Final Project Topic Proposal Submission

Submission Date: 11/21/16

Project Title: Bomb Squad: FPGA-based Cognitive Pattern Recognition Training System

Team Name: Fluxuators

|  |  |  |
| --- | --- | --- |
| Team Member Name | Team Member E-mail | Team Member Role |
| Katherine Perez | khperez@uh.edu | Leader |
| Rafael Campos | rhcampos@uh.edu | Member |
| Daniel Lopez | dclopez4@uh.edu | Member |
| Sergio Silva | ssilva2@uh.edu | Member |

# Project Description

Bomb Squad is a cooperative game in which the objective is to perform various tasks and solve puzzles on the FPGA board in order to defuse “the bomb”. The instructions for defusing the bomb will be provided in a bomb defusal manual. There are two player roles in this game: the technician and the specialist. The technician is the player performing the actions and is not allowed to look at the bomb defusal manual at any time. The specialist is allowed to look at manual and must verbally communicate with the technician to perform the steps outlined in the manual.

The user will first input their user credentials, as noted in the manual, onto the slide switches. The timer and puzzles will be displayed on the seven-segment displays, as shown in Figure 1. The LCD will notify the user when the challenge will begin. The puzzle will appear on the four rightmost seven segment displays for a predetermined amount of time. The puzzle will then clear and they must tell the specialist the pattern that they saw. The specialist will then read the manual and tell the user what to input onto the board. The technician will use the push buttons to select which bits to turn on in the seven segment display. There will also be countdown timer to complete these puzzles. The timer in the figure represents a 200-second timer. The timer values will vary depending on the level. If the technician is unable to pass the level before the time runs out, the bomb will “explode”.

Figure 1 | Example of Timer and Puzzle

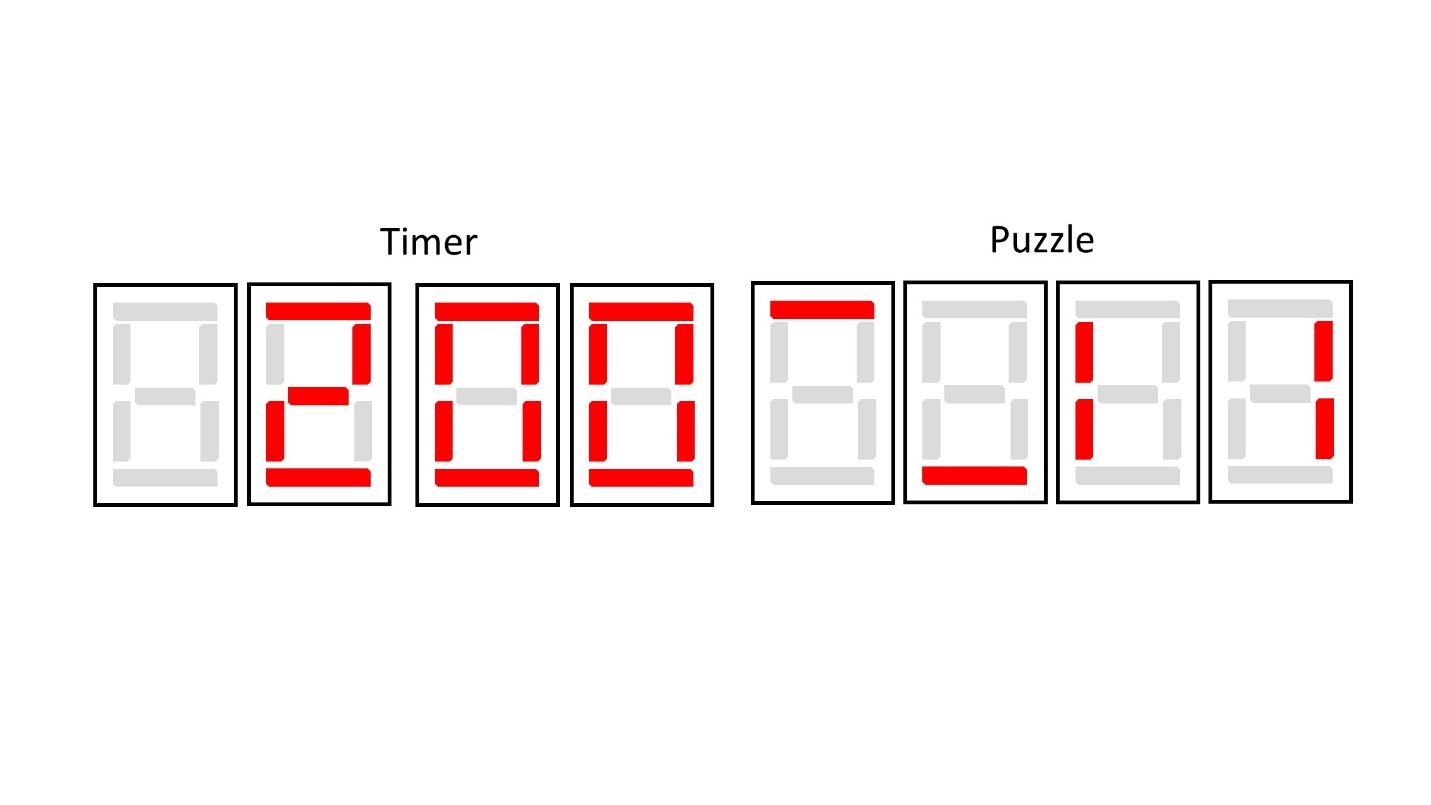


Figure 2 illustrates what the specialist may see in the bomb defusal manual. In this example, the specialist will try to give the correct translation of the sequence given to him or her by the technician. The sequence on the right is what is to be entered by the specialist on the board. Specific instructions to operate the board will be listed in the bomb defusal manual.

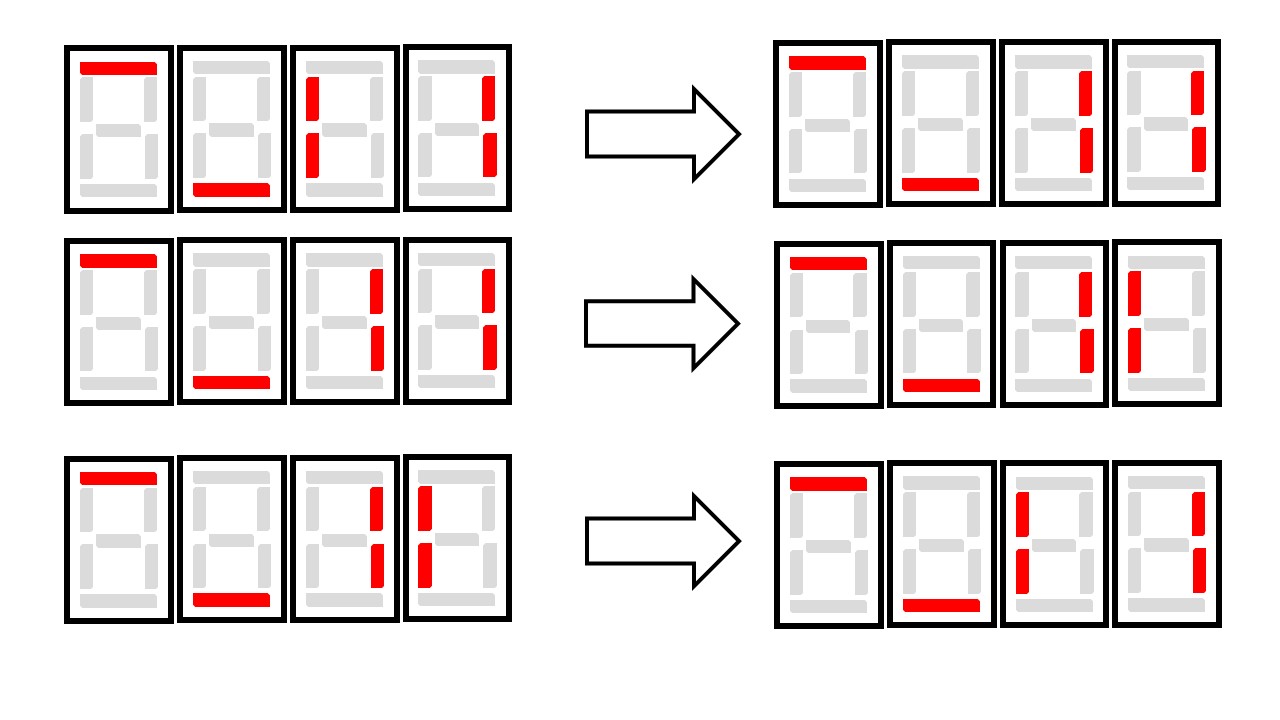


Figure 2 | Excerpt from the Bomb Defusal Manual on Patterns

On the third and final level, the user will need to solve the puzzles similarly to the previous puzzles but they will also need to remember the last segment of the puzzle they solved. Figure 3 shows what the user will need to input on the slide switches. The seven segment display on the left will be the last cell of the last puzzle, as denoted in a red border in figure 3.

Various elements of the FPGA board will be used in the implementation of this game. Figure 4 illustrates the top level architecture of the entire system. ROM will store the welcome display message that the user will see at the beginning of the game and prompt them to enter a number on the slide switches as the password to begin the game. A linear feedback shift register will pseudo-randomly generate the puzzle sequence. Upon successful completion of all the stages, the LCD will display “Bomb Defused”. The game will then end.

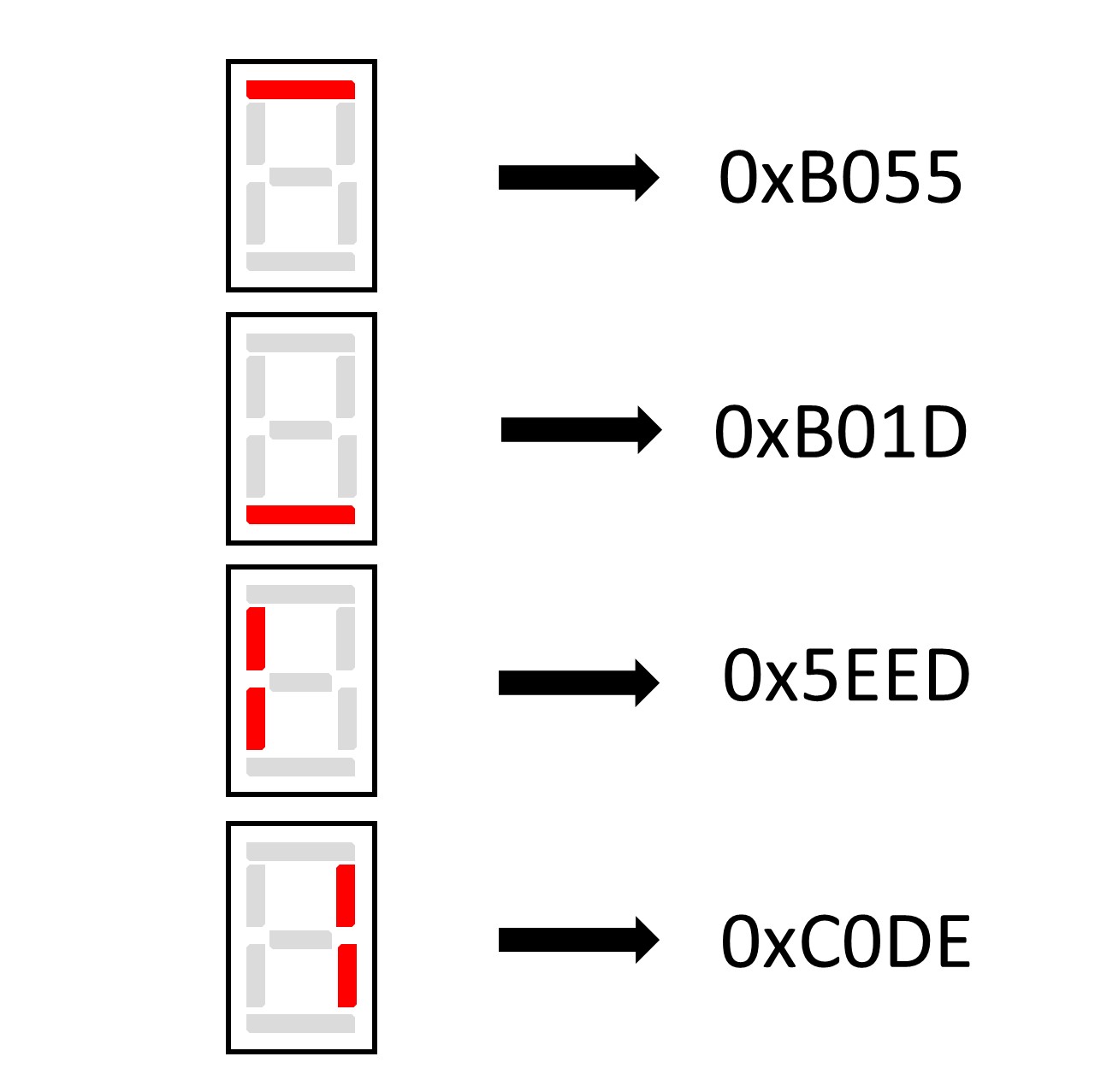
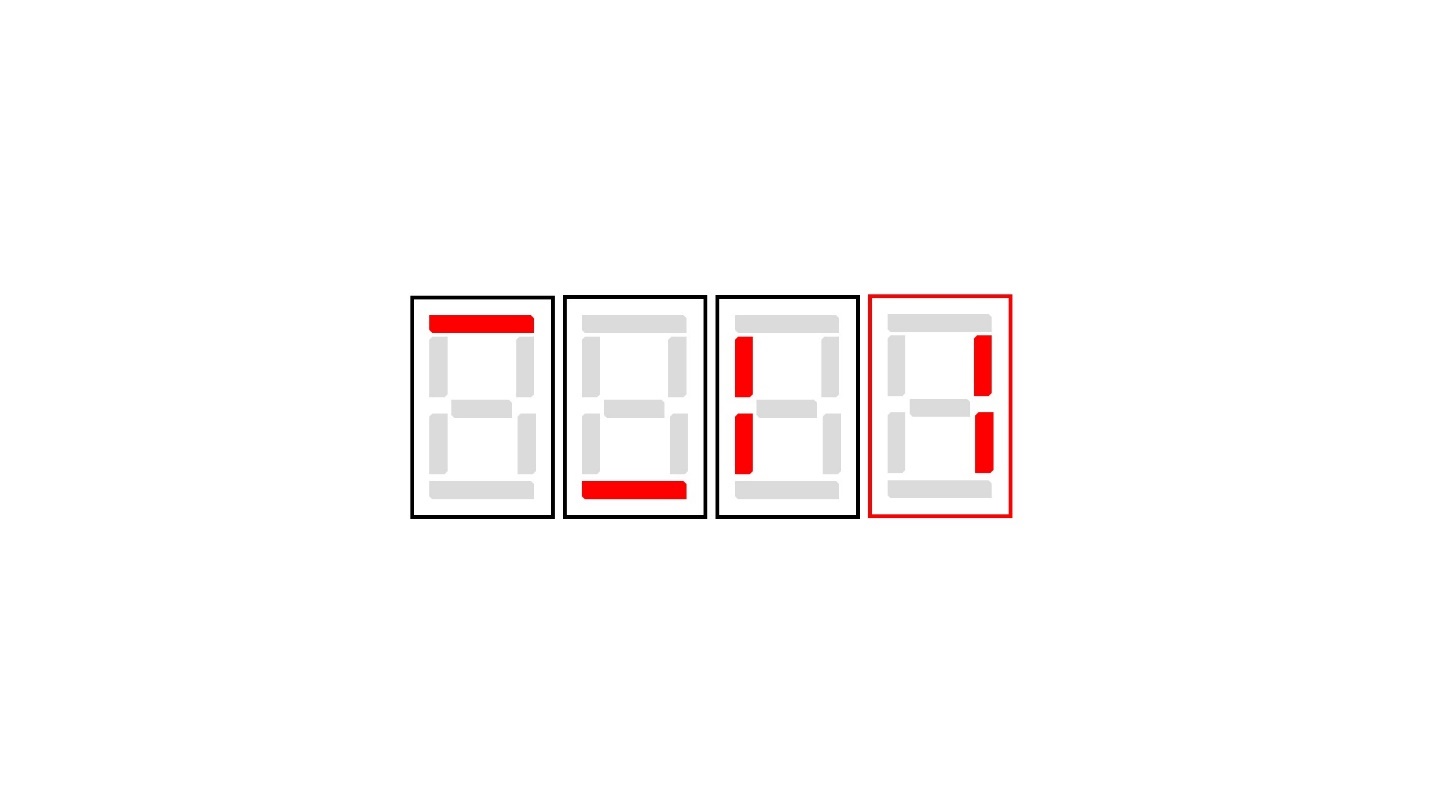


Figure 3 | Excerpt from Manual for Last Level



# Module Descriptions

## GameState

The GameState module is the game controller for the entire game; it determines the state of the game based on signals from the other modules. The first state of the game begins at the user authentication stage. The module receives a signal from the authentication module upon successful authentication. The module outputs a signal s\_current which is used as a control signal to signify the state of the game. This module will perform the logical operations to decide which module’s signals have priority over others. Game ending conditions, such as the time running out or the user entering the incorrect pattern during the user input process, have precedence over all other signals.

## Authentication

The authentication module uses a state machine design to receive an 8-bit identification number from the input switches. It authenticates that ID with credentials stored in the ROM module in the board and passes the result to the game controller. An 8-bit sequence is used to identify users for the game, the first 4 bits corresponds to the user’s ID and the following 4 bits correspond to the user’s password. When a pulse is received from the button shaper module, the authentication module will read the information stored in the first memory position of the ROM and then it will compare that information with the 8-bit sequence input. If the comparison is successful, the module will output a flag signal to the game controller indicating that a user has logged in. Another signal will be sent to the timer module to indicate the “starting game” state. If the comparison fails, the module will read the next memory position in the ROM with information stored in it, this process will continue until the user ID is verified or the module cannot find information to read. If the module cannot find more information to read in the ROM, it will output a flag signal to the game controller indicating the credential entered was incorrect. Table 1 shows an example of the possible user IDs and passwords stored in ROM. The word size is 8 bits. The last entry, “0xFF”, signifies the end of valid entries.

Table 1 | ROM Entries

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Address | +0 | +1 | +2 | +3 | +4 | +5 | +6 | +7 |
| 000 | 0xF0 | 0xE1 | 0xD1 | … | … | … | … | … |
| 008 | … | … | … | … | … | … | … | 0xFF |

## Timer

The timer module consists of a OneSecTimer module, a CountdownSequence module, and a CountdownDisplayDriver module. The OneSecTimer module is a one second timer which continuously send pulses in one second intervals. The CountdownSequence module contains a predetermined starting value in second across three 4-bit internal registers which decrements each time a pulse is sent from the OneSecTimer module. The CountdownDisplayDriver module maps a 4-bit data bus to its corresponding integer expressions on the seven segment display.

## SSD\_Sequence

The SSD\_Sequence module will first map the pseudo-random output of an sequence generator module to display the sequence that will need to be decoded. Secondly, the module will provide the user the ability to view the actions performed in response to operating the switches. The Sequence\_Generator module is a 16-bit many-to-one LFSR which continuously iterates. The output will be sampled to create the game’s sequence which the user will decode.

## SequenceVerifier

The SequenceVerifier module is the module that verifies the puzzle sequence that the user has entered. The sequence verifier will also confirm the user’s input on the slide switches during the very last sequence of the entire game. The module will receive the sequence from the sequence generator and will compare the corresponding solution to the sequence. The verify signal initiates the sequence to be verified by the sequence verifier. The result signal, which is sent to the game controller, is updated every time the user enters a segment of the pattern. If the user incorrectly enters the sequence, the result signal will change to signal the game controller to reach a game over state.

## RAM

The RAM module will store the high score for the group. In addition, the module will also store the personal best scores for each individual user. Table 2 shows an example of RAM entries. Each value represents a score. The last entry, “0xFF”, signifies the end of valid entries.

Table 2 | RAM Entries

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Address | +0 | +1 | +2 | +3 | +4 | +5 | +6 | +7 |
| 000 | 0xA | 0xB | 0xC | … | … | … | … | … |
| 008 | … | … | … | … | … | … | … | 0xFF |

## LCDDriver

The LCDDriver module will control what will be displayed on the LCD during various states of the game. The input signal is the state of the game and the output signal is the data to be displayed on the LCD. The LCD will display simple game instructions at the beginning of the game and will notify the user when the game has begun.

## LEDDriver

The LEDDriver module will control the LEDs on the board and works similarly to the LCDDriver module. The LEDs will be turned on according to the state of the game. During the “bomb exploded” game state, the LEDs will flash in a certain pattern to emulate that the “bomb” has exploded.

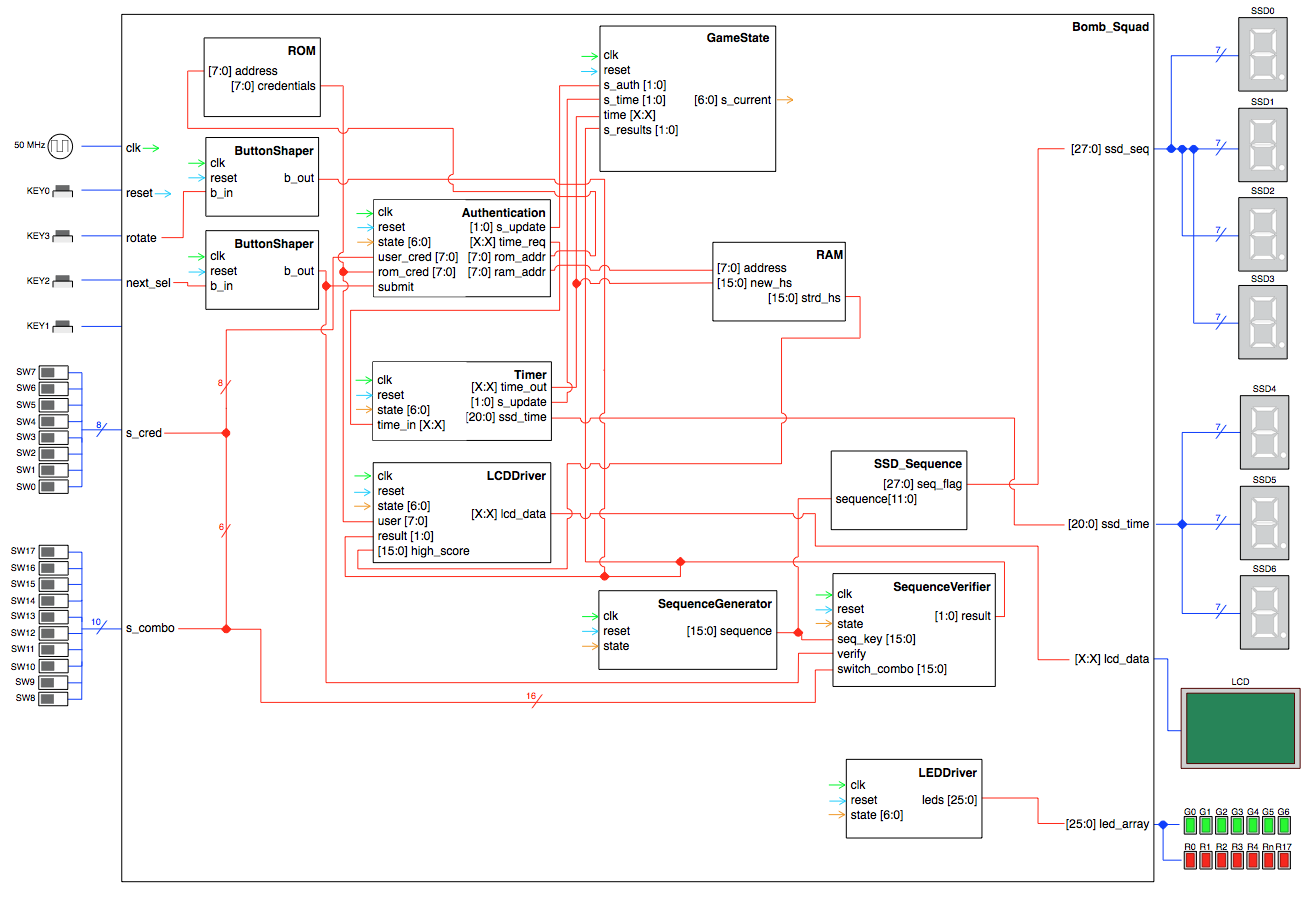


Figure 4 | Bomb Squad Architecture Diagram

Figure 5 | Bomb Squad Top Level Architecture Diagram