Show Xilinx Basic HDL coding video

FPGA overview (SPARTAN) FPGA Fabric only Slide

Virtex-5 Family overview product spec

Show CLB vs Slice document

Show Xilinx Virtex-5 coding techniques video

pause at 15 minutes

---

then cover BRAMS, ROM, FSB in ROM and SRL

Virtex-5 HDL libraries

then resume video

Original FPGA Vendor	Current Owner / Vendor	Key Product Families
Xilinx	AMD	Virtex, Kintex, Artix, Zynq, Versal
Actel	Microchip	ProASIC3, IGLOO, Fusion, Antifuse (RTSX/RTX)
Microsemi	Microchip	PolarFire, RTG4, SmartFusion2, IGLOO2
Atmel (FPGA business)	Microchip	AT6000, FPSLIC
Dialog Semiconductor	Renesas	ForgeFPGA

The AMD/Xilinx Phasing: While you'll still see the "Xilinx" name on legacy documentation and older chips (like the Virtex-5), all new parts and the website are branded as **AMD**.

Virtex-5 Family Overview

DS100 (v5.1) August 21, 2015 **Product Specification**

General Description

Using the second generation ASMBL™ (Advanced Silicon Modular Block) column-based architecture, the Virtex-5 family contains five distinct platforms (sub-families), the most choice offered by any FPGA family. Each platform contains a different ratio of features to address the needs of a wide variety of advanced logic designs. In addition to the most advanced, high-performance logic fabric, Virtex-5 FPGAs contain many hard-IP system level blocks, including powerful 36-Kbit block RAM/FIFOs, second generation 25 x 18 DSP slices, SelectIO™ technology with built-in digitally-controlled impedance, ChipSync™ source-synchronous interface blocks, system monitor functionality, enhanced clock management tiles with integrated DCM (Digital Clock Managers) and phase-locked-loop (PLL) clock generators, and advanced configuration options. Additional platform dependant features include power-optimized high-speed serial transceiver blocks for enhanced serial connectivity, PCI Express® compliant integrated Endpoint blocks, tri-mode Ethernet MACs (Media Access Controllers), and high-performance PowerPC® 440 microprocessor embedded blocks. These features allow advanced logic designers to build the highest levels of performance and functionality into their FPGA-based systems. Built on a 65-nm state-of-the-art copper process technology, Virtex-5 FPGAs are a programmable alternative to custom ASIC technology. Most advanced system designs require the programmable strength of FPGAs. Virtex-5 FPGAs offer the best solution for addressing the needs of high-performance logic designers, high-performance DSP designers, and high-performance embedded systems designers with unprecedented logic, DSP, hard/soft microprocessor, and connectivity capabilities. The Virtex-5 LXT, SXT, TXT, and FXT platforms include advanced high-speed serial connectivity and link/transaction layer capability.

Summary of Virtex-5 FPGA Features

- Five platforms LX, LXT, SXT, TXT, and FXT
  - Virtex-5 LX: High-performance general logic applications
  - Virtex-5 LXT: High-performance logic with advanced serial connectivity
  - Virtex-5 SXT: High-performance signal processing applications with advanced serial connectivity
  - Virtex-5 TXT: High-performance systems with double density advanced serial connectivity
  - Virtex-5 FXT: High-performance embedded systems with advanced serial connectivity

  Cross-platform compatibility
  - LXT, SXT, and FXT devices are footprint compatible in the same package using adjustable voltage regulators
- Most advanced, high-performance, optimal-utilization, FPGA fabric
  - Real 6-input look-up table (LUT) technology
  - Dual 5-LUT option
  - Improved reduced-hop routing
  - 64-bit distributed RAM option
  - SRL32/Dual SRL16 option
- Powerful clock management tile (CMT) clocking
  - Digital Clock Manager (DCM) blocks for zero delay buffering, frequency synthesis, and clock phase shifting
  - PLL blocks for input jitter filtering, zero delay buffering, frequency synthesis, and phase-matched clock division
- 36-Kbit block RAM/FIFOs
  - True dual-port RAM blocks
  - Enhanced optional programmable FIFO logic
  - Programmable
    - True dual-port widths up to x36
    - Simple dual-port widths up to x72
  - Built-in optional error-correction circuitry
  - Optionally program each block as two independent 18-Kbit blocks
- High-performance parallel SelectIO technology
  - 1.2 to 3.3V I/O Operation
  - Source-synchronous interfacing using ChipSync™ technology
  - Digitally-controlled impedance (DCI) active termination
  - Flexible fine-grained I/O banking
  - High-speed memory interface support
- Advanced DSP48E slices
  - 25 x 18, two's complement, multiplication
  - Optional adder, subtracter, and accumulator
  - Optional pipelining
  - Optional bitwise logical functionality
  - Dedicated cascade connections
- Flexible configuration options
  - SPI and Parallel FLASH interface
  - Multi-bitstream support with dedicated fallback reconfiguration logic
  - Auto bus width detection capability
- System Monitoring capability on all devices
  - On-chip/Off-chip thermal monitoring
  - On-chip/Off-chip power supply monitoring
  - JTAG access to all monitored quantities
- Integrated Endpoint blocks for PCI Express Designs
  - LXT, SXT, TXT, and FXT Platforms
  - Compliant with the PCI Express Base Specification 1.1
  - x1, x4, or x8 lane support per block
  - Works in conjunction with RocketIO™ transceivers
- Tri-mode 10/100/1000 Mb/s Ethernet MACs
  - LXT, SXT, TXT, and FXT Platforms
  - RocketIO transceivers can be used as PHY or connect to external PHY using many soft MII (Media Independent Interface) options
- RocketIO GTP transceivers 100 Mb/s to 3.75 Gb/s
  - LXT and SXT Platforms
- RocketIO GTX transceivers 150 Mb/s to 6.5 Gb/s
  - TXT and FXT Platforms
- PowerPC 440 Microprocessors
  - FXT Platform only
  - RISC architecture
  - 7-stage pipeline
  - 32-Kbyte instruction and data caches included
  - Optimized processor interface structure (crossbar)
- 65-nm copper CMOS process technology
- 1.0V core voltage
- High signal-integrity flip-chip packaging available in standard or Pb-free package options

Table 1: **Virtex-5 FPGA Family Members**

Device	Configurable Logic Blocks (CLBs)			DSP48E Slices[2]	Block RAM Blocks			CMTs[4]	PowerPC Processor Blocks	Endpoint Blocks for PCI Express	Ethernet MACs[5]	Max RocketIO Transceivers[6]		Total I/O Banks[8]	Max User I/O[7]
	Array (Row x Col)	Virtex-5 Slices[1]	Max Distributed RAM (Kb)		18 Kb[3]	36 Kb	Max (Kb)					GTP	GTX		
XC5VLX30	80 x 30	4,800	320	32	64	32	1,152	2	N/A	N/A	N/A	N/A	N/A	13	400
XC5VLX50	120 x 30	7,200	480	48	96	48	1,728	6	N/A	N/A	N/A	N/A	N/A	17	560
XC5VLX85	120 x 54	12,960	840	48	192	96	3,456	6	N/A	N/A	N/A	N/A	N/A	17	560
XC5VLX110	160 x 54	17,280	1,120	64	256	128	4,608	6	N/A	N/A	N/A	N/A	N/A	23	800
XC5VLX155	160 x 76	24,320	1,640	128	384	192	6,912	6	N/A	N/A	N/A	N/A	N/A	23	800
XC5VLX220	160 x 108	34,560	2,280	128	384	192	6,912	6	N/A	N/A	N/A	N/A	N/A	23	800
XC5VLX330	240 x 108	51,840	3,420	192	576	288	10,368	6	N/A	N/A	N/A	N/A	N/A	33	1,200
XC5VLX20T	60 x 26	3,120	210	24	52	26	936	1	N/A	1	2	4	N/A	7	172
XC5VLX30T	80 x 30	4,800	320	32	72	36	1,296	2	N/A	1	4	8	N/A	12	360
XC5VLX50T	120 x 30	7,200	480	48	120	60	2,160	6	N/A	1	4	12	N/A	15	480
XC5VLX85T	120 x 54	12,960	840	48	216	108	3,888	6	N/A	1	4	12	N/A	15	480
XC5VLX110T	160 x 54	17,280	1,120	64	296	148	5,328	6	N/A	1	4	16	N/A	20	680
XC5VLX155T	160 x 76	24,320	1,640	128	424	212	7,632	6	N/A	1	4	16	N/A	20	680
XC5VLX220T	160 x 108	34,560	2,280	128	424	212	7,632	6	N/A	1	4	16	N/A	20	680
XC5VLX330T	240 x 108	51,840	3,420	192	648	324	11,664	6	N/A	1	4	24	N/A	27	960
XC5VSX35T	80 x 34	5,440	520	192	168	84	3,024	2	N/A	1	4	8	N/A	12	360
XC5VSX50T	120 x 34	8,160	780	288	264	132	4,752	6	N/A	1	4	12	N/A	15	480
XC5VSX95T	160 x 46	14,720	1,520	640	488	244	8,784	6	N/A	1	4	16	N/A	19	640
XC5VSX240T	240 x 78	37,440	4,200	1,056	1,032	516	18,576	6	N/A	1	4	24	N/A	27	960
XC5VTX150T	200 x 58	23,200	1,500	80	456	228	8,208	6	N/A	1	4	N/A	40	20	680
XC5VTX240T	240 x 78	37,440	2,400	96	648	324	11,664	6	N/A	1	4	N/A	48	20	680
XC5VFX30T	80 x 38	5,120	380	64	136	68	2,448	2	1	1	4	N/A	8	12	360
XC5VFX70T	160 x 38	11,200	820	128	296	148	5,328	6	1	3	4	N/A	16	19	640
XC5VFX100T	160 x 56	16,000	1,240	256	456	228	8,208	6	2	3	4	N/A	16	20	680
XC5VFX130T	200 x 56	20,480	1,580	320	596	298	10,728	6	2	3	6	N/A	20	24	840
XC5VFX200T	240 x 68	30,720	2,280	384	912	456	16,416	6	2	4	8	N/A	24	27	960

Notes:

1. Virtex-5 FPGA slices are organized differently from previous generations. Each Virtex-5 FPGA slice contains four LUTs and four flip-flops (previously it was two LUTs and two flip-flops.)
2. Each DSP48E slice contains a 25 x 18 multiplier, an adder, and an accumulator.
3. Block RAMs are fundamentally 36 Kbits in size. Each block can also be used as two independent 18-Kbit blocks.
4. Each Clock Management Tile (CMT) contains two DCMs and one PLL.
5. This table lists separate Ethernet MACs per device.
6. RocketIO GTP transceivers are designed to run from 100 Mb/s to 3.75 Gb/s. RocketIO GTX transceivers are designed to run from 150 Mb/s to 6.5 Gb/s.
7. This number does not include RocketIO transceivers.
8. Includes configuration Bank 0.

Digitally Controlled Impedance (DCI) Active I/O Termination

- Optional series or parallel termination
- Temperature and voltage compensation
- Makes board layout much easier
  - Reduces resistors
  - Places termination in the ideal location, at the signal source or destination

Configuration

- Support for platform Flash, standard SPI Flash, or standard parallel NOR Flash configuration
- Bitstream support with dedicated fallback reconfiguration logic
- 256-bit AES bitstream decryption provides intellectual property security and prevents design copying
- Improved bitstream error detection/correction capability
- Auto bus width detection capability
- Partial Reconfiguration via ICAP port

Advanced Flip-Chip Packaging

- Pre-engineered packaging technology for proven superior signal integrity
  - Minimized inductive loops from signal to return
  - Optimal signal-to-PWR/GND ratios
- Reduces SSO induced noise by up to 7x
- Pb-Free and standard packages

System Monitor

- On-Chip temperature measurement (±4°C)
- On-Chip power supply measurement (±1%)
- Easy to use, self-contained
  - No design required for basic operation
  - Autonomous monitoring of all on-chip sensors
  - User programmable alarm thresholds for on-chip sensors
- User accessible 10-bit 200kSPS ADC
  - Automatic calibration of offset and gain error
  - DNL = ±0.9 LSBs maximum
- Up to 17 external analog input channels supported
  - 0V to 1V input range
  - Monitor external sensors e.g., voltage, temperature
  - General purpose analog inputs
- Full access from fabric or JTAG TAP to System Monitor
- Fully operational prior to FPGA configuration and during device power down (access via JTAG TAP only)

65-nm Copper CMOS Process

- 1.0V Core Voltage
- 12-layer metal provides maximum routing capability and accommodates hard-IP immersion
- Triple-oxide technology for proven reduced static power consumption

System Blocks Specific to the LXT, SXT, TXT, and FXT Devices

Integrated Endpoint Block for PCI Express Compliance

- Works in conjunction with RocketIO GTP transceivers (LXT and SXT) and GTX transceivers (TXT and FXT) to deliver full PCI Express Endpoint functionality with minimal FPGA logic utilization.
- Compliant with the PCI Express Base Specification 1.1
- PCI Express Endpoint block or Legacy PCI Express Endpoint block
- x8, x4, or x1 lane width
- Power management support
- Block RAMs used for buffering
- Fully buffered transmit and receive
- Management interface to access PCI Express configuration space and internal configuration
- Supports the full range of maximum payload sizes
- Up to 6 x 32 bit or 3 x 64 bit BARs (or a combination of 32 bit and 64 bit)

Tri-Mode Ethernet Media Access Controller

- Designed to the IEEE 802.3-2002 specification
- Operates at 10, 100, and 1,000 Mb/s
- Supports tri-mode auto-negotiation
- Receive address filter (5 address entries)
- Fully monolithic 1000Base-X solution with RocketIO GTP transceivers
- Supports multiple external PHY connections (RGMII, GMII, etc.) interfaces through soft logic and SelectIO resources
- Supports connection to external PHY device through SGMII using soft logic and RocketIO GTP transceivers
- Receive and transmit statistics available through separate interface
- Separate host and client interfaces
- Support for jumbo frames
- Support for VLAN
- Flexible, user-configurable host interface
- Supports IEEE 802.3ah-2004 unidirectional mode

Virtex-5 FPGA Logic

- On average, one to two speed grade improvement over Virtex-4 devices
- Cascadable 32-bit variable shift registers or 64-bit distributed memory capability
- Superior routing architecture with enhanced diagonal routing supports block-to-block connectivity with minimal hops
- Up to 330,000 logic cells including:
  - Up to 207,360 internal fabric flip-flops with clock enable (XC5VLX330)
  - Up to 207,360 real 6-input look-up tables (LUTs) with greater than 13 million total LUT bits
  - Two outputs for dual 5-LUT mode gives enhanced utilization
  - Logic expanding multiplexers and I/O registers

550 MHz Clock Technology

- Up to six Clock Management Tiles (CMTs)
  - Each CMT contains two DCMs and one PLL—up to eighteen total clock generators
  - Flexible DCM-to-PLL or PLL-to-DCM cascade
  - Precision clock deskew and phase shift
  - Flexible frequency synthesis
  - Multiple operating modes to ease performance trade-off decisions
  - Improved maximum input/output frequency
  - Fine-grained phase shifting resolution
  - Input jitter filtering
  - Low-power operation
  - Wide phase shift range
- Differential clock tree structure for optimized low-jitter clocking and precise duty cycle
- 32 global clock networks
- Regional, I/O, and local clocks in addition to global clocks

SelectIO Technology

- Up to 1,200 user I/Os
- Wide selection of I/O standards from 1.2V to 3.3V
- Extremely high-performance
  - Up to 800 Mb/s HSTL and SSTL (on all single-ended I/Os)
  - Up to 1.25 Gb/s LVDS (on all differential I/O pairs)
- True differential termination on-chip
- Same edge capture at input and output I/Os
- Extensive memory interface support

550 MHz Integrated Block Memory

- Up to 16.4 Mbits of integrated block memory
- 36-Kbit blocks with optional dual 18-Kbit mode
- True dual-port RAM cells
- Independent port width selection (x1 to x72)
  - Up to x36 total per port for true dual port operation
  - Up to x72 total per port for simple dual port operation (one Read port and one Write port)
  - Memory bits plus parity/sideband memory support for x9, x18, x36, and x72 widths
  - Configurations from 32K x 1 to 512 x 72 (8K x 4 to 512 x 72 for FIFO operation)
- Multirate FIFO support logic
  - Full and Empty flag with fully programmable Almost Full and Almost Empty flags
- Synchronous FIFO support without Flag uncertainty
- Optional pipeline stages for higher performance
- Byte-write capability
- Dedicated cascade routing to form 64K x 1 memory without using FPGA routing
- Integrated optional ECC for high-reliability memory requirements
- Special reduced-power design for 18 Kbit (and below) operation

550 MHz DSP48E Slices

- 25 x 18 two's complement multiplication
- Optional pipeline stages for enhanced performance
- Optional 48-bit accumulator for multiply accumulate (MACC) operation with optional accumulator cascade to 96-bits
- Integrated adder for complex-multiply or multiply-add operation
- Optional bitwise logical operation modes
- Independent C registers per slice
- Fully cascadable in a DSP column without external routing resources

ChipSync Source-Synchronous Interfacing Logic

- Works in conjunction with SelectIO technology to simplify source-synchronous interfaces
- Per-bit deskew capability built into all I/O blocks (variable delay line on all inputs and outputs)
- Dedicated I/O and regional clocking resources (pins and trees)
- Built-in data serializer/deserializer logic with corresponding clock divider support in all I/O
- Networking/telecommunication interfaces up to 1.25 Gb/s per I/O

RocketIO GTP Transceivers (LXT/SXT only)

- Full-duplex serial transceiver capable of 100 Mb/s to 3.75 Gb/s baud rates
- 8B/10B, user-defined FPGA logic, or no encoding options
- Channel bonding support
- CRC generation and checking
- Programmable pre-emphasis or pre-equalization for the transmitter
- Programmable termination and voltage swing
- Programmable equalization for the receiver
- Receiver signal detect and loss of signal indicator
- User dynamic reconfiguration using secondary configuration bus
- Out of Band (OOB) support for Serial ATA (SATA)
- Electrical idle, beaconing, receiver detection, and PCI Express and SATA spread-spectrum clocking support
- Less than 100 mW typical power consumption
- Built-in PRBS Generators and Checkers

RocketIO GTX Transceivers (TXT/FXT only)

- Full-duplex serial transceiver capable of 150 Mb/s to 6.5 Gb/s baud rates
- 8B/10B encoding and programmable gearbox to support 64B/66B and 64B/67B encoding, user-defined FPGA logic, or no encoding options
- Channel bonding support
- CRC generation and checking
- Programmable pre-emphasis or pre-equalization for the transmitter
- Programmable termination and voltage swing
- Programmable continuous time equalization for the receiver
- Programmable decision feedback equalization for the receiver
- Receiver signal detect and loss of signal indicator
- User dynamic reconfiguration using secondary configuration bus
- OOB support (SATA)
- Electrical idle, beaconing, receiver detection, and PCI Express spread-spectrum clocking support
- Low-power operation at all line rates

PowerPC 440 RISC Cores (FXT only)

- Embedded PowerPC 440 (PPC440) cores
  - Up to 550 MHz operation
  - Greater than 1000 DMIPS per core
  - Seven-stage pipeline
  - Multiple instructions per cycle
  - Out-of-order execution
  - 32 Kbyte, 64-way set associative level 1 instruction cache
  - 32 Kbyte, 64-way set associative level 1 data cache
  - Book E compliant
- Integrated crossbar for enhanced system performance
  - 128-bit Processor Local Buses (PLBs)
  - Integrated scatter/gather DMA controllers
  - Dedicated interface for connection to DDR2 memory controller
  - Auto-synchronization for non-integer PLB-to-CPU clock ratios
- Auxiliary Processor Unit (APU) Interface and Controller
  - Direct connection from PPC440 embedded block to FPGA fabric-based coprocessors
  - 128-bit wide pipelined APU Load/Store
  - Support of autonomous instructions: no pipeline stalls
  - Programmable decode for custom instructions

Architectural Description

Virtex-5 FPGA Array Overview

Virtex-5 devices are user-programmable gate arrays with various configurable elements and embedded cores optimized for high-density and high-performance system designs. Virtex-5 devices implement the following functionality:

- I/O blocks provide the interface between package pins and the internal configurable logic. Most popular and leading-edge I/O standards are supported by programmable I/O blocks (IOBs). The IOBs can be connected to very flexible ChipSync logic for enhanced source-synchronous interfacing. Source-synchronous optimizations include per-bit deskew (on both input and output signals), data serializers/deserializers, clock dividers, and dedicated I/O and local clocking resources.
- Configurable Logic Blocks (CLBs), the basic logic elements for Xilinx® FPGAs, provide combinatorial and synchronous logic as well as distributed memory and SRL32 shift register capability. Virtex-5 FPGA CLBs are based on real 6-input look-up table technology and provide superior capabilities and performance compared to previous generations of programmable logic.
- Block RAM modules provide flexible 36 Kbit true dual-port RAM that are cascadable to form larger memory blocks. In addition, Virtex-5 FPGA block RAMs contain optional programmable FIFO logic for increased device utilization. Each block RAM can also be configured as two independent 18 Kbit true dual-port RAM blocks, providing memory granularity for designs needing smaller RAM blocks.
- Cascadable embedded DSP48E slices with 25 x 18 two's complement multipliers and 48-bit adder/subtracter/accumulator provide massively parallel DSP algorithm support. In addition, each DSP48E slice can be used to perform bitwise logical functions.
- Clock Management Tile (CMT) blocks provide the most flexible, highest-performance clocking for FPGAs. Each CMT contains two Digital Clock Manager (DCM) blocks (self-calibrating, fully digital), and one PLL block (self-calibrating, analog) for clock distribution delay compensation, clock multiplication/division, coarse-/fine-grained clock phase shifting, and input clock jitter filtering.

Additionally, LXT, SXT, TXT, and FXT devices also contain:

- Integrated Endpoint blocks for PCI Express designs providing x1, x4, or x8 PCI Express Endpoint functionality. When used in conjunction with RocketIO transceivers, a complete PCI Express Endpoint can be implemented with minimal FPGA logic utilization.
- 10/100/1000 Mb/s Ethernet media-access control blocks offer Ethernet capability.

LXT and SXT devices contain:

- RocketIO GTP transceivers capable of running up to 3.75 Gb/s. Each GTP transceiver supports full-duplex, clock-and-data recovery.

TXT and FXT devices contain:

- GTX transceivers capable of running up to 6.5 Gb/s. Each GTX transceiver supports full-duplex, clock-and-data recovery.

FXT devices contain:

- Embedded IBM PowerPC 440 RISC CPUs. Each PowerPC 440 CPU is capable of running up to 550 MHz. Each PowerPC 440 CPU also has an APU (Auxiliary Processor Unit) interface that supports hardware acceleration, and an integrated cross-bar for high data throughput.

The general routing matrix (GRM) provides an array of routing switches between each internal component. Each programmable element is tied to a switch matrix, allowing multiple connections to the general routing matrix. The overall programmable interconnection is hierarchical and designed to support high-speed designs. In Virtex-5 devices, the routing connections are optimized to support CLB interconnection in the fewest number of "hops." Reducing hops greatly increases post place-and-route (PAR) design performance.

All programmable elements, including the routing resources, are controlled by values stored in static storage elements. These values are loaded into the FPGA during configuration and can be reloaded to change the functions of the programmable elements.

Chapter 5

Configurable Logic Blocks (CLBs)

CLB Overview

The Configurable Logic Blocks (CLBs) are the main logic resources for implementing sequential as well as combinatorial circuits. Each CLB element is connected to a switch matrix for access to the general routing matrix (shown in Figure 5-1). A CLB element contains a pair of slices. These two slices do not have direct connections to each other, and each slice is organized as a column. Each slice in a column has an independent carry chain. For each CLB, slices in the bottom of the CLB are labeled as SLICE(0), and slices in the top of the CLB are labeled as SLICE(1).

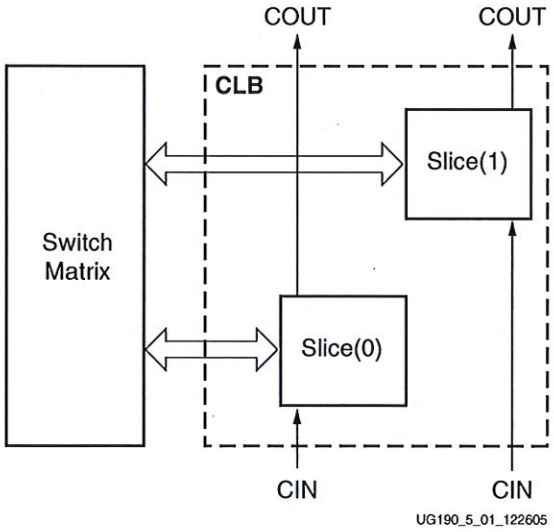


Figure 5-1: **Arrangement of Slices within the CLB**

The Xilinx tools designate slices with the following definitions. An "X" followed by a number identifies the position of each slice in a pair as well as the column position of the slice. The "X" number counts slices starting from the bottom in sequence 0, 1 (the first CLB column); 2, 3 (the second CLB column); etc. A "Y" followed by a number identifies a row of slices. The number remains the same within a CLB, but counts up in sequence from one CLB row to the next CLB row, starting from the bottom. Figure 5-2 shows four CLBs located in the bottom-left corner of the die.

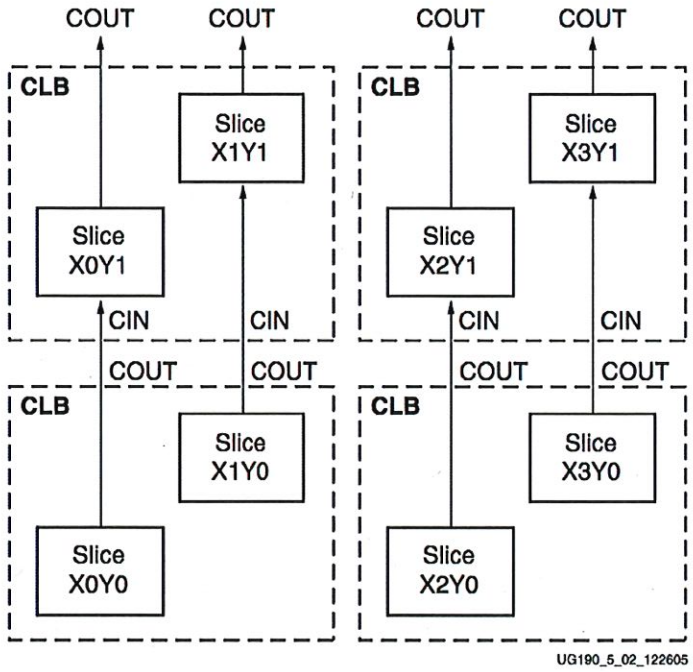


Figure 5-2: **Row and Column Relationship between CLBs and Slices**

Slice Description

Every slice contains four logic-function generators (or look-up tables), four storage elements, wide-function multiplexers, and carry logic. These elements are used by all slices to provide logic, arithmetic, and ROM functions. In addition to this, some slices support two additional functions: storing data using distributed RAM and shifting data with 32-bit registers. Slices that support these additional functions are called SLICEM; others are called SLICEL. SLICEM (shown in Figure 5-3) represents a superset of elements and connections found in all slices. SLICEL is shown in Figure 5-4.

Virtex-5 FPGA Features

This section briefly describes the features of the Virtex-5 family of FPGAs.

Input/Output Blocks (SelectIO)

IOBs are programmable and can be categorized as follows:

- Programmable single-ended or differential (LVDS) operation
- Input block with an optional single data rate (SDR) or double data rate (DDR) register
- Output block with an optional SDR or DDR register
- Bidirectional block
- Per-bit deskew circuitry
- Dedicated I/O and regional clocking resources
- Built-in data serializer/deserializer

The IOB registers are either edge-triggered D-type flip-flops or level-sensitive latches.

IOBs support the following single-ended standards:

- LVTTL
- LVCMOS (3.3V, 2.5V, 1.8V, 1.5V, and 1.2V)
- PCI (33 and 66 MHz)
- PCI-X
- GTL and GTLP
- HSTL 1.5V and 1.8V (Class I, II, III, and IV)
- HSTL 1.2V (Class 1)
- SSTL 1.8V and 2.5V (Class I and II)

The Digitally Controlled Impedance (DCI) I/O feature can be configured to provide on-chip termination for each single-ended I/O standard and some differential I/O standards.

The IOB elements also support the following differential signaling I/O standards:

- LVDS and Extended LVDS (2.5V only)
- BLVDS (Bus LVDS)
- ULVDS
- Hypertransport™
- Differential HSTL 1.5V and 1.8V (Class I and II)
- Differential SSTL 1.8V and 2.5V (Class I and II)
- RSDS (2.5V point-to-point)

Two adjacent pads are used for each differential pair. Two or four IOB blocks connect to one switch matrix to access the routing resources.

Per-bit deskew circuitry allows for programmable signal delay internal to the FPGA. Per-bit deskew flexibly provides fine-grained increments of delay to carefully produce a range of signal delays. This is especially useful for synchronizing signal edges in source-synchronous interfaces.

General purpose I/O in select locations (eight per bank) are designed to be "regional clock capable" I/O by adding special hardware connections for I/O in the same locality. These regional clock inputs are distributed within a limited region to minimize clock skew between IOBs. Regional I/O clocking supplements the global clocking resources.

Data serializer/deserializer capability is added to every I/O to support source-synchronous interfaces. A serial-to-parallel converter with associated clock divider is included in the input path, and a parallel-to-serial converter in the output path.

An in-depth guide to the Virtex-5 FPGA IOB is found in the *Virtex-5 FPGA Tri-Mode Ethernet MAC User Guide*.

Configurable Logic Blocks (CLBs)

A Virtex-5 FPGA CLB resource is made up of two slices. Each slice is equivalent and contains:

- Four function generators
- Four storage elements
- Arithmetic logic gates
- Large multiplexers
- Fast carry look-ahead chain

The function generators are configurable as 6-input LUTs or dual-output 5-input LUTs. SLICEMs in some CLBs can be configured to operate as 32-bit shift registers (or 16-bit x 2 shift registers) or as 64-bit distributed RAM. In addition, the four storage elements can be configured as either edge-triggered D-type flip-flops or level sensitive latches. Each CLB has internal fast interconnect and connects to a switch matrix to access general routing resources.

The Virtex-5 FPGA CLBs are further discussed in the *Virtex-5 FPGA User Guide*.

Block RAM

The 36 Kbit true dual-port RAM block resources are programmable from 32K x 1 to 512 x 72, in various depth and width configurations. In addition, each 36-Kbit block can also be configured to operate as two, independent 18-Kbit dual-port RAM blocks.

Each port is totally synchronous and independent, offering three "read-during-write" modes. Block RAM is cascadable to implement large embedded storage blocks. Additionally, back-end pipeline registers, clock control circuitry, built-in FIFO support, ECC, and byte write enable features are also provided as options.

The block RAM feature in Virtex-5 devices is further discussed in the *Virtex-5 FPGA User Guide*.

Global Clocking

The CMTs and global-clock multiplexer buffers provide a complete solution for designing high-speed clock networks.

Each CMT contains two DCMs and one PLL. The DCMs and PLLs can be used independently or extensively cascaded. Up to six CMT blocks are available, providing up to eighteen total clock generator elements.

Each DCM provides familiar clock generation capability. To generate deskewed internal or external clocks, each DCM can be used to eliminate clock distribution delay. The DCM also provides 90°, 180°, and 270° phase-shifted versions of the output clocks. Fine-grained phase shifting offers higher-resolution phase adjustment with fraction of the clock period increments. Flexible frequency synthesis provides a clock output frequency equal to a fractional or integer multiple of the input clock frequency.

To augment the DCM capability, Virtex-5 FPGA CMTs also contain a PLL. This block provides reference clock jitter filtering and further frequency synthesis options.

Virtex-5 devices have 32 global-clock MUX buffers. The clock tree is designed to be differential. Differential clocking helps reduce jitter and duty cycle distortion.

DSP48E Slices

DSP48E slice resources contain a 25 x 18 two's complement multiplier and a 48-bit adder/subtracter/accumulator. Each DSP48E slice also contains extensive cascade capability to efficiently implement high-speed DSP algorithms.

The Virtex-5 FPGA DSP48E slice features are further discussed in *Virtex-5 FPGA XtremeDSP Design Considerations*.

Routing Resources

All components in Virtex-5 devices use the same interconnect scheme and the same access to the global routing matrix. In addition, the CLB-to-CLB routing is designed to offer a complete set of connectivity in as few hops as possible. Timing models are shared, greatly improving the predictability of the performance for high-speed designs.

Boundary Scan

Boundary-Scan instructions and associated data registers support a standard methodology for accessing and configuring Virtex-5 devices, complying with IEEE standards 1149.1 and 1532.

Configuration

Virtex-5 devices are configured by loading the bitstream into internal configuration memory using one of the following modes:

- Slave-serial mode
- Master-serial mode
- Slave SelectMAP mode
- Master SelectMAP mode
- Boundary-Scan mode (IEEE-1532 and -1149)
- SPI mode (Serial Peripheral Interface standard Flash)
- BPI-up/BPI-down modes (Byte-wide Peripheral interface standard x8 or x16 NOR Flash)

In addition, Virtex-5 devices also support the following configuration options:

- 256-bit AES bitstream decryption for IP protection
- Multi-bitstream management (MBM) for cold/warm boot support
- Parallel configuration bus width auto-detection
- Parallel daisy chain
- Configuration CRC and ECC support for the most robust, flexible device integrity checking

Virtex-5 device configuration is further discussed in the *Virtex-5 FPGA Configuration Guide*.

System Monitor

FPGAs are an important building block in high availability/reliability infrastructure. Therefore, there is need to better monitor the on-chip physical environment of the FPGA and its immediate surroundings within the system. For the first time, the Virtex-5 family System Monitor facilitates easier monitoring of the FPGA and its external environment. Every member of the Virtex-5 family contains a System Monitor block. The System Monitor is built around a 10-bit 200kSPS ADC (Analog-to-Digital Converter). This ADC is used to digitize a number of on-chip sensors to provide information about the physical environment within the FPGA. On-chip sensors include a temperature sensor and power supply sensors. Access to the external environment is provided via a number of external analog input channels. These analog inputs are general purpose and can be used to digitize a wide variety of voltage signal types. Support for unipolar, bipolar, and true differential input schemes is provided. There is full access to the on-chip sensors and external channels via the JTAG TAP, allowing the existing JTAG infrastructure on the PC board to be used for analog test and advanced diagnostics during development or after deployment in the field. The System Monitor is fully operational after power up and before configuration of the FPGA. System Monitor does not require an explicit instantiation in a design to gain access to its basic functionality. This allows the System Monitor to be used even at a late stage in the design cycle.

The Virtex-5 FPGA System Monitor is further discussed in the *Virtex-5 FPGA System Monitor User Guide*.

Virtex-5 LXT, SXT, TXT, and FXT Platform Features

This section briefly describes blocks available only in LXT, SXT, TXT, and FXT devices.

Tri-Mode (10/100/1000 Mb/s) Ethernet MACs

Virtex-5 LXT, SXT, TXT, and FXT devices contain up to eight embedded Ethernet MACs, two per Ethernet MAC block. The blocks have the following characteristics:

- Designed to the IEEE 802.3-2002 specification
- UNH-compliance tested
- RGMII/GMII Interface with SelectIO or SGMII interface when used with RocketIO transceivers
- Half or full duplex
- Supports Jumbo frames
- 1000 Base-X PCS/PMA: When used with RocketIO GTP transceiver, can provide complete 1000 Base-X implementation on-chip
- DCR-bus connection to microprocessors

Integrated Endpoint Blocks for PCI Express

Virtex-5 LXT, SXT, TXT, and FXT devices contain up to four integrated Endpoint blocks. These blocks implement Transaction Layer, Data Link Layer, and Physical Layer functions to provide complete PCI Express Endpoint functionality with minimal FPGA logic utilization. The blocks have the following characteristics:

- Compliant with the PCI Express Base Specification 1.1
- Works in conjunction with RocketIO transceivers to provide complete endpoint functionality
- 1, 4, or 8 lane support per block

Virtex-5 LXT and SXT Platform Features

This section briefly describes blocks available only in LXT and SXT devices.

RocketIO GTP Transceivers

4 - 24 channel RocketIO GTP transceivers capable of running 100 Mb/s to 3.75 Gb/s.

- Full clock and data recovery
- 8/16-bit or 10/20-bit datapath support
- Optional 8B/10B or FPGA-based encode/decode
- Integrated FIFO/elastic buffer
- Channel bonding and clock correction support
- Embedded 32-bit CRC generation/checking
- Integrated comma-detect or A1/A2 detection
- Programmable pre-emphasis (AKA transmitter equalization)
- Programmable transmitter output swing
- Programmable receiver equalization
- Programmable receiver termination
- Embedded support for:
  - Out of Band (OOB) signalling: Serial ATA
  - Beaconing, electrical idle, and PCI Express receiver detection
- Built-in PRBS generator/checker

Virtex-5 FPGA RocketIO GTP transceivers are further discussed in the *Virtex-5 FPGA RocketIO GTP Transceiver User Guide*.

Virtex-5 TXT and FXT Platform Features

This section describes blocks only available in TXT and FXT devices.

RocketIO GTX Serial Transceivers (TXT/FXT)

8 - 48 channels RocketIO serial transceivers capable of running 150 Mb/s to 6.5 Gb/s

- Full Clock and Data Recovery
- 8/16/32-bit or 10/20/40-bit datapath support
- Optional 8B/10B encoding, gearbox for programmable 64B/66B or 64B/67B encoding, or FPGA-based encode/decode
- Integrated FIFO/Elastic Buffer
- Channel bonding and clock correction support
- Dual embedded 32-bit CRC generation/checking
- Integrated programmable character detection
- Programmable de-emphasis (AKA transmitter equalization)
- Programmable transmitter output swings
- Programmable receiver equalization
- Programmable receiver termination
- Embedded support for:
  - Serial ATA: Out of Band (OOB) signalling
  - PCI Express: Beaconing, electrical idle, and receiver detection
- Built-in PRBS generator/checker

Virtex-5 FPGA RocketIO GTX transceivers are further discussed in the *Virtex-5 FPGA RocketIO GTX Transceiver User Guide*.

One or Two PowerPC 440 Processor Cores (FXT only)

- Superscalar RISC architecture
- 32-bit Book E compliant
- 7-Stage execution pipeline
- Multiple instructions per cycle
- Out-of-order execution
- Integrated 32 KB Level 1 Instruction Cache and 32KB Level 1 Data Cache (64-way set associative)
- CoreConnect™ Bus Architecture
- Cross-bar connection for optimized processor bandwidth
- PLB Synchronization Logic (Enables non-integer CPU-to-PLB clock ratios)
- Auxiliary Processor Unit (APU) interface with an integrated APU controller
  - Optimized FPGA-based Coprocessor connection
    - Automatic decode of PowerPC floating-point instructions
  - Allows custom instructions
  - Extremely efficient microcontroller-style interfacing

The PowerPC 440 processors are further discussed in the *Embedded Processor Block in Virtex-5 FPGAs Reference Guide*.

Intellectual Property Cores

Xilinx offers IP cores for commonly used complex functions including DSP, bus interfaces, processors, and processor peripherals. Using Xilinx LogiCORE™ products and cores from third party AllianceCORE participants, customers can shorten development time, reduce design risk, and obtain superior performance for their designs. Additionally, the CORE Generator™ system allows customers to implement IP cores into Virtex-5 FPGAs with predictable and repeatable performance. It offers a simple user interface to generate parameter-based cores optimized for our FPGAs.

The System Generator for DSP tool allows system architects to quickly model and implement DSP functions using handcrafted IP and features an interface to third-party system level DSP design tools. System Generator for DSP implements many of the high-performance DSP cores supporting Virtex-5 FPGAs including the Xilinx Forward Error Correction Solution with Interleaver/De-interleaver, Reed-Solomon encoder/decoders, and Viterbi decoders. These are ideal for creating highly-flexible, concatenated codecs to support the communications market.

Using Virtex-5 FPGA RocketIO transceivers, industry leading connectivity and networking IP cores include leading-edge PCI Express, Serial RapidIO, Fibre Channel, and 10 Gb Ethernet cores can be implemented. The Xilinx SPI-4.2 IP core utilizes the Virtex-5 FPGA ChipSync technology to implement dynamic phase alignment for high-performance source-synchronous operation.

Xilinx also provides PCI cores for advanced system-synchronous operation.

The MicroBlaze™ 32-bit processor core provides the industry's fastest soft processing solution for building complex systems for the networking, telecommunication, data communication, embedded, and consumer markets. The MicroBlaze processor features a RISC architecture with Harvard-style separate 32-bit instruction and data buses running at full speed to execute programs and access data from both on-chip and external memory. A standard set of peripherals are also CoreConnect™ enabled to offer MicroBlaze designers compatibility and reuse.

All IP cores for Virtex-5 FPGAs are found on the Xilinx IP Center Internet portal presenting the latest intellectual property cores and reference designs using Smart Search for faster access.

Virtex-5 FPGA LogiCORE Endpoint Block Plus Wrapper for PCI Express

This is the recommended wrapper to configure the integrated Endpoint block for PCI Express delivered through the CORE Generator system. It provides many ease-of-use features and optimal configuration for Endpoint application simplifying the design process and reducing the time-to-market. Access to the core, including bitstream generation capability can be obtained through registration at no extra charge.

Virtex-5 Device and Package Combinations and Maximum I/Os

Table 2: **Virtex-5 Device and Package Combinations and Maximum Available I/Os**

Package	FF323 FFG323 FFV323		FF324 FFG324 FFV324		FF676 FFG676 FFV676		FF1153 FFG1153 FFV1153(1)		FF1760 FFG1760 FFV1760(2)		FF665 FFG665 FFV665		FF1136 FFG1136 FFV1136(3)		FF1156 FFG1156		FF1738 FFG1738 FFV1738(4)		FF1759 FFG1759	
Size (mm)	19 x 19		19 x 19		27 x 27		35 x 35		42.5 x 42.5		27 x 27		35 x 35		35 x 35		42.5 x 42.5		42.5 x 42.5	
Device	GTs	I/O	GTs	I/O	GTs	I/O	GTs	I/O	GTs	I/O	GTs	I/O	GTs	I/O	GTs	I/O	GTs	I/O	GTs	I/O
XC5VLX30			N/A	220	N/A	400														
XC5VLX50			N/A	220	N/A	440	N/A	560												
XC5VLX85					N/A	440	N/A	560												
XC5VLX110					N/A	440	N/A	800	N/A	800										
XC5VLX155							N/A	800	N/A	800										
XC5VLX220									N/A	800										
XC5VLX330									N/A	1,200										
XC5VLX20T	4 GTPs	172																		
XC5VLX30T	4 GTPs	172									8 GTPs	360								
XC5VLX50T											8 GTPs	360	12 GTPs	480						
XC5VLX85T													12 GTPs	480						
XC5VLX110T													16 GTPs	640			16 GTPs	680		
XC5VLX155T													16 GTPs	640			16 GTPs	680		
XC5VLX220T																	16 GTPs	680		
XC5VLX330T																	24 GTPs	960		
XC5VSX35T											8 GTPs	360								
XC5VSX50T											8 GTPs	360	12 GTPs	480						
XC5VSX95T													16 GTPs	640						
XC5VSX240T																	24 GTPs	960		
XC5VTX150T															40 GTXs	360			40 GTXs	680
XC5VTX240T																			48 GTXs	680
XC5VFX30T											8 GTXs	360								
XC5VFX70T											8 GTXs	360	16 GTXs	640						
XC5VFX100T													16 GTXs	640			16 GTXs	680		
XC5VFX130T																	20 GTXs	840		
XC5VFX200T																	24 GTXs	960		

Notes:
1. **FFV1153 package is not available in the LX155 device.**
2. **FFV1760 package is available in the LX110 device only.**
3. **FFV1136 package is not available in the LX155T and FX100T devices.**
4. **FFV1738 package is available in the LX110T device only.**

Virtex-5 FPGA Ordering Information

Virtex-5 FPGA ordering information shown in Figure 1 applies to all packages including Pb-Free.

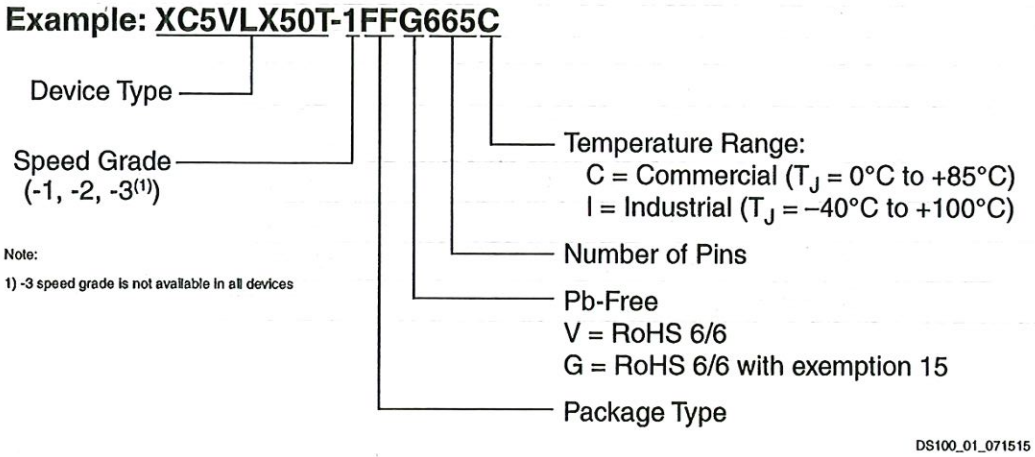


Figure 1: **Virtex-5 FPGA Ordering Information**

Virtex-5 FPGA Documentation

Complete and up-to-date documentation of the Virtex-5 family of FPGAs is available on the Xilinx website. In addition to the most recent *Virtex-5 Family Overview*, the following files are also available for download:

Virtex-5 FPGA Data Sheet: DC and Switching Characteristics (DS202)

This data sheet contains the DC and Switching Characteristic specifications for the Virtex-5 family.

Virtex-5 FPGA User Guide (UG190)

This guide includes chapters on:

- Clocking Resources
- Clock Management Technology (CMT)
- Phase-Locked Loops (PLL)
- Block RAM
- Configurable Logic Blocks (CLBs)
- SelectIO Resources
- SelectIO Logic Resources
- Advanced SelectIO Logic Resources

Virtex-5 FPGA XtremeDSP Design Considerations (UG193)

This guide describes the DSP48E slice and includes reference designs for using DSP48E math functions and various filters.

Virtex-5 FPGA Configuration Guide (UG191)

This all-encompassing configuration guide includes chapters on configuration interfaces (serial and parallel), multi-bitstream management, bitstream encryption, Boundary-Scan and JTAG configuration, and reconfiguration techniques.

Virtex-5 FPGA Packaging and Pinout Specification (UG195)

This specification includes the tables for device/package combinations and maximum I/Os, pin definitions, pinout tables, pinout diagrams, mechanical drawings, and thermal specifications.

Virtex-5 FPGA PCB Designer's Guide (UG203)

This guide provides information on PCB design for Virtex-5 devices, with a focus on strategies for making design decisions at the PCB and interface level.

Virtex-5 FPGA System Monitor User Guide (UG192)

The System Monitor functionality is outlined in this guide.

Virtex-5 FPGA RocketIO GTP Transceiver User Guide (UG196)

This guide describes the RocketIO GTP transceivers available in the Virtex-5 LXT and SXT platforms.

Virtex-5 FPGA RocketIO GTX Transceiver User Guide (UG198)

This guide describes the RocketIO GTX transceivers available in the Virtex-5 TXT and FXT platforms.

Virtex-5 FPGA Tri-Mode Ethernet MAC User Guide (UG194)

This guide describes the dedicated Tri-Mode Ethernet Media Access Controller available in the Virtex-5 LXT, SXT, TXT, and FXT platforms.

Virtex-5 FPGA Integrated Endpoint Block for PCI Express Designs User Guide (UG197)

This guide describes the integrated Endpoint blocks in the Virtex-5 LXT, SXT, TXT, and FXT platforms that are PCI Express compliant.

Embedded Processor Block in Virtex-5 FPGAs Reference Guide (UG200)

This reference guide is a description of the embedded processor block available in the Virtex-5 FXT platform.

RAM Inferencing in Synplify® Software Using Xilinx RAMs

Overview

Many FPGA families provide a mechanism to implement technology-specific RAMs in HDL source code. To take advantage of these optimal RAM implementations, you must manually instantiate the technology-specific RAM cells.

Disadvantages of Instantiation

The following list outlines the disadvantages of instantiating the technology-specific RAM cells.

- The HDL code is no longer technology independent.
- If you use a black box methodology, your synthesis tool might not have access to any timing or area data.

Synplify software version 7.1 addresses these issues by automatically inferring synchronous RAMs directly from your HDL source code. The RTL View of HDL Analyst® then displays the RAM as a simple component, which makes reading the schematic easier. Additionally, the RAM logic is automatically mapped to applicable technology-specific RAM cells. The Synplify software supports synchronous RAMs for Altera, Atmel, Lattice Orca, and Xilinx technology families. This application note specifically covers the RAM inferencing of Xilinx technology families in the Synplify software.

Advantages of Inferencing

RAM inferencing also has the advantages listed below:

- Technology-independent coding style.
- Synplify software provides automatic timing-driven synthesis for RAMs.
- No additional tool dependencies.

The goal for RAM inferencing in the Synplify software is to give you a method that lets you easily specify RAM structures in your HDL source code, while maintaining portability and ensuring that the netlist output after synthesis remains logically correct. Portability across vendors requires that each vendor technology that is mapped has a certain amount of *glue* logic which normally surrounds the technology-specific RAM primitive so that the logic matches the functionality of the specific RAM module in the Synplify HDL-source RAM primitive. Xilinx-specific details regarding *glue* logic are explained in the "Virtex Conflict Resolution" section. The addition of the glue logic can result in a non-optimal RAM implementation. However, if you want a design that most efficiently uses a specific RAM primitive technology, you must instantiate the vendor-specific RAM primitive.

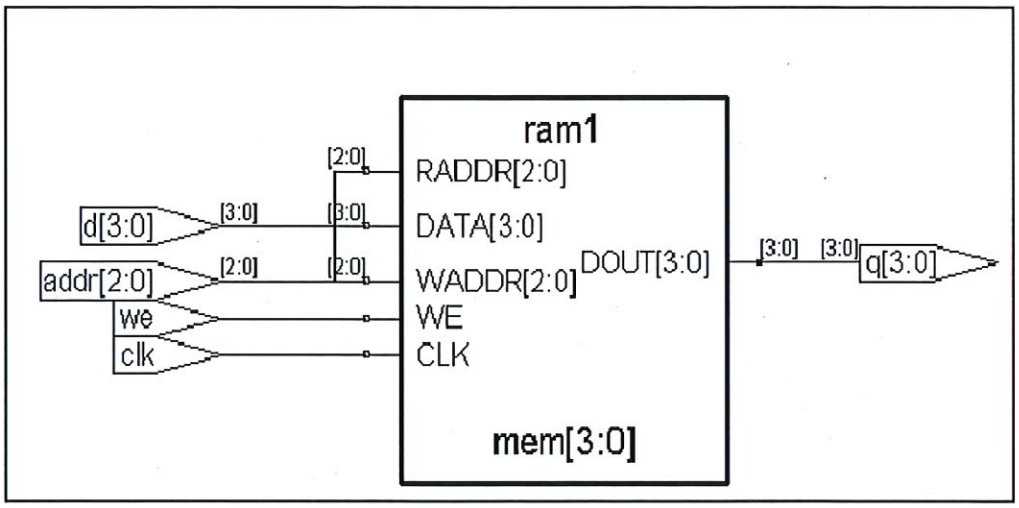


Figure 1: HDL Analyst RTL view of the preceding inferred single-port RAM

Verilog Memory Array

The following code implements a Verilog memory array.

```
module ramtest(z, raddr, d, waddr, we, clk);
output [3:0] z;
input [3:0] d;
input [3:0] raddr, waddr;
input we;
input clk;

reg [3:0] mem [7:0];

assign z = mem[raddr];

always @(posedge clk) begin
if(we) mem[waddr]= d;
end

endmodule
```

NOTE: This will implement RAM in the FPGA fabric (called "distributed RAM")

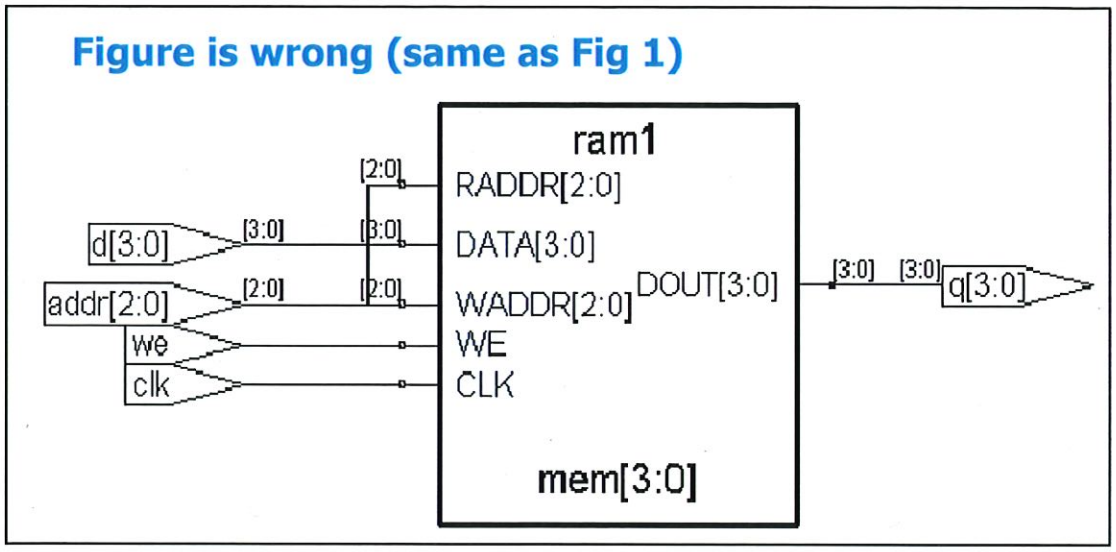


Figure 2: HDL Analyst RTL view of inferred dual -port RAM.

Verilog Code Example of a Dual-Port RAM

The following code illustrates an example of a dual-port RAM.

```
module ram16x8(z, raddr, d, waddr, we, clk);
output [7:0] z;
input [7:0] d;
input [3:0] raddr, waddr;
input we;
input clk;
reg [7:0] z;
reg [7:0] mem0, mem1, mem2, mem3, mem4, mem5, mem6, mem7;
reg [7:0] mem8, mem9, mem10, mem11, mem12, mem13, mem14, mem15;
always @(mem0 or mem1 or mem2 or mem3 or mem4 or mem5 or mem6 or mem7 or
mem8 or mem9 or mem10 or mem11 or mem12 or mem13 or mem14 or mem15 or
raddr)
begin
case (raddr[3:0])
4'b0000: z = mem0;
4'b0001: z = mem1;
4'b0010: z = mem2;
4'b0011: z = mem3;
4'b0100: z = mem4;
4'b0101: z = mem5;
4'b0110: z = mem6;
4'b0111: z = mem7;
4'b1000: z = mem8;
4'b1001: z = mem9;
4'b1010: z = mem10;
4'b1011: z = mem11;
4'b1100: z = mem12;
4'b1101: z = mem13;
4'b1110: z = mem14;
4'b1111: z = mem15;
```

different read/write addr

dual-port RAM allows you to read from one memory location while writing to another memory location

```
endcase
end

always @(posedge clk) begin
if(we) begin
case (waddr[3:0])
4'b0000: mem0 = d;
4'b0001: mem1 = d;
4'b0010: mem2 = d;
4'b0011: mem3 = d;
4'b0100: mem4 = d;
4'b0101: mem5 = d;
4'b0110: mem6 = d;
4'b0111: mem7 = d;
4'b1000: mem8 = d;
4'b1001: mem9 = d;
4'b1010: mem10 = d;
4'b1011: mem11 = d;
4'b1100: mem12 = d;
4'b1101: mem13 = d;
4'b1110: mem14 = d;
4'b1111: mem15 = d;
endcase
end
end
endmodule
```

correct figure

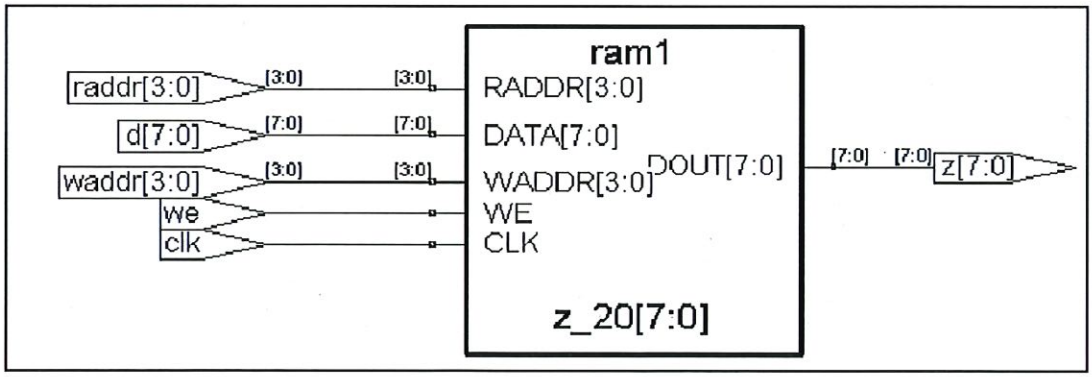


Figure 3: HDL Analyst RTL view of the preceding inferred dual-port RAM.

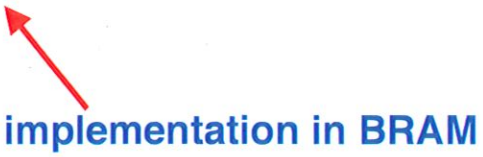
Inferring Block SelectRAMs in Xilinx Technologies

RAM inferencing in the Synplify tool is limited to the coding styles discussed throughout this application note.

Prior to the Synplify 7.0 release, a block SelectRAM could be inferred only if the read address was registered as shown by the following code example.

Verilog Code Example Inferring Single-Port Block SelectRAM

```
module ram_test(q, a, d, we, clk);

output [7:0] q;
input  [7:0] d;
input  [6:0] a;
input  clk, we;

reg [6:0] read_add;

/* The array of an array register ("mem") the RAM will be inferred from.
*/
reg [7:0] mem [127:0] /* synthesis syn_ramstyle = "block_ram */;

assign q = mem[read_add];

always @(posedge clk) begin
if(we)
/* Register RAM Data */
mem[a] <= d;
/* Register Read Address. Basic RAM support does not
require this address register.*/
read_add <= a;
end

endmodule
```



Dual-Port Block SelectRAM with Registered Read Address

When two addresses are used to do the read and the write operation respectively, and the read address is registered, a dual-port block SelectRAM can be inferred as shown by the following example and illustrated in *HDL Analyst Technology view of an inferred Virtex block SelectRAM* on page 9.

Dual-Port Block SelectRAM with Read Address Registered

```
module dualportram(q, a1, a2, d, we, clk, en);
output [7:0] q;
input [7:0] d;
input[6:0] a1;
input[6:0] a2;
input clk, we, en;

reg [6:0] read_addr;
reg[7:0] mem [127:0] /* synthesis syn_ramstyle="block_ram" */;
```

```
assign q = mem[read_addr];
always @(posedge clk) begin
if (we)
mem[a2] <= d;
read_addr <= a1;
end

endmodule
```

different addr

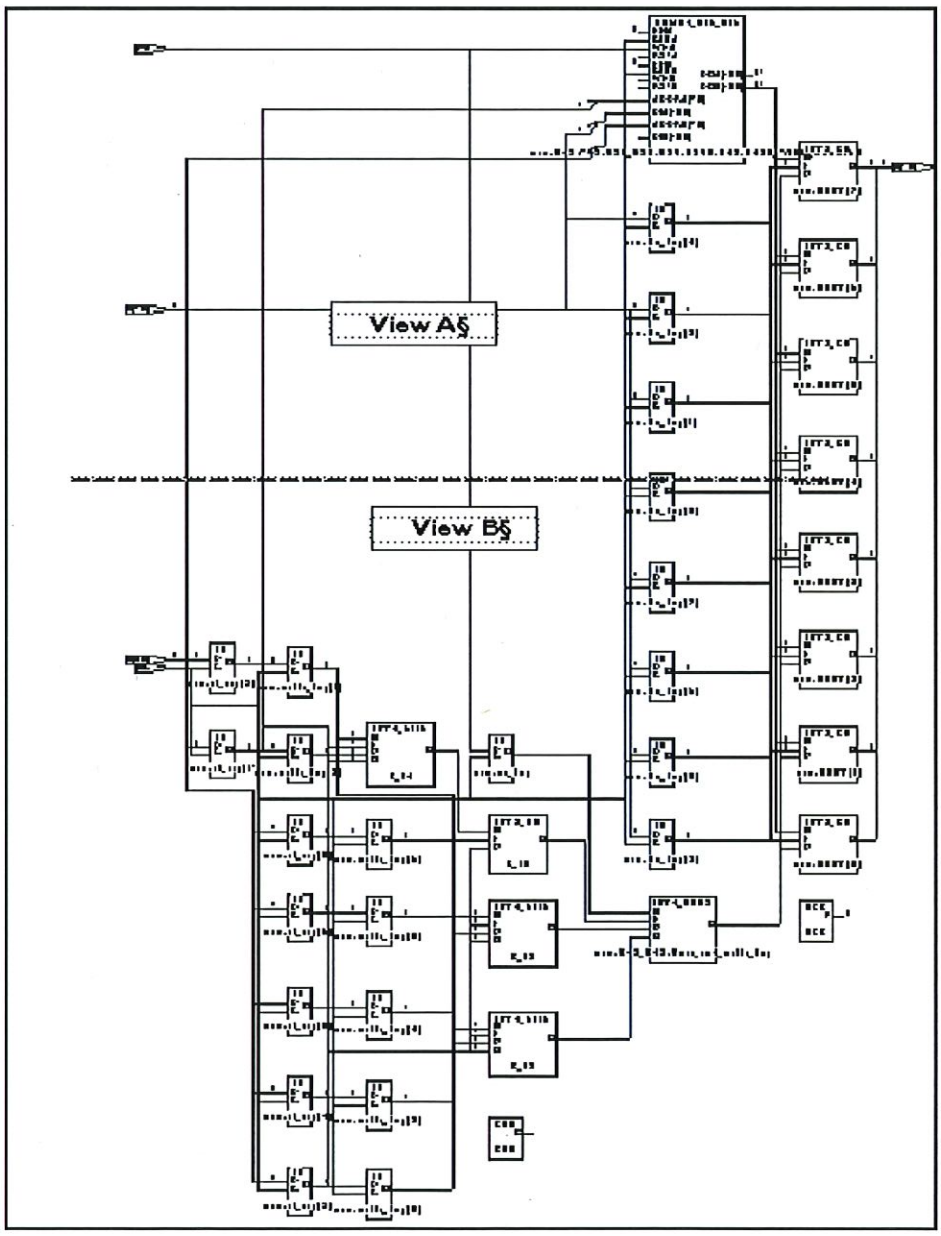


Figure 5: HDL Analyst Technology view of an inferred Virtex block SelectRAM

This figure shows a dual-port RAM inferred by the code in the preceding example, *Dual-Port Block SelectRAM with Registered Read Address* on page 8.

Read Operation

In latch mode, the read operation uses one clock edge. The read address is registered on the read port, and the stored data is loaded into the output latches after the RAM access time. When using the output register, the read operation will take one extra latency cycle to arrive at the output.

Write Operation

A write operation is a single clock-edge operation. The write address is registered on the write port, and the data input is stored in memory.

Write Modes

Three settings of the write mode determines the behavior of the data available on the output latches after a write clock edge: WRITE_FIRST, READ_FIRST, and NO_CHANGE. The Write mode attribute can be individually selected for each port. The default mode is WRITE_FIRST. WRITE_FIRST outputs the newly written data onto the output bus. READ_FIRST outputs the previously stored data while new data is being written. NO_CHANGE maintains the output previously generated by a read operation.

WRITE_FIRST or Transparent Mode (Default)

In WRITE_FIRST mode, the input data is simultaneously written into memory and stored in the data output (transparent write), as shown in Figure 2. These waveforms correspond to latch mode when the optional output pipeline register is not used.

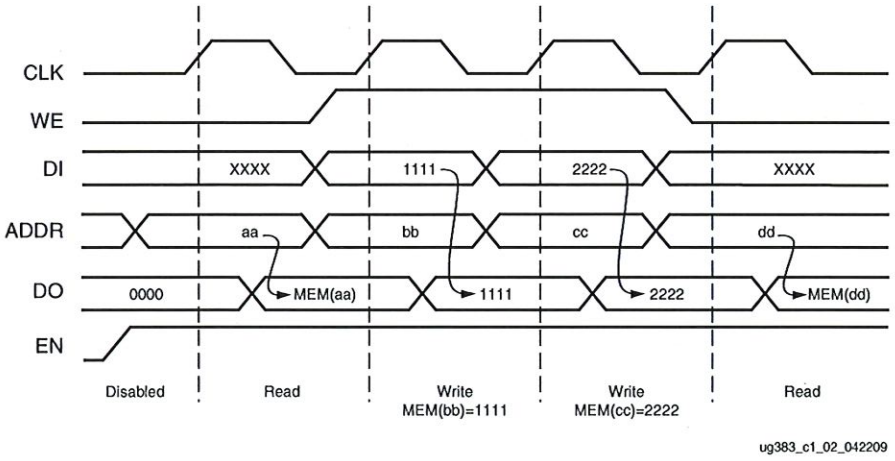


Figure 2: **WRITE_FIRST Mode Waveforms**

READ_FIRST or Read-Before-Write Mode

In READ_FIRST mode, data previously stored at the write address appears on the output latches, while the input data is being stored in memory (read before write). The waveforms in Figure 3 correspond to latch mode when the optional output pipeline register is not used.

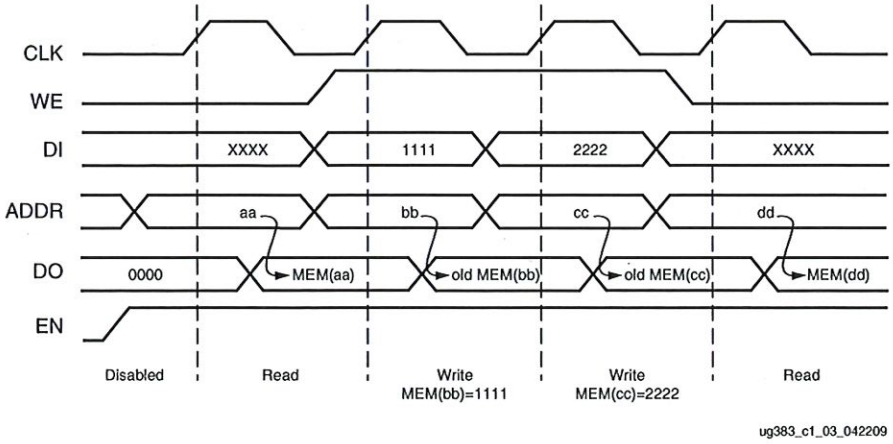


Figure 3: **READ_FIRST Mode Waveforms**

NO_CHANGE Mode

In NO_CHANGE mode, the output latches remain unchanged during a write operation. As shown in Figure 4, data output remains the last read data and is unaffected by a write operation on the same port. These waveforms correspond to latch mode when the optional output pipeline register is not used.

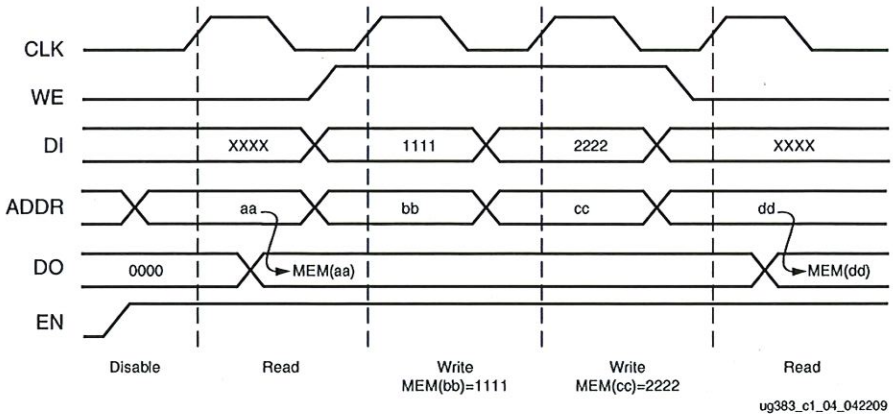


Figure 4: **NO_CHANGE Mode Waveforms**

Block RAM Operating Mode Inference Example

	Verilog

	`// 'write first' or transparent mode`
	`always @(posedge clk) begin`
	`  if(we) begin`
	`  do <= data;`
	`  mem[address] <= data;`
	`  end else`
	`  do <= mem[address];`
	`end`

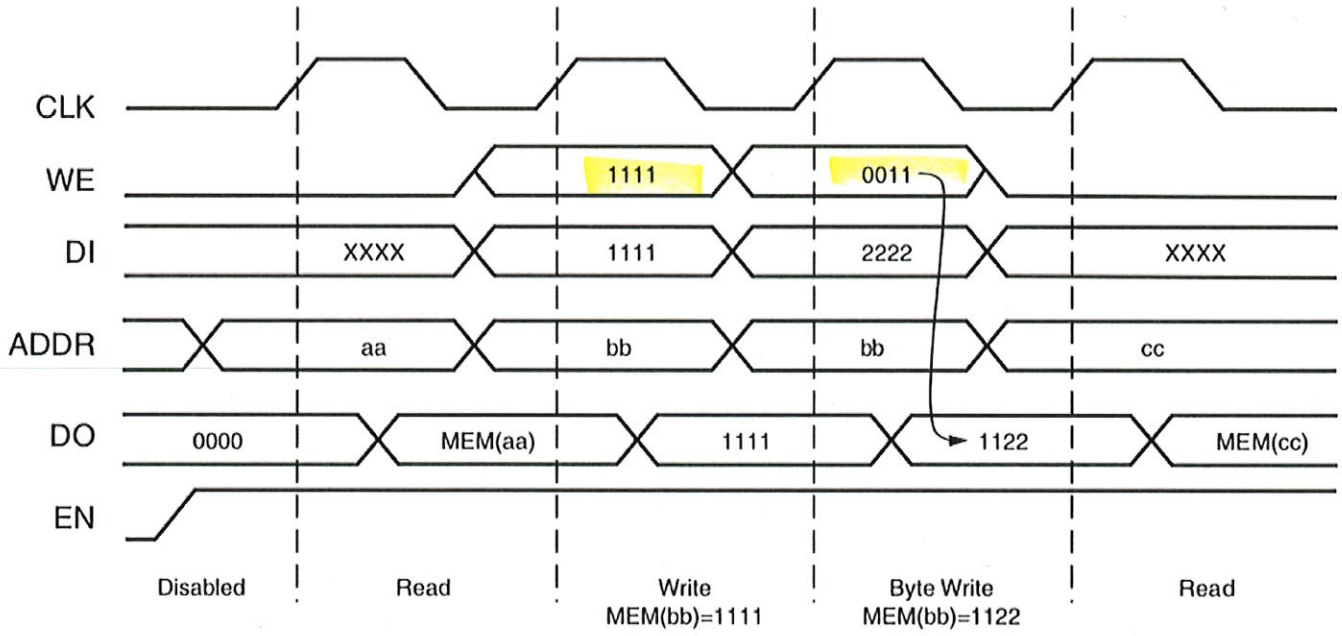
Note non-blocking assignments. Hence, the output databus contains the previous mem[address] contents. 	`// 'read first' or read before write mode(slower)`
	`always @(posedge clk) begin`
	`  if (we)`
	`  mem[address] <= data;`
	`  do <= mem[address];`
	`end`

	`// 'no change' mode`
	`always @(posedge clk)`
	`  if (we)`
	`  mem[address] <= data;`
	`  else`
	`  do <= mem[address];`
	`end`

Byte-wide Write Enable

The byte-wide write enable feature of block RAM gives the capability to write eight bit (one byte) portions of incoming data. There are up to four independent byte-wide write enable inputs to the true dual-port RAM. Each byte-wide write enable is associated with one byte of input data and one parity bit. This feature is useful when using block RAM to interface with a microprocessor.

note: WE is not a 1-bit signal



Byte-wide Write Operation Waveforms

(* register_duplication = "{yes|no}" *)

XCF Syntax Example One

MODEL "*entity_name*" register_duplication={yes|no|true|false};

XCF Syntax Example Two

BEGIN MODEL "*entity_name*"

NET "*signal_name*" register_duplication={yes|no|true|false};

END;

ISE® Design Suite

Process > Process Properties > Xilinx Specific Options > Register Duplication

ROM Extraction (ROM_EXTRACT)

The ROM Extraction (**ROM_EXTRACT**) constraint enables or disables ROM macro inference.

Typically, a ROM can be inferred from a case statement where all assigned contexts are constant values.

Architecture Support

Applies to all FPGA devices. Does not apply to CPLD devices.

Applicable Elements

Applies to the entire design, or to a design element or signal.

Propagation Rules

Applies to the entity, component, module, or signal to which it is attached.

Syntax

-rom_extract {yes|no}

- **yes** (default)
- **no**
- **true** (XCF only)
- **false** (XCF only)

Syntax Examples and Settings

The following syntax examples and settings show how to use this constraint or command line option with particular tools or methods. If a tool or method is not listed, you cannot use this constraint or command line option with it.

VHDL

Declare as follows:

attribute rom_extract: string;

Specify as follows:

attribute rom_extract of {*signal_name*|*entity_name*} : {signal|entity} is "{yes|no}";

Verilog

Place immediately before the module or signal declaration:

(* rom_extract = "{yes|no}" *)

XCF Syntax Example One

MODEL "*entity_name*" rom_extract={yes|no|true|false};

XCF Syntax Example Two

BEGIN MODEL "*entity_name*"

NET "*signal_name*" rom_extract={yes|no|true|false};

END;

XST Command Line

xst run -rom_extract {yes|no}

ISE® Design Suite

Process > Process Properties > HDL Options > ROM Extraction

ROM Style (ROM_STYLE)

The ROM Style (ROM_STYLE) constraint controls the way the macrogenerator implements the inferred ROM macros.

Caution! ROM Extraction (ROM_EXTRACT) must be set to **yes** in order to use ROM Style (ROM_STYLE).

Architecture Support

Applies to all FPGA devices. Does not apply to CPLD devices.

Applicable Elements

Applies to the entire design, or to an entity, component, module, or signal.

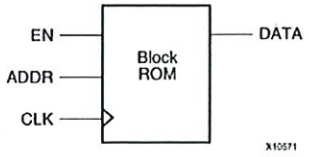
Propagation Rules

Applies to the entity, component, module, or signal to which it is attached.

ROM With Registered Output Verilog Coding Example Two

```
//
// ROMs Using Block RAM Resources.
// Verilog code for a ROM with registered output (template 2)
//

module v_rams_21b (clk, en, addr, data);

    input      clk;
    input      en;
    input      [5:0] addr;
    output reg [19:0] data;
    reg        [19:0] rdata;


    always @(addr) begin
        case(addr)
                6'b000000: rdata <= 20'h0200A;   6'b100000: rdata <= 20'h02222;
                6'b000001: rdata <= 20'h00300;   6'b100001: rdata <= 20'h04001;
                6'b000010: rdata <= 20'h08101;   6'b100010: rdata <= 20'h00342;
                6'b000011: rdata <= 20'h04000;   6'b100011: rdata <= 20'h0232B;
                6'b000100: rdata <= 20'h08601;   6'b100100: rdata <= 20'h00900;
                6'b000101: rdata <= 20'h0233A;   6'b100101: rdata <= 20'h00302;
                6'b000110: rdata <= 20'h00300;   6'b100110: rdata <= 20'h00102;
                6'b000111: rdata <= 20'h08602;   6'b100111: rdata <= 20'h04002;
                6'b001000: rdata <= 20'h02310;   6'b101000: rdata <= 20'h00900;
                6'b001001: rdata <= 20'h0203B;   6'b101001: rdata <= 20'h08201;
                6'b001010: rdata <= 20'h08300;   6'b101010: rdata <= 20'h02023;
                6'b001011: rdata <= 20'h04002;   6'b101011: rdata <= 20'h00303;
                6'b001100: rdata <= 20'h08201;   6'b101100: rdata <= 20'h02433;
                6'b001101: rdata <= 20'h00500;   6'b101101: rdata <= 20'h00301;
                6'b001110: rdata <= 20'h04001;   6'b101110: rdata <= 20'h04004;
                6'b001111: rdata <= 20'h02500;   6'b101111: rdata <= 20'h00301;
                6'b010000: rdata <= 20'h00340;   6'b110000: rdata <= 20'h00102;
                6'b010001: rdata <= 20'h00241;   6'b110001: rdata <= 20'h02137;
                6'b010010: rdata <= 20'h04002;   6'b110010: rdata <= 20'h02036;
                6'b010011: rdata <= 20'h08300;   6'b110011: rdata <= 20'h00301;
                6'b010100: rdata <= 20'h08201;   6'b110100: rdata <= 20'h00102;
                6'b010101: rdata <= 20'h00500;   6'b110101: rdata <= 20'h02237;
                6'b010110: rdata <= 20'h08101;   6'b110110: rdata <= 20'h04004;
                6'b010111: rdata <= 20'h00602;   6'b110111: rdata <= 20'h00304;
                6'b011000: rdata <= 20'h04003;   6'b111000: rdata <= 20'h04040;
                6'b011001: rdata <= 20'h0241E;   6'b111001: rdata <= 20'h02500;
                6'b011010: rdata <= 20'h00301;   6'b111010: rdata <= 20'h02500;
                6'b011011: rdata <= 20'h00102;   6'b111011: rdata <= 20'h02500;
                6'b011100: rdata <= 20'h02122;   6'b111100: rdata <= 20'h0030D;
                6'b011101: rdata <= 20'h02021;   6'b111101: rdata <= 20'h02341;
                6'b011110: rdata <= 20'h00301;   6'b111110: rdata <= 20'h08201;
                6'b011111: rdata <= 20'h00102;   6'b111111: rdata <= 20'h0400D;
        endcase
    end

    always @(posedge clk) begin
        if (en)
            data <= rdata;
    end
endmodule
```

ROM With Registered Address Diagram



ROM With Registered Address Pin Descriptions

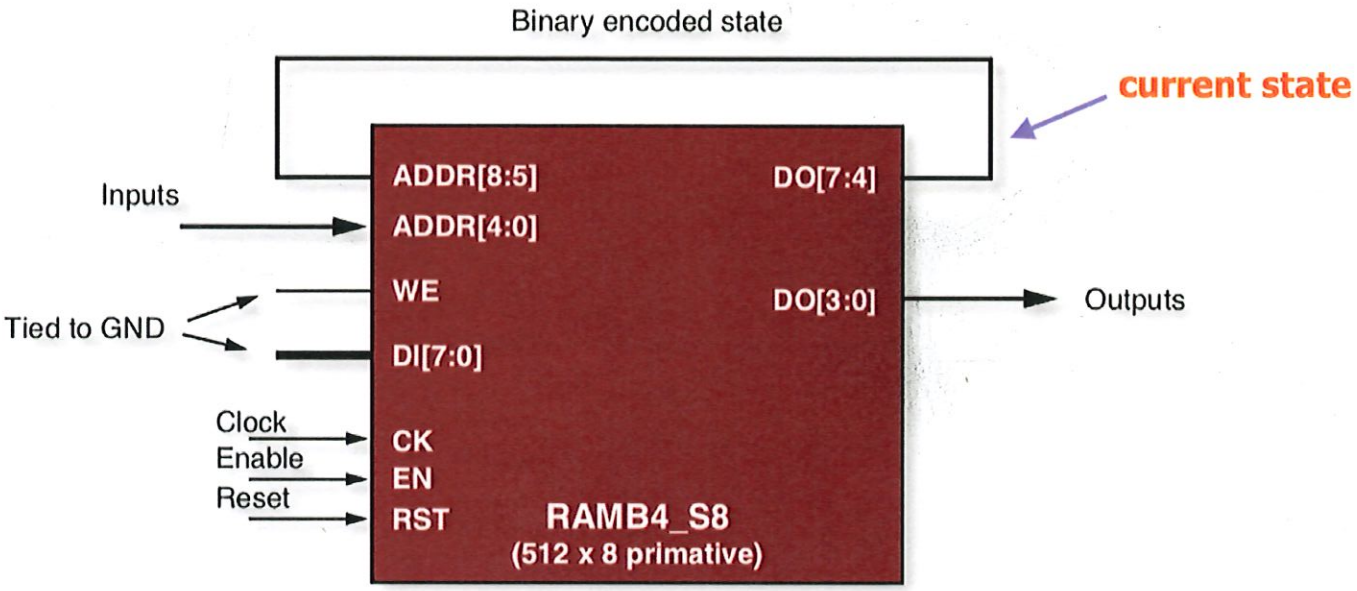
IO Pins	Description

ROM With Registered Output Verilog Coding Example One

```
//
// ROMs Using Block RAM Resources.
// Verilog code for a ROM with registered output (template 1)
//


module v_rams_21a (clk, en, addr, data);

    input       clk;
    input       en;
    input       [5:0] addr;
    output reg [19:0] data;

    always @(posedge clk) begin
        if (en)
            case(addr)
                6'b000000: data <= 20'h0200A;   6'b100000: data <= 20'h02222;
                6'b000001: data <= 20'h00300;   6'b100001: data <= 20'h04001;
                6'b000010: data <= 20'h08101;   6'b100010: data <= 20'h00342;
                6'b000011: data <= 20'h04000;   6'b100011: data <= 20'h0232B;
                6'b000100: data <= 20'h08601;   6'b100100: data <= 20'h00900;
                6'b000101: data <= 20'h0233A;   6'b100101: data <= 20'h00302;
                6'b000110: data <= 20'h00300;   6'b100110: data <= 20'h00102;
                6'b000111: data <= 20'h08602;   6'b100111: data <= 20'h04002;
                6'b001000: data <= 20'h02310;   6'b101000: data <= 20'h00900;
                6'b001001: data <= 20'h0203B;   6'b101001: data <= 20'h08201;
                6'b001010: data <= 20'h08300;   6'b101010: data <= 20'h02023;
                6'b001011: data <= 20'h04002;   6'b101011: data <= 20'h00303;
                6'b001100: data <= 20'h08201;   6'b101100: data <= 20'h02433;
                6'b001101: data <= 20'h00500;   6'b101101: data <= 20'h00301;
                6'b001110: data <= 20'h04001;   6'b101110: data <= 20'h04004;
                6'b001111: data <= 20'h02500;   6'b101111: data <= 20'h00301;
                6'b010000: data <= 20'h00340;   6'b110000: data <= 20'h00102;
                6'b010001: data <= 20'h00241;   6'b110001: data <= 20'h02137;
                6'b010010: data <= 20'h04002;   6'b110010: data <= 20'h02036;
                6'b010011: data <= 20'h08300;   6'b110011: data <= 20'h00301;
                6'b010100: data <= 20'h08201;   6'b110100: data <= 20'h00102;
                6'b010101: data <= 20'h00500;   6'b110101: data <= 20'h02237;
                6'b010110: data <= 20'h08101;   6'b110110: data <= 20'h04004;
                6'b010111: data <= 20'h00602;   6'b110111: data <= 20'h00304;
                6'b011000: data <= 20'h04003;   6'b111000: data <= 20'h04040;
                6'b011001: data <= 20'h0241E;   6'b111001: data <= 20'h02500;
                6'b011010: data <= 20'h00301;   6'b111010: data <= 20'h02500;
                6'b011011: data <= 20'h00102;   6'b111011: data <= 20'h02500;
                6'b011100: data <= 20'h02122;   6'b111100: data <= 20'h0030D;
                6'b011101: data <= 20'h02021;   6'b111101: data <= 20'h02341;
                6'b011110: data <= 20'h00301;   6'b111110: data <= 20'h08201;
                6'b011111: data <= 20'h00102;   6'b111111: data <= 20'h0400D;
            endcase
    end

endmodule
```

Finite State Machines in Block RAM

The idea of using initialized BRAM as ROM suggests FSMs can be implemented in BRAM as well. The idea is shown below:



Using fully synchronous RAM blocks for FSM implementation

Here a BRAM has a 9-bit address bus and an 8-bit data output. 4-bits of the output are a binary encoding of the current state and the remaining 4 outputs are FSM outputs. The EN can be used to activate/deactivate the FSM and the RST to put it into its initial state

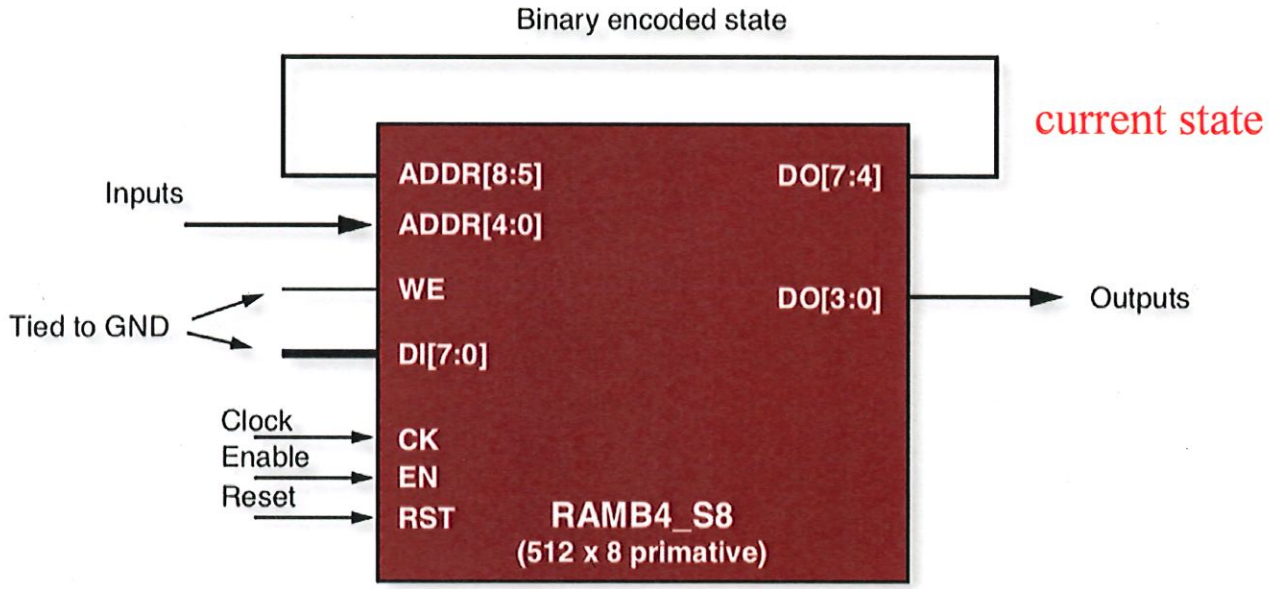
The 9-bit address bus is partitioned into two parts. The 4 MSBs fed back is the current state and the 5 LSBs are FSM inputs.

Thus, the FSM implemented can have up to 16 states, 5 inputs and 4 outputs.

As an example, consider a Moore FSM is currently in state 2 (4'b0010) and the next 5-bit input set is 5'b01001. Then the 9-bit address is 001001001 = 049H. Suppose

$$\text{mem}[001001001] = \text{mem}[049] = \text{D5H}.$$

Then the next state would be DH = 1101 = 13 and the corresponding output would 0101. (No transition or output change until the next clock.)



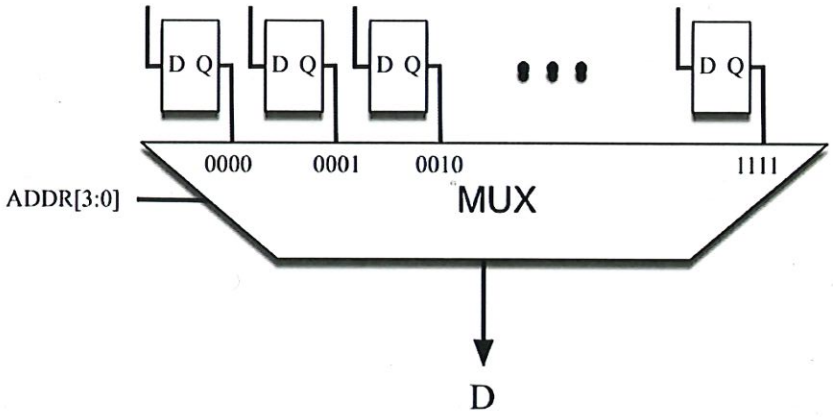
Using fully synchronous RAM blocks for FSM implementation

Shift register LUT

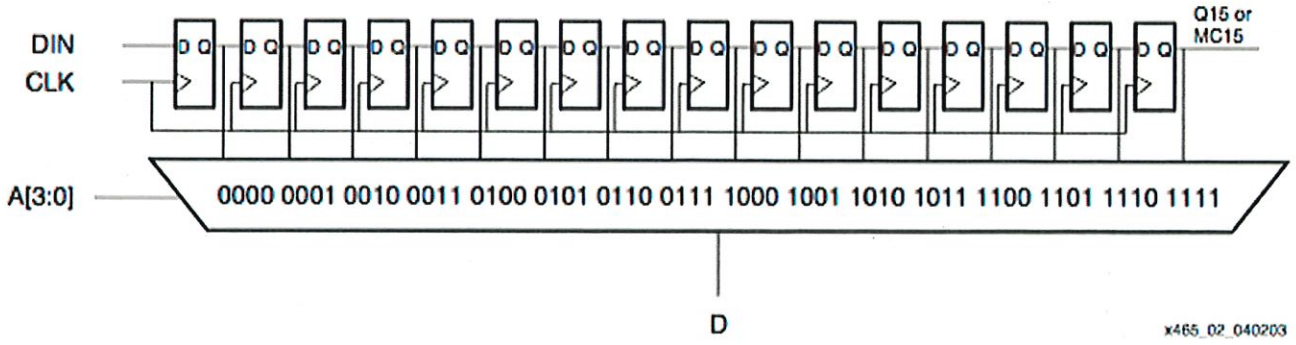
(SRL)

SRL16 Architecture

LUT structure



SRL16 Structure



NOTE: Now when configured $Q \rightarrow D$ connections have been added

Shift Register Inference

When a shift register is described in generic HDL code, synthesis tools infer the use of the SRL16 component. Since the SRL16 does not have either synchronous or asynchronous set or reset inputs, and does not have access to all bits at the same time, using such capabilities precludes the use of the SRL16, and the function is implemented in flip-flops. The cascadable shift register (SRLC16) may be inferred if the shift register is larger than 16 bits or if only the Q15 is used.

In fact, adding a reset is one way to force a synthesis tool to use flip-flops instead of the SRL16 when flip-flops are preferred for performance or other reasons. If a reset is not needed, simply connect a dummy signal and use an appropriate KEEP attribute to prevent the synthesis tool from optimizing it out of the design.

Although the SRL16 shift register does not have a parallel load capability, an equivalent function can be implemented simply by anticipating the load requirement and shifting in the proper data. This requires predictable timing for the load command.

Verilog Inference Code

The following code infers an SRL16 in Verilog.

```
always @ (posedge C)
begin
  Q_INT <= {Q_INT[14:0],D};
end

always @(Q_INT)
begin

 Q <= Q_INT[15];
end
```

NOTE: The 2nd always statement is needed because Q[15] is the output of the SRL. "Q_INT" is only a temporary variable.