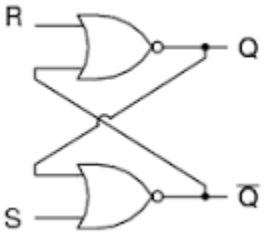
EE 2420 Lab Guide 3: Latches

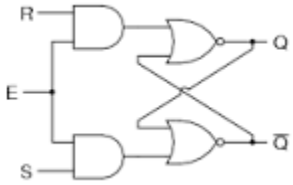
Written by: Grant Seligman, James Starks, Gabe Garves

Example Overview:

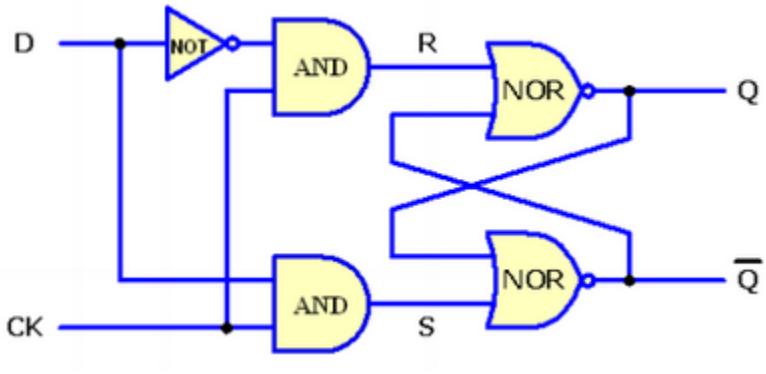
Design an SR latch, gated SR latch, and D flip-flop using Verilog HDL.
The purpose of latches is for memory. Depending on the input, the output will "remember" the previous input. Think of it as a 1-bit memory chip.



SR Latch



Gated SR Latch



D Flip-Flop

Figure 1

Verilog Code Breakdown:

The SR latch only uses 2 NOR gates. The inputs of the NOR gate is the output of the other NOR gate and either S or R.

```
1   //SR Latch (Asynchronous)
2   /*
3   AUTHOR: Gabe Garves
4   DATE: 11/15/2019
5   FROM: TXST SENIOR DESIGN PROJECT FALL 2019
6   FOR: TEXAS STATE UNIVERSITY STUDENT AND INSTRUCTOR USE
7   */
8   //Structural Code
9   module SR(
10      input wire r, s,    //2 inputs: set and reset
11      output wire q, q0   //Technically 1 output.
12      );                  //q represents output is high and
13                          //q0 represents output is low.
14
15      //gate var_name(output1, input1, input2);
16
17      nor n1(q0, s, q);   //The logic
18      nor n2(q, r, q0);
19
20  endmodule
```

Figure 2

The Gated SR latch is very similar to the SR latch. The difference is two AND gates. The inputs of the NAND gates are E and either S or R. Output feeds into the SR latch.

```
1   //Gated SR Latch (Asynchronous)
2   /*
3   AUTHOR: Gabe Garves
4   DATE: 11/15/2019
5   FROM: TXST SENIOR DESIGN PROJECT FALL 2019
6   FOR: TEXAS STATE UNIVERSITY STUDENT AND INSTRUCTOR USE
7   */
8   //Structural Code
9   module SR1(
10      input wire s, r, e,     //3 inputs: set, rest, and enable
11      output wire q, q0       //Technically 1 output.
12      );                      //q represents output is high and
13                              //q0 represents output is low.
14      wire [1:0] w;           //connecting wires
15
16      //gate var_name(output1, input1, input2);
17
18      and a1(w[0], r, e);     //The logic
19      and a2(w[1], s, e);
20      nor n1(q0, w[1], q);
21      nor n2(q, w[0], q0);
22
23  endmodule
```

Figure 3

The D flip-flop is very similar to the Gated SR latch with a few tweaks. Instead of an enable, you have a clock. S and R are replaced by D.

```
1   //D Flip Flop (Synchronous)
2   /*
3   AUTHOR: Gabe Garves
4   DATE: 11/15/2019
5   FROM: TXST SENIOR DESIGN PROJECT FALL 2019
6   FOR: TEXAS STATE UNIVERSITY STUDENT AND INSTRUCTOR USE
7   */
8   //Structural Code
9   module DFF(
10      input wire d, clk,      //2 inputs: d and clock.
11      output wire q, q0       //Technically 1 output.
12      );                      //if q=1 represents output is high and
13                              //if q0=1 represents output is low.
14                              //if q=1 && q0=1 forbidden state
15      wire [2:0] w;           //connecting wires
16
17      not no1(w[0], d);       //The Logic
18      and a1(w[1], d, clk);
19      and a2(w[2], w[0], clk);
20      nor n1(q0, w[1] , q);
21      nor n2(q, w[2] , q0);
22
23  endmodule
```

Figure 4

FPGA Implementation:

Using your knowledge from the previous two labs, implement your verilog code into a **symbol** for each latch. Use any of the orange pins in **Figure 8** as a guide when pin planning the latches. Use **Figures 5, 6** and **7** to get an idea of how the **inputs** and **outputs** to connect with the **symbols** in a **block diagram**. Your **outputs** will go the the on-board **LEDs.**

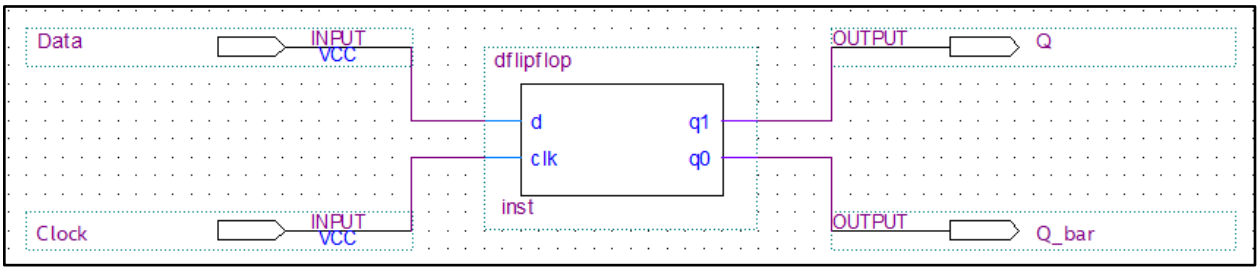


Figure 5

*It is recommended to create a single **.bdf** file where you can insert, compile, and program each latch separately onto the Max10. This will allow you to see each latch function individually and reduce the complexity of the project.*

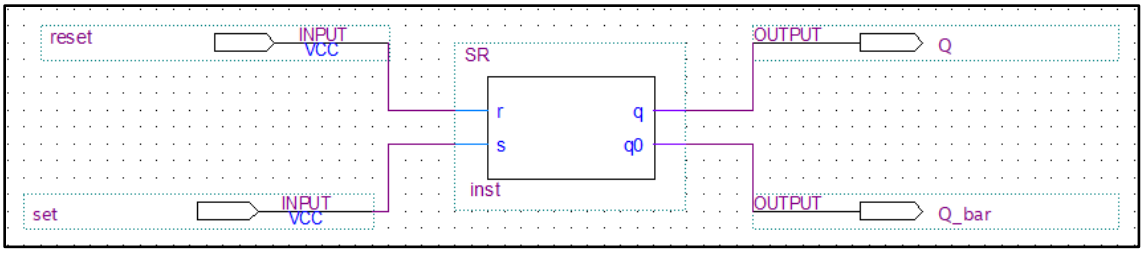


Figure 6

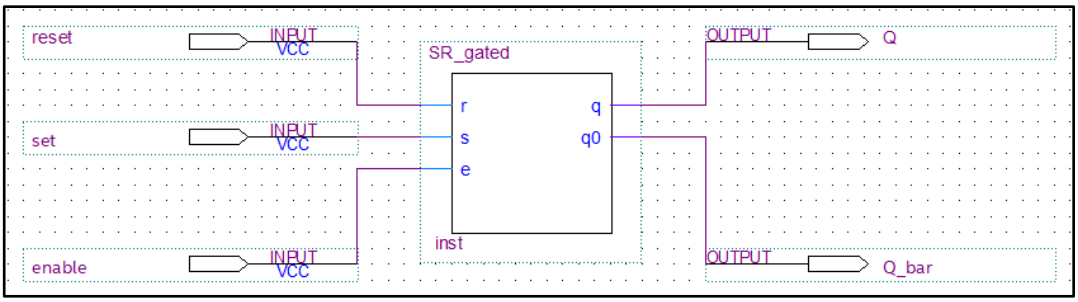


Figure 7

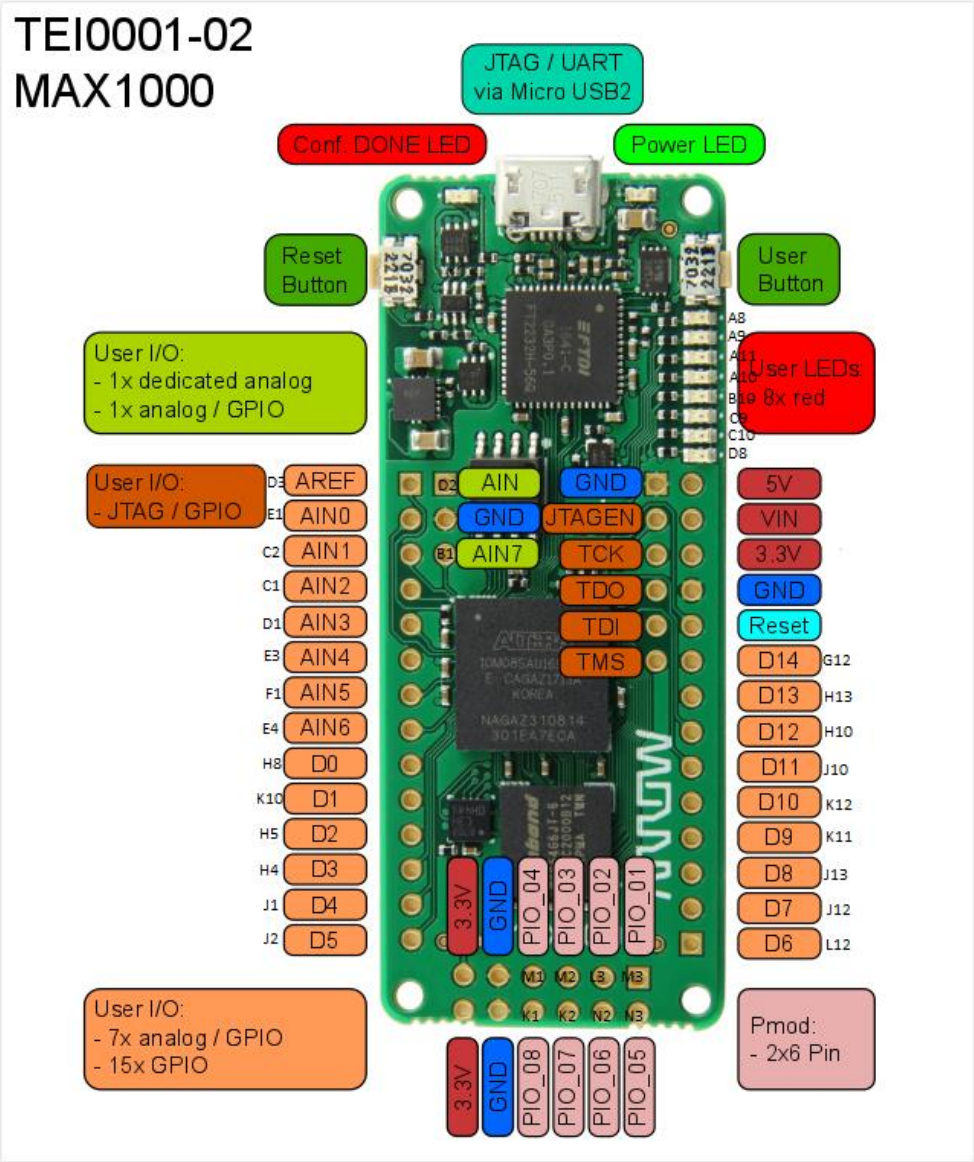

TEI0001-02
MAX1000
JTAG / UART
via Micro USB2
Conf. DONE LED
Power LED
Reset
Button
User
Button
User I/O:
- 1x dedicated analog
- 1x analog / GPIO
User LEDs:
8x red
User I/O:
- JTAG / GPIO
AREF
AIN0
AIN1
AIN2
AIN3
AIN4
AIN5
AIN6
D0
D1
D2
D3
D4
D5
AIN
GND
JTAGEN
AIN7
TCK
TDO
TDI
TMS
5V
VIN
3.3V
GND
Reset
D14
D13
D12
D11
D10
D9
D8
D7
D6
3.3V
GND
PIO_04
PIO_03
PIO_02
PIO_01
3.3V
GND
PIO_08
PIO_07
PIO_06
PIO_05
User I/O:
- 7x analog / GPIO
- 15x GPIO
Pmod:
- 2x6 Pin

Figure 8