ECE 353 Lab 3

MIDI Note Number Display
Lab Info
Board Demonstration
Build Process

- Board Assembly
- Verilog Programming
  - Plan State Machine
  - Synthesis
  - Simulation
  - Burning the Program
- Testing
  - MIDI-OX
  - Logic Analyzer
Board Assembly

- No “correct layout”
- Easy way vs. hard way
  - Plan ahead
  - Practice neat wiring
  - Check electrical connections
    - Double check the schematic
  - Test using multimeter
Verilog Synthesis

- Use the full compile option when synthesizing
- Check warnings if your design does not work
  - Some harmless, some indicate major issues.
- State machine viewer, RTL viewer
  - Good for checking overall design
  - RTL viewer more useful for designs with multiple modules
Waveform Simulation

- Simulate design using Vector Waveform Files
  - Add inputs pins to your waveform, create input signals
  - Add outputs and internal registers to “watch” values
  - Modify simulation settings (under “Assignments”) to specify a simulation length.

- Make sure your design simulates properly before moving on!
Programming the CPLD

- Altera USB ByteBlaster (JTAG)
- Quartus interface
  - After Verilog is synthesized and programmer
    - tools -> programmer -> select program
    - then click start
  - Check Quartus output to confirm programming succeeded
Testing with MIDI-OX

- Set up Yamaha UX16 as output device in MIDI-OX
- Use keyboard input to send notes to breadboard
- Confirm number on board agrees with played note
Saleae USB Logic Analyzer

- **Settings**
  - Make sure to set a proper sampling rate
  - Set a trigger based on incoming MIDI signal

- **Examine MIDI protocol**
  - Confirm output of MIDI-OX
  - Confirm clock signal

- **Not critical for this lab**
  - Still good practice
Pitfalls and How to Avoid Them

- **Breadboarding**
  - Bad connections
  - Miswired parts (Have everyone in your team check)
  - Dead chips possible
  - Red wire for power, black wire for ground

- **MIDI-OX**
  - Simultaneous notes

- **Verilog/Quartus**
  - Confirm Quartus pin layout agrees with board
  - Poor programming style can lead to mismatch between simulation and device behavior
  - Take time to understand blocking/non-blocking, and plan out how you are going to structure your program.
Quartus II Misc.

- **Create new Project**
  - Page 1: Name Appropriately
  - Page 2: Nothing
  - Page 3: Change Family and Name as given in lecture
  - Page 4/5: Nothing

- **File -> new -> Verilog HDL file**

- **Start Compilation – Purple arrow**

- **Assignments -> Pins**
  - Make sure you have right chip, set on Page 3
  - Click and drop I/O to pins, close re-compile.

- **Tools -> Programmer**
  - Check connection/settings, click start wait for 100%
General Advice

- Start early. Plan before you start.
- Don't panic if it doesn't work the first time!
  - Simulate a complete MIDI message in Quartus
  - Check breadboard wiring
  - Confirm the CPLD is working properly with a test program (use LEDs)
- Reference lecture slides on Verilog frequently.
  - Correctly segregate sequential and combinational parts
- Google and Wikipedia are your friend
  - Understand the MIDI protocol
  - Learn from Verilog and Quartus examples
Tips and Tricks

- Use neat, color-coded, wiring (wiring is most common source of bugs)
- Do NOT install the Altera ByteBlaster (parallel port programmer) drivers. Quartus recognizes the programmer and installing the driver will break the JTAG driver.
- Use LED lights in debugging process.
- Use your tools: logic analyzer, oscilloscope, multimeter
- If CPLD gets hot, check VCC and GND of CPLD
- Test hypotheses, isolate variables. Don’t just start randomly swapping components.
Tips and Tricks

- If running in virtual machine, make sure USB devices are configured correctly