Verilog – Module 1 Introduction

#### Jim Duckworth

ECE Department, WPI

Jim Duckworth, WPI

Module 1

# Topics

- Background to VHDL
- Introduction to language
- Programmable Logic Devices
  - CPLDs and FPGAs
  - FPGA architecture
  - Nexys 2 Board
- Using VHDL to synthesize and implement a design
- Verilog overview

# Hardware Description Languages

- Example HDL's : ABEL, VERILOG, VHDL
- Advantages:
  - Documentation
  - Flexibility (easier to make design changes or mods)
  - Portability (if HDL is standard)
  - One language for modeling, simulation (test benches), and synthesis
  - Let synthesis worry about gate generation
    - Engineer productivity
- However: A different way of approaching design
  - engineers are used to thinking and designing using graphics (schematics) instead of text.

Jim Duckworth, WPI

# Verilog background

- 1983: Gateway Design Automation released Verilog HDL "Verilog" and simulator
- 1985: Verilog enhanced version "Verilog-XL"
- 1987: Verilog-XL becoming more popular (same year VHDL released as IEEE standard)
- 1989: Cadence bought Gateway
- 1995: Verilog adopted by IEEE as standard 1364
  - Verilog HDL, Verilog 1995
- 2001: First major revision (cleanup and enhancements)
  - Standard 1364-2001 (or Verilog 2001)
- System Verilog under development
  - Better system simulation and verification support

# Books

- "FPGA Prototyping by Verilog Examples", 2008, Pong P. Chu, Wiley 978-0-470-18532-2
- "Starters Guide to Verilog 2001" by Ciletti, 2004, Prentice Hall 0-13-141556-5
- "Fundamentals of Digital Logic with Verilog Design" by Brown and Vranesic, 2003, McGraw-Hill, 0-07-282878-7
- "Advanced Digital Design with the Verilog HDL", by Ciletti, 2003, Prentice-Hall, 0-13-089161-4
- "HDL Chip Design" by Smith, 1996, Doone Publications, 0-9651934 8
- "Verilog Styles for Synthesis of Digital Systems" by Smith and Franzon, 2000, Prentice Hall, 0-201-61860-5
- "Verilog HDL" by Palnitkar", 2003, Prentice Hall, 0-13-044911-3
- "Verilog for Digital Design" by Vhadi and Lysecky, 2007, Wiley, 978-0-470-05262-4

Jim Duckworth, WPI

# Create Verilog Module

| Image: Second state diagram         Image: State diagram | Eile Name:<br>mux<br>Logation:<br>c:\verilog\verilog_examples | <b>Define Verilog Source</b><br>Module Name mu | ХХ                            |        |     | × |
|---|---|--|-------------------------------|--------|-----|---|
| VHDL Package  |   | Port Name                                      | Direction                     | MSB    | LSB |   |
| M VHDL Test Bench   |   | а  | input                         |        |     |   |
|   |   | Ь  | input                         |        |     |   |
|   |   | sel  | input                         |        |     |   |
|   |   | У  | output                        |        |     |   |
|   |   |  | input                         |        |     |   |
|   | Add to project  |  | input                         |        |     |   |
|   |   |  | input                         |        |     |   |
|   |   |  | input                         |        |     |   |
| Z Back Ne   | vt Cancel Help  |  | input                         |        |     |   |
| ( Dapy The  |   |  | input                         |        |     |   |
|   |   |  | input                         |        |     | _ |
|   |   |  | Input                         |        |     |   |
|   |   |  | Input                         |        |     |   |
|   |   |  | ( <u>B</u> ack <u>N</u> ext > | Cancel | Hel | p |

# Module Created

- No separate entity and arch just module
- Ports can be input, output, or inout
- Note: Verilog 2001 has alternative port style:
  - (input a, b, sel, output y);
  - Also place in column:
  - (
    - input a,
  - input b,
    input sel,
  - output y
  - );



### Add assign statement

- Similar to VHDL conditional signal assignment continuous assignment
- Same hardware produced as with VHDL



### Verilog - general comments

- VHDL is like ADA and Pascal in style
  - Strongly typed more robust than Verilog
  - In Verilog it is easier to make mistakes
    - Watch for signals of different widths
    - No default required for case statement, etc
- Verilog is more like the 'c' language
- Verilog IS case sensitive
- White space is OK
- Statements terminated with semicolon (;)
- Verilog statements between
  - module and endmodule
- Comments // single line and /\* and \*/

#### Jim Duckworth, WPI

# Verilog

- Four-value logic system
  - 0 logic zero, or false condition
  - $1 \log ic 1$ , or true condition
  - x, X unknown logic value
  - z, Z high-impedance state
- Number formats
  - **b**, **B** binary
  - d, D decimal (default)
  - h, H hexadecimal
  - o, O octal
- 16'H789A 16-bit number in hex format
- 1'b0 1-bit

#### Example Synthesis Results (not Xilinx)



# Programmable Logic Devices

- Xilinx user programmable devices
  - FPGAs Field Programmable Gate Array
    - Virtex 4, 5, 6, 7 !
    - Spartan 3, Spartan 6
    - Consist of configurable logic blocks
      - Provides look-up tables to implement logic
      - Storage devices to implement flip-flops and latches
  - CPLDs Complex Programmable Logic Devices
    - CoolRunner-II CPLDS (1.8 and 3.3 volt devices)
    - XC9500 Series (3.3 and 5 volt devices)
    - Consist of macrocells that contain programmable and-or matrix with flip-flops
- Altera has a similar range of devices

#### Electronic Components (Xilinx)



#### **Acronyms**

SPLD = Simple Prog. Logic Device PAL = Prog. Array of Logic CPLD = Complex PLD FPGA = Field Prog. Gate Array

#### **Common Resources**

Configurable Logic Blocks (CLB)

- Memory Look-Up Table
  - AND-OR planes
- Simple gates
- Input / Output Blocks (IOB)

- Bidirectional, latches, inverters, pullup/pulldowns

#### Interconnect or Routing

- Local, internal feedback, and global

#### Xilinx Products (Xilinx)



#### Overview of Xilinx FPGA Architecture (Xilinx)



# Spartan-3 FPGA Family

- "Designed to meet the needs of high-volume, costsensitive consumer electronic applications"
- 326 MHz system clock rate
- Programmed by loading configuration data into static memory cells – place serial PROM on board

| Device   | System<br>Gates | CLBs<br>(4 slices) | CLB<br>flip-flops | Block<br>Ram (bits) | User<br>IO | Price<br>(250K) |
|----------|-----------------|--------------------|-------------------|---------------------|------------|-----------------|
| XC3S200  | 200K            | 480                | 3,840             | 216K                | 173        | \$2.95          |
| XC3S1000 | 1M              | 1,920              | 15,360            | 432K                | 391        | \$12            |
| XC3S4000 | 4M              | 6,912              | 55,296            | 1,728K              | 712        | \$100           |

# Spartan-3E FPGA Family

- Also "specifically designed to meet the needs of high volume, costsensitive consumer electronic applications.
- builds on the success of the earlier Spartan-3 family by increasing the amount of logic per I/O, significantly reducing the cost per logic cell. New features improve system performance.
- Because of their exceptionally low cost, are ideally suited to a wide range of consumer electronics applications, including broadband access, home networking, display/projection, and digital television equipment".

| Device    | System<br>Gates | CLBs<br>(4 slices) | CLB<br>flip-flops | Block<br>Ram (bits) | User<br>IO | Price<br>(250K) |
|-----------|-----------------|--------------------|-------------------|---------------------|------------|-----------------|
| XC3S100E  | 100K            | 240                | 1,920             | 72K                 | 108        | <\$2            |
| XC3S500E  | 500K            | 1,164              | 9,312             | 360K                | 232        | \$25 1-off      |
| XC3S1600E | 1.6M            | 3,688              | 29,504            | 648K                | 376        | <\$9            |

# Spartan-6 FPGA Family

- "Delivers an optimal balance of low risk, low cost, and low power for cost-sensitive applications"
- Offers advanced power management technology, up to 150K logic cells, PCI express blocks, memory support, DSP slices, and transceivers
- LX and LXT (Integrated transceivers and PCI Express Endpoint block) devices
- DSP48A slice contains 18 by 18 multiplier, adder and accumulator

| Device   | CLBs<br>slices | CLB<br>flip-flops | DSP48A<br>slices | Block<br>Ram<br>(18 Kb) | User<br>IO | Price<br>(1 off) |
|----------|----------------|-------------------|------------------|-------------------------|------------|------------------|
| XC6LX4   | 600            | 4,800             | 8                | 12                      | 132        | \$10             |
| XC6LX16  | 2,278          | 18,224            | 32               | 32                      | 232        | \$30             |
| XC6LX150 | 23,038         | 184,304           | 180              | 268                     | 576        | \$180            |

# Programmable Functional Elements

- Configurable Logic Blocks (CLBs)
  - RAM-based look-up tables to implement logic
  - Storage elements for flip-flops or latches
- Input/Output Blocks
  - Supports bidirectional data flow and 3-state operation
  - Supports different signal standards including LVDS
  - Double-data rate registers included
  - Digitally controlled impedance provides on-chip terminations
- Block RAM provides data storage
  - 18-Kbit dual-port blocks
- Multiplier blocks (accepts two 18-bit binary numbers)
- Digital Clock Manager (DCM)
  - Provides distribution, delaying, mult, div, phase shift of clocks

# Slices and CLBs (Xilinx)

- Each Virtex<sup>™</sup>-II CLB contains four slices
  - Local routing provides feedback between slices in the same CLB, and it provides routing to neighboring CLBs
  - A switch matrix provides access
    - to general routing resources



# Simplified Slice Structure (Xilinx)

- Each slice has four outputs
  - Two registered outputs, two non-registered outputs
  - Two BUFTs associated with each CLB, accessible by all 16 CLB outputs
- Carry logic runs vertically, up only
  - Two independent carry chains per CLB



### Detailed Slice Structure (Xilinx)

- The next slides will discuss the slice features
  - LUTs
  - MUXF5, MUXF6, MUXF7, MUXF8 (only the F5 and F6 MUX are shown in the diagram)
  - Carry Logic
  - MULT\_ANDs
  - Sequential Elements



# Look-Up Tables (Xilinx)

- Combinatorial logic is stored in Look-Up Tables (LUTs)
  - Also called Function Generators (FGs)
  - Capacity is limited by number of inputs,
    - not complexity
- Delay through the LUT is constant





# Flexible Sequential Elements (Xilinx)

- Can be flip-flops or latches
- Two in each slice; eight in each CLB
- Inputs can come from LUTs or from an independent CLB input
- Separate set and reset controls
  - Can be synchronous or asynchronous
- All controls are shared within a slice
  - Control signals can be inverted locally within a slice





# IOB Element (Xilinx)

- Input path
  - Two DDR registers
- Output path
  - Two DDR registers
  - Two 3-state enable DDR registers
- Separate clocks and clock enables for I and O
- Set and reset signals are shared



# SelectIO Standard (Xilinx)

- Allows direct connections to external signals of varied voltages and thresholds
  - Optimizes the speed/noise tradeoff
  - Saves having to place interface components onto your board
- Differential signaling standards
  - LVDS, BLVDS, ULVDS
  - LDT
  - LVPECL
- Single-ended I/O standards
  - LVTTL, LVCMOS (3.3V, 2.5V, 1.8V, and 1.5V)
  - PCI-X at 133 MHz, PCI (3.3V at 33 MHz and 66 MHz)
  - GTL, GTLP
  - and more!

# Digital Controlled Impedance (DCI)

- DCI provides
  - Output drivers that match the impedance of the traces
  - On-chip termination for receivers and transmitters
- DCI advantages
  - Improves signal integrity by eliminating stub reflections
  - Reduces board routing complexity and component count by eliminating external resistors
  - Internal feedback circuit eliminates the effects of temperature, voltage, and process variations

## Block SelectRAM Resources (Xilinx)

- Up to 3.5 Mb of RAM in 18kb blocks
  - Synchronous read and write
- True dual-port memory
  - Each port has synchronous read and write capability
  - Different clocks for each port
- Supports initial values
- Synchronous reset on output latches
- Supports parity bits
  - One parity bit per eight data bits



D\$031\_11\_102000

# Dedicated Multiplier Blocks (Xilinx)

- 18-bit twos complement signed operation
- Optimized to implement multiply and accumulate functions
- Multipliers are physically located next to block SelectRAM<sup>TM</sup> memory



#### Nexys2 Board (\$99)



#### Nexys3 Board (\$119)



# Logic Synthesis

• A process which takes a digital circuit description and translates it into a gate level design, optimized for a particular implementation technology.



# Xilinx Design Process (Xilinx)



#### Jim Duckworth, WPI



# Program the FPGA (Xilinx)



#### Decoder Tutorial Demo Example



# Verilog Source Code

| 1                 | SE Text Edi      | tor (P.15xf) - [decoder.v*]                              |          | ×      |   |
|-------------------|------------------|--|----------|--------|---|
|                   | ile <u>E</u> dit | <u>V</u> iew <u>W</u> indow La <u>v</u> out <u>H</u> elp |          | - 5    | × |
| 6                 |                  | 5  |          |        |   |
| ₹≣                | 18               | // Additional Comments:                                  |          |        | ~ |
|                   | 19               | //   |          |        |   |
| -                 | 20               | ///////////////////////////////////////                  | //////   | 111.   |   |
|                   | 21               | module decoder(  |          |        |   |
| 12                | 22               | input [2:0] sw,  |          |        |   |
|                   | 23               | output [7:0] led   |          |        |   |
| =                 | 24               | );   |          |        |   |
| 2                 | 25               |  |          |        |   |
|                   | 26               | assign led = (sw == 3'b000) ? 8'b00000001 :              |          |        |   |
| -                 | 27               | (sw == 3'b001) ? 8'b0000010 :                            |          |        |   |
| 24                | 28               | (sw == 3'b010) ? 8'b0000100 :                            |          |        |   |
| 34                | 29               | (sw == 3'b011) ? 8'b00001000 :                           |          |        |   |
|                   | 30               | (sw == 3.0100) ? 8.00010000 :                            |          |        |   |
| 20-               | 31               | (3W == 3.5101) + 8.500100000 :                           |          |        | Ξ |
| 0                 | 32               | (SW == 3.DII0) / 8.DUI000000 :                           |          |        |   |
|                   | 33               | 8. B1000000;   |          |        |   |
| $\mathbf{\Theta}$ | 25               | endmodule  |          |        |   |
|                   | 35               | endilodate   |          |        |   |
|                   | 50               |  |          |        | Ŧ |
|                   | ۰ III            |  |          | ÷.     |   |
|                   |                  | decoder.v*   |          |        |   |
|                   |                  | Ln1  | Col 1 Ve | erilog |   |

# Synthesizing the Design

| <pre>identify the implementation imp</pre>   | Sources ×  | EPGA Design Summary                     |  | decoder Project Status           | (08/29/2008 - 11:58-22)                 |                           |            |
|--|--|---|--|----------------------------------|---|---------------------------|------------|
| <ul> <li>Somery</li> <li>Somery</li> <li>Somery</li> <li>Somery</li> <li>Model Levellison</li> <li>Model Secondary</li> <li>Model Secondary</li> <li>Model Secondary</li> <li>Model Secondary</li> <li>Model Levellison</li> <li>Model Secondary</li> <li>Model Condelison</li> <li>Model Secondary</li> <li>Model Condelison&lt;</li></ul>  | Sources for: Implementation 💌  | 🖶 Design Overview                       | Project File                                 | decader ire                      | Current State:                          | Programming File General  | ted        |
| Control of the second sec  | 🔄 decoder  | - 🖹 Summary                             | Module Name:                                 | decoder                          | • Errors:                               | No Error                  |            |
|  | xc3s500e-4fg320  | - BIOB Properties                       | Terest Device:                               |                                  | - Lindia.                               | Ma Marina                 |            |
| Concern Verlander Verlander:     Concern Verlander:     Concer   | Do decoder - Behavioral (decoder.vhd)  | Module Level Utilization                | Target Device.                               | 105 40 4 00 5 Hz                 | warnings.                               | nu wanings                |            |
| ter source i free is source in the form of the source is a former of   | · [s] decoder.uct (decoder.uct)  | Timing Constraints                      | Product Version:                             | SE TUTTU2 - Foundation Simulator | Houting Results:                        | Al Signais Compretely Hou | 100        |
| <pre>     Success Up for a Second Look      Second Look</pre>   |  | Princut Heport                          | Design Goal:                                 | Jalanced                         | Timing Constraints:                     |                           |            |
| Number of warrings in grant   Add favioral   Add favioral   Add favioral   Add favioral   Over Despisione   Destinition favioral   Destinition fa   | 🗝 Sources 🜔 Files 🛛 👩 Snapshot 🌔 Librarie:   | Uock Heport                             | Design Strategy:                             | Kilinx Default (unlocked)        | <ul> <li>Final Timing Score:</li> </ul> | 0 (Timing Report)         |            |
| Concerne bite decoder. Behaviord     Concerne bite decoder.     Co   | Processes X  | Ellois and warrings                     |  |                                  |   |                           |            |
| Add failing Source Image Add failing Source   Add failing Source Image Add failing Source   Deart Has Source Image Add failing Source   Deart Add failing Source Image Add failing   | Processes for decoder - Rehavioral   | Translation Messages                    |  | decoder Partition Sun            | mary                                    |                           |            |
| Control New Joace       Device Utilization Summary       Image of the Strange of the Strang   | Add Evidine Carees   | Map Messages                            | No partition information was found.          |                                  |   |                           |            |
| Construction       Device Utilization Summary       Image Metages<br>(a) Construction       Device Utilization Summary       Image Metages<br>(b) Construction       Image Metages<br>(b) Construction       Construction       Utilization       Number of Secondaria       Utilization       Number of Secondaria   | Create New Source  | Place and Route Messages                |  |                                  |   |                           |            |
| Image Disputibility       Used       Available       Utilization       Note         User Disputibility       Desp Tails       0  | View Design Summary  | Timing Messages                         |  | Device Utilization Su            | mmary                                   |                           |            |
| Image Constraints:   | 🗑 😼 Design Utilities   | - 📑 Bitgen Messages                     | Logic Utilization                            | Used                             | Available                               | Utilization               | Note(s     |
| Process Trigo Constants:       0<  | 🖃 😼 User Constraints   | Al Current Messages                     | Number of 4 input LUTs                       | 8                                | 9,312                                   | 1%                        |            |
| Proceeds no - Pre-symbolics By symbolics Report   Proceeds no - Pre-symbolics Mathed of coccept Sites:   Proceeds no - Pre-symbolics Pre-stable Sites:   Pre-stable Sites: Pre-stable Sites:   Pro-stable Sites: Pre-stable Sites:   Pre-stable Sites: Pre-stable Sites:   Pre-stable Site: Pre-stable Sites:   Pre-stable Site: Pre-stable Site:   Pre-stable Site: Pre-stable Site: <t< td=""><td>Create Timing Constraints</td><td>Detailed Reports</td><td>Logic Distribution</td><td></td><td></td><td></td><td></td></t<>  | Create Timing Constraints  | Detailed Reports                        | Logic Distribution                           |                                  |   |                           |            |
| Image: Decoder Area //O Lings: Postsym       Image: Postsym       Image: Decoder Area //O Lings: Postsym       Image: Decoder Area //O Lings: Postsym       Image: Postsym       Imag  | Floorplan IO - Pre-Synthesis   | Synthesis Report                        | Number of occupied Slices                    | 4                                | 4.650                                   | 12                        |            |
| i Construint of Start Contrarting unided Start Contrart Contrecont Contrecont Contrart Contrart Contrart Contrart C   | Floorplan Area / 10 / Logic - Post-Synt  | 🔚 Translation Report 🛛 💌                | Number of Occupies SIGES                     | 4                                | 4,030                                   | 1/6                       |            |
| a Contracts of Second   | 🖶 🍋 Synthesize - XST   | Project Properties                      | womeet or bildes containing only felated log | <i>p</i> 4                       | 4                                       | 100%                      |            |
| a W Compares Programmer Resident Heads <ul> <li></li></ul>   | Complement Design  | Enable Enhanced Design Summary          | Number of Sinces containing unrelated logic  | 0                                | 4                                       | 0%                        |            |
| iii Contigues Taget Device       Performance Summary Control         iii Sono Floring Constraints       Iiii Sono Floring Constraints         iii Sono Floring Constraints       Iiiiii Sono Floring Constraints         iiii Sono Floring Constraints       Iiiiiiii Sono Floring Constraints         iiiiiii Constraints       Iiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiii   | Generate Programming File  | Display Incremental Messager            | Total Number of 4 input LUTs                 | 8                                | 9,312                                   | 1%                        |            |
|  | Configure Target Device  | Enhanced Design Summary Contents        | Number of bonded IOBs                        |                                  |   |                           |            |
| Show Ends<br>Show Ends |  | Show Partition Data                     | Number of bonded                             | 11                               | 232                                     | 4%                        |            |
| Understand     Description       Image: Source Sour  |  | - Show Errors                           |  |                                  |   |                           |            |
| Find Limits Score:     0     Provide:     Pr  |  | - U Show Warnings                       |  | Performance Summ                 | aary                                    |                           |            |
| Round Section     Round Section     Additional Completed Readed     Clock Date     Octo Reg       ***     What's New inits Design Sule 101     ***     ************************************  |  | Show Clock Report                       | Final Timing Score:                          | 0                                | Pinout Data:                            |                           | Pinout Rep |
| Taming Constraint:     Taming Constraint:     What's New in ISE Design Suite 101     Design Summay     Cencerating Report     Mumber of warnings: 0   Total Line: 2 sees   Process "Generate Post-Place & Route Static Timing" completed successfully Statted: "Generate Programming File". Process "Generate Progr   |  | and other deservice point               | Routing Results:                             | All Signals Completely Routed    | Clock Data:                             |                           | Clock Repr |
| C Concest "Generate Programming File". Process   |  |   | Timing Constraints:                          |                                  |   |                           |            |
| Image: Concept of Concep  |  |   |  |                                  |   |                           | <u></u>    |
| Image: Terminal Status       What's New wild's Design Submit 10       Image: Design Submit 10       Image: Design Submit 10       Image: Design Submit 10         Generating Report       Number of Warnings: 0       Total time: 2 secs       Total time: 2 secs         Process "Generate Programming File".       Process "Generate Programming File" completed successfully         Fraces "Generate Programming File" completed successfully         Image: Consult @ Encore Memory in Total time: Image: Im   | S# Deserves  | ·                                       |  | Datailail Danaste                | -                                       |                           |            |
| Generating Report<br>Number of Warnings: 0<br>Total Like: 2 secs<br>Process "Generate Post-Place & Route Static Timing" completed successfully<br>Started : "Generate Programming File".<br>Process "Generate Programming File" completed successfully<br>Concess "Generate Programming File".<br>Process "Generate Programming File".<br>Process "Generate Programming File".   | The second secon | What's New in ISE Design Suite 10.1 🛛 😰 | Design Summary 📉 decoder.vhd 🕅               | decoder.ucf                      |   |                           |            |
| Number of Varings: 0<br>Total time: 2 secs<br>Process "Generate Programming File".<br>Process "Generate Programming File" completed successfully<br>Encounder Completed successfully<br>Encounder Completed Successfully   | X Generating Report  |   |  |                                  |   |                           |            |
| Number of warnings: 0<br>Total time: 2 secs<br>Process "Generate Post-Place & Route Static Timing" completed successfully<br>Statted : "Generate Programming File".<br>Process "Generate Programming File" completed successfully<br>Concess "Generate Programming File" completed successfully  | interesting report in  |   |  |                                  |   |                           |            |
| Total time: 2 secs<br>Process "Generate Programming File".<br>Process "Generate Programming File" completed successfully<br>Concess "Generate Programming File" completed successfully<br>Concess @ Encore & Warning @ TotShel @ Fridn Files   | Number of warnings: 0  |   |  |                                  |   |                           |            |
| Process "Generate Post-Place & Route Static Timing" completed successfully<br>Started : "Generate Programming File".<br>Process "Generate Programming File" completed successfully   | Total time: 2 secs   |   |  |                                  |   |                           |            |
| Focess "Generate Frogramming File". Frogramming File". Frogramming File". Exacted : "Generate Programming File". Exacted : "Generate Pr   | Deserves Million Prove Diserves  | · Development · mining                  |  |                                  |   |                           |            |
| Statted : "Generate Programming File". Process "Generate Programming File" completed successfully  | Process "Generate Post-Place   | s source Static Timing" comp            | ietea successfully                           |                                  |   |                           |            |
| Process "Generate Programming File" completed successfully   | Started : "Generate Programm   | ing File".                              |  |                                  |   |                           |            |
| Process "Generate Programming File" completed successfully   |  | and a second of                         |  |                                  |   |                           |            |
| Concole 😋 Encos  | Process "Generate Programmin   | g File" completed successful            | ly   |                                  |   |                           |            |
| Conside Conside Consider Consider Constant Const   | ~  |   |  |                                  |   |                           |            |
| 🗐 Console 👩 Exrors 🔬 Warnings 🔃 Tcl Shet 🙀 Find in Files   |  |   |  |                                  |   |                           |            |
|  | Console Console  | Tcl Shell 🙀 Find in Files               |  |                                  |   |                           |            |
|  | 🔤 🔤 🥁 citoro 🚺 wainingo  |   |  |                                  |   |                           |            |

HDL Synthesis

\_\_\_\_\_

Synthesizing Unit <decoder>.
 Related source file is "C:\ece3829\decoder\decoder.v".
 Found 8x8-bit Read Only RAM for signal <led>
 Summary:
 inferred 1 RAM(s).
Unit <decoder> synthesized.

\*

\*

#### View the Schematic Representation



#### Decoder Implemented on FPGA

| Stillinx FPGA Editor - decoder.ncd         File       Edit       View       Lools       Window       Help         Decolor       Color       Color       Color       Color       Color   |   |
|---|---|
| Arroy1       Image: State | exit<br>add<br>attrib<br>clear<br>delay<br>delete<br>drc<br>editblock<br>editmode<br>find<br>yellow ▼ hilite<br>ila<br>info<br>probes<br>autoprobe<br>route<br>route -fanout<br>swap<br>unroute |
| Building chip graphics  | ▲<br>▼  |
| For Help, press F1         xc3s200-4ft256         No L  | .ogic Changes //,   |

Jim Duckworth, WPI

#### Zooming in on Logic Slice



# Assigning Package Pins



Module 1

#### New Implementation to Match Target

Jim Duckworth, WPI

# Verilog and VHDL – Reminder

- VHDL like Pascal and Ada programming languages
- Verilog more like 'C' programming language
- But remember they are Hardware Description Languages -They are NOT programming languages
  - FPGAs do NOT contain an hidden microprocessor or interpreter or memory that executes the VHDL or Verilog code
  - Synthesis tools prepare a hardware design that is *inferred* from the behavior described by the HDL
  - A bit stream is transferred to the programmable device to configure the device
  - No shortcuts! Need to understand combinational/sequential logic
- Uses subset of language for synthesis
- Check could you design circuit from description?