MICROPROCESSOR ENGINEERING

Mastering the 8085 Microprocessor: A Comprehensive Technical Guide to Architecture, Instruction Sets, and Interrupt Systems

Published: March 05, 2025 | Tags: 8085 Microprocessor | Computer Architecture | Microprocessor MCQs | Assembly Language | Interrupt Systems

The **Intel 8085** remains one of the most significant milestones in the evolution of computing. Introduced in 1976 as a successor to the 8080, it is an 8-bit general-purpose microprocessor capable of addressing 64 KB of memory. While modern computing has moved into the realm of multi-core 64-bit architectures, the 8085 remains the fundamental pedagogical tool for understanding computer organization, assembly language programming, and hardware interfacing. This guide provides an exhaustive technical analysis of the 8085 microprocessor, exploring its internal architecture, instruction set, addressing modes, and interrupt structures in granular detail.

The Internal Architecture of the 8085 Microprocessor

The architecture of the 8085 is designed around an 8-bit internal data bus, which connects all internal components including the Arithmetic Logic Unit (ALU), registers, and control units. The processor is manufactured using **N-channel Metal-Oxide-Semiconductor (NMOS)** technology and is packed into a 40-pin Dual In-line Package (DIP).

1. The Arithmetic Logic Unit (ALU)

The ALU is the computational core of the 8085. It performs arithmetic operations such as addition and subtraction, and logical operations such as AND, OR, XOR, and bitwise rotation. The ALU interacts directly with the **Accumulator** (Register A) and a temporary register. The results of ALU operations typically update the **Flag Register**, which provides status information about the result (e.g., whether the result was zero or negative).

2. The Register Array

The 8085 features a versatile set of registers that minimize the need for frequent memory access, thereby increasing execution speed. These registers are categorized as follows:

- Accumulator (A): An 8-bit register used for all arithmetic and logical operations. It serves as one of the operands and usually stores the final result.
- General-Purpose Registers: There are six 8-bit registers-B, C, D, E, H, and L. These can be used individually or in pairs (BC, DE, HL) to act as 16-bit registers for specific operations.

- Program Counter (PC): A 16-bit register that stores the memory address of the next instruction to be executed. It ensures the sequential execution of the program.
- Stack Pointer (SP): A 16-bit register used to manage the stack memory. It points to the current top of the stack, facilitating subroutines and interrupt handling.
- Temporary Register and W/Z Registers: These are internal registers used by the CPU during execution and are not accessible to the programmer.

3. The Flag Register (Status Word)

The 8085 includes a 1-byte status register containing five active flags. These flags are set or reset based on the outcome of ALU operations:

Flag Symbol	Name	Description
S	Sign Flag	Set if the most significant bit (D7) of the result is 1 (negative).
Z	Zero Flag	Set if the result of an operation is exactly zero.
AC	Auxiliary Carry	Set if there is a carry from bit D3 to D4 (used for BCD arithmetic).
P	Parity Flag	Set if the result contains an even number of 1s (even parity).
CY	Carry Flag	Set if an arithmetic operation results in a carry or borrow.

Addressing Modes in 8085

Addressing modes define the method by which the microprocessor identifies the operand (data) for an instruction. Understanding these modes is critical for optimizing code for both speed and memory efficiency. The 8085 supports five distinct addressing modes:

1. Immediate Addressing Mode

In this mode, the data is part of the instruction itself. These instructions are usually 2 or 3 bytes long. For example, moves the immediate value 32H into the accumulator. The 'I' in the mnemonic typically indicates immediate addressing.

2. Register Addressing Mode

This mode specifies that the data is located within one of the internal registers. Since no memory access is required, these are the fastest instructions to execute. Example: copies the content of register B into register A.

3. Direct Addressing Mode

In direct addressing, the 16-bit memory address of the operand is explicitly provided within the instruction. These are 3-byte instructions. Example: loads the contents of memory location 2000H directly into the accumulator.

4. Indirect Addressing Mode

Here, the instruction specifies a register pair (usually the HL pair) that contains the memory address where the operand is located. Example: . The symbol 'M' refers to the memory location pointed to by the HL register pair.

5. Implied/Implicit Addressing Mode

These instructions do not require an operand because the operand is implied by the opcode itself. Example: (Complement Accumulator) or (Set Carry Flag). These are always 1-byte instructions.

The 8085 Instruction Set Architecture (ISA)

The 8085 instruction set consists of 74 basic instructions which expand into 246 functional opcodes. These instructions are categorized based on their function and their length in bytes.

Functional Categorization

1. Data Transfer Instructions: These instructions move data between registers, or between registers and memory. Examples: MOV, MVI, LXI, LDA, STA. Crucially, data transfer instructions do not affect the flags.

2. Arithmetic Instructions: Perform addition, subtraction, increment, or decrement operations. Examples: ADD, SUB, INR, DCR. All flags are generally affected based on the result.
3. Logical Instructions: Include operations like AND, OR, XOR, Compare, and Rotate. Examples: ANA, ORA, XRA, RLC, RRC. Logical instructions are vital for masking bits and bitwise manipulation.
4. Branching Instructions: Change the flow of program execution. This includes unconditional and conditional jumps, calls, and returns. Examples: JMP, JZ (Jump if Zero), CALL, RET.
5. Machine Control Instructions: Control the state of the processor. Examples: HLT (Halt), NOP (No Operation), DI (Disable Interrupts), EI (Enable Interrupts).

Instruction Word Size

Instructions in the 8085 are classified by their memory footprint:

- 1-Byte Instructions: Include the opcode and operand in a single byte (e.g., MOV A, B).
- 2-Byte Instructions: The first byte is the opcode; the second byte is an 8-bit data or port address (e.g., MVI A, 05H).
- 3-Byte Instructions: The first byte is the opcode; the following two bytes are a 16-bit address or data (e.g., LXI H, 2050H).

Interrupt Structure and Management

Interrupts are signals sent by external devices to the microprocessor, requesting immediate attention. When an interrupt is received, the 8085 suspends its current task, saves the program counter on the stack, and executes a **Service Routine (ISR)**.

Hardware Interrupts

The 8085 has five hardware interrupt pins, ranked by priority:

Interrupt Name	Priority	Maskability	Type	Vector Address
TRAP	1 (Highest)	Non-maskable	Edge & Level	0024H
RST 7.5	2	Maskable	Edge Triggered	003CH
RST 6.5	3	Maskable	Level Triggered	0034H
RST 5.5	4	Maskable	Level Triggered	002CH
INTR	5 (Lowest)	Maskable	Level Triggered	Externally supplied

TRAP is a unique interrupt because it cannot be disabled by software (non-maskable). It is typically reserved for critical system failures like power loss. The **RST (Restart)** interrupts are vectored, meaning the CPU automatically knows the memory address of the ISR. **INTR** is non-vectored; the interrupting device must provide the opcode (usually an RST instruction) during the interrupt acknowledge cycle.

Software Interrupts

There are eight software interrupts: **RST 0** through **RST 7**. These are 1-byte instructions that function like a subroutine call. They are often used by programmers to implement breakpoints or system calls. The vector address for any instruction is calculated as (in decimal), converted to hexadecimal.

The 8085 Pin-Out and Signal Groups

The 40 pins of the 8085 can be grouped into functional categories that explain how the processor interacts with the outside world:

1. Address and Data Bus

The 8085 uses a multiplexed address/data bus (AD0-AD7) to save pins. During the first T-state of a machine cycle, these pins carry the lower 8 bits of the address. During subsequent T-states, they carry 8-bit data. The **ALE (Address Latch Enable)** signal is used to distinguish between the two. Pins A8-A15 carry the higher 8 bits of the address and are not multiplexed.

2. Control and Status Signals

- RD (Read): An active-low signal indicating the selected memory or I/O device is to be read.
- WR (Write): An active-low signal indicating data on the bus is to be written to the selected device.
- IO/M: A status signal that differentiates between I/O operations (High) and memory operations (Low).
- S0, S1: Status signals that define the type of machine cycle (e.g., Fetch, Read, Write, Halt).

3. Power Supply and Clock

The 8085 operates on a single +5V DC supply. It features an on-chip clock generator. A crystal is connected across X1 and X2; the internal clock frequency is half the crystal frequency. For example, a 6.25 MHz crystal results in a 3.125 MHz internal operating frequency.

Machine Cycles and Timing Diagrams

The execution of an instruction is broken down into **Machine Cycles**, and each machine cycle is composed of **T-states** (Clock Periods). A typical instruction starts with an **Opcode Fetch** cycle, which usually takes 4 T-states. Depending on the instruction, additional Memory Read or Memory Write cycles (3 T-states each) may be required.

Understanding timing is essential for hardware interfacing. For instance, the **WAIT** state can be induced via the **READY** pin if a memory device is slower than the CPU, ensuring that data is not lost during a read/write operation.

Practical Implementation: Interfacing and Programming

Programming the 8085 involves writing assembly code, which is then converted into hex codes. A common task is data block transfer. To move a block of data from memory location 2000H to 3000H, a programmer would utilize the HL and DE register pairs as pointers and use a loop involving the and instructions.

Case Study: 8-Bit Addition with Carry

Consider the task of adding two 8-bit numbers where the sum might exceed 255 (FFH). The logic involves:

1. Clearing a register to store the carry (e.g., MVI C, 00H).
2. Loading the first number into the Accumulator.
3. Adding the second number using ADD.
4. Using the conditional jump JNC (Jump if No Carry) to skip the increment of the carry register.
5. Storing the result (Accumulator) and the carry (Register C) in memory.

Troubleshooting Common 8085 Operational Challenges

Engineers often face specific hurdles when working with 8085 systems:

- Bus Contention: Occurs when multiple devices attempt to drive the data bus simultaneously. This is typically solved through proper address decoding and ensuring Tri-state logic is respected.
- Stack Overflow: If subroutines are nested too deeply or interrupts occur too frequently without proper RET instructions, the Stack Pointer can overwrite program data. Proper SP initialization at the top of RAM is critical.
- Interrupt Latency: For real-time applications, the time taken to respond to an interrupt (latency) must be minimized. Using vectored interrupts (RST 7.5) instead of INTR can reduce this delay.

Conclusion: The Enduring Legacy of the 8085

The Intel 8085 serves as the blueprint for understanding how microprocessors function at a hardware and software level. Its architecture encapsulates the core principles of the von Neumann model, illustrating the seamless interaction between the ALU, registers, and memory. While silicon technology has progressed to 5nm processes and billions of transistors, the logic of the 8085-its flags, its fetch-execute cycles, and its interrupt handling-remains the foundation upon which all modern computing is built. For students and professionals alike, mastering the 8085 is not just a lesson in history, but a deep dive into the fundamental mechanics of digital logic and system design.