Experiment 3
Combinational Circuit Using SSI Devices

1. Purpose
   - To observe the propagation delay and the glitch of digital circuits.
   - To use SSI devices to implement combinational circuits.

2. Prelab
   - Determine the logic expression of \( f \) in section 6.2.
   - Design \( f_{\text{learn}} \) circuit of section 6.5. Use Max Plus II to draw the circuit diagram of 6.5(e) and then do simulation. Each group member has to do his/her own work. Bring the design paper, circuit diagram printout and simulation printout to lab and let instructor/TA initial it. Attach the initialed copies to the end of your report.

3. Reference
   Sections 3.6, 4.3 and 4.5 of *Digital Design: Principles and Practice*

4. Discussion

4.1. Propagation delay of digital circuits

One major difference between an ideal gate (i.e., an operation in Boolean algebra) and a physical gate is that a physical gate cannot change its output value instantaneously. It will take some time to change. The *propagation delay* of a circuit is the amount of time that it takes for a change in the input signal to produce a change in the output signal. The propagation delay of an inverter is shown below. The \( t_{\text{PHL}} \) and \( t_{\text{PLH}} \) are the propagation delays output changes from *High to Low* and *Low to High* respectively.

![Propagation Delay Diagram](attachment:image.png)

In a real circuit, the waveforms cannot change abruptly and they look more like the following:

![Real Waveform Diagram](attachment:image.png)

The voltage threshold for LS TTL devices is 1.3V.

In a real combinational circuit, there are may be many components in a signal path, and it is very difficult to consider all the possible cases of high-to-low and low-to-high transitions. We
frequently just consider the worst-case delay of a component, which is defined as the maximum of \( t_{PHL} \) and \( t_{PLH} \).

In addition to postpone the output response, the propagation delays of multiple paths sometimes may generate a short transient faulty pulse in output signal. The pulse is known a glitch. Although glitches can be avoided by adding extra circuitry, it is not worth the effort. The normal approach is to wait until the circuit is settled and then read the output value.

### 4.2. Testing combinational circuits using Logic analyzer

One way to test your combinational circuit is to connect the inputs/outputs of a circuit to the switches/LED indicators of CADET and manually go through each test pattern. It is slow and tedious and also cannot observe any timing information (such as delay and glitches). A better way to do it is to utilize a logic analyzer and a test pattern generator shown below:

![Logic Analyzer Diagram]

The test pattern generator is an instrument that can be programmed to generate the desired testing vector. The output of the test pattern generator and the response of the circuit are connected to logic analyzer to observe and verify the operation of the circuit under test.

Our lab is not equipped with test pattern generators. We replace it with a 74163 counting circuit, similar to the circuit implemented in project test 2 of Experiment 2. The circuit is wired in free-running mode and works as a mod 16 counter:

![74163 Counter Diagram]

The counting patterns of output signals \( Q_d \), \( Q_c \), \( Q_b \) and \( Q_a \) are \( 0000, 0001, 0010, 0011, \ldots, 1110, 1111 \). Note that the patterns include all the possible combinations of four input signals, and thus can be used as a pattern generator to exhaustively test a circuit with 4 input signals. Similarly, the counting patterns of output signals \( Q_c \), \( Q_b \) and \( Q_a \) are \( 000, 001, 010, 011, \ldots, 110, 111 \) and they can be used to exhaustively test a circuit with 3 input signals.

### 4.3. Propagation Delay of Max Plus II Simulation

Max Plus II software is intended to do logic synthesis and simulation for programmable logic devices. Although it can be used to simulate TTL circuits, there are few limitations.
• First, the propagation delays shown in Max Plus II simulation are the delays of the programmable devices rather than those of LS TTL devices. Thus, the delays in simulation will be different from the delay of physical TTL implementation. Also, due to the variation in delay, glitches may occur in simulation but not in TTL circuits and vice versa.

• Second, Max Plus II may do some logic optimization and removes unnecessary circuits during the compiling process. For example, Max Plus II will remove two cascading since they are logically redundant.

In summary, you can use Max Plus II to check the “functional correctness” of TTL circuits (i.e., when the circuits reach steady state), but not the “transient phenomena” (i.e., anything related to propagation delay, such as glitches).

5. Components

741 S00, 741 S04, 741 S10, 741 S20, 741 S37, 741 S163, and one 390 Ω resistors

6. Procedures

6.1. Propagation delay of a NAND gate

(a). Use one NAND gate of 74LS00 and wired it as follows:

![NAND gate diagram]

Note that there is a resistor connected to the output of the NAND gate. It is used to simulate the load that the gate is driving.

(b). Set the function generator to about 10K Hz and connected it to input of the NAND gate.

(c). Connect the input and output of the NAND gate to the A1 and A2 channels of HP54645D.

(d). Adjust the oscilloscope’s setting so that the high-to-low transition of NAND output can be clearly observed. Determine t\text{PDL} of the NAND gate (recall that 1.3V is used as threshold in LS TTL device).

(e). Repeat parts (c) and (d) but use the D0 and D1 channels of HP54645D.

6.2. Glitch

(a). Use one 74LS00 device to implement the following circuit:

![Glitch circuit diagram]

(b). Set the function generator to about 10K Hz and connected it to the input.

(c). Connect the input a and output f of the circuit to the A1 and A2 channels of HP54645D.
(d). Adjust the oscilloscope’s setting so that the glitch of output can be clearly observed.
Determine the width (in term of ns) of the glitch (use 1.3V as the threshold).

(e). Connect the signals a, b, c, d and f of the circuit the D0, D1, D2, D3 and D4 channels of HP54645D. Adjust the logic analyzer setting so that the glitch of output can be clearly observed. Determine the delays of among these signals.

6.3. Verification of DeMorgan’s law
(a). Functions f and g have inputs a and b, and are defined as follows:
\[ f = (a \cdot b)' \quad g = a' + b' \]
DeMorgan’s law states that \( f = g \).
(b). Use one 74LS00 to implement \( f \), and use one 74LS04 and one 74LS32 to implement \( g \).
(c). Use two switches of CADET as the inputs. Connect \( f \) and \( g \) to two LED indicators. Verify that DeMorgan’s law is correct.

6.4. Testing of 74LS10
(a). Implement the test pattern generator the circuit described in section 4.2.
(b). Use the generator to test one of NAND gates of 74LS10. Use logic analyzer to observe the inputs and response of the NAND gate.

6.5. Implementation of a comparator
(a). \( f_{\text{great}} \) is a 1-output, 4-input function that compares two unsigned 2-bit numbers; and outputs a "1" when the first number is greater than the second number. Let the inputs be \( A_1, A_0, B_1 \), and \( B_0 \), where \( A_1A_0 \) represents the first number and \( B_1B_0 \) represents the second number.
(b). Derive the truth table of \( f_{\text{great}} \), use Karnaugh map to minimize this function and express it in sum-of-product form. Use Inverters and NAND gates to implement the minimized expression.
(c). Use minimal number of Inverter (74LS04) and NAND (74LS00, 74LS10, 74LS20) chips to implement this circuit.
(d). Use switches and LED indicators of CADET to verify its operation.
(e). Connect the circuit to 74163 test pattern generator and use the logic analyzer to verify its operation. Make sure that the channels of logic analyzer are properly ordered so that the data can be easily observed.