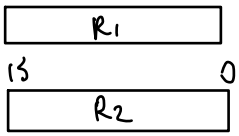
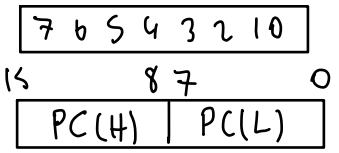
Ch. 4

MAR → Memory address register → Holds address for memory unit.

Block diagram of registers

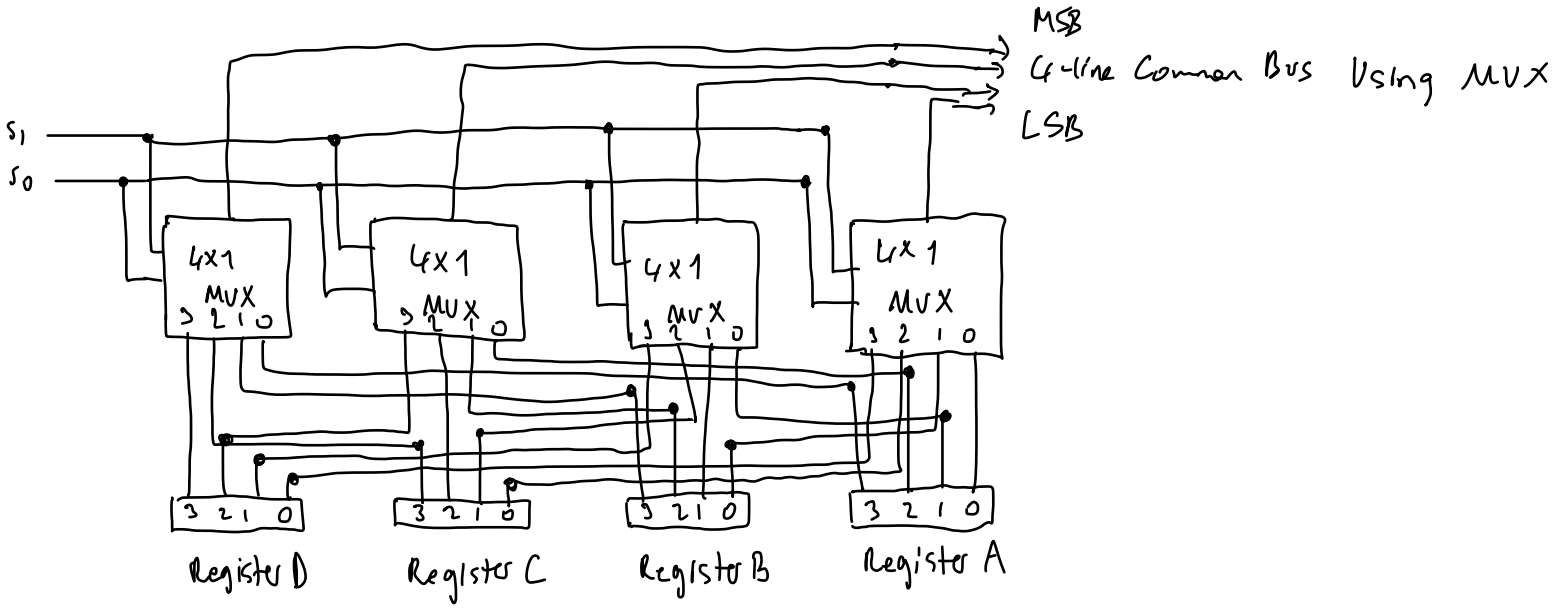


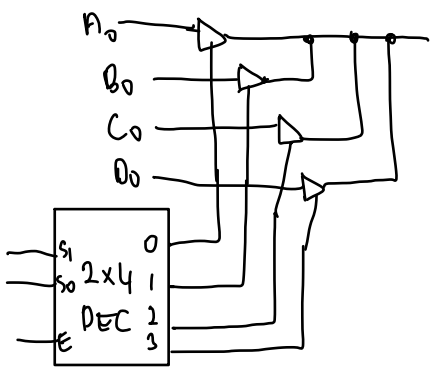


PC(H) → PC(15-8)
PC(L) → PC(7-0)

. P: $R_2 \leftarrow R_1$ means that if P=1, then transfer R_1 into R_2.

. T: $R_2 \leftarrow R_1$, $R_1 \leftarrow R_2$ comma is used to perform operations simultaneously. This operation exchanges the datas in registes.





Bus line for Bit 0 using 3-state buffer

- Selective Set → $A \leftarrow A \vee B$ the B's are used to perform selected operation.
- Selective Complement → $A \leftarrow A \oplus B$
- Selective Clear → $A \leftarrow A \wedge \overline{B}$
- Mask → $A \leftarrow A \wedge B$
- Insert → $A \leftarrow A \wedge B$ then $A \leftarrow A \vee B$ (Those two B's are different.)
- Clear → $A \leftarrow A \oplus B$

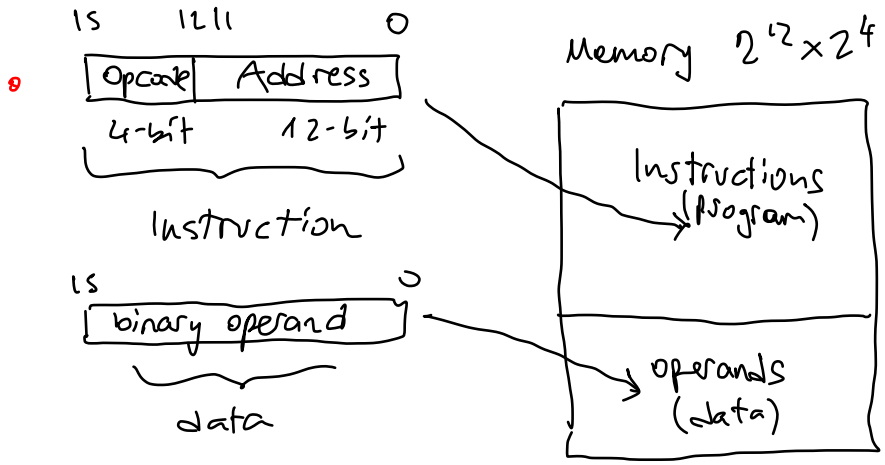
Shift operations

- $R_1 \leftarrow$ shl R_1 → $R_7 R_6 \cdots\cdots R_1 R_0$ → $R_6 R_5 \cdots\cdots R_1 R_0\ 0$ R_7 is lost
- $R_2 \leftarrow$ shr R_2 → $R_2 R_6 \cdots\cdots R_1 R_0$ → $0\ R_7 R_6 \cdots\cdots R_1$ R_0 is lost

- $R_1 \leftarrow$ cil R_1 → $R_7 R_6 \cdots\cdots R_1 R_0$ → $R_6 \cdots\cdots R_1 R_0 R_7$ } Nothing is lost.
- $R_2 \leftarrow$ cir R_2 → $R_7 R_6 \cdots\cdots R_1 R_0$ → $R_0 R_7 R_6 \cdots\cdots R_2 R_1$

- $R_1 \leftarrow$ ashl R_1 → $R_7 R_6 \cdots\cdots R_1 R_0$ → $R_6 R_5 \cdots\cdots R_1 R_0\ 0$ R_7 is lost. & $V_s = R_7 \oplus R_6$ (to detect overflow)
- $R_2 \leftarrow$ ashr R_2 → $R_7 R_6 \cdots\cdots R_1 R_0$ → $R_7 R_7 R_6 \cdots\cdots R_2 R_1$ R_0 is lost.

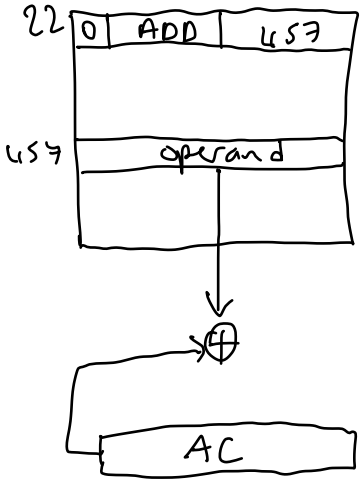
The purpose of arithmetic shift is remaining the sign same.
left → multiply, right → division



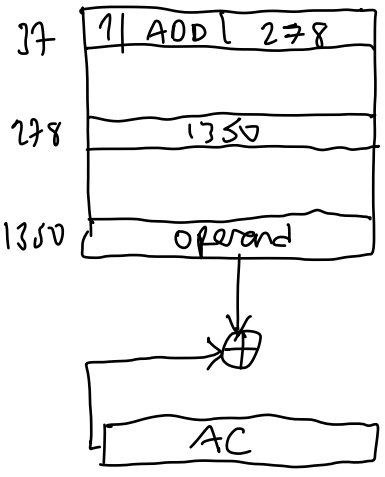
- The operation is performed with the memory operand and the content of AC (accumulator).
- Some instructions don't have the operand from memory. The 12-bit address used for other purposes such as clear AC or increment AC data.

- If the second part of the instruction is operand, the instruction is said to have an immediate operand.
- If the second part of the instruction is the addres of the operand, the instruction is said to have direct address.
- Indirect address, where the second part of the instruction depicts a address of a memory which shows the operand.
- We can specify whether the instruction holds direct or indirect address of operand from the 16-bit of instruction

If I=0 → Direct
I=1 → indirect



(direct address)



(indirect add.)

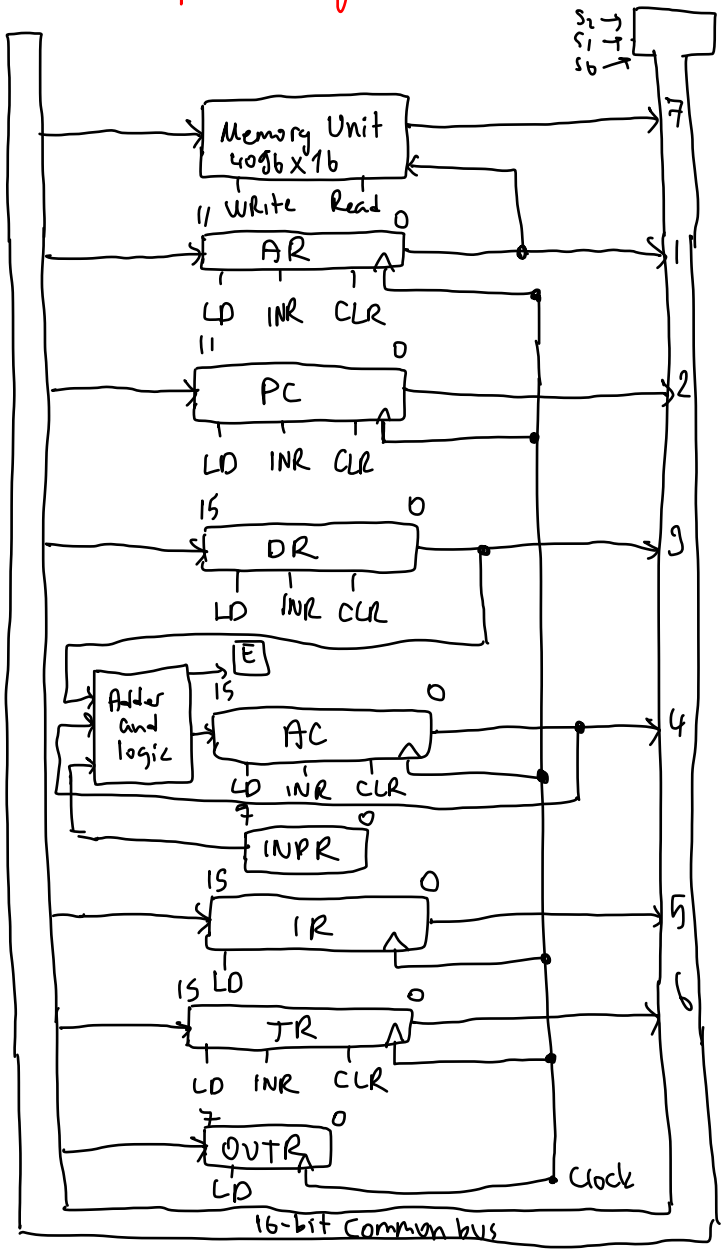
- The address that holds the operands is known as effective address.
- In the first figure the "457", and the second figure "1350" is the effective address.
- The indirect address generally stores pointer to an array.

the most common registers used in a basic computer

Register Symbol	Num of bits	Register Name
AC	16	Accumulator or Processor Reg.
IR	16	Instruction Register
PC	12	Program Counter
DR	16	Data Register
AR	12	Address Register
TR	16	Temporary Register
INPR	8	Input Register
OUTR	8	Output Register

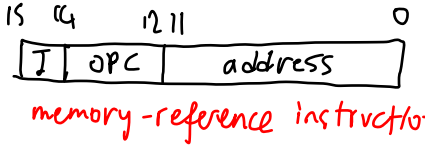
- AC → manipulates the data. It holds the data results from operations.
- IR → holds the instruction read from memory.
- DR → holds the operand read from memory.
- TR → holds 16-bit temporary data.
- INPR → holds 8-bit char. from input device
- OUTR → holds 8-bit char. for output device
- PC → It holds the address of next instruction. PC counting a sequence of instructions until a branch instruction is encountered. The address of branch instruction is transferred to the PC as a next instruction.

basic computer registers connected to a common bus

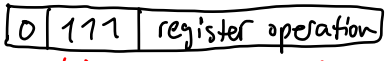


- When the 12-bit PC and AR contents are applied to the common bus, the four MSB are set to 0's. When AR and PC receives data from CB only the 12 LSB bits of CB are transferred.
- The INPR and OUTR communicates with the 8 LSB of CB.
- E is the carry-out from the addition of AC and DR.

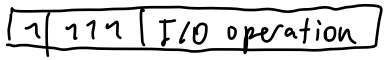
Instruction Formats



memory-reference instruction



register reference instruction



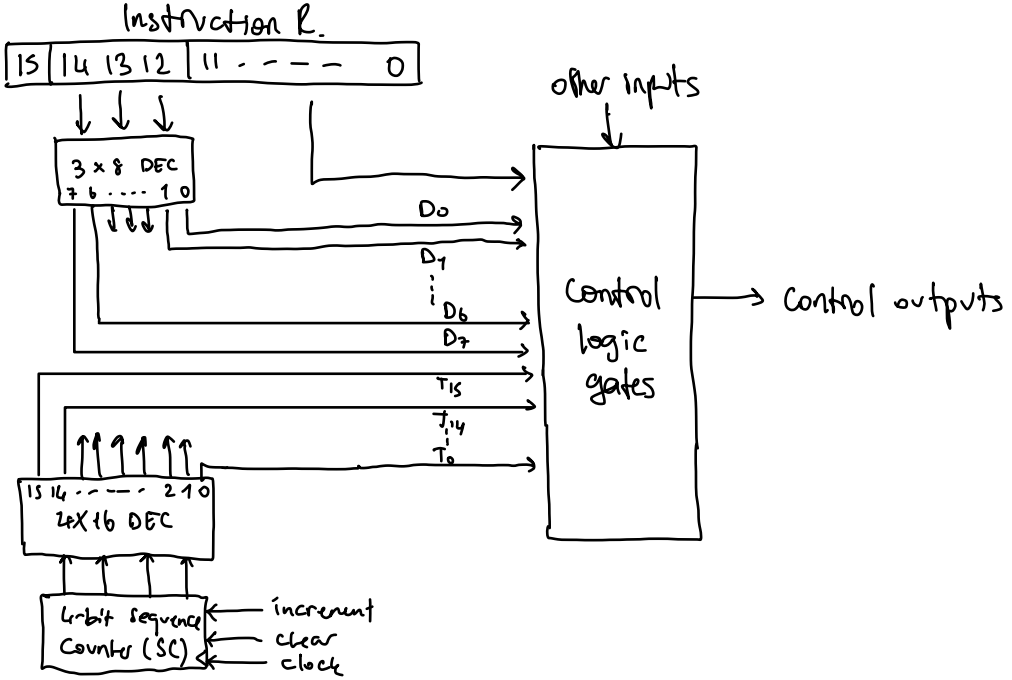
I/O instruction

opcode = [000, 110]

- There are 25 instruction for basic computer
- Instruction set is complete if it includes a sufficient number of instructions from following categories.

1) Arithmetic, logical and shift operations
2) Instructions for moving information to and from memory and AC.
3) Program control instructions together with instructions that check status conditions.
4) Input and output instructions

Control unit of basic computer



- $D_3T_4 : SC \leftarrow 0$
- SC responds to the clock signal. When 5 clock cycles passed if D3 is active then SC is set to 0.

- Instruction Cycle

1) Fetch the instruction
2) Decode
3) Read the effective address
4) Execute

This loop continues indefinitely unless a HALT instruction is encountered.

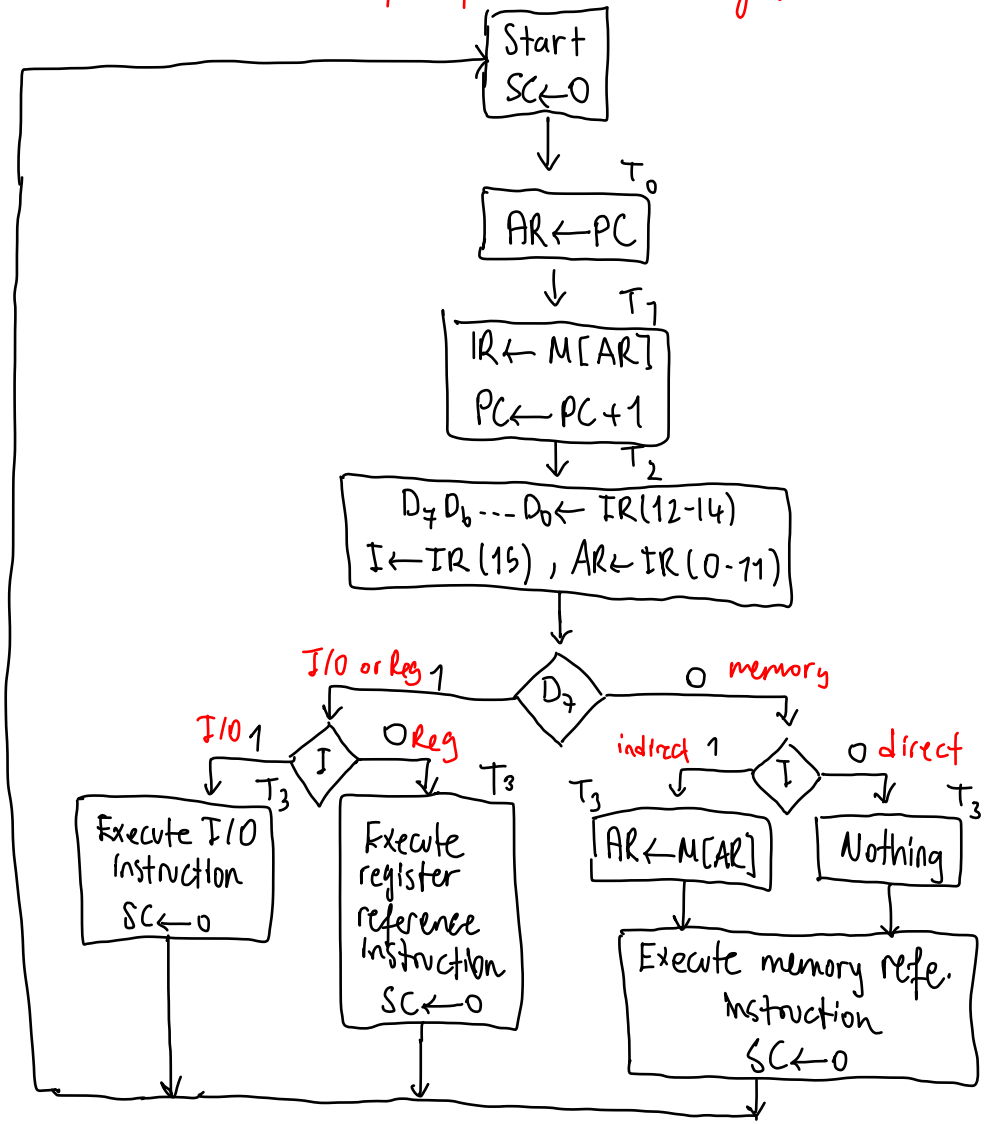
Fetch and Decode

T_0: $AR \leftarrow PC$

T_1: $IR \leftarrow M[AR]$, $PC \leftarrow PC + 1$

T_2: $D_0 \cdots D_7 \leftarrow$ Decode $IR(12\text{-}14)$, $AR \leftarrow IR(0\text{-}11)$, $I \leftarrow IR(15)$

Flowchart of Instruction Cycle



Execute

$D_7 I T_3$: Execute I/O instruction

$D_7 I' T_3$: Execute Register-reference instruction

$D_7' I T_3$: $AR \leftarrow M[AR]$

$D_7' I' T_3$: Nothing

Register-Reference Instructions

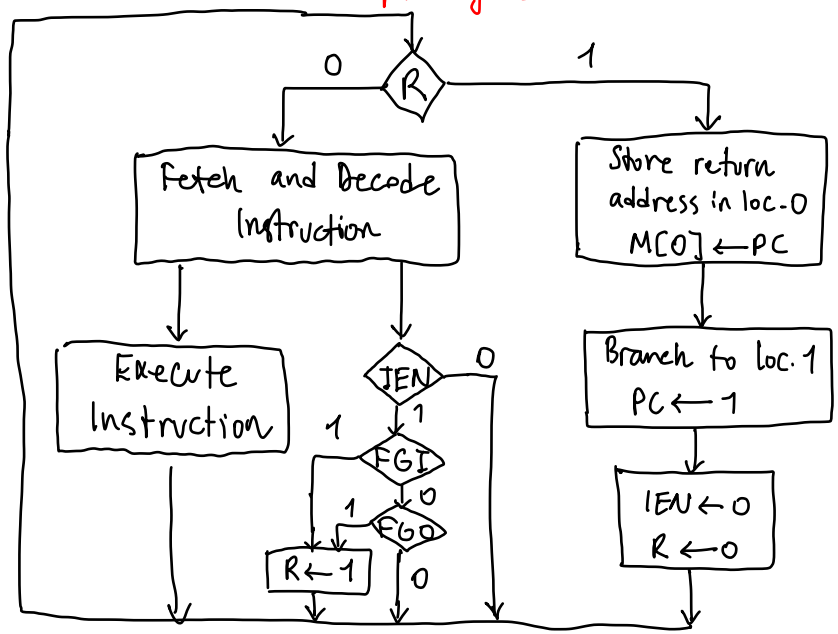
$r \leftarrow D_7 I' T_3$

- To select operation performed with register reference instr., the IR(11-0) bit determines which operation will be done.
- IR(11-0) = 0100 0000 0000 = B_{10} → The operation corresponding to $r B_{10}$ will be performed.
- There are 12 different operation for register-reference instruction.

Input/Output Instruction

- If (FGI=0) the information can be shifted into INPR. If it is happened then FGI is set to 1.
- If (FGI=1) the " in the INPR is shifted to AC. Then the FGI is set to 0.
- If (FGO=1) " " in the AC " " " OUTR. " " FGO " " " 0.
- If (FGO=0) " " in the OUTR printed out. " " FGO " " " 1.
- The initial value of FGI and FGO is 0 and 1, respectively.

Interrupt Cycle



```
        ORG 64
FETCH:  PCTAR           U    JMP   NEXT
        INCPC, READ     U    JMP   NEXT
        DRTAR           U    MAP
        ORG 67
INDRCT: READ            U    JMP   NEXT
        DRTAR           U    RET
        ORG 0
ADD:    NOP             I    CALL  INDRCT
        READ            U    JMP   NEXT
        ADD             U    JMP   FETCH
        ORG 4
BRANCH: NOP             S    JMP   OVER
        NOP             U    JMP   FETCH
OVER:   NOP             I    CALL  INDRCT
        ARTPC           U    JMP   FETCH
```

AR ← PC
PC ← PC+1 , DR ← M[AR]