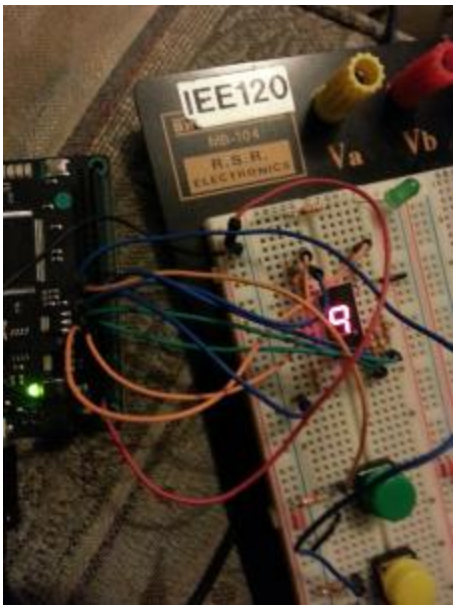
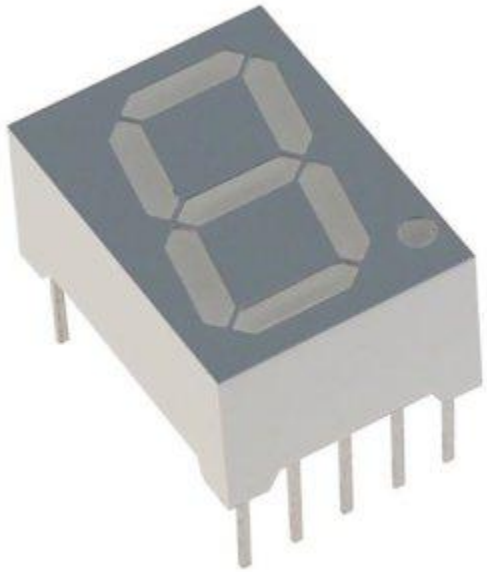
Controlling A Seven Segment Display Using Mojo

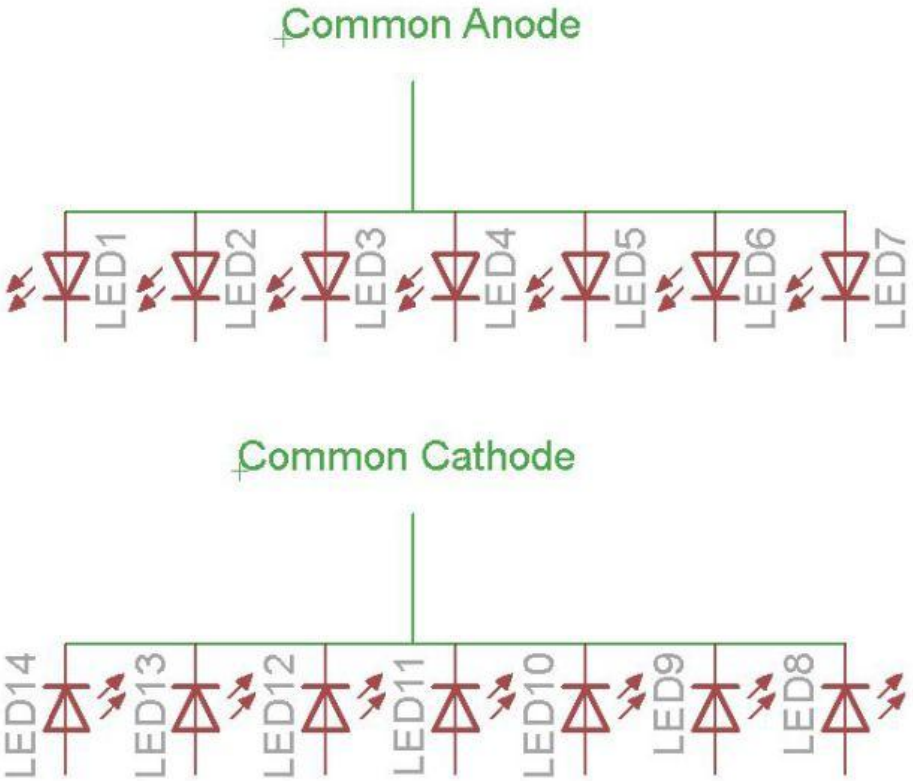


Remember a couple weeks ago when I said that I will be working on tutorials for the Mojo? Today's post will be the first post that will be a Mojo tutorial. Although I originally planned to show you guys how to create a simple half adder with a 7 segment display, I realized we have to learn how to control the seven segment display first. Today, I will show you not only how to connect your seven segment display to the Mojo, but how to implement a digital design to control the display on the Mojo.

What Are Seven Segment Displays?



A seven segment display is just seven LEDs arranged to display a number ranging from 0-9. However, seven segment displays have two different connections: common anode and common cathode.Seven segment displays with a common anode connection means that all of the LEDs have their anodes connected together. However, seven segment displays with common cathode connection means all of the LEDs cathodes are connected to each other. The picture on the bottom shows common anode and common cathode seven segment displays.



When you have a common anode display, you just need one current limiting resistor, but 7 devices to sink the current for each LED. A common cathode display requires multiple current limiting resistors, but one ground connection to sink the current for all of the LEDs. For today's tutorial, we will use a common cathode seven segment display.

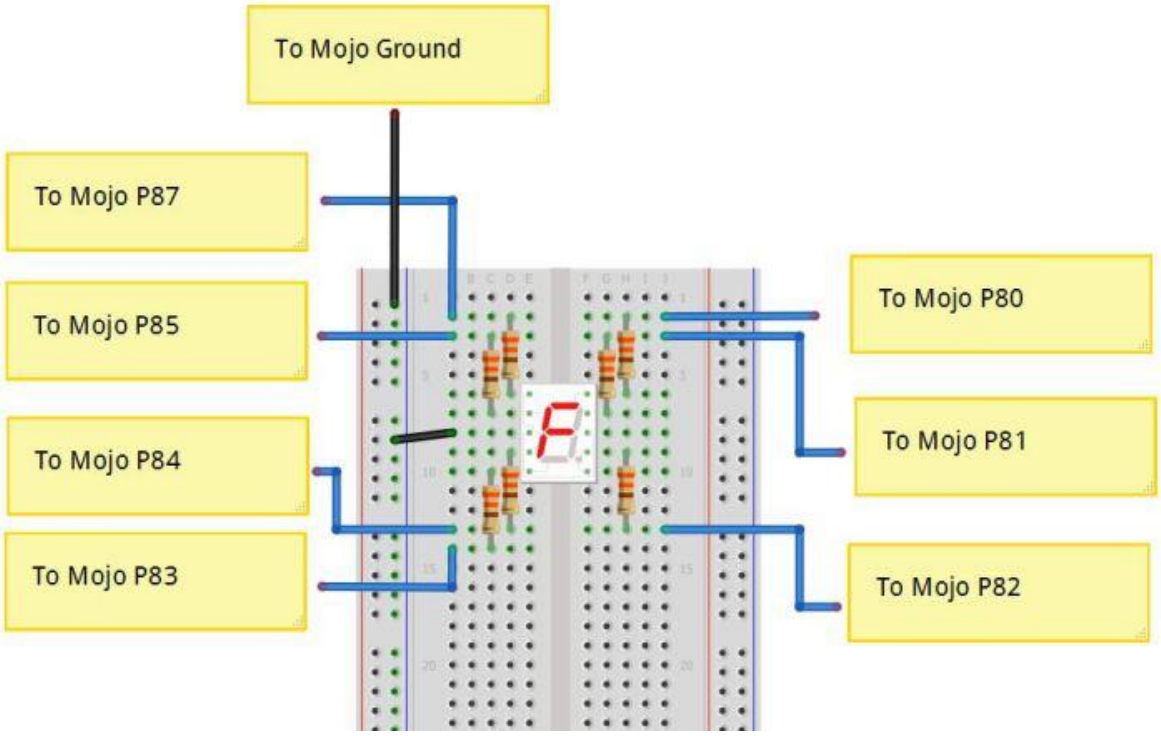
What Will You Need?

1) (1x) Mojo Board

2) (1x) LTS-4301JR 7 Segment Display
3) (7x) 330 ohm resistor
4) (1x) Micro USB cable

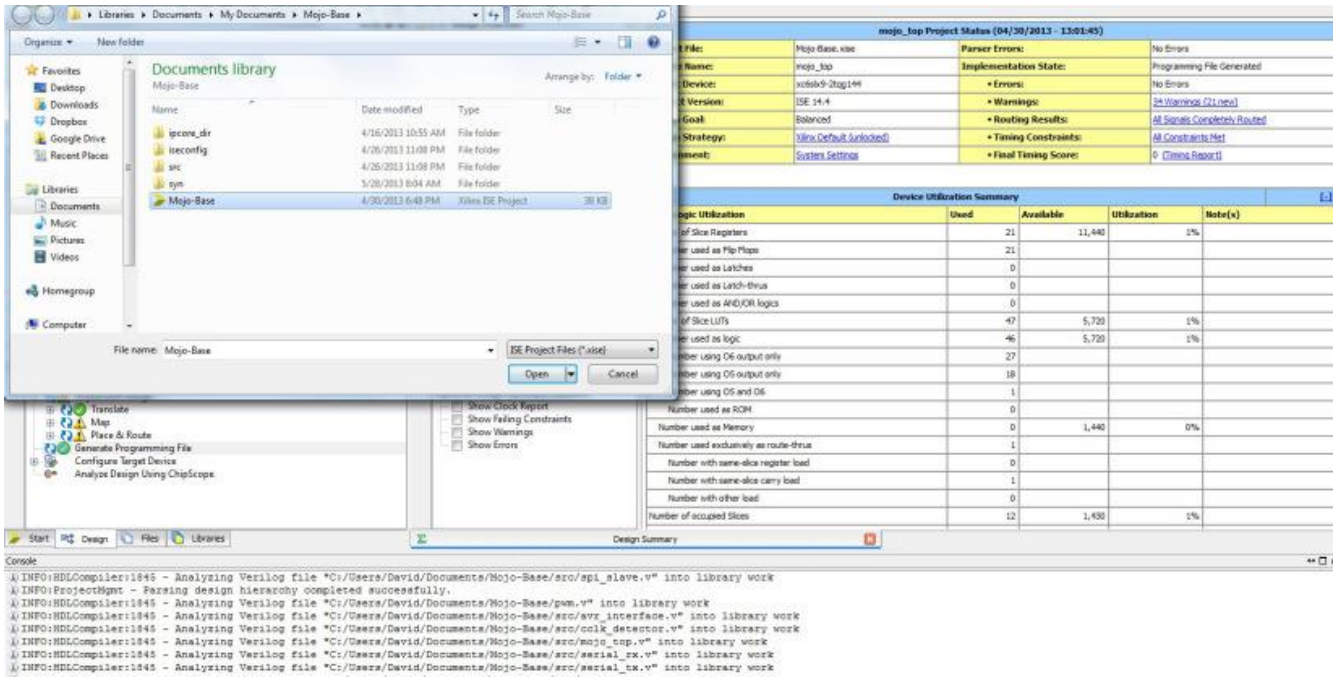
Wiring Connection

The picture below shows how to connect the 7 segment display to the Mojo.



How To Implement Your Digital Design Onto The Mojo?

For the next part, you'll probably need to visit my getting started with FPGAs guide as alot of the steps from that tutorial can be applied for this one such as downloading the Xilnix Web ISE.
Afterwards, you'll need to download the Mojo project base.
After downloading and extracting the Mojo project base, open Xilnix Web ISE. Once you open up the Xilnix ISE, go to file->open project. Navigate to your Mojo Project base and select the Xilnix ISE Project file inside the folder.



From the hierarchy menu in the Xilnix Web ISE, double click on the Mojo Top.v file, and edit it to look like this.

```
1   module mojo_top(
2       input clk,
3       input rst_n,
4       input cclk,
5       output[7:0]led,
6       output spi_miso,
7       input spi_ss,
8       input spi_mosi,
```

```
9         input spi_sck,
10        output [3:0] spi_channel,
11        input avr_tx,
12        output avr_rx,
13        input avr_rx_busy,
14         output [6:0] display
15        );
16
17    wire rst = ~rst_n;
18
19    assign spi_miso = 1'bz;
20    assign avr_rx = 1'bz;
21    assign spi_channel = 4'bzzzz;
22
23    assign led = 8'b0;
24    seven_segment_display_decoder(.clk(clk),.number(9),.seven_segment_display_control_lines(di
25    endmodule
```

Don't worry about line 24. We're going to add the seven_segment_display_decoder module right now. Right click on Mojo Top.v and add a new verilog file. We're going to call it "seven_segment_display_decoder." After creating the file, add the following code to the file.

```
1  `timescale 1ns / 1ps
2  module seven_segment_display_decoder(clk,number,seven_segment_display_control_lines);
3  input clk;
4  input [8:0] number;
5  output reg [6:0] seven_segment_display_control_lines;
6
7  always @ (posedge clk)
8  begin
9     case(number)
10    1:seven_segment_display_control_lines = 7'b0110000;
11    2:seven_segment_display_control_lines = 7'b1101101;
12    3:seven_segment_display_control_lines = 7'b1111001;
13    4:seven_segment_display_control_lines = 7'b0110011;
14    5:seven_segment_display_control_lines = 7'b1011011;
15    6:seven_segment_display_control_lines = 7'b1011111;
```

```
16        7:seven_segment_display_control_lines = 7'b1110000;
17        8:seven_segment_display_control_lines = 7'b1111111;
18        9:seven_segment_display_control_lines = 7'b1110011;
19        default:seven_segment_display_control_lines = 7'b1111110;
20        endcase
21    end
22    endmodule
```

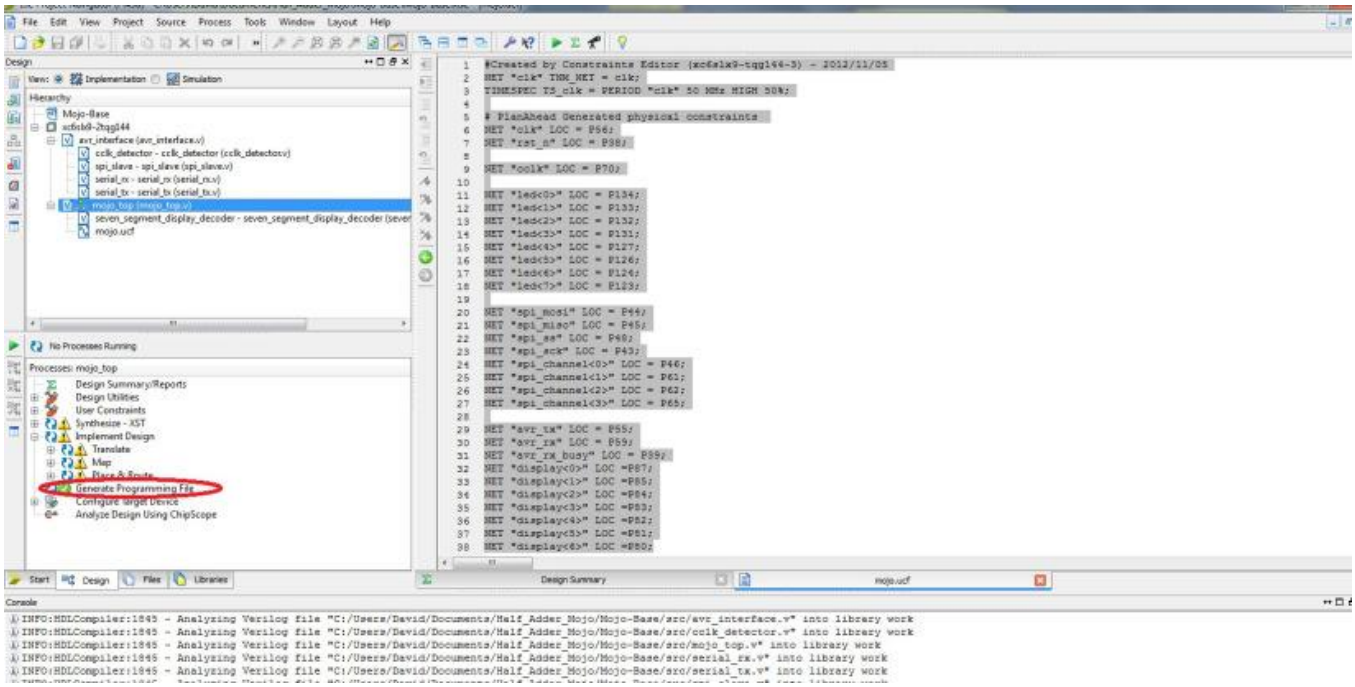
Finally, select your mojo.ucf file. Edit it to make sure it look like this.

```
1     #Created by Constraints Editor (xc6slx9-tqg144-3) - 2012/11/05
2     NET "clk" TNM_NET = clk;
3     TIMESPEC TS_clk = PERIOD "clk" 50 MHz HIGH 50%;
4
5     # PlanAhead Generated physical constraints
6     NET "clk" LOC = P56;
7     NET "rst_n" LOC = P38;
8
9     NET "cclk" LOC = P70;
```

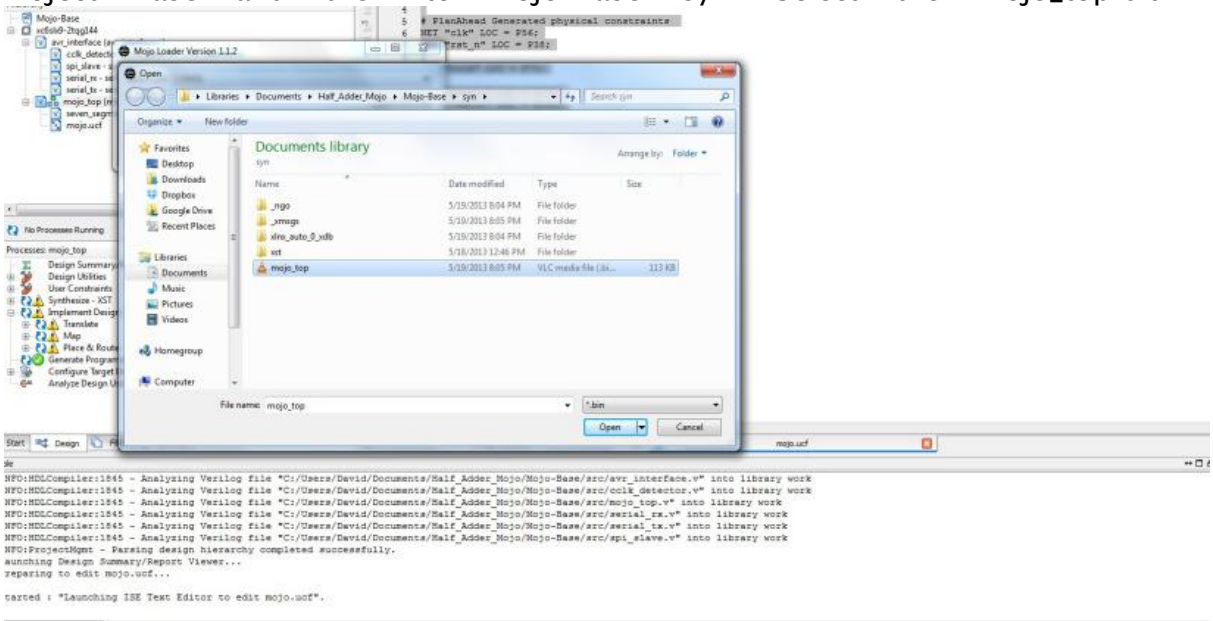
```
10
11  NET "led<0>" LOC = P134;
12  NET "led<1>" LOC = P133;
13  NET "led<2>" LOC = P132;
14  NET "led<3>" LOC = P131;
15  NET "led<4>" LOC = P127;
16  NET "led<5>" LOC = P126;
17  NET "led<6>" LOC = P124;
18  NET "led<7>" LOC = P123;
19
20  NET "spi_mosi" LOC = P44;
21  NET "spi_miso" LOC = P45;
22  NET "spi_ss" LOC = P48;
23  NET "spi_sck" LOC = P43;
24  NET "spi_channel<0>" LOC = P46;
25  NET "spi_channel<1>" LOC = P61;
26  NET "spi_channel<2>" LOC = P62;
```

```
27    NET "spi_channel<3>" LOC = P65;
28
29    NET "avr_tx" LOC = P55;
30    NET "avr_rx" LOC = P59;
31    NET "avr_rx_busy" LOC = P39;
32    NET "display<0>" LOC =P87;
33    NET "display<1>" LOC =P85;
34    NET "display<2>" LOC =P84;
35    NET "display<3>" LOC =P83;
36    NET "display<4>" LOC =P82;
37    NET "display<5>" LOC =P81;
38    NET "display<6>" LOC =P80;
```

So we finished all of the code for the Mojo. Now we need to create the bit file. Before we create the bit file, make sure you save everything. Once you save everything, select mojo_top.v. Underneath the hierarchy menu is the processes menu. Look for “Generate Programming File” in the Processes menu. Click on it. Xilnix will now try to create a bitfile aka your digital design.



Now we need to burn the bitfile onto the Mojo board. If you have not done so, download the Mojo Loader. Once your download and extract the mojo loader and configure your computer to recognize the Mojo, open the application file inside the Mojo Loader folder. From the Mojo Loader select “Open Bin File.” Navigate to your Mojo Project Base and then to Mojo-Base->syn. Select the mojo_top.bit file.



Finally, select load file. When the bit file is uploaded to your Mojo, you should see a 9 on the seven segment display.

Source: http://coolcapengineer.wordpress.com/2013/06/03/controlling-seven-segment-display-using-mojo/