D14	P243	F28	AD29
W_{72}	-	-	-	D32	AD30
W_{73}	-	-	-	D30	AD31
V_{CC}	**-**	**-**	**-**	**-**	**V_{CC}***
GND	**-**	**-**	**-**	**GND***	**GND***
W_{74}	-	-	-	-	-
W_{26}	P47	A17	P244	E29	AC28
$D5/W_{11}$	P46	C14	P245	B30	AC29
W_{75}	-	E13	P246	D28	AC30
W_{76}	-	B15	P247	F26	AB28
GND	**P45**	**A15**	**P248**	**GND***	**GND***
W_{28}	P44	D13	P249	C29	AB29
W_{30}	P43	B14	P250	B28	AB30
W_{32}	P42	C13	P251	E27	AB31
$D6/W_{13}$	P41	A14	P252	E25	AA29
V_{CC}	**P40**	**A16**	**P253**	**V_{CC}***	**V_{CC}***
W_{77}	-	-	P254	-	-

Note: 1. Pins with a dual function have the 'Control' signal shown first. See section "I/O Architecture" on page 10 for details.

XC6264 Pinouts - West Side

Pin Description	HQ240	PG299	HQ304	PG411	BG432
W_{78}	-	-	-	-	-
D16/W_{33}	P39	B13	P255	F24	AA30
W_{34}	P38	E12	P256	C27	Y28
GND	**-**	**-**	**-**	**GND***	**GND***
W_{79}	-	-	-	A27	Y29
W_{80}	-	-	-	B26	Y30
W_{81}	-	-	-	C25	Y31
W_{82}	-	-	-	E32	W28
W_{83}	-	-	-	-	-
W_{84}	-	-	-	-	-
W_{85}	-	D12	P257	D24	W29
W_{86}	-	C12	P258	A25	W30
GND	**P37**	**E16**	**-**	**GND***	**GND***
V_{CC}	**-**	**-**	**-**	**V_{CC}***	**V_{CC}***
W_{87}	-	-	-	-	-
W_{88}	-	-	-	-	-
W_{89}	-	-	-	F22	W31
W_{90}	-	-	-	C23	V28
W_{91}	-	A13	P259	B24	V29
W_{92}	-	B12	P260	A23	V30
D7/W_{15}	P36	A12	P261	D22	U29
D17/W_{35}	P35	D11	P262	E21	U28
GND	**-**	**-**	**-**	**GND***	**GND***
W_{36}	P34	E11	P263	B22	U30
D18/W_{37}	P33	C11	P264	C21	U31
W_{93}	-	-	-	-	-
W_{94}	-	-	-	-	-
D8/W_{17}	P32	B11	P265	B20	T29
W_{38}	P31	B10	P266	F20	T30
V_{CC}	**P30**	**A11**	**P267**	**V_{CC}***	**V_{CC}***
GND	**P29**	**A10**	**P268**	**GND***	**GND***
D19/W_{39}	P28	C10	P269	C19	T31
W_{40}	-	D10	P270	C17	R29
W_{95}	-	-	-	-	-
W_{96}	-	-	-	-	-
D9/W_{19}	P27	A9	P271	B18	R28
D20/W_{41}	P26	E10	P272	E19	R30
GND	**-**	**-**	**-**	**GND***	**GND***
W_{42}	-	B9	P273	A17	R31
D21/W_{43}	P25	C9	P274	D18	P29

XC6264 Pinouts - West Side

Pin Description	HQ240	PG299	HQ304	PG411	BG432
D10/W_{21}	P24	A8	P275	D16	P28
W_{44}	-	B8	P276	B16	P30
W_{97}	-	-	-	C15	N30
W_{98}	-	-	-	F18	N29
V_{CC}	**-**	**-**	**-**	**V_{CC}***	**V_{CC}***
GND	**-**	**-**	**-**	**GND***	**GND***
W_{46}	-	D9	P277	A15	N28
D22/W_{45}	P23	A7	P278	E13	N31
W_{99}	-	-	-	E17	M31
W_{100}	-	-	-	B14	M29
W_{101}	-	-	-	-	-
W_{103}	-	-	-	D12	M28
W_{105}	-	-	-	A13	M30
GND	**P22**	**-**	**-**	**GND***	**GND***
W_{48}	P21	E9	P279	C13	L30
W_{50}	P20	C8	P280	F16	L29
NC	-	-	P281(1)	-	-
V_{CC}	**P19**	**A6**	**P282**	**V_{CC}***	**V_{CC}***
D24/W_{49}	P18	B7	P283	E15	K31
D11/W_{23}	P17	C7	P284	E11	K30
D23/W_{47}	P16	D8	P285	B12	K29
D25/W_{51}	P15	B6	P286	C11	K28
GND	**P14**	**A5**	**P287**	**GND***	**GND***
W_{52}	-	B5	P288	F14	J30
W_{54}	-	E8	P289	F10	H31
W_{56}	P13	C6	P290	B10	J29
D12/W_{25}	P12	D7	P291	D10	J28
W_{107}	-	-	-	-	-
GND	**-**	**-**	**-**	**GND***	**GND***
V_{CC}	**-**	**-**	**-**	**-**	**V_{CC}***
W_{58}	P11	A4	P292	F12	H30
D26/W_{53}	P10	C5	P293	E9	G30
D27/W_{55}	P9	B4	P294	C9	H29
D13/W_{27}	P8	E7	P295	B8	H28
W_{109}	-	-	-	D8	F31
W_{111}	-	-	-	C7	F30
GND	**-**	**-**	**-**	**GND***	**GND***
V_{CC}	**-**	**-**	**-**	**V_{CC}***	**V_{CC}***
W_{60}	-	D6	P296	A5	G29
W_{62}	-	A3	P297	A7	G28

Note: 1. Pin not connected.

XC6264 Pinouts - West Side

Pin Description	HQ240	PG299	HQ304	PG411	BG432
W_{113}	-	-	-	C5	E31
W_{115}	-	-	-	F8	E30
W_{117}	-	-	-	B6	F29
W_{119}	-	-	-	E7	F28
W_{121}	-	-	-	-	-
W_{123}	-	-	-	-	-
GND	**-**	**-**	**-**	**GND***	**GND***
W_{125}	-	-	-	-	-
W_{127}	-	-	-	-	-
$D28/W_{57}$	P7	C4	P298	G9	D31
$D14/W_{29}$	P6	D5	P299	B2	D30
$D29/W_{59}$	P5	E6	P300	D4	E29
$D15/W_{31}$	P4	B3	P301	B4	E28
$D30/W_{61}$	P3	B2	P302	F6	C30
$D31/W_{63}$	P2	D4	P303	H8	D29
GND	**P1**	**B1**	**P304**	**GND***	**GND***

XC6264 Pinouts - South Side

Pin Description	HQ240	PG299	HQ304	PG411	BG432
V_{CC}	**P61**	**B20**	**P228**	**V_{CC}***	**V_{CC}***
$Rd\overline{Wr}/S_1$	P62	D17	P227	G33	AJ28
$\overline{CS}/S_3$	P63	B19	P226	D36	AK29
W_{12}/S_0	P64	C19	P225	C37	AH27
$\overline{OE}/S_5$	P65	F16	P224	F34	AK28
W_{10}/S_2	P66	E17	P223	J33	AJ27
$\overline{Reset}/S_7$	P67	D18	P222	D38	AL28
W_8/S_4	P68	C20	P221	G35	AH26
S_{64}	-	-	-	-	-
S_{65}	-	-	-	-	-
GND	**-**	**-**	**-**	**GND***	**GND***
S_{66}	-	-	-	-	-
S_{67}	-	-	-	E37	AK27
S_{68}	-	-	-	H34	AJ26
S_{69}	-	-	-	E39	AL27
S_{70}	-	-	-	K34	AH25
W_6/S_6	P69	F17	P220	F38	AK26
W_4/S_8	P70	G16	P219	G37	AL26
V_{CC}	**-**	**-**	**-**	**V_{CC}***	**V_{CC}***
GND	**-**	**-**	**-**	**GND***	**GND***
W_2/S_{10}	P71	D19	P218	H38	AH24
W_0/S_{12}	P72	E18	P217	J37	AJ25
S_{14}	P73	D20	P216	G39	AK25
S_{16}	P74	G17	P215	M34	AJ24
S_{71}	-	-	-	K36	AH23
S_{72}	-	-	-	K38	AK24
S_{73}	-	-	-	-	-
S_{74}	-	-	-	-	-
V_{CC}	**-**	**-**	**-**	**-**	**V_{CC}***
GND	**-**	**-**	**-**	**GND**	**GND***
S_{75}	-	-	-	-	-
S_{76}	-	-	-	-	-
S_{18}	-	F18	P214	N35	AL24
S_{20}	-	H16	P213	P34	AH22
S_{22}	-	E19	P212	J35	AJ23
S_{24}	-	F19	P211	L37	AK23
GND	**P75**	**E20**	**P210**	**GND***	**GND***
S_{26}	P76	H17	P209	M38	AJ22
S_{33}	P77	G18	P208	R35	AK22
$\overline{Serial}/S_9$	P78	G19	P207	H36	AL22

XC6264 Pinouts - South Side

Pin Description	HQ240	PG299	HQ304	PG411	BG432
Wait/S_{11}	P79	H18	P206	T34	AJ21
NC	-	-	P205(1)	-	-
V_{CC}	**P80**	**F20**	**P204**	**V_{CC}***	**V_{CC}***
S_{28}	P81	J16	P203	N37	AH20
S_{35}	P82	G20	P202	N39	AK21
S_{77}	-	-	-	-	-
GND	**P83**	**-**	**-**	**GND***	**GND***
S_{78}	-	-	-	P38	AJ20
S_{79}	-	-	-	L35	AH19
S_{80}	-	-	-	U35	AK20
S_{81}	-	-	-	R39	AJ19
S_{30}	-	J17	P201	M36	AL20
S_{32}	-	H19	P200	V34	AH18
GND	**-**	**-**	**-**	**GND***	**GND***
V_{CC}	**-**	**-**	**-**	**V_{CC}***	**V_{CC}***
S_{34}	-	H20	P199	R37	AK19
S_{36}	-	J18	P198	T38	AJ18
GClk/S_{13}	P84	J19	P197	T36	AL19
S_{38}	P85	K16	P196	V36	AK18
GClr/S_{15}	P86	J20	P195	U37	AH17
S_{37}	P87	K17	P194	U39	AJ17
S_{82}	-	-	-	-	-
S_{83}	-	-	-	-	-
GND	**-**	**-**	**-**	**GND***	**GND***
S_{84}	-	-	-	-	-
S_{85}	-	-	-	-	-
S_{86}	-	-	-	-	-
S_{87}	-	-	-	W35	AK17
S_{88}	-	-	-	AC39	AL17
S_{40}	P88	K18	P193	V38	AJ16
S_{39}	P89	K19	P192	W37	AK16
V_{CC}	**P90**	**L20**	**P191**	**V_{CC}***	**V_{CC}***
GND	**P91**	**K20**	**P190**	**GND***	**GND***
G1/S_{17}	P92	L19	P189	Y34	AL16
G2/S_{19}	P93	L18	P188	AC37	AH15
S_{89}	-	-	-	Y38	AL15
S_{90}	-	-	-	AA37	AJ15
S_{91}	-	-	-	-	-
S_{92}	-	-	-	-	-
GND	**-**	**-**	**-**	**GND***	**GND***

Note: 1. Pin not connected

XC6264 Pinouts - South Side

Pin Description	HQ240	PG299	HQ304	PG411	BG432
S_{42}	P94	L16	P187	AB38	AK15
S_{41}	P95	L17	P186	AD36	AJ14
S_{44}	-	M20	P185	AA35	AH14
S_{43}	-	M19	P184	AE37	AK14
S_{46}	-	N20	P183	AB36	AL13
S_{48}	-	M18	P182	AD38	AK13
V_{CC}	**-**	**-**	**-**	**V_{CC}***	**V_{CC}***
GND	**-**	**-**	**-**	**GND***	**GND***
E_0/S_{50}	P96	M17	P181	AB34	AJ13
E_2/S_{52}	P97	M16	P180	AE39	AH13
S_{93}	-	-	-	AM36	AL12
S_{94}	-	-	-	AC35	AK12
S_{95}	-	-	-	AL35	AJ12
S_{96}	-	-	-	AF38	AK11
S_{97}	-	-	-	-	-
S_{98}	-	-	-	-	-
GND	**P98**	**-**	**-**	**GND***	**GND***
S_{100}	-	-	-	-	-
E_4/S_{54}	P99	N19	P179	AG39	AH12
S_{45}	P100	P20	P178	AG37	AJ11
V_{CC}	**P101**	**T20**	**P177**	**V_{CC}***	**V_{CC}***
NC	-	-	P176(1)	-	-
ShiftDOut/S_{21}	P102	N18	P175	AD34	AL10
SEData/S_{23}	P103	P19	P174	AN39	AK10
E_6/S_{56}	P104	N17	P173	AE35	AJ10
S_{47}	P105	R19	P172	AH38	AK9
GND	**P106**	**R20**	**P171**	**GND***	**GND***
S_{102}	-	N16	P170	AJ37	AL8
S_{104}	-	P18	P169	AG35	AH10
E_8/S_{58}	P107	U20	P168	AF34	AJ9
S_{49}	P108	P17	P167	AH36	AK8
S_{106}	-	-	-	-	-
S_{108}	-	-	-	-	-
GND	**-**	**-**	**-**	**GND***	**GND***
S_{110}	-	-	-	-	-
S_{112}	-	-	-	AK38	AJ8
S_{114}	-	-	-	AP38	AH9
S_{51}	P109	T19	P166	AK36	AK7
S_{53}	P110	R18	P165	AM34	AL6
S_{55}	-	P16	P164	AH34	AJ7

Note: 1. Pin not connected.

XC6264 Pinouts - South Side

Pin Description	HQ240	PG299	HQ304	PG411	BG432
S_{57}	-	V20	P163	AJ35	AH8
GND	**-**	**-**	**-**	**GND***	**GND***
V_{CC}	**-**	**-**	**-**	**V_{CC}***	**V_{CC}***
E_{10}/S_{60}	P111	R17	P162	AL37	AK6
E_{12}/S_{62}	P112	T18	P161	AT38	AL5
$\overline{SECE}/S_{25}$	P113	U19	P160	AM38	AH7
SEReset/S_{27}	P114	V19	P159	AN37	AJ6
S_{116}	-	-	-	AK34	AK5
S_{118}	-	-	-	AR39	AL4
GND	**-**	**-**	**-**	**GND***	**GND***
S_{120}	-	-	-	-	-
S_{122}	-	-	-	-	-
S_{124}	-	-	-	AR37	AH6
S_{126}	-	-	-	AU37	AJ5
S_{59}	P115	R16	P158	AN35	AK4
S_{61}	P116	T17	P157	AL33	AH5
SEClk/S_{29}	P117	U18	P156	AV38	AK3
ConfigOK/S_{31}	P118	X20	P155	AT36	AJ4
GND	**P119**	**W20**	**P154**	**GND***	**GND***
S_{63}	P120	V18	P153	AR35	AH4

XC6264 Pinouts - East Side

Pin Description	HQ240	PG299	HQ304	PG411	BG432
V_{CC}	**P121**	**X19,T16**	**P152**	**V_{CC}***	**V_{CC}***
S_{99}/E_{14}	P122	U17	P151	AN33	AH3
$A0/E_{1}$	P123	W19	P150	AM32	AJ2
S_{101}/E_{16}	P124	W18	P149	AP34	AG4
$A1/E_{3}$	P125	T15	P148	AW39	AG3
S_{103}/E_{18}	P126	U16	P147	AN31	AH2
E_{64}	-	-	-	AV36	AH1
E_{65}	-	-	-	AR33	AF4
E_{66}	-	-	-	-	-
GND	**-**	**-**	**-**	**GND***	**GND***
$A2/E_{5}$	P127	V17	P146	AP32	AF3
S_{105}/E_{20}	P128	X18	P145	AU35	AG2
E_{67}	-	-	-	AV34	AG1
E_{68}	-	-	-	AW35	AE4
E_{69}	-	U15	P144	AW33	AE3
E_{70}	-	T14	P143	AU33	AF2
V_{CC}	**-**	**-**	**-**	**V_{CC}***	**V_{CC}***
GND	**-**	**-**	**-**	**GND***	**GND***
$A3/E_{7}$	P129	W17	P142	AV32	AF1
S_{107}/E_{22}	P130	V16	P141	AU31	AD4
S_{109}/E_{24}	P131	X17	P140	AR31	AD3
S_{111}/E_{26}	P132	U14	P139	AP28	AE2
E_{71}	-	-	-	AP30	AD2
E_{72}	-	-	-	AT30	AC4
E_{73}	-	-	-	-	-
V_{CC}	**-**	**-**	**-**	**-**	**V_{CC}***
GND	**-**	**-**	**-**	**GND***	**GND***
E_{74}	-	-	-	-	-
E_{75}	-	-	-	-	-
$A4/E_{9}$	P133	V15	P138	AT32	AC3
S_{113}/E_{28}	P134	T13	P137	AV30	AD1
E_{76}	-	W16	P136	AR29	AC2
E_{77}	-	W15	P135	AP26	AB4
GND	**P135**	**X16**	**P134**	**GND***	**GND***
S_{115}/E_{30}	P136	U13	P133	AU29	AB3
S_{117}/E_{32}	P137	V14	P132	AV28	AB2
$A5/E_{11}$	P138	W14	P131	AT28	AB1
$A16/E_{33}$	P139	V13	P130	AR25	AA3
V_{CC}	**P140**	**X15**	**P129**	**V_{CC}***	**V_{CC}***

XC6264 Pinouts - East Side

Pin Description	HQ240	PG299	HQ304	PG411	BG432
E_{78}	-	-	P128	-	-
E_{79}	-		-	-	-
S_{119}/E_{34}	P141	T12	P127	AP24	AA2
$A17/E_{35}$	P142	X14	P126	AU27	Y2
GND	**-**	**-**	**-**	**GND***	**GND***
E_{80}	-	-	-	AR27	Y4
E_{81}	-	-	-	AW27	Y3
E_{82}	-	-	-	AU25	Y1
E_{83}	-	-	-	AV26	W1
E_{84}	-	-	-	-	-
E_{85}	-	U12	P125	AT24	W4
E_{86}	-	W13	P124	AR23	W3
GND	**P143**	**-**	**-**	**GND***	**GND***
V_{CC}	**-**	**-**	**-**	**V_{CC}***	**V_{CC}***
E_{87}	-	-	-	AW25	W2
E_{88}	-	-	-	AW23	V2
E_{89}	-	X13	P123	AP22	V4
E_{90}	-	V12	P122	AV24	V3
$A6/E_{13}$	P144	W12	P121	AU23	U1
S_{121}/E_{36}	P145	T11	P120	AT22	U2
GND	**-**	**-**	**-**	**GND***	**GND***
E_{37}	P146	X12	P119	AR21	U4
S_{123}/E_{38}	P147	U11	P118	AV22	U3
E_{91}	-	-	-	-	-
E_{92}	-	-	-	-	-
$A7/E_{15}$	P148	V11	P117	AP20	T1
E_{39}	P149	W11	P116	AU21	T2
V_{CC}	**P150**	**X10**	**P115**	**V_{CC}***	**V_{CC}***
GND	**P151**	**X11**	**P114**	**GND***	**GND***
$A8/E_{17}$	P152	W10	P113	AU19	T3
S_{125}/E_{40}	P153	V10	P112	AV20	R1
E_{93}	-	-	-	-	-
$A9/E_{19}$	P154	T10	P111	AV18	R2
E_{41}	P155	U10	P110	AR19	R4
GND	**-**	**-**	**-**	**GND***	**GND***
S_{127}/E_{42}	P156	X9	P109	AT18	R3
E_{43}	P157	W9	P108	AW17	P2
E_{94}	-	X8	P107	AV16	P3
E_{44}	-	V9	P106	AP18	P4

XC6264 Pinouts - East Side

Pin Description	HQ240	PG299	HQ304	PG411	BG432
N_{127}/E_{95}	-	-	-	AU17	N1
E_{96}	-	-	-	AW15	N2
N_{125}/E_{97}	-	-	-	-	-
V_{CC}	**-**	**-**	**-**	**V_{CC}***	**V_{CC}***
GND	**-**	**-**	**-**	**GND***	**GND***
E_{46}	-	U9	P105	AR17	N3
E_{48}	-	T9	P104	AT16	N4
E_{98}	-	-	-	AV14	M1
N_{123}/E_{99}	-	-	-	AW13	M2
E_{100}	-	-	-	-	-
N_{121}/E_{101}	-	-	-	AU15	M3
N_{119}/E_{103}	-	-	-	AU13	M4
GND	**P158**	**T5**	**-**	**GND***	**GND***
$A10/E_{21}$	P159	W8	P103	AR15	L2
E_{50}	P160	X7	P102	AP16	L3
N_{117}/E_{105}	-	-	-	-	-
V_{CC}	**P161**	**X5**	**P101**	**V_{CC}***	**V_{CC}***
NC	-	-	P100(1)	-	-
E_{45}	P162	V8	P99	AV12	K1
E_{52}	P163	W7	P98	AR13	K2
$A11/E_{23}$	P164	U8	P97	AU11	K3
E_{47}	P165	W6	P96	AT12	K4
GND	**P166**	**X6**	**P95**	**GND***	**GND***
E_{54}	-	T8	P94	AP14	J2
E_{56}	-	V7	P93	AR11	J3
E_{58}	P167	X4	P92	AV10	J4
E_{49}	P168	U7	P91	AT8	H1
N_{115}/E_{107}	-	-	-	-	-
N_{113}/E_{109}	-	-	-	-	-
GND	**-**	**-**	**-**	**GND***	**GND***
V_{CC}	**-**	**-**	**-**	**-**	**V_{CC}***
$A12/E_{25}$	P169	W5	P90	AT10	H2
E_{51}	P170	V6	P89	AP10	H3
E_{53}	P171	T7	P88	AP12	H4
E_{55}	P172	X3	P87	AR9	G2
N_{111}/E_{111}	-	-	-	AU9	G3
N_{109}/E_{113}	-	-	-	AV8	F1
N_{107}/E_{115}	-	-	-	-	-
N_{105}/E_{117}	-	-	-	-	-
GND	**-**	**-**	**-**	**GND***	**GND***

Note: 1. Pin not connected.

XC6264 Pinouts - East Side

Pin Description	HQ240	PG299	HQ304	PG411	BG432
V_{CC}	**-**	**-**	**-**	**V_{CC}***	**V_{CC}***
$A13/E_{27}$	P173	U6	P86	AU7	G4
E_{57}	P174	V5	P85	AW7	F2
N_{103}/E_{119}	-	-	-	AW5	F3
N_{101}/E_{121}	-	-	-	AV6	E1
N_{99}/E_{123}	-	-	-	AU5	F4
N_{97}/E_{125}	-	-	-	AP8	E2
GND	**-**	**-**	**-**	**GND***	**GND***
N_{95}/E_{127}	-	-	-	-	-
E_{60}	-	W4	P84	AR7	E3
E_{62}	-	W3	P83	AV4	D1
$A14/E_{29}$	P175	T6	P82	AN9	E4
E_{59}	P176	U5	P81	AW1	D2
$A15/E_{31}$	P177	V4	P80	AP6	C2
E_{61}	P178	X1	P79	AU3	D3
E_{63}	P179	V3	P78	AR5	D4
V_{CC}	**P180**	**W1**	**P77**	**V_{CC}***	**V_{CC}***

XC6264 Pinouts - North Side

Pin Description	HQ240	PG299	HQ304	PG411	BG432
V_{CC}	**P240**	**A2,E5**	**P1**	**V_{CC}***	**V_{CC}***
N_{64}	-	-	-	-	-
N_1	P239	C3	P2	G7	D28
N_3	P238	D3	P3	E5	C28
W_{102}/N_0	P237	E4	P4	D2	B29
W_{104}/N_2	P236	F5	P5	C3	D27
W_{106}/N_4	P235	C2	P6	H6	B28
W_{108}/N_6	P234	D2	P7	J7	C27
N_{65}	-	-	-	-	-
GND	**-**	**-**	**-**	**GND***	**GND***
N_{66}	-	-	-	-	-
N_{67}	-	-	-	G5	D26
N_{68}	-	-	-	F2	A28
N_{69}	-	-	-	E3	B27
N_{70}	-	-	-	E1	C26
W_{110}/N_8	-	E3	P8	G1	D25
W_{112}/N_{10}	-	F4	P9	K6	A27
V_{CC}	**-**	**-**	**-**	**V_{CC}***	**V_{CC}***
GND	**-**	**-**	**-**	**GND**	**GND***
N_5	P233	C1	P10	G3	B26
NC	-	-	P11(1)	-	-
N_7	P232	G5	P12	H4	D24
W_{114}/N_{12}	P231	F3	P13	H2	C25
W_{116}/N_{14}	P230	E2	P14	J3	A26
N_{71}	-	-	-	J5	B25
N_{72}	-	-	-	L5	D23
N_{73}	-	-	-	-	-
N_{74}	-	-	-	-	-
V_{CC}	**-**	**-**	**-**	**-**	**V_{CC}***
GND	**-**	**-**	**-**	**GND***	**GND***
N_{75}	-	-	-	-	-
N_{76}	-	-	-	-	-
N_{77}	-	-	-	-	-
N_{78}	-	-	-	-	-
W_{118}/N_{16}	P229	G4	P15	M6	C24
W_{120}/N_{18}	P228	D1	P16	P6	B24
W_{122}/N_{20}	-	G3	P17	K4	C23
W_{124}/N_{22}	-	H5	P18	K2	D22
GND	**P227**	**F1**	**P19**	**GND***	**GND***
W_{126}/N_{24}	P226	F2	P20	M4	A24

Note: 1. Pin not connected.

XC6264 Pinouts - North Side

Pin Description	HQ240	PG299	HQ304	PG411	BG432
N_{26}	P225	H4	P21	N5	B23
N_{33}	P224	G2	P22	T6	C22
N_{28}	P223	H3	P23	L3	B22
NC	-	-	P24(1)	-	-
V_{CC}	**P222**	**E1**	**P25**	**V_{CC}***	**V_{CC}***
N_{30}	-	G1	P26	M2	A22
N_{32}	-	H2	P27	R5	C21
GND	**-**	**-**	**-**	**GND***	**GND***
N_{79}	-	-	-	-	-
N_{80}	-	-	-	-	-
N_{81}	-	-	-	N3	D20
N_{82}	-	-	-	R3	B21
N_{83}	-	-	-	N1	C20
N_{84}	-	-	-	P2	B20
N_{34}	-	J5	P28	T4	A20
N_{36}	-	J4	P29	U5	D19
GND	**-**	**-**	**-**	**GND***	**GND***
V_{CC}	**-**	**-**	**-**	**V_{CC}***	**V_{CC}***
N_{35}	P221	J3	P30	R1	C19
N_{9}	P220	H1	P31	U3	B19
GND	**P219**	**-**	**-**	**GND***	**GND***
N_{11}	P218	J2	P32	V6	A19
N_{38}	P217	J1	P33	U1	B18
N_{37}	P216	K4	P34	T2	D18
N_{40}	P215	K5	P35	V4	C18
GND	**-**	**-**	**-**	**GND***	**GND***
N_{85}	-	-	-	-	-
N_{86}	-	-	-	-	-
N_{87}	-	-	-	W5	B17
N_{88}	-	-	-	V2	C17
N_{39}	P214	K3	P36	Y2	A17
N_{13}	P213	K2	P37	W3	D17
V_{CC}	**P212**	**K1**	**P38**	**V_{CC}***	**V_{CC}***
GND	**P211**	**L1**	**P39**	**GND***	**GND***
N_{15}	P210	L2	P40	Y6	A16
N_{17}	P209	L3	P41	AA3	B16
N_{89}	-	-	-	AC3	C16
N_{90}	-	-	-	AB2	A15
N_{91}	-	-	-	-	-
N_{92}	-	-	-	-	-

Note: 1. Pin not connected.

XC6264 Pinouts - North Side

Pin Description	HQ240	PG299	HQ304	PG411	BG432
GND	**-**	**-**	**-**	**GND***	**GND***
N_{93}	-	-	-	-	-
N_{94}	-	-	-	-	-
N_{19}	P208	L4	P42	AA5	B15
N_{42}	P207	M1	P43	AD4	C15
N_{41}	P206	L5	P44	AC1	D15
N_{44}	P205	M2	P45	AE3	B14
GND	**P204**	**-**	**-**	**GND***	**GND***
N_{43}	P203	M3	P46	AB4	A13
N_{21}	P202	N1	P47	AD2	C14
V_{CC}	**-**	**-**	**-**	**V_{CC}***	**V_{CC}***
GND	**-**	**-**	**-**	**GND***	**GND***
N_{46}	-	N2	P48	AB6	B13
N_{48}	-	M4	P49	AH4	D14
N_{96}	-	-	-	AE1	A12
N_{98}	-	-	-	AC5	C13
N_{100}	-	-	-	AJ5	B12
E_{126}/N_{102}	-	-	-	AF2	D13
E_{124}/N_{104}	-	-	-	-	-
GND	**-**	**-**	**-**	**GND***	**GND***
N_{50}	-	P1	P50	AG1	C12
N_{52}	-	M5	P51	AG3	B11
V_{CC}	**P201**	**R1**	**P52**	**V_{CC}***	**V_{CC}***
NC	-	-	P53(1)	-	-
N_{23}	P200	N3	P54	AD6	D12
N_{54}	P199	P2	P55	AM4	C11
N_{45}	P198	P3	P56	AE5	A10
N_{56}	P197	N4	P57	AH2	B10
GND	**P196**	**T1**	**P58**	**GND***	**GND***
E_{122}/N_{106}	-	R2	P59	AJ3	C10
N_{47}	P195	T2	P60	AL5	B9
N_{58}	P194	N5	P61	AF6	C9
N_{49}	P193	R3	P62	AG5	D10
E_{120}/N_{108}	-	-	-	-	-
GND	**-**	**-**	**-**	**GND***	**GND***
V_{CC}	**-**	**-**	**-**	**-**	**V_{CC}***
E_{118}/N_{110}	-	-	-	-	A8
E_{116}/N_{112}	-	-	-	AK2	B8
E_{114}/N_{114}	-	-	-	AN1	D9
N_{51}	P192	P4	P63	AK4	C8

Note: 1. Pin not connected.

XC6264 Pinouts - North Side

Pin Description	HQ240	PG299	HQ304	PG411	BG432
N_{53}	P191	U1	P64	AP2	B7
N_{55}	P190	T3	P65	AH6	C7
N_{57}	P189	U2	P66	AL3	D8
GND	**-**	**-**	**-**	**GND***	**GND***
V_{CC}	**-**	**-**	**-**	**V_{CC}***	**V_{CC}***
N_{60}	-	P5	P67	AM2	A6
N_{62}	-	R4	P68	AM6	B6
N_{59}	P188	V1	P69	AN3	D7
N_{25}	P187	U3	P70	AK6	A5
E_{112}/N_{116}	-	-	-	AR1	C6
E_{110}/N_{118}	-	-	-	AR3	B5
GND	**-**	**-**	**-**	**GND***	**GND***
E_{108}/N_{120}	-	-	-	-	-
E_{106}/N_{122}	-	-	-	AN5	D6
E_{104}/N_{124}	-		-	AT2	A4
N_{27}	P186	T4	P71	AL7	C5
N_{61}	P185	R5	P72	AM8	B4
N_{63}	P184	V2	P73	AV2	D5
N_{29}	P183	W2	P74	AT4	B3
GND	**P182**	**X2**	**P75**	**GND***	**GND***
N_{31}	P181	U4	P76	AN7	C4
E_{102}/N_{126}	-	-	-	-	-

Pads labelled GND* are internally bonded to a Ground plane within the associated package. They have no direct connection to any package pin.

Pads labelled V_{CC}* are internally bonded to a V_{CC} plane within the associated package. They have no direct connection to any package pin.

XC6264: V_{CC} & Ground Connections on PG411 Package

V_{CC}	GND
A3	A9
A11	A19
A21	A29
A31	A37
C39	C1
D6	D14
F36	D20
J1	D26
L39	D34
W1	F4
AA39	J39
AJ1	L1
AL39	P4
AP4	P36
AT34	W39
AU1	Y4
AW9	Y36
AW19	AA1
AW29	AF4
AW37	AF36
	AJ39
	AL1
	AP36
	AT6
	AT14
	AT20
	AT26
	AU39
	AW3
	AW11
	AW21
	AW31

XC6264: V_{CC} & Ground Connections on BG432 Package

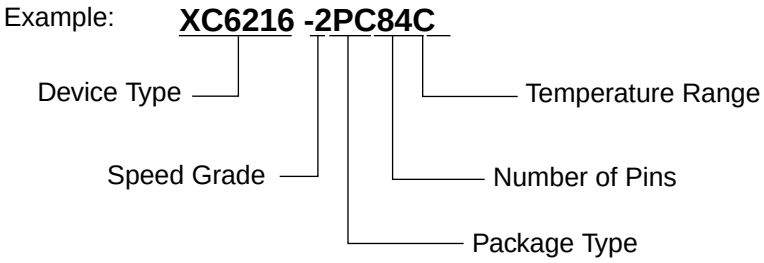
V_{CC}	GND	GND
A1	A2	AL3
A11	A3	AL7
A21	A7	AL9
A31	A9	AL14
C3	A14	AL18
C29	A18	AL23
D11	A23	AL25
D21	A25	AL29
L1	A29	AL30
L4	A30	
L28	B1	
L31	B2	
AA1	B30	
AA4	B31	
AA28	C1	
AA31	C31	
AH11	D16	
AH21	G1	
AJ3	G31	
AJ29	J1	
AL1	J31	
AL11	P1	
AL21	P31	
AL31	T4	
	T28	
	V1	
	V31	
	AC1	
	AC31	
	AE1	
	AE31	
	AH16	
	AJ1	
	AJ31	
	AK1	
	AK2	
	AK30	
	AK31	
	AL2	

For a detailed description of the device architecture, see page 3.

For a detailed description of the device timing, see “Electrical Parameters” on page 37.

For package physical dimensions and thermal data, see package section of Xilinx databook.

Ordering Information



Speed Options

-2	Faster option
-3	Slower option

Packaging Options

PC84	84-Pin PLCC
HT144	144-Pin TQFP
BG225	225-Pin Ball-Grid-Array
HQ240	240-Pin PQFP
HQ304	304-Pin PQFP
PG299	299-Pin-Grid-Array
PG411	411-Pin-Grid-Array
BG432	432-Pin Ball-Grid-Array

Temperature Options

C	Commercial,	T_J=0°C to 85°C (Junction)
I	Industrial,	T_J=-40°C to 100°C (Junction)
M	Military,	T_C=-55°C to 125°C (Case)