Digital Design Verification

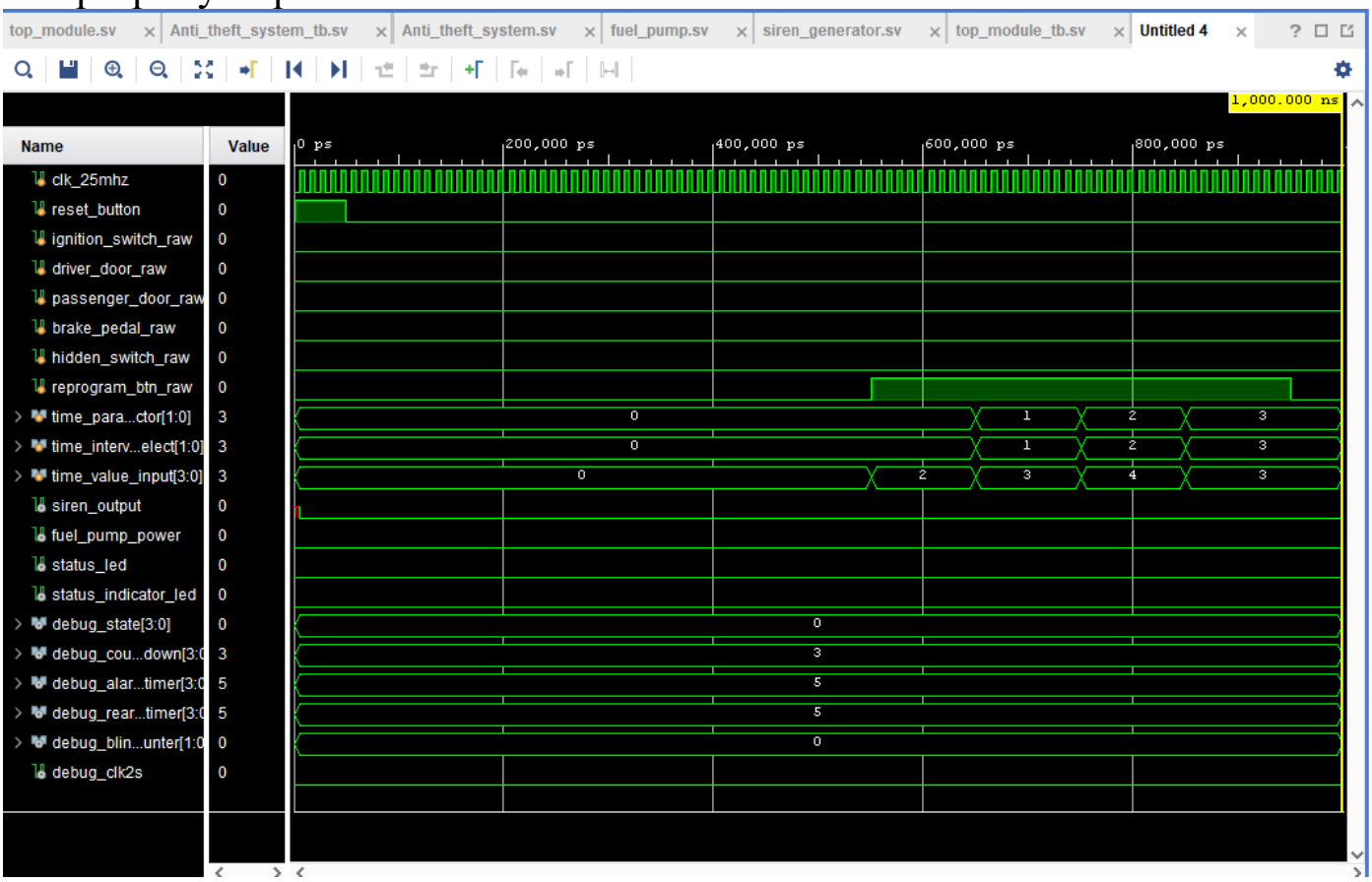
Lab Manual # 10 – Finite State Machine

(Anti-Theft System)

Name	*Khalil Rehman*
Instructor	*Sir Musadiq*
Date	*06 Aug 2025*

1. In-Lab Tasks: (*Write your lab task & screenshots here*)

Not properly implemented:



Q#1

If an input is not synchronized to the system clock, it can cause metastability in sequential circuits. This occurs when a flip-flop/FSM receives a signal that is changing close to the clock edge, resulting in an unpredictable or unstable output (neither clearly 0 nor 1).

Q#2

1. Fuel Pump Logic:

The fuel pump logic ensures that fuel is only supplied to the engine after a secure unlock sequence. It follows this rule:

fuelPumpPower = (ignition && fuelPumpPowerReg) || (ignition && brake && hidden);

- If the ignition is ON and both the brake and hidden switch are pressed, power to the fuel pump is enabled.
- Once enabled, the pump stays ON as long as ignition remains ON.
- If the ignition turns OFF, the system resets and requires the unlock sequence again.

2. Siren Sound Generator:

The siren generator outputs a square wave sound signal for the siren speaker. It uses a counter and clock divider to alternate between 440 Hz and 880 Hz, creating a siren effect:
A tone counter is increasing continuously.

- The high bits of the counter are used to vary the output frequency.
- When siren signal is 1, the speaker output toggles based on the generated tone.

Q#3

The divider module is implemented inside the timer module. It generates a 1 Hz clock pulse from the 25 MHz system clock using a 25-bit counter:

if (clk_count == 25_000_000 - 1)

- The counter resets every 25 million cycles (1 second at 25 MHz).
- A one-cycle pulse (clk_1) is generated at that moment.
- This clk_1 signal is used for: FSM countdown timers (countdown, alarm_timer, rearm_timer)
- Status LED blinking every 2 seconds (via clk2s)

Q#4

Describe the design flow for your system.

The system was designed using a modular FSM-based approach. Here's the design flow:

Step 1: Requirements Analysis
- I began by thoroughly understanding the operating modes: Armed, Triggered, Siren, Disarmed
- Identify required timers:
  ARM_DELAY, DRIVER_DELAY, PASSENGER_DELAY, ALARM_ON
- Then Recognize input/output devices: switches, doors, ignition, speaker, fuel pump

Step 2: FSM Design
- I draw the FSM state diagram based on operating conditions
- Assign meaningful state names
- Identify transitions and output logic for each state

Step 3: Module Creation with their Testbench
- Anti_theft_system: Central FSM controller
- Debounce: Cleans mechanical input signals
- Timer & Divider: Generate 1 Hz pulses for real-time delays
- Time Reprogram Module: Allows dynamic time configuration
- Fuel Pump Logic: Separate module for hidden sequence logic
- Siren Generator: Provides alternating sound wave to speaker

Step 4: Integration
- Instantiate all modules in top_module
- Connect signals properly using cleaned (debounced) inputs
- Need to check FSM states match expected behavior however, during testing, I initially encountered some troubleshooting issues which I will resolved later.

Step 5: Simulation & Debugging
- Using testbench (top_module_tb) to simulate all possible testcases