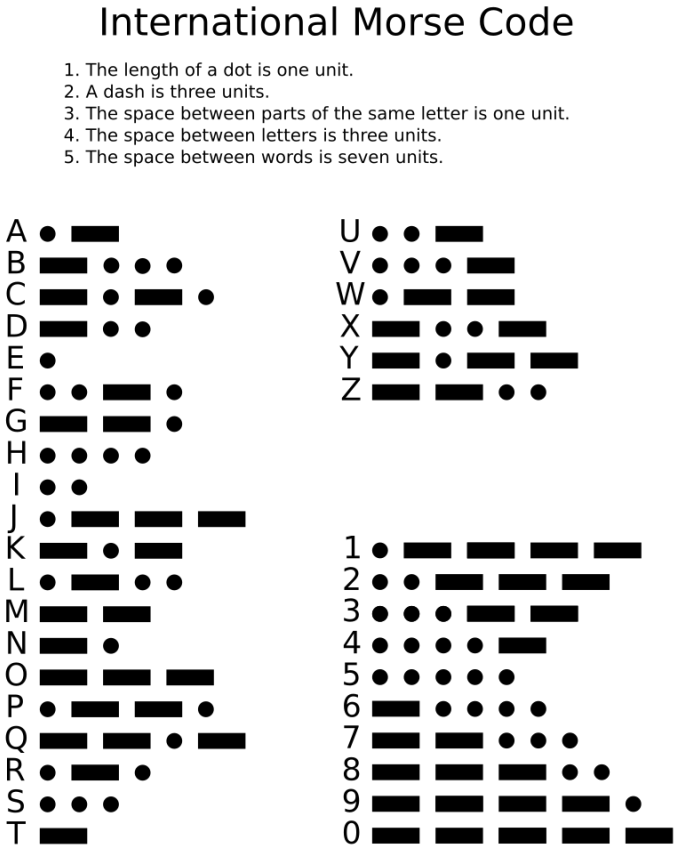
Morse Code Translator

AJ Chiaravalloti, Maria Davey, Kelvin Huang, Bryan Lam

Goal/Motivation

- Morse code takes time to learn, and message translations are prone to human error.
- A program that translates incoming transmissions could remove these potential errors.
- Our design could transmit letters faster than forcing a receiver to translate the code.
- It could remove the need for timing training (at the cost of more buttons).

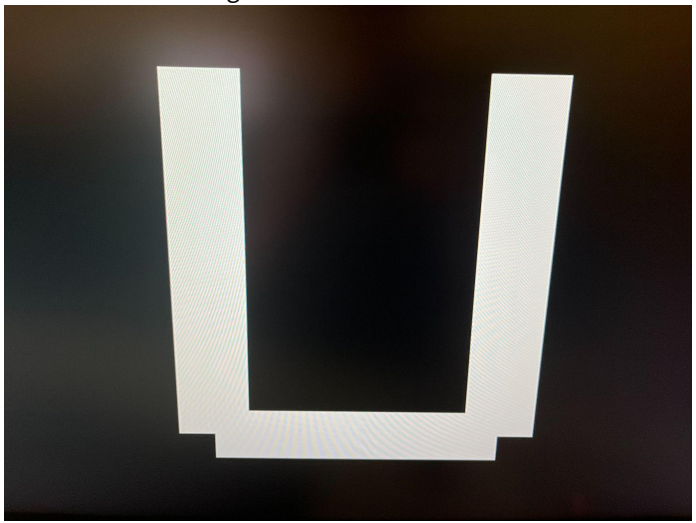


Functionality

- Takes inputs of “dots” and “dashes” and outputs the corresponding letter (via VGA).
- The user inputs the code, hits the “send” button on the FPGA, and then the corresponding letter will appear on screen.
- The user can also hit the “reset” button in case of mistakes.

Letter U’s Morse code

Our U that we designed



Specifications

Requirements

- Push sequence of buttons for various durations.
- ASCII text appear on host computer screen.
- 26 English alphabet letters (plus numbers & maybe special symbols).
- Uses VGA.

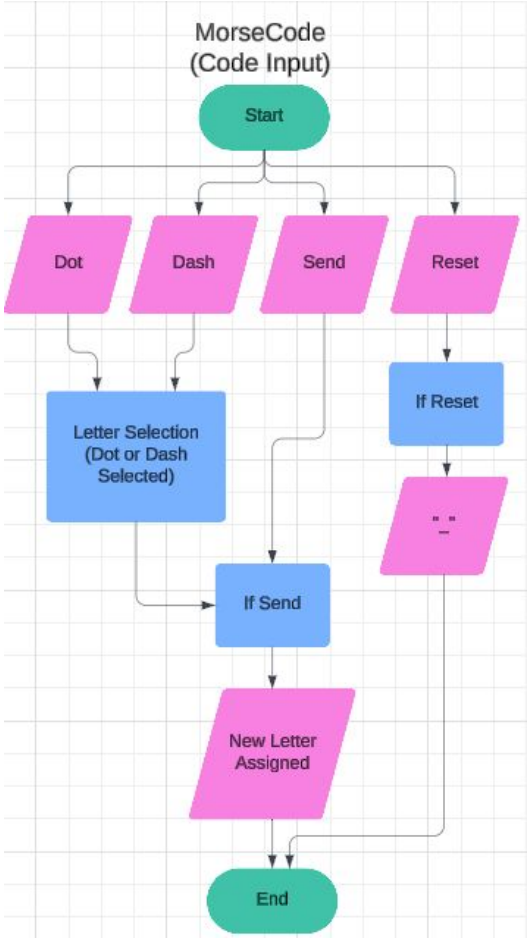
The Goal

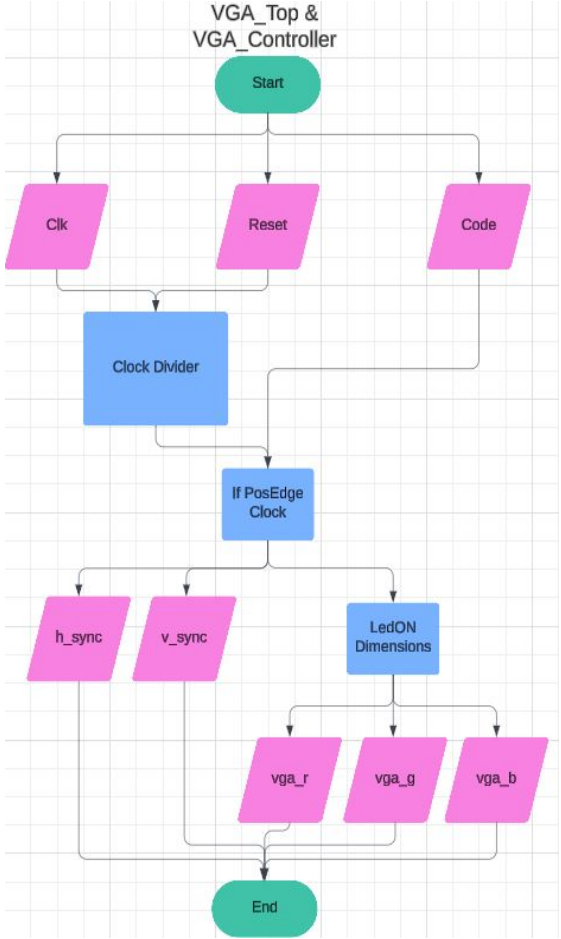
Our Specifications As of December 2nd

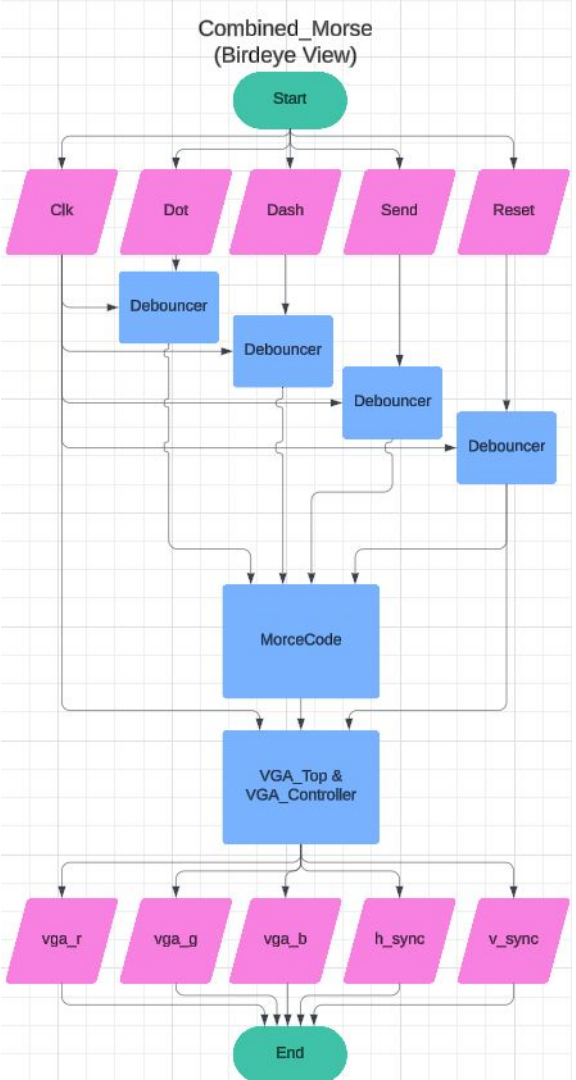
- Four buttons: dot, dash, reset, and send.
- English alphabet “exists” in code.
- CANNOT get every letter
- Text is NOT typed out (letters appear/disappear on screen).
- Uses VGA

Currently

Detailed Block Diagrams (Made with Lucidchart)





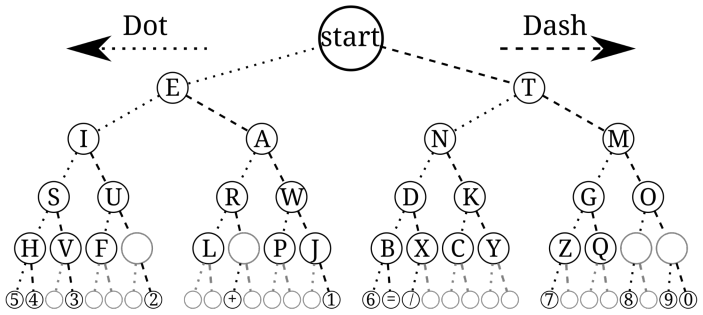


Code Snippets

Implementation of State Transitioning:Determining ASCII Characters

Showcased Example: Case statements on input of “dash”

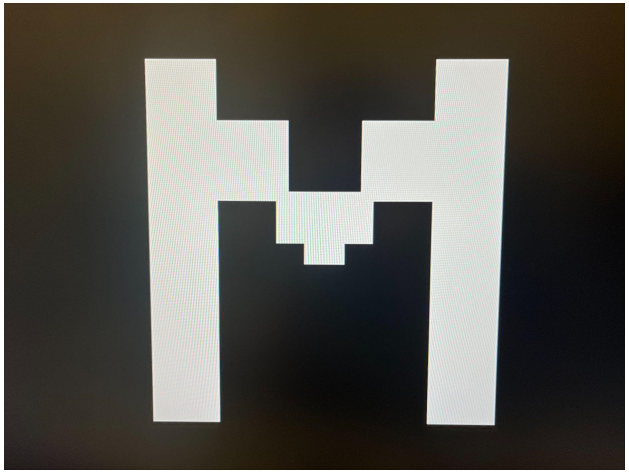
- Challenges:
  - Ensuring that the proper letter is defined on any combination of user input
  - ->Need to make sure that the transitions between letter-states are correct
- Solution:
  - Apply current_letter to “store” the letter before sending
  - On “send” input- assign output register “code” to the last state of letter




```
always @ (posedge reset or posedge send or posedge dot or posedge dash) begin
    if (reset) begin
        code <= 5'b11111;
        current_letter = _;
        letter=current_letter;

    end
    else if (send) begin

        code <= letter;


    end

    else if (dash) begin
```

```
else if (dash) begin
    case(current_letter)
        _:current_letter=T;
        T:current_letter=M;
        M:current_letter=O;
        O:current_letter=_;

        G:current_letter=Q;
        Q:current_letter=_;

        N:current_letter=K;
        K:current_letter=Y;
        Y:current_letter=_;

        D:current_letter=X;
        X:current_letter=_;

        E:current_letter=A;
        A:current_letter=W;
        W:current_letter=J;
        J:current_letter=_;

        I:current_letter=U;
        U:current_letter=_;

        S:current_letter=V;
        V:current_letter=_;

        H:current_letter=_;
        F:current_letter=_;
        B:current_letter=_;
        R:current_letter=_;
        L:current_letter=_;
        C:current_letter=_;
        P:current_letter=_;
        Z:current_letter=_;

    endcase

    letter = current_letter;

    end
```

Code Snippets



Implementation of VGA for the display of letters:

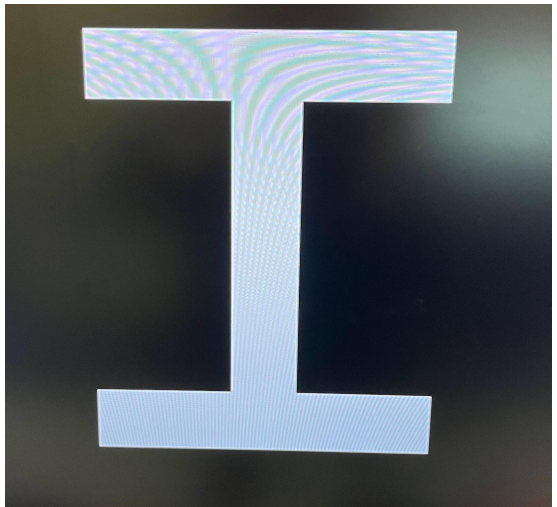
Showcased Example: VGA Output of Letter “M”

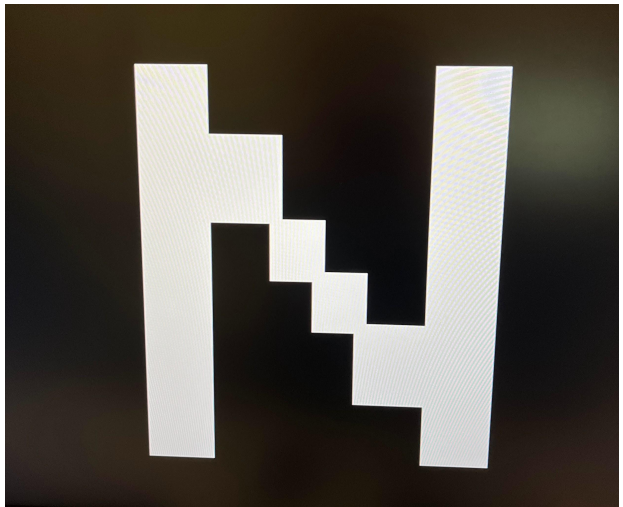
```
5'b01100:begin
    // Letter "M"
    inBoundsa = ((widthPos > (290)) & (widthPos < (350)) & (heightPos < (245)) & (heightPos > (170))) ? 1'b1:1'b0; // Lowermost Horizontal Line
    inBoundsb = ((widthPos > (250)) & (widthPos < (300)) & (heightPos < (460)) & (heightPos > (115))) ? 1'b1:1'b0; // Leftmost Vertical Line
    inBoundsc = ((widthPos > (450)) & (widthPos < (500)) & (heightPos < (460)) & (heightPos > (115))) ? 1'b1:1'b0; // Rightmost (Lower) Vertical Line
    inBoundsd = ((widthPos > (400)) & (widthPos < (460)) & (heightPos < (245)) & (heightPos > (170))) ? 1'b1:1'b0; // Uppermost Horizontal Line
    inBoundse = ((widthPos > (360)) & (widthPos < (390)) & (heightPos < (305)) & (heightPos > (275))) ? 1'b1:1'b0; // Rightmost (Upper) Vertical Line
    inBoundsf = ((widthPos > (340)) & (widthPos < (410)) & (heightPos < (285)) & (heightPos > (235))) ? 1'b1:1'b0; // Middle Horizontal Line
    end
```

- Challenges:
  - Creation of 26 unique letters and proper selection based on input
- Solution:
  - Implementation of a 5 bit register to a create display of chosen letter

Success (As of December 2nd)

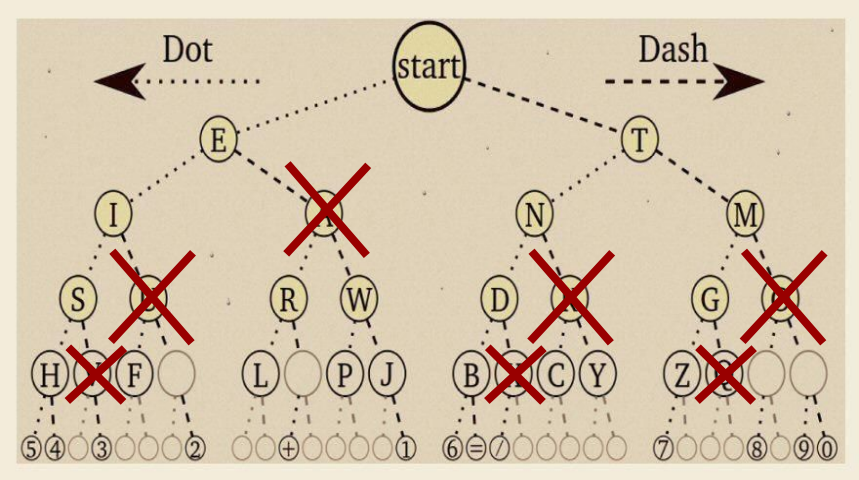
- Created 26 cases for vga display of each letter
- Can translate certain button input combinations to corresponding letters,
  - Example:dot-dot-send
    - Correctly outputs the letter I
- Possible to get other letters with incorrect combination,
  - Example: dot-dash-dot-send
    - Outputs the letter N, should be: dash-dot-send.





Failures (As of December 2nd)

- Design display characters momentarily.
- Cases are limited to the 26 English alphabet.
- Letters we have can be a little hard to read.
- Strange behaviors with “dash”
  - We must initialize a dot before a dash.
  - Example: Path to M
    - Expected: Dash Dash
    - Currently: Dot Dash Dot Dash



References (Where We Got Our Pictures)

https://web.archive.org/web/20121106142719/http://www.itu.int/rec/R-REC-M.1677-1-200910-I/

https://militarymachine.com/u-morse-code/

https://www.forbes.com/sites/jimgorzelany/2019/07/23/here-are-the-coolest-new-cars-for-2020/

https://stock.adobe.com/search?k=old+broken+down+car&asset_id=7894582

https://commons.wikimedia.org/wiki/File:Morse-code-tree.svg