Serial Communication Between Two 8085 Microprocessors

Using Keyboard and 7 Segment Displays Interfaced with 8255A PPI

Parija Malgaonkar
BT18ECE040
ECE 3rd Semester (BATCH 2)

Ayushi Tandon
BT18ECE036
ECE 3rd Semester (BATCH 2)

Laasya Cherukuri
BT18ECE037
ECE 3rd Semester (BATCH 2)

***Abstract*—This project report highlights the points and methodology involving the serial communication between two 8085 microprocessors, using programmable peripheral interfacing with an 8255A and matrix keyboards and 7 segment LCD displays.**

Keywords—microprocessor, interfacing, serial communication, programming

I. INTRODUCTION

The main aim of the project is to connect two 8085 microprocessors and send messages among them using 8255A PPI as an interfacing device, accompanied by 4x4 matrix keyboard and & 7 segment LCD displays. We will be using the SID and SOD pins on the 8085 to make the communication happen, along with the SIM (Set interrupt mask) and RIM (Read interrupt mask) instructions.
The input and output devices will be interfaced using the various ports available on the 8255A. The matrix keyboard and the 7-segment display each require 8 pins to be interfaced, hence we will be using all the ports. We will be using two 8085, hence we will require 2 of each of the components, that is, two 8255A, two matrix keyboards and two 7-segment displays. The set of instructions will be the same of both the microprocessors and will be given at the same time so that they work in sync.
All the components are thoroughly discussed in the next section, along with their use in this project

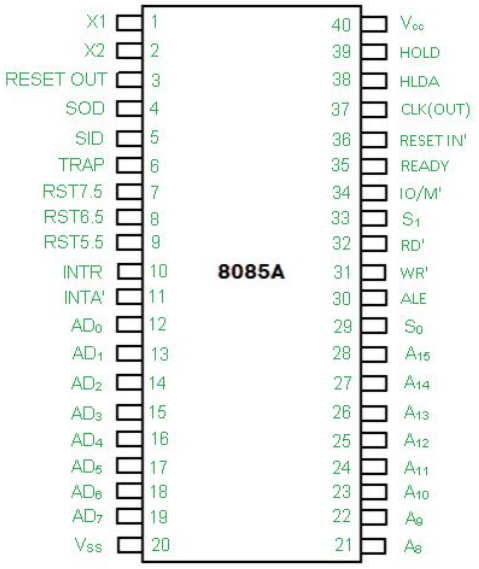
II. COMPONENTS USED

A. *8085 Microprocessor*

The 8085 microprocessors were invented by Intel Corporation in 1976. Its clock speed is 5 MHz, which is about one thousandth of the latest microprocessor such as Intel Core i7.
It has 80 instructions and 246 commands. **[9]** The 8-bit microprocessor is capable of addressing up to 64 K bytes (i.e. 216 = 65536 bytes) of memory.

It has the following configuration −

- 8-bit data bus
- 16-bit address bus, which can address up to 64KB
- A 16-bit program counter
- A 16-bit stack pointer
- Six 8-bit registers arranged in pairs: BC, DE, HL
- Requires +5V supply to operate at 3.3 MHZ single phase clock



The 8085 microprocessor has a certain number of functional units that carry out tasks such as storage and processing of the data. The functional units are **[1]**:

- **Register Section**: contains eight addressable 8-bit registers namely:
  (a) Accumulator (A register)
  (b) Flag register (PSW)
  (c) BC, DE, HL register pairs.
- **Arithmetic and Logic Unit (ALU):** ALU performs arithmetic operations like addition, subtraction, multiplication, division and logical operations like logical OR, logical AND, Exclusive OR, Exclusive NOR, complement, Clear, increment, decrement etc**. (7)**

- **Timing and Control Section:** It provides timing and control signal to the microprocessor to perform operations **[11]**

- **Interrupt Control Section:** When a microprocessor is executing a main program and whenever an interrupt occurs, the microprocessor shifts the control from the main program to process the incoming request. After the request is completed, the control goes back to the main program.

- **Serial Input / Output Control:** Serial input/output control circuit is used for data transmission. SID and SOD are two pins of this circuit **(1)**

    - *SOD (Serial Output Data):*

    SOD takes 1 bit from Accumulator to serial port of 8085.It takes bit from MSB (8th position) of the Accumulator and 7th bit of the Accumulator is transferred with the help of SIM (Set Interrupt Mask) instruction **[10]**

    - *SID (Serial Input Data):*

    SID takes 1-bit input from serial port of 8085 to Accumulator. It stores the bit at MSB (8th position) of the Accumulator and bit 7 is transferred with the help of RIM (Read Interrupt Mask) instruction **(10)**

We will be using the 8085 microprocessor to carry out the tasks on taking the input from the interfaced devices and give output to the output devices. The major parts that we will be using are the registers and the serial input and output functional units (SID & SOD)

B. 8255A PPI

The 8255A is a general purpose programmable I/O device designed to transfer the data from I/O to interrupt I/O under certain conditions as required. It can be used with almost any microprocessor.

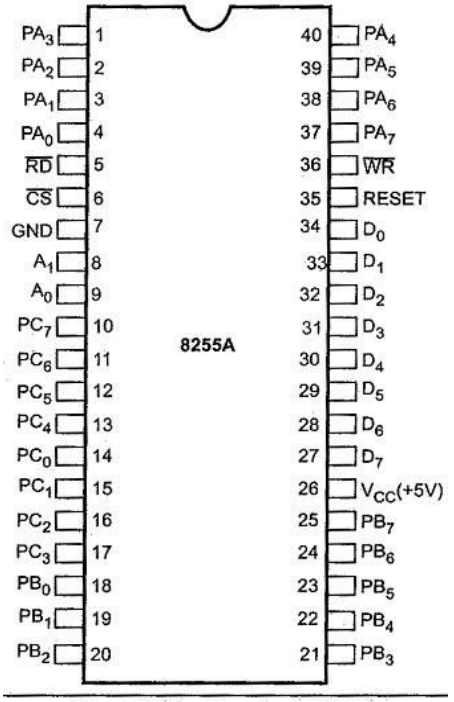


Fig. 14.1 Pin diagram of 8255A

It consists of three 8-bit bidirectional I/O ports (24I/O lines) which can be configured as per the requirement.

8255A has three ports, i.e., PORT A, PORT B, and PORT C.

- Port A contains one 8-bit output latch/buffer and one 8-bit input buffer.
- Port B is similar to PORT A.
- Port C can be split into two parts, i.e. PORT C lower (PC0-PC3) and PORT C upper (PC7-PC4) by the control word. **[11]**

There are 2 modes in 8255:

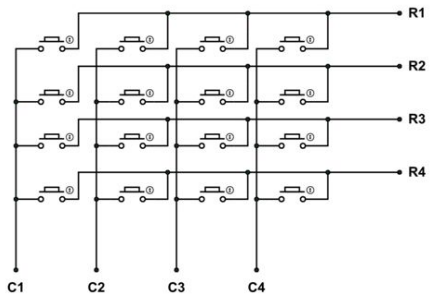
- 1. **Bit set reset (BSR) mode** – This mode is used to set or reset the bits of port C only, and selected when the most significant bit (D7) in the control register is 0. This mode affects only one bit of port C at a time because, as user set the bit, it remains set until and unless user changes it. User needs to load the bit pattern in control register to change the bit.
- **2. Input/output mode (I/O)** – This mode is selected when the most significant bit (D7) in the control register is 1.
    - ***Mode 0 – Simple or basic I/O mode*:** Port A, B and C can work either as input function or as output function. The outputs are latched but the inputs are not latched. It has interrupt handling capability.
    - ***Mode 1 – Handshake or strobed I/O:*** In this either port A or B can work and port C bits are used to provide handshaking. The outputs as well as inputs are latched
    - ***Mode 3 – Bidirectional I/O:*** In this mode only port A will work, port B can either is in mode 0 or 1 and port C bits are used as

handshake signal. The outputs as well as inputs are latched. **[13]**

We will be operating with **MODE 0** for our project. The devices that will be interfaced with the 8255 is the matrix keyboard as well as the 7-segment display. The keyboard requires 4 terminals for the rows and 4 terminals for the columns, hence we will be providing 4 pins from PORT B for the rows and 4 pins from PORT C (upper). The 7-segment display requires 8 terminals which we will give using the entire PORT A.

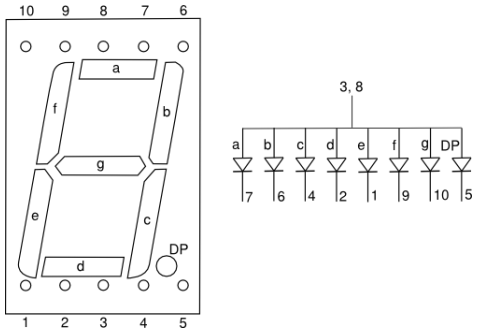
C. 4x4 Matrix Keyboard Input Device

Matrix keypads use a combination of four rows and four columns to provide button states to the host device. It has 8 terminals. These 8 pins are driven out from 16 buttons present in the module. Those 16 alphanumeric digits on the module surface are the 16 buttons arranged in matrix formation. .**[14]**

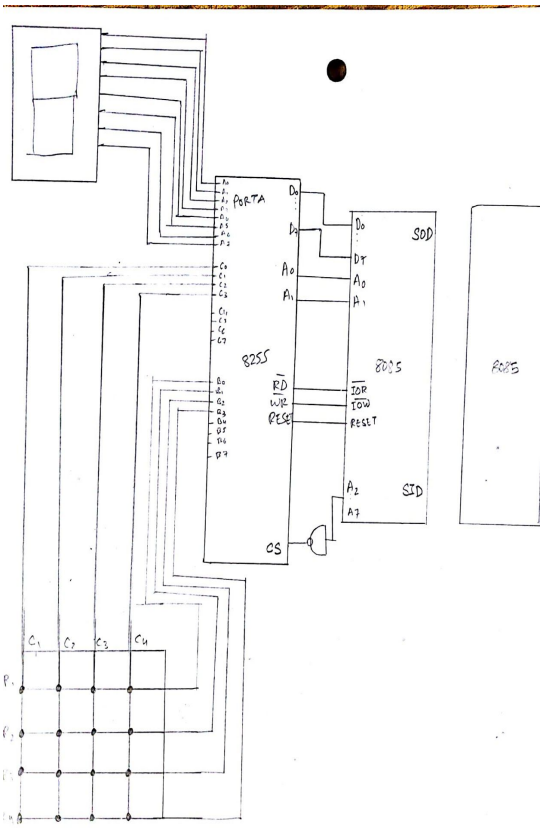


D. 7 - Segment Display

A 7 - Segment Display consists of seven LEDs (hence its name) arranged in a rectangular fashion as shown. Each of the seven LEDs is called a segment because when illuminated the segment forms part of a numerical digit (both Decimal and Hex) to be displayed. An additional 8th LED is sometimes used within the same package thus allowing the indication of a decimal point, (DP) when two or more 7-segment displays are connected together to display numbers greater than ten **[11]**



III. CIRCUIT DIAGRAM:



IV. CODES

MAIN:

```
LXI SP 0000H        (INITIALIZE STACK POINTER)
MVI A 88H           (INITIALIZE CWR LOCATED AT 83H)
OUT 83H             (OUTPUT T0 83H)
MVI A 00H           (00H -> A)
OUT 80H             (OUTPUT TO 80H)
MVI A CEH           (To set SIM Instruction)
SIM                 (Set interrupt RST 5.5)
EI                  (Enable interrupt)
MVI A 89H           (A<- 89H)
RIM                 (Read interrupts given by 89H)
ANI 08H             (AND IMMEDIATE DATA WITH A)
JZ LOOP1            (JUMP if Z=0)
CALL DISPLAY        (CALL DISPLAY FUNCTION)
LOOP1: IN 82H       (INPUT FROM 82H)
ANI C0H             (AND IMMEDIATE DATA WITH A)
JZ LOOP1            (JUMP IF Z=1)
CALL DELAY          (CALL DELAY FUNCTION)
CALL RDKBD          (CALL RDKBD FUNCTION)
MVI A 00H           (00H -> A)
OUT 80H             (OUTPUT TO 80H)
LOOP2: IN 82H       (INPUT FROM 82H)
ANI C0H             (AND IMMEDIATE DATA FROM A)
```

```
JNZ LOOP2          (JUMP IF Z=!0)
CALL DELAY         (CALL DELAY FUNCTION)
JMP LOOP1          (JUMP TO LOOP1)
```

RDKBD:

```
LXI H 1510H        (INITIALIZE REGISTER PAIR)
PUSH H             (PUSH VALUE IN SP)
LXI H 1500H        (INITIALIZE REGISTER PAIR)

MVI B 77H          (B <- 77H)
MVI C 04H          (C <- 04H)
L1: MOV A, B
OUT 81H            (OUTPUT TO 81H)
RRC                (ROTATE RIGHT WITHOUT CARRY)
MOV B, A
MVI D 04H
IN 82H             (INPUT FROM 82H)
L2 : RLC
JC DOWN            (JUMP IF CARRY)
MOV E,M
XTHL               (Exchange Top of stack with REGISTER PAIR)
MOV M, E
DCX H              (DECREMENT H)
XTHL               (Exchange Top of stack with REGISTER PAIR)
DOWN:INX H         (INCREMENT H)
DCR D              (DECREMENT D)
JNZ L2             (JUMP IF Z! =0)
DCR C              (DECREMENT C)
JNZ L1             (JUMP IF Z! =0)
POP H              (POP VALUE OF STACK TO REGISTER)
RET
```

Vectored address 002C H (RST 5.5)

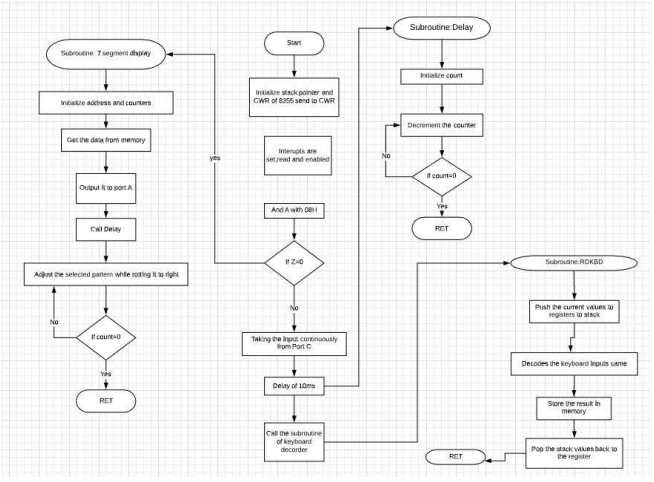
```
LXI H, A200H       (INITIALIZE REGISTER PAIR)
CALL DISPLAY       (CALL DISPLAY FUNCTION)
MOV A, H
SIM                (Set interrupt RST 5.5)
EI                 (Enable interrupt)
RET
```

DISPLAY:

```
DISPLAY: MVI B, 04H      (Load count)
MVI C, 77H :             (Load select pattern
LXI H, A200H :           (Starting address of message)
DISP 1: MOV A, C
OUT 80H                  (OUTPUT TO 80H)
MOV A, M :               (Get data)
OUT P:                   (Display data)
CALL DELAY :             (CALL DELAY FUNCTION)
DISP 1: MOV A, C
RRC                      (ROTATE RIGHT W/O CARRY)
MOV C, A :               (ADJUST SELECTION PATTERN)
INX H
DCR B :                  (DECREMENT B)
JNZ DISP 1
RET
Delay: LXI D, Count :
Back: DCX D              (DECREMENT D)
MOV A, D
ORA E                    (OR contents of E with the A)
JNZ Back                 (JUMP IF Z! =0)
RET
```

DELAY:

```
DELAY : MVI A 0A H
INNER: MVI B D5H
D1 : DCR B               (DECREMENT REGISTER)
JNZ D1                   (JUMP IF Z! =0)
DCR A                    (DECREMENT REGISTER)
JNZ INNER                (JUMP IF Z! =0)
RET
```



V. RESULT:

Hence, from the aforementioned code and comments, we see the entire process of sending the input of on device into one 8085 to the output device of another 8085 connected to the previous 8085 using the serial input and output functional unit. We also successfully interfaced and input device, a matrix keyboard and an output device, a 7-segment display, using an 8255A programmable peripheral interface device.

VI. CONCLUSION:

Thus, we conclude that serial communication is possible and was done using two 8085 microprocessors, 8255A PPI and input/output peripheral devices.

REFERENCES

[1]. Kaushik D.K; An Introduction to Microprocessor 8085,Dhanpat Rai Publishing Company.2010

[2]. Durga Devi R, Uma S, Mirunalinie S. A survey on Embedded System; International Journal of Engineering Research 3 (2), 2015, 5

- 8

[4]. http://scanftree.com/microprocessor/Evolution-and-Classification-of-Microprocessors

[5]. http://www.intel.com/pressroom/kits/quickrefyr.htm#1976

[6]. Yadav J, Bhatia K & Kaushil K; Microprocessor, IJRIT International Journal of Research in Information Technology, Volume 2,

Issue 10, October 2014, Pg. 14-18

[7]. RAM.B; Fundamentals of Microprocessor and Microcomputers; DhanpatRai Publications. 8

th Edition, 2012

[8]. Kathuria K, Malhotra B, RaiK; 8085 Microprocessors,IJRIT International Journal of Research in Information Technology, Volume

2, Issue 10, October 2014, Pg. 258-263

[9]. http://www.daenotes.com/electronics/digital-electronics/Intel-8085-8-bit-microprocessor

[10]. Tatla, G. S. Pin Diagram of 8085.

[11] Tutorials Point

[12] Over view of Microprocessor 8085 and its application - Kimasha Borah, Assistant Professor,

Centre for Computer Studies, Dibrugarh University, Dibrugarh, Assam, Indi

[13] Geeks for Geeks

[14] Parallax.com

.