# Basic Computer Organization

Recitation 2

CENG 232 METU

Volkan Atalay

2020

(during COVID-19 lockdown)

| $S_2$ | $S_1$ | $S_0$ | Register |
|---|---|---|---|
| 0 | 0 | 0 | x |
| 0 | 0 | 1 | AR |
| 0 | 1 | 0 | PC |
| 0 | 1 | 1 | DR |
| 1 | 0 | 0 | AC |
| 1 | 0 | 1 | IR |
| 1 | 1 | 0 | TR |
| 1 | 1 | 1 | Memory |

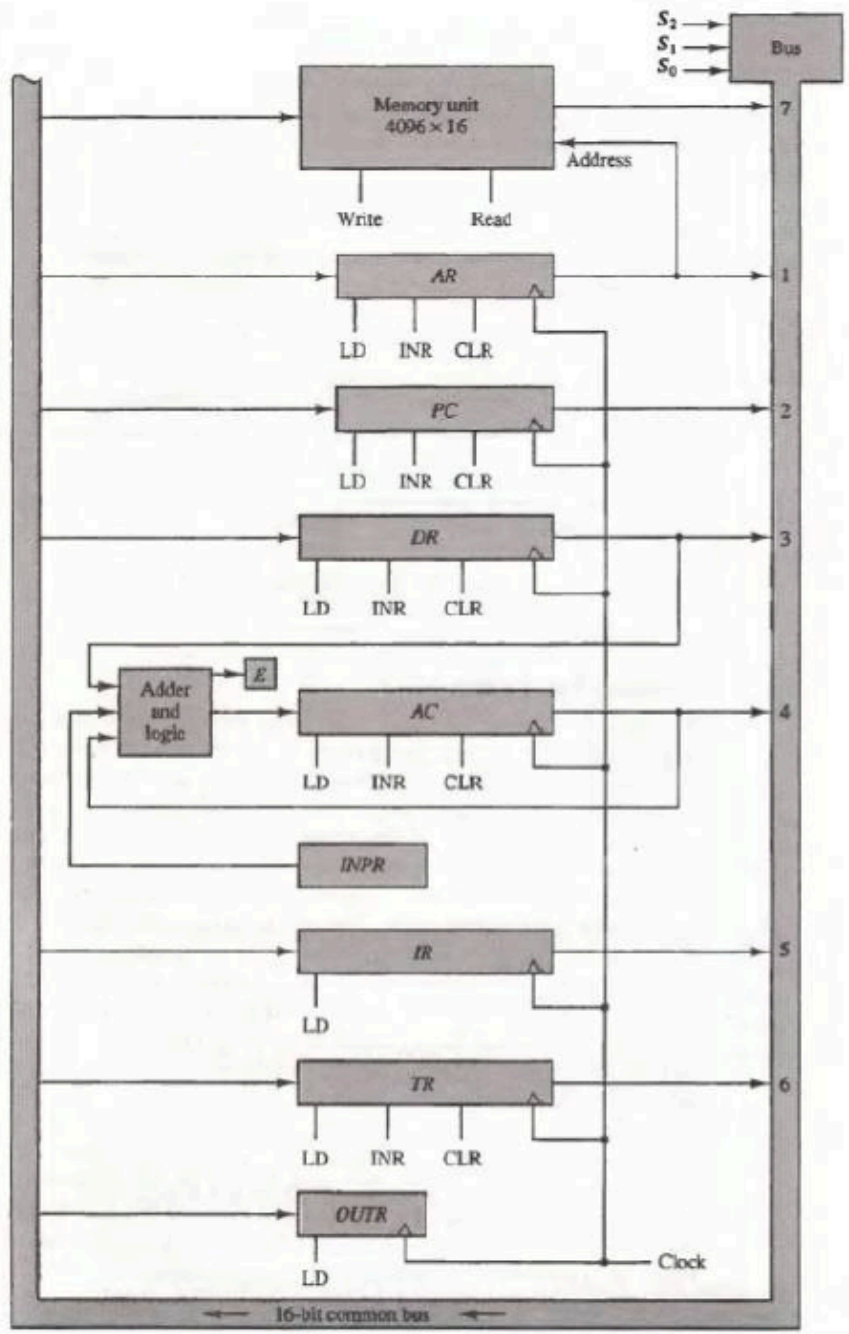



**Figure 5-4** Basic computer registers connected to a common bus.

5.4 The following register transfers are to be executed in the system of Figure 5.4. For each transfer, specify: 1) the binary value that must be applied to bus select inputs $S_2$, $S_1$, and $S_0$; 2) the register whose LD control must be active (if any); 3) a memory read or write operation (if needed); and 4) the operation in the adder and logic circuit (if any).

a. AR <- PC
b. IR <- M[AR]
c. M[AR] <- TR
d. AC <-DR, DR <- AC (done simultaneously)

| $S_2$ | $S_1$ | $S_0$ | Register |
|---|---|---|---|
| 0 | 0 | 0 | x |
| 0 | 0 | 1 | AR |
| 0 | 1 | 0 | PC |
| 0 | 1 | 1 | DR |
| 1 | 0 | 0 | AC |
| 1 | 0 | 1 | IR |
| 1 | 1 | 0 | TR |
| 1 | 1 | 1 | Memory |

| | | $S_2$ | $S_1$ | $S_0$ | LD of register | Memory | Adder |
|---|---|---|---|---|---|---|---|
| **a** | AR <- PC | | | | | | |
| **b** | IR <- M[AR] | | | | | | |
| **c** | M[AR] <- TR | | | | | | |
| **d** | AC <-DR, | | | | | | |
| | DR <- AC | | | | | | |

5-5. Explain why each of the following microoperations cannot be executed during a single clock pulse in the system shown in Fig. 5-4. Specify a sequence of microoperations that will perform the operation.

a. IR <- M[PC]

b. AC <- AC+ TR

c. DR <- DR + AC (AC does not change)

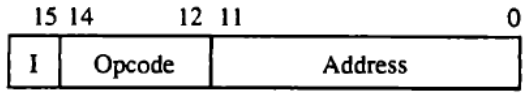
5-6 Consider the instruction formats of the basic computer shown in Fig. 5-5 and the list of instructions given in Table 5-2. For each of the following 16 bit instructions, give the equivalent four digit hexadecimal code and explain in your own words what it is that the instruction is going to perform.

a. 0001 0000 0010 0100

b. 1011 0001 0010 0100

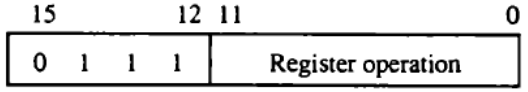
c. 0111 0000 0010 0000

Figure 5-5 Basic computer instruction formats.



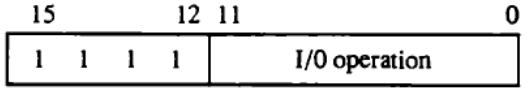

(Opcode = 000 through 110)

(a) Memory – reference instruction




(Opcode = 111, $I = 0$)

(b) Register – reference instruction




(Opcode = 111, $I = 1$)

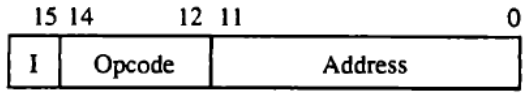
(c) Input – output instruction

0001 0000 0010 0100

TABLE 5-2 Basic Computer Instructions

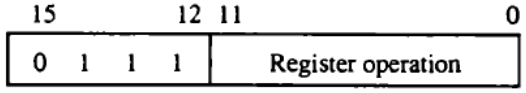
| Symbol | Hexadecimal code $I = 0$ | Hexadecimal code $I = 1$ | Description |
|---|---|---|---|
| AND | 0xxx | 8xxx | AND memory word to *AC* |
| ADD | 1xxx | 9xxx | Add memory word to *AC* |
| LDA | 2xxx | Axxx | Load memory word to *AC* |
| STA | 3xxx | Bxxx | Store content of *AC* in memory |
| BUN | 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 |

Figure 5-5 Basic computer instruction formats.



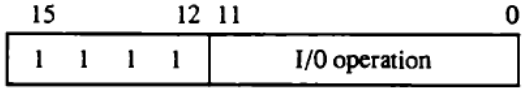

(Opcode = 000 through 110)

(a) Memory – reference instruction




(Opcode = 111, $I = 0$)

(b) Register – reference instruction




(Opcode = 111, $I = 1$)

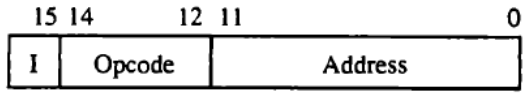
(c) Input – output instruction

1011 0001 0010 0100

TABLE 5-2 Basic Computer Instructions

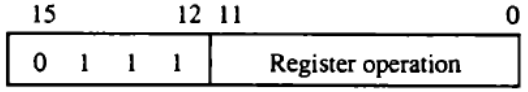
| Symbol | Hexadecimal code $I = 0$ | Hexadecimal code $I = 1$ | Description |
|---|---|---|---|
| AND | 0xxx | 8xxx | AND memory word to *AC* |
| ADD | 1xxx | 9xxx | Add memory word to *AC* |
| LDA | 2xxx | Axxx | Load memory word to *AC* |
| STA | 3xxx | Bxxx | Store content of *AC* in memory |
| BUN | 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 |

Figure 5-5 Basic computer instruction formats.



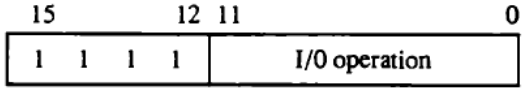

(Opcode = 000 through 110)

(a) Memory – reference instruction




(Opcode = 111, $I = 0$)

(b) Register – reference instruction




(Opcode = 111, $I = 1$)

(c) Input – output instruction

0111 0000 0010 0000

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 |

5-7. What are the two instructions needed in the basic computer in order to set the E flip-flop to 1?