RUTGERS UNIVERSITY
Department of Electrical and Computer Engineering

14:332:233

DIGITAL LOGIC DESIGN LABORATORY

Fall 2012

Contents

1 LABORATORY No. 1
Introduction to Hardware

To make your experience in the Digital Logic Design Laboratory a pleasant one please follow the recommendations in this introductory section throughout the semester. You should become familiar with the protoboard and build the input-control/output-display unit which will be used in all five experiments. To make debugging easier obey as much as possible the rules for wiring when implementing a circuit.

1.1 Equipment

This laboratory runs on a shoestring. Only a power supply and a logic probe are needed.

Power Supply. All the integrated circuits used in the experiments are from the 74LS family of TTL circuits and thus require a supply voltage of +5V. Your power supply may provide several adjustable voltages, however, almost certainly will have a separate, nonadjustable +5V output. You should *only* connect the protoboard to this output, never to an adjustable one which can burn your circuit if the voltage is accidentally increased.

Most power supplies have a switch to cut off the voltage from the output without turning off the device. *To avoid creating short-circuits you must always cut off the supply voltage from the protoboard when introducing or removing wires!!!*

The supply voltage should be connected to the protoboard with the ground (GND) to the black jack of the protoboard and the +5V to the red jack of the protoboard.

Logic Probe. To analyze and debug your circuit you will use the LP10A logic probe manufactured by Wavetek. The probe can display the logic level at any point in the circuit by touching the point with the metal tip. For logic values 1 (high) the red LED is on, for logic values 0 (low) the green LED is on. The switches on the logic probe should be set to

- TTL
- MEM

since we are using a TTL family and will not deal with short impulses.

The power is supplied to the logic probe from the protoboard. The wires connecting the supply bars of the protoboard to the input jacks (see Section 1.2) should have a short stripped end at the jacks. The black alligator clip of the logic probe is attached to the wire through the GND jack and the red alligator clip to the wire through the +5V jack of the protoboard. *Make sure that the two stripped ends are well separated and once the alligator clips are attached they are completely covered. Of course you should attach the probe with the supply voltage cut off.*

1.2 Protoboard

You will build the circuits on the provided protoboard. None of the experiments require more room than one protoboard, but you should be neat in mounting the ICs in the later experiments. At the end of a laboratory session you should remove the ICs and the wires but keep the input-control/output-display unit (see Section 1.3) which will be needed again in the next session.

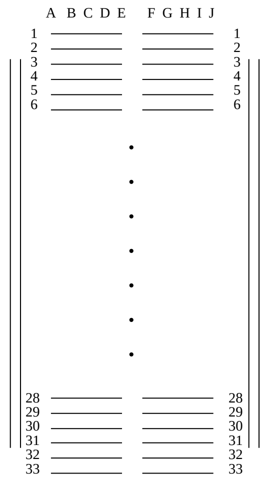

Figure 1: The connections on a block of the protoboard.

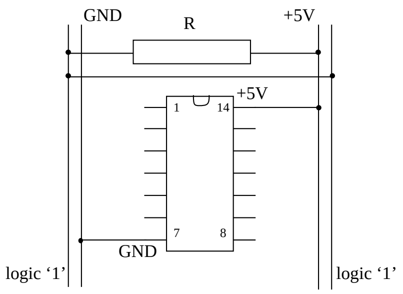

Figure 2: Recommended mounting of an IC and allocation of the supply bars.

The protoboard has four identical blocks. Each block is pre-wired as shown in Figure 1. It is recommended to use the vertical supply bars for the supply voltages (GND, +5V) and for the logic ‘1’ signal as shown in Figure 2. The supply voltages of the first block (containing the input-control/output-display unit, see Section 1.3) should be connected to the jacks, while the remaining three blocks should be powered by connecting their supply bars to the first one. Since these connections will remain on the board arrange them neatly and use wires of adequate length.

The horizontal bars allow connecting several wires together. It is recommended to mount all the ICs with the key facing the top of the protoboard (where the jacks are). The pins on left side of the IC are mounted in the E column, and the pins on the right side in the F column.

The ICs used in the experiments are either 14 (as in Figure 2) or 16 pin packages. In both cases the bottom left pin (7 or 8) is the GND and the top right pin (14 or 16) is the +5V. Often you will need to connect an input into a gate to GND or logic ‘1’. Since two adjacent supply bars wired to GND and +5V increase the chances of creating a short-circuit, the other two bars should be connected through a pull-up resistor of $R = 620\Omega$ to +5V as shown in Figure 2. The GND inputs can be wired as the rest of the connections.

Try to use as much as possible the following color convention

- black wires for connections to GND (supply, inputs),
- red wires for connections to +5V and logic ‘1’, *but be careful not to use logic ‘1’ as supply voltage*,
- green wires for all the other connections.

You should strip the ends of the wires long enough to have a solid grip in the holes of the protoboard, but not too long to avoid having an unisolated part visible. Before turning on the power for an implemented circuit check if all ICs have their GND, +5V connected correctly. *To remove an IC from the protoboard always use the metallic tip of the logic probe!!! Gently push the tip below the IC and by slowly rocking it pry the IC loose.*

The pin-outs for the ICs used in the experiments are given on the inner covers of this manual.

1.3 The Input-Control/Output-Display Unit (ICOD)

This unit is your interface with the circuits you will design. It will be used in each of the six laboratory sessions so it is recommended you implement it carefully and give some thought on how to arrange it in the *upper left* block of the protoboard which should be used only for this purpose.

The following components are needed to build the ICOD unit:

– quad miniature single-pole single-throw switch : 3
– LED: 9
– resistors, $R = 620\Omega$: 18
– 74LS04, hex inverter: 2

The unit contains nine single-pole single-throw miniature switches for providing logic '0' or logic '1' inputs, and nine LEDs for displaying the logic values of up to nine signals.

Switches. Each of the nine switches should be wired as shown in Figure 3. The pull-up resistor is $R = 620\Omega$. When a switch is open the circuit provides logic '1' to the gate input where it is connected. When the switch is closed it provides logic '0'. You should

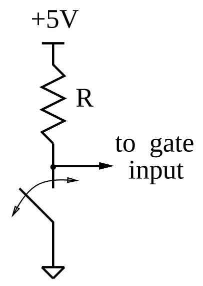

Figure 3: How to connect a single-pole single-throw switch.

arrange the nine switches in a column with all of them having the same polarity. Thus with a glimpse you can read the logic value of your (maximum) 4-bit input signal.

LEDs. A light emitting diode (LED) is turned on whenever the voltage drop across it exceeds a threshold voltage (typically 1.6V). The brightness of the LED depends on the current through it, and values larger than 5mA should yield satisfactory results. The ability of a gate to supply (or sink) limits the current upward. The resistor $R = 620\Omega$ makes sure that the current cannot exceed the maximum value allowed for a TTL gate. The negative polarity of the LED is the pin close to the flattened bevel.

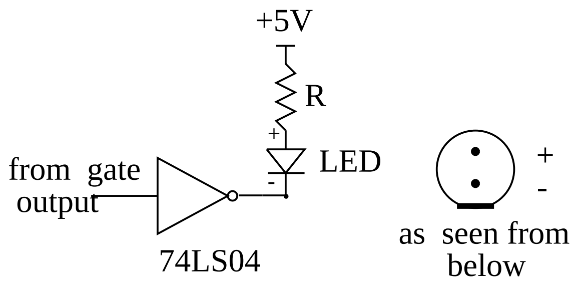

Figure 4: How to connect an LED.

Arrange the nine LEDs in a column to have a (maximum) 9-bit output signal display.

To have the bright LED correspond to logic '1' the displayed signal must be inverted first. Be sure that you understand why. The IC 74LS04 contains six inverters. Mount both of these ICs in the same protoboard block.

To have a reliable ICOD unit which will endure the whole semester, the connections within the unit should have short wires and the leads of the nine resistors cut short enough to lie on the protoboard. Connect long wires to the nine inputs and the nine outputs and label each of them with their name, e.g., IN0 or OUT2.

Once the ICOD unit is implemented, connect the nine input lines to the nine output lines and verify that the LEDs show what is set by the switches. Whenever in an experiment an LED is not in use connect its gate input to the GND. The unused input lines should be connected to *different* empty horizontal bars. *Never leave any wire hanging in the air!!!.*

1.4 A First Experiment

To become familiar with what is in store for the future will implement a simple logic function

$$\mathsf{F(A,B,C) = A{\cdot}B{\cdot}C' + A{\cdot}B + B{\cdot}C}.$$

Without using Boolean algebra we would need three different ICs to implement this circuit.

List of ICs needed ?!

IC Number	IC Name
74LS04	hex inverter (1)
74LS08	quad 2-input AND (1)
74LS32	quad 2-input OR (1)

However, this would be a huge waste of resources since with a little (digital) magic we can use only three ICs that are the same.

The IC which does all

IC Number	IC Name
74LS00	quad 2-input NAND (3)

Draw the NAND-NAND realization of the logic function. This should take 9 NAND gates. To make an INVERTER, wire both inputs of a NAND gate to the same input. Take three 74LS00s and wire them on the protoboard according to your circuit schematic. Later in the course, you will learn how to make this circuit using only 6 NAND gates.

Connect the A, B, C inputs to three switches and the output F to a display unit (LED). Generate all possible inputs and verify that the output obeys the logic function. If it does, you are done. If it doesn't, you got a wonderful opportunity to practice debugging.

1.5 IMPORTANT NOTICE

Starting the next laboratory you must prepare the complete pre-lab *before* coming to lab. Failure to do so will result in a 40% grade penalty for the lab in question.

2 LABORATORY No. 2
Combinational SSI Circuits

In this set of experiments you will design, simulate and build simple circuits containing only small scale ICs. Given a logic function as a minterm list you are asked to find the corresponding minimal sum-of-products expression. You will simulate the circuit by drawing the waveforms and eliminate the static-1 timing hazard present. The circuit with the timing hazard eliminated will be implemented in hardware. A second circuit will be also analyzed starting from its hardware implementation. In the final report you will be asked some additional questions about the two circuits.

The theoretical background required for this laboratory

– Boolean algebra,
– Karnaugh maps,
– minimizing sum-of-products,
– minimizing product-of-sums,
– static timing hazards,

is available from your lecture notes and textbook.

Your grade is computed based on the following weights:

– *Pre-lab report* as defined in Section 2.1.6: 40%.
– *Experimental work* in the laboratory, as defined in Section 2.2: 40%.
– *Final report* as defined in Section 2.3, and submitted at the beginning of the next meeting: 20%.

List of ICs Used in the Experiments

IC Number	IC Name	Qty
74LS00	quad 2-input NAND	1
74LS04	hex inverter	1
74LS10	triple 3-input NAND	1
74LS32	quad 2-input OR	1

2.1 MATERIAL YOU MUST PREPARE AHEAD

You are given the following logic function as a minterm list

$$\mathsf{F}(\mathsf{A},\mathsf{B},\mathsf{C},\mathsf{D}) = \Sigma_{\mathsf{A,B,C,D}}(1,3,5,6,7,14)\ .$$

Table 1: Truth Table for Full Adder

Inputs			Outputs	
A	*B*	*Carryin*	*Sum*	*Carryout*
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

2.1.1 Circuit Minimization

Starting from the minterm list derive the the Karnaugh map of the function. Find the minimal sum-of-products expression for F(A, B, C, D).

2.1.2 Simulation of F(A, B, C, D)

Draw the circuit implementing the obtained expression. Use a NAND-NAND realization for the function (bubble-to-bubble logic).

To simulate the circuit the input should go through all the possible combinations of A, B, C, D from 0000 to 1111 in the natural sequence. The clock cycle is taken to be 100ns. Assume that all the gates have the same 10ns propagation delay for any transition. Draw the following waveforms (in this order): A, B, C, D and F. Specify the time scale!

2.1.3 Timing Hazard

There is a static-1 timing hazard at the 0111 $\rightarrow$ 0110 transition, so you should be careful when analyzing this step. Explain why it happened. Using the Karnaugh map, or the consensus term define the expression of G(A, B, C, D) in which the timing hazard is eliminated.

2.1.4 Simulation of G(A, B, C, D)

Draw the circuit implementing G(A, B, C, D). Since you have used a NAND-NAND realization it will be very easy to transform the previous circuit still using the available resources. Analyze the transition generating timing hazard in the circuit implementing G(A, B, C, D).

2.1.5 Design of a Full-Adder

Table 1 shows the truth table for the full adder, which realizes one bit position of two's-complement arithmetic in every adder used today.

Write the 3-variable Karnaugh maps for the output functions *Sum* and *Carryout* in terms of the inputs *A*, *B*, and *Carryin*. Produce an optimized version of the Boolean function, and implement the circuit in NAND, NOR, and INVERTER gates.

2.1.6 Pre-lab Report

Your pre-lab report should contain the following

– The Karnaugh map of the logic function F(A, B, C, D).
– The minimal sum-of-products expression.
– The logic diagram of the circuit implementing F(A, B, C, D).
– The obtained waveforms.
– Discussion about the timing hazard you analyzed.
– The logic diagram of the circuit implementing G(A, B, C, D).
– The waveforms for the eliminated timing hazard.
– The Karnaugh maps of the full adder.
– The optimized logic diagram of the full adder implementation with NAND, NOR, and INVERTER gates.

You must present the pre-lab report to your TA in order to be allowed to perform the experiments. The pre-lab report will be marked by the TA during the laboratory and must be attached to your final report.

2.2 Experiments

Connect the four input signals A, B, C, D to the four switches of the ICOD unit, and the implemented functions to the first LED (i.e., to the inverter gate controlling it). The remaining three LEDs of the ICOD unit should be connected to the ground.

2.2.1 Experiment 2.1

Implement the *timing hazard free* circuit realizing G(A, B, C, D) you obtained in the pre-lab.

Complete the truth table, and verify that it agrees with the minterm list defining the function.

A	B	C	D	G(A, B, C, D)
0	0	0	0	
0	0	0	1	
0	0	1	0	
0	0	1	1	
0	1	0	0	
0	1	0	1	
0	1	1	0	
0	1	1	1	
1	0	0	0	
1	0	0	1	
1	0	1	0	
1	0	1	1	
1	1	0	0	
1	1	0	1	
1	1	1	0	
1	1	1	1	

Even if your circuit has no errors, for at least one truth table entry trace the intermediate values using the logic probe.

Disconnect the consensus term from the input into the second-level gate. Connect a logic '1' signal instead. (Connect the removed wire of the consensus term to an *unused* horizontal bar.) Check for several entries of the truth table that indeed $\mathsf{G}(\mathsf{A}, \mathsf{B}, \mathsf{C}, \mathsf{D}) \equiv \mathsf{F}(\mathsf{A}, \mathsf{B}, \mathsf{C}, \mathsf{D})$.

Show your TA the circuit and verify some truth table values in his/her presence. After that remove the wires connecting the gates (but leave the IC supply wires) and proceed to the next experiment. You may want to reuse the removed wires.

2.2.2 Experiment 2.2

In this experiment you are given the logic diagram in Figure 5. Using the ICs available for

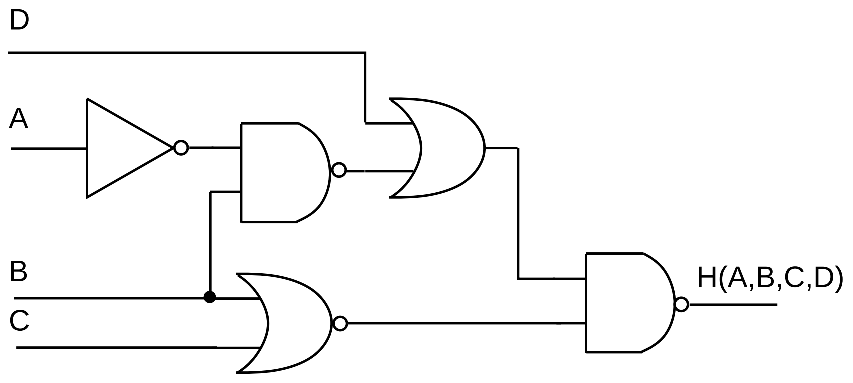

Figure 5: The circuit to be implemented in Experiment 2.2.

this lab implement the circuit in hardware. Note that you have only OR but not NOR gates! Connect the four inputs to the four switches of the ICOD unit and the output to an LED.

Set the variables B and C to logic '0', and complete the truth table of H(A, 0, 0, D) below.

A	D	H(A, 0, 0, D)
0	0	
0	1	
1	0	
1	1	

Show your TA the circuit and the obtained truth table.

2.2.3 Experiment 2.3: Full Adder

Build your full adder and show that it behaves correctly for all input combinations to the TA. That's all for today!

2.3 Final Report

The final report is composed of two parts. Your pre-lab report with all the material requested in Section 2.1.6, as marked by the TA. In addition a short paper should contain the following.

– Experiment 2.1. Obtain the minimal product-of-sums expression for F(A, B, C, D). (*Hint.* Use the Karnaugh map of $F'(A, B, C, D)$ and the DeMorgan theorem.) Do *not* consider the terms eliminating possible timing hazards. Prove using Boolean algebra theorems (and none of the other methods!) that the two expressions represent indeed the same circuit.

– Experiment 2.2. To explain the observed truth table write down the expression of H(A, B, C, D). Using Boolean algebra (and none of the other methods!) obtain the simplest possible equivalent expression. Discuss the result.

– Experiment 2.3. To explain the adder function show how a four-bit adder can be produced by cascading four of your one-bit adders together with *Carryout* signals for bit position n flowing into *Carryin* signals for bit position $n-1$ (assume that the MSB is bit position 0). Discuss the result and its circuit delay.

3 LABORATORY No. 3

Combinational MSI Circuits

In this set of experiments you will design, simulate and build a circuit containing a few medium scale ICs. Your circuit will take a 3-bit binary code, transform it into Gray code and decode it back to binary code.

The Gray code has the characteristics that only one bit position changes between two consecutive decimal numbers represented by the code. The 3-bit Gray code is defined as

Decimal	Binary	Gray
	B2B1B0	G2G1G0
0	000	000
1	001	001
2	010	011
3	011	010
4	100	110
5	101	111
6	110	101
7	111	100

The advantage of Gray code is that it reduces the probability of false outputs caused by multiple bit changes. It is widely used in optical sensors translating mechanical processes into electrical signals.

You will design your binary/Gray encoder/decoder first and then implement it in hardware. In the final report you will have to provide another design for the decoder stage.

The theoretical background required for this laboratory
- multiplexers,
- exclusive ORs,
- decoders,

is available from your lecture notes and textbook.

Your grade is computed based on the following weights:
- *Pre-lab report* as defined in Section 3.1.3: 40%.
- *Experimental work* in the laboratory, as defined in Section 3.2: 40%.
- *Final report* as defined in Section 3.3, and submitted at the beginning of the next meeting: 20%.

List of ICs Used in the Experiments

IC Number	IC Name	Qty
74LS86A	quad 2-input XOR	1
74LS151A	8-input, 1-bit (8-to-1) MUX	3

3.1 MATERIAL YOU MUST PREPARE AHEAD

3.1.1 Circuit Design

The encoder part of your circuit will be realized using an 8-to-1 multiplexer for each of the three Gray code bits. The eight input lines of a MUX have to be connected to logic '0' or '1' based on the code conversion table above. The three input binary bits, BIN0, BIN1, BIN2 control the selection process.

To design the decoder, you must first find the expression for the binary code bits BOUT0, BOUT1, BOUT2 as function of the three Gray code bits G0, G1, G2. Use the Karnaugh maps derived from the code conversion table. Be sure that you take the Gray code bits as variables and the binary code bits as the values of the logic function. That is, your truth table is from right to left now!

The expressions become more complicated as you move from BOUT2 to BOUT0. Prove that BOUT1 is an XOR function between G1 and G2

$$\mathsf{BOUT1} = \mathsf{G1} \oplus \mathsf{G2} \ ,$$

and that BOUT0 is a double XOR function between all three Gray code bits

$$\mathsf{BOUT0} = \mathsf{G0} \oplus \mathsf{G1} \oplus \mathsf{G2} \ .$$

3.1.2 Simulation

Draw the encoder/decoder circuit using the available resources. Arrange your MUXs at the left of the schematics and your XOR gates at the right.

Simulate the circuit. The binary input should go through all the possible combinations of BIN2, BIN1, BIN0 from 000 to 111 in the natural sequence. The period of BIN0 should be 100ns. Assume a 20ns propagation delay for any transition in the MUX or XOR. Represent the following waveforms (in this order): BIN2, BIN1, BIN0, G2, G1, G0 and BOUT2, BOUT1, BOUT0. Specify the time scale!

3.1.3 Pre-lab Report

Your pre-lab report should contain the following

– The three Karnaugh maps of the decoder.

- Discussion of why you can substitute the obtained minimal sum-of-products expressions with the XOR realizations.
- The logic diagram of the circuit implementing the binary/Gray encoder/decoder.
- The obtained waveforms.

You must present the pre-lab report to your TA in order to be allowed to perform the experiments. The pre-lab report will be marked by the TA during the laboratory and must be attached to your final report.

3.2 Experiments

The three input signals BIN0, BIN1, BIN2 should be connected to the first three switches of the ICOD unit. The first three LEDs (i.e., the inverter gates controlling them) will be switched between G0, G1, G2 and BOUT0, BOUT1, BOUT2 as you go along with the implementation.

3.2.1 Experiment 3.1

Implement the encoder part of your circuit using the specified 74LS circuits. Verify that your output follows the code conversion table. Show your TA the circuit and verify some truth table values in his/her presence.

3.2.2 Experiment 3.2

Add the decoder part to your circuit. Move the LEDs to the binary outputs. Verify that the LEDs reproduce the binary words set by the input switches. Show your TA the circuit and verify some truth table values in his/her presence. That's all for today, but do not forget the final report.

3.3 Final Report

The final report is composed of two parts. Your pre-lab report with all the material requested in Section 3.1.3, as marked by the TA. In addition a short paper should contain the following.

- The MSI decoder circuit 74138 can be used to take the Gray code bits and generate eight active low signals corresponding to the eight possible binary outputs. Draw the connections of a 74138 realizing this function. Connect all the control inputs to logic '0' or '1' as necessary. Label the signal inputs/outputs showing clearly how the eight output lines, from BOUTL0_L to BOUTL7_L are allocated.
- Assume now that you are given these eight output lines and have to generate as in Experiment 3.2.2 the three active high bits BOUT0, BOUT1, BOUT2. Explain how this can be achieved by using three 4-input NAND gates and give the logical expressions realized.

4 LABORATORY No. 4

Four-Bit Arithmetic Unit

In this set of experiments you will design, simulate and build a circuit performing addition/subtraction/increment/decrement of 4-bit words in two's-complement representation. The numbers you can handle thus have the representation

Decimal	Two's-complement
-8	1000
-7	1001
-6	1010
-5	1011
-4	1100
-3	1101
-2	1110
-1	1111
+0	0000
+1	0001
+2	0010
+3	0011
+4	0100
+5	0101
+6	0110
+7	0111

Make sure that you are familiar with this representation.

Your unit will be built around an MSI 4-bit binary adder. To perform other operations beside addition you will have to use a multiplexer for each bit as well as a few additional gates. The nature of the operation is controlled by the inputs S0 and S1.

Code S1S0	Operation
00	addition
01	subtraction
10	increment
11	decrement

After you have designed the complete unit, you will predict the result of several operations and then implement it in hardware. In the final report you will be asked to explain the results of the computations.

The theoretical background required for this laboratory
- two's-complement representation,
- arithmetic operations with two's-complement representation,
- 4-bit binary MSI adder circuit, 74LS283,

is available from your lecture notes and textbook.

Your grade is computed based on the following weights:
- *Pre-lab report* as defined in Section 4.1.3: 40%.
- *Experimental work* in the laboratory, as defined in Section 4.2: 40%.
- *Final report* as defined in Section 4.3, and submitted at the beginning of the next meeting: 20%.

List of ICs Used in the Experiments

IC Number	IC Name	Qty
74LS04	hex inverter	1
74LS86A	quad 2-input XOR	1
74LS153	4-input, 2-bit (dual 4-to-1) MUX	2
74LS283	4-bit binary adder	1

4.1 MATERIAL YOU MUST PREPARE AHEAD

4.1.1 Circuit Design

Since all four bits are processed the same way you have to design only the handling of one bit. Let the two binary words used in addition or subtraction be X3X2X1X0 (X[3:0]) and Y3Y2Y1Y0 (Y[3:0]). The structure of your circuit (for one bit) should be like in Figure 6.

The first addend X is applied directly to the adder. To perform all four desired operations with the adder, the second addend must be chosen with the help of the two control variables S1S0. However, to know how to wire the four inputs of the 4-to-1 multiplexer to either of the signals: a bit of Y, the complement of the same bit, logic'0', logic '1', you must decide first about the carry-in generation circuit. *The order of MUX inputs in Figure 6 is not correct!*

Recall that in the two's-complement subtraction you need to have the CIN signal asserted to logic '1'. You can implement the increment operation as setting Y[3:0] to 0000 but providing an asserted CIN signal. Similarly, the decrement operation can be implemented as an addition with -1.

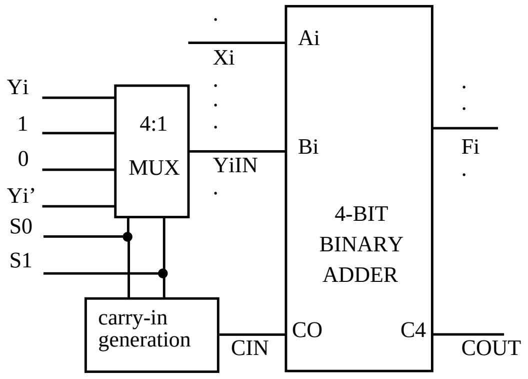

Figure 6: The structure of the arithmetic unit.

Why should the increment/decrement be done this way? Examine the control codes for subtraction and increment, the two operations when CIN is asserted to one. You should be able to see how simple the carry-in generation circuit is. The multiplexer inputs should be easy to derive now from the way the operations were defined.

4.1.2 Simulation

Draw the 4-bit arithmetic unit. Arrange your MUXs and additional gates on the left of the schematics and your adder on the right.

To simulate the circuit assume that the word X[3:0] is set to represent +2, and the word Y[3:0] to represent +5. For all the possible combinations of S1S0 in the natural sequence from 00 to 11 provide the table of the output F3F2F1F0.

4.1.3 Pre-lab Report

Your pre-lab report should contain the following

– Discussion of how you arrived to your design.
– The logic diagram of the circuit implementing the 4-bit arithmetic unit.
– The truth table of the investigated case.

You must present the pre-lab report to your TA in order to be allowed to perform the experiments. The pre-lab report will be marked by the TA during the laboratory and must be attached to your final report.

4.2 Experiments

There are ten input signals. Use two switches of the ICOD unit to have access to the control signals S0 and S1. The other two switches should be connected to X3 and Y3, i.e. the sign bits of the two operands. Connect the four LEDs to the output bits of the adder, F0 to F3. The COUT signal should be checked with the logic probe.

4.2.1 Experiment 4.1

Using the switches set both sign bits to logic '0'. Connect with wires the bits X2, X1 and X0 to logic '0' or logic '1' so that X[3:0] represents +2. Similarly, connect Y2, Y1 and Y0 so that Y[3:0] represents +5.

Verify that you obtain the same (correct!) results as in your truth table. Show your TA the circuit and verify some simulation values in his/her presence.

4.2.2 Experiment 4.2

Change Y3 to logic '1' with the switch. What is the number represented now by Y[3:0]? Fill in the following table

Decimal X Y	Operation	Result	
		Binary	Decimal
+2	Addition		
+2	Subtraction		

4.2.3 Experiment 4.3

Change *also* X3 to logic '1' with the switch. What is the number represented now by X[3:0]? Fill in the following table

Decimal X Y	Operation	Result		
		Binary	Decimal	COUT
	Addition			
	Subtraction			
	Increment			
	Decrement			

4.2.4 Experiment 4.4

Set the sign bit X3 to logic '0' and let the sign bit Y3 to be logic '1'. Change X[3:0] from +2 to +3 by moving the X0 wire from logic '0' to logic '1'. Execute addition and record the result as well as the value of COUT. That's all for today!

4.3 Final Report

The final report is composed of two parts. Your pre-lab report containing all the material requested in Section 4.1.3, as was marked by the TA. In addition a short paper should contain the following.

- The table from Experiment 4.2.2, with discussion of the results.
- The table from Experiment 4.2.3, with discussion of the results. Do not forget COUT. In which case does an overflow appear?
- Discussion of the result obtained in Experiment 4.2.4. Explain the value of COUT.

5 LABORATORY No. 5

Sequential Circuits. State Machine Analysis

In this set of experiments you will test several flip-flops and will convince yourself why switch debouncing is needed. A simple state machine, a 3-bit linear feedback shift register counter will be also analyzed and simulated.

An n-bit linear feedback shift register (LFSR) counter is a circuit which goes through its 2^n states (corresponding to numbers from 0 to $2^n - 1$) in non-binary order. The LFSR counters are used to generate pseudo-random sequences and are a central component of any communication device which has error detection/correction or encryption.

The theoretical background required for this laboratory

– flip-flops,
– switch debouncing,
– state machine analysis,

is available from your lecture notes and textbook.

Your grade is computed based on the following weights:

– *Pre-lab report* as defined in Section 5.1.5: 40%.
– *Experimental work* in the laboratory, as defined in Section 5.2: 40%.
– *Final report* as defined in Section 5.3, and submitted at the beginning of the next meeting: 20%.

List of ICs Used in the Experiments

IC Number	IC Name	Qty
74LS08	quad 2-input AND	1
74LS74A	dual positive edge triggered D Flip-flop	3
74LS86A	quad 2-input XOR	1
74LS112A	dual negative edge triggered JK Flip-flop	1

one miniature single-pole double-throw switch, two resistors $R = 620\Omega$.

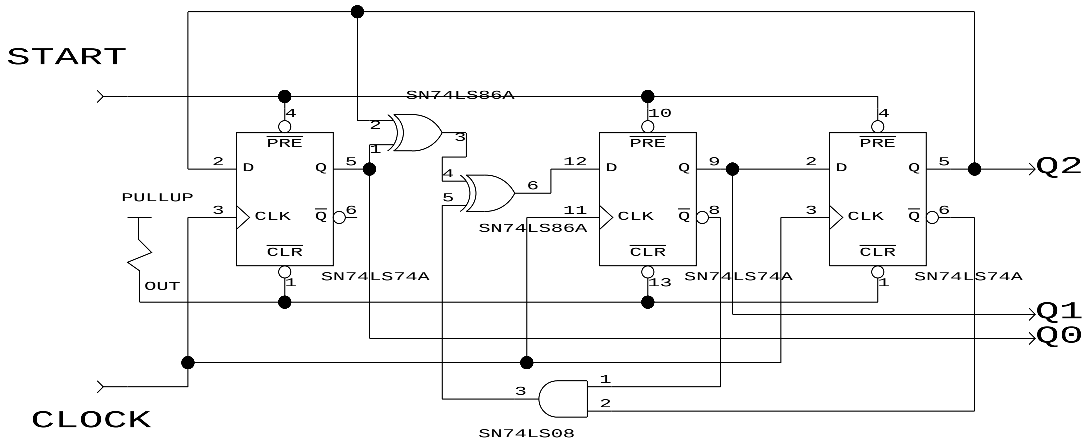

Figure 7: The linear feedback shift register circuit.

5.1 MATERIAL YOU MUST PREPARE AHEAD

The LFSR counter to be analyzed and implemented is shown in Figure 7. The LFSR will be considered as a state machine. Let the 8 states be

State	Q2Q1Q0
A	000
B	001
C	010
D	011
E	100
F	101
G	110
H	111

5.1.1 Transition equations

Write down the excitation equations of the three D flip-flops carrying the state variables Q0, Q1 and Q2. Derive the transition equations of the state machine.

5.1.2 State transition table

Using the transition equations and the state names, derive the state transition table.
The transition equation of D1 looks somewhat scary. Prove, using Boolean algebra that

$$\mathsf{D1} = \mathsf{Q0} \oplus [\mathsf{Q1}' + \mathsf{Q2}]$$

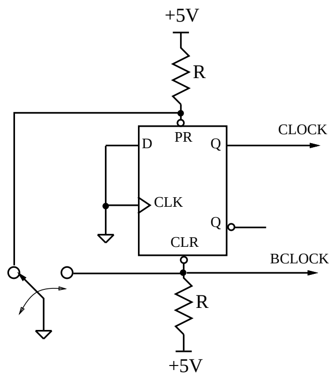

Figure 8: The circuit generating the CLOCK signal.

5.1.3 State diagram

Using the state transition table draw the state diagram of the LFSR counter. Assume that the initial state is H, i.e., the digit 7. Why should we do that? Prove that the generated pseudo-random sequence is:

$$\cdots\ 7\ 5\ 1\ 0\ 2\ 4\ 3\ 6\ 7\ \cdots$$

5.1.4 Simulation

Draw the waveforms of the state variables Q2, Q1 and Q0 starting from the initial state H over nine clock cycles.

5.1.5 Pre-lab Report

Your pre-lab report should contain the following

- The state machine analysis as defined in Sections 5.1.1 to 5.1.3.
- The obtained waveforms.

You must present the pre-lab report to your TA in order to be allowed to perform the experiments. The pre-lab report will be marked by the TA during the laboratory and must be attached to your final report.

5.2 Experiments

Before implementing the LFSR counter you will need to build a circuit generating the CLOCK signal. This circuit will be used again in Laboratory No. 6 so place it in the block of the ICOD unit if you have room left. Otherwise use the top part of another block on the protoboard.

At the end of this laboratory do *not* remove the clock generation circuit from the protoboard.

5.2.1 Experiment 5.1

Mechanical switches are noisy, i.e., a single switching actually contains several on/off transitions due to the elasticity of the contacts. To be able to use a mechanical switch as a clock signal for flip-flops, the switch must be debounced first. The circuit in Figure 8 achieves this with the help of the asynchronous inputs of a D flip-flop. Be sure that you understand how it functions.

Implement the debouncing circuit. Use the single-pole double-throw switch provided. The resistors are $R = 620\Omega$. Attach a long wire to the CLOCK output. For the moment connect the wire to an LED of the ICOD unit.

Generate several clock cycles. How many switchings are needed for one cycle? Draw the waveform of CLOCK for two cycles and indicate the position of the switch for each signal level.

5.2.2 Experiment 5.2

Mount a second D flip-flop IC (do not use the remaining D flip-flop from the clock generation circuit!) and the J-K flip-flop. Connect all their asynchronous inputs to logic '1'. Connect one switch of the ICOD unit to the D input of the D flip-flop and other two switches to the J and K inputs of the J-K flip-flop. Connect the active high outputs of the flip-flops to two LEDs of the ICOD unit.

Remove the CLOCK signal from the LED (Section 5.2.1) and connect it to the CLK inputs of the two flip-flops. Note that the D flip-flop is positive edge triggered while the J-K flip-flop is negative edge triggered.

For each of the two flip-flops apply all input combinations and draw the waveforms for the output as the clock goes through *two* cycles. Check that changes in the input have no influence on the output when the CLOCK signal is constant.

5.2.3 Experiment 5.3

To experience the bouncing of the switch you should use in this experiment the signal BCLOCK as clock. See Figure 8. Why has this signal the same polarity as CLOCK? Disconnect CLOCK from the J-K flip-flop and connect BCLOCK instead.

Reset the J-K flip-flop to have the output at logic '0'. (No need if it is already in that state.) Otherwise apply the necessary input and then trigger it with the clock signal. If the change from 1 to 0 does not happen when you know it should, trigger it again till you get the desired output.

Set now the inputs of the J-K flip-flop as to emulate a T flip-flop. Apply *ten* clock cycles. Count carefully the number of switchings you do. After each cycle predict what the output should be in the theory. Compare it with the one shown by the LED. Record how many times the result was not the same. Your prediction should use the *previous theoretical* value not the observed one! Comment on the results.

5.2.4 Experiment 5.4

Remove the J-K flip-flop and its wires. Mount the third D flip-flop and implement the LFSR counter you analyzed in the pre-lab. Connect the START signal to a switch of the ICOD unit, the CLOCK to your *debounced* clock signal and the three state variables Q0, Q1 and Q2 to the LEDs.

Reset the state machine to state H and check that it produces the pseudo-random sequence from Section 5.1.3. If your circuit behaves differently you must debug it. Use the logic probe and check if the inputs into the flip-flops follow the values from the excitation equations. When the error(s) were found and corrected show your TA the circuit and verify some simulation values in his/her presence. That's all for today!

5.3 Final Report

The final report is composed of two parts. Your pre-lab report containing all the material requested in Section 5.1.5, as marked by the TA. In addition a short paper should contain the following.

– The waveform from Experiment 5.2.1, with discussion.
– The waveforms from Experiment 5.2.2, with discussion.
– The result from Experiment 5.2.3, with discussion.
– Discussion on Experiment 5.2.4. Did you need to debug? How did you find the error?

6 LABORATORY No. 6

State Machine Synthesis

In this set of experiments you will design, simulate and implement a simple controller of an elevator in a three story building. Well, to keep the project simple we cut some corners.

The theoretical background required for this laboratory

– state machine synthesis with D flip-flops,

is available from your lecture notes and textbook.

Your grade is computed based on the following weights:

– *Pre-lab report* as defined in Section 6.1.5: 60%. *Note the change of weight!*
– *Experimental work* in the laboratory, as defined in Section 6.2: 40%.
– There is no final report.

List of ICs Used in the Experiments

IC Number	IC Name	Qty
74LS04	hex inverter	1
74LS08	quad 2-input AND	1
74LS10	tri 3-input NAND	1
74LS32	quad 2-input OR	1
74LS74A	dual positive edge triggered D flip-flop	1

6.1 MATERIAL YOU MUST PREPARE AHEAD

The elevator controller will use only four states defined by the state variables Q0 and Q1.

Symbol	Q1Q0	Meaning
FLR1	00	Elevator stopped at the first floor
FLR2	01	Elevator stopped at the second floor
MOV2	10	Elevator moving *through* the second floor
FLR3	11	Elevator stopped at the third floor

The input to the controller is the request for moving to one of the floors, carried by the bits R0 and R1.

R1R0	Meaning
00	No request
01	Request from (move to) first floor
10	Request from (move to) second floor
11	Request from (move to) third floor

The following state transition table is assumed

Current	R1R0			
State	00	01	11	10
FLR1	FLR1	FLR1	MOV2	FLR2
FLR2	FLR2	FLR1	FLR3	FLR2
FLR3	FLR3	MOV2	FLR3	FLR2
MOV2	FLR1	FLR1	FLR3	FLR2

Note that the next state after MOV2 was chosen more to keep the excitation logic the simplest possible than to follow reality. (We are sure you are appreciating this.)

6.1.1 State Diagram

Draw the state diagram of the elevator controller. Mark each arc in the diagram with its transition expression. Verify that the diagram is unambiguous, i.e, for any input combination only one arc leaving a state is validated, while the logical sum of the transition expressions of all arcs leaving a state is always 1.

6.1.2 Excitation Table

The controller will be implemented with D flip-flops. Using the state transition table from above for D flip-flops derive the excitation table of the state machine...

6.1.3 Excitation Maps

...and the excitation map.

6.1.4 Simulation

Draw the the elevator controller. You must use the ICs listed for this laboratory, i.e., you will need bubble-to-bubble logic to implement the OR functions. Connect the CLK inputs of the flip-flops to a CLOCK signal, and the asynchronous CLR inputs to an active low RESET signal.

The sequence of commands to be applied for simulating the elevator controller is (starting from the left):

R1R0 00 01 10 01 11 11 01 01 10 00 10 11 10 11 00 11

Describe in words the actions simulated as they are executed. For example, move from first to the third floor, move from the third to the second floor, etc. Give the table with the corresponding values of the state variables Q0 Q1.

6.1.5 Pre-lab Report

Your pre-lab report should contain the following

- The state machine synthesis following Sections 6.1.1 to 6.1.3.
- The logic diagram of the circuit implementing the elevator controller.
- The verbal description of the simulation and the obtained waveforms.

You must present the pre-lab report to your TA in order to be allowed to perform the experiments. The pre-lab report will be marked by the TA during the laboratory and together with his/her assessment of the experimental part constitutes your grade for this laboratory.

6.2 Experiments

Implement the elevator controller in hardware. Connect the output of the debounced clock (built in Laboratory No. 5) to the CLOCK signal. Connect a switch of the ICOD unit to the RESET signal. Note that this signal is active low. Connect two LEDs of the ICOD unit to the state variables Q0 and Q1.

6.2.1 Experiment 6.1

Repeat the operations simulated (described in Section 6.1.4) and check that your machine behaves the same way. If it does not, use the logic probe to debug it. Show your TA the circuit and verify the elevator controller in his/her presence.

That's all!!!

The
Digital Logic Design Laboratory
was developed in 1997 and 1999
by
Peter Meer and revised in 2012 by Mike Bushnell
Narasimhan Narayanan
Heather Durko
Phillip Stanley-Marbell.