Common Bus System

- The basic computer has eight registers, a memory unit, and a control unit (to be presented in Sec. 5-4).
- Paths must be provided to transfer information from one register to another and between memory and registers.
- The number of wires will be excessive if connections are made between the outputs of each register and the inputs of the other registers.
- A more efficient scheme for transferring information in a system with many registers is to use a common bus.

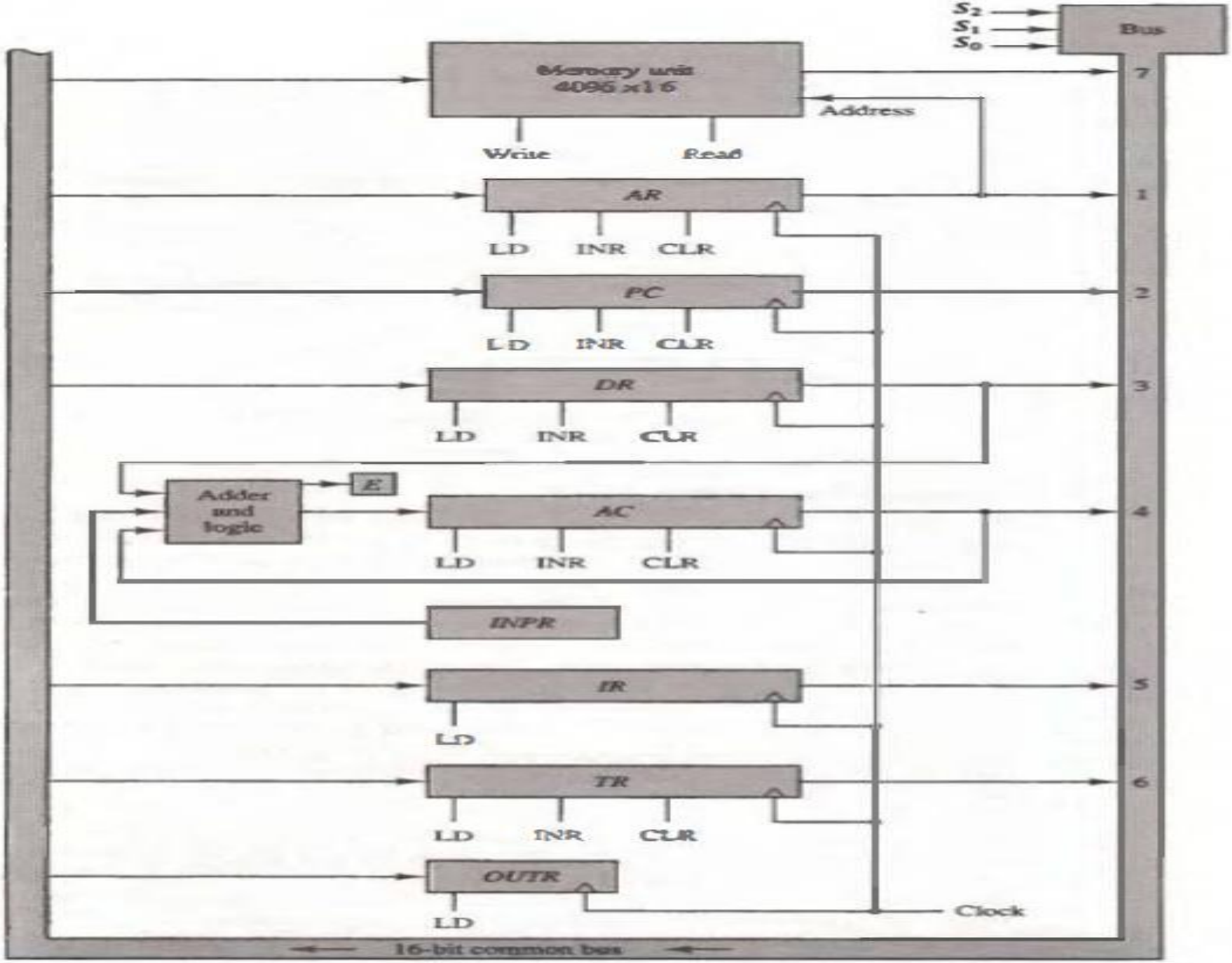


Figure 5-4 Basic computer registers connected to a common bus.

Common Bus System

- The connection of the registers and memory of the basic computer to a common bus system is shown in Fig. 5-4.

- The outputs of seven registers and memory are connected to the common bus. The specific output that is selected for the bus lines at any given time is determined from the binary value of the selection variables S2, S1, and S0.

- The number along each output shows the decimal equivalent of the required binary selection. For example, the number along the output of DR is 3. The 16-bit outputs of DR are placed on the bus lines when S2S1S0 = 011 since this is the binary value of decimal 3.

Common Bus System

- The lines from the common bus are connected to the inputs of each register and the data inputs of the memory.
- The particular register whose LD (load) input is enabled receives the data from the bus during the next clock pulse transition.
- The memory receives the contents of the bus when its write input is activated.
- The memory places its 16-bit output onto the bus when the read input is activated and S2S1S0 = 1 1 1 .

Common Bus System

- Four registers, D R , AC, IR, and TR, have 16 bits each. Two registers, AR and PC, have 1 2 bits each since they hold a memory address.

- When the contents of AR or PC are applied to the 16-bit common bus, the four most significant bits are set to O’s.

- When AR or PC receive information from the bus, only the 12 least significant bits are transferred into the register.

Common Bus System

- The input register INPR and the output register OUTR have 8 bits each and communicate with the eight bits in the bus.
- INPR is connected to provide information to the bus but OUTR can only receive information from the bus.
- This is because INPR receives a character from an input device which is then transferred to AC.
- OUTR receives a character from AC and delivers it to an output device.

Common Bus System

- The 16 lines of the common bus receive information from six registers and the memory unit. The bus lines are connected to the inputs of six registers and the memory.

- Five registers have three control inputs: LD (load), INR (increment), and CLR (clear). The increment operation is achieved by enabling the count input of the counter. Two registers have only a LD input.

- The input data and output data of the memory are connected to the common bus, but the memory address is connected to AR . Therefore, AR must always be used to specify a memory address.

- The content of any register can be specified for the memory data input during a write operation. Similarly, any register can receive the data from memory after a read operation.

Common Bus System

- The 16 inputs of AC come from an adder and logic circuit. This circuit has three sets of inputs. One set of 16-bit inputs come from the outputs of AC .

- They are used to implement register micro-operations such as complement AC and shift AC.
- Another set of 16-bit inputs come from the data register DR . The inputs from DR and AC are used for arithmetic and logic micro-operations, such as add DR to AC or AND DR to AC .

- The result of an addition is transferred to AC and the end carry-out of the addition is transferred to flip-flop E (extended AC bit). A third set of 8-bit inputs come from the input register INPR .

Common Bus System

- Note that the content of any register can be applied onto the bus and an operation can be performed in the adder and logic circuit during the same clock cycle. The clock transition at the end of the cycle transfers the content of the bus. For example, the two micro-operations

$$DR \leftarrow AC \quad \text{and} \quad AC \leftarrow DR$$

can be executed at the same time. This can be done by placing the content of AC on the bus (with $S_2S_1S_0 = 100$), enabling the LD (load) input of DR, transferring the content of DR through the adder and logic circuit into AC, and enabling the LD (load) input of AC, all during the same clock cycle. The two transfers occur upon the arrival of the clock pulse transition at the end of the clock cycle.

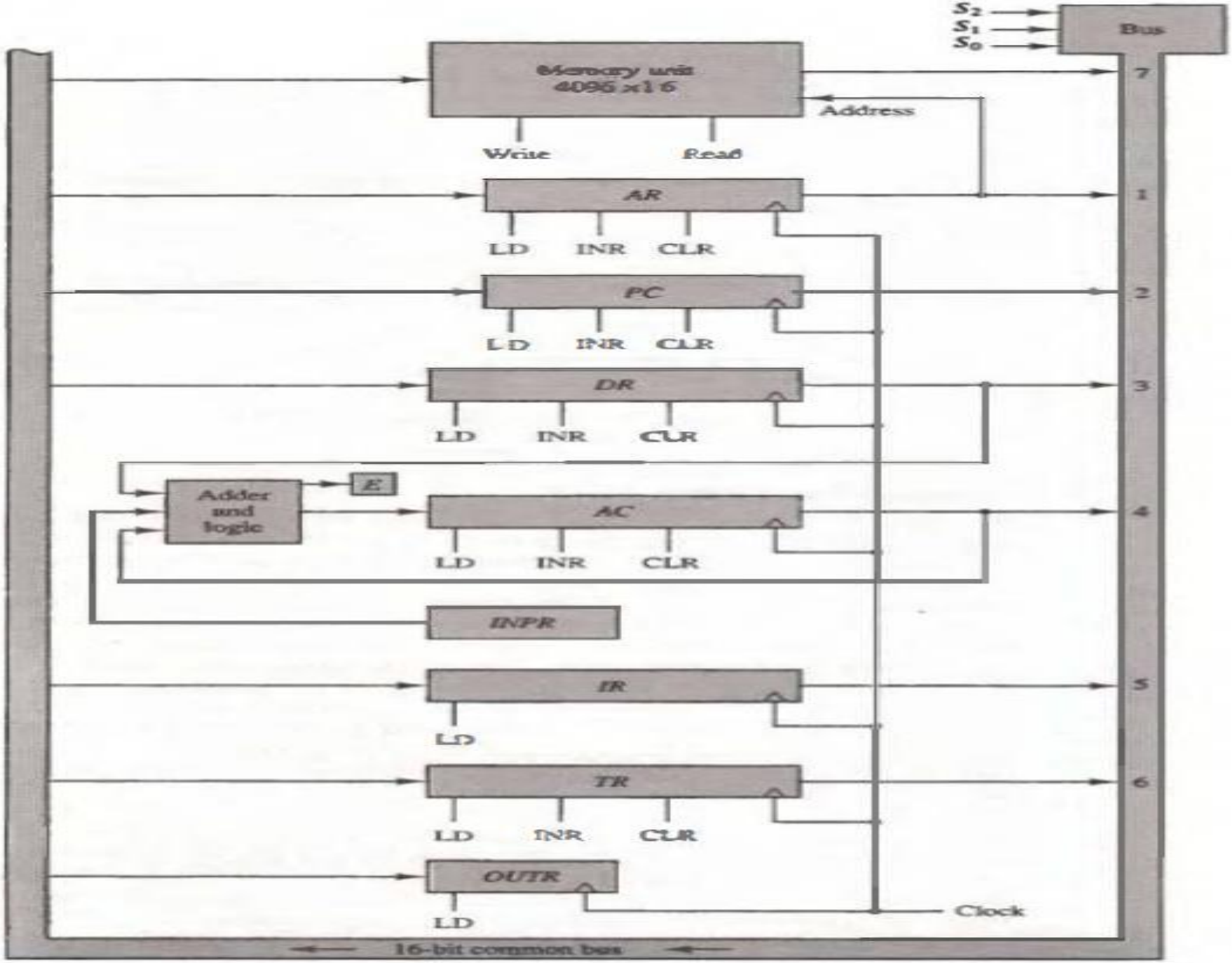


Figure 5-4 Basic computer registers connected to a common bus.

Computer Instructions

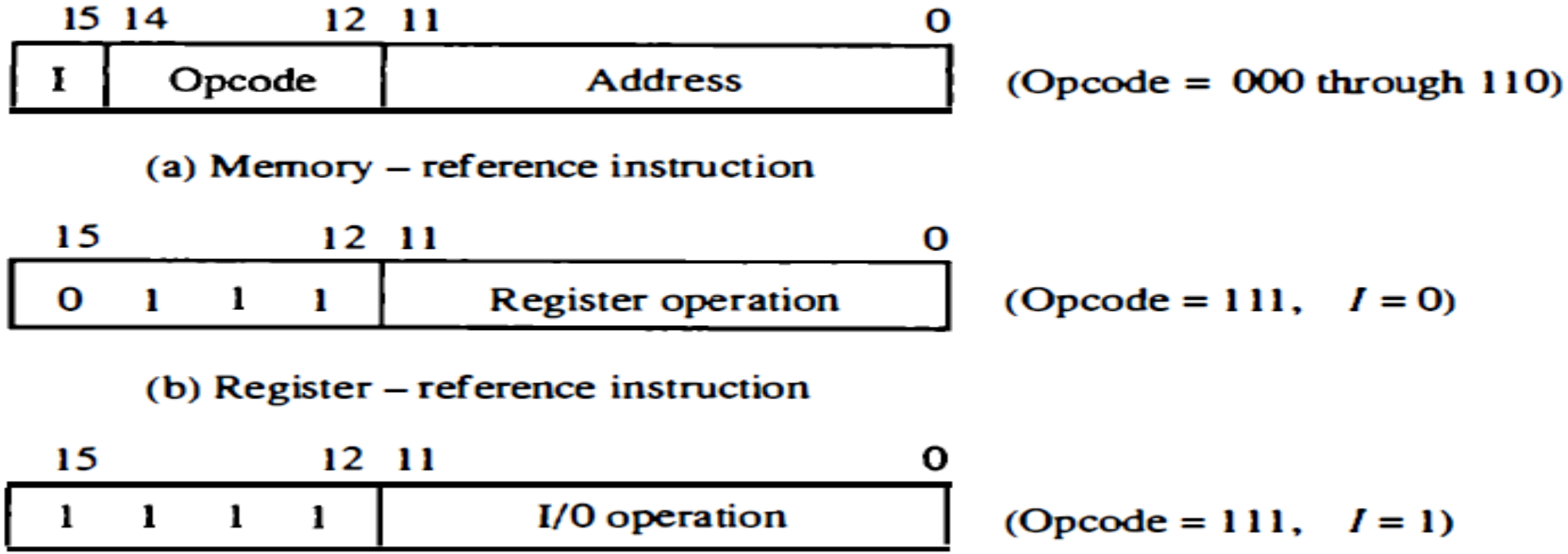
- The basic computer has three instruction code formats, as shown in Fig. 5-5. Each format has 16 bits.

- The operation code (opcode) part of the instruction contains three bits and the meaning of the remaining 13 bits depends on the operation code encountered.

- A memory-reference instruction uses 12 bits to specify an address and one bit to specify the addressing mode I. I is equal to 0 for direct address and to 1 for indirect address.

Computer Instructions

Figure 5-5 Basic computer instruction formats.



Computer Instructions

- The register reference instructions are recognized by the operation code Ill with a 0 in the leftmost bit (bit 15) of the instruction.

- A register-reference instruction specifies an operation on or a test of the AC register. An operand from memory is not needed; therefore, the other 12 bits are used to specify the operation or test to be executed.

- Similarly, an input-output instruction is recognized by the operation code 111 with a 1 in the leftmost bit of the instruction. The remaining 12 bits are used to specify the type of input-output operation or test performed.

Computer Instructions

- The type of instruction is recognized by the computer control from the four bits in positions 12 through 15 of the instruction.
- If the three opcode bits in positions 12 though 14 are not equal to Ill, the instruction is a memory-reference type and the bit in position 15 is taken as the addressing mode I.
- If the 3-bit opcode is equal to 111, control then inspects the bit in position 15. If this bit is 0, the instruction is a register-reference type.
- If the bit is 1, the instruction is an input--output type. Note that the bit in position 15 of the instruction code is designated by the symbol I but is not used as a mode bit when the operation code is equal to 1 1 1 .

Computer Instructions

- The type of instruction is recognized by the computer control from the four bits in positions 12 through 15 of the instruction.

- The instructions for the computer are listed in Table 5-2.

TABLE 5-2 Basic Computer Instructions

Symbol	Hexadecimal code $I = 0$	Hexadecimal code $I = 1$	Description
AND	0xxx	8xxx	AND memory word to AC
ADD	1xxx	9xxx	Add memory word to AC
LDA	2xxx	Axxx	Load memory word to AC
STA	3xxx	Bxxx	Store content of AC in memory
BUN	4xxx	Cxxx	Branch unconditionally
BSA	5xxx	Dxxx	Branch and save return address
ISZ	6xxx	Exxx	Increment and skip if zero
CLA	7800		Clear AC
CLE	7400		Clear E
CMA	7200		Complement AC
CME	7100		Complement E
CIR	7080		Circulate right AC and E
CIL	7040		Circulate left AC and E
INC	7020		Increment AC
SPA	7010		Skip next instruction if AC positive
SNA	7008		Skip next instruction if AC negative
SZA	7004		Skip next instruction if AC zero
SZE	7002		Skip next instruction if E is 0
HLT	7001		Halt computer
INP	F800		Input character to AC
OUT	F400		Output character from AC
SKI	F200		Skip on input flag
SKO	F100		Skip on output flag
ION	F080		Interrupt on
IOF	F040		Interrupt off

Symbol	Description
SKI	These are the status flags. They skip the next instructions when flag = 1
SKO	It is similar to SKI.

An interrupt temporarily stops or terminates a service or a current process.

Computer Instructions

- The hexadecimal code is equal to the equivalent hexadecimal number of the binary code used for the instruction.

- By using the hexadecimal equivalent we reduced the 16 bits of an instruction code to four digits with each hexadecimal digit being equivalent to four bits. A memory-reference instruction has an address part of 12 bits.

- The address part is denoted by three x's and stand for the three hexadecimal digits corresponding to the 12-bit address.

- The last bit of the instruction is designated by the symbol I. When I = 0, the last four bits of an instruction have a hexadecimal digit equivalent from 0 to 6 since the last bit is 0.
- When I = I, the hexadecimal digit equivalent of the last four bits of the instruction ranges from 8 to E since the last bit is I.

Computer Instructions

- Register-reference instructions use 16 bits to specify an operation. The leftmost four bits are always 0111, which is equivalent to hexadecimal 7.
- The other three hexadecimal digits give the binary equivalent of the remaining 12 bits.
- The input-output instructions also use all 16 bits to specify an operation.
- The last four bits are always 1 1 1 1 , equivalent to hexadecimal F.

Timing and Control

- The timing for all registers in the basic computer is controlled by a master clock generator.
- The clock pulses are applied to all flip-flops and registers in the system, including the flip-flops and registers in the control unit.
- The clock pulses do not change the state of a register unless the register is enabled by a control signal.
- The control signals are generated in the control unit and provide control inputs for the multiplexers in the common bus, control inputs in processor registers, and microoperations for the accumulator.

Timing and Control

- There are two major types of control organization: hardwired control and microprogrammed control.
- In the hardwired organization, the control logic is implemented with gates, flip-flops, decoders, and other digital circuits.
- It has the advantage that it can be optimized to produce a fast mode of operation.
- In the microprogrammed organization, the control information is stored in a control memory.
- The control memory is programmed to initiate the required sequence of microoperations.

Timing and Control

- A hardwired control, as the name implies, requires changes in the wiring among the various components if the design has to be modified or changed.

- In the microprogrammed control, any required changes or modifications can be done by updating the microprogram in control memory.

Control unit

- The block diagram of the control unit is shown in Fig. 5-6.
- It consists of two decoders, a sequence counter, and a number of control logic gates.
- The position of this register in the common bus system is indicated in Fig. 5-4.
- The instruction register is shown again in Fig. 5-6, where it is divided into three parts: the I bit, the operation code, and bits 0 through 1 1 .
- The operation code in bits 12 through 14 are decoded with a 3 x 8 decoder.

Control unit

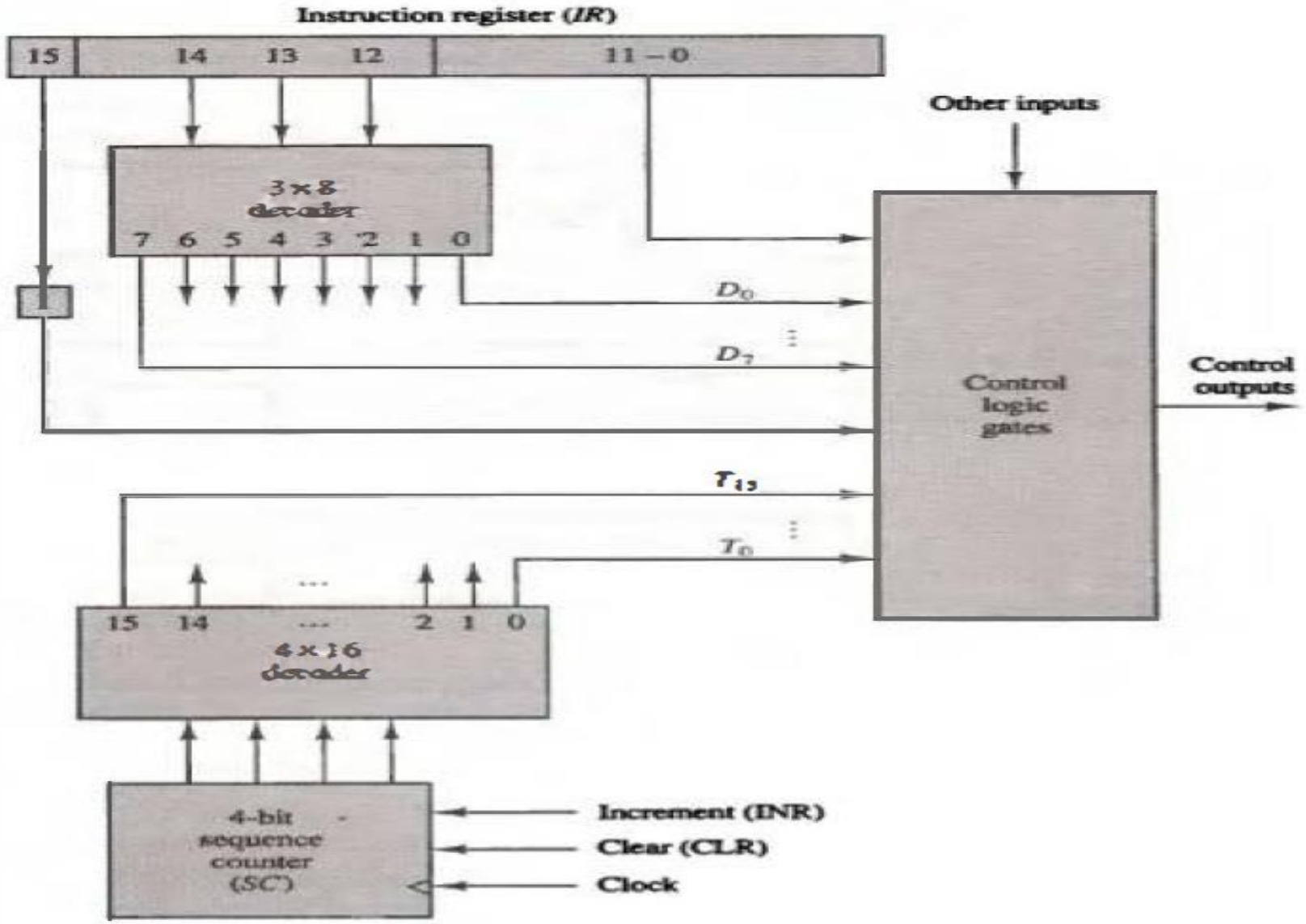


Figure 5-6 Control unit of basic computer.

Control unit

- The eight outputs of the decoder are designated by the symbols D0 through D7.
- The subscripted decimal number is equivalent to the binary value of the corresponding operation code.
- Bit 15 of the instruction is transferred to a flip-flop designated by the symbol I. Bits 0 through 11 are applied to the control logic gates.
- The 4-bit sequence counter can count in binary from 0 through 15. The outputs of the counter are decoded into 16 timing signals T0 through T15.

Control unit

- The sequence counter SC can be incremented or cleared synchronously.
- Most of the time, the counter is incremented to provide the sequence of timing signals out of the 4 x 16 decoder.
- Once in awhile, the counter is cleared to 0, causing the next active timing signal to be To.

Control unit

T_0. As an example, consider the case where SC is incremented to provide timing signals T_0, T_1, T_2, T_3, and T_4 in sequence. At time T_4, SC is cleared to 0 if decoder output D_3 is active. This is expressed symbolically by the statement

$$D_3T_4: \quad SC \leftarrow 0$$

Control unit

- The timing diagram of Fig. 5-7 shows the time relationship of the control signals.
- The sequence counter SC responds to the positive transition of the clock.
- Initially, the CLR input of SC is active.

Control unit

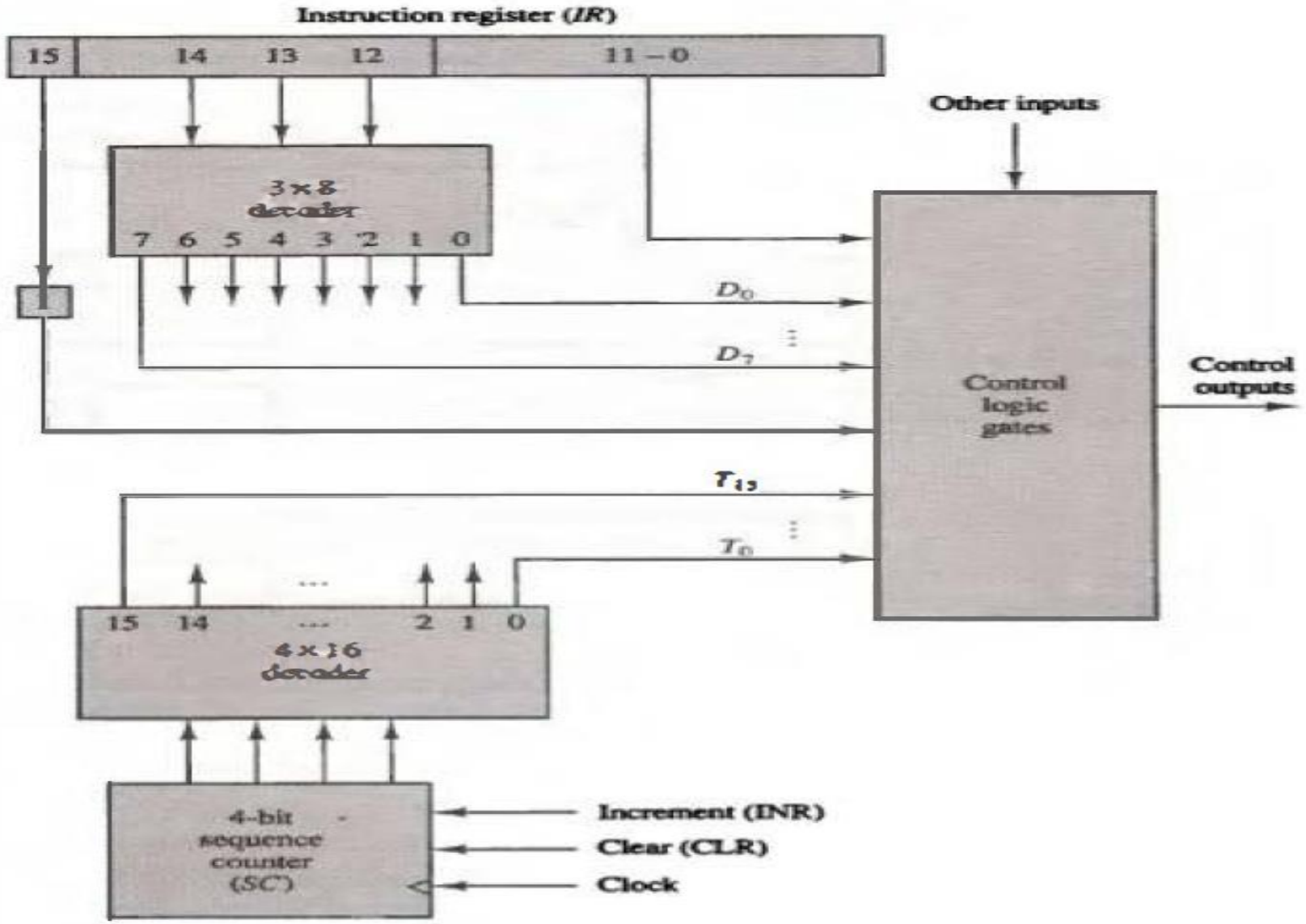


Figure 5-6 Control unit of basic computer.

Control unit

- clock clears SC to 0, which in tum activates the timing signal T0 out of the decoder. T0 is active during one clock.

- The positive clock transition labeled T0 in the diagram will trigger only those registers whose control inputs are connected to timing signal To.

- SC is incremented with every positive clock transition, unless its CLR input is active. This produces the sequence of timing signals To, T1, T2, T3, T4, and so on, as shown in the diagram. (Note the relationship between the timing signal) and its corresponding positive clock transition.)

- If SC is not cleared, the timing signals will continue with T5, T6, up to T15 and back T0.

Control unit

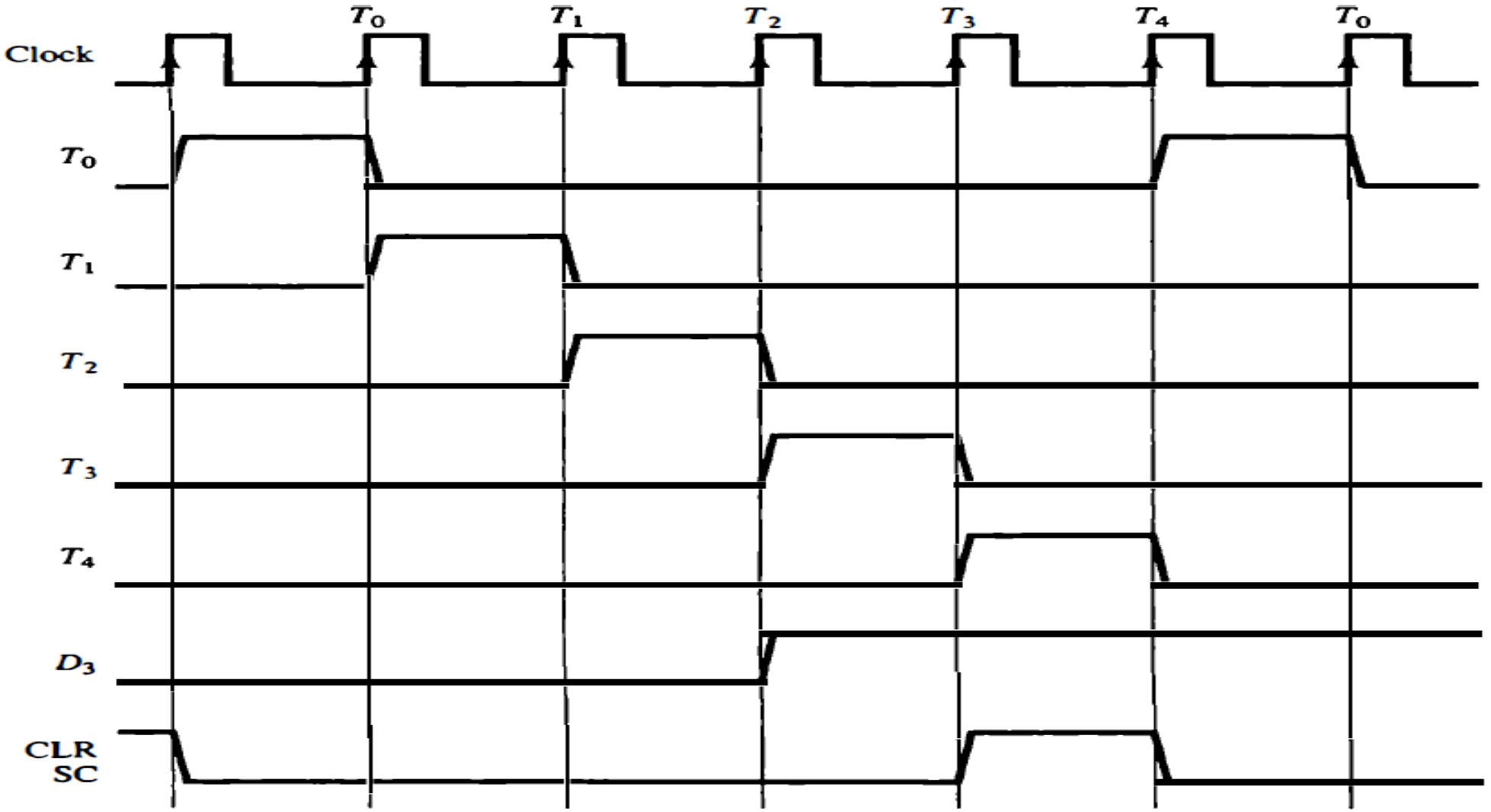


Figure 5-7 Example of control timing signals.

Control unit

- The last three waveforms in Fig. 5-7 show how SC is cleared when D3T4 = 1.
- Output D3 from the operation decoder becomes active at the end of timing signal T2.
- When timing signal T4 becomes active, the output of the AND gate that implements the control function D3T4 becomes active.
- This signal is applied to the CLR input of SC. On the next positive clock transition (the one marked T, in the diagram) the counter is cleared to 0.
- This causes the timing signal To to become active instead of T5.

Control unit

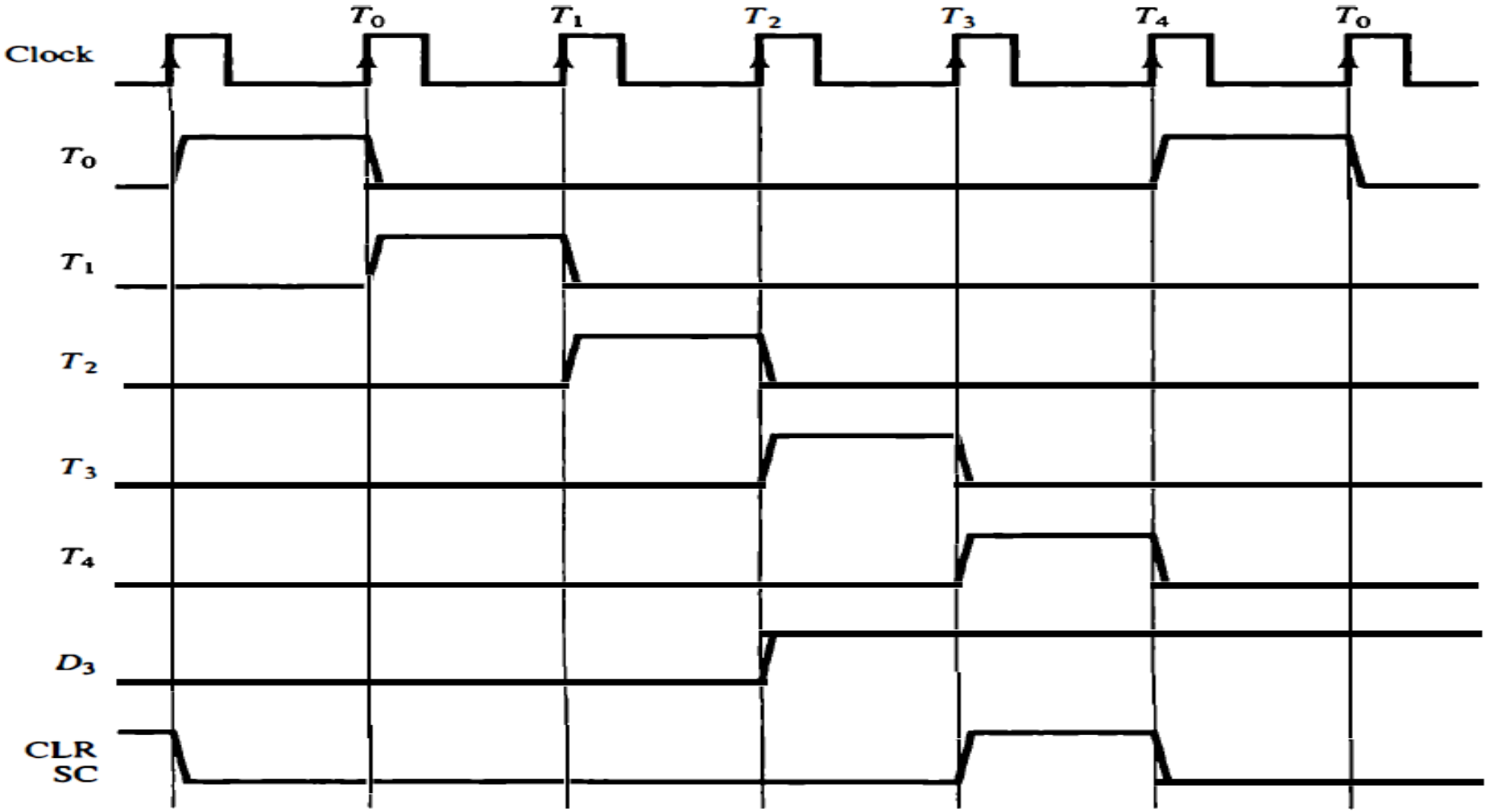


Figure 5-7 Example of control timing signals.

Control unit

- A memory read or write cycle will be initiated with the rising edge of a timing signal.
- It will be assumed that a memory cycle time is less than the clock cycle time.
- According to this assumption, a memory read or write cycle initiated by a timing signal will be completed by the time the next clock goes through its positive transition.
- The clock transition will then be used to load the memory word into a register.
- This timing relationship is not valid in many computers because the memory cycle time is usually longer than the processor clock cycle.
- In such a case it is necessary to provide wait cycles in the processor

Control unit

- until the memory word is available. To facilitate the presentation, we will assume that a wait period is not necessary in the basic computer.

To fully comprehend the operation of the computer, it is crucial that one understands the timing relationship between the clock transition and the timing signals. For example, the register transfer statement

$$T_0: \quad AR \leftarrow PC$$

Control unit

- specifies a transfer of the content of PC into AR if timing signal To is active.
- T0 is active during an entire clock cycle interval During this time the content of PC is placed onto the bus (with S2S1S0 = 010) and the LD (load) input of AR is enabled.
- The actual transfer does not occur until the end of the clock cycle when the clock goes through a positive transition.
- This same positive clock transition increments the sequence counter SC from 0000 to 0001 . The next clock cycle has T1 active and T0 inactive.