

Binary Memory Units: SR Latch, D and JK Flip-Flops Design of a Ripple Counter

Introduction

In the previous experiments we have investigated the development and realization of combinational logic functions. In this experiment we study the binary memory units called latches and flip-flops, which in conjunction with combinational logic, allow the design of sequential logic circuits. First, we investigate three of the most commonly used elements: the SR latch, and the D and JK flip-flops. Then we employ the JK flip-flops in the design of a modulo 16/10 ripple counter. Board Architect is introduced to assign component pin numbers for the hardware lab. Finally, VHDL simulation with ModelSim provides another method for digital system design.

Part MG4 (First Week)

Part MG4A: Latches and Flip-Flops for Sequential Logic Design

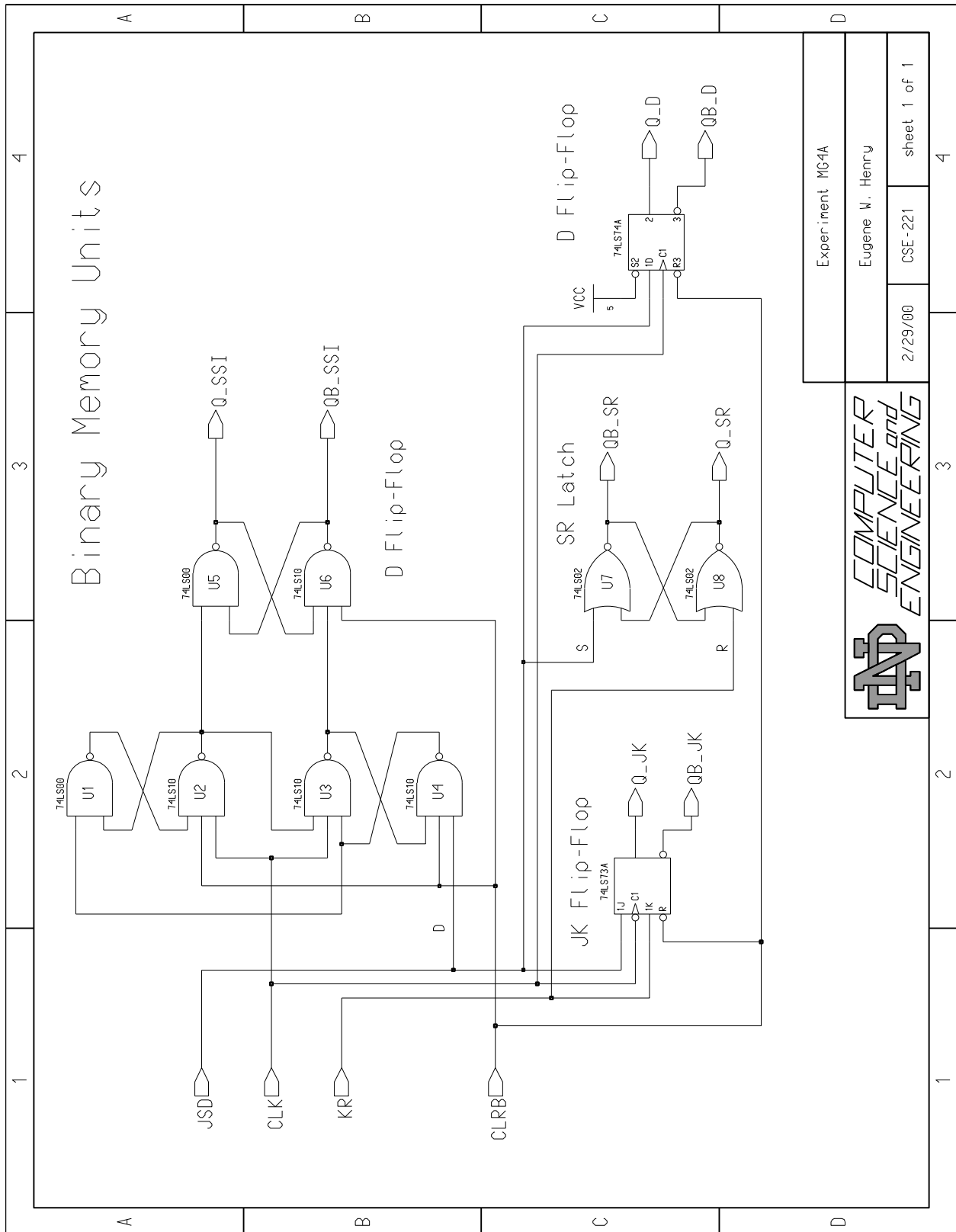
The Design Architect schematic on the next page, labeled Experiment MG4A shows four one-bit memory configurations. The circuit of NAND gates U1 through U6 constitutes a SSI realization of a positive edge-triggered D flip-flop with an asynchronous active-low clear input and complementary outputs. This is the gate-level equivalent of the MSI 74LS74A unit shown at location 4C of the schematic, except that the latter has an active-low preset as an additional input. Cross-coupled NOR gates U7 and U8 form a Set-Reset (SR) latch with complementary outputs. A MSI 74LS73A negative edge triggered JK flip-flop with an active-low clear input and complementary outputs is at location 2C. The 74LS76A JK flip-flop, which has both clear and preset inputs, is used in parts MG4B and HD4.

The voltage-level table for the SR latch shown at the right indicates that when both inputs are high, both outputs are low, contrary to their usual complementary nature. For this reason, and the possibility of oscillation on exiting this state, the designer must assure that the S and R inputs never go high (active) simultaneously.

The voltage-level tables for the D and the JK flip-flops are shown on Page 3. Note that these are clocked flip-flops, which are activated on the edge of the clock input. The other inputs must be stable for specified times before and after the activating clock transition.

Latch with Active-High
Set and Reset

S	R	Q ⁺	QB ⁺
L	L	Q	QB
L	H	L	H
H	L	H	L
H	H	L	L



COMPUTER
SCIENCE and
ENGINEERING

Experiment MG4A	
Eugene M. Henry	
2/29/00	CSE-221
sheet 1 of 1	

74LS74A D Flip-Flop

Positive-Edge-Triggered, with Active-Low Asynchronous Preset and Clear

_CLR	_PRE	D	CLK	Q⁺	QB⁺
L	L	X	X	H	H
L	H	X	X	L	H
H	L	X	X	H	L
H	H	X	≠↑	Q	QB
H	H	L	↑	L	H
H	H	H	↑	H	L

74LS73A (*No _PRE input) and 74LS76A JK Flip-Flops

Negative-Edge-Triggered, with Active-Low Asynchronous Clear and Preset*

_CLR	_PRE*	J	K	_CLK	Q⁺	QB⁺
L	L	X	X	X	H	H
L	H	X	X	X	L	H
H	L	X	X	X	H	L
H	H	L	L	↓	Q	QB
H	H	L	H	↓	L	H
H	H	H	L	↓	H	L
H	H	H	H	↓	\overline{Q}	\overline{QB}
H	H	X	X	≠↓	Q	QB

Simulation with QuickSim II

1. Invoke QuickSim on this design by typing
quicksim \$MGC_HOME/user/logic/MG4A &
2. Click on OPEN SHEET in the QuickSim Palette. A window labeled */:sheet1* containing the schematic diagram from the last page should appear.
3. Create a Trace window for all ports in the schematic. Configure the traces to be in the following order, from top to bottom:
Q_SSI, QB_SSI, Q_D, QB_D, Q_JK, QB_JK, Q_SR, QB_SR, JSD, KR, CLK, CLR
4. Load the forcing waveforms for the four inputs. Use the RMB in the Trace window to activate the pop-up menu, and select **Force > From File**

In the resulting dialog box, enter exactly the following Path name from the keyboard:

\$MGC_HOME/user/logic/MG4A_forces_1000

5. In the SETUP Palette, click on the TIMING MODE button. In the resulting dialog window, change the timing mode on **all** instances to *Full Delays Typ*, and click **OK**. This prepares for simulation with typical gate delays.
6. Perform the simulation for **1000** time units.
7. To prepare printed copy of your simulation output, use the pop-up menu in the trace window to select **Setup > Window**. In the resulting dialog window, enter the following:

<i>Domain label interval</i>	100	<i>Curve height</i>	30
<i>Domain pixels interval</i>	100	<i>Curve spacing</i>	20
<i>Domain area height</i>	40	<i>Margin</i>	10
<i>Name area width</i>	70		

Print the trace window with *Begin domain 0, End domain 1030*

8. Study the simulated output waveforms and discuss the results. Note in particular the difference between the positive and negative edge triggered flip-flops, and between the clocked flip-flops and the latch. Note the oscillatory behavior of the RS latch, and the conditions under which it occurs. Note that delays associated with the NAND implementation of the D flip-flop are different from those of the 74LS76A flip-flop. Which memory unit retains true complementary behavior of the Q and QB outputs for all time?
9. Exit QuickSim without saving.

Part MG4B: A Modulo 16/10 Ripple Counter

The JK flip-flop becomes a T (Trigger) flip-flop when both J and K inputs are held high. That is, under these conditions the flip-flop changes state on each activating clock edge. You are to design a ripple counter using four 74LS76A negative edge triggered JK flip-flops with active low asynchronous preset and clear inputs.

A clock signal with label *CLK* will drive the clock input of the first flip-flop, and the Q output of that flip-flop will drive the clock input of the second flip-flop. The Q output of the second flip-flop will drive the clock input of the third flip-flop, and so on. All of the J and K inputs, as well as the preset inputs will be connected high (to *Vcc*). Thus whenever a flip-flop output goes from high to low, it will trigger a change in the state of the next flip-flop, and the group of four flip-flops will act as a four-bit binary counter.

Another input, labeled *CLRB*, will clear all of the flip-flops to 0 when *CLRB* is low.

A third input, labeled *MOD10*, has no effect when it is low, but when it is high, it will clear the counter whenever the count reaches binary 1010. Thus when *MOD10* is low and *CLRB* is high the counter will cycle through 16 states, whereas when *MOD10* and *CLRB* are both high,

the counter will cycle through only the ten states 0000 through 1001. The output of the circuit is the set of four flip-flop outputs connected via a bus to a single output port labeled *CNT(3:0)*. The output of the first flip-flop (driven by the *CLK*) is the counter's least significant bit, *CNT(0)*.

1. Develop a circuit to meet these specifications, using only the four 74LS76A flip-flops (in two IC packages) and three 74LS10 3-input NAND gates (in one IC package).
2. Make a separate directory under **Private/cse221** named **lab4**. Directly invoke Design Architect by typing **da&** at command line under the directory you just created. Create a schematic called **MG4B** to satisfy the requirements, and obtain a printer plot of your circuit. Create a bus for the *CNT(3:0)* port, and use [TEXT] SEQUENCE TEXT to assign appropriate numbers to the 4x1-ripper R-values. Be sure that 0 corresponds to the least significant bit of the counter.
3. Use QuickSim to simulate your circuit. Load the force file using the following path:

\$MGC_HOME/user/logic/MG4B_forces_3000

Arrange the waveforms in the Trace window in the following order, from top to bottom:

CLRB, CLK, CNT(3:0), MOD10.

Use *Full delay typ* timing mode with run time **3000** ns to verify your counter operation.

4. To prepare printed copy of your simulation output, use the pop-up menu in the trace window to select **Setup > Window**. In the resulting dialog window, enter the following:

<i>Domain label interval</i>	400	<i>Curve height</i>	40
<i>Domain pixels interval</i>	100	<i>Curve spacing</i>	40
<i>Domain area height</i>	40	<i>Margin</i>	10
<i>Name area width</i>	90		

Obtain a printer plot of the Trace window for the domain **0** to **3100** ns.

To observe the ripple effect, reset the *Domain label interval* to **50**, and obtain a printer plot of the Trace window for the domain from **1380** to **1720** ns.

5. Exit QuickSim without saving, and exit Design Architect.

Part MG4C: HD4 Schematic Preparation

To annotate the components in your MG4B schematic with IC package identifiers and pin numbers for use in the hardware lab HD4, we will use the Design Manager to invoke the Design Viewpoint Editor (DVE) to create a Printed Circuit Board (PCB) viewpoint, and then invoke Board Architect (brd_arch) to automatically perform the packaging and pin numbering.

1. Type **dmgr &** to invoke the Design Manager, and then select your *MG4B* component in the navigator window.
2. Use the Right Mouse Button to invoke pop-up item **Open > DVE**
3. In the DVE window, use the LMB to invoke pull-down menu item **Setup > PCB**, and when the setup is finished, terminate DVE with pull-down **MGC > Exit**. Also, close the auxiliary DVE window.

4. In Design Manager, select your MG4B icon, and use the RMB to invoke pop-up menu item **Open > brd_arch**. Note that Board Architect displays the schematic with package identifiers U1, U2, U3, and with all pin numbers in red. To retain this information in the schematic, use pull-down **Miscellaneous > Merge Annotations > All**
5. Use pull-down **File > Print Sheet** to obtain a copy for HD4. Finally, quit using pull-down **MGC > Exit**, without saving.

Note that the power pins Vcc and Gnd are not shown on the schematic, although they must be appropriately connected in the hardware laboratory. As shown on Page 7, the 74LS10 chip has Gnd at Pin 7 and Vcc at Pin 14, which are conventional locations. However, *the 74LS76A flip-flop has unconventional power locations of Pin 5 for Vcc and Pin 13 for Gnd.*

Part MG4D: VHDL Simulation of Counter-Introduction to ModelSim

In this exercise we will fetch a VHDL structural description of the Mod10/16 ripple counter, compile it, and simulate it to become acquainted with the operation of the ModelSim software.

1. In a unix terminal window, type the commands shown below in bold:

```
cd Private/cse221           -- Go to the class directory
mkdir mg4d                -- Create a new directory for this VHDL session.
cd mg4d                   -- Go to the new directory.
cp $MGC_HOME/user/logic/mg4d_ctr.vhd .  -- Obtain the counter source file.
vlib work                 -- Create a library directory for compiled files.
vmap work work           -- Map the logical work file to the physical file work.
vcom mg4d_ctr.vhd        -- Compile the source file.
vsim                     -- Invoke the simulator.
```

2. In the resulting *Load Design* window, select Design Unit **ripple4** and click **Load**.
3. In the *ModelSim* window, select pull-down **View > All**
4. In the *Structure* window, select Package **bit_pack** and scan its contents in the *Source* window.
5. In the *Structure* window, select **ripple4** and scan the VHDL code in the *Source* window. Note that keywords are blue, comments are green, and numerical data are red.
6. In the *Signals* window, set up the traces and input forces as follows:

Select pull-down **View > Wave > Signals in Region**

Select **clrb**, then **Edit > force** with *Value: 1, Delay For: 25, OK*

Select **mod10**, then **Edit > force** with *Value: 1, Delay For: 2000, OK*

Select **clk**, then **Edit > clk** with *Period: 100, and with defaults, OK*

7. At the prompt in the *ModelSim* window, type **run 3000**
8. In the *Wave* window, select **/ripple4/cnt**, then pull-down **Format > Radix > Decimal** and **Format > Radix > Unsigned**. Scan and study the waveforms.
9. In the *ModelSim* window: Select **File > Print Postscript >> Full Range**, and finally, select **File > Quit >> Yes**.

Report

For Part *MG4A*, include the simulation traces from Part 7, and discussion from Part 8.

For Part *MG4B*, comment on your results, including the advantages and disadvantages of a ripple counter vs. a synchronous counter, and the use of the three NAND gates for modulus variation. Include printouts of the schematic from Part 2 and the two simulation traces from Part 4.

For Part *MG4C*, include the schematic from Part 5, and comment on the utility of Board Architect.

For Part *MG4D*, include the waveforms from Part 9, and a listing of the source file *mg4d_ctr.vhd*. Comment on your initial impressions of ModelSim.

Part HD4 (Second Week): Modulo 16/10 Ripple Counter

This hardware laboratory will experimentally verify the operation of the modulo 16/10 ripple counter simulated in Part MG4B and MG4D. **Bring your MG4C schematic to lab.**

1. Insert a 74LS04 hex Inverter into the second component column of the IDL-800 breadboard. Insert two 74LS76A dual JK flip-flops into the third column, and one 74LS10 triple 3-input NAND gate into the fourth column. Pin numbering diagrams for 74LS76A and 74LS10 are shown below. The diagram for the 74LS04 was given in Experiment HD1.
2. Use IDL-800 switches S7 and S6 as the inputs *CLRB* and *MOD10*, respectively. The *CLK* signal is obtained from the output of a 74LS04 inverter whose input is connected to OUT on the Function Generator *after* selecting a Square waveform and setting the Amplitude at 10 o'clock. Use LEDs L7, L6, L5 and L4 to display the outputs *CNT(3:0)*. Also display the outputs on the rightmost 7-segment LED by connecting D1 to GND, L4 to A, L5 to B, L6 to C and L7 to D. Draw these connections, as well as all pin assignments on your notebook diagram. Show also the +5V and ground connections with appropriate pin numbers. Wire the circuits in accordance with your diagram.
3. Set the clock to 2 Hz and observe the behavior of your circuit in response to the other inputs. Record in your notebook the counting sequence for both positions of the *MOD10* switch S6. Note that the 7-segment decoder displays only the 10 decimal digits 0 through 9.
4. Comment on your results, and explain why this is called a ripple counter.

