Chapter 6

The Programmable Keyboard/Display Interface (8279)
Outline

- **Introduction**
  - General Features

- **Functional Block diagram**

- **Operation Modes**
  - **Input mode**
    - Scanned Keyboard mode
    - Scanned Sensor Matrix mode
    - Strobed Input mode
  - **Output mode**
    - Left Entry (typewriter type)
    - Right Entry (calculator type)

- **Programming the 8279**
Introduction

- The disadvantages of the s/w method of interfacing keyboard and display with 8085 is that the processor has to refresh the display and scan the status of the keyboard periodically using polling technique.
- Thus a considerable amount of CPU time is wasted, reducing the system operating speed.
- The 8279 is a hardware approach to interfacing a matrix keyboard and a multiplexed display.
- It is a 40 pin chip, and requires +5vdc supply for its operation.
Intel’s 8279 is a general purpose Keyboard Display controller that simultaneously drives the display of a system and interfaces a Keyboard with the CPU.

The Keyboard Display interface scans the Keyboard to identify if any key has been pressed and sends the code of the pressed key to the CPU. It also transmits the data received from the CPU, to the display device.
contd.

**Keyboard segment**
Scans the keyboard, detects key press, and transmits to CPU the characteristic of key

**Display segment**
It puts data from the CPU to display devices

8279
contd.

**Keyboard Segment**
- Connected to a 64 contact key matrix
- Keyboard entries and debounced and stored in FIFO
- Interrupt signal is generated with each entry

**Display Segment**
- 16 character scanned display
- 16x8 R/W memory (RAM)
- Right entry or left entry
The keyboard display controller chip 8279 provides:

a) a set of four scan lines and eight return lines for interfacing keyboards

b) A set of eight output lines for interfacing display.

The Keyboard is interfaced either in the interrupt or the polled mode.

In the interrupt mode, the processor is requested service only if any key is pressed, otherwise the CPU can proceed with its main task.

In the polled mode, the CPU periodically reads an internal flag of 8279 to check for a key pressure.
The Keyboard section can interface an array of a maximum of 64 keys with the CPU.

The Keyboard entries (key codes) are debounced and stored in an 8-byte FIFO RAM, that is further accessed by the CPU to read the key codes.

If more than eight characters are entered in the FIFO (i.e. more that eight keys are pressed), before any FIFO read operation, the overrun status is set.
If a FIFO contains a valid key entry, the CPU is interrupted (in interrupt mode) or the CPU checks the status (in polling) to read the entry.

Once the CPU reads a key entry, the FIFO is updated, i.e. the key entry is pushed out of the FIFO to generate space for new entries.

The 8279 normally provides a maximum of sixteen 7-seg display interface with CPU. It contains a 16-byte display RAM that can be used either as an integrated block of 16x8-bits or two 16x4-bit block of RAM.

The data entry to RAM block is controlled by CPU using the command words of the 8279.
Features of 8279

- It is designed by Intel
- Simultaneous keyboard and display operation.
- It supports a 64 contact key matrix with two more keys: "CONTROL" and "SHIFT".
- Has 3 input operating modes for keyboard interface:
  1. Scanned keyboard mode
  2. Scanned sensor matrix mode
  3. Strobed Input mode.
- It has an inbuilt debounce key.
- It provides two output modes:
  1. Left entry (Typewriter type).
  2. Right entry (Calculator type).
**Block Diagram of 8279**

**Four major sections:**
1. Keyboard
2. Display
3. Scan
4. MPU interface
Pin Description

- **DB7-DB0**: These are bidirectional data bus lines. The data and command words to and from the CPU are transferred on these lines.

- **CLK**: Clock from system used to generate internal timing.

- **RESET**: This pin is used to reset 8279. A high on this line resets 8279.

- **CS chip select**: A low on this line enables 8279 for normal read or write operations. Otherwise this pin should be high.
Pin Description

- **Ao**: A high on the Ao line indicates the transfer of a command or status information. A low on this line indicates the transfer of data. This is used to select one of the internal registers of 8279.

**RD**: when low, the CPU can read from selected register(-display RAM, FIFO status, or FIFO RAM)

**WR**: when low, the CPU can write command/data into selected register(control register, Display RAM)
Pin Description

- **IRQ**: This interrupt output line goes **high** when there is data in the FIFO RAM.
- The interrupt line goes **low** with each FIFO RAM read operation.
- However, if the FIFO RAM further contains any Key-code entry to be read by the CPU, this pin again goes **high** to generate an interrupt to the CPU.
- **Vss, Vcc**: These are the ground and power supply lines for the circuit.
**Pin Description**

- **SL0-SL3 – Scan Lines:** These output lines are used to scan the keyboard matrix and display digits. These lines can be programmed as encoded or decoded, using the mode control register.

- **RL7-RL0 (Return Lines):** Inputs used to sense key depression in the keyboard matrix.

- **CNTL/STB (Control/Strobe):** Connected to the control key on the keyboard.

- **SHIFT:** Shift connects to Shift key on keyboard.
**Pin Description**

- **BD – Blank Display**: This output pin is used to blank the display during digit switching or by a blanking command.

- **OUT A3-A0**: Output port that sends data to the most significant nibble of display.

- **OUT B3-B0**: Output port that sends data to the least significant nibble of display.

These two ports may also be considered as one 8-bit port.

<table>
<thead>
<tr>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>A3</td>
<td>A2</td>
<td>A1</td>
<td>A0</td>
<td>B3</td>
<td>B2</td>
<td>B1</td>
<td>B0</td>
</tr>
</tbody>
</table>
contd.

- It consists 4 main section.
  1. CPU interface and control section.
  2. Scan section
  3. Keyboard Section
  4. Display section.
It includes 8 bidirectional data lines ($DB0-DB7$, $IRQ$, buffer address line ($A0$), and other five lines ($RD$, $WR$, $CS$, $RESET$, $CLK$) for interfacing.

It consists of
1. Data buffers
2. I/O control
3. Control and timing registers.
4. Timing and control logic.

Data Buffers:

- 8-bit bidirectional buffer.
- Used to connect the internal data bus and external data bus.
contd.

- The I/O control section controls the flow of data to/from the 8279.
- The I/O section is enabled only if CS is low.
- The pins A0, RD and WR select the command, status or data read/write operations carried out by the CPU with 8279.
- The data buffers interface the external bus of the system with internal bus of 8279.
**I/O control:**

- I/O control section uses the A0, CS, RD and WR signals to control the data flow.
- The data flow is enabled by CS=0 otherwise it is the high impedance state.
- A0=0 means the data is transferred.
- A0=1 means status or command word is transferred.

<table>
<thead>
<tr>
<th>A0</th>
<th>RD</th>
<th>WR</th>
<th>Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>Data from CPU to 8279</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>Data from 8279 to CPU</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>Command word from CPU to 8279</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>Status word from 8279 to CPU</td>
</tr>
</tbody>
</table>
These registers store the keyboard and display modes and other operating conditions programmed by CPU.

The registers are written with A0=1 and WR=0. The Timing and control unit controls the basic timings for the operation of the circuit.

Scan counter divide down the operating frequency of 8279 to derive scan keyboard and scan display frequencies.
Scan section

- The Scan Counter has two modes to scan the key matrix and refresh the display.
  1. Encoded mode
  2. Decoded mode.

- In the Encoded mode, the counter provides a binary count that is to be externally decoded to provide the scan lines for keyboard and display. i.e. In the **encoded mode**, the **scan counter** provides a **binary count** from **0000 to 1111** on the four scan lines (**SL3-SL0**). This binary count must be **externally decoded** to provide **16 scan lines**. Display can use all 16 scan lines to interface 16 digit 7-segment display, but keyboard can use only 8 scan lines out of 16 scan lines.
Scan section

- In the **decoded mode**, the internal decoder decodes the least significant 2 bits of binary count and provides four possible combinations on the scan lines (SL3-SL0).

- These lines can be used directly to interface 4 digit 7-segment display, 8x4 matrix keyboard.

- In general, SL3-SL0 can be connected to the rows of a matrix keyboard and the digit drivers of a multiplexed display.
Keyboard section

- This section has keyboard debounce and control, 8x8 FIFO/Sensor RAM, 8 Return lines (RL0 – RL7) and CNTL/STB and shift lines. i.e.
  - This is consist of,
    - Return buffers.
    - Keyboard debounce control.
    - FIFO / sensor RAM.
    - FIFO / sensor RAM status.

Return Buffers and Keyboard De-bounce and Control:
- This section scans for a key closure row wise. If a key closer is detected, the keyboard debounce unit debounces the key entry (i.e. wait for 10 ms).
- After the debounce period, if the key continues to be detected, The code of key is directly transferred to the sensor RAM along with SHIFT and CONTROL key status.
contd. KEYBOARD DEBOUNCE AND CONTROL

- It is enabled only when keyboard mode is selected.
- In this mode, return lines are scanned whether any keys are closed in the row.
- If debounce circuit is detect any closed switch it wait about 10 msec. i.e. If it is detected, the Keyboard debounce unit debounces the key entry (i.e. wait for 10 ms).
- It is continued, the status of SHIFT and CONTROL keys are transferred into RAM.
In the keyboard debounce and control unit, keys are automatically debounced and the keyboard can be operated in two modes.

1. Two key lock out
2. N-key roll over

In the two key lockout mode, if two keys are pressed almost simultaneously, only the first key is recognized if all other keys are released before the first one. i.e.

If two keys are pressed within a debounce cycle (simultaneously), no key is recognized till one of them remains closed and the other is released. The last key, that remains depressed is considered as single valid key depression.
2. N - key roll over

- In the N-key rollover mode, simultaneous keys are recognized and these codes are stored in the internal buffer.
Scanned Keyboard with N-Key Rollover: In this mode, each key depression is treated independently.

When a key is pressed, the debounce circuit waits for 2 keyboards scans and then checks whether the key is still depressed. If it is still depressed, the code is entered in FIFO RAM.

Any number of keys can be pressed simultaneously and recognized in the order, the keyboard scan recorded them. All the codes of such keys are entered into FIFO.

In this mode, the first pressed key need not be released before the second is pressed.

All the keys are sensed in the order of their depression, rather in the order the keyboard scan senses them, and independent of the order of their release.
contd.

**FIFO/SENSOR RAM:**

- In Keyboard or strobed input mode, this block acts as 8-byte first-in-first-out (FIFO) RAM.
- Each key code of the pressed key is entered in the order of the entry, and in the meantime, read by the CPU, till the RAM becomes empty.
- The **FIFO/Sensor RAM**—consists of 8x8 registers that can store eight keyboard entries; each is then read in the order of entries.
FIFO/SENSOR RAM Status:

- This is used to tell the status of FIFO/SENSOR RAM.
- The status of logic also makes IRQ signal is High, when FIFO is empty.
- The **FIFO/Sensor RAM status** - keeps track of the number of entries and provides an **IRQ** (interrupt request) signal when the FIFO is not empty.
Display section

- It consists of,

1. Display RAM.
2. Display Address registers.
3. Display registers.

**DISPLAY RAM:**

- It is a 16*8 RAM.
- Which stores 16 digits display codes.
- It can be accessed by CPU directly. i.e. The MPU can write into or read from any of these display RAM registers.
- In Decoded mode, 8279 uses only first four location of Display RAM.
- In Encoded mode, 8279 uses all 16 location for 16 digits display.
contd. Display section

DISPLAY ADDRESS REGISTERS:

- Used to hold address of the byte currently write or read by the CPU and scan count value.
- In auto increment mode, address in the register is automatically incremented for each write or read.

DISPLAY REGISTERS:

- It is a Two 4-bit registers such as A and B.
- They hold the bit patterns of character to be displayed.
- The content of display registers A and B can be blanked and inhibited individually.
The OUTA3-A0 and OUTB3-B0 output ports – these are used for sending data to display drivers from Display RAM and can be connected to the segment input of 7 segment display or row inputs of dot matrix displays. The data from these outputs is synchronized to the scan lines (SL0-SL3) for multiplexed digit display (i.e., when value on SL3-SL0 is 0000, ports will have data from register 0 of Display RAM, and so on.

The BD line is used to blank the display or the two 4-bit ports can be blanked independently.
Operation modes of 8279

- It is two types,
  1. Input modes.
  2. Display modes.

INPUT MODES:

- It is basically 3 types,
  1. Scanned keyboard.
  2. Scanned sensor matrix.
  3. Strobed mode.
contd.

- Scanned keyboard mode
- Scanned sensor matrix mode
- Strobed input mode

Operating modes of 8279

- Display scan
- Display entry

INPUT (keyboard) MODES

OUTPUT (display) MODES
Scanned Keyboard Mode

- This mode allows a key matrix to be interfaced using either encoded or decoded scans. i.e.
- Keyboard can be scanned in two ways.
  1. Encoded Scan  2. Decoded Scan.

Encoded Scan:

- In this scan, scan lines (SL2-SL0) are decoded externally to provide 8 scan lines.
- Additionally it provides 8 return lines.
- So the size of matrix keyboard is 8*8 (i.e Scan * Return)=64.
- When the key is pressed, it is stored the status of return lines, Scan lines, SHIFT and CNTL/STB keys into FIFO RAM.
- The Scanned keyboard structure is,
Decoded scan – least significant 2 bits of scan lines are decode internally to produce four scan lines (SL3-SL0) and used directly, therefore the maximum size of keyboard is $4\times 8 = 32$.

The key code is similar to encoded code, only bit 5 (B5) is always zero.
Scanning keys of a Matrix Keyboard

contd.
Scanned Sensor Matrix:

- In this mode, a sensor array can be interfaced with 8279 using either encoder or decoder scans.
  - In this mode, image of the sensor matrix is kept in the sensor RAM.
  - The status of sensor switches are input directly to the sensor RAM.
  - 8279 scans row one by one and store the status of each row in the corresponding memory location.
  - the debounce logic is inhibited.
Strobed Input:

- In this mode, if the control line goes low, the data on return lines, is stored in the FIFO byte by byte.
- Data on return lines are transferred to FIFO on rising edge of CNTL/STB.
Output (Display) mode:

- 8279 provides two output modes for selecting the display options.

  1. Display Scan:
  
  2. Display Entry:

  **Display scan:** 8 or 16 character multiplexed displays that can be organized as dual 4-bit or single 8-bit.

  **Display Entry format:**- Right entry or Left entry display formats.
Output (Display) Modes: 8279 provides two output modes for selecting the display options.

Display Entry
( right entry or left entry mode )
- **8279** allows options for data entry on the displays.
- The display data is entered for display either from the right side or from the left side.

Display Scan:
- In this mode 8279 provides 8 or 16 character multiplexed displays those can be organized as dual 4-bit or single 8-bit display units.

**contd.**
Display format

- **Left entry**

  . each display position directly corresponds to a byte (or nibble) position in the display RAM.

  . the 8279 displays characters from left to right in the multiplexed displays like a typewriter.

**AUTOINCREMENT IN LEFT ENTRY:**

- In left entry mode, Auto increment flag is set after each operation display RAM address is incremented.

- **Right entry**

  . the 8279 displays characters from right to left in the multiplexed display like a calculator.

**AUTOINCREMENT IN RIGHT ENTRY:**

- In right entry mode, Auto increment flag is set after each operation display RAM address is incremented.
Left Entry Mode (TYPE WRITER)
- In the left entry mode, the data is entered from left side of the display unit.
- **Address 0** of the display RAM contains the leftmost display characters and **address 15** of the RAM contains the right most display characters.
- It is just like writing in our note books i.e. left to right.
- In **autoincrement mode**, address is automatically updated with successive reader or writes.
- The first entry is displayed on the leftmost display and the sixteenth entry on the rightmost display.
- The **seventeenth entry** is again displayed at the leftmost display position.

Right Entry Mode (CALCULATOR)
- In this right entry mode, the first entry to be displayed is entered on the rightmost display.
- The next entry is also placed in the right most display but after the **previous display is shifted** left by one display position.
- The leftmost characters is shifted out of that display at the **seventeenth entry and is lost**, i.e. it is pushed out of the display RAM.
The 8279 can be programmed to perform various functions through eight command words.

1. Keyboard/Display Mode Set
2. Program Clock
3. Read FIFO/Sensor RAM
4. Read Display RAM
5. Write Display RAM
6. Display Write Inhibit/Blanking
7. Clear
8. End Interrupt/Error Mode Set

All the command words or status words are written or read with $A0=1$ and $CS=0$ to or from 8279.
The format of the command word to select different modes of operation of 8279 is given below with its bit definitions.

<table>
<thead>
<tr>
<th>D</th>
<th>D</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>D</th>
<th>D</th>
<th>K</th>
<th>K</th>
<th>K</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Eight No. of 8-bit character display</td>
<td>Left entry</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Sixteen No. of 8-bit character display</td>
<td>Left entry</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Eight No. of 8-bit character display</td>
<td>Right entry</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Sixteen No. of 8-bit character display</td>
<td>Right entry</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>K</th>
<th>K</th>
<th>K</th>
<th>K</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Encoded Scan Keyboard - 2-Key Lockout</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>Decoded Scan Keyboard - 2-Key Lockout</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>Encoded Scan Keyboard - N-Key Rollover</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>Decoded Scan keyboard - N-Key Rollover</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Encoded Scan Sensor Matrix</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>Decoded Scan Sensor Matrix</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>Strobed Input, Encoded Display Scan</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>Strobed Input, Decoded Display Scan</td>
</tr>
</tbody>
</table>
Program Clock

- All timing and multiplexing signals for the 8279 are generated by an internal prescaler.

- The clock for operation of 8279 is obtained by dividing the external clock input signal by a programmable constant called prescaler.

- This prescaler divides the external clock (pin 3) by a programmable integer, \( N \).

- bits PPPPP determine the value of this integer which ranges from 2 to 31.

\[
\text{Prescaler, } N = \frac{\text{external clock}}{100\text{KHz}}
\]

<table>
<thead>
<tr>
<th>Code</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
</tr>
</tbody>
</table>
Example

- If pin 3 is clocked by 2MHz, then PPPPP is set to 10100 to divide the clock by \( N=20 \), to yield the proper 100kHz operating frequency.

i.e. \( N=2MHz/100kHz=20 \)
Exercise

- Find the control word to operate 8279 in 100 KHz if the processor (system) clock frequency is 1 MHz.
The CPU sets up the 8279 for a read of the FIFO/Sensor RAM by first writing this command.

In the Scan Keyboard Mode, the Auto-Increment flag (AI), and the RAM address bits (AAA) are irrelevant. The 8279 will automatically drive the data bus for each subsequent read (A0=0) in the same sequence in which the first entered the FIFO. All subsequent reads will be from the FIFO until another command is issued.

In the Sensor Matrix Mode, AAA select one of the 8 rows of the sensor RAM. If the AI flag is set (AI=1), each successive read will be from the subsequent row of the sensor RAM. (i.e. to increase the address after reading a byte from RAM.)
Read Display RAM

- This command word enables a programmer to read the content of display RAM. (16 bytes)
- The CPU sets up the 8279 for a read of the Display RAM by first writing this command.
- The CPU writes this command word to 8279 to prepare it for display RAM read operation.

- The address bits **AAAA** select one of the 16 rows of the Display RAM. If the AI flag is set (**AI=1**), this row address will be incremented after each following read from the Display RAM.
Example

- Write a command word to read fourth location with auto increment of the display RAM.
- Solution command word

```
0 1 1 1 0 1 0 0 =74H
```
Exercise

1. Write a command word to read seventh location with auto increment of the display RAM.
2. Write a command word to read third location without auto increment of the display RAM.
Write Display RAM

- This command enables a programmer to write the display RAM data. i.e. to write data into display RAM, it is necessary to set 8279 in write display RAM mode.
- The CPU sets up the 8279 for a write to the Display RAM by first writing this command.

```
1 0 0 AI A A A A
```

- After writing this command with A0=1, all subsequent writes with A0=0 will be to the Display RAM. If the AI flag is set (AI=1), this row address will be auto-incremented after each following write to the Display RAM.
- AAAAA – Selects one of the 16 rows of the Display RAM.
Example

- Write a command word to write fifth location with auto increment of the display RAM.

- Solution command word

```
1 0 0 1 0 1 0 1
```

=95H
Exercise

1. Write a command word to write seventh location with auto increment of the display RAM.

2. Write a command word to write fourth location without auto increment of the display RAM.
Display Write Inhibit/Blanking

- We know that display RAM data is sent on the two 4-bit ports (B3-B0 and A3-A0). This two 4-bit ports can be individually inhibited or blanked with display write inhibit/blanking command.
- The IW (Inhibit write flag) bits are used to mask the individual nibble.

- The IW bits can be used to mask nibble A and nibble B in applications requiring separate 4-bit display ports.

If IW=1 for one of the ports, the port becomes masked so that entries to the Display RAM from the CPU do not affect that port.
The **BL bits** are used to blank each 4-bit display ports. Both BL flags must be set to blank a display formatted with a *single 8-bit port*. This code defaults to all zeros after a reset. The Clear command determines the code to be used as a “blank”.

Here Do and D2 corresponds to OUTBo – OUTB3 while D1 and D3 corresponds to OUTA0-OUTA3 for blanking and masking respectively.
The IW (inhibit write flag) bits are used to mask the individual nibble.
The output lines are divided into two nibbles (OUTA0 – OUTA3) and (OUTB0 – OUTB3), those can be masked by setting the corresponding IW bit to 1.
Once a nibble is masked by setting the corresponding IW bit to 1, the entry to display RAM does not affect the nibble even though it may change the unmasked nibble.
The blank display bit flags (BL) are used for blanking A and B nibbles.
D0, D2 corresponds to OUTB0 – OUTB3
D1, D3 corresponds to OUTA0 - OUTA3 for blanking and masking.
If the user wants to clear the display, blank (BL) bits are available for each nibble as shown in format.
Both BL bits will have to be cleared for blanking both the nibbles.
Example

- Write a command word to inhibit nibble A of the display

- Solution command word

```
  1 0 1 x 1 1 0 0 0
```

=A8H

Exercise

Write a command word to blank nibble B of the display
Clear Display

- The **CD** bits are used to clear all rows of the Display RAM to a selectable *blanking code* as follows:

<table>
<thead>
<tr>
<th>Code</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 1 0</td>
<td>CD</td>
<td>CD</td>
<td>CD</td>
<td>CF</td>
<td>CA</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **CD**
  - 0 x: All Zeros (for common cathode displays)
  - 1 0: AB = Hex 20 (0010 0000) alphanumeric display
  - 1 1: All Ones (for common anode displays)
- Enable clear display when = 1 (or by \( C_A = 1 \))
contd.

<table>
<thead>
<tr>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>CD2</td>
<td>CD1</td>
<td>CD0</td>
<td>CF</td>
<td>CA</td>
</tr>
</tbody>
</table>

- **CD2 must be 1** for enabling the clear display command.
- If CD2 = 0, the clear display command is invoked by setting **CA(CLEAR ALL) =1** and maintaining CD1, CD0 bits exactly same as above.
- If **CF(CLEAR FIFO RAM STATUS) =1**, FIFO status is cleared and IRQ line is pulled down and the sensor RAM pointer is set to row 0.
- If **CA=1**, this combines the effect of CD and CF bits.

**0X** - All zeros (x don’t care) AB=00
**10** - A3-A0 = 2 (0010) and B3-B0=00 (0000)
**11** - All ones (AB =FF), i.e. clear RAM
the most significant bit of the FIFO status word is set during clearing the Display RAM.

- If CF=1, the FIFO status is cleared and the interrupt output line is reset. Also, the Sensor RAM pointer is set to row 0.

- CA (clear all bits) flag, has the combined effect of CD and CF; it uses the selected CD clearing on the Display RAM, and also clears FIFO status.
Example

- Write a command word to set blanking code for common anode display and to clear the FIFO status.

- Solution: Blanking code for common anode display is all ones and which can be set by writing CD1=1 and CD0=1

<table>
<thead>
<tr>
<th>1</th>
<th>1</th>
<th>0</th>
<th>1</th>
<th>1</th>
<th>1</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
</table>

=DEH
End Interrupt/Error Mode Set

- For sensor matrix mode (IRQ is used to read the key pressed).
- For the sensor matrix mode, this command lowers the IRQ line and enables further writing into the RAM. Otherwise, if a charge in sensor value is detected, IRQ goes high that inhibits writing in the sensor RAM.
- For the N-key rollover mode - if $E=1$, the chip will operate in the Scanned keyboard special Error mode.
- In the special error mode, if two keys are depressed during single debounce, the error flag in the FIFO status word is set.
Status word: - It is used in the keyboard and strobed input modes. It contains the FIFO status, error, and display unavailable signals.

- Overrun error occurs when the entry of another character into a full FIFO is attempted.

- Underrun error occurs when the CPU tries to read an empty FIFO.

- DU - During clear display or clear all command, display RAM is not available for user.

- S/E - In the sensor matrix, a S/E bit is set to indicate that at least one sensor closure indication is contained in the sensor RAM.

  - In the scanned keyboard special error mode, a S/E bit is set to indicate that a simultaneous multiple closure error has occurred.
**Data format in FIFO**

- **Scanned keyboard mode**: The character entered into the FIFO corresponds to the position of the switch in the keyboard plus the status of the CNTL and SHIFT. The position of the switch on the keyboard is determined by the values in scan counter and column counter.

  MSB  |  LSB
  CNTL |  SHIFT |  Scan |  Return

- **Sensor matrix mode**: The data on the return lines (RL7-RL0) is entered directly in the row of the Sensor RAM that corresponds to the row in the matrix being scanned. Each switch position maps directly to a Sensor RAM position.

  MSB  |  LSB
  RL7  |  RL6  |  RL5  |  RL4  |  RL3  |  RL2  |  RL1  |  RL0

- **Strobed Input mode**: The data in the return lines is entered into the FIFO RAM.

  MSB  |  LSB
  RL7  |  RL6  |  RL5  |  RL4  |  RL3  |  RL2  |  RL1  |  RL0
Example:

- Find the key code for given condition below:
  - CNTL/STB SHIFT keys are open.
  - The pressed keys are to scan lines 2 and return lines 4.

SOLUTION:

<table>
<thead>
<tr>
<th>B₇</th>
<th>B₆</th>
<th>B₅</th>
<th>B₄</th>
<th>B₃</th>
<th>B₂</th>
<th>B₁</th>
<th>B₀</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

- CNTL=1
- SHIFT=1
- Scan mode=010 (Scan line 2)
- Return mode=100 (Return line 4)
- Key code =D4 H
System Design
**Problem**

Program Statement: Interface an 8 × 8 matrix keyboard to 8085 through 8279 in 2-key lockout mode and write an assembly language program to read keycode of the pressed key. The external clock frequency is 2MHz. Use I/O mapped I/O technique.

Solution: The 8 × 8 matrix keyboard can be interfaced to 8085 through 8279 in two ways.

1. Without interrupt signal
2. With interrupt signal (Interrupt driven Input)

(Dont use any Interrupts)
contd.

Fig. — Interfacing of 8 × 8 matrix keyboard
Keyboard/Display command word:

To interface $8 \times 8$ matrix keyboard we need 8 scan lines and 8 return lines.
To get 8 scan lines We have to select encoded scan keyboard mode.

\[
\begin{array}{ccccccc}
0 & 0 & 0 & 0 & D & D & K & K & K \\
0 & 0 & 0 & 0 & X & X & 0 & 0 & 0 \\
\end{array} = 00H
\]

Note : $000 \rightarrow$ Encoded scan keyboard - 2 key lockout

Program clock command word:

External clock frequency is 2 MHz

\[
\therefore \text{Prescaler value} = \frac{2 \text{MHz}}{100 \text{kHz}}
\]

\[
= 20 = (10100)_2
\]

\[
\begin{array}{ccccccc}
P & P & P & P & P & P & P \\
0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\
\end{array} = 34H
\]

Read FIFO/Sensor RAM command word:

we want to read first entry from the FIFO RAM.

\[
\begin{array}{ccccccc}
\text{AI} & A & A & A & A \\
0 & 1 & 0 & 0 & X & 0 & 0 & 0 & 0 \\
\end{array} = 40H
\]
FLOWCHART

Start

Initialize keyboard / display mode of 8279

Initialize prescaler count

Read FIFO status word

Is no. of characters in FIFO = 0 ?

Yes

No

Read FIFO RAM

End

contd.
MVI A, 00H : Initialize keyboard/display
OUT 81H : in encoded scan keyboard-2 keylockout mode
MVI A, 34H
OUT 81H : Initialize prescaler count
BACK: IN 81H : Read FIFO status word
ANI 07H : Mask bit B3 to B7
JZ BACK : If 0, key is not pressed wait for key press else read FIFO RAM
MVI A, 40H : Initialize 8279 in read
OUT 81H : FIFO RAM mode
IN 80H : Read FIFO RAM (keycode)
HLT : Stop program execution.
**Program Statement:** Interface an $8 \times 8$ matrix keyboard to 8085 through 8279 in 2-key lockout mode and write an assembly language program to read keycode of the pressed key. The external clock frequency is 2MHz. Use I/O mapped I/O technique.

With interrupt signal (Interrupt driven Input)
Problem 4: Interface 8 7-segment digits (common cathode) to 8085 through 8279 and write an 8085 assembly language program to display 1 to 8 on the eight seven segment digits. External clock frequency is 3 MHz.
As shown in the above fig., eight display lines (B₀ - B₃ and A₀ - A₃) are buffered with the help of transistor and used to drive display digits. These buffered lines are connected in parallel to all display digits. S₀, S₁ and S₂ lines are decoded and decoded lines are used for selection of one of the eight digits.

To display 1 to 8 numbers on the eight 7-segment digits we have to load 7-segment codes for 1 to 8 numbers in the corresponding display locations.

7-Segment codes for common cathode display

<table>
<thead>
<tr>
<th>Number</th>
<th>h</th>
<th>g</th>
<th>f</th>
<th>e</th>
<th>d</th>
<th>c</th>
<th>b</th>
<th>a</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>06</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>5B</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>4F</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>66</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>6D</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>7D</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>07</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>7F</td>
</tr>
</tbody>
</table>
Keyboard/Display command word:

To interface 8 digit 7 segment display we need 8/8-bit character display mode with left entry. For selection of 8 digits we need encoded scan mode.

\[
\begin{array}{ccccccc}
D & D & K & K & K & K & \text{= 00H} \\
0 & 0 & 0 & 0 & 0 & 0 & 0
\end{array}
\]

Program clock command word:

External clock frequency is 3MHz.
\[
\therefore \text{Prescaler value} = \frac{3\text{MHz}}{100\text{MHz}} = 30 = (11110)_2
\]

\[
\begin{array}{ccccccc}
P & P & P & P & P & P & \text{= 3EH} \\
0 & 0 & 1 & 1 & 1 & 1 & 0
\end{array}
\]

Write Display RAM command word:

We want to write first eight locations of display RAM with corresponding 7 segment codes. So we start from first location with autoincrement mode.

\[
\begin{array}{ccccccc}
A_1 & A_3 & A_2 & A_1 & A_0 & \text{= 90H} \\
1 & 0 & 0 & 1 & 0 & 0 & 0
\end{array}
\]
LOOK UP TABLE

<table>
<thead>
<tr>
<th>Memory Address</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>6200</td>
<td>66</td>
</tr>
<tr>
<td>6201</td>
<td>5B</td>
</tr>
<tr>
<td>6202</td>
<td>4F</td>
</tr>
<tr>
<td>6203</td>
<td>66</td>
</tr>
<tr>
<td>6204</td>
<td>6D</td>
</tr>
<tr>
<td>6205</td>
<td>7D</td>
</tr>
<tr>
<td>6206</td>
<td>07</td>
</tr>
<tr>
<td>6207</td>
<td>7F</td>
</tr>
</tbody>
</table>
Source program:

LXI B, 6200H
MVI C, 08H
MVI A, 00H
OUT 81H
MVI A, 3EH
OUT 81H
MVI A, 90H
OUT 81H
BACK : MOV A, M
OUT 80H
INX H
DCR C
JNZ BACK
HLT

: Initialize lookup table pointer
  : Initialize counter
    : Initialize keyboard/display
      : Mode
        : Initialize prescaler count
          : Initial size 8279 in write Display
            : RAM-mode
              : Get the 7-segment code
                : Write 7-segment code in display RAM
                  : Increment lookup table pointer
                    : Decrement counter
                      : if count = 0 stop, otherwise go to back
                        : Stop program execution