Here is the "Experiment 3: Parallel I/O Interfacing with 8085 (8255 PPI)" module designed in a practical file format.

PRACTICAL FILE

College/University Name: [Insert College/University Name Here]

Department: [Insert Department Name Here]

Course Title: Microcontroller Lab

Course Code: [Insert Course Code Here]

EXPERIMENT NO. 3

TITLE: Parallel I/O Interfacing with 8085 (8255 Programmable Peripheral Interface)

DATE PERFORMED: [DD/MM/YYYY - To be filled by student]

OBJECTIVE:

To understand the operation of the 8255 Programmable Peripheral Interface (PPI), learn its control word format and operating modes, interface it with the 8085 microprocessor, and write assembly programs to control LEDs (output) and read switch inputs (input) using 8255 in Mode 0.

SYLLABUS COVERAGE:

Interfacing with peripherals - parallel I/O.

---

1. THEORY / BACKGROUND

1.1 Introduction to Parallel Input/Output (I/O)

In microcomputer systems, Input/Output (I/O) operations are crucial for the CPU to interact with external devices, known as peripherals. Parallel I/O involves transmitting multiple bits of data simultaneously over separate lines. This contrasts with serial I/O, where data is sent bit by bit over a single line. Parallel I/O offers higher data transfer rates over shorter distances and is commonly used for interfacing with devices like printers, keyboards, displays, and sensors where multiple data lines are available.

To manage parallel I/O efficiently, microprocessors often use dedicated peripheral interface chips. The 8255 Programmable Peripheral Interface (PPI) is one such widely used general-purpose parallel I/O device.

1.2 Intel 8255 Programmable Peripheral Interface (PPI)

The 8255 is a versatile, programmable peripheral interface device designed to interface microprocessors with parallel I/O devices. It provides 24 programmable I/O

pins, which can be configured by software in various modes to suit different application requirements.

1.2.1 8255 Block Diagram and Features:

The 8255 PPI consists of the following functional blocks:

- **Data Bus Buffer: This is a tristate 8-bit bidirectional buffer that interfaces the 8255 with the system data bus (D0-D7 of 8085). It allows the CPU to read data from or write data to the 8255's internal registers (Port A, Port B, Port C, or Control Word Register).**
- **Read/Write Control Logic: This block manages the internal read and write operations. It accepts control signals from the microprocessor (RD, WR, A0, A1, CS) and generates appropriate internal control signals for the 8255's various functional units.**
- **Group A Control: This block controls the functionality of Port A and the upper 4 bits of Port C (PC4-PC7). It handles the configuration of these ports based on the control word written by the CPU.**
- **Group B Control: This block controls the functionality of Port B and the lower 4 bits of Port C (PC0-PC3). It configures these ports based on the control word.**
- **Port A (PA0-PA7): An 8-bit I/O port, programmable as either an 8-bit input or an 8-bit output. It can operate in Mode 0, Mode 1, or Mode 2.**
- **Port B (PB0-PB7): An 8-bit I/O port, programmable as either an 8-bit input or an 8-bit output. It can operate in Mode 0 or Mode 1.**
- **Port C (PC0-PC7): An 8-bit I/O port, which can be divided into two 4-bit nibbles:**
    - **Port C Upper (PC4-PC7): Controlled by Group A Control.**
    - **Port C Lower (PC0-PC3): Controlled by Group B Control.**
      Port C can be configured for input or output in Mode 0, or its bits can be individually set or reset in Bit Set/Reset (BSR) mode. It also serves for handshaking signals in Mode 1 and Mode 2.

1.2.2 8255 Pin Description (Key Pins for 8085 Interfacing):

- **D0-D7 (Data Bus): 8-bit bidirectional data lines for communication with the microprocessor.**
- **A0, A1 (Register Select): These are input pins connected to the microprocessor's lower address lines (A0, A1). Along with the Chip Select (CS) signal, they select one of the 8255's internal registers for CPU access:**
    - **A1 A0 | Selection**
    - **----- | ----------**
    - **0 0 | Port A**
    - **0 1 | Port B**
    - **1 0 | Port C**
    - **1 1 | Control Word Register**
- **CS (Chip Select): An active-low input. When CS is low, the 8255 is enabled for communication with the CPU. If high, the 8255 is disabled, and its data bus pins (D0-D7) are in a high-impedance state.**

- **RD (Read): An active-low input. When CS is low and RD is low, the 8085 reads data from the selected port or internal register.**
- **WR (Write): An active-low input. When CS is low and WR is low, the 8085 writes data to the selected port or internal register.**
- **RESET: An active-high input. When high, it clears all internal registers, sets all ports to input mode, and clears the control word. It should be connected to the 8085's RESET OUT.**
- **Vcc: +5V power supply.**
- **GND: Ground reference.**
- **PA0-PA7: 8-bit I/O lines for Port A.**
- **PB0-PB7: 8-bit I/O lines for Port B.**
- **PC0-PC7: 8-bit I/O lines for Port C.**

1.3 8255 Internal Addressing and Control Word Register

To communicate with the 8255, the 8085 needs to know its unique addresses for Port A, Port B, Port C, and the Control Word Register (CWR). These addresses are determined by the external address decoding logic that generates the CS signal for the 8255, combined with the 8255's internal A0 and A1 pins.

Let's assume the address decoding logic enables the 8255 for base address 80H.

Then, the I/O addresses for the 8255 would be:

- **Port A: 80H (A1=0, A0=0)**
- **Port B: 81H (A1=0, A0=1)**
- **Port C: 82H (A1=1, A0=0)**
- **Control Word Register (CWR): 83H (A1=1, A0=1)**

1.3.1 8255 Operating Modes:

The 8255 can operate in two primary modes:

1. **Bit Set/Reset (BSR) Mode: Only applicable to Port C. Allows individual bits of Port C to be set (high) or reset (low) without affecting other bits.**
2. **I/O Mode: Configures Ports A, B, and C as input or output ports. This mode has three sub-modes:**
    - **Mode 0 (Basic I/O): All ports (A, B, C) can be configured as simple latched outputs or buffered inputs. No handshaking signals are used. This is the simplest mode and most common for basic parallel I/O.**
    - **Mode 1 (Strobed I/O): Used for data transfer with handshaking signals. Port A and Port B use Port C lines for handshaking.**
    - **Mode 2 (Bidirectional I/O): Only Port A can be configured in this mode. It allows Port A to be used for both transmitting and receiving data simultaneously, with handshaking.**

1.3.2 Control Word Format for 8255:

The functionality of the 8255 is configured by writing a specific 8-bit Control Word to its Control Word Register (CWR). The format of this word depends on whether you are setting up I/O modes or using the Bit Set/Reset feature.

A. I/O Mode Set Control Word (D7 = 1):

This control word is written to address 83H (or corresponding CWR address) to configure the modes and directions of Ports A, B, and C.

```
D7 D6 D5 D4 D3 D2 D1 D0
-------------------------
 1  M1 M0 PA PCU M2 PB PCL
```

- **D7 (Mode Set Flag): Always `1` for I/O mode set.**
- **D6, D5 (Group A Mode Select - M1 M0):**
  - **`00` = Mode 0 (Basic I/O for Port A)**
  - **`01` = Mode 1 (Strobed I/O for Port A)**
  - **`1X` = Mode 2 (Bidirectional I/O for Port A)**
- **D4 (Port A Direction): `1` = Input, `0` = Output**
- **D3 (Port C Upper Direction - PC4-PC7): `1` = Input, `0` = Output**
- **D2 (Group B Mode Select - M2):**
  - **`0` = Mode 0 (Basic I/O for Port B)**
  - **`1` = Mode 1 (Strobed I/O for Port B)**
- **D1 (Port B Direction): `1` = Input, `0` = Output**
- **D0 (Port C Lower Direction - PC0-PC3): `1` = Input, `0` = Output**

Numerical Example for I/O Mode Control Word:

Let's configure the 8255 as follows:

- **Port A = Output (Mode 0)**
- **Port B = Output (Mode 0)**
- **Port C Lower (PC0-PC3) = Output (Mode 0)**
- **Port C Upper (PC4-PC7) = Input (Mode 0)**

Breaking down the bits for the Control Word:

- **D7 = 1 (I/O Mode Set)**
- **D6, D5 = 00 (Group A - Mode 0)**
- **D4 = 0 (Port A Output)**
- **D3 = 1 (Port C Upper Input)**
- **D2 = 0 (Group B - Mode 0)**
- **D1 = 0 (Port B Output)**
- **D0 = 0 (Port C Lower Output)**

Combining these bits: 1 00 0 1 0 0 0 (binary) = 10001000b = 88H (hexadecimal).

So, to achieve this configuration, you would write 88H to the Control Word Register (e.g., address 83H).

B. Bit Set/Reset (BSR) Mode Control Word (D7 = 0):

This control word is used to individually set or reset any one of the 8 bits of Port C. It is also written to the Control Word Register (e.g., address 83H).

```
D7 D6 D5 D4 D3 D2 D1 D0
-------------------------
 0  X  X  X  B2 B1 B0 S/R
```

- **D7 (Mode Set Flag): Always `0` for BSR mode.**
- **D6, D5, D4 (Don't Care): Value does not matter (often set to 0).**
- **D3, D2, D1 (Bit Select - B2 B1 B0): These three bits select which of the 8 bits of Port C (PC0-PC7) is to be set or reset.**
  - **000 = PC0, 001 = PC1, 010 = PC2, 011 = PC3,**
  - **100 = PC4, 101 = PC5, 110 = PC6, 111 = PC7**
- **D0 (Set/Reset): `1` = Set the selected bit, `0` = Reset the selected bit.**

Numerical Example for BSR Mode Control Word:

1. **To Set PC4 (make PC4 high):**
   - **D7 = 0**
   - **D6, D5, D4 = X (e.g., 000)**
   - **Bit Select (for PC4) = 100b**
   - **S/R = 1 (Set)**
   - **Control Word: `0 000 100 1` (binary) = `00001001b` = `09H`.**
2. **To Reset PC0 (make PC0 low):**
   - **D7 = 0**
   - **D6, D5, D4 = X (e.g., 000)**
   - **Bit Select (for PC0) = 000b**
   - **S/R = 0 (Reset)**
   - **Control Word: `0 000 000 0` (binary) = `00000000b` = `00H`.**

1.4 Interfacing 8255 with 8085

Connecting the 8255 to the 8085 involves connecting data lines, control lines, and address lines, along with address decoding logic.

- **Data Lines: The D0-D7 pins of 8255 are connected directly to the AD0-AD7 (multiplexed data lines) of 8085.**
- **Control Lines:**
  - **8255.RD is connected to 8085.RD.**
  - **8255.WR is connected to 8085.WR.**
  - **8255.RESET is connected to 8085.RESET OUT.**
- **Address Lines and Chip Select (CS):**
  - **8255.A0 is connected to 8085.A0 (from the de-multiplexed address bus).**

    - **8255.A1 is connected to 8085.A1 (from the de-multiplexed address bus).**
    - **The CS (Chip Select) pin of the 8255 needs to be activated (brought low) when the 8085 wants to communicate with the 8255. This is achieved using address decoding logic. This logic takes higher-order address lines from the 8085 (e.g., A2-A15) and potentially the IO/M signal, and generates the active-low CS signal for the 8255 based on a desired I/O address range.**

Example of Simple Address Decoding:

Let's assume we want to place the 8255's I/O ports at addresses 80H-83H.

- **This means, for addresses 80H, 81H, 82H, 83H:**
    - **A7 A6 A5 A4 A3 A2 A1 A0**
    - **1 0 0 0 0 0 A1 A0**
- **So, if we use a simple NAND gate for address decoding, we could connect A7 to one input, and A2, A3, A4, A5, A6 to other inputs (inverted if active high logic). A simpler common approach uses a 74LS138 (3-to-8 decoder) or a combination of logic gates. For example, if we decode the address 8Xh (where X is 0-F), we can use A7=1, A6=0, A5=0, A4=0, A3=0, A2=0, and combine them to generate an enable signal. The IO/M signal from 8085 should also be high for I/O operations.**

A common setup might use a NAND gate for simpler kits: if A7 is high and A2-A6 are low, and IO/M is high, then CS is active.

Simplified Interfacing Diagram Concept:

```
   8085 Microprocessor          |        8255 PPI
-------------------------------------------------------------------
AD0-AD7 (Data Bus) ----------------|---------- D0-D7 (Data Bus)
RD (Read) -------------------------|---------- RD
WR (Write) ------------------------|---------- WR
RESET OUT -------------------------|---------- RESET
A0 (De-multiplexed)----------------|---------- A0
A1 (De-multiplexed)----------------|---------- A1
A2-A15 (Address lines) ----> Address Decoding Logic --> CS (Chip Select)
IO/M (Control Signal) ----^
```

The specific implementation of address decoding depends on the trainer kit's design. Many kits already have an 8255 integrated with pre-decoded addresses (e.g., 80H-83H, or C0H-C3H).

---

2. APPARATUS / SOFTWARE REQUIRED

- **8085 Microprocessor Trainer Kit (with integrated 8255 PPI)**
  OR
  8085 Microprocessor Simulator Software (e.g., GNUSIM8085) with 8255 simulation capabilities.
- **DC Power Supply (for hardware kit)**
- **LEDs (8-16 LEDs, for output ports, if not integrated on kit)**
- **Push buttons/switches (8-16 switches, for input ports, if not integrated on kit)**
- **Connecting Wires (if external LEDs/switches are used)**
- **8085 Trainer Kit User Manual (for specific I/O addresses and commands)**
- **Personal Computer (for simulator use)**

---

3. PROCEDURE

This experiment uses an 8085 Microprocessor Trainer Kit with an integrated 8255 PPI. Ensure the kit is powered on and reset before starting. Assume the 8255 is mapped to I/O addresses 80H (Port A), 81H (Port B), 82H (Port C), and 83H (Control Word Register). Adjust addresses based on your specific trainer kit manual.

3.1 System Setup and 8255 Initialization

1. **Power on the 8085 trainer kit and press the `RESET` button.**
2. **Identify the I/O addresses of Port A, Port B, Port C, and the Control Word Register of the 8255 PPI on your trainer kit (e.g., 80H, 81H, 82H, 83H respectively).**
3. **Connect LEDs to the output port(s) you plan to use (e.g., Port A or Port B) if they are not integrated. Ensure proper current limiting resistors are used if wiring directly.**
4. **Connect switches to the input port(s) (e.g., Port B or Port C) if they are not integrated. Ensure pull-up/pull-down resistors are used as needed.**
5. **Calculate the appropriate 8255 Control Word (I/O Mode Set Word) based on the desired port configurations for each program (e.g., Port A Output, Port B Input, etc.).**
6. **Write the Control Word to the 8255's Control Word Register (CWR) using an 8085 assembly program. This is usually the first step in any 8255 application.**
   - **Example: `MVI A, 88H` (Load desired Control Word into Accumulator)**
   - **`OUT 83H` (Output Accumulator content to CWR at address 83H)**

3.2 Program Entry and Execution

1. **Enter Assembly Programs: Use the trainer kit's keypad and monitor program commands (e.g., `EXAM MEM` then input hex codes) to enter the assembly language programs provided in Section 4 into the designated memory locations (e.g., starting from 2000H).**
2. **Verify Program Entry: Before execution, you can re-examine the memory locations to ensure the hex codes were entered correctly.**
3. **Execute Program: Use the `GO` or `EXEC` command followed by the program's starting address (e.g., `GO 2000`).**

4. **Single-Stepping (Optional but Recommended): For initial debugging, use the `STEP` command to execute one instruction at a time and observe the effects, especially after `OUT` instructions to see if LEDs/outputs change.**

3.3 Observation

1. **Observe Output LEDs: For programs configuring ports as output, visually observe the state of the connected LEDs (ON/OFF) as the program executes.**
2. **Observe Input Switches: For programs configuring ports as input, change the state of the connected switches and then read the corresponding input port using the program. Verify the read data on the display or by checking register contents.**
3. **Examine Registers/Memory: Use the trainer kit's commands (`EXAM REG`, `EXAM MEM`) to verify the contents of the Accumulator and other registers, especially after `IN` instructions to confirm input data was correctly read.**

---

4. PROGRAMS TO BE EXECUTED

Assume 8255 I/O addresses: Port A = 80H, Port B = 81H, Port C = 82H, Control Word Register (CWR) = 83H.

4.1 Program 1: 8255 Initialization and Static Output to Port A

Objective: To configure 8255 for Mode 0 (Port A as Output, Port B as Input, Port C Lower as Output, Port C Upper as Input) and then output a fixed data byte (55H) to Port A.

Control Word Calculation (I/O Mode):

- **D7 = 1 (I/O Mode)**
- **D6, D5 (Group A Mode) = 00 (Mode 0)**
- **D4 (Port A Direction) = 0 (Output)**
- **D3 (Port C Upper Direction) = 1 (Input)**
- **D2 (Group B Mode) = 0 (Mode 0)**
- **D1 (Port B Direction) = 1 (Input)**
- **D0 (Port C Lower Direction) = 0 (Output)**
- **Resulting Control Word (Binary): `10001010b` = `8AH` (Hex)**

Assembly Code:

```
Code snippet

; Program to initialize 8255 and output static data to Port A
; Starting Address: 2000H

ORG 2000H

MVI A, 8AH      ; Load Accumulator with 8255 Control Word for desired configuration
OUT 83H         ; Write Control Word to 8255 CWR (Port A=Out, B=In, C_L=Out, C_U=In)
```

```
MVI A, 55H      ; Load Accumulator with data to be sent to Port A
OUT 80H         ; Output data (55H) to Port A (connected to LEDs)

HLT             ; Halt processor execution
```

Memory Layout (Hex Code):

Address	Hex Code	Instruction	Comments
2000H	3E	MVI A, 8AH	
2001H	8A		
2002H	D3	OUT 83H	
2003H	83		
2004H	3E	MVI A, 55H	
2005H	55		
2006H	D3	OUT 80H	
2007H	80		
2008H	76	HLT	

Expected Outcomes:

- **Port A LEDs should display the binary pattern for 55H (01010101b).**
- **Registers: A = 55H, PC = 2009H.**

4.2 Program 2: Blinking LEDs Connected to Port A

Objective: To configure 8255 Port A as output and continuously blink LEDs connected to it (turn ON, delay, turn OFF, delay).

Control Word Calculation (I/O Mode):

- **D7 = 1 (I/O Mode)**
- **D6, D5 = 00 (Group A - Mode 0)**
- **D4 (Port A Direction) = 0 (Output)**
- **D3 (Port C Upper Direction) = 1 (Input - chosen arbitrarily as it doesn't affect this program's core logic)**
- **D2 = 0 (Group B - Mode 0)**
- **D1 (Port B Direction) = 1 (Input - chosen arbitrarily)**
- **D0 (Port C Lower Direction) = 0 (Output - chosen arbitrarily)**

- **Resulting Control Word (Binary): `10001010b` = `8AH` (Hex) (Same as previous if this config works)**

Assembly Code:

```
Code snippet

; Program to blink LEDs connected to Port A of 8255
; Starting Address: 2000H

ORG 2000H

; --- 8255 Initialization ---
MVI A, 8AH      ; Load 8255 Control Word (Port A=Out, B=In, C_L=Out, C_U=In)
OUT 83H         ; Write Control Word to 8255 CWR

; --- Main Blinking Loop ---
LOOP_START:
   MVI A, FFH   ; Load Accumulator with FFH (All LEDs ON)
   OUT 80H      ; Output to Port A
   CALL DELAY   ; Call delay subroutine

   MVI A, 00H   ; Load Accumulator with 00H (All LEDs OFF)
   OUT 80H      ; Output to Port A
   CALL DELAY   ; Call delay subroutine

   JMP LOOP_START ; Jump back to LOOP_START to repeat blinking

; --- Delay Subroutine (Simple software delay) ---
DELAY:
   MVI C, 0FFH   ; Load C register with FFL (outer loop count)
DELAY_OUTER:
   MVI B, 0FFH   ; Load B register with FFL (inner loop count)
DELAY_INNER:
   DCR B        ; Decrement B
   JNZ DELAY_INNER ; If B not zero, jump back to DELAY_INNER
   DCR C        ; Decrement C
   JNZ DELAY_OUTER ; If C not zero, jump back to DELAY_OUTER
   RET          ; Return from subroutine
```

Memory Layout (Example, exact addresses will depend on subroutine size):

| Address | Hex Code | Instruction |

| :------ | :------- | :----------------- |

| 2000H | 3E | MVI A, 8AH |

| 2001H | 8A | |

| 2002H | D3 | OUT 83H |

| 2003H | 83 | |

| 2004H | 3E | MVI A, FFH |

| 2005H | FF | |

| 2006H | D3 | OUT 80H |

| 2007H | 80 | |

| 2008H | CD | CALL 2018H (approx) |

| 2009H | 18 | |

| 200AH | 20 | |

| 200BH | 3E | MVI A, 00H |

| 200CH | 00 | |

| 200DH | D3 | OUT 80H |

| 200EH | 80 | |

| 200FH | CD | CALL 2018H (approx) |

| 2010H | 18 | |

| 2011H | 20 | |

| 2012H | C3 | JMP 2004H |

| 2013H | 04 | |

| 2014H | 20 | |

| ... (Delay Subroutine starts here) | |

| 2018H | 0E | MVI C, FFH |

| 2019H | FF | |

| 201AH | 06 | MVI B, FFH |

| 201BH | FF | |

| 201CH | 05 | DCR B |

| 201DH | C2 | JNZ 201CH |

| 201EH | 1C | |

| 201FH | 20 | |

| 2020H | 0D | DCR C |

| 2021H | C2 | JNZ 201AH |

| 2022H | 1A | |

| 2023H | 20 | |

| 2024H | C9 | RET |

Expected Outcomes:

- **LEDs connected to Port A should continuously blink (all ON for a short duration, then all OFF for a short duration).**

4.3 Program 3: Read Switch Inputs from Port B and Display on Port C Lower

Objective: To configure Port B as input and Port C Lower as output (Mode 0). Read the state of 4 switches connected to PB0-PB3, then output this 4-bit data to PC0-PC3 to display on LEDs.

Control Word Calculation (I/O Mode):

- **D7 = 1 (I/O Mode)**
- **D6, D5 = 00 (Group A - Mode 0)**
- **D4 (Port A Direction) = 1 (Input - arbitrary as not used)**
- **D3 (Port C Upper Direction) = 1 (Input - arbitrary)**
- **D2 = 0 (Group B - Mode 0)**
- **D1 (Port B Direction) = 1 (Input)**
- **D0 (Port C Lower Direction) = 0 (Output)**
- **Resulting Control Word (Binary): `10011010b` = `9AH` (Hex)**

Assembly Code:

```
Code snippet

; Program to read switches from Port B and display on Port C lower
; Starting Address: 2000H

ORG 2000H

; --- 8255 Initialization ---
MVI A, 9AH      ; Load 8255 Control Word (Port A=In, B=In, C_L=Out, C_U=In)
OUT 83H         ; Write Control Word to 8255 CWR

; --- Main Loop to Read and Display ---
READ_LOOP:
```

```
IN 81H      ; Read data from Port B (connected to switches) into Accumulator
ANI 0FH     ; Mask out higher nibble (keep only lower 4 bits for switches PB0-PB3)
OUT 82H     ; Output Accumulator content to Port C (PC0-PC3 will show switch states)
JMP READ_LOOP; Jump back to READ_LOOP to continuously read
```

Memory Layout (Hex Code):

Address	Hex Code	Instruction
2000H	3E	MVI A, 9AH
2001H	9A	
2002H	D3	OUT 83H
2003H	83	
2004H	DB	IN 81H
2005H	81	
2006H	E6	ANI 0FH
2007H	0F	
2008H	D3	OUT 82H
2009H	82	
200AH	C3	JMP 2004H
200BH	04	
200CH	20	

Expected Outcomes:

- **LEDs connected to PC0-PC3 should reflect the exact binary state of the switches connected to PB0-PB3. For example, if PB0 and PB2 are ON (switches closed, assuming active high or pull-up/low logic), and PB1, PB3 are OFF, the LEDs on PC0 and PC2 should be ON.**

---

5. OBSERVATIONS AND RESULTS

Record your observations in the tables below after executing each program. Compare the actual behavior of LEDs/read data with the expected outcomes.

5.1 Program 1: Static Output to Port A

Expected State of LEDs on Port A (55H = 01010101b)	Observed State of LEDs on Port A
PA7: OFF, PA6: ON, PA5: OFF, PA4: ON	
PA3: OFF, PA2: ON, PA1: OFF, PA0: ON	

5.2 Program 2: Blinking LEDs on Port A

Observation	Confirmation (Yes/No)
LEDs on Port A blink continuously (ON/OFF).	
The blinking rate is perceptible.	

5.3 Program 3: Read Switch Inputs and Display on Port C Lower

Fill in the table for a few different switch combinations.

Switch States (PB3 PB2 PB1 PB0)	Expected LEDs on PC3 PC2 PC1 PC0	Observed LEDs on PC3 PC2 PC1 PC0	Read Data (Accumulator after IN)
0 0 0 0 (All OFF)	0 0 0 0		
0 0 0 1 (PB0 ON)	0 0 0 1		
0 0 1 0 (PB1 ON)	0 0 1 0		
1 0 1 1 (PB3, PB1, PB0 ON)	1 0 1 1		

[Add more combinations]			

---

6. CONCLUSION

Based on the observations, summarize your understanding of the 8255 PPI. Discuss how its different ports can be configured as inputs or outputs in Mode 0 using the Control Word. Explain the process of writing data to an output port and reading data from an input port. Comment on the success of controlling LEDs and reading switch inputs using the 8255 interfaced with the 8085.

---

7. VIVA VOCE QUESTIONS

1. **What is the main function of the 8255 Programmable Peripheral Interface?**
2. **How many I/O pins does the 8255 provide, and how are they organized?**
3. **Explain the purpose of the CS, RD, and WR pins of the 8255.**
4. **If the base address of an 8255 is C0H, what are the I/O addresses for Port A, Port B, Port C, and the Control Word Register?**
5. **What is the significance of the D7 bit in the 8255 Control Word?**
6. **Explain the difference between Mode 0 and Mode 1 of the 8255.**
7. **Calculate the 8255 Control Word to configure Port A as input, Port B as output, and Port C as input (all in Mode 0).**
8. **How would you individually set bit PC5 using the BSR mode? What control word would be needed?**
9. **Why is it essential to send a control word to the 8255 before using its I/O ports?**
10. **What typically happens to the 8255's port configurations when the system is reset?**