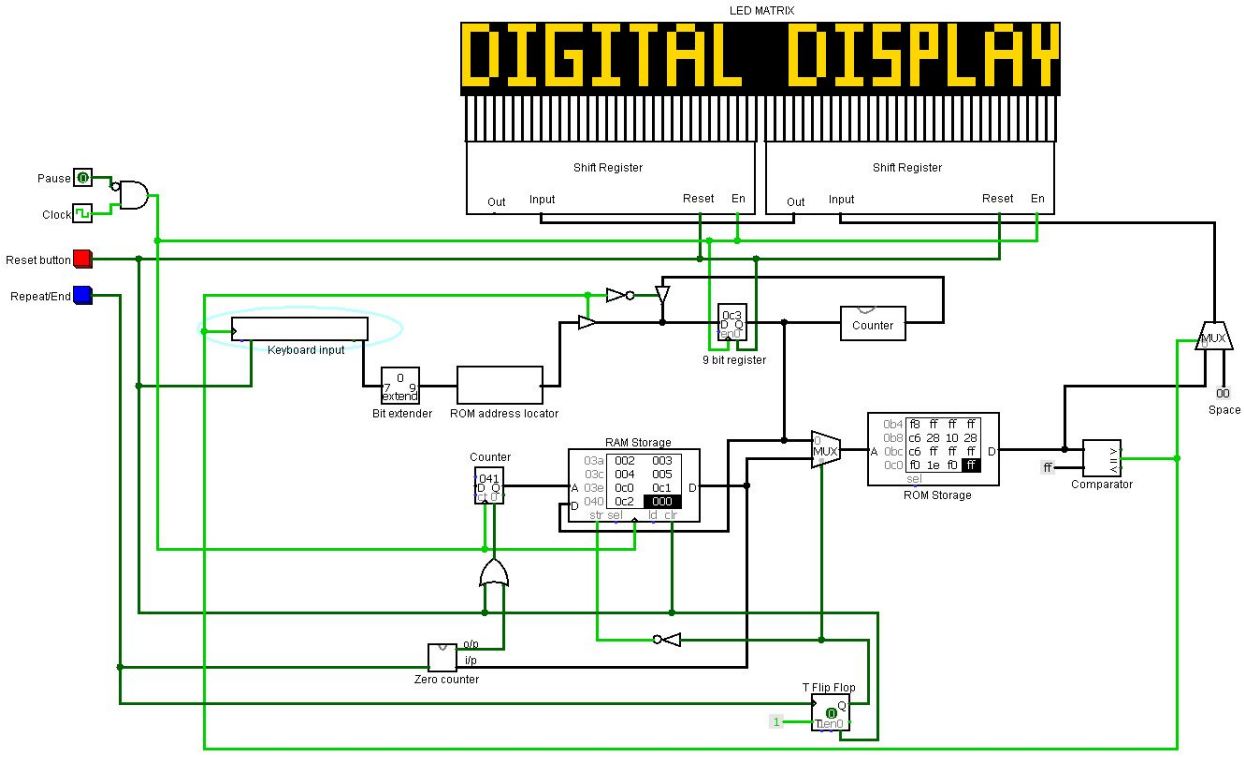
EEE F215(Digital Design)

Design Assignment

Group 59

Problem Statement

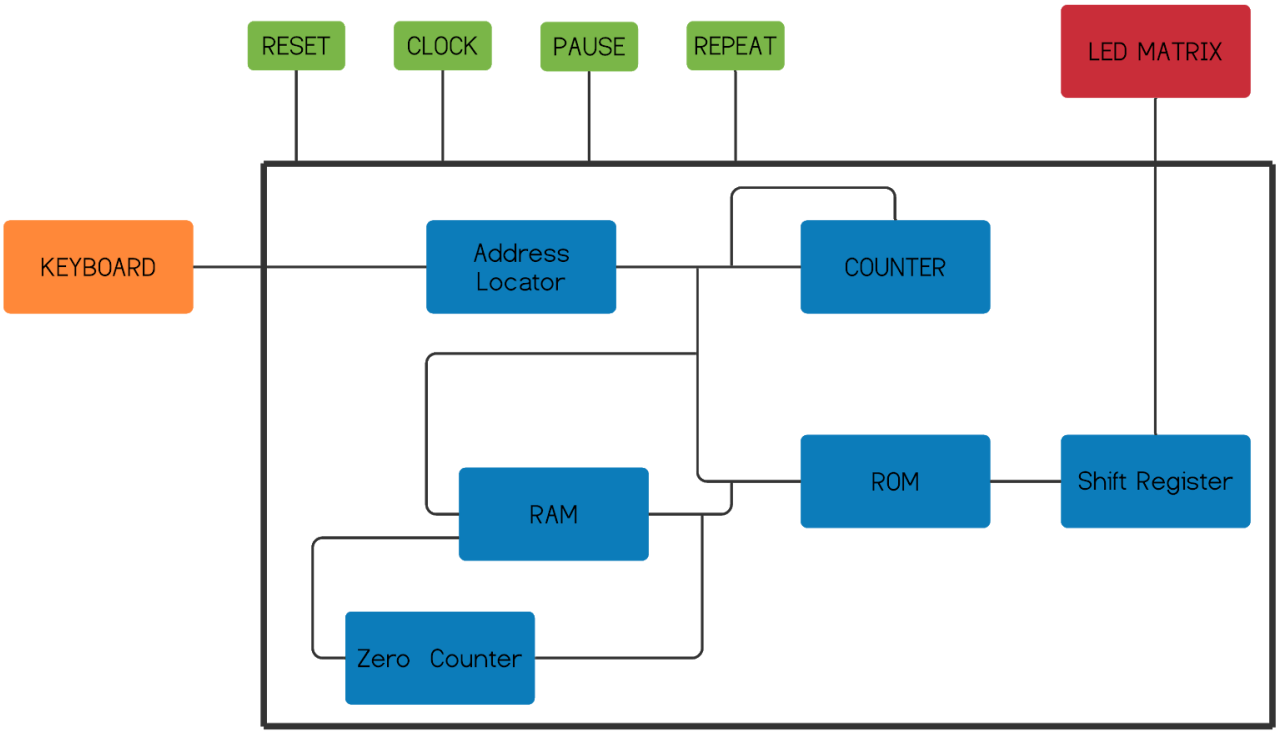
Design a scrolling display for text. The text can have a length of a minimum of 6 alphanumeric characters. Use a display such that at least six characters are displayed at a time.



Group Members

Saurabh Santosh Gandhi	2019A3PS0479G
Mrudul Anil Dwivedi	2019A3PS0440G
Shreyas Somvanshi	2019A3PS0434G
Aarohi Ramdas Khodkumbhe	2019A3PS0451G
Ashita Bhardwaj	2019A3PS0457G
Archit Rungta	2019A3PS0450G

Top-Level Block Diagram



INPUTS:

1. **Input from the Keyboard:** Inputs the 7-Bit ASCII code for each character.
2. **Initiate Button:** Goes momentarily high to initiate repetitive scrolling.
3. **Pause:** Freeze the output display.
4. **Clock:** The design consists of sequential components. The clock decides the frequency of output display.
5. **Reset:** To clear volatile data and reset the circuit to the initial condition.

OUTPUTS:

1. **Shift Register:** The shift register slides the data at every clock pulse to create a scrolling output which is provided to the LED Matrix.
2. **LED MATRIX:** A matrix of 32x8 LEDs which displays the sequence of characters to produce a scrolling display.

Assumptions

- The keyboard outputs the ASCII values of the entered characters as a 7-bit binary number.
- Alphabets are to be entered into the keyboard in lower-case. Otherwise, they will be regarded as invalid.
- If the user wants to enable recursive scrolling the input must be typed into the keyboard before the clock is simulated.
- If recursive scrolling is enabled the number of characters entered into the keyboard is less than 125(approximately).

Keybindings

Input on Keyboard	Output on Screen	Input on Keyboard	Output on Screen
a	A	w	W
b	B	x	X
c	C	y	Y
d	D	z	Z
e	E	1	1
f	F	2	2
g	G	3	3
h	H	4	4
i	I	5	5
j	J	6	6
k	K	7	7
l	L	8	8
m	M	9	9
n	N	Spacebar	Space
o	O	.	.
p	P	,	,
q	Q	?	?
r	R	!	!
s	S	*	Pacman emoji
t	T	\	Smiley emoji
u	U	/	Arrow
v	V		

Design Approach and Methodology

The requirement of the given design problem is to input alphanumeric characteristics and display the resulting text on a scrolling display.

Keyboard- It takes input from the user. The input is regarded valid if it can be recognized as a lower-case alphabet [a-z], a number[0-9] or any of the following special characters -'?', '!', '.', ',. The characters '/', '\', '*' represent Arrow, Smiley and Pacman respectively. Each input character is a 7 bit long binary number in ASCII format and is passed to the bit extender in parallel.

Bit Extender- It extends the 7-bit binary value of ASCII code to a 9-bit binary number by appending two binary 0's at the MSB end of the 7-bit number.

ROM Address Locator- It inputs the 9-bit binary number from the bit extender and compares the number with the ASCII values of all the characters. Once a value is matched, it passes the corresponding ROM address to the register.

9-bit register- It stores the value of the address obtained at each clock pulse.

ROM- It inputs a 9-bit address from the register and outputs an 8-bit binary number corresponding to that address at each clock pulse.

Counter- This takes the input as the 9-bit address stored in register and adds 1 to the address so that we get the next ROM address at every clock pulse till the ff arrives. (ff arrives when there is no input to the keyboard and when the 1st character is completed. It serves as the signal to the keyboard and the keyboard is enabled for the next character to be entered by the user).

Comparator- Output of the ROM goes to the comparator which compares the whether it is ff or not and if it's not ff then it will pass through MUX and then to the input of the shift register. If the output is ff then the MUX will pass 00(space) as the input to the 32-bit shift register. The output of this 32-bit shift register is passed to the next 32 bit shift register as an input to continue the scrolling display.

Shift registers- There are 2 shift registers designed using 32 registers in each. The 1st register in the series of 32 registers inside the 1st shift register receives 8 bit input from the MUX. This input is transferred to the next register after every clock pulse. 32 lines of each shift register are connected to the LED matrix. They transfer the data in 32 shift registers to the LED matrix at every clock cycle.
Working of the 2nd shift register- Design of 2nd shift register is the same as that of the 1st. Output of the first shift register is connected to the input of the 2nd shift register. Same process repeats for this shift register too which continues the scrolling display.

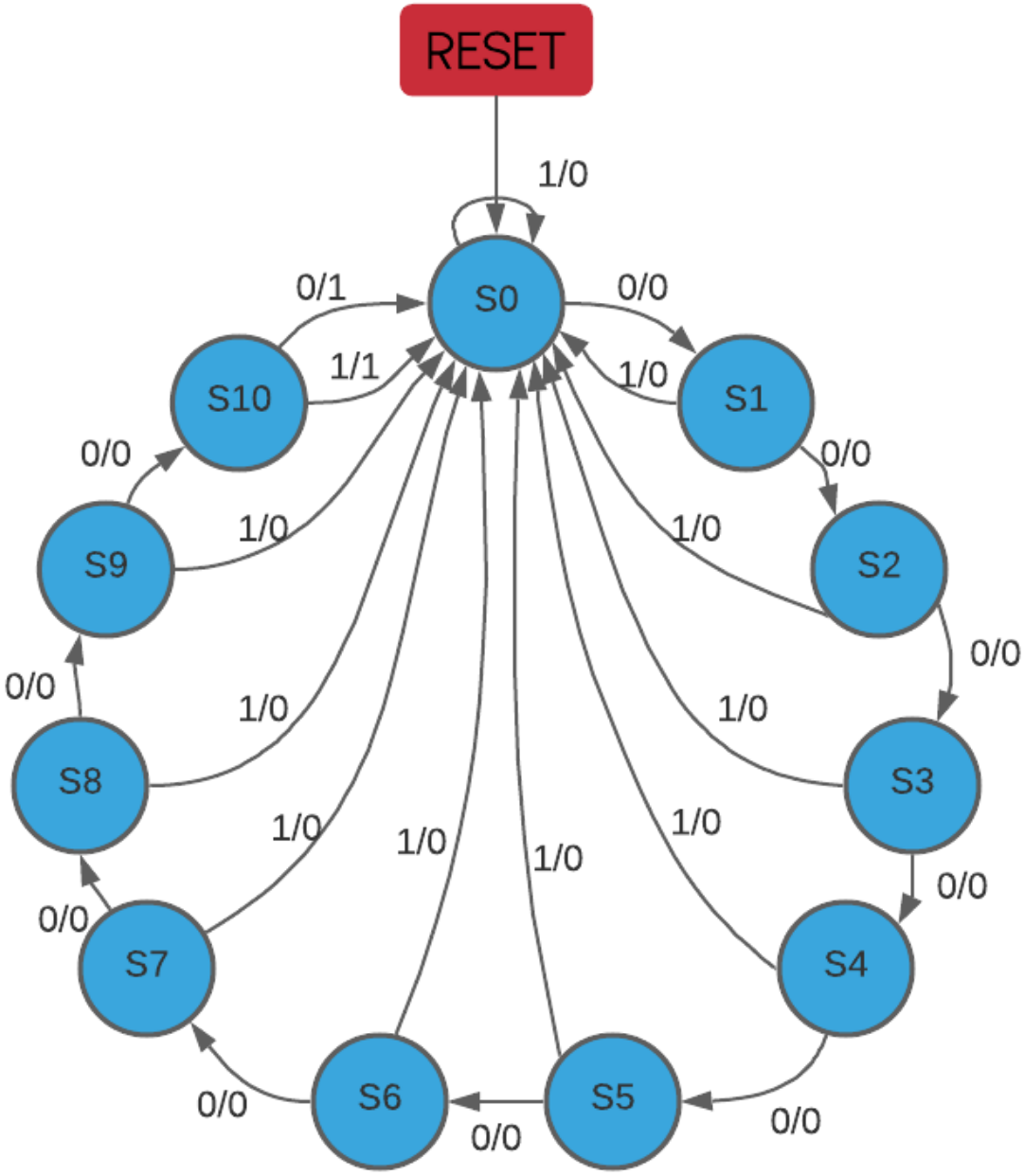
Zero counter- It functions to reset the counter used for iterating through the RAM memory locations. As soon as it encounters 10 consecutive spaces it resets the counter connected to the RAM input pin.

LED Matrix- Two 32-bit shift registers are connected to LED matrix which produce a scrolling display that re-renders at each clock cycle.

State Diagram (Moore Implementation)

The Zero Counter:

The Zero counter adds 10 spaces before the scrolling repeats. It counts the number of times zero is encountered in the RAM.

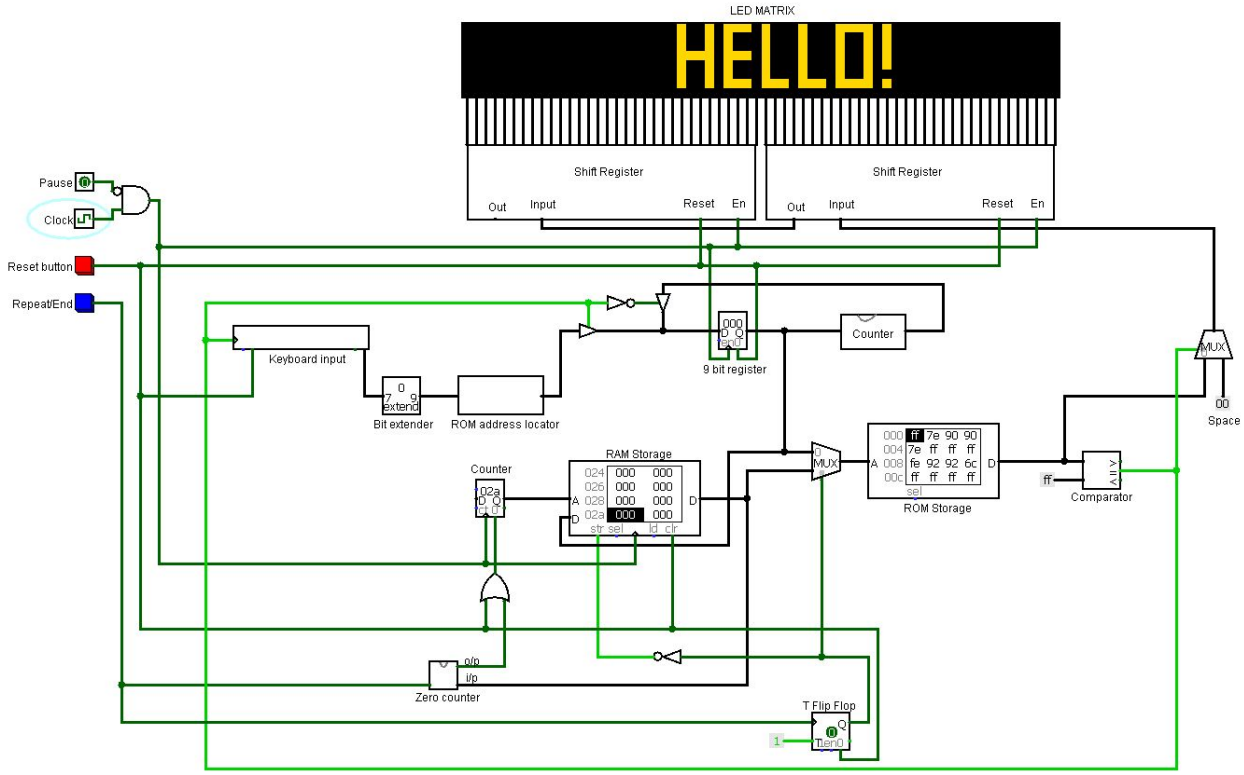


State Table

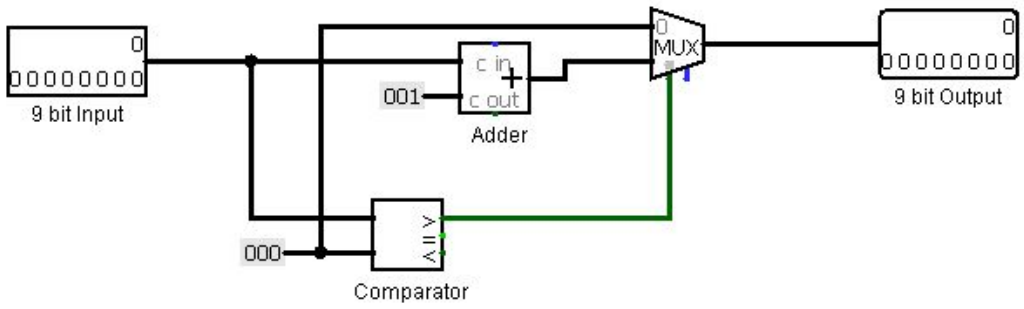
Previous State	Next State		Output	
	x=0	x=1	x=0	x=1
S0	S1	S0	0	0
S1	S2	S0	0	0
S2	S3	S0	0	0
S3	S4	S0	0	0
S4	S5	S0	0	0
S5	S6	S0	0	0
S6	S7	S0	0	0
S7	S8	S0	0	0
S8	S9	S0	0	0
S9	S10	S0	0	0
S10	S0	S0	1	1

Logisim implementation

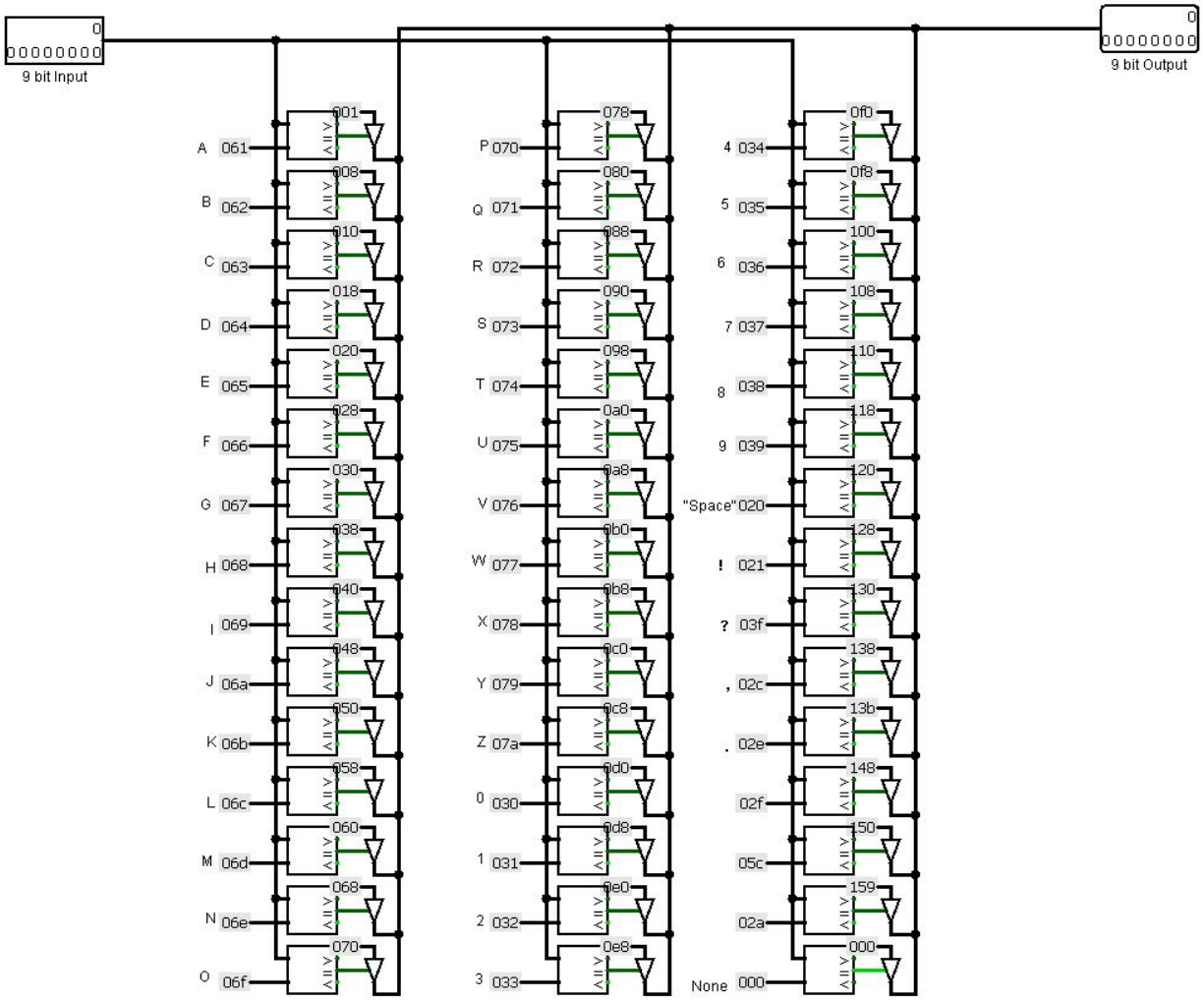
Main circuit



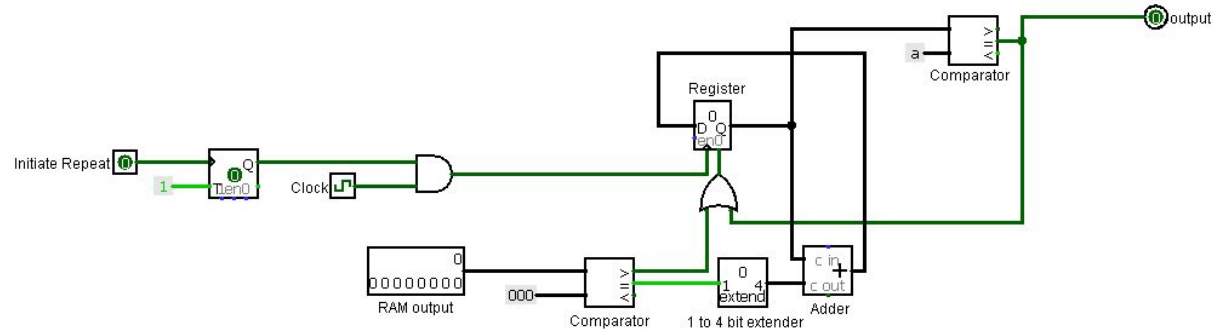
Counter



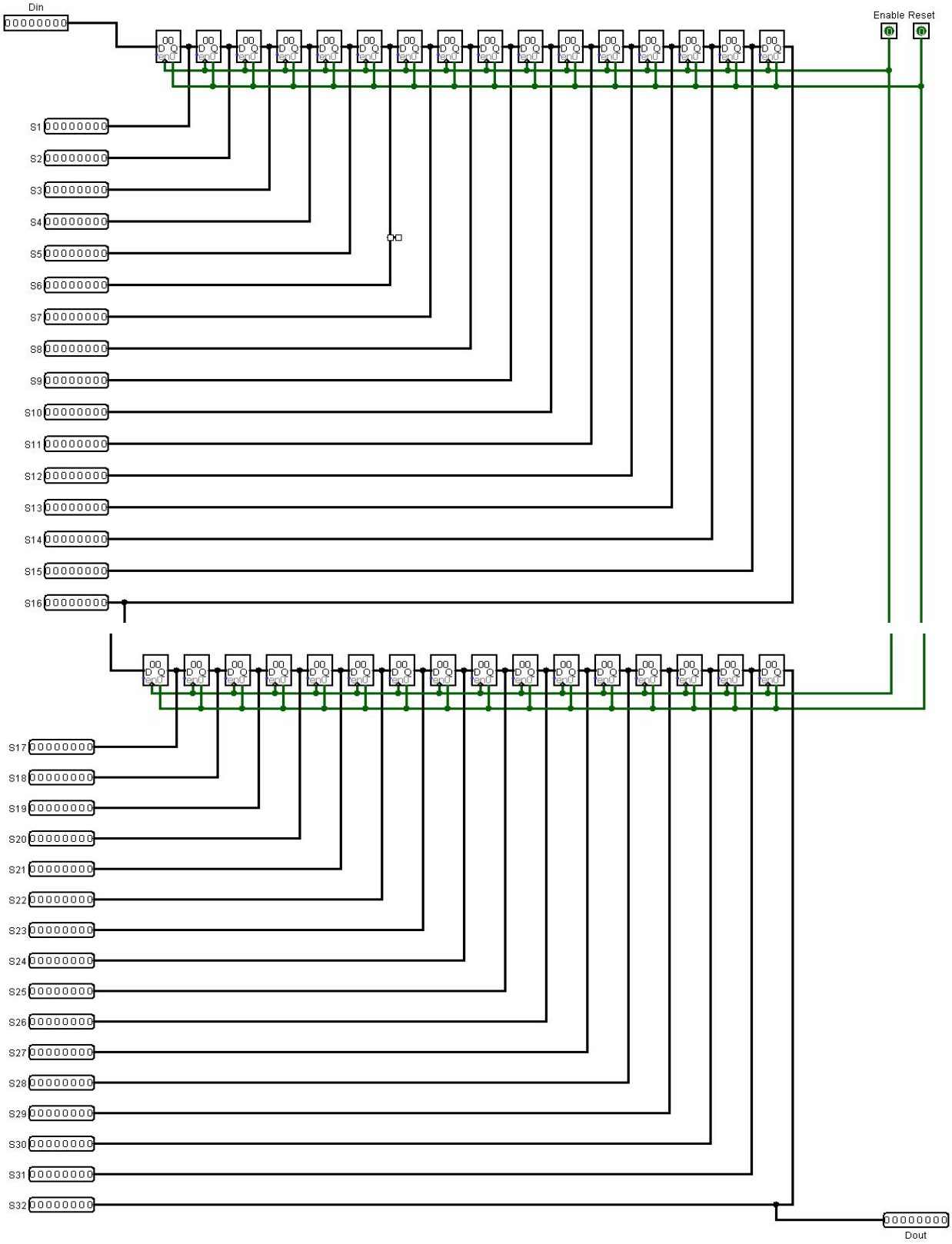
ROM Address Locator



Zero counter

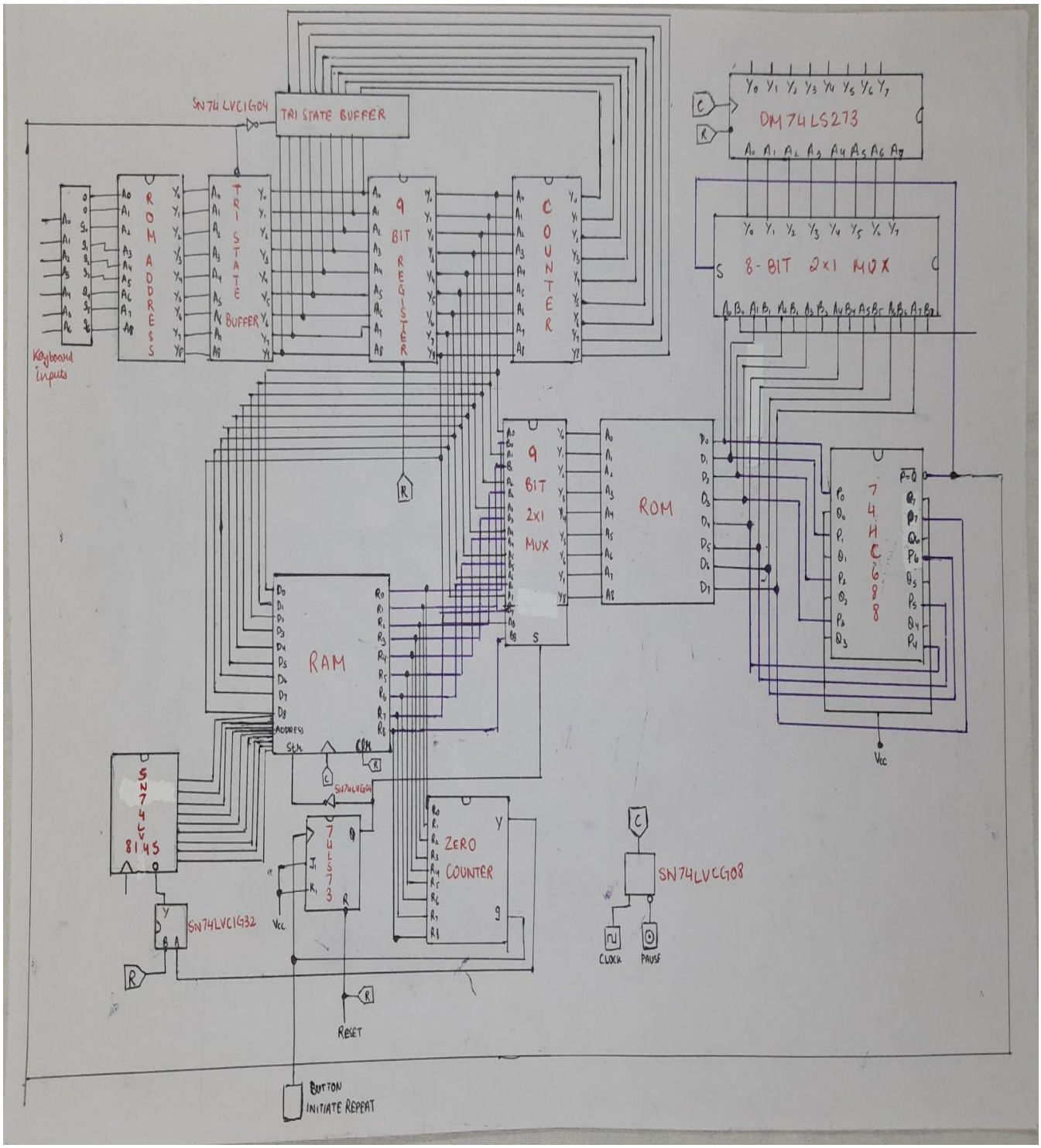


Shift Register

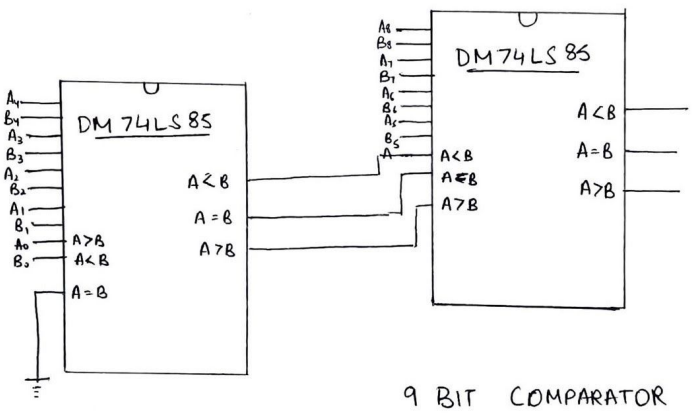


Pinout diagrams

Main circuit

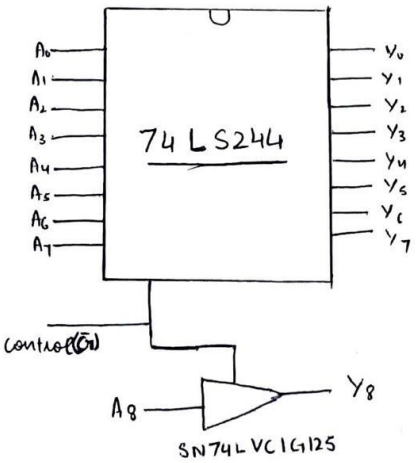


9 Bit Comparator



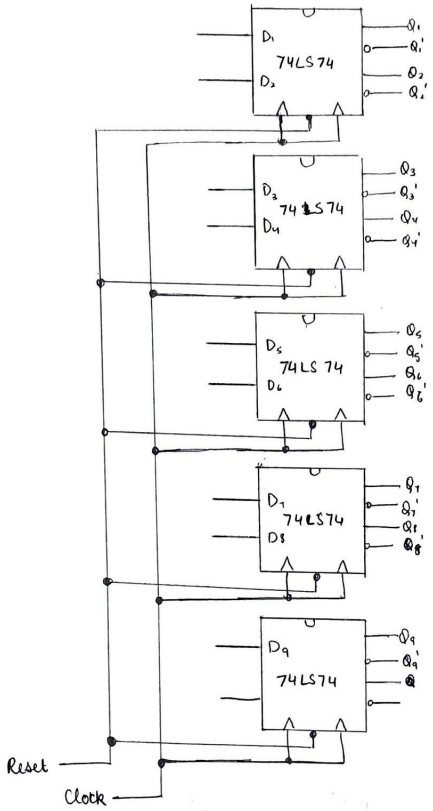
9 BIT COMPARATOR

9 Bit Tri State Buffer



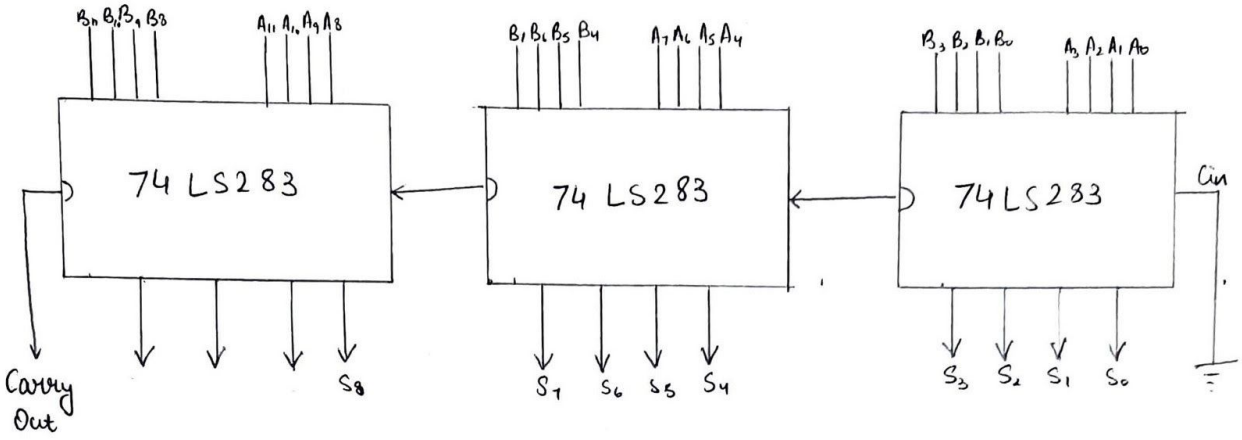
9 BIT BUFFER

9 Bit Register



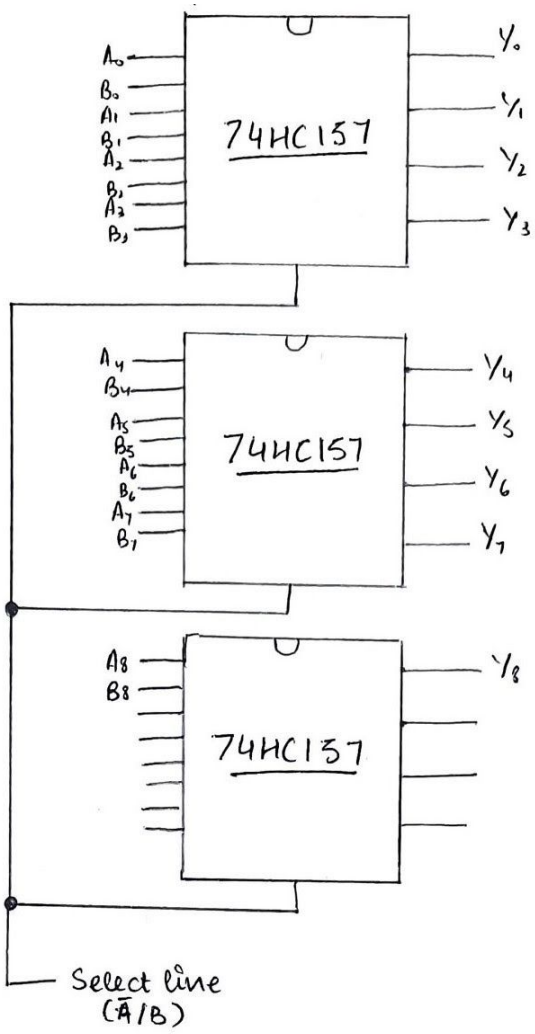
9 BIT REGISTER

9_Bit Adder

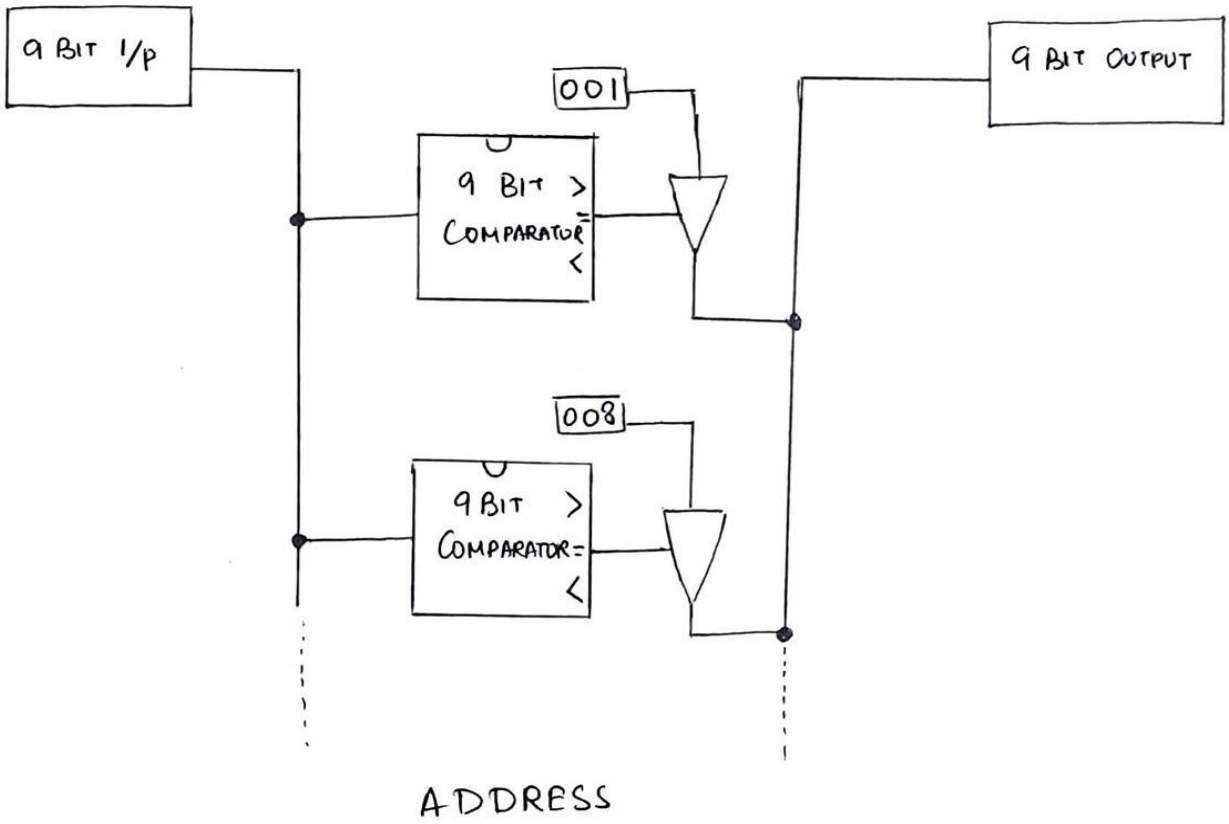


9 BIT ADDER

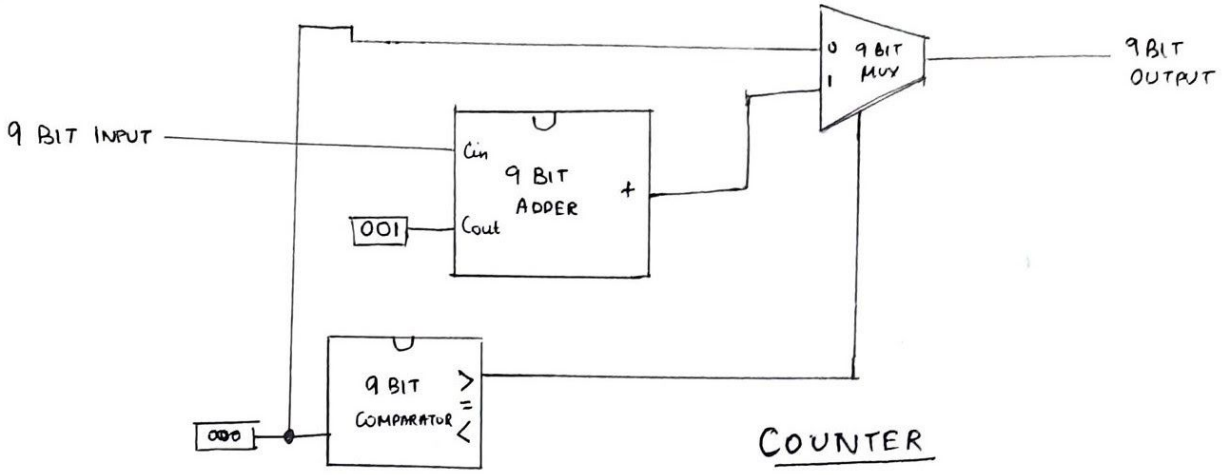
9 Bit 2x1 MUX



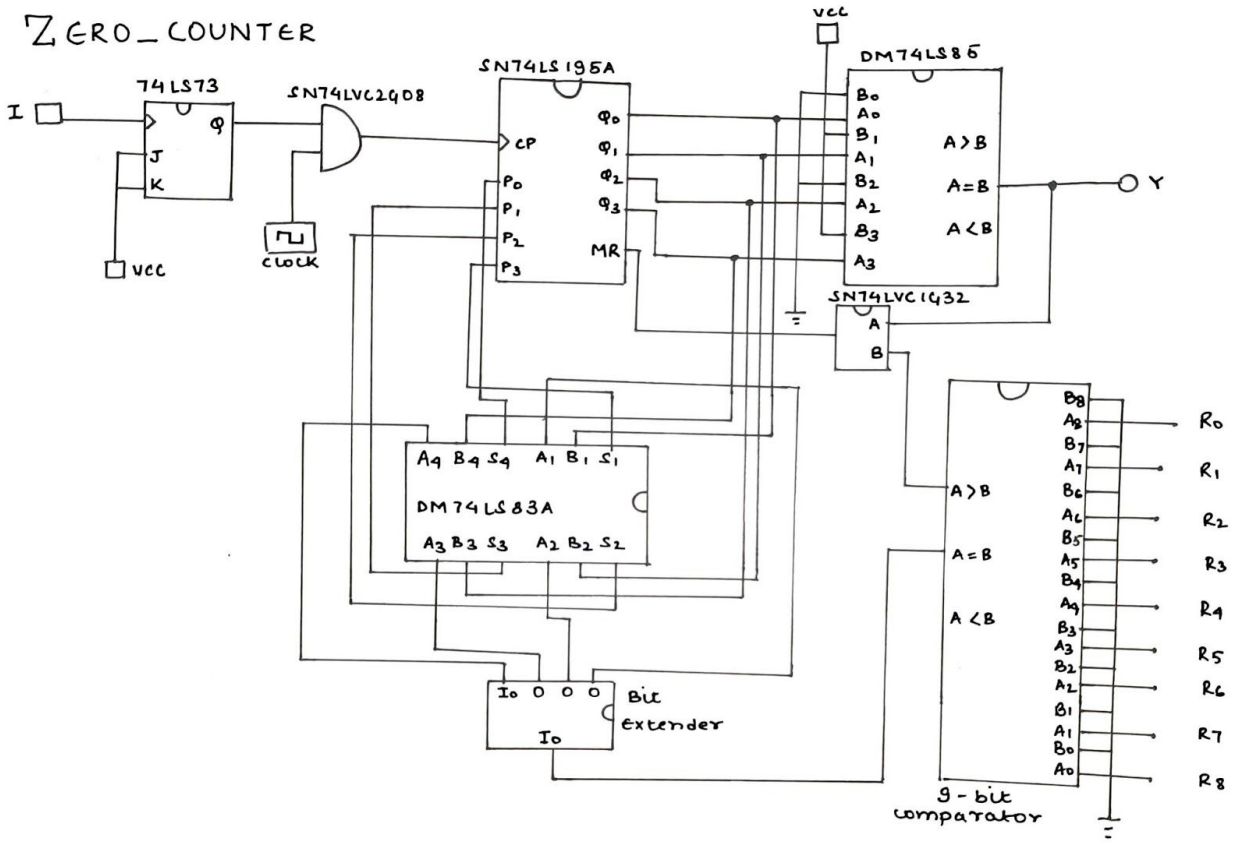
ROM Address Locator



Counter



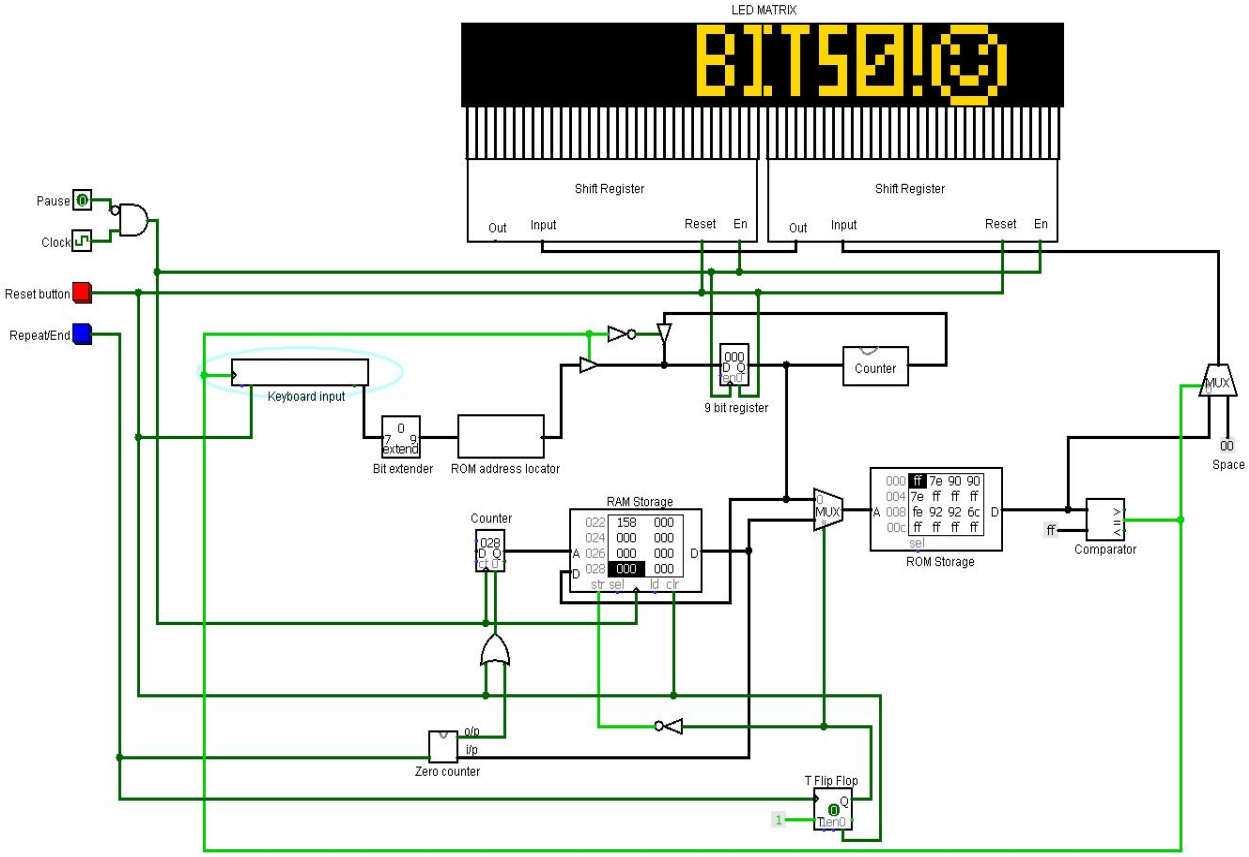
Zero Counter



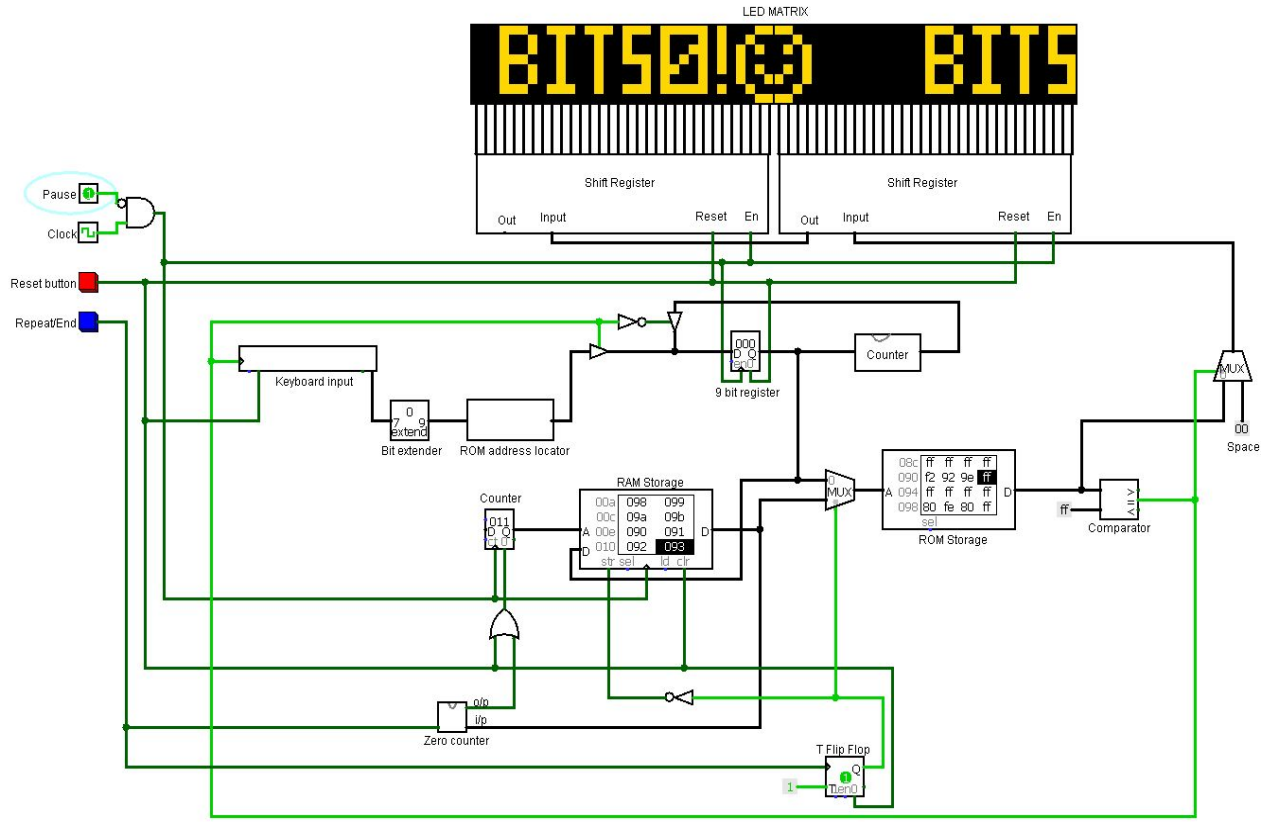
Sample input and output

INPUT: bitso!/
OUTPUT: BITSo!*smiley*

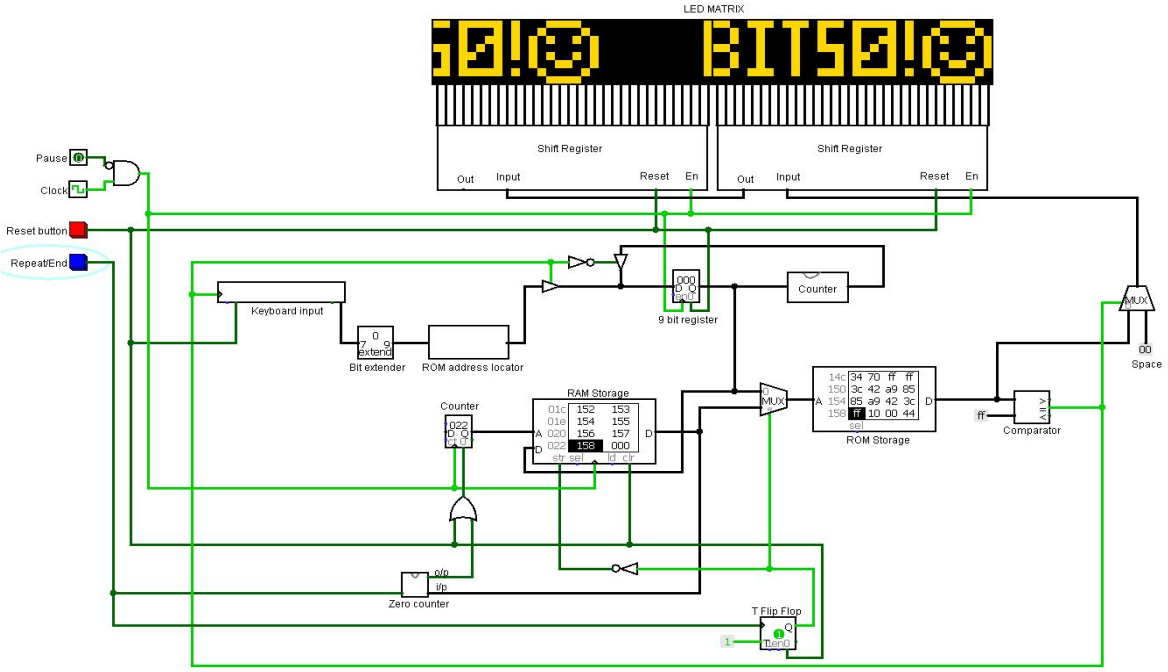
Scroll:



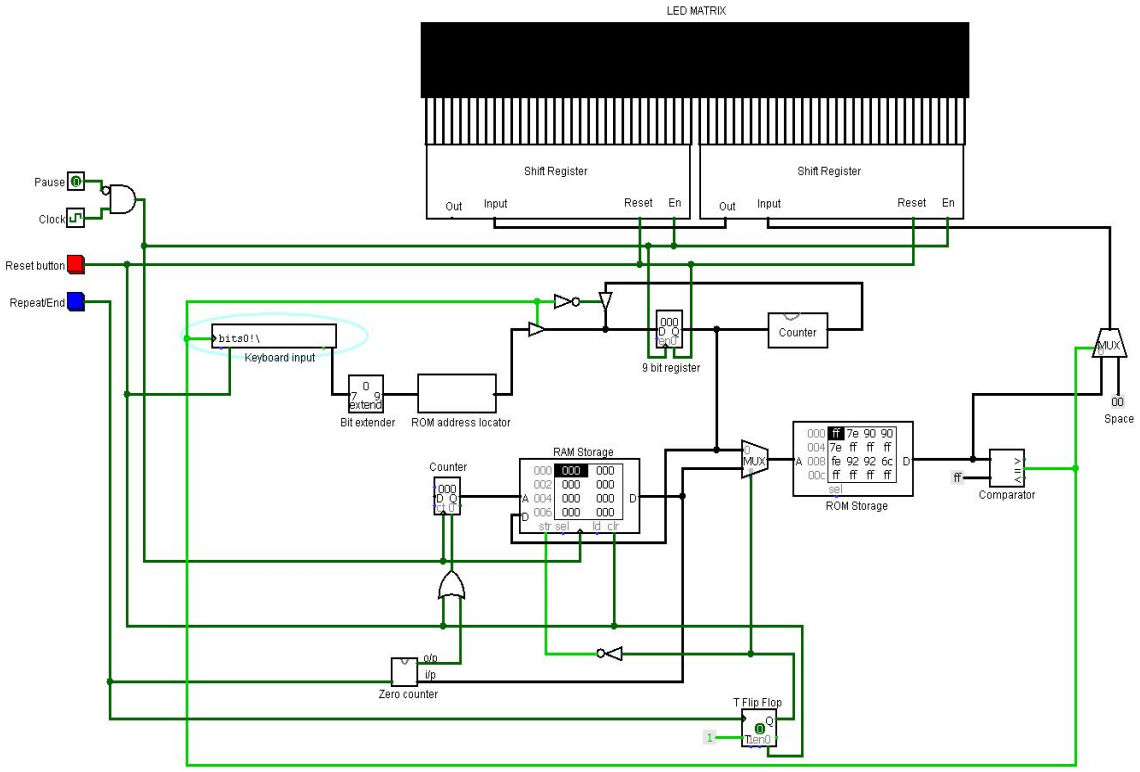
Pause:



Repeat:



Reset:



Logging Table:

Pause	T FF	Count_Out	0_Count	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15	S16	S17	S18	S19	S20
0	0	000	0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0	0	009	0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0	0	00a	0	fe	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0	0	00b	0	92	fe	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0	0	00c	0	92	92	fe	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0	0	00d	0	6c	92	92	fe	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0	0	041	0	00	6c	92	92	fe	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0	0	042	0	82	00	6c	92	92	fe	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0	0	043	0	fe	82	00	6c	92	92	fe	00	00	00	00	00	00	00	00	00	00	00	00	00
0	0	044	0	82	fe	82	00	6c	92	92	fe	00	00	00	00	00	00	00	00	00	00	00	00
0	0	099	0	00	82	fe	82	00	6c	92	92	fe	00	00	00	00	00	00	00	00	00	00	00
0	0	09a	0	80	00	82	fe	82	00	6c	92	92	fe	00	00	00	00	00	00	00	00	00	00
0	0	09b	0	fe	80	00	82	fe	82	00	6c	92	92	fe	00	00	00	00	00	00	00	00	00
0	0	09c	0	80	fe	80	00	82	fe	82	00	6c	92	92	fe	00	00	00	00	00	00	00	00
0	0	091	0	00	80	fe	80	00	82	fe	82	00	6c	92	92	fe	00	00	00	00	00	00	00
0	0	092	0	f2	00	80	fe	80	00	82	fe	82	00	6c	92	92	fe	00	00	00	00	00	00
0	0	093	0	92	f2	00	80	fe	80	00	82	fe	82	00	6c	92	92	fe	00	00	00	00	00
0	0	094	0	9e	92	f2	00	80	fe	80	00	82	fe	82	00	6c	92	92	fe	00	00	00	00
0	0	0d1	0	00	9e	92	f2	00	80	fe	80	00	82	fe	82	00	6c	92	92	fe	00	00	00
0	0	0d2	0	fe	00	9e	92	f2	00	80	fe	80	00	82	fe	82	00	6c	92	92	fe	00	00
0	0	0d3	0	8a	fe	00	9e	92	f2	00	80	fe	80	00	82	fe	82	00	6c	92	92	fe	00
0	0	0d4	0	92	8a	fe	00	9e	92	f2	00	80	fe	80	00	82	fe	82	00	6c	92	92	fe
0	0	0d5	0	a2	92	8a	fe	00	9e	92	f2	00	80	fe	80	00	82	fe	82	00	6c	92	92
0	0	0d6	0	fe	a2	92	8a	fe	00	9e	92	f2	00	80	fe	80	00	82	fe	82	00	6c	92
0	0	129	0	00	fe	a2	92	8a	fe	00	9e	92	f2	00	80	fe	80	00	82	fe	82	00	6c
0	0	12a	0	fa	00	fe	a2	92	8a	fe	00	9e	92	f2	00	80	fe	80	00	82	fe	82	00
0	0	151	0	00	fa	00	fe	a2	92	8a	fe	00	9e	92	f2	00	80	fe	80	00	82	fe	82
0	0	152	0	3c	00	fa	00	fe	a2	92	8a	fe	00	9e	92	f2	00	80	fe	80	00	82	fe
0	0	153	0	42	3c	00	fa	00	fe	a2	92	8a	fe	00	9e	92	f2	00	80	fe	80	00	82
0	0	154	0	a9	42	3c	00	fa	00	fe	a2	92	8a	fe	00	9e	92	f2	00	80	fe	80	00
0	0	155	0	85	a9	42	3c	00	fa	00	fe	a2	92	8a	fe	00	9e	92	f2	00	80	fe	80
0	0	156	0	85	85	a9	42	3c	00	fa	00	fe	a2	92	8a	fe	00	9e	92	f2	00	80	fe
0	0	157	0	a9	85	85	a9	42	3c	00	fa	00	fe	a2	92	8a	fe	00	9e	92	f2	00	80
0	0	158	0	42	a9	85	85	a9	42	3c	00	fa	00	fe	a2	92	8a	fe	00	9e	92	f2	00
0	0	159	0	3c	42	a9	85	85	a9	42	3c	00	fa	00	fe	a2	92	8a	fe	00	9e	92	f2
0	0	000	0	00	3c	42	a9	85	85	a9	42	3c	00	fa	00	fe	a2	92	8a	fe	00	9e	92
0	0	000	0	00	00	3c	42	a9	85	85	a9	42	3c	00	fa	00	fe	a2	92	8a	fe	00	9e
0	0	000	0	00	00	00	3c	42	a9	85	85	a9	42	3c	00	fa	00	fe	a2	92	8a	fe	00
0	0	000	0	00	00	00	00	3c	42	a9	85	85	a9	42	3c	00	fa	00	fe	a2	92	8a	fe
0	0	000	0	00	00	00	00	00	3c	42	a9	85	85	a9	42	3c	00	fa	00	fe	a2	92	8a
0	0	000	0	00	00	00	00	00	00	3c	42	a9	85	85	a9	42	3c	00	fa	00	fe	a2	92

Additional Functionalities

1. **Recursive scrolling**: The recursive scrolling functionality allows the user to automate the display of text repeatedly. This feature may be used to enrich the visual appeal of display advertisements to customers. It can also be used at places such as airports to display announcements (such as flight arrivals/departure) that have to run for long intervals of time .To use this feature the user must first enter some text into the keyboard and then press the "Repeat/End" button (marked in blue). There is no time constraint on the usage of this functionality, that is, it can work for as long as the user wishes it to. However, at the beginning of each simulation the user must decide whether to enable or disable this functionality. If the user chooses to enable it, he/she would have to reset the circuit and retype the text in order to enable it. If the user wishes to discontinue this feature they only need to press the "Repeat/End" button once again. This design is implemented using a RAM, a 2:1 MUX, a 9-bit counter and a user-defined counter(Zero counter). The zero counter has its purpose in resetting the RAM address to 000 as soon as it encounters ten consecutive spaces in the led matrix.

2. **Pause button**: If the user wishes to temporarily/permanently pause the simulation they may do so using the "pause" button. This feature comes into use if the user wishes to display a static message.

3. **Variety of available characters**: To ensure that the text is rich there are a variety of characters available as input comprising alphabets [a-z], numbers [0-9], some special characters [' ! ', ' ? ', ' . ', ' , ', space] and a few emojis. On top of this, the ROM is made big enough to bind a keyboard key to a specific user-defined phrase.

4. **Error detection**: The circuit stops working if an invalid character is entered into the keyboard. This ensures that the circuit does not malfunction due to an invaild input.

5. **Scalability of Display:** The Shift Registers and the LED matrices are designed such that they can be easily scaled up as per the needs of the user

Bill of Materials

IC	Function	Quantity
DM74LS283	4 bit adder	4
SN74SS195A	4 bit register	1
DM74LS273	8 bit register	64
DM74LS85	4 bit comparator	95
74HC688	8 bit Comparator	1
74LS244	8 bit tri state buffer	92
74HC157	Quad 2-input multiplexer	5
SN74LS73A	Dual J-K Flip Flops	12
SN74LVC1G34	Single Buffer Gate	46
SN74LVC2G08-EP	2 Input AND Gate	2
SN74LVC1G32-EP	2 Input OR Gate	1
SN74LVC1G04	Single Inverter Gate	2

Appendix

Links to All Datasheets

- DM74LS273.pdf
- SN74LVC1G34.pdf
- DM74LS283.pdf
- SN74LVC1G04.pdf
- SN74LS73A.pdf
- SN74LVC1G32-EP.pdf
- SN74lVC2G08-EP.pdf
- 74HC157.pdf
- DM74LS85.pdf
- 74HC688.pdf
- 74LS244.pdf
- SN74LS195A.pdf