Lab 5 - PAL-based Combination Lock

Introduction

You will design, assemble and test a combination lock based on a programmable logic device (PLD). The behaviour of this device will be same as the device in the previous lab but now you will implement it using a PLD.

You will be supplied with a Lattice GAL16V8 PAL that you will use to build the system on a prototyping board in the lab. You will use the CUPL logic design language to generate the fuse map for the PAL and you will program the device using the same device programmer you used in Lab 4. You will demonstrate your device to the TA and hand in a listing of the CUPL code.

The GAL16V8

Each group of two students will be supplied with a GAL16V8 PAL. This device's configuration is stored in EE (electrically erasable) programmable memory.

This PAL is an example of simple PLD. It has 8 outputs capable of computing 8 sum-of-product functions from 8 inputs, the 8 outputs and their complements (a maximum of 32 variables in each product term). The equation for each output is limited to a maximum of 8 terms (i.e. only 8 of the 2^{16} possible inputs can result in a '1' output). Each of the 8 outputs can be configured as a flip-flop if desired (clocked from an external clock input). This PAL's outputs are set to 1's when the device is first powered up. A complete data sheet is available on the course web page.

Specifications

Inputs and Outputs

The device has four slide switches that allow you to enter a number (in binary) and one pushbutton switch to "enter" the number. This pushbutton acts as the clock signal for the state machine.

The device has one LED output which is used to display the state of the lock: closed (LED off) or open (LED on). For diagnostic purposes you should also connect the state variables to LEDs.

The lock should open when the correct sequence of three digits has been entered. These three digits should be the first three digits of *your* student number. If an error is made in entering any of the digits, the lock should start looking for the first digit again (i.e. the correct digits have to be entered in order and without any errors). Once the last digit has been entered the microcontroller should turn on the lock LED to indicate that the lock has been opened. The next button push should close the door again (all LEDs off) and the lock will start waiting for the three digits to be entered again.

Design a state machine that implements this device.

Circuit Description

You will wire up your interface circuit on the same "DigiDesigner" prototyping unit used for the previous lab.

You should connect the switches, pushbutton and LEDs to the pins indicated below.

Please observe the warnings given in the previous lab concerning the susceptibility of the devices to incorrect voltages and static.

Assembling the Circuit

A schematic of the circuit is shown in Figure 1.

- Plug the chip into the breadboard.
- Connect pin 10 to the ground (0V) power supply output.
- Connect pin 20 to the +5 volt power supply output.

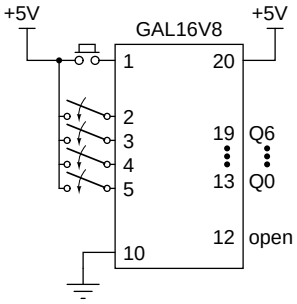

Figure 1: Lock circuit.

- Connect the pushbutton to pin 1 (the clock input).
- Connect the slide switch outputs to pins 2 through 5.
- Connect the "open" LED to pin 12.
- Connect LEDs to display the current state to pins 13 through 19 (you may not need this many state variables).
- Make sure the chip is properly connected to both +5 V and ground rails. Reverse-biasing the chip or it's inputs will destroy the chip. Double check your connections before applying power.

Compiling your Design

You will use a free version of a CUPL "compiler" that runs under Windows on the PCs in the lab. If want to run CUPL on another PC you can download the file from the course web page and install it. A tutorial manual is available at (`http://www.chipprogrammers.com/cupl/docs/index.htm`) although the lecture notes and on-line help will probably be enough.

Enter the Design

Run WinCUPL and use *File—New* to create a template file. Fill in the comments section at the beginning with appropriate values. Fill in the pin definition section with mnemonic pin names for the signals listed above. Enter the appropriate logic equations into the section for logic equations. Save your design file using *File—Save*.

CUPL has features that simplify the design of state machines but for this lab you should do the design "by hand" and enter in the logic equations.

"Compile"

Check the Options dialog box to make sure a JEDEC file and a listing file will be generated. Use the *Run—Device Specific Compile* to generate the file into a fuse map in "JEDEC" format that can be used by the device programmer. Copy the resulting file (e.g. `lab5.jed`) to a floppy and use it to program the chip as described below.

Programming the PLD

You will use the same device programmer as in the previous lab. Please use this computer only for programming the devices. Type `ACCESS` to run the device programmer software. The device programmer software is menu-driven. Select the following options:

Device	PAL
MFR	Lattice
Type	GAL16V8A

This will start a second program that is specific to PAL devices. First select the option to "Load File". Enter the name of your file (e.g. `lab5.jed`). Then select the option to automatically erase, blank check, program and verify.

It will take a few seconds for the device to be programmed. You may remove the device when the LED on the programmer goes on.

Submitting The Lab

Once your device is working properly print a program listing (e.g. `lab5.lst`) and demonstrate its operation to the TA. Return the chip to the TA at the end of the lab. You must return the part to get a mark for the assignment.

Your lab report should include a description of the state machine including state transition and output tables (for your own student number), a state transition diagram and the CUPL source listing.