You have to work in a group of two
Main Task: Design a digital system using VHDL, verify the code using VHDL simulator and realize it using FPGA
Please submit your answers (files) electronically through Moodle.

Project : Car Security System Design

You and your partner are assigned by your senior supervisor the task of a digital car security system.

1) Design

Your designed system should have the following functional requirements.

1.1) Functional Requirements:

- The system is armed automatically after the ignition has been turned off, both doors have been closed, and ARM_DELAY has passed. The delay is restarted if a door is opened and reclosed before the alarm has been armed.
- Once the system has been armed, the system begins a countdown when the driver's door has been opened. If the ignition is not turned on within the countdown interval (DRIVER_DELAY), the siren turns on. The siren remains on as long as the door is open and for some additional interval (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.
- The system also begins a countdown when the passenger's door has been opened. However, always a paragon of politeness, the driver opens the passenger door first if the driver is transporting a guest. When the passenger door is opened first, a separate, presumably longer, delay (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.
- The system timings are based on four parameters, as shown in the following table.

Symbol	Descriptions	Parameter Number	Default Time (sec)	Time Value
ARM_DELAY	Delay between exiting the car and the arming of the alarm	00	6	0110
DRIVER_DELAY	Length of the countdown before the alarm sounds after opening the driver's door	01	8	1000
PASSENGER_DELAY	Length of the countdown before the alarm sounds after opening the passenger 's door	10	15	1111
ALARM_ON	Length of time the siren sounds	11	10	1010

Each parameter has a default value, but can be set to other values configuring as follows:

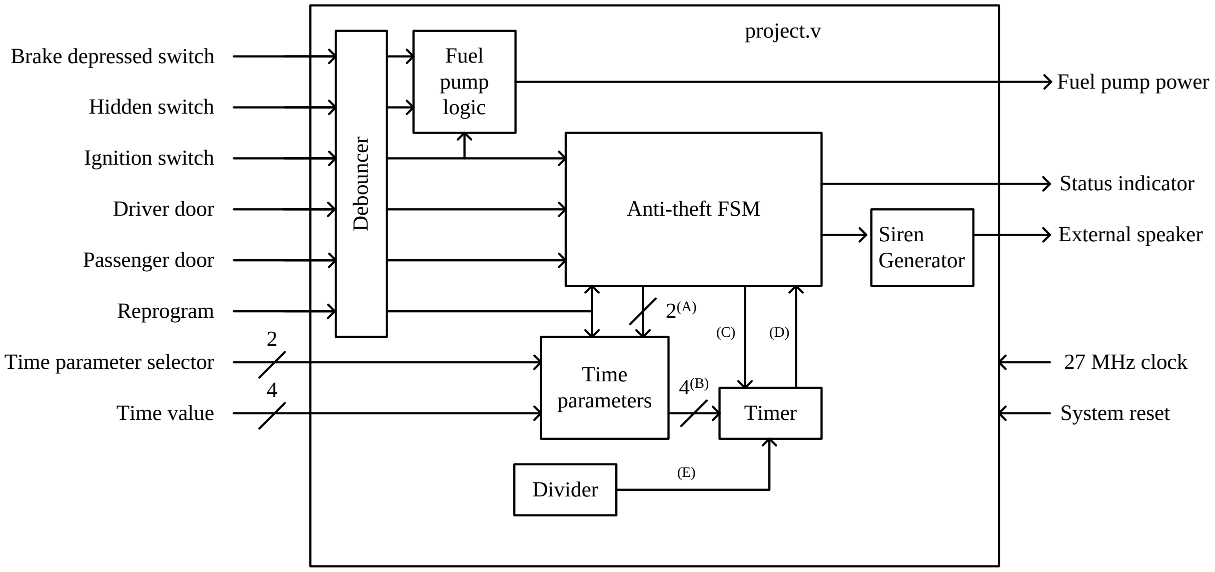
  - `Time_Parameter_Selector`: It specifies which parameter has to be changed.
  - `Time_Value`: It specifies the value (in seconds between 0 and 15) to be programmed.
  - `Reprogram`: The system will set the selected parameter (indicated by `Time_Parameter_Selector`) to `Time_Value`, when Reprogram is ON.

  Note that the system should behave normally if one or several parameters is set to 0.

- There should be a status indicator LED on the dash. It blinks with a three-second period when the system is armed. It is constantly illuminated 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.
- Smart thieves might disable the siren and then just drive off with the car. So an additional secret deterrent has to be added for controlling of the 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.

1.2) Architectural Design

Based on the functional requirements, your supervisor has designed a digital system architecture (with I/O pin assignment) for implementation (You may assume that the clock signal runs at 1 Hz.):



(A) : interval (B) : value (C) : start_timer (D) : expired (E) : 1Hz_enable

Sensor/Actuator	Pin	Sensor/Actuator	Pin
`Hidden switch`	button0	`Ignition switch`	switch[7]
`Brake depressed switch`	button1	`Time_parameter_selector`	switch[5:4]
`Driver door switch`	button2	`Time_value`	switch[3:0]
`Passenger door switch`	button3	`Reprogram`	bt_enter
`Status light`	led[0]	`Siren output`	user1[0]

Fuel pump power	led[1]	System reset	bt_down

A more detailed description of each module is as follows:

1.2.1) 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 car security 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 through configurations. 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).

1.2.2) 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 the clock signal is asserted. When the internal counter reaches zero, the Expired signal is asserted and the countdown halts until `Start_Timer` is once again asserted.

1.2.3) Divider

The divider converts the 27MHz master clock into an one_hz_enable signal that's asserted for just 1 cycle out of every 27,000,000 cycles (i.e., once per second). The one_hz_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 one_hz_enable after the timer starts to count comes a full second after the timer has been started.

1.2.4) Car Security FSM

This FSM controls the sequencing for the system. The system has four major modes of operation:

1. **"Armed"**: The status indicator should be blinking with a three-second period, and the siren should be off. If the ignition 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 **"Alarm"** Mode.
3. **"Alarm"**: The status indicator light and siren should be constantly on. The alarm should continue to sound until either `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 `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 four states.

1.2.5) Fuel Pump Logic

This logic module controls the power to the fuel pump. Power is disabled when the ignition switch is turned off and only re-enabled when the appropriate sequence of sensor values is received.

1.2.6) 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, based on the audio codec chip on the FPGA. At a minimum your generator should alternate at couple of second intervals between a 400Hz tone and a 700Hz tone. So once you've implemented the basic functionality described above, create a different sound effect with your siren generator. Some ideas are as follows:

- 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 during left and right channels

You may need to spend time on understanding audio codec chip and SD card reader for generation of sounds.

1.3) Hints for the Design

You are suggested to implement the system by programming each module individually and then instantiating and connecting the modules together in the high-level module. Similarly, you can start the verification by building a test procedure for each module and then the overall test procedure to verify the expected behavior of the system on several sequences of inputs that you expect to test on a real system.

2) Deliverables for Assessments

Please turn in each of the following items:

- Design files:
  - Save your overall designed digital system design as `car_s_xx.vhd`, where `xx` are your group number. Your VHDL should include readable code with comments and consistent indenting, correct use of blocking and non-blocking assignments, use of "others" in case statements. There should be no long nested if statements.
- A PDF report, which contains:
  - Your FSM design, including a completed state transition diagram for your FSM.

(This homework is adopted from MIT 6.111 Introductory Digital Systems Laboratory. Copyrights of the adapted materials thus belong to the contributor, and should be observed strictly.)