EE120A Logic Design
Department of Electrical Engineering
University of California – Riverside

Laboratory #2
EE 120 A
Winter 2009

Instructor: Roman Chomko

LABORATORY # 2
LAB MANUAL

Decoders and Muxes

PART 1[*] (decoder) **Design of Sprinkler Valve Controller**

PART 2 (multiplexer) **Design of Computer Data Bus**

[*] Design Example guided through by TA

Objectives

Lab 2 contains 2 parts: **Part 1** – guided design and **Part 2** – individual design. Its purposes are to get familiar with:

1. Xilinx ISE Design software system usage;
2. Simulation and Design of controller systems based on combinational logic;
3. Generation of testbenches for logic design testing and verification;
4. Generation of waveform

Equipment

- PC or compatible
- Xilinx ISE Design Software Suite 10.1
- ModelSim XE III modeling software

Parts

- N/A

PART 1. Design of a Sprinkler Valve Controller[†]

In this guided software experiment, we will design and test a 3 x 8 decoder (with "enable" switch) for a sprinkler valve controller system:

Specification

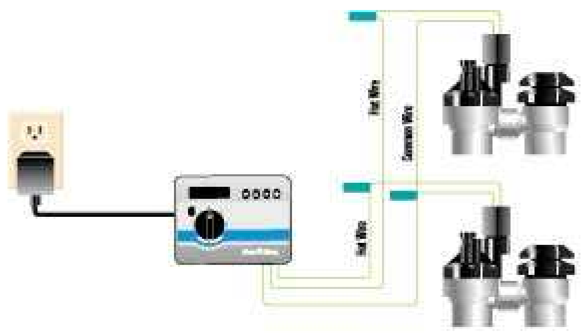


Figure 1. Sprinkler Digital Controller System

Automatic lawn sprinkler systems control the opening and closing of water valves. A sprinkler system must support several different zones, such as the backyard, left side yard, right side yard, etc. Only one zone's valve can be opened at a time to maintain enough pressure in the sprinklers in that zone. In this design assignment a sprinkler system must support up to 8 zones. Note that typically a sprinkler system is controlled by a small microcontroller unit (MCU) which executes a program that opens each valve only at specific times of the day and for specific durations. However, we will limit ourselves to a sub-project that is dealing only with opening and closing of the valves. The system must also provide a facility to disable the opening of any valve.

Analysis and Design

Assuming a microcontroller has only four output pins a system based on a 3 x 8 decoder (with "enable" switch) will do the job.

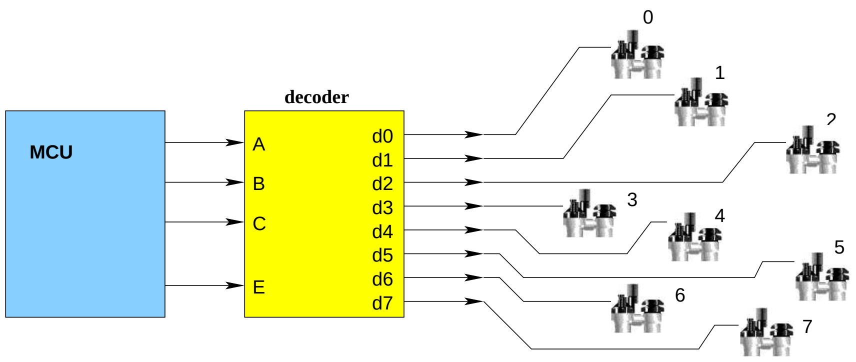


Figure 2

[†] Both parts of the lab are based on examples from Frank Vahid's "Digital Design"

MCU has one pin to indicate whether the system is active (enabled) and the other three pins indicate the binary number of a valve to be opened. The system is a combinational logic circuit that has 4 inputs: E (enabler) and A, B, C (the binary value of the active zone), and 8 outputs d7, …, d0 (the valve controls). The truth table of the system is shown below.

E	A	B	C	d0	d1	d2	d3	d4	d5	d6	d7
0	x	x	x	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	0	0	0	0
1	0	0	1	0	1	0	0	0	0	0	0
1	0	1	0	0	0	1	0	0	0	0	0
1	0	1	1	0	0	0	1	0	0	0	0
1	1	0	0	0	0	0	0	1	0	0	0
1	1	0	1	0	0	0	0	0	1	0	0
1	1	1	0	0	0	0	0	0	0	1	0
1	1	1	1	0	0	0	0	0	0	0	1

Table 1. Truth Table of the Sprinkler System ('x' stands for "don't care")

Following the procedure outlined in Lecture 4, Slide 13 constructing sum-of-products (SOP) minterm based logic expression for each of the data outputs d.

Step 1 – Capture the function

d0 = E . A' . B' . C'
d1 = E . A' . B' . C
d2 = E . A' . B . C'
d3 = E . A' . B . C
d4 = E . A . B' . C'
d5 = E . A . B' . C
d6 = E . A . B . C'
d7 = E . A . B . C

Step 2 – Convert to equations and/or minimize. Nothing to do here.

Step 3 – Implement as a gate based logic circuit

It is shown below using a non-standard[‡] 4-input AND gates and inverters.

[‡] Usage of non-standard gates largely clarifies a circuit schematic but in many cases requires a creation of a gate library manually within Logic Design Software Environments such as Xilinx ISE Design Suite

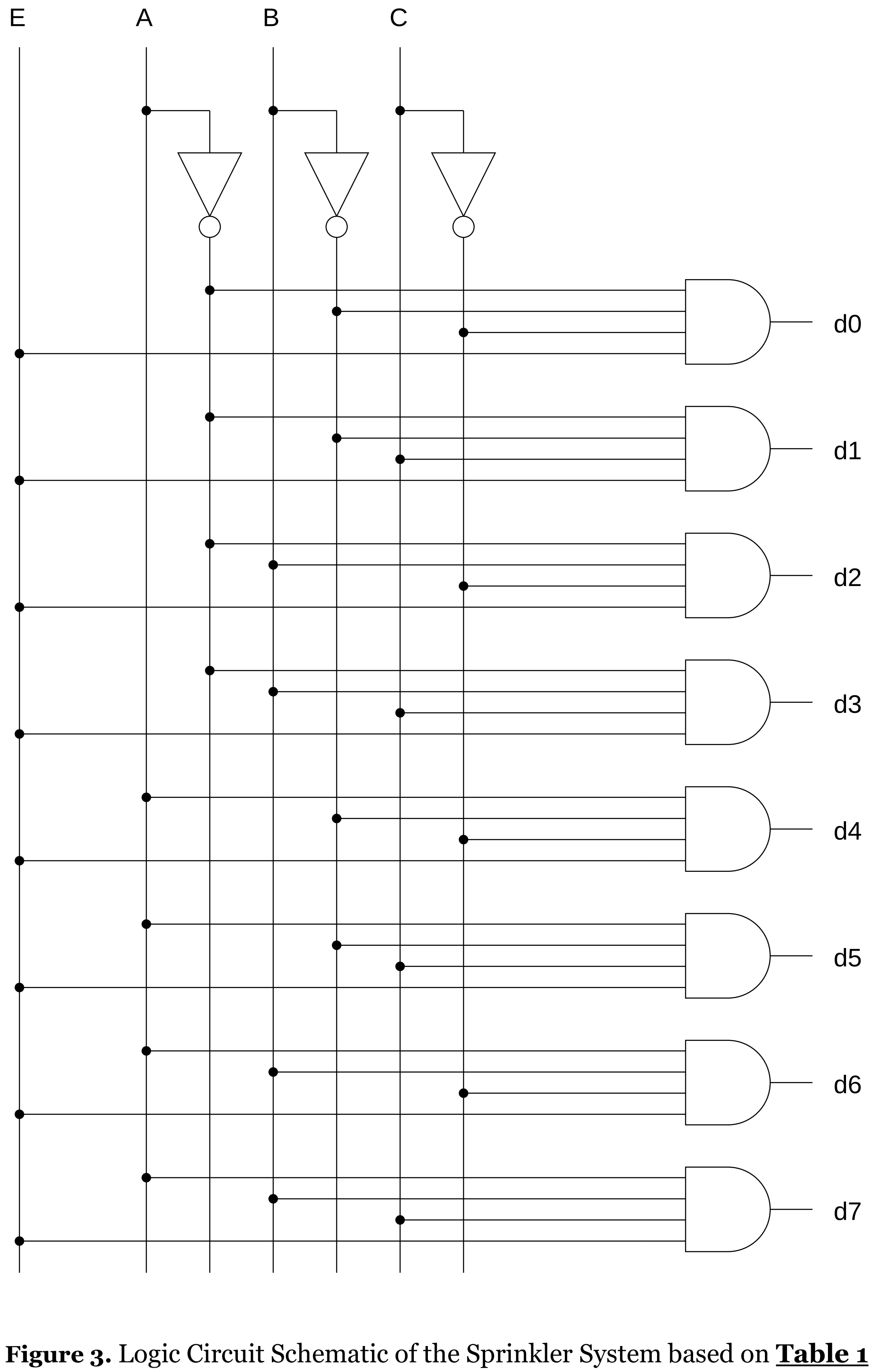


Figure 3. Logic Circuit Schematic of the Sprinkler System based on **Table 1**

Circuit Schematic Capture

Using Xilinx ISE WebPack 10.1.

`File -> New Project -> “ee120a_L2P1_sprinkler_valve_controller” ->` (accept defaults[§] in the following pop-up hardware target wizard by pressing `Next) -> New Source (schematic, “sprinkler_circuit”) -> Next -> Finish`

In the middle panel window one can see a design summary tab. Switch to the “sprinkler_circuit.sch” tab. This is our schematic window.

In the “Sources” panel choose logic symbols for the 4-input AND gate “and4” and the invertor “inv”. Placing them according to the schematic[**] in Figure 3 and wire them properly with “Add Wire” .

Add Inputs/Outputs with “Add I/O Marker”.

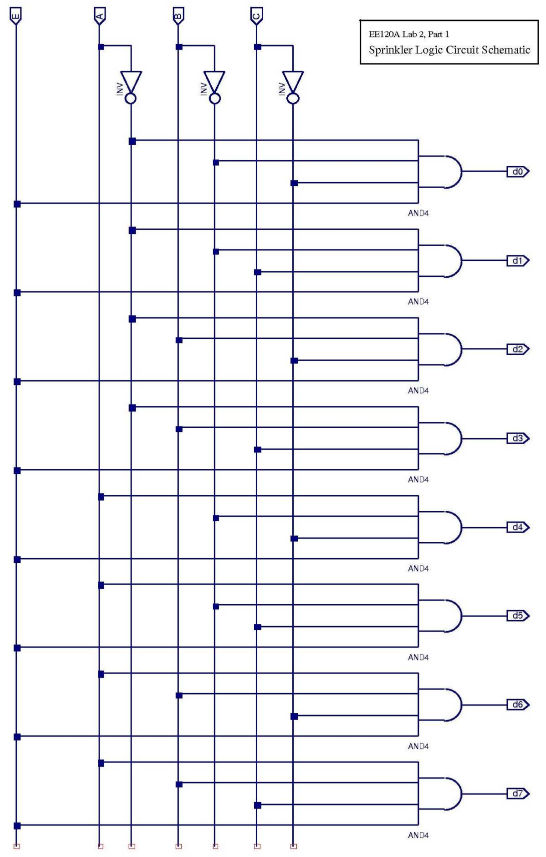


Figure 4. Sprinkler Controller logic circuit schematic created within ISE WebPack

[§] We accepted defaults here since in this project we just simulate a circuit. If we were to actually synthesize the circuit and implement it in a bit-code to be downloaded to a FPGA board we would have to choose an appropriate hardware platform.

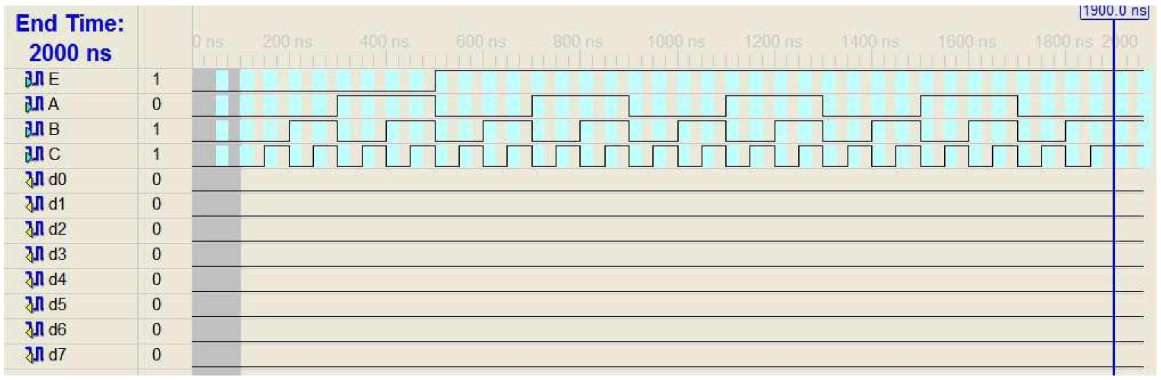
[**] In general there is no need to create two separate logic circuits with different software. One will suffice.

Circuit Logic Behavioral Simulation and Verification

Next we will perform "exaustive" circuit testing as outlined in Lecture 4, Slide 16.

In "Sources" panel go to the "Sources" tab and select `sprinkler_circuit.sch` Now go to the "Processes" panel -> Create New Source -> Test Bench Waveform -> `sprinkler_circuit_tb` (Xilinx doesn't like this way anymore, ... reasonable).

In "Clocking Wizard" select "Combinatorial (or internal clock)" and Initial Length of Test Bench of 2000 ns, 50 ns period (25 ns ON, 25 ns OFF) -> Finish. Observe that this operation will produce a testbench for 20 cycles which is sufficient to simulate 16 rows of our truth table including "don't care"'s needed for a complete verification.



Simulation is done using ModelSim (if installed) by double clicking `ModelSim Simulator -> Simulate Behavioral Model` in the tb processes panel window.

Zoom out the timing diagram window within ModelSim to see the whole waveform. Click anywhere in the waveform window so a yellow sliding time-line appears. Click at any clock time of interest to see the timing diagram for each of In/Out signals to do a visual verification.

The sequence of input and output signals can be re-arranged by pressing and holding the signal line and moving it up or down.

Note that we need ModelSim for two purposes:

1. It provides more flexibility in waveform analysis than Xilinx ISE
2. VHDL coding we will do within ModelSim since it has a better thought out VHDL programming environment. Xilinx ISE's primary goal is synthesizing code for programming FPGA devices, not the whole VHDL language set.

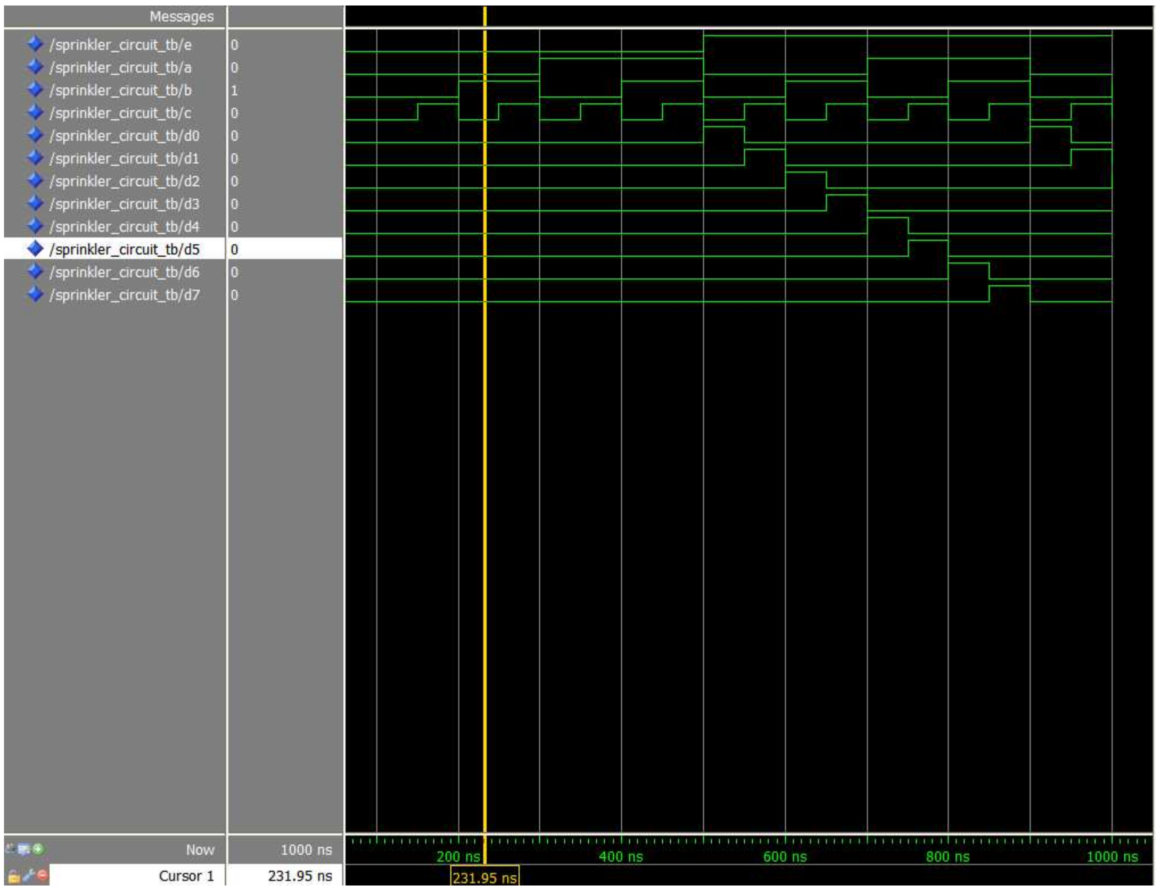

Messages
/sprinkler_circuit_tb/e 0
/sprinkler_circuit_tb/a 0
/sprinkler_circuit_tb/b 1
/sprinkler_circuit_tb/c 0
/sprinkler_circuit_tb/d0 0
/sprinkler_circuit_tb/d1 0
/sprinkler_circuit_tb/d2 0
/sprinkler_circuit_tb/d3 0
/sprinkler_circuit_tb/d4 0
/sprinkler_circuit_tb/d5 0
/sprinkler_circuit_tb/d6 0
/sprinkler_circuit_tb/d7 0
Now 1000 ns
Cursor 1 231.95 ns
200 ns
400 ns
600 ns
800 ns
1000 ns
231.95 ns

Figure 6. Behavioral Simulation of Sprinkler Controller Circuit

Demonstration

See Figure 6. Observe, when E=0 all outputs are 0's no matter what is in A,B,C inputs. Other rows from the truth table a correct.

Questions

1. What is a waveform?
2. What is a testbench?
3. Can we replace the 4-input AND gates in the circuit with the 2-input AND gates? If yes, how?

Conclusion for Part 1

We have gone through the whole cycle of system design, analysis and circuit logic behavioral verification.

PART 2. Design of Computer Data Bus[††]

In this assignment, we will design a 4 x 1 multiplexer that will control the flow of data in a single wire data bus and study its basic properties. This technology allows to accomplish, for example, partial serial communication with multiple peripheral devices using just one output pin of a microcontroller.

Specification

Design a 1-wire data bus controller that performs the following function:

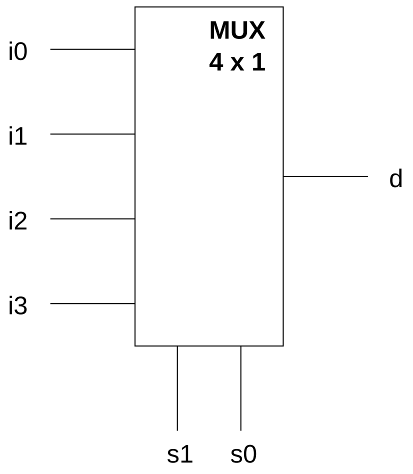


Figure 7. Single-wire data bus multiplexer

Inputs s0,s1 control which of the input data i0, i1, i2, i3 is present in the output d where the binary s1,s0 represents the input pin number.

Example: s1,s0 = 1,0 indicates that data in the input line i2 appear in the output d.

Design a logic circuit that will perform this function and verify the logic functionality using ISE/ModelSim sofware environment.

Demonstration

Demo the waveform obtained during the behavioral simulation and explain why it is correct. Provide also the truth table, algebraic expression of the logic function, and logic circuit schematic.

[††] This part must be completed individually, **NOT in groups**

Procedures

1. Xilinx ISE Design environment;
2. Logic circuit schematic capture;
3. Manipulation of input signal timings in ISE testbenches;
4. Behavioral Simulation and Waveform analysis of logic circuits.

Presentation and Report

Must be presented according to the general EE120A lab guidelines posted in iLearn.

Prelab

1. Familiarize yourself with ISE and ModelSim tutorials posted in iLearn;
2. Review Lectures 1-5;
3. Try to answer all the questions, do all necessary computations