ELEC2221
Digital Systems and Signal Processing
Mark Zwolinski (mz@ecs)
– Digital Systems, 1/2
Soon (Michael) Xin Ng (sxn@ecs),
Mohammed El-Hajjar (meh@ecs)
– Signal Processing, 1/2
Schedule 2018/2019

• Lectures
  – Wednesday 09:00, Avenue 65/1133 (L/T A) – MEH
  – Wednesday 10:00, Avenue 65/1133 (L/T A) – SXN/MZ
  – Friday 09:00, 29/1101 – MZ/SXN
  – Friday 15:00, 54/4011 – MZ/SXN
  – Not all timetabled lecture slots will be needed and lecturers may vary – see webpage

• Coursework (SystemVerilog based design) – 15%
  – 2 laboratory sessions, working individually
  – Demonstration and report

• Laboratory – 2 digital and 1 signal processing experiments – 15%
  – M4 - FPGA design of sequential logic
  – C4 – Testing of a state machine
  – C6 - Signal simulations in MATLAB

• Exam - 70%
Digital Design (1/2 of the course) - Topics

- SystemVerilog – more advanced aspects of the language
- Synthesis of complex systems for FPGA implementation
- Microprocessor architecture
- Testing Digital Circuits and Design for Test
Types of PLDs

- CPLD
- FPGA

Equivalent logic gates

- Lattice ispGAL 22v10
- Altera MAX 9000
- Lattice MachXO
- Altera Cyclone IV
- Xilinx Virtex-6
- Altera Stratix V, eq. 15M ASIC gates
- Xilinx Virtex UltraScale, eq. 50M ASIC gates
- Altera Stratix V, eq. 15M ASIC gates
- Xilinx Virtex UltraScale, eq. 50M ASIC gates
Three major types of PLDs

- **PLD (Simple Programmable Logic Devices)**
  - Typically equivalent to a few hundred logic gates
  - Fixed internal routing (PAL or PLA)

- **CPLD (Complex Programmable Logic Devices)**
  - Multiple PLDs on a single chip
  - Programmable routing

- **FPGA (Field Programmable Gate Arrays)**
  - An array of logic blocks
  - Equivalent to up to 1mln logic gates
  - Large memories both RAM and flip-flop based
  - Dedicated hardware blocks, e.g. DSP or arithmetic

<table>
<thead>
<tr>
<th>Range</th>
<th>Kintex UltraScale</th>
<th>Virtex UltraScale</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logic Cells (K)</td>
<td>355–1,160</td>
<td>627–4,433</td>
</tr>
<tr>
<td>Block Memory (Mb)</td>
<td>19.0–75.9</td>
<td>44.3–132.9</td>
</tr>
<tr>
<td>DSP (Slices)</td>
<td>1,700–5,520</td>
<td>600–2,880</td>
</tr>
<tr>
<td>DSP Performance (GMAC/s)</td>
<td>8,180</td>
<td>4,268</td>
</tr>
<tr>
<td>Transceivers</td>
<td>16–64</td>
<td>36–120</td>
</tr>
<tr>
<td>Peak Transceiver Speed (Gb/s)</td>
<td>16</td>
<td>33</td>
</tr>
<tr>
<td>Peak Serial Bandwidth (full duplex) (Gb/s)</td>
<td>2,086</td>
<td>5,886</td>
</tr>
<tr>
<td>PCle® Interface</td>
<td>2–6</td>
<td>2–6</td>
</tr>
<tr>
<td>Memory Interface Performance (Mb/s)</td>
<td>2,400</td>
<td>2,400</td>
</tr>
<tr>
<td>I/O Pins</td>
<td>312–832</td>
<td>364–1,456</td>
</tr>
<tr>
<td>I/O Voltage</td>
<td>1.0–3.3V</td>
<td>1.0–3.3V</td>
</tr>
</tbody>
</table>
Example of state-of-the art FPGA: Xilinx Spartan 6 – low cost FPGA

- 45 nm process optimized for cost and low power
- Low-power 1.0V core voltage, 3.3V to 1.2V I/O standards and protocols
- Up to 1050 Mb/s data transfer rate per I/O
- High-speed serial transceivers, Up to 3.125 Gb/s
- High speed interfaces including: Serial ATA, 1G Ethernet, PCI Express, DisplayPort, etc.
- Extensive DSP support
  - High-performance arithmetic and signal processing
  - One DSP slice contains:
    - Fast 18 x 18 multiplier and 48-bit accumulator
    - Pre-adder to assist filter applications
- Integrated Memory Controller blocks, DDR, DDR2, DDR3, up to 800 Mb/s (12.8 Gb/s peak bandwidth)
- Abundant logic resources, up to
  - 147,444 logic cells, (4 LUTs and 8 flip-flops each)
  - 1.3 MB distributed RAM
  - 190 DSP slices
- Digital Clock Managers (DCMs) eliminate clock skew and duty cycle distortion
  - Phase-Locked Loops (PLLs) for low-jitter clocking
  - Sixteen low-skew global clock networks
- Enhanced security for design protection
  - Unique Device DNA identifier for design authentication
  - AES bitstream encryption in the larger devices
- Fast embedded processing with enhanced, low cost, MicroBlaze™ soft processor
- Price approx $100/chip
Example of state-of-the art FPGA: Altera Stratix V

- Three variants
  - Stratix V GT – dedicated to ultra-high bandwidth communication applications
  - Stratix V GX – supports optical comms modules
  - Stratix V GS - abundance of DSP blocks, in total 3,926 18x18 multipliers
- 28 nm process
- 234,720 High-performance Adaptive Logic Modules (ALMs)
  - Each ALM: 8-input LUT, 2 full adders, 4 flip-flops
  - Some ALMs can be configured as true dual port 640-bit synchronous memory blocks.
- 512 variable precision 18x18 bit DSP blocks
  - Each DSP: two 18x18 multipliers, two filter coefficient banks, 64-bit adder, accumulator, etc.
- 2,560 M20K RAM blocks (20KB each)
- PCI Express support
- 4 x 28Gbps and 31 12.5 Gbps tranceivers

- Typical applications include
  - 100Gb ethernet card
  - Military radar applications
  - Studio video server

- Price approx $5K-$7K/ chip (Stratix IV; Stratix V)
  - Price of Cyclone V ~$250, Cyclone IV - $80
Design Tools

- Simulation
  - Essential! Need to verify up to 5M logic elements.
  - ModelSim is leading simulator
- Formal verification is now widely used (but not covered here in depth)
- Synthesis
  - Synplify (general purpose), Quartus (Altera only)
  - VHDL, Verilog and SystemVerilog are standard languages
  - SystemC (high-level C++ package), C++, OpenCL starting to be used
  - FPGA synthesis is not the same as ASIC synthesis – different tools, different objectives, different constraints
FPGA Applications

• Prototypes
  – Emulation is faster than simulation

• Products
  – FPGAs are "Systems on Chip"; almost all include RAM, DSP structures, processors (ARM or PowerPC)

• Accelerators
  – Growing use in High Performance Computing – faster, more energy efficient than standard, parallel computers
  – But difficult to use, hence OpenCL
Revision – Design of Vending Machine

• A drink costs 40c. The machine accepts 20c and 10c coins (all other coins are rejected by the mechanics of the system).

• Once 40c have been inserted, the drink is dispensed. If more than 40c are inserted, all coins are returned.

• The machine has two lights: one to show that it is ready for the next transaction, and one to show that further coins need to be inserted.
SystemVerilog describes concurrent hardware
- Each always block describes a piece of hardware
- Therefore we can have more than one always block in a module
- This model has one combinational block (Next State and Output) and one sequential (State Register)
SystemVerilog Model of State Machine

- Use an enumerated type for the states
  - Don’t need to do a state assignment
- Combinational part is modelled with always_comb
  - Assign to next state and to outputs
  - Case statement – one branch for each state
  - If statements in each branch for Mealy (conditional) outputs and for selecting next state
  - Default assignments to each output and next state at start, to avoid accidental memory
- Sequential part is modelled with always_ff
  - Just like a flip-flop
module vending (output logic ready, dispense, ret, coin
  input logic clock, n_reset, twenty, ten);

  enum {A, B, C, D, F, I} present_state, next_state;

always_ff @(posedge clock, negedge n_reset)
  begin: SEQ
    if (!n_reset)
      present_state <= A;
    else
      present_state <= next_state;
  end
always_comb
begin: COM
ready = '0;
dispense = '0;
ret = '0;
coin = '0;
next_state = present_state;

unique case (present_state)
 A : begin
  ready = '1;
  if (twenty)
    next_state = D;
  else if (ten)
    next_state = C;
  end
 B : begin
  dispense = '1;
  next_state = A;
  end
 C : begin
  coin = '1;
  if (twenty)
    next_state = F;
  else if (ten)
    next_state = D;
  end
endcase
end
endmodule
SystemVerilog – Comments

- `present_state` and `next_state` must be variables of enumerated type.
- Care is needed with combinational process - can easily create latches.
  - Default values for outputs and `next_state`.
- `always` blocks are labelled – need `begin` and `end`.
- Use blocking assignment (=) in `always_comb`; use nonblocking assignment (<=) in `always_ff`.
  - NEVER mix them!
  - This avoids race conditions and indeterminate simulations.
- Unique case checks that all states are included.
- There are other ways to write state machines, but this style is recommended.
Initial Values

• The initial value of a logic variable is 'X. Shows in simulation as a red line. Useful for debugging.
• (In real life, an 'X or a 'Z will always resolve to '1 or '0.)
• Initial value of an enumerated type is the 'left-hand' value. i.e. A in the example. Less useful for debugging.
• `enum logic [2:0] {A, B, C, D, F, I}
  present_state, next_state;
• Initial value (in simulation) for present_state and next_state are now 3'bXXX
  – Can easily see if the state machine is not reset.
• We will look at state assignments in a later lecture.