8085 based Braille Translator

May 18, 2019

Vishal Singh
202/EC/15

Vivek Kumar
204/EC/15

Electronics and Communication Engineering

Netaji Subhas
Institute of technology

Contents

1 Acknowledgement

We would like to express our gratitude to Professor Dhananjay V. Gadre, without whom this project couldn't see the light of the day.

We would like to thank our parents for their constant motivation and encouragement throughout this project.

We would like thank our Microprocessor professor Mr. Abhay Sharma for his outstanding lectures on the subject.

We would like to express our gratitude to our freinds who were both critical and supportive when needed.

2 Introduction

The Braille system is around for nearly 200 years, being invented in the year 1824. Technology has progressed at lightspeed during these years, and paper is increasingly becoming obsolete.

In this electronic age, everything is becoming more and more digital. Paper documentation is being used less, and instead is stored on digital devices instead. Moreover, there is also the issue of increasing deforestation and its effect on the environment. The Braille Translator is an attempt to provide a paperless reading device for the visually impaired. This device translates anything you type on the keyboard to braille on the coil box.

This project is part of the 8085 Microprocessor Practical Laboratory curriculum of ECE branch in NSIT. Seeing this project to completion made us achieve the dual purpose of developing a useful device while also gaining knowledge about primitive yet maieutically important computer architectures. Doing this enhanced our practical knowledge of the interfacing and programming of a basic microprocessor which is necessary if someone wants to learn about the working of processors and controllers more complex in nature.

3 User Guide

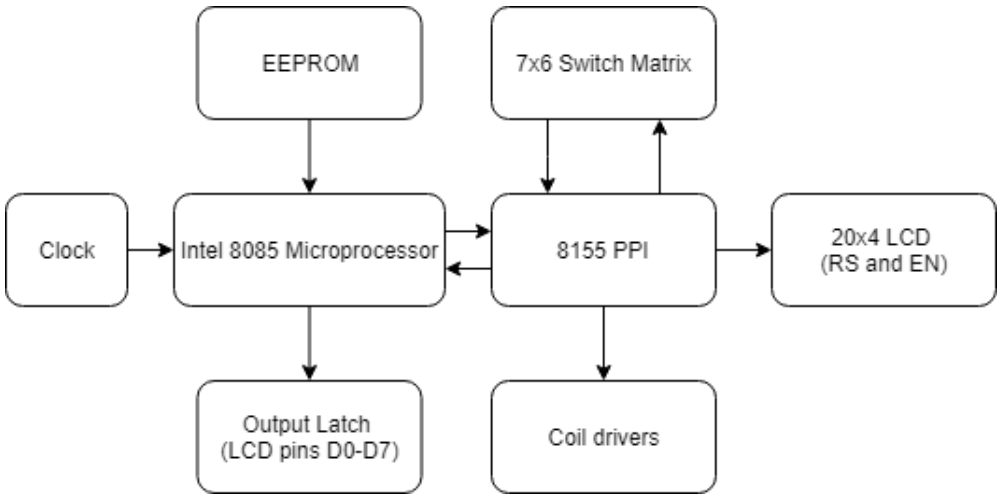
The user has to type the desired words or paragraphs on the keyboard provided. The number of characters which can be typed at once is limited to 80. The typed letters are printed on the LCD which is also included in the device. The user can also 'Backspace' a wrongly typed letter, or just reset and remove all typed characters. Once the desired paragraph is typed completely, the user can then press 'Enter' to transfer the characters to the user memory and translate them.

Once the characters are entered and passed, the characters then appear in braille script, imprinted on the coil box, one by one. The user can choose whether the braille characters are forwarded one by one on the the press of a button, or automatically increment one by one through some time delay.

4 Hardware Description

4.1 Hardware Overview

- The device is divided in two PCBs. The lower PCB houses the microprocessor, memory and interfacing ICs, while the upper PCB has the keyboard interface and LCD.
- The device takes input from a 5V jack. A power on LED is also provided. item8085 is used as the microprocessor chip. Also, pulled down switch is connected to th SID pin, and a LED is connected in series with a resistor to the SOD pin.
- 28C256 is the chip used for ROM, while 8155 PPI is used for RAM as well as for input/output interfacing.
- A latch is used to latch the lower order adress bus when ALE is high.
- A decoder is used to decode the memory adress for ROM operations.
- Another latch in addition to 8155 PPI is used for output operations. The outputs of this latch are connected to the LCD data pins. Another decoder is used to decode for this latch.
- A box housing 6 coils is used as the braille embosser.
- 6 BD139 transistors are used to provide power to the coils.
- A buzzer is used for audio output.
- A 7x6 keyboard interface is housed on the upper PCB.
- A 20x4 character LCD is also provided.

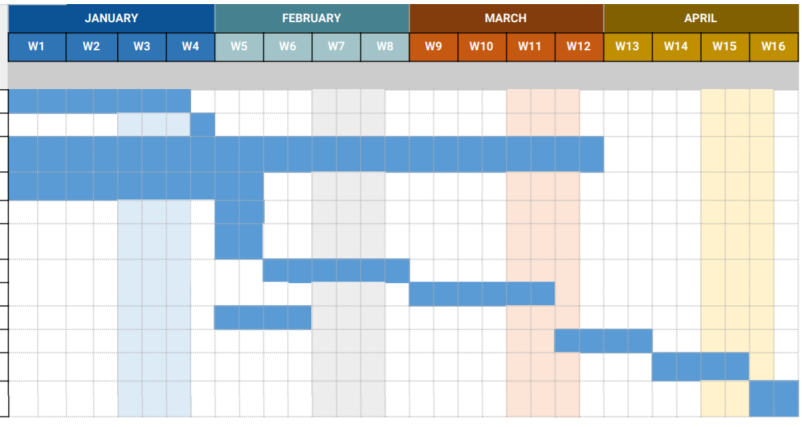


Block Diagram

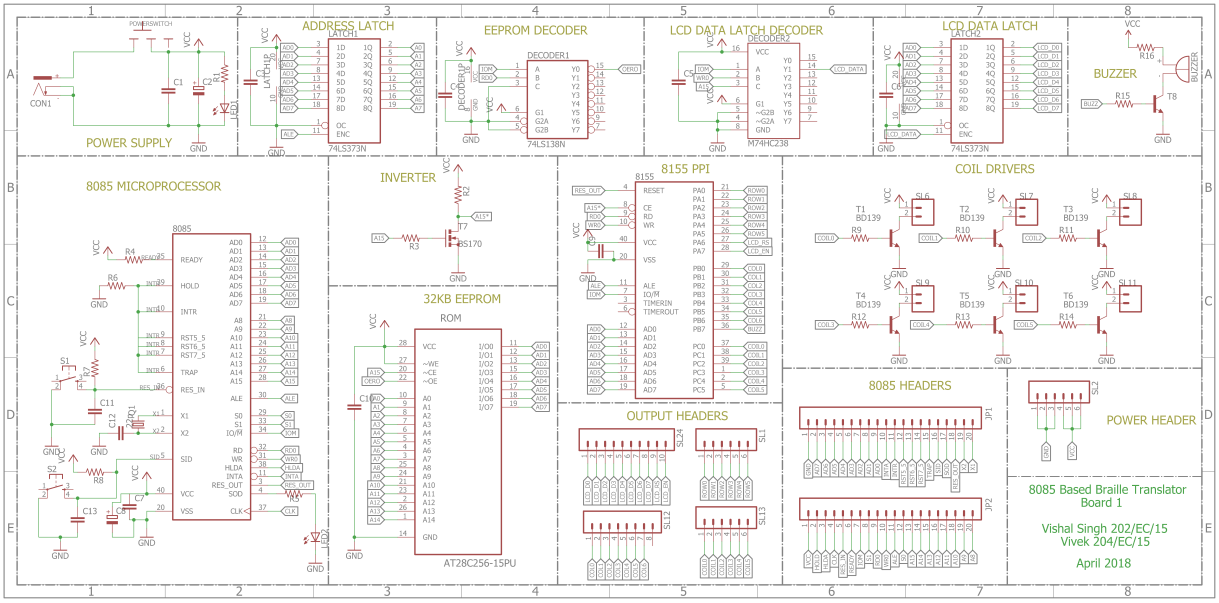
GANTT CHART

PROJECT TITLE Braille Transla

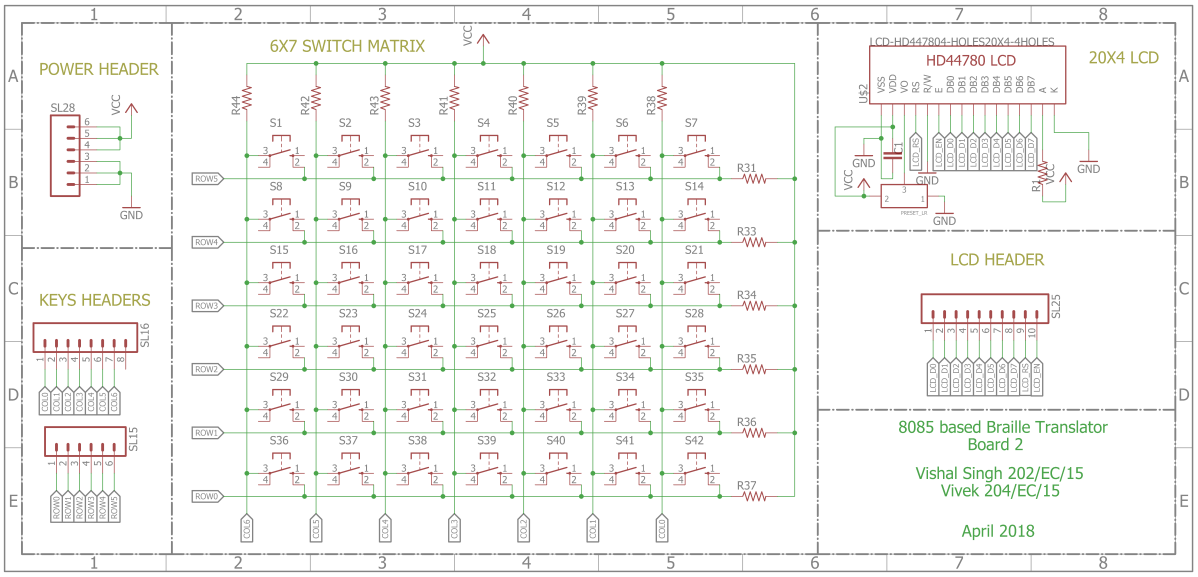
S. NO.	TASK TITLE	ALLOTED TIME (WEEKS)
1	Exploring ideas and finalise project	2
2	Project approval	1
2	Learning 8085 architecture and interfacing	12
3	Back research	5
4	Arduino testing	1
5	Synopsis preparation and submission	1
6	Schematic preparation	3
7	Board layout	3
8	Woodwork	2
9	EEPROM programming	2
10	Testing and debugging	2
11	Project documentation and video shoot	1



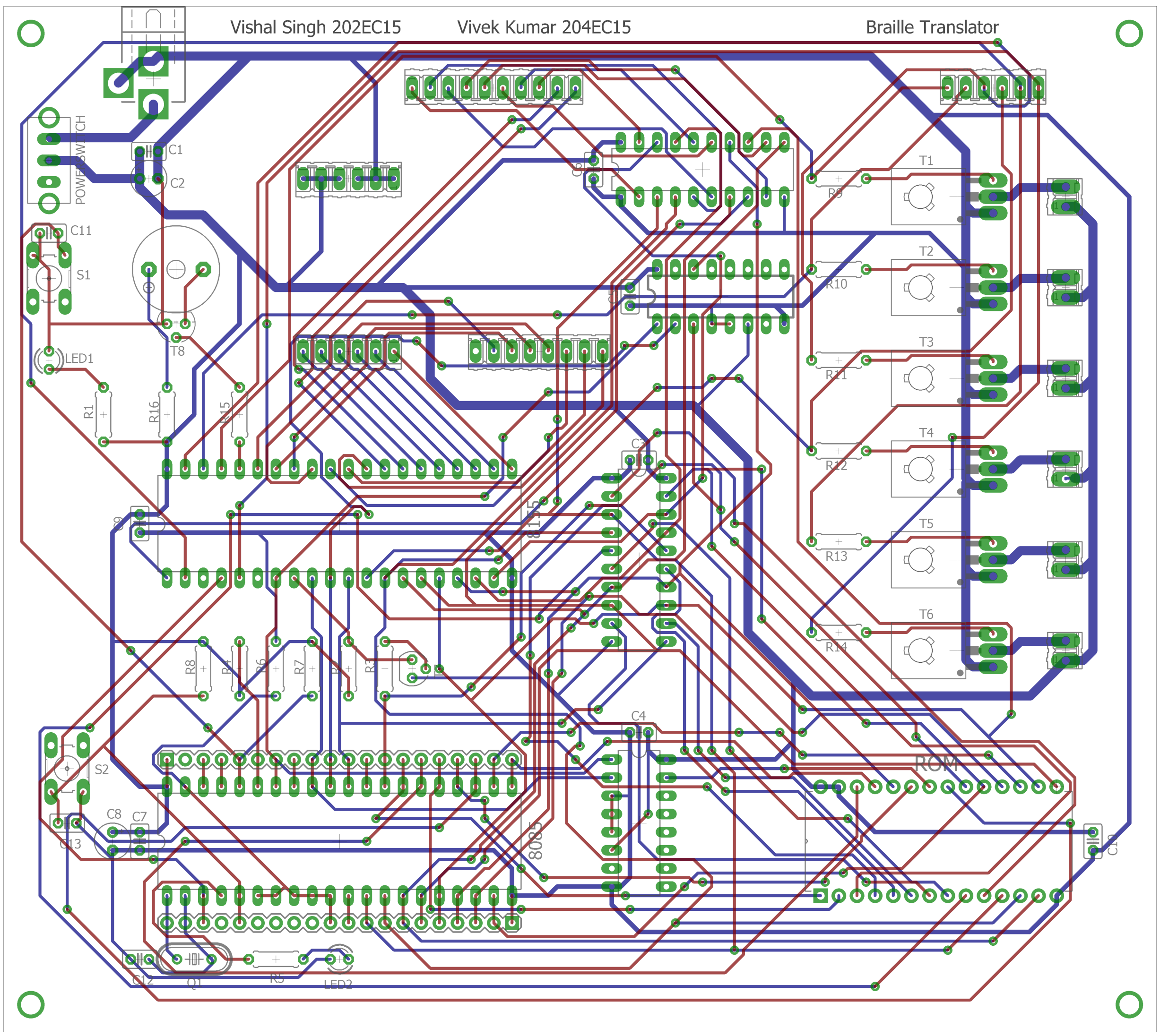
4.1.1 Schematic -1



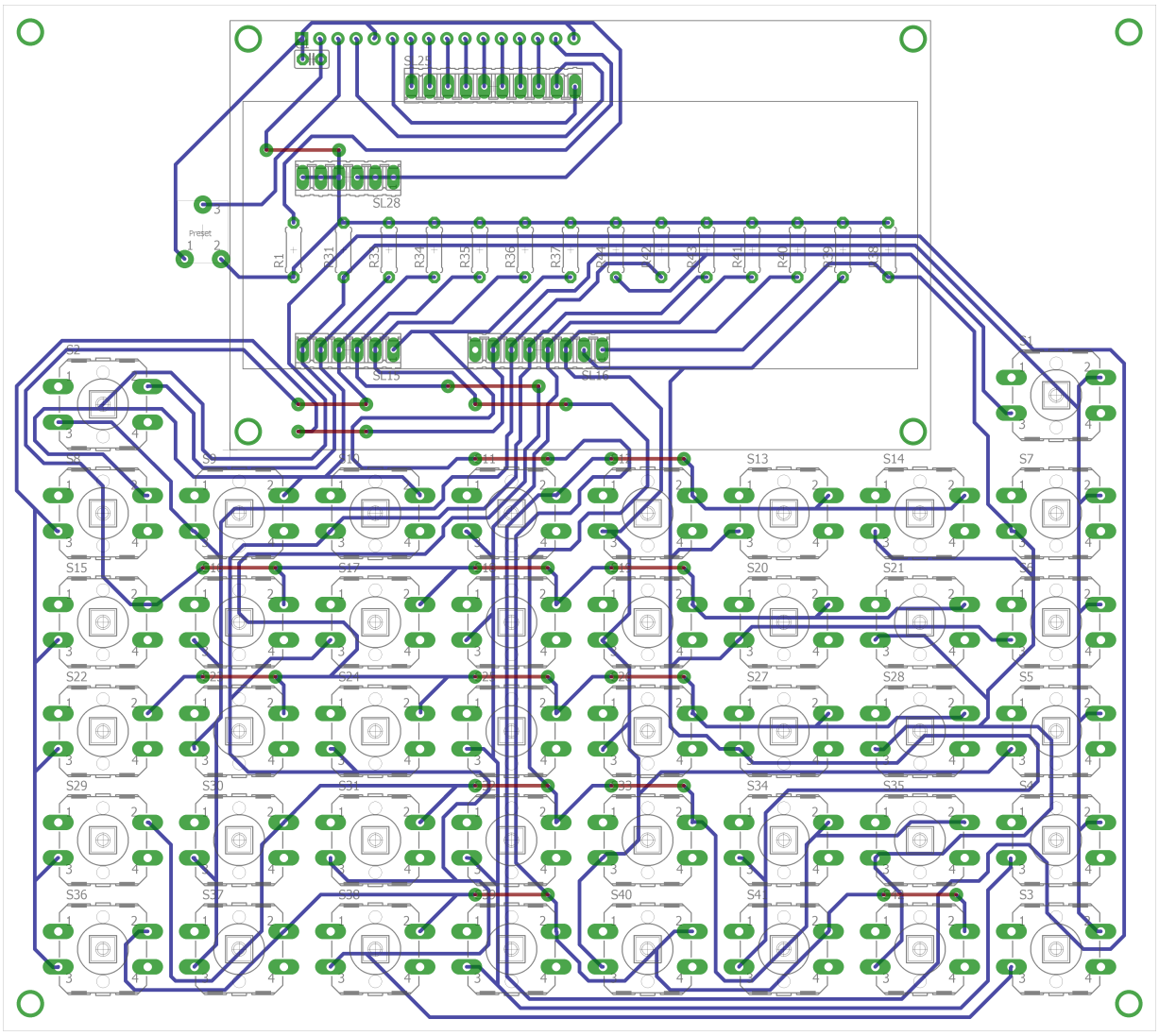
4.1.2 Schematic -2



4.1.3 Layout -1

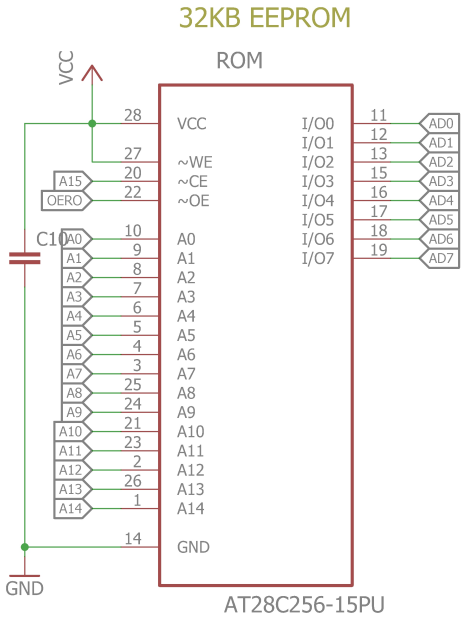

Vishal Singh 202EC15
Vivek Kumar 204EC15
Braille Translator
POWER SWITCH
C1
C2
C11
S1
LED1
T8
R1
R16
R15
T1
T2
T3
T4
T5
T6
R9
R10
R11
R12
R13
R14
C4
ROM
8085
S2
C8
C7
C12
Q1
R5
LED2
R8
R6
R7

4.1.4 Layout -2


SL25
SL28
SL13
SL16
Preset
R1
R31
R32
R34
R35
R36
R37
R24
R2
R33
R41
R40
R39
R38
S2
S8
S9
S10
S11
S13
S14
S7
S15
S17
S20
S21
S22
S24
S27
S28
S5
S29
S30
S31
S33
S34
S35
S36
S37
S38
S40
S41
S3

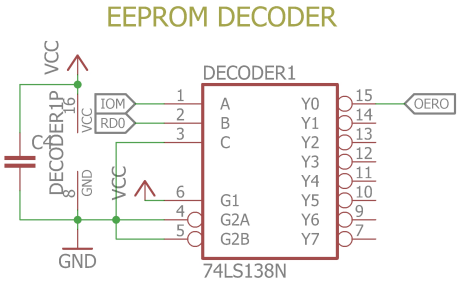
4.2 Memory Interfacing

4.2.1 32kB ROM interfacing



We have used a 32kB ROM. A15 is connected to CE(Chip enable), so it is enabled whenever the A15 pin goes low. The other adress pins are connected to the ROM adress pins which point to the ROM register locations. Hence the adress of the ROM registers is from 0000H to 7FFFH.

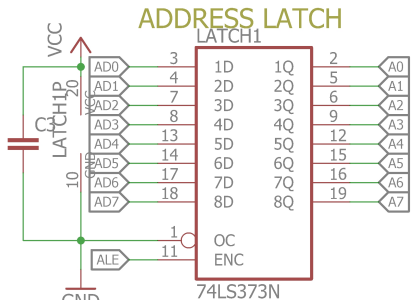
4.2.2 ROM decoder



This is used to generate IOM* and IOW* signals, and enable the ROM using the IOM* signal. Hence, whenever the IO/M pin is low, the RD0 pin is low and the ROM is decoded at the suitable adress, the ROM places the data from the

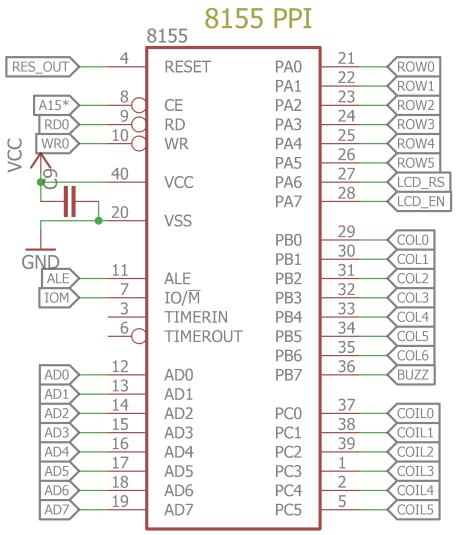
applied adress on the data bus. The output of the decoder is connected to output enable.

4.2.3 Adress Latch



The data bus and the lower order adress bus have to be demultiplexed for their use in 8085 microprocessor. For this purpose, we use a 74ls373 latch, which latches the lower order adress bus whenever ALE goes high. When ALE is low, the pins AD0-AD7 are recognised as data bus.

4.3 8155 PPI



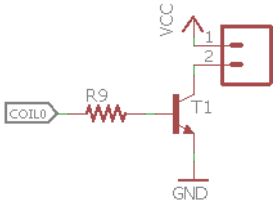
The 8155 PPI includes a 256B RAM, a timer, two 8-bit ports and one 6-bit port. We have used the RAM and the three ports only. A15* is used as the chip enable, so the adress of the RAM is from 8000H to 80FFH. The RAM can also be accessed on other adress due to partial decoding. There is no need to demultiplex the lower order adress bus as there is already an internal latch for the same purpose in the 8155 chip. The control and status signals from the 8085 are connected directly the resoective signals on the memory chip.

In I/O operation, the IO adress are also decoded by the A15* pin. Adress lines A2-A0(after decoding) are used to select one of the three port registers and control word register. The other adress lines are dont-cares. The I/O ports are configured by writing a control word in the control register. Since the 8155 is enabled by the A15* line, the I/O operations are decoded by the same logic. The adress of the control word register is, therefore, 80H, while that of Port A, Port B, and Port C are 81H, 82H, and 83H respectively.

In addition to the 8155 ports, and additional output latch is used for the data pins of the LCD. A decoder is used is used to decode for the latch. IOW* signal is generated along with active low A15 line chip enable. Therefore, the output adress of the latch is 00H, while other adresses with A15(or A7) low could also be used due to partial decoding.

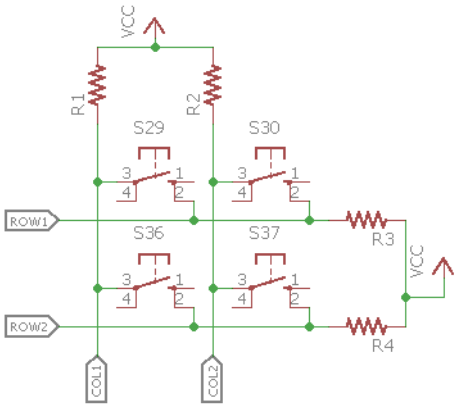
4.4 Other various circuits

4.4.1 Coil Drivers



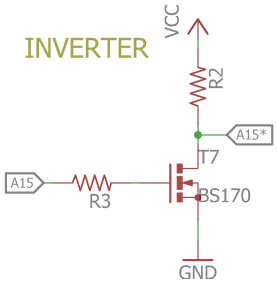
A BD139 transistor is used to drive the coil. When the input is low, the BE junction of the transistor is cut off, and the coil is not energised. When the input goes high, the transistor goes into saturation, and the coil gets energised, a magnetic field is set inside it, and it pushes the magnet inside the coil upwards.

4.4.2 Switch Matrix



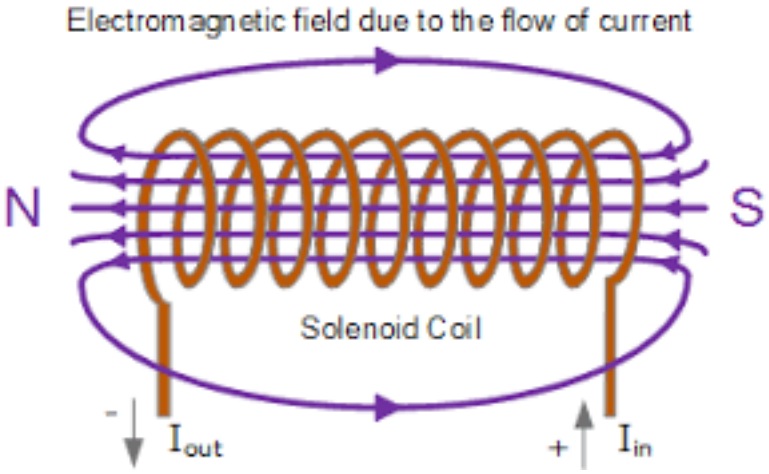
We have used a 7x6 switch matrix as keyboard interface. A 2x2 matrix is shown above for description. All the pins in rows and columns are pulled high. We output a 0 to the rows. Whenever a switch is pressed, the corresponding column pin goes low. Then we have to output 0 to the columns and check the rows. This way, we get the corresponding number of the switch which is pressed.

4.4.3 Inverter Circuit



It is a NOT gate using a transistor. When the input is low, the BE junction is cut off, hence the output is pulled high. When the input goes high, the transistor goes into saturation, and the output is shorted to ground. This way, it serves as a basic logic inverter. The BS170 MOSFET is used since it produces less amount of noise.

4.4.4 Electromagnetic Coil



A neodymium magnet placed inside an electromagnetic coil is used to impress Braille characters. 6 of them are required to represent Braille aplhabet. When the coil is energised, a magnetic field is set up inside the coil, which pushes the neodymium magnet upward, while it stays down when it is not energised. Different combination of coils are energised for each character. A flywheel diode is also connected in parallel to the coil, to prevent damage to the circuit from back emf of the coil.

5 Software Description

5.1 System specifications

5.1.1 Memory Map

	ROM	RAM
Size	32 KB	256 Byte
Start	0000h	8000h
End	7fffh	80ffh

5.1.2 IO ports

Device	Port address	Function
LCD data latch	00h	Input to LCD data lines
Porta 8155	81h	output to rows, lcd register select, lcd enable
Portb 8155	82h	input from columns
Portc 8155	83h	pin control – output braille code

We have reserved first six RAM locations for data manipulation and temporary storage of results. Characters input from keyboard are registered from memory location 8006h onwards.

5.1.3 Reserved RAM locations

Device	Port address	Function
LCD data latch	00h	Input to LCD data lines
Porta 8155	81h	output to rows, lcd register select, lcd enable
Portb 8155	82h	input from columns
Portc 8155	83h	pin control – output braille code

5.2 Peripheral specifications

5.2.1 Keyboard

Keycode of every key is determined by its location in matrix i.e. by its row and columns number (both following zero based indexing) given by

code = row * cols + col

where row is row index of key, col is column index and cols is number of columns in matrix i.e.

cols=7

code is keycode of pressed key. Keys are organised in such a way that facilitates easy calculation of corresponding ascii value by addition of an offset value. For instance

Keys with keycode 0 to 9 (both inclusive) have characters from 0 to 9. Ascii value is obtained by adding ascii value of '0' to each input keycode. Thus on pressing key with chracter 6 (and keycode 6) accumulator gets a value of 06h to which 30h (ascii value of '0') is added to get 36h which is ascii value of '6'. A similar scheme is adopted for successive keys, with a different offset (keycode 0ah has character 'A' , so offset is 37h for keys till 'Z'). Special characters are dealt with separately using conditional jump statements.

A lookup table is present in memory that maps every keycode to corresponding 6 bit braille code. Entries in this table are laid out sequentially as per keycodes. This facilitates indexing of table using keycode of depressed key. What this means is – Base address of table contains braille code for character with keycode 0, next address has braille code for keycode 1 and so on.

5.2.2 KYBORD SUBROUTINE LOGIC

Whenever KYBORD is called, 8085 polls keyboard matrix to check pressed key. Since this subroutine is being used for repeatedly taking user input, a firsthand check is made to ensure that previously pressed key was released before checking next key press. Software debouncing is done by providing a delay of 10ms upon detection of key press before further processing.

Pseudo code
1. Ground all rows by outputting C0h on PortA
2. Read columns (PortB)
3. IF (one of the bits is low):
...3.1 GOTO 2
4. ELSE:
...4.1 CALL DBONCE ;10 ms delay
...4.2 Read columns
...4.3 IF (none of the bits is low):
.....4.3.1 GOTO 4.2
...4.4 ELSE:
.....4.4.1 CALL DBONCE
.....4.4.2 CALL COL-CHECK
.....4.4.3 CALL ROW-CHECK
.....4.4.4 CALL MULT
.....4.4.5 Set A := A + M[8001h]
5. RETURN

Here COL-CHECK is a subroutine which reads columns from PortB and shifts A bits until it finds a low bit incrementing the memory location 8001h after each shift. Column index of depressed key is stored in 8001h.

Similarly ROW-CHECK checks for row index. However this is done by grounding each row and checking the columns for a low bit starting with row 0 and incrementing memory location 8000h each time until the row is found. Row index is stored in 8000h.

MULT is a multiplication subroutine for multiplying row index with number of columns. Finally A holds a value given by

A = row * cols + col

5.2.3 Display

We are using a 20 x 4 LCD display to show characters input by user. Register select and enable signals to LCD are controlled by bit 6 and bit 7 of porta respectively. Data latch connected to the lcd has address 00h. LCD operates in 8 bit 2 line mode. Since DDRAM addresses of LCD are such that after last address of first line comes the first address of third line (and after last address of third line comes first address of second line), memory locations have been reserved to keep track of number of characters in current line and current line number. Appropriate subroutines have been used to move cursor to next line as soon as current line is filled and to next page as soon as all 4 lines are filled.

5.2.4 SHOW SUBROUTINE LOGIC

This subroutine displays key character pressed. The ascii value calculation is done based on keycode (see Keyboard section) and it is stored in 8005h.

Pseudo code
1. Load A := M[8004h] ; Memory location of current line number
2. IF (A > 3): ; Line numbers follow zero based indexing
...2.1 CALL NXT-PAGE
3. Load A := M[8005h] ; stored ascii value
4. OUT PORT-LCD ; 00h
5. CALL PULSE-LCD-DATA
6. Load A := M[8003h] ; characters in current line
7. Set A := A + 1
8. Store M[8003h] = A
9. IF (A > MAX-CHAR-COUNT) ; 20 characters are max
...9.1 CALL NXT-LINE
10. RETURN

Here the subroutine PULSE LCD-DATA is for giving pulse to LCD E pin. NXT-LINE places cursor on next line of LCD and sets character count of current line to 00h and increments the line number.

5.2.5 REMOVE-LCD SUBROUTINE LOGIC

This subroutine is called when A holds keycode of 'BACKSPACE'. It essentially clears the last output character on LCD by writing 'SPACE' character in its place and decrementing DDRAM address. It also decrements value of C. Appropriate helper subroutines have been used which move the cursor to the left/write, place the cursor at appropriate location in appropriate line and also manage number of characters in and line number of current line.

5.3 Application Logic

We have used C register to store offset of last character input by user from base address i.e. 8006h. Ports A and C of 8155 have been configured to be in output mode and Port B in input mode.

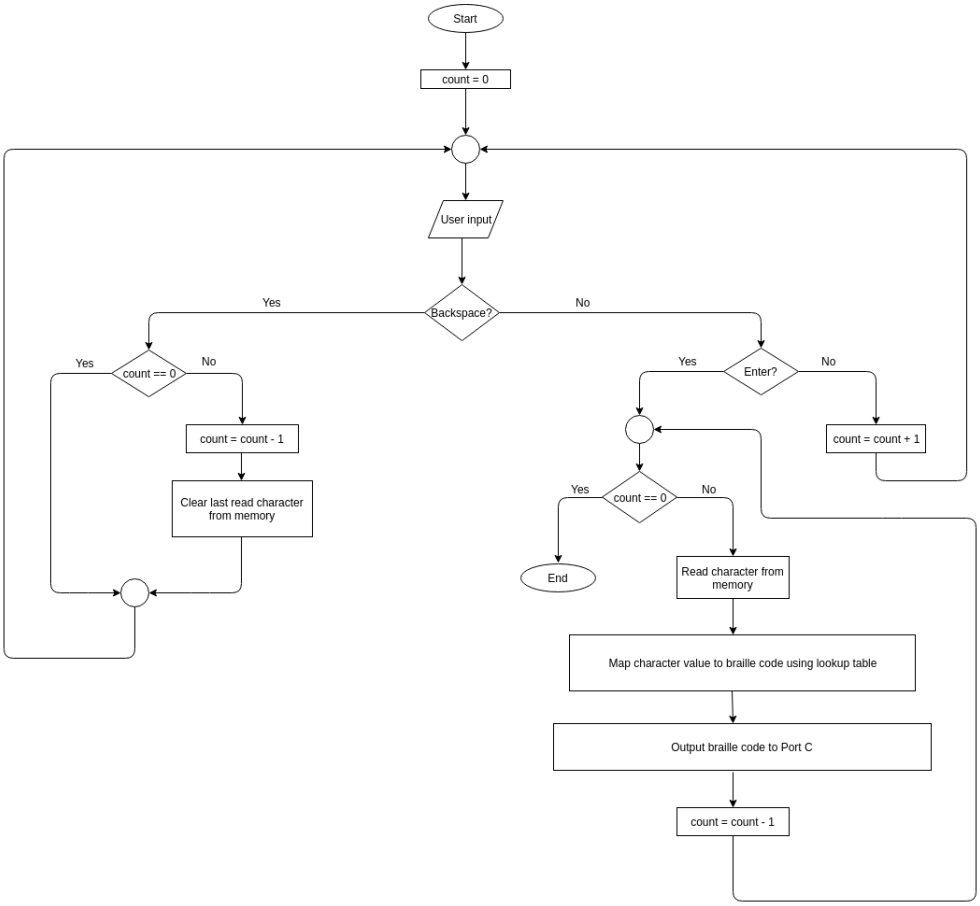
Braille transcriber pseudo code
1. Set SP := 8100h
2. Set A := 00h
3. Set C := FFh
4. Set B := 00h
5. Set HL := 8006h
6. Initialise 8155, LCD
7. WHILE (A != keycode of 'RETURN' OR C != Ffh):
...7.1 CALL KYBORD
...7.2 IF (A = keycode of 'BACKSPACE'):
.....7.2.1 IF (C = FFh):
.....7.2.1.1 GOTO 7
.....7.2.2 ELSE:
.....7.2.2.1 Set C := C – 1
.....7.2.2.2 CALL REMOVE-LCD
.....7.2.2.3 GOTO 7
...7.3 ELSE IF (A = keycode of 'RETURN'):
.....7.3.1 CALL EMBOSS
.....7.3.2 GOTO 1
...7.4 ELSE:
.....7.4.1 Set C := C + 1

.....7.4.2 Set M[HL + BC] := A
.....7.4.3 CALL SHOW
.....7.4.4 GOTO 7
8. GOTO 1

5.3.1 EMBOSS SUBROUTINE LOGIC

This subroutine maps each keycode to corresponding braille code and outputs them to PortC on pressing ‘FORWARD’ or ‘BACKWARD’ key. 6 bit braille code of every key is computed from using a lookup table (see Keyboard) and the code is output to PortC which governs flow of current in the 6 coils which control the pins. On pressing the ‘FORWARD’ key, next character is output and on pressing the ‘BACKWARD’ key, previous character is output.

6 Flow Chart


Start
count = 0
User input
Backspace?
Yes
No
count == 0
Yes
No
count = count - 1
Clear last read character from memory
Enter?
Yes
No
count = count + 1
count == 0
Yes
No
End
Read character from memory
Map character value to braille code using lookup table
Output braille code to Port C
count = count - 1

7 Bill Of Materials

Qty	Value	Device
42		40-XX_Omron_Switch
1		8085
1		8155
10		C-EU025-025X050
2		CPOL-EUE2.5-5
1		CRYSTALHC49S
1		POWER_CON
13		R-US_0207/7
1	0.1u	C-EU025-025X050
1	100	R-US_0207/7
2	10k	R-US_0207/7
5	1k	R-US_0207/7
2	220	R-US_0207/7
1	22p	C-EU025-025X050
7	3.3k	R-US_0207/7
1	74LS138N	74LS138N
2	74LS373N	74LS373N
1	AT28C256-15PU	AT28C256-15PU
1	BC547	BC547
6	BD139	BD139
1	BS170	BS170
1	BUZZER	BUZZER
1	LCD-HD447804-HOLES20X4	LCD-HD44780
2	LED3MM	LED3MM
1	M74HC238	M74HC238
2	OMRON_SWITCH_10-XX	OMRON_SWITCH_10-XX
1	PRESET_LR	PRESET_LR
1	SLIDESWITCH-BIG	SLIDESWITCH-BIG

8 Future improvements and applications

Further additions to this device can include a USB interface to read a document from USB. An SD card could be used for the sam purpose. The electromagnetic coil box can be made to be wirelessly connected to a transmitter which transmits text documents, and the device would then translate the characters in the document to braille for the visually impaired to read upon.

The Braille translator provides an excellent opportunity for an e-solution to the paper problem, which is a small step in controlling the growing deforestation problem.

9 Video Link

https://https://www.youtube.com/watch?v=72datb2MRO0