Final Project

**Objective:**

To create a design a state machine bike lock that can change between Unarmed, Armed and Alarm states by entering or failing to enter a hardcoded passcode.

**Design:**

Unarmed States (State Diagram 1):

A series of states with outputs L = 0 and A = 0. Progress to the Armed series of states by properly entering the hard-coded passcode “2-0-3-3-1-4”. Return to the reset state by entering any wrong input or disconnecting the latch input (setting H = 0). Only one input can be accepted at a time.

Armed States (State Diagram 2):

A series of states with output L = 1 and A = 0. Progress to the Unarmed series of states by properly entering the hard-coded passcode “2-0-3-3-1-4”. Return to the first state of the series under two conditions:

1. If any of the first 5 inputs is “4”
2. If any of the last 5 inputs is the sequence “2-2”

If the passcode is entered incorrectly at any point in the sequence, transition to the Alarm series of states after accepting a total of 6 inputs, unless one of the reset sequences are entered.

Immediately transition to the Alarm series of state if H = 0 at any point.

Alarm States (State Diagram 3):

A series of states with output L = 1 and A = 1. Progress to the Unarmed series of states by properly entering the hard-coded passcode “2-0-3-3-1-4”. Return to the first state of the series with any incorrect input. Allow progression of the sequence regardless of value of the latch input, ‘H’.

VHDL Circuit (VHDL Printout 1):

Design the state machine with the above series of states by declaring states in VHDL.

Debug Circuit (VHDL Printout 2):

Add an additional process to the above design to add three new outputs tied to LEDs. Have an output set to one for each starting state. Set the first output to 1 only when in the starting state for the Unarmed series of states, the second only while in the starting state for the Armed series of states, and third only while in the starting state for the Alarm series of states.

This additional portion of the design is to assist in testing the transition and resetting of the different state series.

**Procedure:**

* Set the I/O pins
  + Set the lock and **Alarm** outputs to the rightmost LEDs
  + Set the clock input and the chain input to the leftmost switch (or automatic clock) and one in from the left, respectively
  + Set the inputs 0-4 to switches with the LSB being the rightmost switch
* Download to FPGA

**Original Test Procedure:**

* Test the circuit as following:
  + First run the “correct” inputs to transition between **Unarmed** -> **Armed** -> **Unarmed**
    - Remember that states should only transition on the positive clock edge, if using a manual clock.
    - The code to change states is “2-0-3-3-1-4”
  + Test at least one reset in the **Unarmed** state
    - “2-0-3-3-1-1-2-0-3-3-1-4” Should end in the **Armed** state with output 10
    - Test with other inputs followed by the state change code each time are in the **Unarmed** state
  + Test transition from **Armed** to **Alarm**
    - First test with only failing with the last input
    - “2-0-3-3-1-1” Should end in the **Alarm** state with output 11
  + In **Alarm** state, test at least one reset and the state change to **Unarmed**
    - “2-0-3-3-3-2-0-3-3-1-4” Should end in the **Unarmed** state with output 00
  + Test an early reset in **Unarmed** state while transitioning to **Armed** state
    - “3-2-3-2-0-3-3-1-4” Should end in the **Armed** state with output 10
  + Test a last input reset and then a first input failed input to transition from **Armed** to **Alarm**
    - “3-2-0-3-3-2-2” Should reset the **Armed** state
      * Verify that output is still 10
    - “3-2-0-3-3-4” Should transition to **Alarm** state with output 11
  + Transition from **Alarm** to **Unarmed** with a different reset
    - “3-2-0-3-3-1-4” Should end in the **Unarmed** state with output 00
  + Test a mid-sequence reset and transition from **Armed** to **Unarmed**
    - “2-0-3-2-2-0-3-3-1-4” Should end in **Armed** state with output 10
  + Test a mid-sequence reset with and then a mid-sequence failed input to transition from **Armed** to **Alarm**
    - “2-0-3-4-0-2-4” Should reset the **Armed** state
      * Verify that the output is still 10
    - “2-0-3-1-2-3” Should end in the **Alarm** state with output 11

**Data:**

**Data Analysis:**

There was one issue encountered between the simulations and testing on the board. The 2-2 reset sequence in the Armed series of states had a typo in the hold state, expecting either inputs 3 or 4 to be held until release instead of inputs 2 or 4. Due to this error, the board transitioned to unused states and had to manually be reset. The simulation did not encounter this error since the input changed each clock cycle. Correcting the typo allowed the circuit to function properly.

**Conclusion:**

This project introduced the concept of writing states directly in VHDL instead of designing the appropriate flip-flops in the code and assigning states from there. This made the process of designing a state-machine significantly faster and allowed for a more complicated design. The project also demonstrated how rapidly designs can expand due to accounting for each possibility within a design, and the benefits of constraining those possibilities to keep a project’s scope reasonable. In the end, I successfully created a circuit that could be applied to electronics to function as the described bike lock.