What We Learned - ASICs

- High volume, high performance, custom digital logic IC
- ASICs are built with HDLs, not schematics
- ASICs are built with scripts, not GUIs

What We Learned - System Verilog

- Our language of choice
- Rapidly becoming a standard HDL
- Describes the structure of digital logic, not an algorithm
- Can't be treated like another language
- If you can't draw it, don't try to code it
- If you can't draw what you've coded, rewrite till you can

What We Learned - System Verilog

- We build digital logic by:
  - Making combo logic: always_comb or assign
  - Making Flip-flops : always_ff

What We Learned - System Verilog

- State machines
  - always_ff for state storage

```
always_ff @(posedge clk, negedge rst_n)
  if (!rst_n) sm_ps <= IDLE;   //at reset, go to idle state
  else        sm_ps <= sm_ns;  //otherwise, go to the next state
```

  - always_comb for next state assignment and outputs

```
always_comb begin
  sm_ns   = XX;   //default assignment, ns vector
  out_sig = 1'b0; //default assignment, output signal
  case (sm_ps)
    IDLE :
      if (req)       sm_ns = BUSY;
      else           sm_ns = IDLE;
  ...
   endcase
  end //always
```

- Keep FFs and ns logic separate for each state machine
- Non blocking assignment under always_ff
- Blocking assignment under always_comb

What We Learned - Digital Design

- Data path first
  - Make block diagram
- Control logic second
  - Create state machines and timing diagram
- Define everything, then begin coding
- Coding should be a typing speed limited, not designing

What We Learned - Digital Design

- Debug piece by piece
- Check with eyeballs, then testbench
- Testbench with TCL (dofiles), Verilog, diff

What We Learned - Synthesis

- Driven by constraints
- Different structure, but same functionality
- Clean RTL a must, intentions clear
- Garbage-in, Garbage-out still applies

What We Learned - Design for Test

- Required by ASIC process
- Checks manufacturing process, not functionality
- Excite fault, propagate out to pins
- Tester time, and I/O pins are expensive
- Need to test with minimal set of vectors
- Scan insertion can cause timing problems

What We Learned - ASIC Design Process

- Fully automate via scripts
- Bash, TCL, SVerilog, Perl, Python.....
- Self-documenting, repeatable process

What We Left out....

- Place and route with backannotation
- Usually not done by ASIC designers
- Clock uncertainty, routing delays resolved with new .sdf
- I/O shell wrapper