CS 207 Digital Logic - Spring 2020
Lab 9

James Yu

Apr. 15, 2020

Objective

1. Try synchronous sequential circuit design.

Lab Exercise Submission

1. You should name all source code as instructed. Mis-named files will not be recognized.
2. You should submit all source code files with an extension "`.v`".
3. You should submit all source code directly into the sakai system below. **Do not compress them into one folder.**

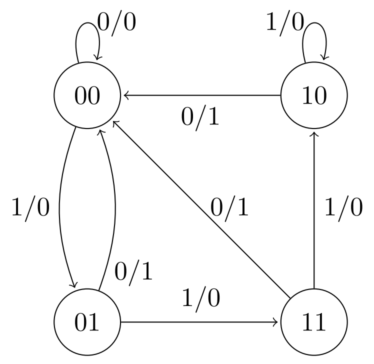
   `https://sakai.sustech.edu.cn/portal/site/ebf68254-68c5-4cfe-9d42-b26758f854ee`

4. Lab exercises should be submitted before the deadline, typically one week after the lab session. No late submission policy applies to lab exercises.

1 Mealy and Moore model circuits

1.1 Mealy model

Implement a Mealy model circuit with input X and output Y with clock signal clk. The state diagram is as follows:



We output the internal state of this model as A and B:

mealy.v

```
module mealy(Y, A, B, X, clk);
output reg Y;
output reg A, B;
input X;
input clk;

always @(X or A or B)
case({A, B, X})
    2, 4, 6: Y <= 1'd1;
    default: Y <= 1'd0;
endcase

always @(posedge clk)
case({A, B, X})
    0, 2, 4, 6: {A, B} <= 2'b00;
    1: {A, B} <= 2'b01;
    5, 7: {A, B} <= 2'b10;
    3: {A, B} <= 2'b11;
    default: {A, B} <= 2'b00;
endcase
endmodule
```

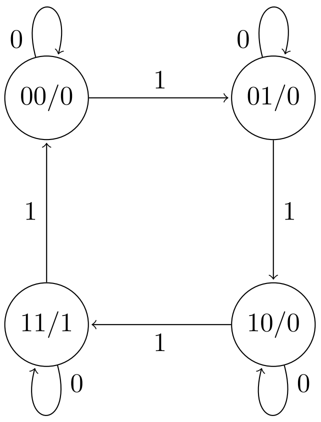
Here note that in mealy model, both the clock signal and the input X can drive the change of output Y. This module can be tested by the following testbench:

mealy.v

```
module mealy_tb;
reg CLK;
reg X;
wire y;
wire [1:0] Q;

initial begin
    CLK <= 1'b0;
    X <= 1'b0;
end

always @(CLK)
begin
    #1 CLK<=!CLK;
end

mealy mealy1(y, Q[1], Q[0], X, CLK);

initial begin
    $dumpfile("mealy.vcd");
    $dumpvars(0, mealy_tb);
end

initial begin
    $monitor("%3t: X = %b, y = %b, Q[1] = %b, Q[0] = %b", $time, X, y, Q[0], Q[1]);
    #2 X <= 1'b0; //00
    #2 X <= 1'b1; //01
    #2 X <= 1'b1; //11
    #2 X <= 1'b1; //10
    #2 X <= 1'b0; //10
    #2 X <= 1'b0; //00
    #2 X <= 1'b1; //01
    #2 X <= 1'b0; //00
    #2 X <= 1'b1; //01
    #2 X <= 1'b1; //11
    #2 X <= 1'b0; //00
    #2 $finish;
end
endmodule
```

1.2 Moore Model

Implement a Moore model circuit with input X and output Y with clock signal clk. The state diagram is as follows:



We output the internal state of this model as A and B:

moore.v

```
module moore(Y, A, B, X, clk);
output reg Y;
output reg A, B;
input X;
input clk;

always @(A or B)
case({A, B})
    3: Y <= 1'd1;
    default: Y <= 1'd0;
endcase
always @(posedge clk)
case({A, B, X})
    0, 7: {A, B} <= 2'b00;
    1, 2: {A, B} <= 2'b01;
    3, 4: {A, B} <= 2'b10;
    5, 6: {A, B} <= 2'b11;
    default: {A, B} <= 2'b00;
endcase
endmodule
```

Here note that in moore model, only the clock signal can drive the change of output Y. This module can be tested by the following testbench:

moore.v

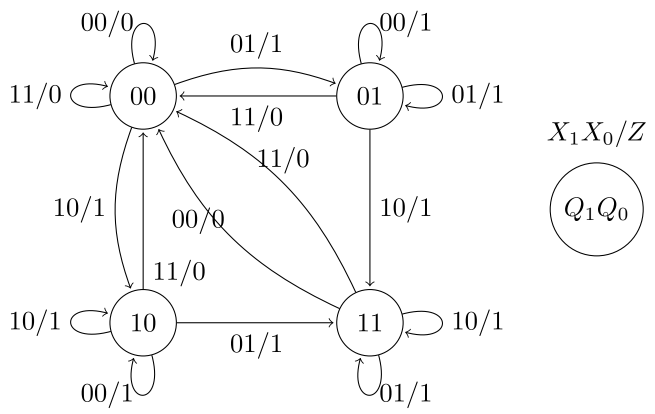
```
module moore_tb;
reg CLK;
reg X;
```

```verilog
wire y;
wire [1:0] Q;

initial begin
  CLK <= 1'b0;
  X <= 1'b0;
end

always @(CLK)
begin
  #1 CLK <= !CLK;
end

moore moore1(y, Q[1], Q[0], X, CLK);

initial begin
    $dumpfile("moore.vcd");
    $dumpvars(0, moore_tb);
end

initial begin
    $monitor("%3t: X = %b, y = %b, Q[1] = %b, Q[0] = %b", $time, X, y, Q[0], Q[1]);
    #2 X <= 1'b0;
    #2 X <= 1'b1;
    #2 X <= 1'b1;
    #2 X <= 1'b1;
    #2 X <= 1'b0;
    #2 X <= 1'b0;
    #2 X <= 1'b1;
    #2 X <= 1'b0;
    #2 X <= 1'b1;
    #2 X <= 1'b1;
    #2 X <= 1'b0;
    #2 $finish;
end
endmodule
```

2 Exercise

2.1 Exercise 1

Implement the following sequential circuit:



The module should take a 2-bit input X to produce a 1-bit output Z. The internal state is labeled by a 2-bit Q. The module should follow the design below:

mealy.v

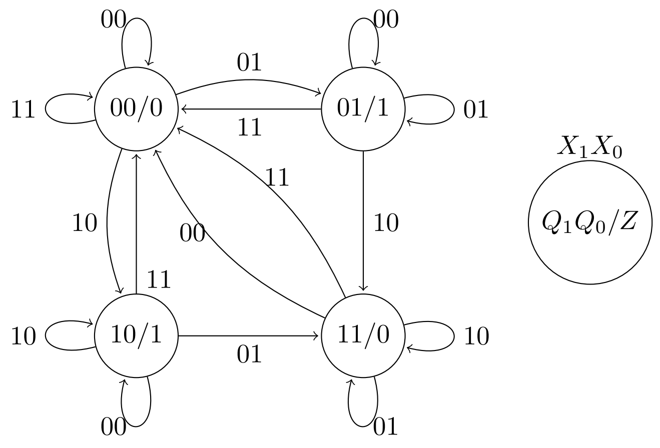
```verilog
module mealy(Z, Q, X, clk);
// Module Code
endmodule
```

and save in file `mealy.v`.

Assignment
Save the source code in **`mealy.v`**. Upload this file to Sakai under **Assignments → Lab Exercise 9**.

2.2 Exercise 2

Implement the following sequential circuit:



The module should take a 2-bit input X to produce a 1-bit output Z. The internal state is labeled by a 2-bit Q. The module should follow the design below:

moore.v

```
module moore(Z, Q, X, clk);
// Module Code
endmodule
```

and save in file `moore.v`.

Assignment
Save the source code in **`moore.v`**. Upload this file to Sakai under **Assignments** → **Lab Exercise 9**.

Assignment
When you finished Lab Exercise 9, there should be two `.v` files in the Sakai system: **`mealy.v`** and **`moore.v`**.