MILESTONE 2

CPLD Familiarization

Objectives

- Getting started with EPM240 CPLD
- Familiarization with various methods of design entry
- Creation of more modules for design reuse

2.1 Pre-Requisites

Download and follow the EPM240 Board checkout procedure in AN04 EPM240 Board :
http://raden.fke.utm.my/appnotes/AN04%20EPM240%20Board.pdf

After going through the example in AN04, you should get a blinking LED driven by the EPM240 CPLD.

2.2 System Overview

In this milestone, you will build a Knight Rider lights using the CPLD and 6 LEDs. The overview of the system is in Fig. 2.1.

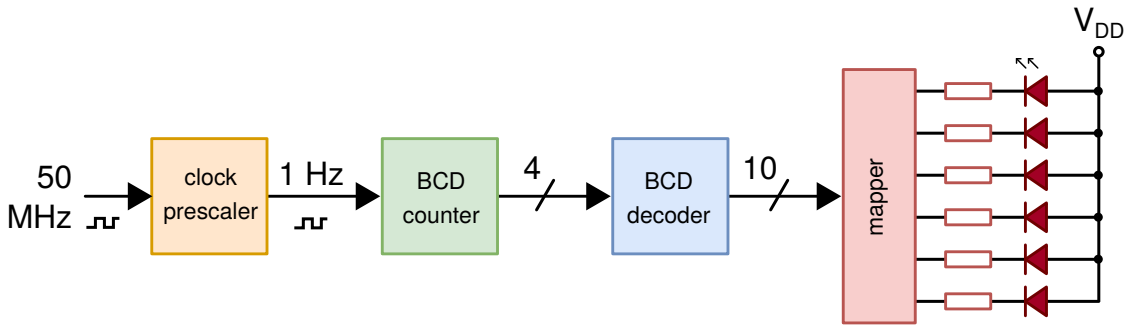

Figure 2.1: Overview of the system.

We will create all low level modules first. Afterwards, they are integrated at the top level schematic diagram.

2.3 Mapper

Section Objective
Design entry using primitives.

The mapper is a custom circuit that maps 10 BCD outputs to 6 LEDs.

1. Create a new block diagram file
2. Insert four **bnor2** gates, 10 input ports and 6 output ports.
3. Make the connection shown in Figure 2.2.

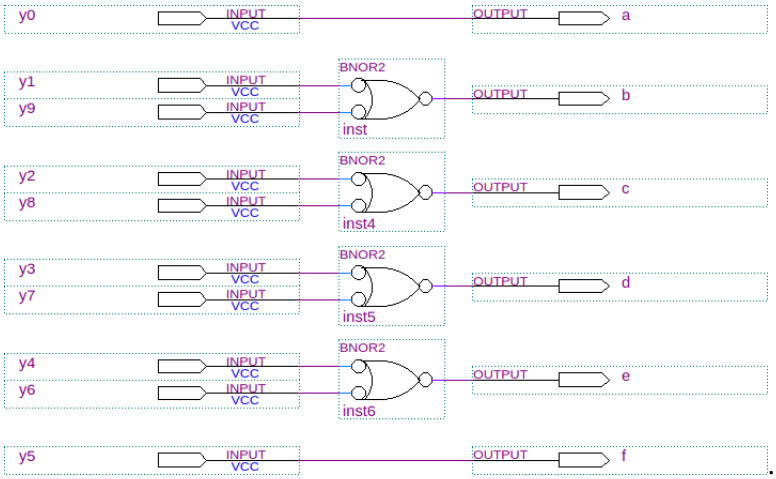

Figure 2.2: Mapper schematic.

4. Save it as mapper.bdf.
5. Set it as Top-Level Entity (temporarily) and compile it.
6. Save it as mapper symbol file (**mapper.bsf**).

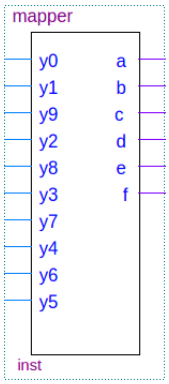

Figure 2.3: Mapper symbol. Note: you can see this symbol by opening another block diagram document and importing it.

2.4 Prescaler

Section Objective
Design entry using Verilog HDL.

The prescaler module slows the clock from 50 MHz to exactly 1 Hz.

1. Create a Verilog file and enter this code.

```
module prescaler( clkin, clkout );
  input clkin;
  output reg clkout;
  reg[25:0] counter;

  always @(posedge clkin)
    begin
      if (counter == 0)
        begin
          counter <= 24999999;
          clkout <= ~clkout;
        end
      else
        counter <= counter - 1;
    end
endmodule
```

2. Save as **prescaler.v**.

3. Set it as Top-Level Entity (temporarily) and compile it.

4. Save the prescaler as Symbol file (**prescaler.bsf**).

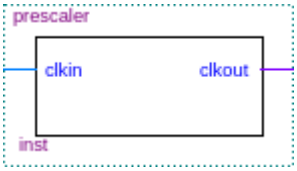

Figure 2.4: Prescaler symbol. Note: you can see this symbol by opening another block diagram document and importing it.

2.5 BCD Counter

Section Objective
Using the Library of Parameterized Modules (LPM) in the block diagram/schematic editor.

The BCD counter counts in Binary Coded Decimal, i.e. 0, 1, 2, ⋯ , 9, 0.

1. Create a new block diagram file.
2. Save the blank file as knightrider (or milestone2 etc) and set it as Top-Level Entity.
3. Call up the New Symbol dialog and type **lpm_counter** in the Name field.

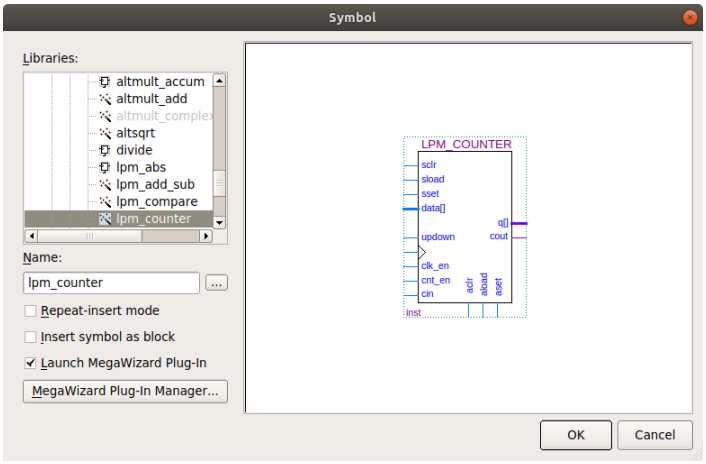

Figure 2.5: Insert a new lpm_counter symbol.

4. On Page 2c, click Next >.
5. On Page 3, set the width of the 'q' output bus to 4 bits. Click Next >.
6. On Page 4, set counter type to Modulus, with a count modulus of 10. Click Finish.
7. On Page 7, click Finish.
8. Place the new **lpm_counter** symbol anywhere on the schematic editor.
9. Add one input port.
10. Add a bus at the output of the counter and label it **q[3..0]**. Refer Figure 2.6.

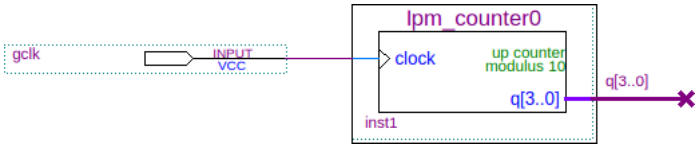

Figure 2.6: Connecting the input port and lpm_counter.

2.6 BCD Decoder

Section Objective
Using traditional TTL IC equivalent symbols in the block diagram/schematic editor.

A 7442 BCD decoder has 10 outputs and accepts a value in the range of 0..9. The output is active low. A value greater than 9 causes all outputs to be high.

1. Still in the schematic diagram from Section 2.5, call up the New Symbol dialog and type 7442 in the Name field.

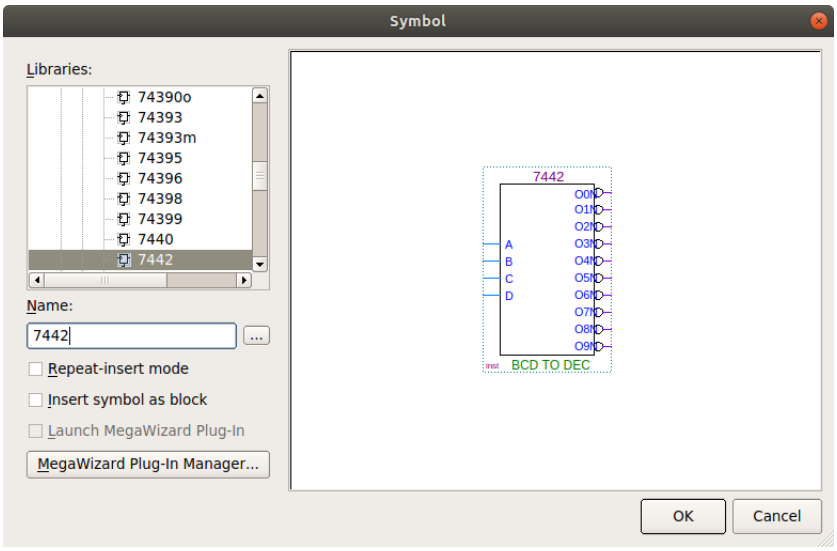

Figure 2.7: Insert a new 7442 device.

2. Insert the mapper symbol from Section 2.3.

3. Refer to Figure 2.8. Label the 7442 inputs as q0, q1, q2, and q3. Connect the outputs of the 7442 and mapper using the diagonal node tool Add outputs and label accordingly.

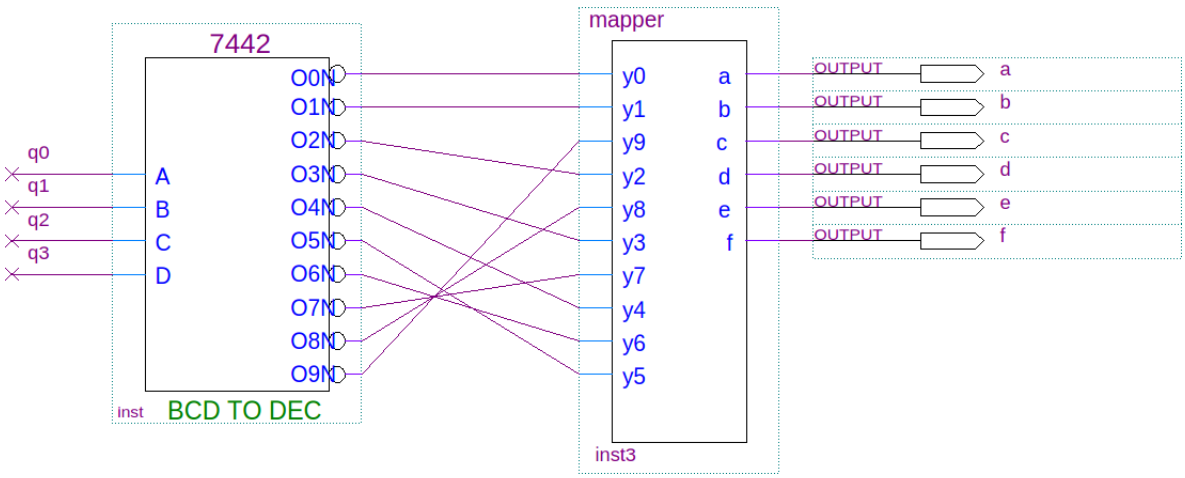

Figure 2.8: Detailed connection of 7442, mapper and outputs.

2.7 Simulating the Circuit

1. The final top-level file should look like Figure 2.9.

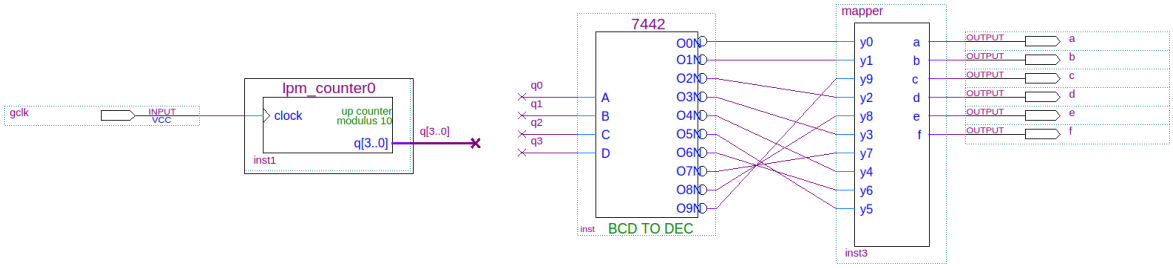

Figure 2.9: The complete circuit minus prescaler.

2. Compile this file.
3. Create a new waveform file.
4. Insert all pins into the waveform file.
5. Using the default setting for end time, set the clock period so that you can see at least 10 clock cycles in the simulation window.

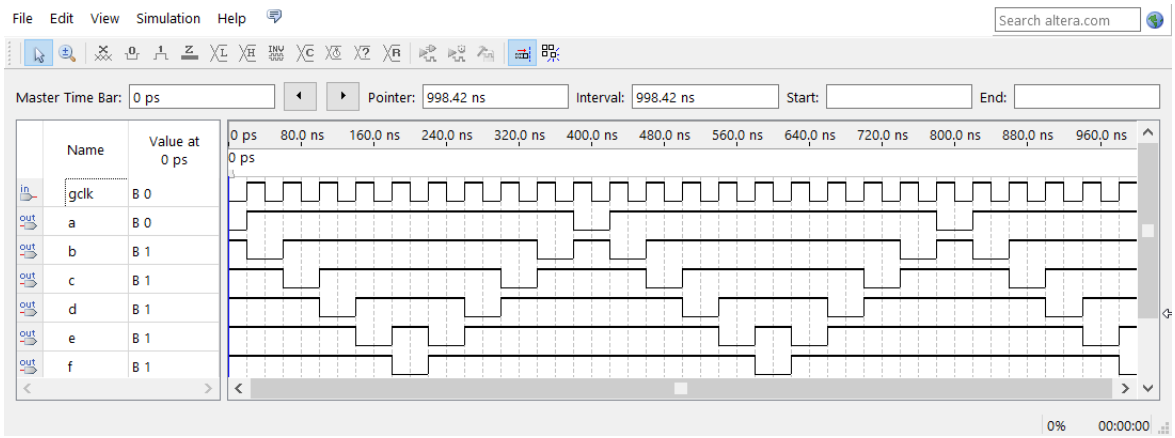

Figure 2.10: Simulation using clock period of 40 ns.

6. When the simulation waveform appears similar to Figure 2.10, you can add the prescaler and program the circuit.

2.8 Programming the CPLD

1. In the top-level schematic diagram, add the prescaler module. The circuit now look like Figure 2.11. *After adding the precaler, the circuit is not practical to be simulated.*

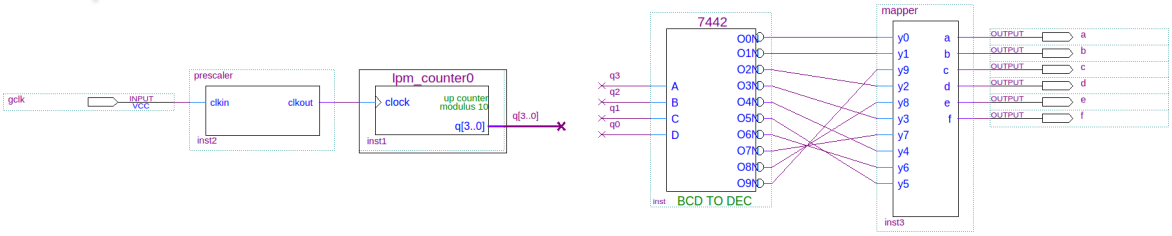

Figure 2.11: Adding the prescaler.

2. From the **Assignments ➥ Device** menu verify that you have chosen the EPM240T100C5 device.

3. In the **Assignments ➥ Pin Planner** verify that pin 64 is set to gclk.

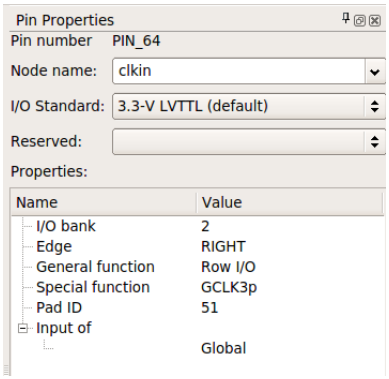

Figure 2.12: Set pin 64 to gclk.

4. Assign 6 more pins for the output ports. You can use any I/O bank. For example, you can use the settings from Figure 2.13 to use pins closest to ground pin on the EPM240 connector block. However, any other combination can be used.

Node Name	Direction	Location	I/O Bank	Fitter Location	I/O Standard	Reserved	urrent Strengt
out a	Output	PIN_74	2	PIN_74	3.3-V ...fault)		16mA ...ault)
out b	Output	PIN_76	2	PIN_76	3.3-V ...fault)		16mA ...ault)
out c	Output	PIN_78	2	PIN_78	3.3-V ...fault)		16mA ...ault)
out d	Output	PIN_82	2	PIN_82	3.3-V ...fault)		16mA ...ault)
out e	Output	PIN_84	2	PIN_84	3.3-V ...fault)		16mA ...ault)
out f	Output	PIN_86	2	PIN_86	3.3-V ...fault)		16mA ...ault)
in gclk	Input	PIN_64	2	PIN_14	3.3-V ...fault)		16mA ...ault)

Figure 2.13: A possible selection of pins.

(The two I/O bank can theoretically use different supply voltages for interfacing. On the red EPM240 board, this is not possible because the power supply rails are tied together. Therefore, any I/O bank can be used.)

I/O bank	VCCIO	GNDIO
1	Pin 9, 31,45	Pin 10, 32, 46
2	Pin 59, 80, 94	Pin 60, 79, 93

5. **Recompile the design**. The assigned pins should appear at the top level schematic.

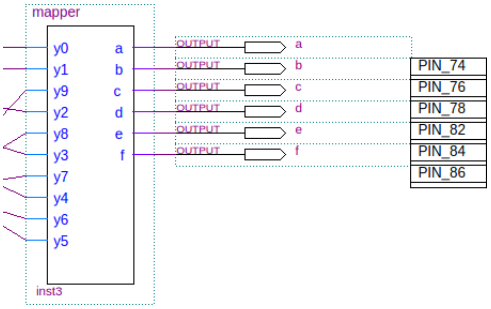

Figure 2.14: Assignment of pins on top level schematic.

6. Connect the LEDs according to Figure 2.1.
7. Program it. The LEDs should display an interesting pattern.

Your lecturer may add another pattern to program on the CPLD.