# Lab 07 – Shift register

## Dates

## Start this assignment **during** Week 07 (Feb. 20-23) after your Lab 06 has been scored.

## Blackboard upload of written question responses are due at the start of your lab session in Week 08 (Feb. 27-Mar. 02). You will also demonstrate your circuit and answer live questions in your Week 08 lab session.

**>>> IMPORTANT: Read the entire lab document before starting to design the circuit, and complete your design before starting to build. <<<**

## Grading

**Questions – upload answers to Blackboard**

**Question 1 [10 points]** Fill in the missing entries in the truth table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **LOAD** | **SHIFT** | **Q0(t+1)** | **Q1(t+1)** | **Meaning** |
| 0 | 0 | Q0(t) | Q1(t) | Hold |
| 0 | 1 | D-SHIFT | **Q0(t)** | **SHIFT** |
| 1 | 0 | **D0** | D1 | **LOAD** |
| 1 | 1 | **X** | **X** | **X** |

**Question 2 [10 points]** Fill in this table for FF1.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Command | | Inputs | | Output | Logic circuit outputs to drive FF1 | |
| LOAD | SHIFT | **D0** | **D1** | Q1(t+1) | J1 | K1 |
| 0 | 0 | **X** | **X** | **Q1(t)** | **0** | **0** |
| 0 | **1** | **X** | **0**  **1** | **Q0(t)** | **1**  **0** | **0**  **1** |
| 1 | 0 | **0**  **1** | **X** | **D1** | **1**  **0** | **0**  **1** |
| 1 | 1 | Command not allowed | | | | |

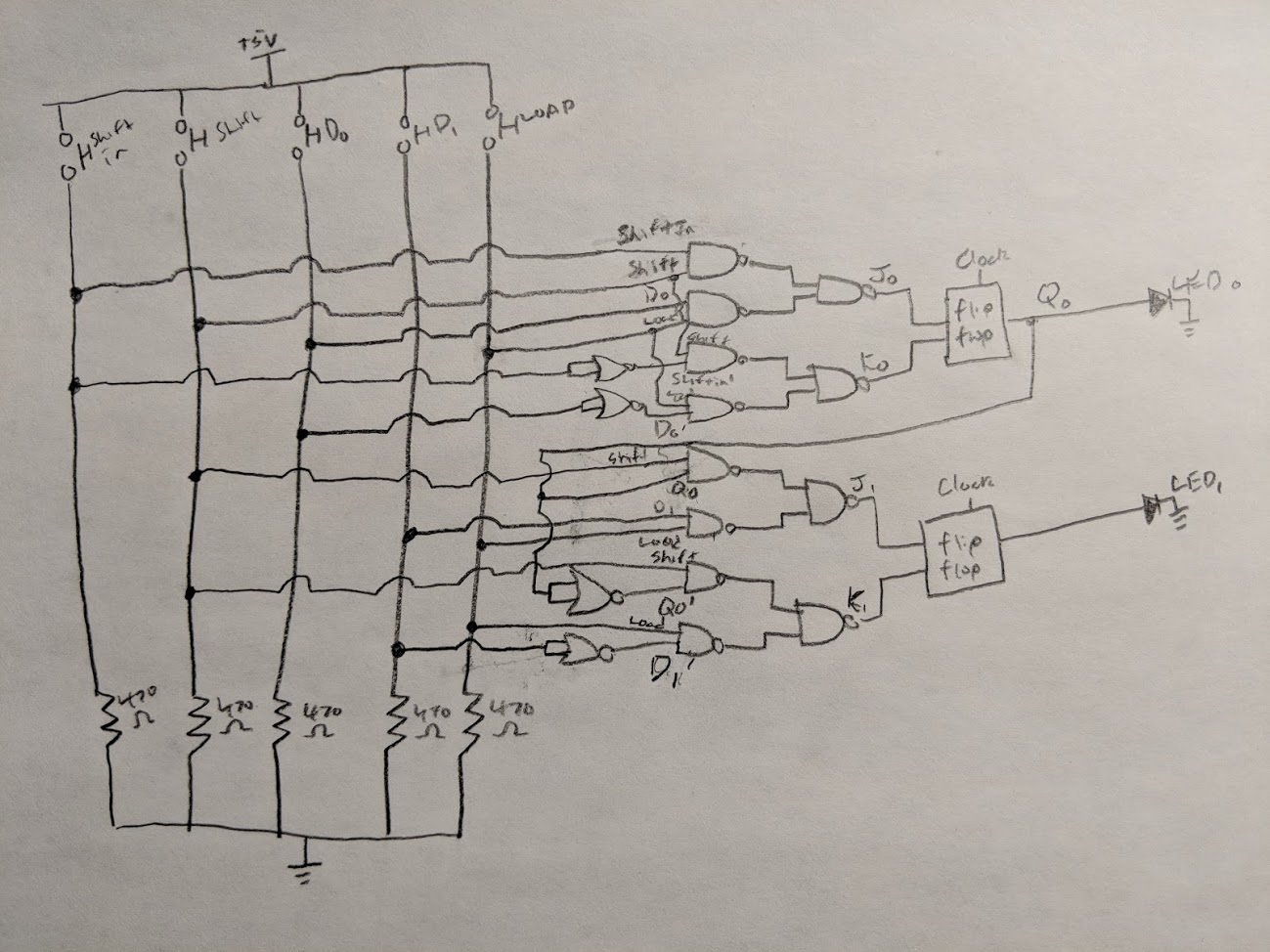
**Question 3 [10 points]** Write the logic equations for K0, J1, and K1 in terms of the shift register control and input signals.

**K0 = (SHIFT and SHIFT\_IN’) or (LOAD and D0’)**

**J1 = (SHIFT and Q0) or (LOAD and D1)**

**K1 = (SHIFT and Q0’) or (LOAD and D1’)**

**Question 3 [20 points]** Draw the circuit diagram for your design. Use box symbols with J and K inputs and a Q output for the flip-flops.



**Circuit demonstration and live questions**

**[20 points]** Demonstrate your 2-bit shift register to your TA in your next lab session.

NOTE: You may find that, like a new video game, the “action” of your shift register is too fast to easily control in the way you wish. If this is so, then you can practice at half the speed by revising the clock circuit to run half as fast using available lab kit components. This little hardware “cheat” is just fine for you to use in this lab, even when demonstrating your circuit. With enough practice you may want to remove the cheat.

**[10 points] If your circuit works correctly, you may link your breadboard for the remaining 10 demonstration points. If your circuit does not work correctly, you may not link.**Link your breadboard with those of ~4 other students anddemonstrate your ~10-bit shift register to your TA. No live questions if you successfully link.

**[Alternative 10 point for circuits that do not work]** Answer two live questions from your TA to earn the final 10 points of the lab in the event that your circuit does not work.

**Instructions**

A K-bit shift register holds K bits of memory and can shift the bits in the register to adjacent bit positions. Suppose we have an 8-bit shift register, and the current bit string in the shift register is:

0110 1001

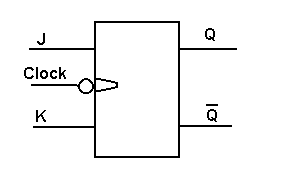
After performing a “left-shift” operation, all of the bits in the register move “left” by 1 bit position, discarding the leftmost bit and injecting a new rightmost bit of value determined by us as designers.If we choose to shift in a zero bit, then the register contents after the left-shift is:

1101 001**0**

Traditionally, the leftmost bit, which is lost to the register circuit, is said to have “fallen into the bit bucket.”

**1. Introduction to the J-K Flip-flop**

J-K flip-flop



There are many types of flip-flops. Flip-flops are improved versions of the latch circuit that have a clock input signal. Most typically, a flip-flop is edge-triggered, like the 74163 counter chip, so that flip-flop action occurs on the rising edge or falling edge of the clock.

The simplest flip-flop in terms of functionality is the D, or data, flip-flop: the clock commands, the value stored by the flip-flop (memory) takes on the value of the data input, input D, and is made available to other circuits as D flip-flop outputs Q and Q’.

The JK flip-flop in the lab kit is more capable than S’R’ or D, as shown in its characteristic table and excitation table.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Characteristic Table** | | | **Excitation Table** | | | |
| **J** | **K** | **Flip-Flop Behavior** | **Q(t)** | **Q(t+1)** | **J** | **K** |
| 0 | 0 | Hold current memory bit value: Q(t+1) = Q(t) | 0 | 0 | 0 | X |
| 0 | 1 | Reset the memory bit to 0: Q(t+1) = 0 | 0 | 1 | 1 | X |
| 1 | 0 | Set the memory bit to 1: Q(t+1) = 1 | 1 | 0 | X | 1 |
| 1 | 1 | Invert current memory bit: Q(t+1) = NOT( Q(t) ) | 1 | 1 | X | 0 |

The 74LS112 JK flip-flop chip data sheet is at <http://www.ti.com/lit/ds/symlink/sn74s112a.pdf> . The flip-flops are negative-edge triggered, meaning their outputs change on the falling edge of the clock signal (as the clock transitions from high to low).

The J-K flip-flops in the 74LS112 have two additional control inputs for each flip-flop: PRESET and RESET, both of which are active-low and asynchronous. If RESET is asserted, the flip-flop output changes to 0; if PRESET is asserted, the flip-flop output changes to 1. Because these inputs are asynchronous, the PRESET/RESET will happen at once (immediately) when they are asserted, rather than waiting for the clock to trigger the operation. Therefore, you should not use these inputs. You may deactivate them by connecting the pins to a constant high voltage (thus de-asserting the active-low inputs).

**2. Specifications for the shift register inputs and control signals**

**>>> Note: Before placing any components on your breadboard, read Section 3. <<<**

In this lab, you will build a 2-bit left-shift register. The 74LS112 chip, containing two J-K flip-flops, will hold the two bits of register memory. You will design the logic circuits to command the JK flip-flops so as to produce the desired shift-register behavior.

**2.1 Clock and Data inputs**

This circuit needs a clock, so use the 555 timer chip to provide a clock with a negative edge interval of 1 second. You may with to practice controlling the register with a slower clock first while you build up skill. Feel free to revise the resistors of the 555 circuit to reduce the clock speed to half of the 1 clock per second design.

There are three data input signals to the shift register: data bit input 0 (D0), data bit input 1 (D1), and the data bit to shift in when shifting, called D\_SHIFT. Produce all three input signals with push-button voltage divider circuits.

**2.2 Shift Register Control Signals**

There are two control inputs to the shift register, also produced by push-button voltage divider circuits. The control signals are LOAD and SHIFT. These control inputs will direct the shift register operation as follows.

**2.2.1 LOAD, an active-high signal**

When LOAD is asserted, the shift register will load the current values of the data inputs, D0 and D1, into the two JK flip-flops at the time of the negative clock edge. D0 is the least significant bit of the register. That is, the output of flip-flop 0 (FF0), Q0, should assume the value of D0, and Q1 of FF1 should assume the value of D1 when the clock transitions from high to low. In equation form,  
Q0(t+1) = D0 and Q1(t+1) = D1with the negative clock edge defining the time t+1.

**2.2.2 SHIFT, an active-high signal**

When SHIFT is asserted, the shift register should perform a 1-bit left shift with every occurrence of a negative clock edge. That is, Q1(t+1) = Q0(t) and Q0(t+1) = D-SHIFT with every negative clock edge. Shifting left loads a bit into FF0 from the D-SHIFT input rather than from D0 and loads a bit into FF1 from FF0 instead of from D1. If D1 is thought of as having the weighted positional significance associated with 21 and D0 has the significance of 20 then it may be convenient to think of the D-SHIFT bit as having significance 2-1. Interestingly, Q0(t) which is typically thought of as an output only, also serves as a data input when shifting left.  
  
The bit shifted out of FF1 must go into a bit bucket that you implement. Don’t overthink what is required to implement a bit bucket.

**2.3 Remaining register functionality and hardware specifications**

* You may assume that SHIFT and LOAD will not be asserted simultaneously. If they are, the operation of your circuit will be undefined.
* When neither LOAD nor SHIFT are asserted, the shift register should retain the currently stored bits for as long as electrical power is supplied. Mathematically, this means:  
  if LOAD’ AND SHIFT’ then Q0(t+1) = Q0(t) and Q1(t+1) = Q1(t).  
    
  The shift register outputs are those from the J-K flip-flops: Q0 and Q1.
* Make the four shift register inputs CLK, D-SHIFT, D0, and D1 visible with green LEDs. Make the two register outputs Q0 and Q1 visible using red LEDs.

**3. Circuit Implementation – IMPORTANT**

This circuit will use quite a few chips and buttons, so a good approach is to design your J and K driver circuits, then count up the chips and buttons, add the JK chip, and then place these on the breadboard before any wires, resistors, capacitors, and LEDs.

**We will be linking the breadboard circuits of ~5 students to demonstrate a ~10-bit shift register. Read the following instructions for placing your circuit design on your breadboard.**  
**The following implementation requirements, a form of hardware interface definition, are imposed:**

1. **Pin 8 of 74112 (JK chip) must be placed in breadboard tie point adjacent to the breadboard end and farthest from the 10 ohm resistor.**
2. **All chips used in your design should be placed as close together as possible so that the circuit can fit on the breadboard.**
3. **The pushbutton inputs must be placed side by side at the 10 ohm resistor end of the breadboard with the button for SHIFT-IN closest of all the pushbuttons to the 10 ohm resistor.**
4. **D-SHIFT and LOAD must be implemented as active high pushbutton inputs.**
5. **The LED for Q0 must be placed at 74112 pin 9, and LED for Q1 at pin 5 and not mounted remotely to the 74112 chip.**
6. **Two wires sufficiently long to link the ground and power rails of one breadboard to the adjacent breadboard must be available. (Only one of the linked breadboards will be connected to USB power.)**
7. **One wire sufficiently long to reach from 74112 Clock 2 pin to the same pin of the same chip on an adjacent breadboard must be available. (This wire is for sharing one clock signal from one breadboard to the next.)**
8. **Each student will know how to quickly disconnect their on-board D-SHIFT signal from their circuit and replace this signal with a connection to pin 5 of the 74112 chip, the Q1 output signal on the breadboard containing the register bit that is one position less significant than the students D0 register bit.**

Breadboards will be positioned so that their long sides are adjacent and JK flip-flop ends align. In this configuration, breadboards may be interconnected using the small “ears” along the long edge closes to tie points column A.

**3.1 Tips for designing circuits to command JK flip-flops**

To design the driving logic circuits that take inputs LOAD, SHIFT, D0, and D-SHIFT and compute the J0 and K0 command signals to input to the FF0 flip-flop, make a truth table showing what should happen when the LSB of the register is commanded. Here is a partial truth table showing this information. What should be entered in the three table cells that are blank?

Table for FF0.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Command | | Inputs | | Output | Logic circuit outputs to drive FF0 | |
| LOAD | SHIFT | D0 | D-SHIFT | Q0(t+1) | J0 | K0 |
| 0 | 0 | X | X | Q0(t) | 0 | 0 |
| 0 | 1 | X | 0  1 | 0  1 | 0  1 |  |
| 1 | 0 | 0  1 | X | 0  1 | **0**  **1** |  |
| 1 | 1 | Command not allowed | | | | |

The table when filled in, and perhaps with K-map assist, shows that the SOP expression for J0 is  
 J0 = (SHIFT and D-SHIFT) or (LOAD and D0).

What is the expression for K0? Make a similar table to work out J1 and K1. Then draw out your complete schematic, count the chips and buttons Bill of Materials (BOM) for your circuit, place the buttons and chips on the breadboard per the hardware interface definition in Section 3, and then connect wires.