Massachusetts Institute of Technology
Department of Electrical Engineering and Computer Science
6.111 - Introductory Digital Systems Laboratory (Spring 2008)
Laboratory 2 (Car Anti-Theft System) Check Off Sheet

Student Name: _____

TA Signature / Date:_____

Please have the following available during checkoff:

- FSM State Transition Diagram ..... ☐
- Verilog Code (available on your computer screen) ..... ☐

Be able to demonstrate your design working on the labkit:

- You will first be asked to demonstrate regular operation with default values ..... ☐
- You will be asked to reprogram your time values and continue operation ..... ☐
- You will be asked to demonstrate functionality of the fuel pump logic ..... ☐
- You will be asked to demonstrate functionality of your siren sound generator ..... ☐

Be able to respond to any of the following questions (and possibly others) …… ☐

- What could happen if an input were not synchronized to the clock?
- Describe your synchronizer module and why it is important.
- Describe your fuel pump and siren sound generator logic.
- Describe your divider module.
- Describe the design flow for your system.

Massachusetts Institute of Technology
Department of Electrical Engineering and Computer Science
6.111 - Introductory Digital Systems Laboratory

Laboratory 2 – Car Anti-Theft System (Spring 2008)

Issued: February 20, 2008
Check-Off Due: March 3, 2008 at 10PM in lab
Report Due: March 5, 2008 in class

Porsche Anti-Theft System

A recent MIT grad just completed a very successful IPO of her Cambridge startup and celebrated by purchasing a new Porsche. Though the car has a built in anti-theft system, she is concerned since this is a standard factory unit and many people know how to disable it. So she's looking for someone to design and build a system with some hidden security features only you two will know about! Your job is to create a working prototype and a written report that will win you the job.

In this lab you will implement an anti-theft system that uses several interacting FSMs to process sensor inputs and generate the appropriate actuator control signals. This lab provides you with a design methodology that will be useful in future labs and your final project. Although not required, it is suggested that you schedule a conference with a member of the teaching staff to review your design. This will help catch any major mistakes early in the process.

Procedure

1. Read through the lab to understand the desired functionality, determine your approach and complete a design. Although not required, you may find it helpful to schedule a conference with a member of the course staff to review your design. This will help catch any major mistakes early in the process.
2. Prototype your design using the labkit. After you verify the anti-theft system's functionality get a member of the course staff to check you off. Be ready to demonstrate the functionality of your implementation and to answer the questions proposed in the Checkoff List. Send your Verilog code to one of the TAs as an e-mail attachment.
3. Write a detailed report (see the writing guidelines at the end of this writeup) that will satisfy the written portion of the EECS CI-M requirement. You'll submit a draft of the report for evaluation by the Writing Across the Curriculum Program and then, based on their comments, submit a final revision for grading.

Description of Anti-Theft System

Since your client is completely focused on her start-up, she wants an anti-theft system that's highly automated. The system is armed automatically after she turns off the ignition exits the car (i.e., the driver's door has opened and closed). The system arms itself T_ARM_DELAY after all

the doors have been closed; that delay is restarted if a door is opened and reclosed before the alarm has been armed.

Once the system has been armed, opening the driver's door the system begins a countdown. If the ignition is not turned on within the countdown interval (T_DRIVER_DELAY), the siren sounds. The siren remains on as long as the door is open and for some additional interval (T_ALARM_ON) after the door closes, at which time the system resets to the armed but silent state. If the ignition is turned on within the countdown interval, the system is disarmed.

Always a paragon of politeness, your client opens the passenger door first if she's transporting a guest. When the passenger door is opened first, a separate, presumably longer, delay (T_PASSENGER_DELAY) is used for the countdown interval, giving her extra time to walk around to the driver's door and insert the key in the ignition to disarm the system.

There is a status indicator LED on the dash. It blinks with a two-second period when the system is armed. It is constantly illuminated when either the system is in the countdown waiting for the ignition to turn on or if the siren is on. The LED is off when the system is disarmed.

So far this all is ordinary alarm functionality. But you're worried that a knowledgable thief might disable the siren and then just drive off with the car. So you've added an additional *secret* deterrent -- control of power to the fuel pump. When the ignition is off power to fuel pump is cut off. Power is only restored when first the ignition is turned on and then the driver presses both a hidden switch and the brake pedal simultaneously. Power is then latched on until the ignition is again turned off.

The diagram below lists all the sensors (inputs) and actuators (outputs) connected to the system.

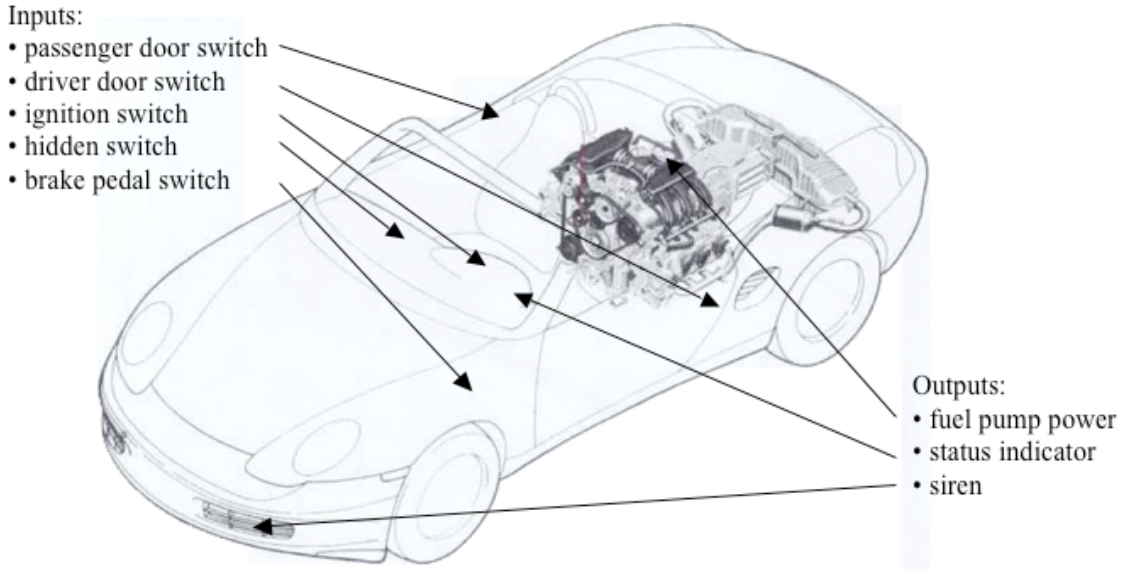

Figure 1: System diagram showing sensors (inputs) and actuators (outputs)

The system timings are based on four parameters (in seconds):

1. the delay between exiting the car and the arming of the alarm (T_ARM_DELAY),
2. the length of the countdown before the alarm sounds after opening the driver's door (T_DRIVER_DELAY)
3. the length of the countdown before the alarm sounds after opening the passenger door (T_PASSENGER_DELAY),
4. and the length of time the siren sounds (T_ALARM_ON).

The default value for each parameter is listed in the table below, but each may be set to other values using the Time_Parameter_Selector, Time_Value, and Reprogram signals. Time_Parameter_Selector switches specify the parameter number of the parameter to be changed. Time_Value switches are a 4-bit value representing the value to be programmed -- a value in seconds between 0 and 15. Pushing the Reprogram button tells the system to set the currently selected parameter to Time_Value. Note that your system should behave correctly even if one or more of the parameters is set to 0.

Default Timing Parameters

Interval Name	**Symbol**	**Parameter Number**	**Default Time (sec)**	**Time Value**
Arming delay	T_ARM_DELAY	00	6	0110
Countdown, driver's door	T_DRIVER_DELAY	01	8	1000
Countdown, passenger door	T_PASSENGER_DELAY	10	15	1111
Siren ON time	T_ALARM_ON	11	10	1010

Block Descriptions/Implementation

The following diagram illustrates a possible organization of your design into modules. This diagram is a high level view and does not show every necessary signal.

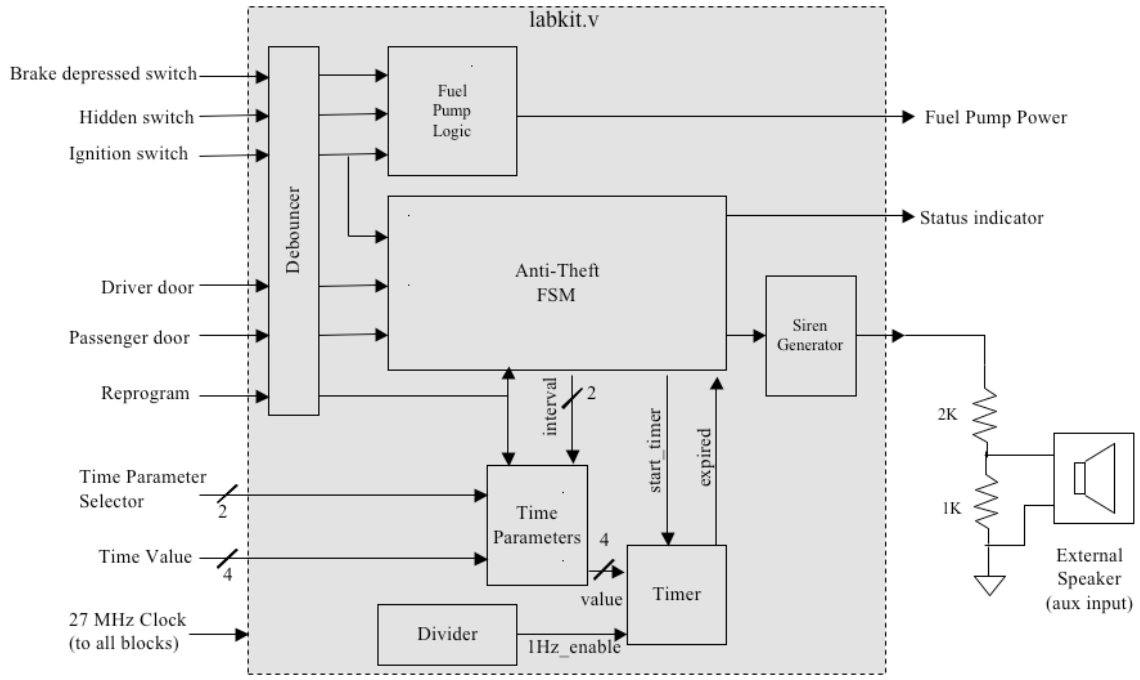

Figure 2: Block Diagram of Anti-Theft System.

You should implement this lab by programming each module individually and then instantiating and connecting the modules together in the toplevel labkit.v module. Then compile your implementation using the Xilinx tools, download it to your kit's FPGA, and demonstrate its operation. Please use the following labkit devices for the various sensors and actuators:

Sensor/Actuator	Labkit device
Hidden switch	button0
Brake depressed switch	button1
Driver door switch	button2
Passenger door switch	button3
Ignition switch	switch[7]
Time_parameter_selector	switch[5:4]
Time_value	switch[3:0]
Reprogram	button_enter
Status light	led[0]
Fuel pump power	led[1]
Siren output	user1[0]

Below is a more detailed description of each module:

Debouncer

Your clocked state machine is controlled by several asynchronous inputs that might be changed by the user at any time, potentially creating a problem with metastability in the state registers if one of the inputs changes too near a rising clock edge. In general asynchronous inputs need to be synchronized to the internal clock before they can be used by the internal logic. synchronize.v (available on the Handouts page of the course website) is a Verilog implementation of a pulse synchronizer that can convert an asynchronous signal to a synchronous one with acceptably low probability of metastability in the synchronous signal.

A second problem arises from the mechanical "bounce" inherent in switches: as a metal contact opens and closes it may bounce a couple of times, creating a sequence of on/off transitions in rapid succession. So you need to use debouncing circuitry to filter out these unwanted transitions. debounce.v (available on the Handouts page of the course website) is a Verilog implementation of a digital retriggerable one-shot that requires that an input transition be stable for 0.01sec before reporting a transition on its output. This module happens to produce a synchronous output, so a separate synchronizer is not required. You should use an instance of the debounce module to debounce any switch inputs you use in your design.

Time Parameters

The time parameters module stores the four different time parameter values. The module acts like a 4-location memory that's initialized with default values at power on, but may be reprogrammed by the user at any time. Using the 2-bit Interval signal, the Anti-theft FSM selects one of the four parameters to be used by the Timer module.

On power on, the parameters should be set to the default values specified above. However the user may modify any of the values by manipulating Time_Parameter_Selector (2 bits), Time_Value (4 bits), and Reprogram. Whenever a parameter is reprogrammed, the FSM should be reset to its ARMED state (after which it may transition immediately to another state depending on the sensor inputs).

Divider

The divider converts the 27MHz master clock into an 1HZ_enable signal that's asserted for just 1 cycle out of every 27,000,000 cycles (i.e., once per second). The 1HZ-enable is used by the Timer module and for making the LED blink with a two-second period. The divider needs to reset when Start_Timer is asserted (see Timer module below) so that the first 1HZ_enable after the timer starts to count comes a full second after the timer has been started.

Timer

The timer counts down the number of seconds specified by the Time Parameter module. It initializes its internal counter to the specified Value when Start_Timer is asserted and decrements the counter when 1Hz_enable is asserted. When the internal counter reaches

zero, the Expired signal is asserted and the countdown halts until Start_Timer is once again asserted.

Anti-theft FSM

This finite state machine controls the sequencing for the system. The system has four major modes of operation:

1. Armed. The status indicator should be blinking with a two-second period; the siren is off. If the ingition switch is turned on go to Disarmed mode, otherwise when a door opens start the appropriate countdown and go to Triggered mode. This is the state the FSM should have when the system is powered on.
2. Triggered. The status indicator light should be constantly on; the siren is off. If the ignition switch is turned on, go to Disarmed mode. If the countdown expires before the ignition is turned on, go to Sound Alarm Mode.
3. Sound Alarm. The status indicator light and siren should be constantly on. The alarm should continue to sound until either T_ALARM_ON seconds after all the doors have closed (at which point go to Armed mode) or the ignition switch is turned on (at which point go to Disarmed mode).
4. Disarmed. The status indicator light and siren should be off. Wait until the ignition switch is turned off, followed by the driver's door opening and closing, then after T_ARM_DELAY seconds go to Armed mode.

Note that more than one FSM state may be needed to implement the required functionality of each mode, i.e, your state transition diagram will have many more than 4 states.

Fuel Pump Logic

This simple FSM controls the power to the fuel pump. Power is disabled when the ignition switch is turned off and only reenabled when the appropriate sequence of sensor values is received (see description above).

Siren Generator

This module generates an audio-frequency square wave (i.e., a sequence of alternating 0's and 1's) that can be used to drive an external speaker. At a minimum your generator should alternate at couple of second intervals between a 400Hz tone and a 700Hz tone. But fancier effects are possible -- see below.

Connect the output of the siren generator to the user1[0] output of the FPGA, which appears on a connector at the top of the labkit board. Use a wire to connect that signal to the breadboard and use a couple of resistors as shown in the block diagram above to create a circuit on the protoboard where you can connect up the auxillary input of the external Radio Shack speaker box (there should be 12 speaker boxes kicking around the lab -- check over by the TA table). Note that the speakers have their own power supply module which must be plugged in order for the speakers to work.

You are anxious to make a good impression on your client in the hopes of getting the design commisions. So once you have implemented the basic functionality described above, create a different sound effect with your Siren Generator. Some ideas:

- sweep the frequency of the audio tone from 400Hz to 700Hz (or vice versa) and then repeat. Produces a repeating rising/falling tone instead just alternating between the two frequencies.
- produce a warbling tone (rapidly switch between a couple of frequencies).
- Alternate between a tone and silence.
- Sequence through different effects.

Labkit.v

The labkit.v template top-level file can be downloaded by right-clicking on the link and choosing "Save As". Labkit.v allows a user to utilize all of the labkit features including buttons, switches, leds, logic analyzer pins, user input/output pins, audio video capabilities, etc. We suggest that you work through the software tutorials "Getting Started with ISE" and "Programming the Labkit", which guide you through how to create a simple counter using the labkit LEDs and how to program it onto the FPGA. After working through the tutorials you will be ready to program your anti-theft system onto the labkit FPGA.

You will need to instantiate all submodules in this top level file as well as create wires to connect the submodules. For example, to connect and instantiate your divider and timer blocks you could add something like the following lines in the labkit.v file:

```
//declare wires as shown in Figure 2 to connect the submodules:
wire reset_sync;
wire one_hz_enable;
wire [3:0] value;
wire expired, start_timer;

//instantiate the submodules and wire their inputs and outputs
//(use the labkit's clock_27mhz as the clock to all blocks)
divider divider1 (clock_27mhz, reset_sync, one_hz_enable);
timer timer1(clock_27mhz, reset_sync, value, expired, start_timer);
```

In addition, you will need to connect the input buttons and switches that you use to the appropriate submodules. There are verilog statements in labkit.v that set default values for all outputs, e.g., the leds and user1[0]. You'll need to replace those statements with your own logic that generates the appropriate driving values.

Debugging

Inevitably your design won't be 100% correct the first time. Debugging by just observing the outputs talked about above may not work -- you'll need more info to figure out what's happening. Consider using the display_16hex.v module (available on the Handouts webpage) to show 16 hex digits of info (64 bits) on the labkit's 16-character flourescent display. You might use one digit to show the current state of the FSM, another digit to indicate currently selected time

parameter, yet another digit to show the current value of the timer's module internal counter, etc. You'll want to use this handy module for displaying debugging info for all your designs!

Another useful tool for capture sequences of debugging data is the logic analyzer. There are adapter cards that hook the analyzer data pods directly to the FPGA signals analyzer1_data[15:0] (pods A0 and A1) and analyzer3_data[15:0] (pods A2 and A3). The FPGA signal analyzer1_clk is hooked to CK1 of the logic analyzer and analyzer3_clk is hooked to CK0. See analyzer_mapping.pdf for a diagram of the connections.

Guidelines for Lab 2 Report

6.111 will count as a Communications Intensive Major (CI-M) subject. This document specifies our technical expectations for the formal write-up of Laboratory 2 and outlines the quality of writing we hope to see in your report. Laboratory 2 will eventually count for 20% of your final grade with ten percent being assigned primarily on the technical merits of your lab work and the content of your lab report. This portion of your grade will be assigned by the 6.111 staff. The other ten percent will be assigned by the writing department. The writing department will provide feedback on your writing for the first version of your lab 2 report. You are required to make a revised version which will then be graded by the writing department. Your grade is based on the quality of writing found in your revised formal lab report. **Please turn in two copies of your Lab 2 report. One version (to be evaluated by the staff) should include the required appendix and the check-off sheet. Clearly mark the version going to the writing department on the cover page (Submitted as a part of the CI-M requirement)**

Your report should be organized as a design proposal from an engineering team written with the goal of convincing the client that your design will meet/exceed the functional requirements and has been carefully engineered for reliable operation.

Keep in mind that your aim is to get your client to choose *both* your design and your engineering team; you are competing to win a job. Your design has to fulfill the requirements specified by the client. Your writing has to clearly and completely convey to the client what you have done and what you will do if you win the job.

Your goal is for both your design and your writing to be essentially flawless. Your professional presentation and communication constitute an argument; they provide evidence to the client that you are conscientious and professional in everything you do. If your design is excellent but your proposal betrays a lack of attention to detail, your client will likely choose another firm. If there are obvious flaws in your written work, she may reason that your attention to digital design may also be flawed.

If someone found your report, with an attached letter of transmittal lying on Vassar Street, you would want them to fully believe they had found business documents from an engineering firm: clear, complete, and compelling. The documents would reflect the precision, attention to detail, and clear presentation of information and analysis that you would want from an excellent engineering firm. Anyone finding these documents would want to seek you out to commission you to design a car alarm for them!

Your submission will consist of the following parts:

- Letter of Transmittal
- Design Report
  - Title and Abstract
  - Table of Contents
  - List of Figures
  - Overview
  - Description
  - Conclusion
  - References
  - Appendices

Your report should be prefaced by a letter of transmittal. See Mayfield for a sample letter and guidelines. For the most part, these letters will say the same thing "attached, please find our proposal." You will have to invent a little information for your client, Jane Porsche, including an address. Make your own address-your personal address or the address of your company-6111 Massachusetts Avenue. Do NOT put a class assignment header on your letter, or on your report. Do NOT double space your letter or your report.

The combined written parts of the assignment should be in a range of 4000-6000 words not including appendicies. As long as you convey all the information you need to convey, and do so clearly and in good prose, you do better to aim for concision.

The core parts of the report -- the overview, description, and conclusion -- should read logically and smoothly, providing all the high level information the client needs to understand your design. Much of the more dense detail should be relegated to appendices. There should be no verilog code in the body of the report, for example. Which diagrams or figures you choose to keep in the body of the text is a judgment call for you to make. Make sure that you cite any parts of the report that are not your own creation, including figures or diagrams.

The Overview section should describe the overall goals of the design followed by a concise specification of the functionality. Find a way to present the specification so that the client can easily determine if it does everything she would like it to.

In this case your client is technically sophisticated, so the Description section should provide information about how each function is implemented. Focus on the key/unique elements of the design, e.g., the Anti-Theft FSM (a state transition diagram would be good here) your siren generator (did you extend the basic functionality? How does this part of your circuit work?).

The report guide referred to at the top of this document provides an overview of and advice on each of the constituent parts of the report. The writing side of the assignment will also be discussed in class on Friday February 22, 2008. Should you have any questions about the assignment, they should be directed to the WAC Program lecturers assigned to this course: Don Unger (donunger@mit.edu) or Ben Miller (bjmiller@mit.edu).

Writing Considerations

While the 6.111 staff will be grading your lab report for technical content, we will also be taking the quality of your writing into consideration when assigning grades. As such we ask that all figures and tables be created using a computer (i.e. no handwritten submissions). Because of the fact that ten percent of your grade in 6.111 is based on the writing quality found in your formal report we expect you to dedicate a significant amount of your time in writing this report to polishing and editing. On the course webpage we have made available to you a number of writing resources that we hope you will find useful. If you have any questions or concerns relating to your formal writeup please contact the 6.111 staff and we will do our best to help you.

Useful links

- labkit.v
- labkit.ucf
- synchronize.v
- debounce.v
- display_16hex.v
- analyzer_mapping.pdf
- Report guide [HTML]
- The Mayfield Handbook of Technical and Scientific Writing
- Writing Guidelines for Engineering and Science Students (see the Design Reports section)