UNIVERSITY OF CALIFORNIA, DAVIS
Department of Electrical and Computer Engineering

EEC180A **DIGITAL SYSTEMS I**

LAB 5: COMBINATIONAL NETWORK DESIGN USING MSI AND PAL DEVICES

The purpose of this lab is to learn how to design and implement a combinational logic network using MSI chips and to introduce Programmable Logic Devices.

Hardware Required:

New parts:

1 pc.	74LS83	4-Bit Full Adder
2 pcs.	74LS153	Dual 4-to-1 Multiplexers
1 pc.	PALCE22V10	Programmable Logic Device

Parts from previous labs:

2 pc.	74LS00	Quad 2-input NAND gate
1 pc.	74LS04	Hex Inverter
1 pc.	74LS10	Triple 3-input NAND gate
1 pc.	7730	7-Segment Display

Preparation (Pre-lab)

Do the complete paper design for Part I. Show the K-maps for the output signals and the hardware implementation using muxes and gates. For each of the four functions to be implemented using a multiplexer, you should show two different designs as specified below.

Part I: Using 4-Bit Adder, Muxes and Gates

a) For this part, the 7483 4-bit adder will be used to add two 2-bit numbers and a 1-bit number to produce a 3-bit output as shown in Figure 1. (You will need to use the carry-in bit, C0.) Design a combinational logic network which accepts the 3-bit output from the adder and drives a 7-segment display such that the result is displayed in decimal (0-7).

You will need to design logic functions to drive each of the 7-segment display drivers, a-g. Design and implement four of these functions by using four 4-to-1 multiplexers and the other three by using logic gates.

For each function implemented using a multiplexer, show at least ***two*** designs for the function with ***different*** sets of variables used on the selection inputs! Your TA will decide which design you will implement in the lab. Be sure to have your TA verify your working circuit.

Also remember to connect a 330 to 500 ohms of resistor to ***each*** segment input of the 7-Segment Display.

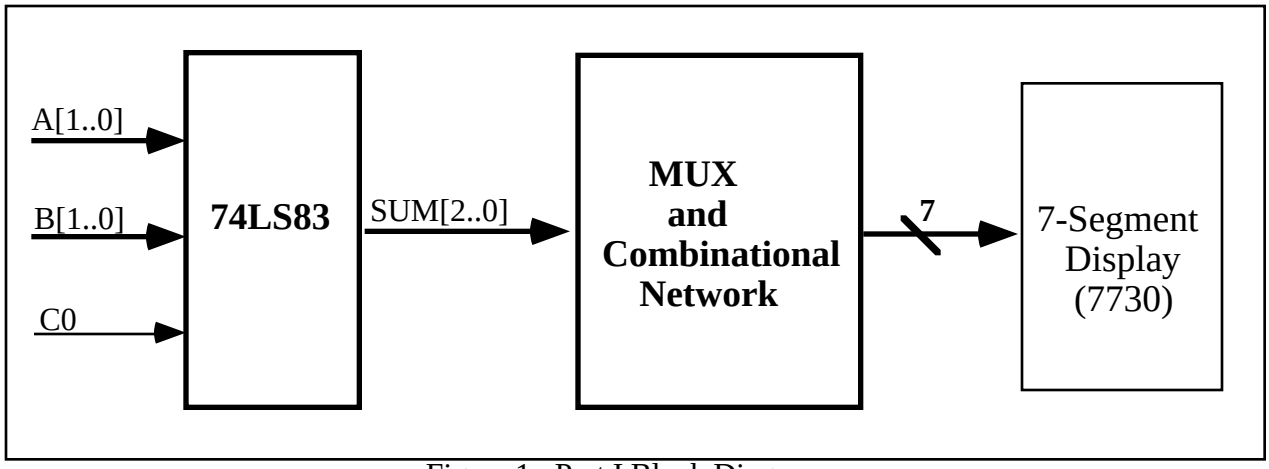

Figure 1: Part I Block Diagram

Part II: Using Programmable Array Logic Device

a) In this part of the lab, you will implement your 2-bit adder using a Programmable Array Logic (PAL) device. The entire combinational network designed in Part I can fit onto a single PAL chip, as shown in Figure 2. (Note that in this part you do not need to include the carry-in bit,C0, although you can include it if you choose. Since 5-variable Karnaugh maps are difficult to work with, we have reduced the number of input variables to 4.)

The PALCE22V10 device has 10 pins which can be configured as outputs and 22 pins which can be used for inputs. (See datasheet on Web.) These chips are reprogrammable so they can be erased and used again. To implement a design using a PAL, we use PALASM software along with a PAL programmer device to actually program the chip. Your TA will ***help*** you in programming the PAL but you must understand the procedures he/she follows in the process.

Develop a set of equations for the 7-segment display inputs in terms of the two 2-bit input numbers, i.e., A_1, A_0, B_1, and B_0.

Your logic equations should be in sum of products form. There are from 8 to 16 product terms available to each of the 10 outputs for the 22V10, with a total of 120 products maximum. Use "*" for product terms, "+" for summation and "/" for complements. For example, an equation with valid syntax would be

A = A1*/B0 + /A1*/B1*B0

Note that "*" takes precedence over "+" in these equations. The PALASM software will check for syntax error during the compilation process. Of course, it can't detect logic errors in your equations.

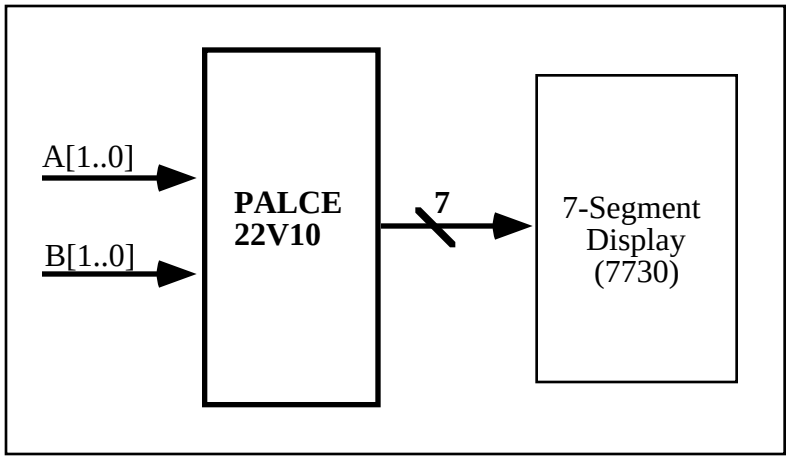

Figure 2: Part II Block Diagram

Using PALASM to enter your design:

1. From a DOS window, create a working directory for yourself under c:\usr.

   i.e. C:\> cd usr

   C:\USR> md yourname

   C:\USR> cd yourname

2. Run the PALASM program from your DOS window.

   i.e. C:\USR\YOURNAME>palasm

   After you press any key, the program will display the main menu page with the FILE selection opened.

3. Highlight **Change directory** and press enter. Type in the name of your working directory at the prompt.

4. Highlight **New design file** using cursor keys and press enter. A window will appear for you to enter your file name. The extension must be .pds, i.e., test.pds

Figure 3 shows a typical format for the input file which will be created in the next steps. The equations will be entered through a PALASM screen editor, as explained below.

5. Next a template page will appear. Enter in the header information such as Title, Pattern, Revision, Author, Company, and Date. These are optional fields so don't worry about your entries. You must enter a Chip Name. This can be any string to identify your design (i.e., Mealy, Template, etc.). Next highlight the correct device type. For this lab, you will use PAL22V10. Make sure this choice is specified correctly.

6. To complete the template, you need to identify the pin connections for your chip. Follow the example shown in Figure 3. When you are done specifying the pin connections, press F10 to save the template data. The program will automatically enter the editor so that you can type in your equations.

7. Within the editor, change the Part type specified earlier from PAL22V10 to PALCE22V10. Although this choice is not in the Parts menu, the "CE" designation is needed for some of the newer parts.

8. Use the arrow keys to move the cursor to the EQUATIONS section. Type in the equations for your design. When you are done, press F10. You are now ready to compile your program. You can also enter simulation vectors, as shown in Figure 3, in order to verify your design.

9. Use the arrow keys to move to the RUN menu. Highlight **compile** and press return. Your program will take several minutes to compile. If you entered simulation vectors, you can highlight **Both** in order to compile and simulate your design.

10. Assuming that your program compiled without errors, you are now ready to check the simulation. Use the arrows to open the VIEW menu and highlight **Simulation trace** to view the simulation results. Once you have verified your design, you are ready to transfer the result file, with a (.jed or .jdc) extension, to the PAL programmer. Exit the palasm program by pressing Escape and answering Y to the question.

11. Setting up the PAL programmer:

    The TA will help you with this part.

    Be sure to verify that the checksum at the end of your output file is the one calculated by the PAL programmer software. You can also verify the final checksum of the programmed part to make sure that the part has been programmed correctly.

Test your programmed PAL by using it in a circuit with a 7-segment display by properly connecting the inputs and outputs in your circuit.

Have your TA verify your working circuit.

Lab Report

Follow the standard format for lab reports. Include the following items in your report:

- Lab cover sheet with TA verification for pre-lab and for circuit performance.
- Graded pre-lab.
- Complete paper design for Part II, showing K-maps and minimized logic equations.
- PALASM source code

```
;PALASM Design Description

;---------------------------------- Declaration Segment ------------
TITLE    PALASM EXAMPLE FILE
PATTERN  EXAMPLE
REVISION 1
AUTHOR   LH
COMPANY  UCD
DATE     12/14/94

CHIP  EXAMPLE  PALCE22V10

;---------------------------------- PIN Declarations ---------------
PIN  2        w              COMBINATORIAL ; INPUT
PIN  3        x              COMBINATORIAL ; INPUT
PIN  4        y              COMBINATORIAL ; INPUT
PIN  5        z              COMBINATORIAL ; INPUT
PIN  12       gnd                    ;
PIN  14       a              COMBINATORIAL ; OUTPUT
PIN  15       b              COMBINATORIAL ; OUTPUT
PIN  16       c              COMBINATORIAL ; OUTPUT
PIN  17       d              COMBINATORIAL ; OUTPUT
PIN  18       e              COMBINATORIAL ; OUTPUT
PIN  19       f              COMBINATORIAL ; OUTPUT
PIN  20       g              COMBINATORIAL ; OUTPUT
PIN  24       vcc                    ;

;---------------------------------- Boolean Equation Segment ------
EQUATIONS

a = /z*/y*/x*w + /z*y*/x*/w + z*y*/x*w + z*/y*x*w
b = /z*y*/x*w + y*x*/w + z*y*/w + z*x*w
c = /z*/y*x*/w + z*y*x + z*y*/w
d = /y*/x*w + y*x*w + /z*y*/x*/w + z*/y*x*/w
e = /z*w + /y*/x*w + /z*y*/x
f = z*y*/x*w + /z*x*w + /z*/y*w + /z*/y*x
g = /z*/y*/x + /z*y*x*w + z*y*/x*/w

;---------------------------------- Simulation Segment ------------
SIMULATION

trace_on z y x w a b c d e f g
setf /z /y /x /w
setf /z /y /x w
setf /z /y x /w
setf /z /y x w
setf /z y /x /w
setf /z y /x w
setf /z y x /w
trace_off

;---------------------------------------------------------------------
```

Figure 3. EXAMPLE PALASM FILE