DIGITAL DESIGN

DESIGN ASSIGNMENT

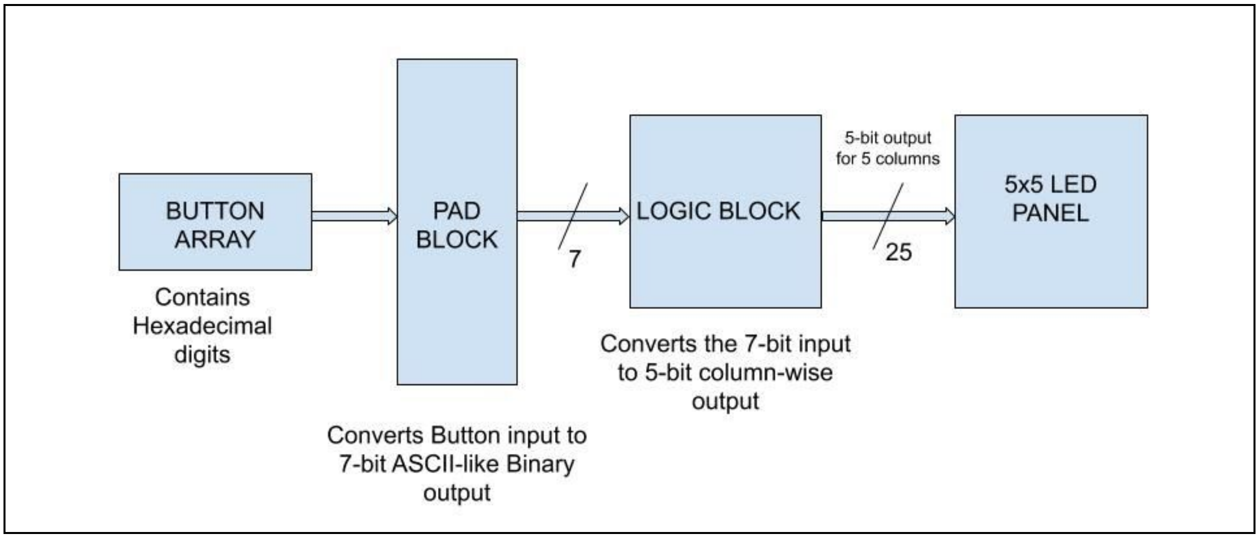
GROUP 50

Design a system that works like a dot-matrix printer. The user inputs a hexadecimal number, the output is a 5x5 LED bank that
gets "written" from left to right.

REPORT SUBMITTED BY

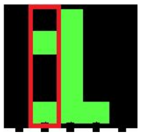
SAHAJ JAIN	2019AAPS0057G
KAUSTUBH LAD	2019AAPS0187G
ARUNAABH SINGH	2019A8PS0664G
JASKARAN CHHABRA	2019A8PS0653G
PAREEKSHITH N ACHAR	2019A8PS1101G
RAGHAV DHIR	2019AAPS0113G

TOP LEVEL BLOCK DIAGRAM


BUTTON ARRAY
Contains Hexadecimal digits
PAD BLOCK
Converts Button input to 7-bit ASCII-like Binary output
7
LOGIC BLOCK
Converts the 7-bit input to 5-bit column-wise output
5-bit output for 5 columns
25
5x5 LED PANEL

DESIGN AND EXPLANATION

- The button array contains the Hexadecimal digits (0-F) with a *clear* function and forms the input to the circuit. The output is represented in the form of a 5x5 LED Panel designed to take inputs column-wise.
- The *PAD*, with the help of priority encoders, generates a 7-bit binary signature inspired by the ASCII values of the digits and stores it in a register. It also outputs a '*change*' signal which goes high when a button is pressed and resets any previous display.
- This 7-bit binary output along with the '*change*' signal goes into the *LOGIC.* Every *column block* in the *LOGIC* has two decoders (one each for a letter and a number and enabled accordingly) which form the basic logic of the output using **minterms**. This logic is common across all 5 columns. Each *column block* generates a 5-bit binary output corresponding to the LED Panel column. As an example, if the '*button 1'* is pressed, the 7-bit binary signature will be '001**0001**' and the 5-bit output of *COLUMN 2* will be '01001'. This pattern can be seen in the accompanying figure if we move from top to bottom in *COLUMN 2*.



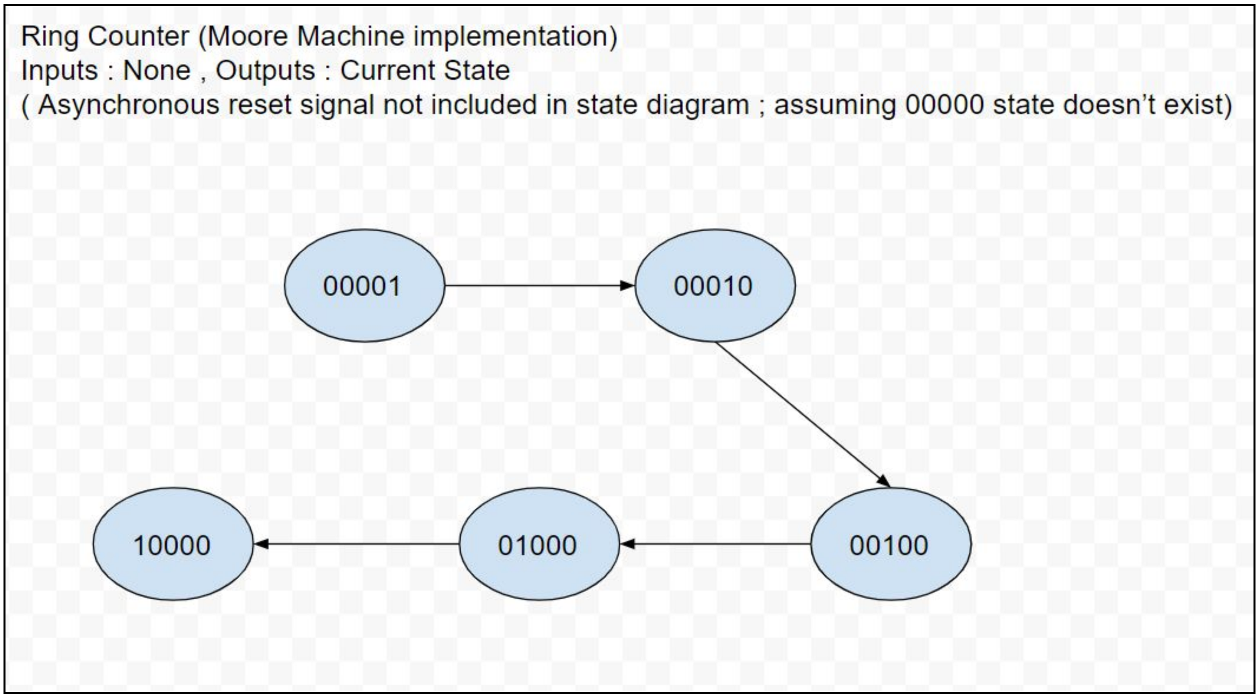
- This 5-bit output is fed into a 2:1 MUX. If the input is a valid Hexadecimal digit then this 5-bit output is passed through. Otherwise, a premade symbol '-' is designated for erroneous input.
- The multiplexer output goes into a 5-bit register that stores the value for its corresponding column. Each register is triggered using a custom *Ring Counter* which stops after 5 ticks of the *clock* and the output appears in a *left to right* fashion. The *change* signal is used here to asynchronously reset all registers and the *Ring Counter.*
- Finally, all the 5-bit output from the registers form one 25-bit output which is displayed through the 5x5 LED Panel

ASSUMPTIONS IN DESIGN

- An underlying assumption made in the circuit is that the output must be ready within one clock cycle. This means that we have **neglected the effect of propagation delay**. However, we have made several intelligent design choices to make the circuit more efficient and less prone to the effect of propagation delay and human input errors.
- In case of the ring counter, the state 00000 may hypothetically exist, however, for normal working, the CHANGE signal resets counter to 10000 before every transition and hence **00000 state never occurs in working conditions**.
- **Resistances, Inductance and Capacitance effects of all connection components and any external field effects are assumed to be negligible** and hence do not affect the normal functioning of IC's.
- In case of added functionality, the **user must not operate logisim keyboard and adder mode simultaneously**.

STATE DIAGRAM

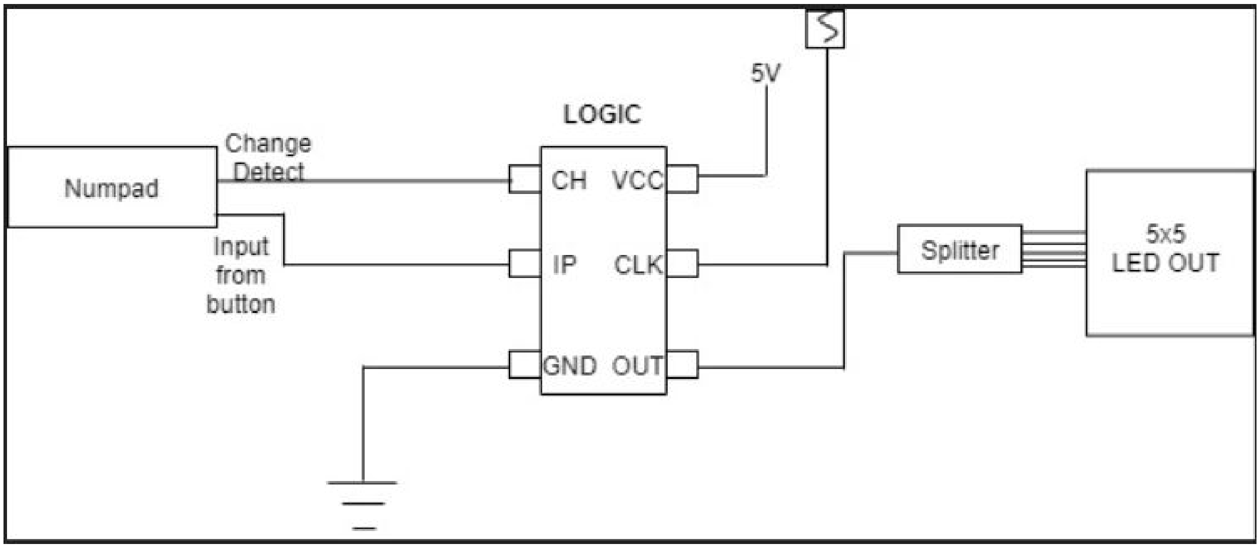
RING COUNTER (Within *LOGIC BLOCK*)



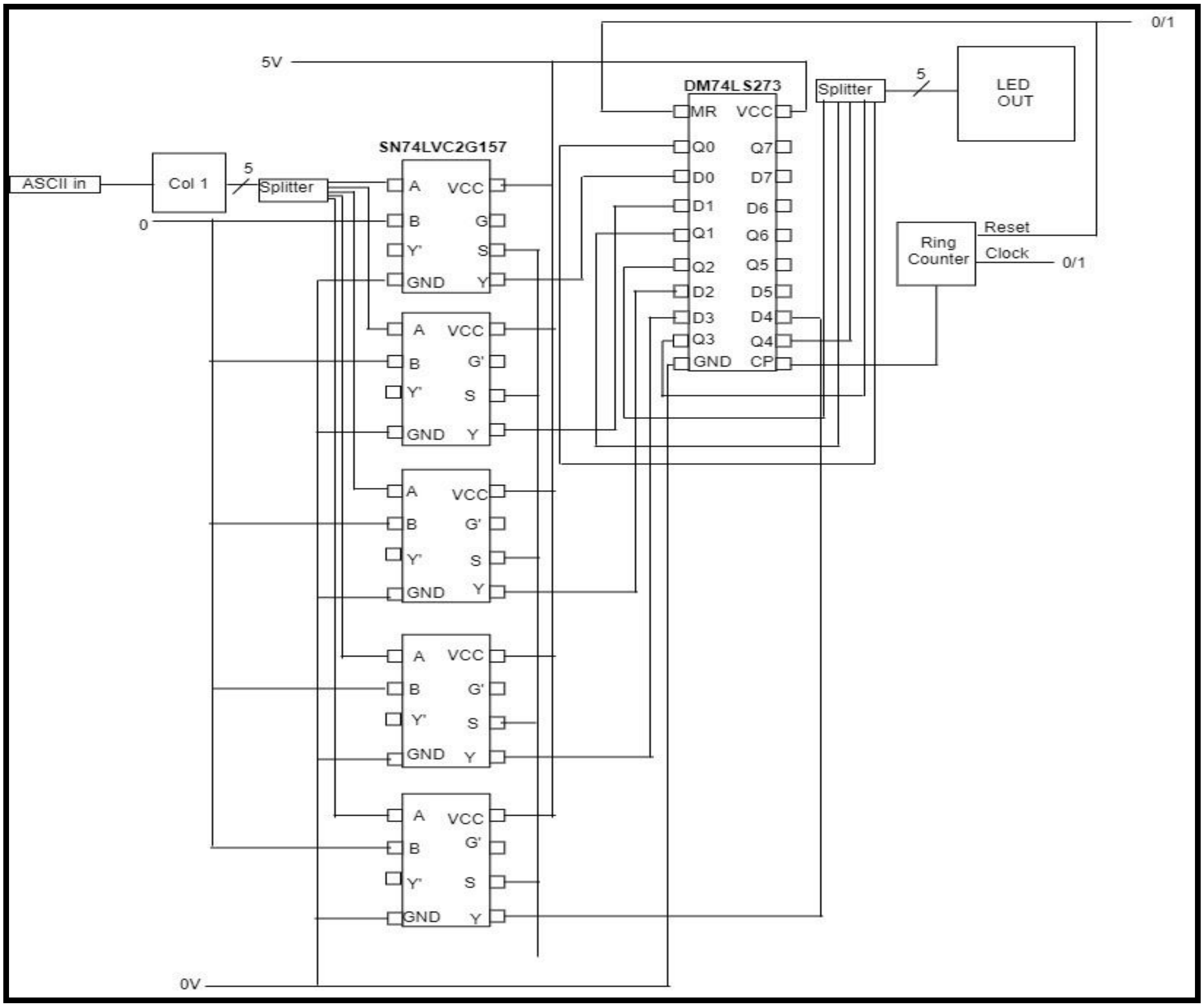
Ring Counter (Moore Machine implementation)
Inputs : None , Outputs : Current State
(Asynchronous reset signal not included in state diagram ; assuming 00000 state doesn't exist)

PINOUT DIAGRAMS

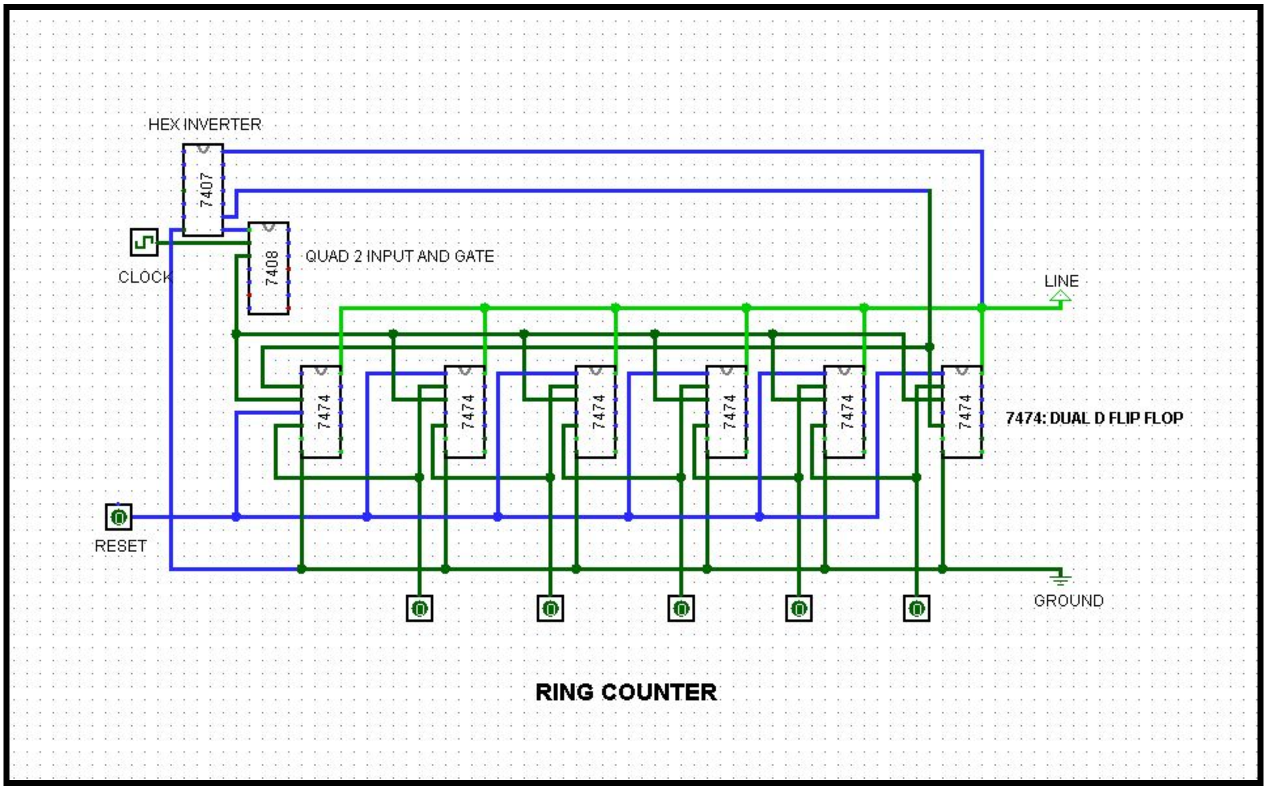
1. MAIN CIRCUIT



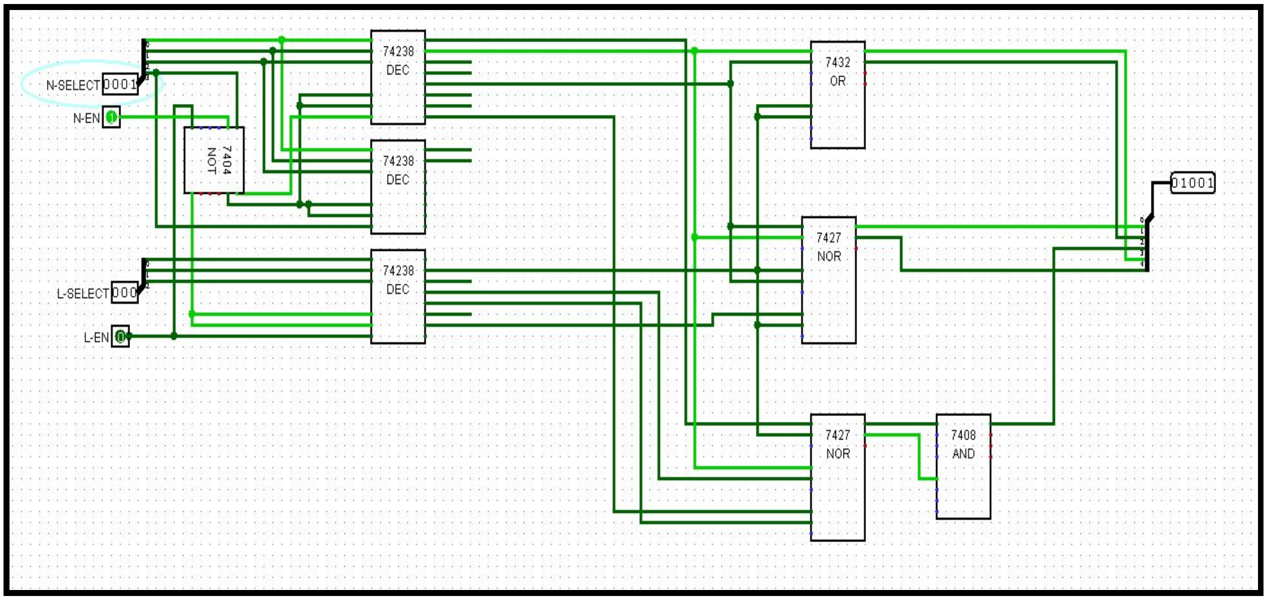
2. LOGIC BLOCK (Column 1 taken for reference)



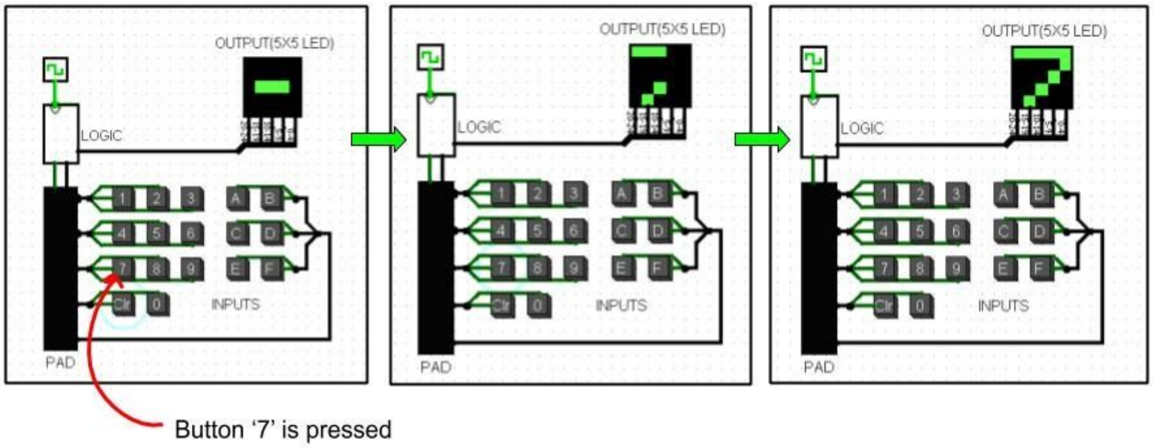
3. RING COUNTER



4. COLUMN PINOUT DIAGRAM (Column 2 taken for reference)



SAMPLE INPUT/OUTPUT

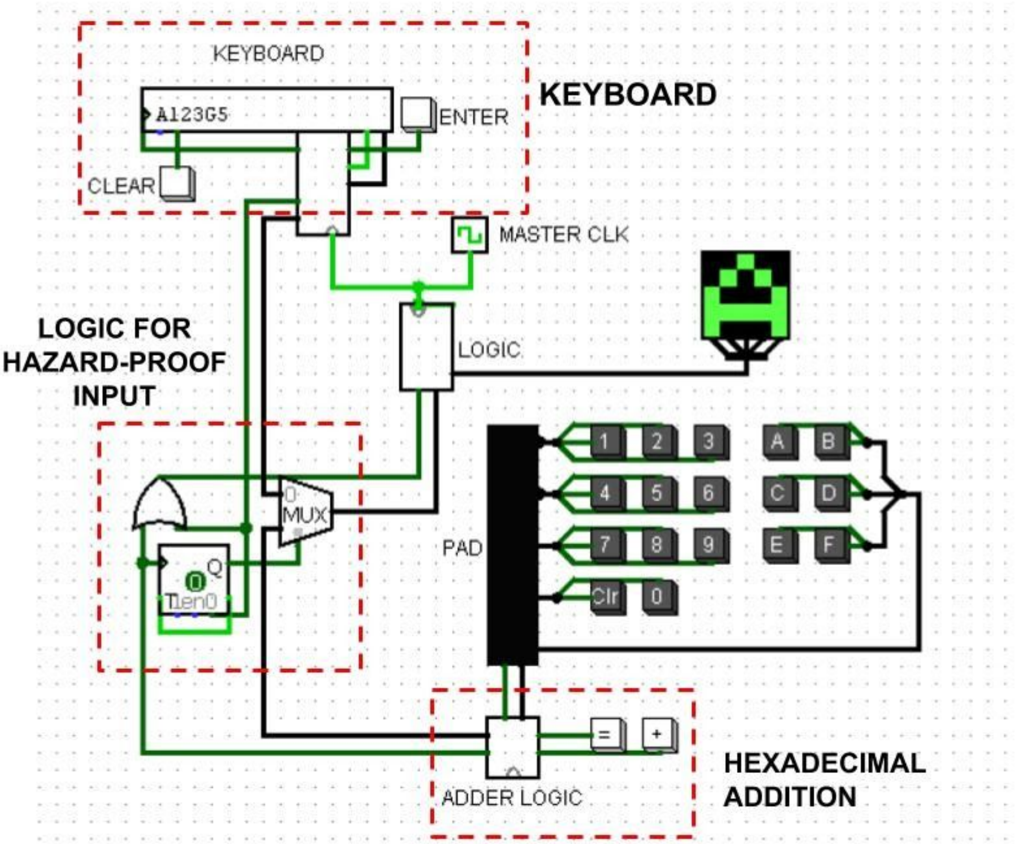


Button ‘7’ is pressed

ADDITIONAL FUNCTIONALITIES

- **KEYBOARD**
  A major functionality implemented in the circuit is the use of the Logisim in-built keyboard to take the input. This keyboard can have multiple characters in its buffer at an instance. It inputs characters directly from the external keyboard device and can cycle through them one at a time when *ENTER* is pressed. The keyboard outputs a 7-bit ASCII binary code corresponding to the character taken out of its buffer which can be directly used in the circuit (as it was already based on 7-bit ASCII-like binary signatures).
  Additional logic with T Flip-Flop is used as a memory to the select bit of MUX to choose between the two modes of inputs (keyboard and button array).
- **HEXADECIMAL ADDITION**
  Owing to the button array design, we have implemented an “addition” function which adds two hexadecimal numbers and displays their Hexadecimal sum on the LED Panel. It is again based on the 7-bit ASCII-like binary signatures and is easy to use just like a conventional calculator.
- **ERROR-PROOF DESIGN**
  Our design is completely error and hazard-proof to user error. The user can enter any character (through the keyboard) and at any instance of the working circuit. If the character is not a Hexadecimal number, the display would simply output ‘-’. If the user enters another character while the first character is being displayed from *left to right* then the display would asynchronously respond to it.

CIRCUIT FOR ADDITIONAL FUNCTIONALITIES



LIST OF COMPONENTS AND APPENDIX

Name of the component	Quantity	datasheet
7474 Dual D flip flop	**3**	https://drive.google.com/file/d/154X6p3K0EWvLg9Buhkc0tK9hmhSofMXW/view?usp=sharing
7408 Quad 2 input AND gate	**2**	https://drive.google.com/file/d/1GtgMA69OHorIR8_KcWH1Zn935LhMSYxe/view?usp=sharing
7404 NOT gate	**6**	https://drive.google.com/file/d/1y-oaJpBPLDQCh9GvBcgGGQZroU59ncC8/view?usp=sharing
74238 Decoder	**10**	https://drive.google.com/file/d/1AK_eyYPh3N6yQM_LzvZI4t5lyQ9u6Mhf/view?usp=sharing
7432 OR gate	**15**	https://drive.google.com/file/d/1JR4r_Ury6DEJoWdmEI7iuZeyhaws2h12/view?usp=sharing
7427 triple NOR gate	**11**	https://drive.google.com/file/d/18eohnl6JGMVZ2CM8qsjLWdRP0kCXg1MK/view?usp=sharing
DM74LS273 :8 Bit register with Clear	**5**	https://drive.google.com/file/d/1s_ZJ-sfDBAe7gNcpeCMILysif0VJpyuc/view?usp=sharing
SN74LVC2G157:single 2 line to 1 line data selector multiplexer	**25**	https://drive.google.com/file/d/1zwcYrWMM13jbLNi36UQgs-ogTRDFI_4k/view?usp=sharing
74HC147: 10:4 Priority Encoder (Used as 16:4 Priority encoder)	**2**	https://drive.google.com/file/d/1wIQ9WKWGkYIOIUqXh83uDc5GRyqryXtM/view?usp=sharing