

Registers and Counters

Design of a Universal Shift Register, Ring Counter and BCD Counter

Introduction

The previous experiment, MG4/HD4, demonstrated the flip-flop as a memory element capable of storing one bit of information, and showed how JK flip-flops could be used in the design of a modulo 16/10 counter. Integrated sets of flip-flops which act under common control to perform specific operations are called **REGISTERS** or **COUNTERS**. A **REGISTER** is a memory device used for storing and manipulating data. Registers may be classified according to how their stored information is entered or removed. A **SERIAL** or **SHIFT** register is one in which the data is entered or removed one bit at a time, and a **PARALLEL** register accepts or transfers all bits of data simultaneously. A synchronous **COUNTER** is a special type of register which periodically progresses through fixed number of states upon successive clock edges.

In this experiment we investigate a "universal" shift register which handles data input and output in both serial and parallel modes. We also design a 4-bit bi-directional ring counter and a modulo 100 BCD ripple counter.

NOTE: Two *Preliminary Preparations* are required, as indicated on Pages 6 and 7.

Part MG5 (First Week)

Part MG5A: Design of a Universal Shift Register

The voltage-level table and conventional schematic symbol for the 74LS194A 4-bit universal shift register are shown on Page 2 and 3, respectively. This unit is called "universal" because it incorporates virtually all of the features a system designer may want in a shift register. The circuit provides parallel inputs, parallel outputs, right-shift and left-shift serial inputs, operating-mode-control inputs, and a direct overriding clear line. In the parallel load mode, the unit functions as a set of four D flip-flops. The two mode control bits, S1 and S0, provide four modes of operation:

- (S1, S0) = 0 0: Retain the present state (do nothing).
- 0 1: Shift Right (in the direction QA toward QD).
- 1 0: Shift Left (in the direction QD toward QA).
- 1 1: Parallel (Broadside) Load of A, B, C, D into QA, QB, QC, QD.

The shift and load operations occur synchronously on the rising edge of the single clock input which is connected internally to all four flip-flops. Study the table to learn the operations.

74LS194A 4-Bit Universal Shift Register

Positive Edge Triggered with Series and Parallel Inputs, and Asynchronous Clear

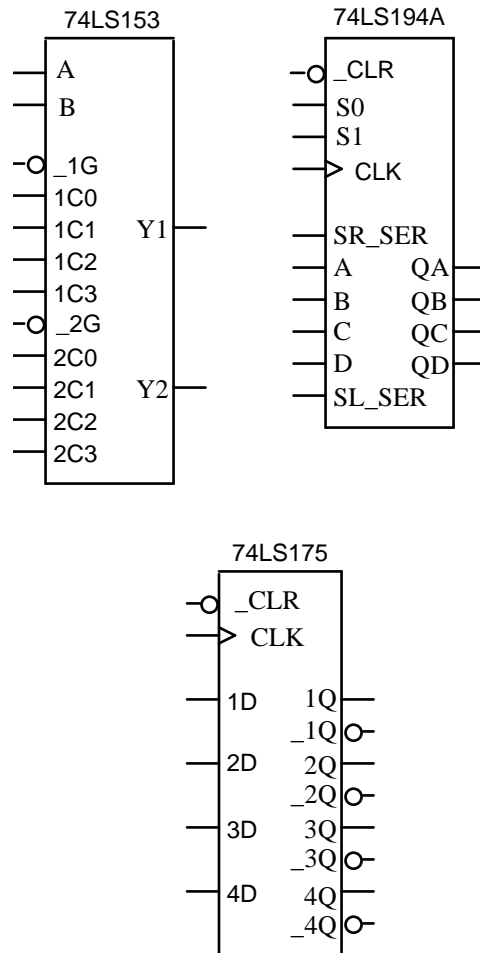
R _CLR	MODE		C4 CLK	SERIAL IN		PARALLEL IN				OUTPUTS			
	S1	S0		SL	SR	A	B	C	D	Q_A⁺	Q_B⁺	Q_C⁺	Q_D⁺
L	X	X	X	X	X	X	X	X	X	L	L	L	L
H	X	X	≠↑	X	X	X	X	X	X	Q _A	Q _B	Q _C	Q _D
H	H	H	↑	X	X	a	b	c	d	a	b	c	d
H	L	H	↑	X	H	X	X	X	X	H	Q _A	Q _B	Q _C
H	L	H	↑	X	L	X	X	X	X	L	Q _A	Q _B	Q _C
H	H	L	↑	H	X	X	X	X	X	Q _B	Q _C	Q _D	H
H	H	L	↑	L	X	X	X	X	X	Q _B	Q _C	Q _D	L
H	L	L	X	X	X	X	X	X	X	Q _A	Q _B	Q _C	Q _D

74LS153 Dual 4-Line to 1-Line Multiplexer

SELECT		DATA INPUTS				EN	OUT
B	A	C0	C1	C2	C3	_G	Y
X	X	X	X	X	X	H	L
L	L	L	X	X	X	L	L
L	L	H	X	X	X	L	H
L	H	X	L	X	X	L	L
L	H	X	H	X	X	L	H
H	L	X	X	L	X	L	L
H	L	X	X	H	X	L	H
H	H	X	X	X	L	L	L
H	H	X	X	X	H	L	H

74LS175 Quad D Flip-Flop with Clear

_CLR	D	CLK	Q⁺	_Q⁺
L	X	X	L	H
H	X	≠↑	Q	_Q
H	H	↑	H	L
H	L	↑	L	H



A VHDL Entity and behavioral Architecture for the universal shift register are shown below. Verify that these descriptions correspond to the symbol and tabulated behavior shown in pages 2-3.

```
-- ushift4.vhd    E. Henry 3/11/00
-- Model for 4-bit universal shift register 74LS194A
-- CSE-221 Experiment MG5

library BITLIB;
use BITLIB.bit_pack.ALL;

ENTITY ushift4 IS
  PORT (clk, clrb, sl_in, sr_in : in bit;
        mode : in bit_vector ( 1 downto 0 );
        d    : in bit_vector ( 3 downto 0 );
        q    : inout bit_vector (3 downto 0 ));
END ushift4;
```

```

ARCHITECTURE behav OF ushift4 IS
BEGIN
  PROCESS (clk, clrb)
  begin
    -- Asynchronous, active-low Clear input:
    if clrb = '0' then q <= "0000" ;
    -- Rising edge-triggered D flip-flops:
    elsif clk'event and clk = '1' then
      case mode is
        -- "Do Nothing" mode: retain current flip-flop outputs
        when "00" => null;
        -- Shift Right Serial Input mode:
        when "01" =>
          q <= (q srl 1) or (sr_in & "000") ;
        -- Shift Left Serial Input mode:
        when "10" =>
          q <= (q sll 1) or ("000" & sl_in) ;
        -- Parallel (Broadside) Load mode:
        when "11" => q <= d ;
      end case;
    end if;
  end process;
END behav;

```

a) Simulation of Shifter Behavioral Architecture with ModelSim

In this exercise we will fetch, compile and simulate the VHDL source file shown above,

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

```

mkdir MG5 -- Create a new directory for experiment MG5.
cd MG5 -- Go to the new directory.
mkdir mg5a -- Create a new directory for Part A, VHDL.
cd mg5a -- Go to the new directory.
cp $MGC_HOME/user/logic/mg5a/* . -- Get all of the mg5a files.
vlib work -- Create a library directory for compiled files.
vmap work work -- Map the logical work file to the physical file work.
vcom ushift4.vhd -- Compile the shifter entity and behavioral
architecture.
vsim -- Invoke the simulator.

```

2. In the resulting *Load Design* window, select Design Unit **ushift4** and click *Load*. The macro file *startup.do* will execute automatically to run a simulation. Use the scroll bar in the *wave* window to observe the behavior of the output *q* as a function of time for successive values of mode = 11 (Parallel Load), 01 (Shift Right), 10 (Shift Left), and 00 (Retain Value).
3. In the *Wave* window, select pull-down **File > Print Postscript >> Full Range > OK**.
4. In the ModelSim window select pull-down **File > Quit >> Yes**

b) Simulation of Shifter Structural Architecture with ModelSim

To obtain a better understanding of the 74LS194A shift register, and to gain further familiarity with multiplexers and flip-flops which you have studied in previous experiments, you are to configure a VHDL structural architecture using two 74LS153 Dual 4-to-1 Multiplexers and one 74LS175 Quad D Flip-Flop so that your circuit with these three ICs accepts the same inputs and performs the same operations as a single 74LS194A universal shift register.

The voltage-level tables and Mentor Graphics conventional symbols for the 153 and 175 chips are given on Page 2 and 3 of this experiment. You have already fetched the VHDL source files *2mux4x1.vhd* and *quaddff.vhd* which are shown below.

Study these files to understand how they describe the symbols and tabulated behavior shown on Pages 2-3.

```
-- dualmux4x1.vhd    E. Henry 3/15/00
-- Model for Dual four to one multiplexer 74LS153
-- CSE-221 Experiment MG5

library BITLIB;
use BITLIB.bit_pack.ALL;

ENTITY dualmux4x1 IS
  PORT (gb1, gb2 : in bit;
        sel  : in bit_vector ( 1 downto 0 );
        d1, d2 : in bit_vector ( 3 downto 0 );
        y1, y2 : out bit);
END dualmux4x1;

ARCHITECTURE behav OF dualmux4x1 IS
BEGIN
  PROCESS (sel, gb1, gb2, d1, d2)
  begin
    if gb1 = '1' then y1 <= '0';
    else y1 <= d1(vec2int(sel));
    end if;
    if gb2 = '1' then y2 <= '0';
    else y2 <= d2(vec2int(sel));
    end if;
  end process;
END behav;

-- quaddff.vhd    E. Henry 3/15/00
-- Model for Quad D Flip-Flop 74LS175
-- CSE-221 Experiment MG5
```

```
library BITLIB;
use BITLIB.bit_pack.ALL;

ENTITY quaddff IS
  PORT (clkf, rstb : in bit;
        df : in bit_vector (3 downto 0);
        qf : inout bit_vector (3 downto 0));
END quaddff;

ARCHITECTURE behav OF quaddff IS
BEGIN
  PROCESS (clkf, rstb)
  begin
    if rstb = '0' then qf <= "0000";
    elsif rising_edge(clkf) then qf <= df;
    end if;
  end process;
END behav;
```

Preliminary Preparation 1:

Sketch a schematic diagram of the design as specified on Page 5, using the components and signal ports given for the *dualmux4x1* and *quaddff* VHDL entities, and bring a written copy of your annotated schematic diagram to the laboratory.

Use your schematic to determine how to fill in the appropriate entries at the four incomplete lines of VHDL code for the corresponding structural architecture listed on Pages 7-8.

-
1. Compile the multiplexer and quad flip-flop entities and behavioral architectures by executing the following commands:

```
vcom dualmux4x1.vhd
```

```
vcom quaddff.vhd
```

2. Use your results from *Preliminary Preparation 1* to edit the file *ushift_struct.vhd*, shown on Pages 7-8, to complete the four lines of code indicated by comments. You have already copied the incomplete file into your directory. Obtain a printout of your modified file.
3. After you have appropriately edited the shifter structural architecture code, compile it, and invoke the simulator:

```
vcom ushift4_struct.vhd
```

```
-- Compile the structural architecture
```

```
vsim
```

```
-- Invoke the simulator
```

4. In the resulting *Load Design* window, expand Design Unit **ushift4**, select **struct**, and click *Load*. The macro file *startup.do* will execute automatically to run a simulation. Use the scroll bar in the *wave* window to observe the behavior of the output *q* as a function of time for successive values of *mode* = 11 (Parallel Load), 01 (Shift Right), 10 (Shift Left), and 00 (Retain Value). Check that the results agree with simulation of the behavioral architecture.
5. In the *Wave* window, select pull-down **File > Print Postscript >> Full Range > OK**.
6. In the ModelSim window select pull-down **File > Quit >> Yes**

```
-- ushift4_struct.vhd  E. Henry  3/21/00
-- Structural architecture for universal register 74LS194A
-- CSE-221 Experiment MG5

ARCHITECTURE struct OF ushift4 IS

-- Structural configuration uses one 74LS175 Quad D flip-flop
-- and two 74LS153 Dual 4 to 1 Multiplexers

COMPONENT quaddff
  port(clkf, rstb : in bit;
        df : in bit_vector (3 downto 0);
        qf : inout bit_vector (3 downto 0));
END COMPONENT;

COMPONENT dualmux4x1
  port (gb1, gb2 : in bit;
        sel : in bit_vector (1 downto 0);
        d1, d2 : in bit_vector (3 downto 0);
        y1, y2 : out bit);
END COMPONENT;

signal x3, x2, x1, x0, y : bit_vector(3 downto 0);
signal low : bit;

BEGIN
  low <= '0';

  x3 <= d(3) & q(2) & sr_in & q(3);
  x2 <=                                     ; -- complete these
  x1 <=                                     ; -- three lines
  x0 <=                                     ; -- of code

  MUX1: dualmux4x1 PORT MAP (low, low, mode, x3, x2, y(3), y(2));

-- Complete the following line of code:

  MUX2: dualmux4x1 PORT MAP ( , , , , , , );
```

```
    DFF: quaddff  PORT MAP (clk, clrb, y, q);  
  
END struct;
```

Part MG5B: A 4-Bit Synchronous Bi-directional Ring Counter

A ring counter consists of a shift register configured so that the bit which is shifted out of one end of the register is inserted at the other end. The counter is initialized so that exactly one bit in the register is set to 1, and all other bits are reset to 0. On successive clock positive edges, the 1 bit cycles around the register "ring." A bi-directional ring counter has a mode control which allows control of the shift direction. In this instance the output of the rightmost flip-flop is connected to the D input of the leftmost flip-flop, and the output of the leftmost flip-flop is connected to the D input of the rightmost flip-flop. The parallel load mode of this device may be used to establish the appropriate initial conditions for the ring counter. This design uses only one component: the 74LS194A universal shift register.

Preliminary Preparation 2:

Perform the design as specified in the previous paragraph, using the 74LS194A universal shift register and signal ports given in Part **a4** below, and bring a written copy of your resulting annotated schematic diagram to the laboratory.

a) Schematic Creation with Design Architect

1. Change to directory MG5, invoke Design Architect and select OPEN SHEET with **MG5B** for the Component Name.
2. Obtain these circuit components from the library: *ground*, *portin*, *portout*, *Vcc*, and *74LS194A*.
3. Place three input ports on the left side, and one output port on the right side of your schematic. Assign the following names to your ports:

_RESET, CLK, MODE(1:0), COUNT(3:0).

Note that you **MUST** use these names to conform with the given force file for Quicksim.

4. Make appropriate copies of the input ports elements, and then "wire" the elements together in accordance with your design. Use buses and associated rippers for the output, and for the input MODE(1:0).
5. Check and save the completed schematic sheet.
6. Obtain a printed copy of the schematic.
7. Minimize the Design Architect window.

b) Simulation with QuickSim II

1. Invoke **quicksim MG5B &**
2. Click on OPEN SHEET in the QuickSim Palette to obtain a schematic window.
3. Create a trace window for all ports in the schematic. Configure the traces to be in the following order, from top to bottom: *_RESET*, *MODE(1:0)*, *CLK*, *COUNT(3:0)*.
4. To obtain the force file for this circuit, invoke either pop-up (*QuickSim*)>*Force*>*From File...* or pull-down *Setup > Force > From File...* In the resulting Load Forcefile dialog window, type exactly **\$MGC_HOME/user/logic/MG5B_forces_3000**
5. Run the simulation for **3000** nanoseconds.
6. Obtain a printer plot of the waveforms. To obtain an attractive plot which fills the page, invoke (*QuickSim*)>*Setup*>*Window* and then enter the following values in the resulting Setup Trace Window dialog box:

<i>Domain label interval</i>	400	<i>Curve height</i>	50
<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>	100		

Select *File > Print > Active Window* Enter the values for the start and end times of the plot as follows: *Begin Domain* **0** *End Domain* **3100**
7. Close Quicksim without saving. Study the output waveforms and discuss the results.

Part MG5C: A Modulo 100 BCD Ripple Counter

The schematic on the last page of this experiment includes a two-digit Binary-Coded-Decimal counter using a dual BCD counter in a single 74LS390 IC package. Each BCD digit is implemented with the cascade of a 1-bit divide by 2 counter and a 3-bit divide by 5 counter, producing a ripple counter that will count from decimal 00 to 99.

1. Use design architect to implement this counter in directory MG5 as component MG5C, with inputs **CLR** and **CNTB**, and output **COUNT(7:0)**. Check and save your schematic, and obtain a printed copy.
2. Use quicksim to simulate your counter over a time interval of 0 to 11200 ns, with a clock period of 100 ns for the input CNTB. Obtain printouts of the traces for CLR, CNTB and COUNT(7:0) for the time intervals 0 to 1300 ns, and 8800 to 10400 ns.
3. Quit quicksim without saving, and quit design architect.

MG5 Report

Your written report for MG5 should contain a commentary on the relative advantages and disadvantages of the registers and counters studied in this experiment, with reference to the following schematics and simulation traces which are required as part of the report:

1. ModelSim traces of the behavioral Shift Register model from MG5A a)3 on Page 4.
2. VHDL source listing of your modified ushift_struct.vhd from MG5A b)2 on Page 6.
3. ModelSim traces of the structural Shift Register model from MG5A b)5 on Page 7.
4. Schematic of the Bi-directional Ring Counter from MG5B a)7 on Page 9.
5. Quicksim traces of the Bi-directional Ring Counter from MG5B b)6 on Page 9.
6. Schematic of the Modulo 100 BCD Counter from MG5C 1) on Page 9.
7. Two Quicksim traces of the Modulo 100 BCD Counter from MG5C 2) on Page 9.

Also comment on any difficulties you encountered in using the Mentor Graphics tools, and explain how you resolved the problems.

Part HD5 (Second Week): *Bring your MG5B & MG5C Results to Lab*

The objectives of this hardware laboratory are to experimentally verify the operation of the Bi-directional Synchronous Ring Counter which was simulated in Part MG5B of this experiment, and the 2-digit BCD Modulo 100 Counter from Part MG5C. The BCD counter output will be viewed on the two 7-Segment LED displays of the IDL-800 Logic Lab.

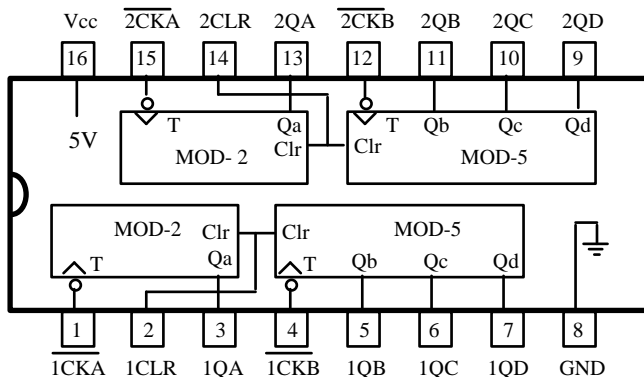
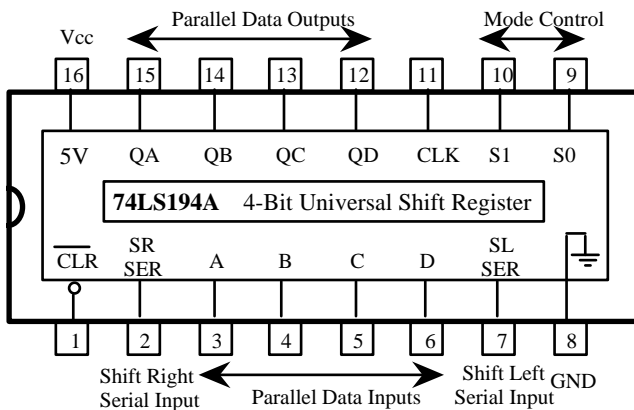
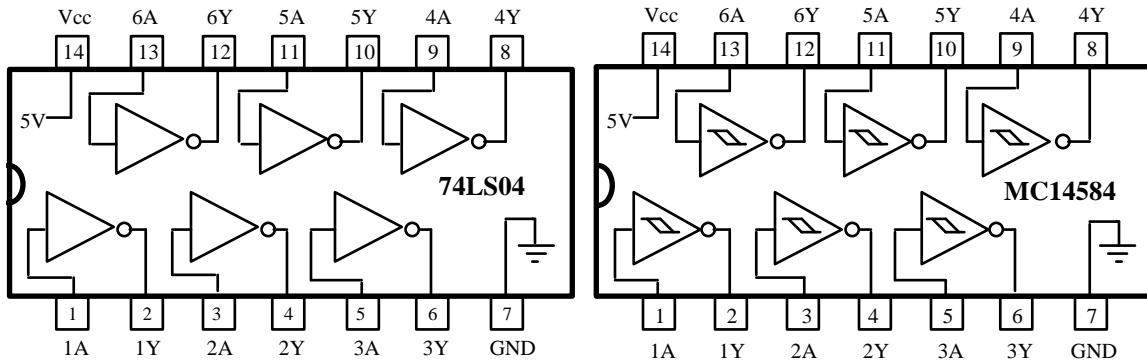
Although the IDL-800 Logic Lab has two 7-Segment LED displays, it has only one BCD to 7-segment decoder. The decoder inputs are labeled A, B, C and D, with A being the least significant bit. Each decoder output is connected through a DIP switch and 330 ohm resistor to the diode anode of the corresponding segment in both LED displays. In each display, the diode cathodes are connected together (Common Cathode) and brought out to a connector below the display. These C/C connectors are labeled D2 and D1, from left to right. In order for a decoded segment to be lighted, the corresponding DIP switch must be on, and the segment C/C connection must be grounded.

To display two digits with one decoder, we must multiplex the BCD digit values to the decoder input, and ground the appropriate C/C terminal only when the data for its digit is being decoded. If we switch between the two digits at a rate of about 50 Hz, then our visual persistence will make both displays appear to be continuously illuminated. We will use the IDL-800 Function Generator to produce the 50 Hz multiplexing control. Two inverters in parallel will be used to provide the current to drive each C/C terminal to ground.

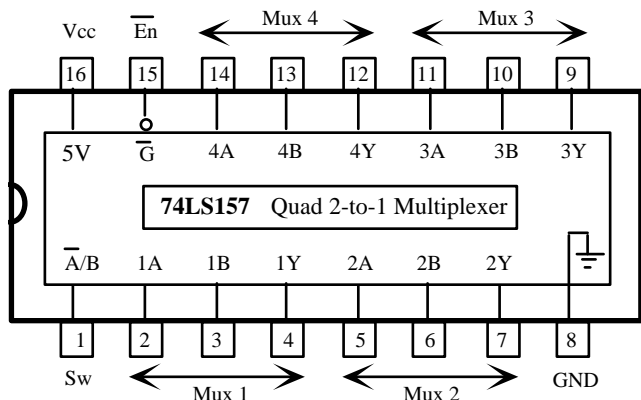
IC package diagrams for the ICs used in this lab are given on Page 13. A complete schematic diagram showing all pin numbers is given on Page 14. A slow clock (approximately 0.6 Hz) to drive the counters will be obtained from a CMOS inverting Schmitt Trigger with associated resistors and capacitor, as shown in the schematic.

1. Insert the *74LS04* Hex Inverter, *74LS157* Quad 2-to-1 Multiplexer and *74LS390* Dual BCD Counter, in that order from top to bottom in the leftmost component column of the IDL-800 breadboard. These parts constitute the 2-digit BCD modulo-100 counter and multiplexing display controller. SW7 will be used as the asynchronous clear input for the counter.
2. Insert the *74LS194A* Universal 4-bit Register and the *MC15484* Hex Inverting Schmitt Trigger, in that order from top to bottom in the third component column of the IDL-800 breadboard. These parts constitute the 4-Bit Bi-directional Ring Counter and 3 Hz clock. SW2 will be the asynchronous active-low clear for the counter, while SW1 and SW0 will be the mode control. The counter output will be displayed on LEDs L7, L6, L5 and L4.
3. Wire the circuits as shown in the schematic on Page 14. Include the 68K resistor, and the 47uF capacitor with polarity as shown. Note that all IC pin numbers are on the diagram.
4. Set the Function Generator for a 50 Hz square-wave output with Amplitude at 10 o'clock.
5. Turn SW7 on to reset the 2-Digit BCD Counter, and then turn it off to verify correct counting with decimal display from 00 through 99. Lower the Function Generator frequency to 1 Hz to observe the display multiplexing.
6. Turn SW2 off to clear the Ring Counter, and then turn it on and use SW1 and SW0 to check the four modes of operation. Start with Load mode 11 to initialize the counter, and then check 00, 01 and 10 for Hold, Shift Right and Shift Left modes, respectively.

7. Use the Tektronix logic analyzer as directed to observe actual responses equivalent to your simulated results from MG5B.
8. Write comments in your lab notebook to confirm your experimental results. Also record difficulties encountered, with steps taken to resolve them.



74LS390
Dual 4-Bit
Decade
Counter



<u>G</u>	<u>A/B</u>	A	B	Y
H	X	X	X	L
L	L	L	X	L
L	L	H	X	H
L	H	X	L	L
L	H	X	H	H

