Addressing mode

8051

Addressing Mode	Example Instruction	Explanation
Immediate	MOV A, #25H	Directly moves value 25H to A.
Register	MOV A, R1	Moves data from register R1 to A.
Direct	MOV A, 30H	Moves data from address 30H to A.
Register Indirect	MOV A, @R0	Moves data from address in R0 to A.
Indexed	MOVC A, @A+DPTR	Accesses code memory using A + DPTR.
Implied	CPL A	Operates directly on accumulator A.

8086

Addressing Mode	Example Instruction	Explanation
Immediate	MOV AX, 1234H	Directly loads value 1234H into AX.
Register	MOV AX, BX	Copies data from BX to AX.
Direct	MOV AX, [1234H]	Accesses memory at address 1234H.
Register Indirect	MOV AX, [BX]	Accesses memory using address in BX.
Based	MOV AX, [BX + 5]	Combines BX with a displacement of 5.
Indexed	MOV AX, [SI + 6]	Combines SI with a displacement of 6.
Based-Indexed	MOV AX, [BX + SI]	Adds BX and SI for the address.
Based-Indexed with Displacement	MOV AX, [BX + SI + 8]	Combines BX, SI, and an offset of 8.
Relative	JMP SHORT LABEL	Branches to a nearby label.
Implied	CLC	Operates on an implied operand.

Feature	Microprocessor	Microcontroller
Definition	A microprocessor is a central processing unit (CPU) on a single chip, typically designed for high-performance applications.	A microcontroller is a small computer on a single chip that contains a CPU, memory, and peripherals.
Components	Only the CPU; external components like memory and I/O devices are required.	Contains CPU, memory (RAM, ROM), and peripherals (I/O, timers, etc.) on a single chip.
Cost	Typically more expensive due to higher performance and complexity.	Usually less expensive because of integrated components.
Speed	Higher processing speed, capable of handling complex tasks.	Slower processing speed compared to microprocessors, suitable for simple tasks.
Power Consumption	Higher power consumption due to more complex architecture.	Low power consumption as it is designed for embedded applications.
Applications	Used in personal computers, laptops, servers, etc., requiring high computation power.	Used in embedded systems like home appliances, automotive systems, industrial machines, etc.
External Components	Requires external memory, I/O devices, and other peripherals to function.	Often does not need external components because it has integrated memory and peripherals.
Programming	Can be programmed using high-level languages or assembly.	Programmed typically in embedded C or assembly language.
Example	Intel Core i7, AMD Ryzen, etc.	Arduino, PIC, AVR, ARM-based microcontrollers.
Flexibility	More flexible for general-purpose computing tasks.	More specialized for controlling specific tasks in embedded systems.
Size	Larger in size due to external components.	Small in size as it integrates all components.

8086 Memory Organization

Memory Type	Description
Total Address Space	1MB (1024 KB), with addresses ranging from 00000H to FFFFFH.
Segmentation	Memory is divided into 4 segments: Code, Data, Stack, and Extra. Each segment is 64 KB in size.
Code Segment (CS)	Stores executable code, with a size of 64 KB.
Data Segment (DS)	Stores data variables, with a size of 64 KB.
Stack Segment (SS)	Stores stack data for function calls and local variables, with a size of 64 KB.
Extra Segment (ES)	Used for extra data storage, with a size of 64 KB.
Memory Addressing	Uses a 16-bit address bus and 20-bit address space through a combination of a 16-bit segment register and a 16-bit offset register (segment:offset).
Data Bus	16-bit data bus for communication with memory.

8051 Memory Organization

Memory Type	Description
ROM (Program Memory)	Typically 4KB of on-chip ROM for storing program code.
RAM (Data Memory)	128 bytes of on-chip RAM for data storage, divided into:
Data Memory (Internal RAM)	- 0x00 to 0x1F: Special Function Registers (SFR). - 0x20 to 0x7F: General-purpose registers for data storage. - 0x80 to 0xFF: Additional 128 bytes available for data storage.
External Memory	8051 supports up to 64 KB of external program and data memory (using external ROM/RAM). This can be accessed via the External Memory Interface.
Bit Addressable Area	The lower 32 bytes (0x00 to 0x1F) of RAM are bit-addressable.
Data Bus	8-bit data bus for communication with memory.
Address Bus	16-bit address bus, addressing up to 64KB of external memory.

ARM instruction set

1. Data Processing Instructions

Data processing instructions perform arithmetic, logical, and comparison operations on registers.

- **Arithmetic:**
  - ADD: Add two values (e.g., ADD R1, R2, R3)
  - SUB: Subtract one value from another
  - MUL: Multiply two values
  - RSB: Reverse subtract (e.g., RSB R1, R2, R3 calculates R1 = R3 - R2)
  - ADC: Add with carry
  - SBC: Subtract with carry
- **Logical:**
  - AND: Logical AND
  - ORR: Logical OR
  - EOR: Exclusive OR (XOR)
  - BIC: Bit clear (AND NOT)
- **Shift/Rotate:**
  - LSL: Logical shift left
  - LSR: Logical shift right
  - ASR: Arithmetic shift right
  - ROR: Rotate right
- **Comparison:**
  - CMP: Compare two registers (sets flags only)
  - CMN: Compare negative
  - TST: Test bits using AND
  - TEQ: Test bits using XOR

2. Branch Instructions

Branch instructions alter program flow.

- **Unconditional Branch:**
  - B: Branch to a target address (e.g., B label)
- **Conditional Branch (based on flags in the CPSR):**
  - BEQ: Branch if equal (zero flag set)
  - BNE: Branch if not equal (zero flag clear)
  - BGT: Branch if greater than (signed)
  - BLT: Branch if less than (signed)
  - BGE: Branch if greater than or equal
  - BLE: Branch if less than or equal
- **Branch with Link:**
  - BL: Branch to a subroutine and save the return address in LR.

---

3. Load and Store Instructions

These instructions move data between memory and registers.

- **Single Register:**
  - LDR: Load data from memory to a register (e.g., LDR R1, [R2])
  - STR: Store data from a register to memory
- **Multiple Registers:**
  - LDM: Load multiple registers from memory
  - STM: Store multiple registers to memory
- **Immediate Offsets:**
  - LDR R1, [R2, #4]: Load from an address offset by 4 bytes.
  - STR R1, [R2, -#8]: Store to an address offset by -8 bytes.

---

4. Software Interrupt Instructions

Software interrupts invoke system-level operations or services.

- SWI n: Software interrupt, where n is the interrupt number.

5. Program Status Register Instructions

- **Reading Status Registers:**
  - MRS: Move status register to a general-purpose register (e.g., MRS R1, CPSR).
- **Writing to Status Registers:**
  - MSR: Move a value to the status register (e.g., MSR CPSR_c, R1 modifies condition flags in CPSR).

6. Loading Constants

ARM has specific instructions for loading constants.

- **Immediate Values:**
  - MOV: Load an immediate value into a register (e.g., MOV R1, #10).
  - Limitation: Immediate values must fit in 8 bits with a rotation.
- **Longer Constants:**
  - LDR R1, =0x12345678: Load a 32-bit constant into a register. The assembler places the constant in memory and loads it indirectly.

---

7. Conditional Execution

One of ARM's unique features is that most instructions can be conditionally executed based on the CPSR flags.

- Syntax: [condition] appended to the instruction.
  - EQ: Equal (zero flag set)
  - NE: Not equal (zero flag clear)
  - GT: Greater than (signed)
  - LT: Less than (signed)
  - GE: Greater than or equal
  - LE: Less than or equal
  - AL: Always (default, unconditional)

Serial Communication Standards

1. RS232:
- A widely used standard for short-distance point-to-point communication.
- Voltage levels: +3V to +15V (logic 0), -3V to -15V (logic 1).
- Used in modems, printers, and legacy systems.

2. RS485:
- Designed for long-distance and multi-device communication.
- Differential signaling reduces noise, supporting up to 32 devices on a single bus.
- Common in industrial automation.

3. I2C (Inter-Integrated Circuit):
- Multi-master, multi-slave protocol for short-distance communication.
- Uses two lines: SDA (data) and SCL (clock).
- Widely used in sensors and microcontrollers.

4. SPI (Serial Peripheral Interface):
- Full-duplex protocol with separate lines for data and clock.
- Supports multiple slaves via chip-select lines.
- High-speed communication for peripherals like ADCs and displays.

5. USB (Universal Serial Bus):
- High-speed serial standard for connecting multiple devices.
- Supports data transfer, power delivery, and plug-and-play functionality.
- Speeds range from 12 Mbps (USB 1.0) to 40 Gbps (USB 4.0).