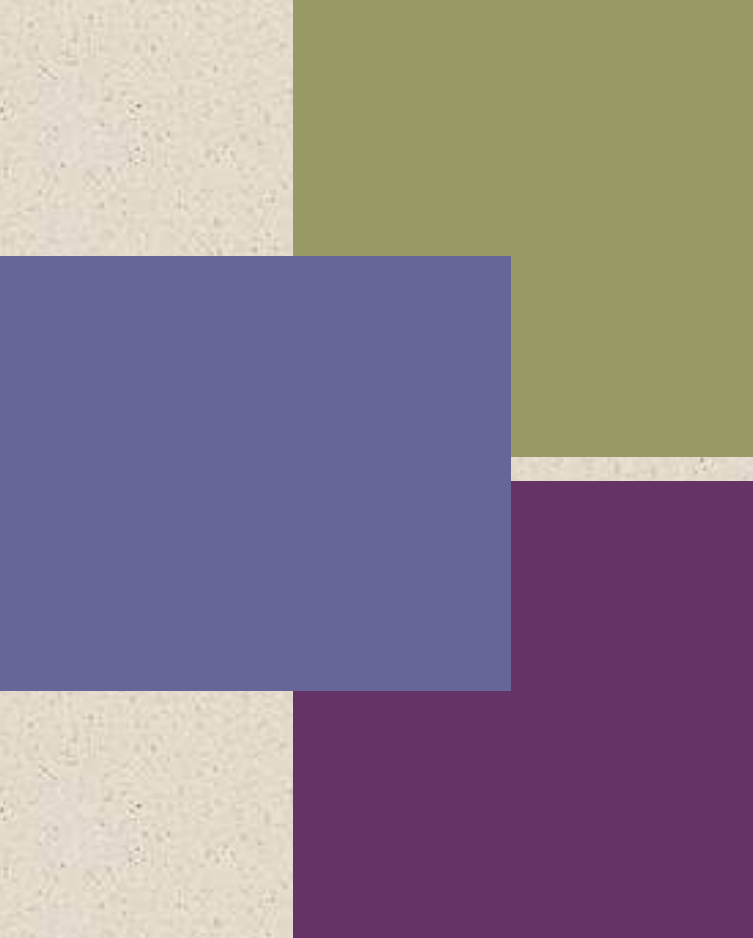


**William Stallings
Computer Organization
and Architecture
10th Edition**



+ Chapter 20

Control Unit Operation



Micro-Operations

- The functional, or atomic, operations of a processor
- Series of steps, each of which involves the processor registers
- *Micro* refers to the fact that each step is very simple and accomplishes very little
- The execution of a program consists of the sequential execution of instructions
 - Each instruction is executed during an instruction cycle made up of shorter subcycles (fetch, indirect, execute, interrupt)
 - The execution of each subcycle involves one or more shorter operations (micro-operations)



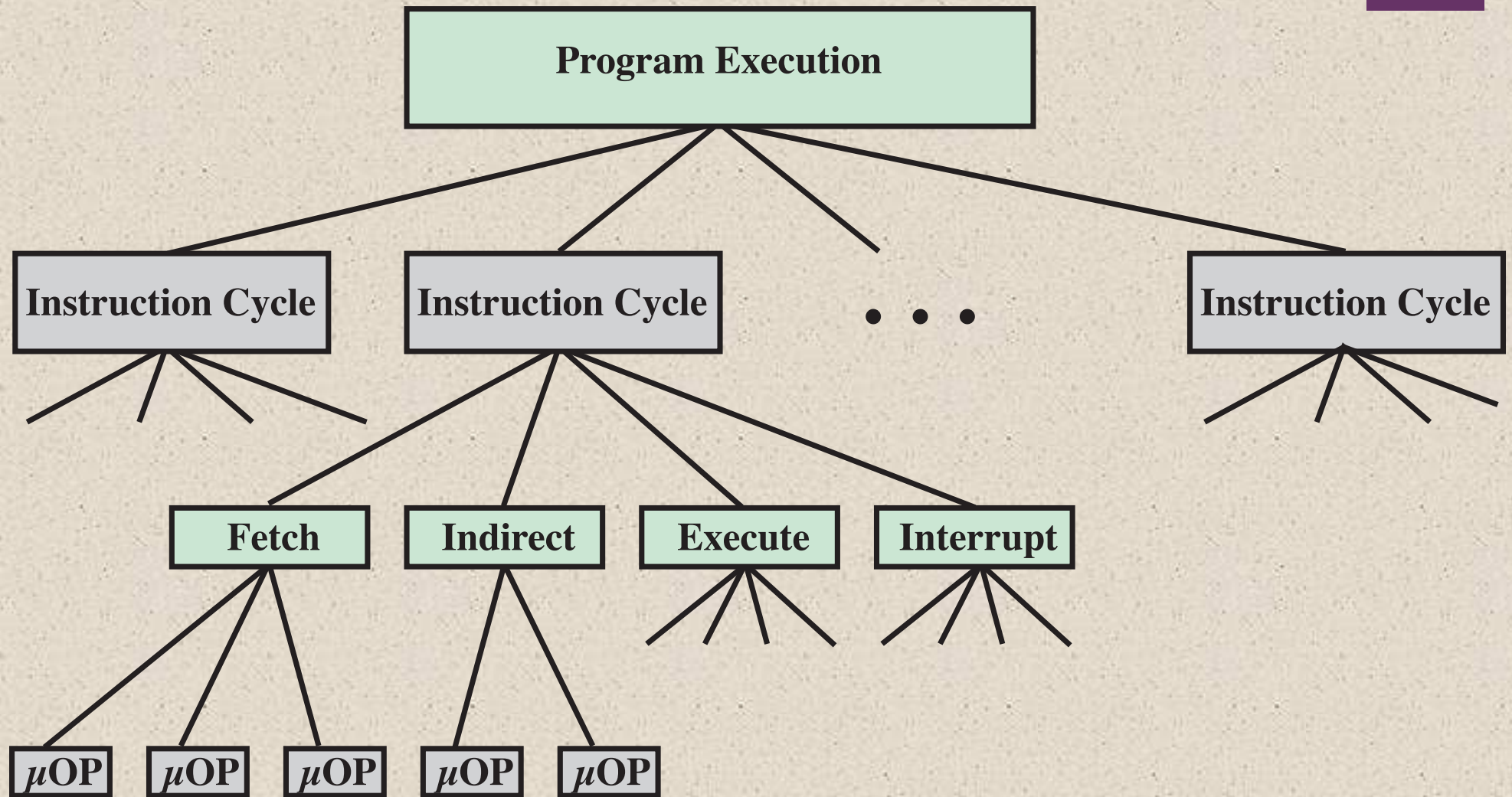


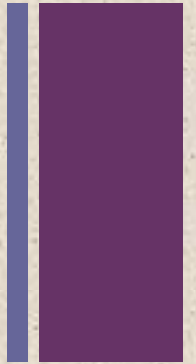
Figure 20.1 Constituent Elements of a Program Execution



The Fetch Cycle

- Occurs at the beginning of each instruction cycle and causes an instruction to be fetched from memory

- Four registers are involved:
 - Memory Address Register (MAR)
 - Connected to address bus
 - Specifies address for read or write operation
 - Memory Buffer Register (MBR)
 - Connected to data bus
 - Holds data to write or last data read
 - Program Counter (PC)
 - Holds address of next instruction to be fetched
 - Instruction Register (IR)
 - Holds last instruction fetched



tMAR	
MBR	
PC	0000000001100100
IR	
AC	

(a) Beginning (before t_1)

MAR	0000000001100100
MBR	
PC	0000000001100100
IR	
AC	

(b) After first step

MAR	0000000001100100
MBR	0001000000100000
PC	0000000001100101
IR	
AC	

(c) After second step

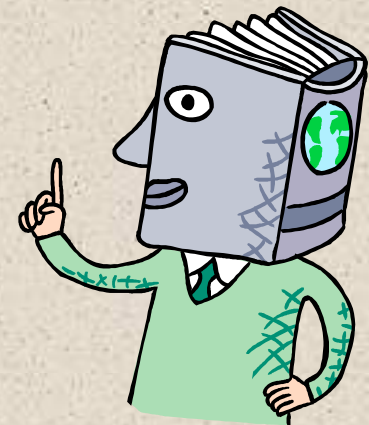
MAR	0000000001100100
MBR	0001000000100000
PC	0000000001100101
IR	0001000000100000
AC	

(d) After third Step

Figure 20.2 Sequence of Events, Fetch Cycle

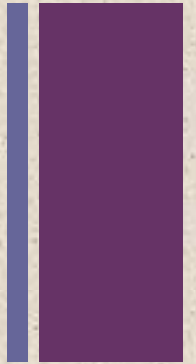
+ Rules for Micro-Operations Grouping

- Proper sequence must be followed
 - $MAR \leftarrow (PC)$ must precede $MBR \leftarrow (\text{memory})$
- Conflicts must be avoided
 - Must not read and write same register at same time
 - $MBR \leftarrow (\text{memory})$ and $IR \leftarrow (MBR)$ must not be in same cycle
- One of the micro-operations involves an addition
 - To avoid duplication of circuitry, this addition could be performed by the ALU
 - The use of the ALU may involve additional micro-operations, depending on the functionality of the ALU and the organization of the processor





Indirect Cycle



- Once an instruction is fetched, the next step is to fetch source operands
- Assuming a one-address instruction format, with direct and indirect addressing allowed:
 - If the instruction specifies an indirect address, then an indirect cycle must precede the execute cycle
 - The address field of the instruction is transferred to the MAR
 - This is then used to fetch the address of the operand
 - Finally, the address field of the IR is updated from the MBR, so that it now contains a direct rather than an indirect address
 - The IR is now in the same state as if indirect addressing had not been used, and it is ready for the execute cycle

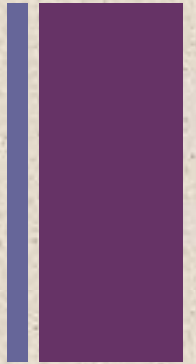


Interrupt Cycle

- At the completion of the execute cycle, a test is made to determine whether any enabled interrupts have occurred, and if so, the interrupt cycle occurs
- The nature of this cycle varies greatly from one machine to another
- In a simple sequence of events:
 - In the first step the contents of the PC are transferred to the MBR so that they can be saved for return from the interrupt
 - Then the MAR is loaded with the address at which the contents of the PC are to be saved, and the PC is loaded with the address of the start of the interrupt-processing routine
 - These two actions may each be a single micro-operation
 - Because most processors provide multiple types and/or levels of interrupts, it may take one or more additional micro-operations to obtain the `Save_Address` and the `Routine_Address` before they can be transferred to the MAR and PC respectively
 - Once this is done, the final step is to store the MBR, which contains the old value of the PC, into memory
 - The processor is now ready to begin the next instruction cycle



Execute Cycle



- Because of the variety of opcodes, there are a number of different sequences of micro-operations that can occur
- Instruction decoding
 - The control unit examines the opcode and generates a sequence of micro-operations based on the value of the opcode
- A simplified add instruction:
 - ADD R1, X (which adds the contents of the location X to register R1)
 - In the first step the address portion of the IR is loaded into the MAR
 - Then the referenced memory location is read
 - Finally the contents of R1 and MBR are added by the ALU
 - Additional micro-operations may be required to extract the register reference from the IR and perhaps to stage the ALU inputs or outputs in some intermediate registers

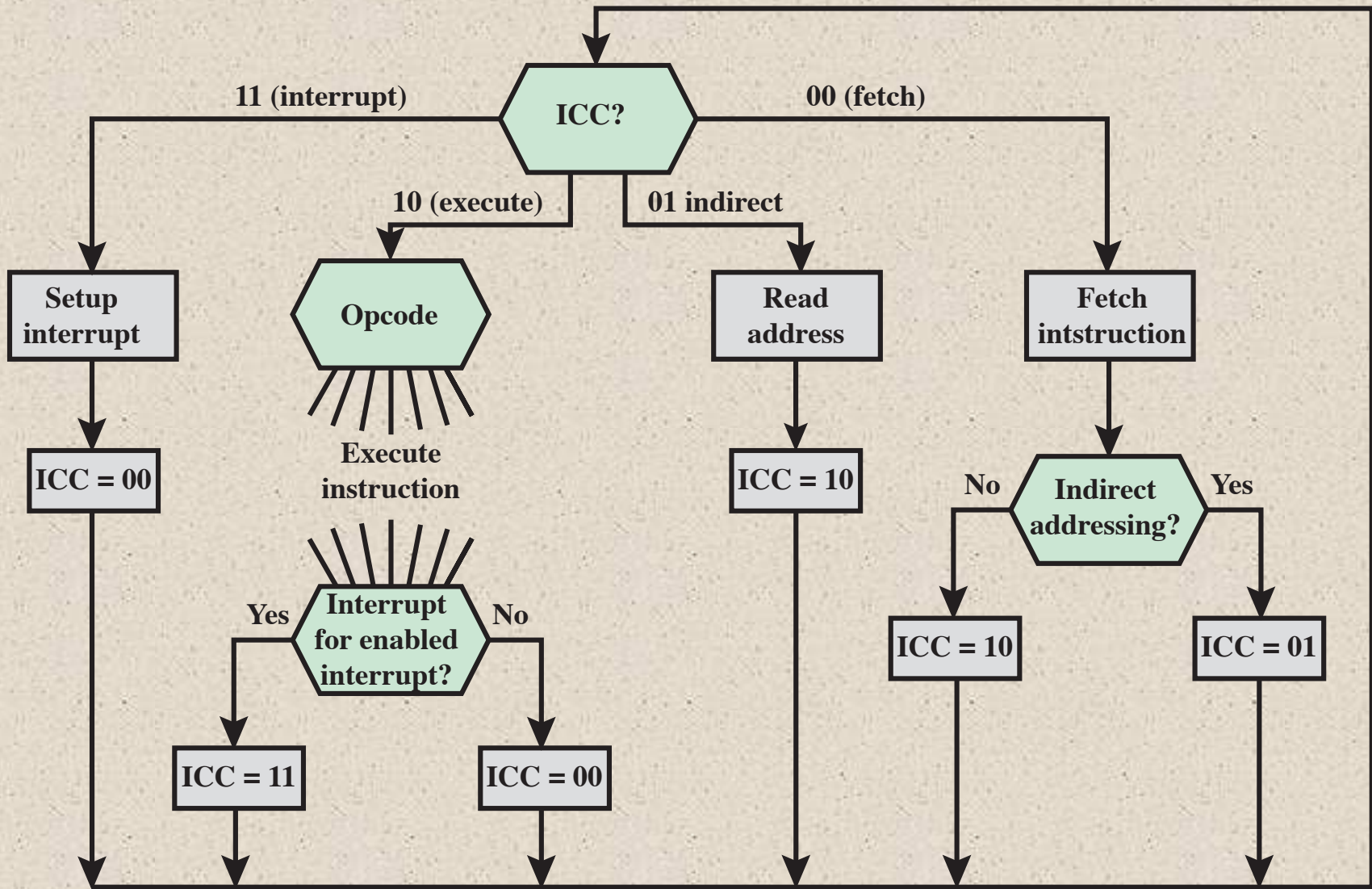


Figure 20.3 Flowchart for Instruction Cycle

+ Control Unit Functional Requirements

- By reducing the operation of the processor to its most fundamental level we are able to define exactly what it is that the control unit must cause to happen
- Three step process to lead to a characterization of the control unit:
 - Define basic elements of processor
 - Describe micro-operations processor performs
 - Determine the functions that the control unit must perform to cause the micro-operations to be performed
- The control unit performs two basic tasks:
 - Sequencing
 - Execution

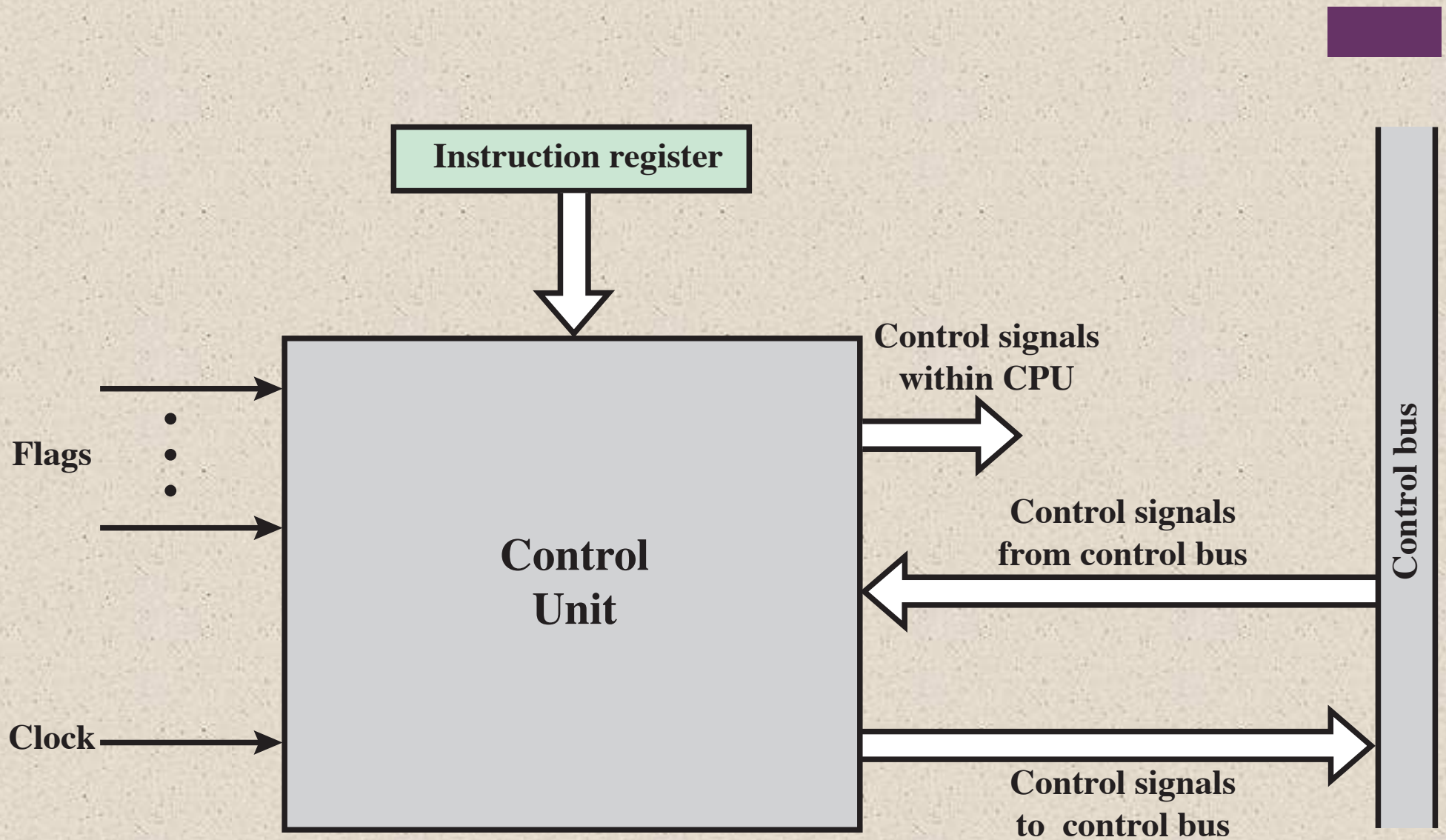


Figure 20.4 Block Diagram of the Control Unit

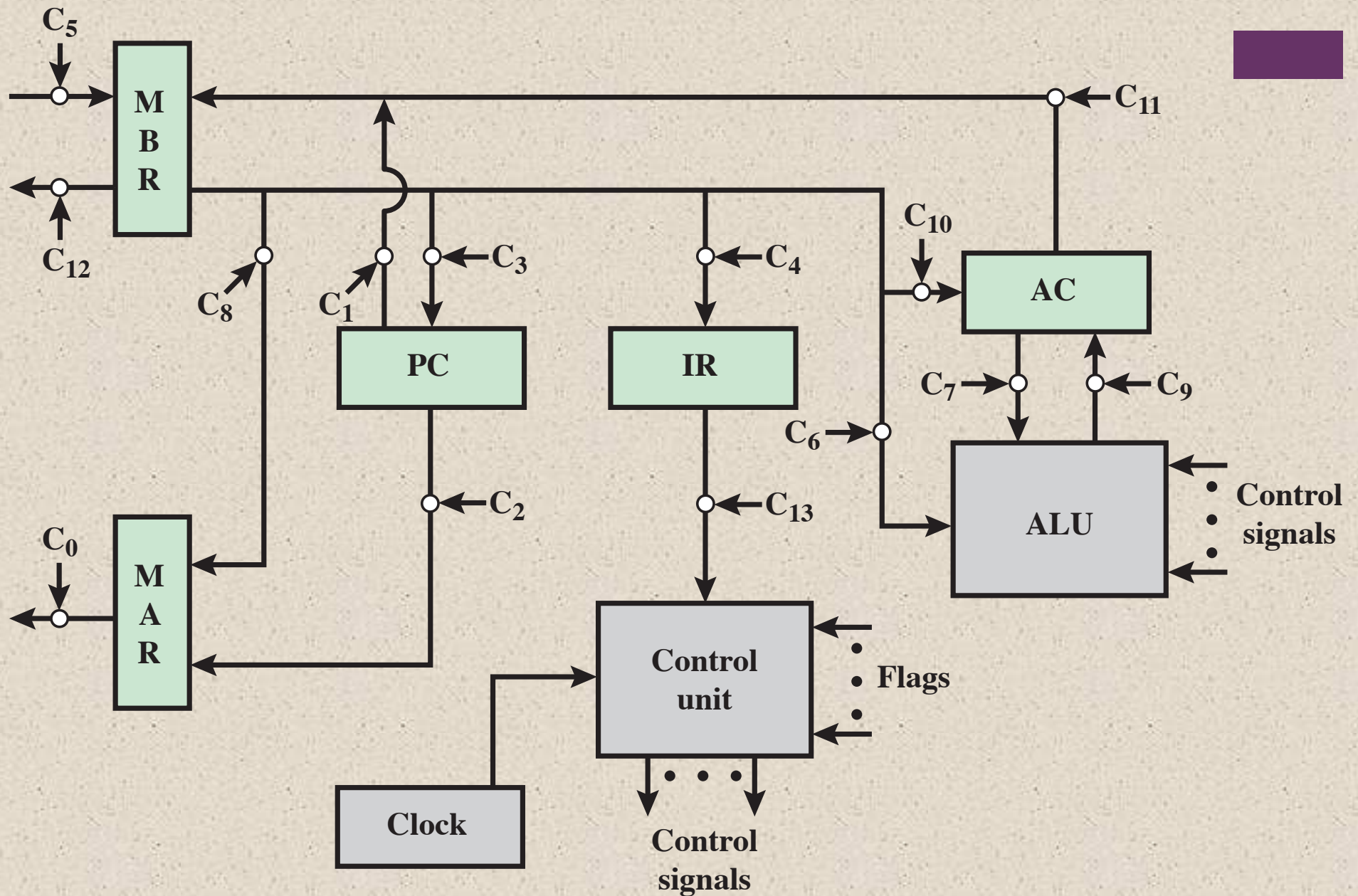


Figure 20.5 Data Paths and Control Signals

Table 20.1 Micro-operations and Control Signals



	Micro-operations	Active Control Signals
Fetch:	$t_1: \text{MAR} \leftarrow (\text{PC})$	C_2
	$t_2: \text{MBR} \leftarrow \text{Memory}$ $\text{PC} \leftarrow (\text{PC}) + 1$	C_5, C_R
	$t_3: \text{IR} \leftarrow (\text{MBR})$	C_4
Indirect:	$t_1: \text{MAR} \leftarrow (\text{IR}(\text{Address}))$	C_8
	$t_2: \text{MBR} \leftarrow \text{Memory}$	C_5, C_R
	$t_3: \text{IR}(\text{Address}) \leftarrow (\text{MBR}(\text{Address}))$	C_4
Interrupt:	$t_1: \text{MBR} \leftarrow (\text{PC})$	C_1
	$t_2: \text{MAR} \leftarrow \text{Save-address}$ $\text{PC} \leftarrow \text{Routine-address}$	
	$t_3: \text{Memory} \leftarrow (\text{MBR})$	C_{12}, C_W

C_R = Read control signal to system bus.

C_W = Write control signal to system bus.

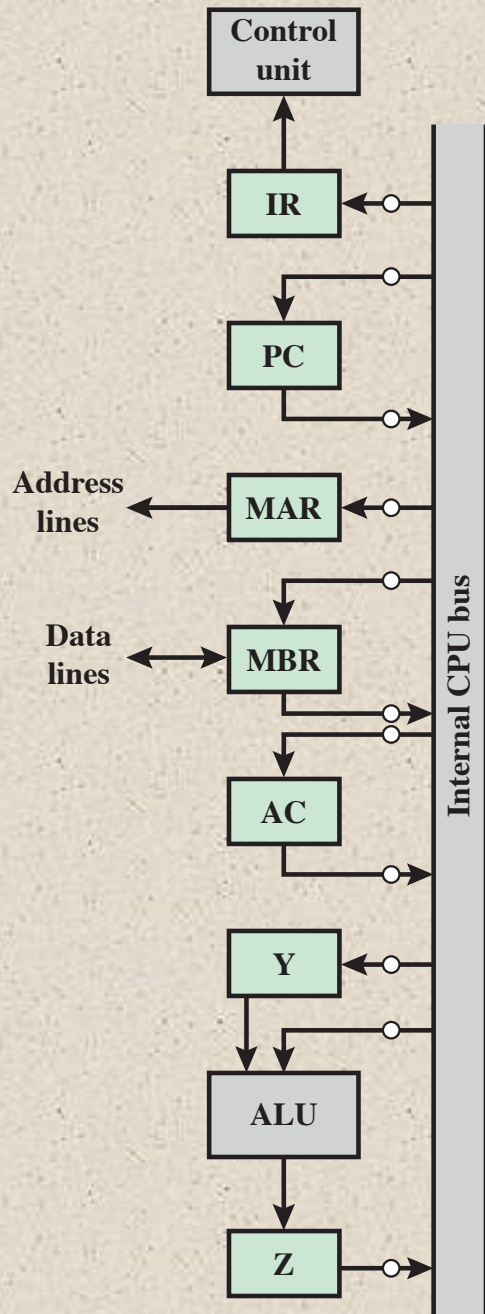


Figure 20.6 CPU with Internal Bus

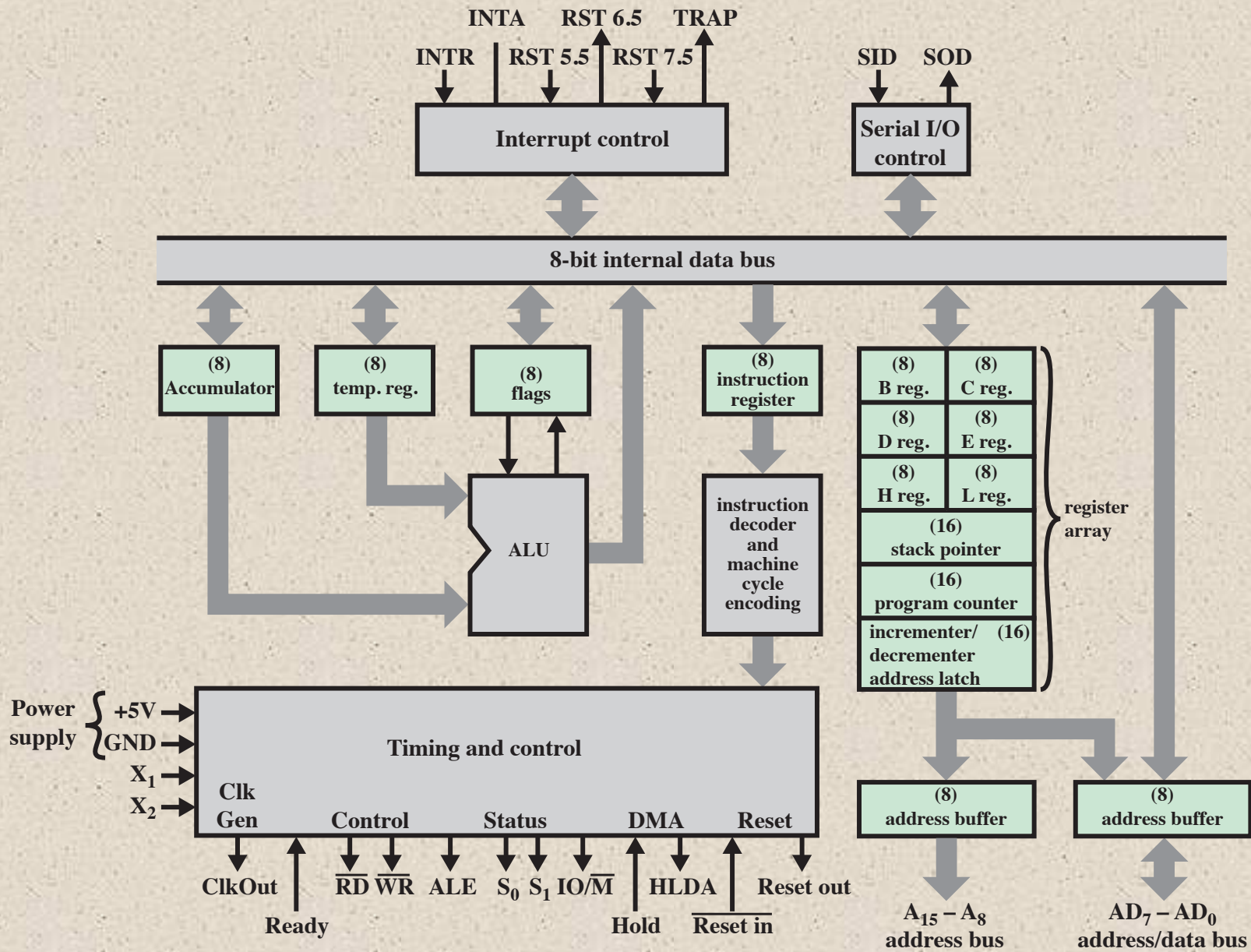


Figure 20.7 Intel 8085 CPU Block Diagram

Address and Data Signals

High Address (A15–A8)

The high-order 8 bits of a 16-bit address.

Address/Data (AD7–AD0)

The lower-order 8 bits of a 16-bit address or 8 bits of data. This multiplexing saves on pins.

Serial Input Data (SID)

A single-bit input to accommodate devices that transmit serially (one bit at a time).

Serial Output Data (SOD)

A single-bit output to accommodate devices that receive serially.

Timing and Control Signals

CLK (OUT)

The system clock. The CLK signal goes to peripheral chips and synchronizes their timing.

X1, X2

These signals come from an external crystal or other device to drive the internal clock generator.

Address Latch Enabled (ALE)

Occurs during the first clock state of a machine cycle and causes peripheral chips to store the address lines. This allows the address module (e.g., memory, I/O) to recognize that it is being addressed.

Status (S0, S1)

Control signals used to indicate whether a read or write operation is taking place.

IO/M

Used to enable either I/O or memory modules for read and write operations.

Read Control (RD)

Indicates that the selected memory or I/O module is to be read and that the data bus is available for data transfer.

Write Control (WR)

Indicates that data on the data bus is to be written into the selected memory or I/O location.

Table 20.2

Intel 8085 External Signals (page 1 of 2)

Memory and I/O Initiated Symbols

Hold

Requests the CPU to relinquish control and use of the external system bus. The CPU will complete execution of the instruction presently in the IR and then enter a hold state, during which no signals are inserted by the CPU to the control, address, or data buses. During the hold state, the bus may be used for DMA operations.

Hold Acknowledge (HOLDA)

This control unit output signal acknowledges the HOLD signal and indicates that the bus is now available.

READY

Used to synchronize the CPU with slower memory or I/O devices. When an addressed device asserts READY, the CPU may proceed with an input (DBIN) or output (WR) operation. Otherwise, the CPU enters a wait state until the device is ready.

Interrupt-Related Signals

TRAP

Restart Interrupts (RST 7.5, 6.5, 5.5)

Interrupt Request (INTR)

These five lines are used by an external device to interrupt the CPU. The CPU will not honor the request if it is in the hold state or if the interrupt is disabled. An interrupt is honored only at the completion of an instruction. The interrupts are in descending order of priority.

Interrupt Acknowledge

Acknowledges an interrupt.

CPU Initialization

RESET IN

Causes the contents of the PC to be set to zero. The CPU resumes execution at location zero.

RESET OUT

Acknowledges that the CPU has been reset. The signal can be used to reset the rest of the system.

Voltage and Ground

VCC

+5-volt power supply

VSS

Electrical ground

Table 20.2

Intel 8085 External Signals (page 2 of 2)

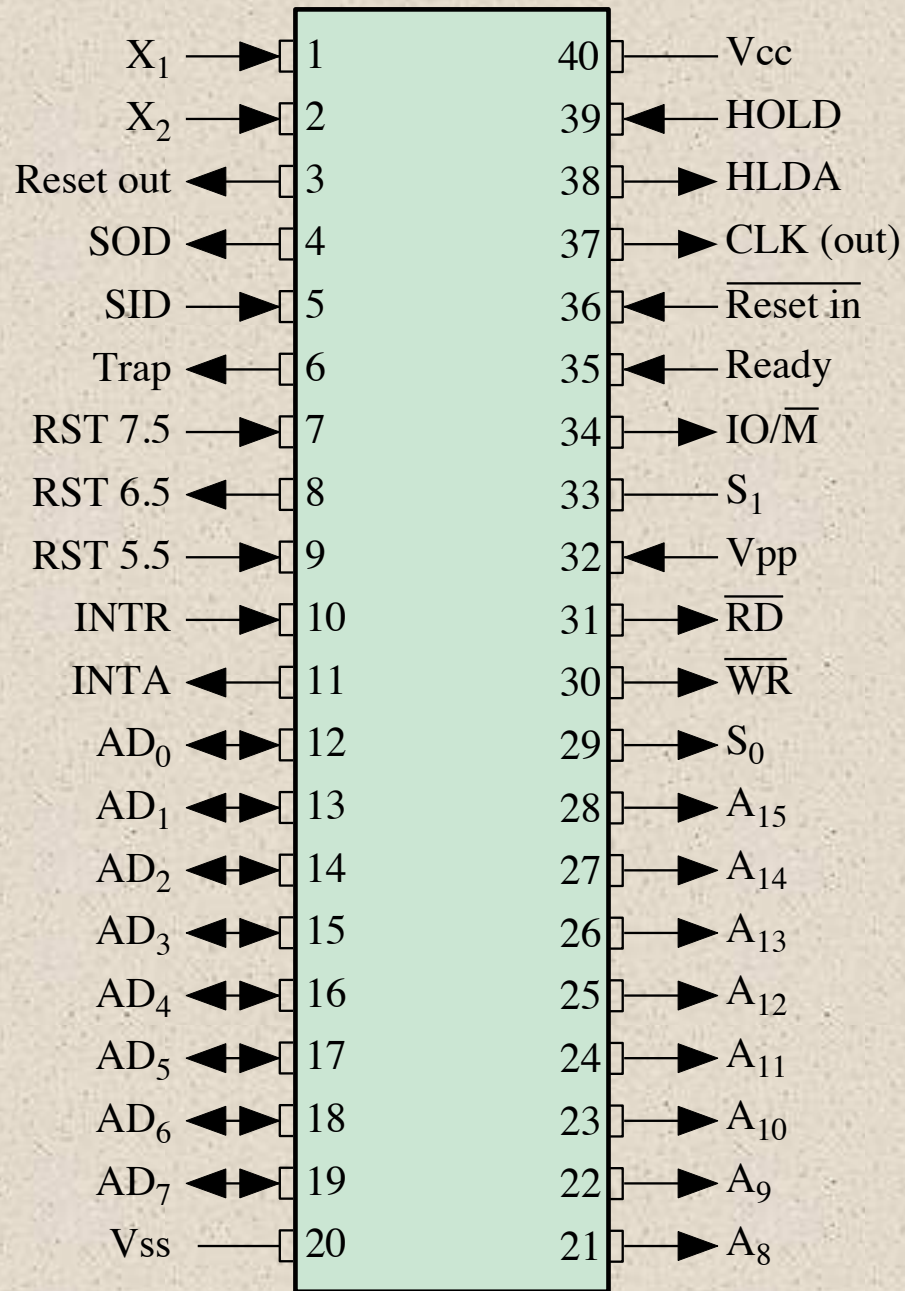


Figure 20.8 Intel 8085 Pin Configuration

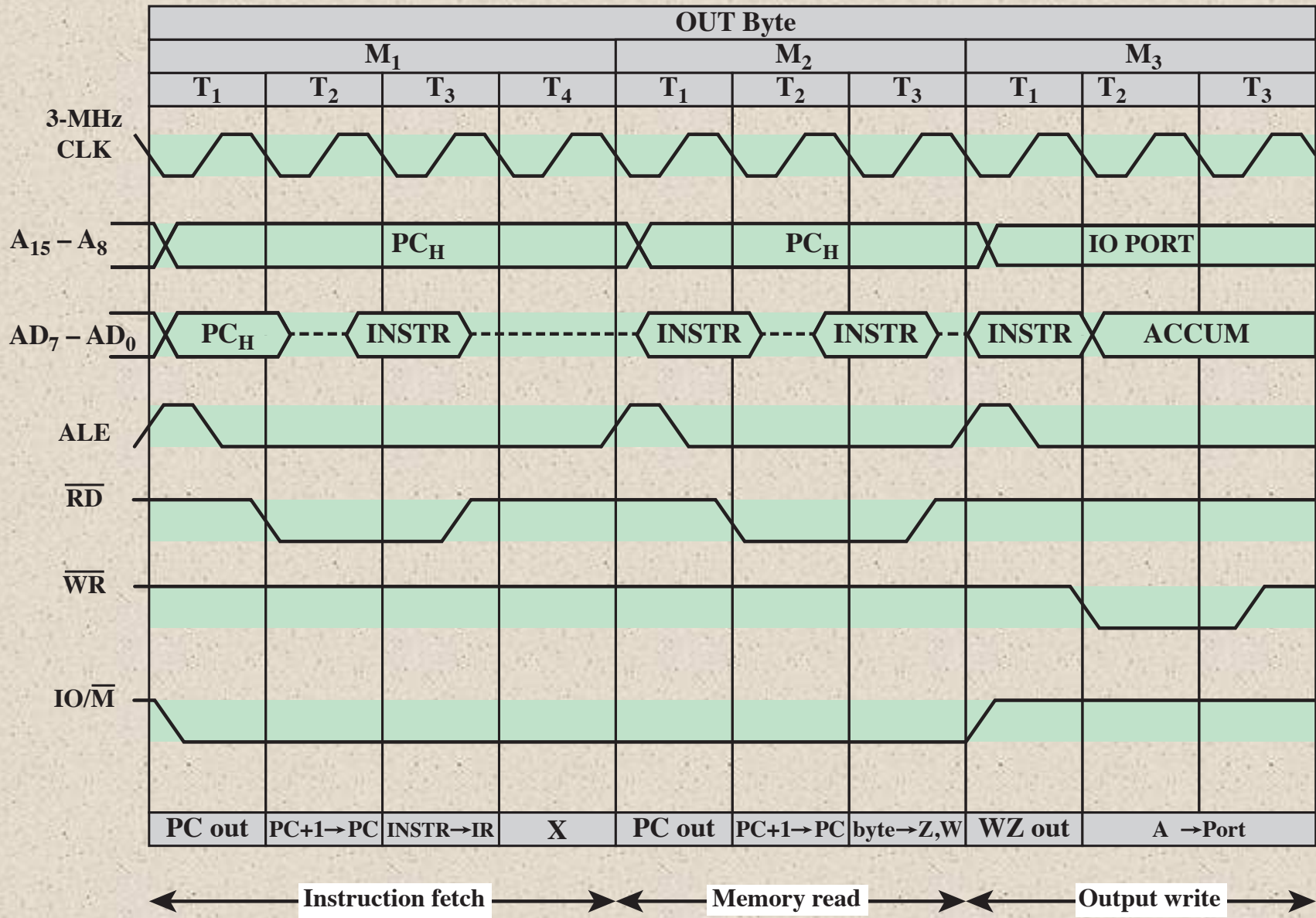


Figure 20.9 Timing Diagram for Intel 8085 OUT Instruction

I1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
I2	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
I3	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
I4	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
O1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
O2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
O3	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
O4	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
O5	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
O6	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
O7	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
O8	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
O9	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
O10	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
O11	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
O12	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
O13	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
O14	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
O15	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
O16	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table
20.3

A
Decoder
With
Four
Inputs
and
Sixteen
Outputs

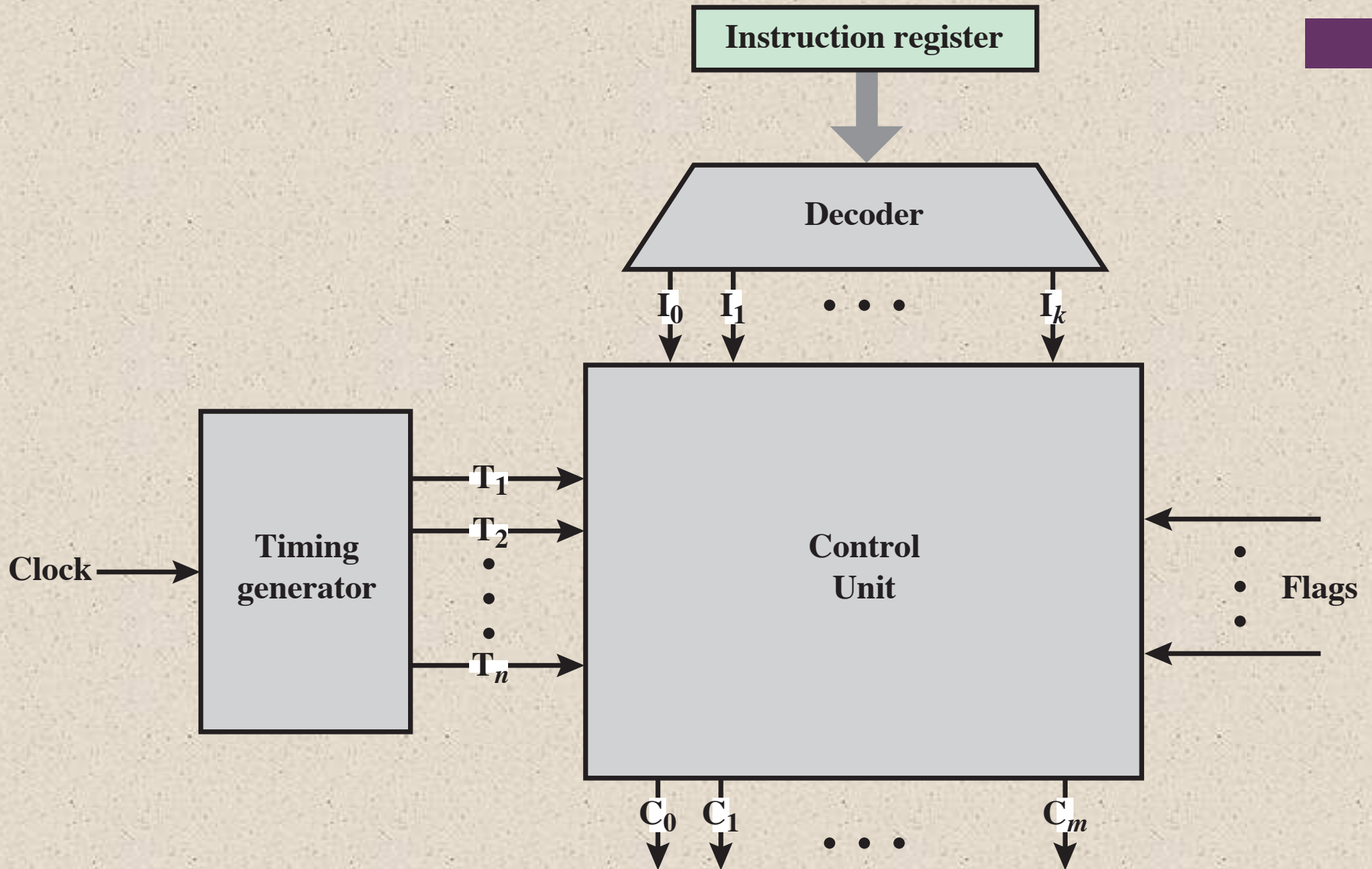


Figure 20.10 Control Unit with Decoded Inputs

+ Summary

Chapter 20

■ Micro-operations

- The fetch cycle
- The indirect cycle
- The interrupt cycle
- The execute cycle
- The instruction cycle

Control Unit Operation

- Control of the processor
 - Functional requirements
 - Control signals
 - Internal processor organization
 - The Intel 8085
- Hardwired implementation
 - Control unit inputs
 - Control unit logic