F_2	Microoperation	Symbol
000	None	NOPE (No operation)
001	AC ← AC − DR	SUB
010	AC ← AC ∨ DR	OR
011	AC ← AC ∧ DR	AND
100	DR ← M[AR]	READ
101	DR ← AC	ACTDR (AC Copy to DR)
110	DR ← DR + 1	INCDR (Increment DR)
111	DR(0-10) ← PC	PCTDR (PC Copy to DR)

F_3	Microoperation	Symbol
000	None	NOPE (No operation)
001	AC ← AC ⊕ DR	XOR
010	AC ← $\overline{AC}$	COM
011	AC ← shl AC	SHL (Shift left AC)
100	AC ← shr AC	SHR (Shift Right AC)
101	PC ← PC + 1	INCPC (Increment in PC)
110	AC ← AR	ARTPC (AR Copy to PC)
111	Reserved	

CD	Condition	Symbol	Comments
00	Always = 1	U	Unconditional Branch
01	DR (15)	I	Indirect address bit
10	AC (15)	S	Sign bit of AC
11	AC = 0	Z	Zero value in AC

BR	Symbol	Function
00	JMP	CAR ← AD If Condition = 1 CAR ← CAR + 1 If Condition = 0
01	CALL	CAR ← AD, SBR ← CAR + 1 If Condition = 1 CAR ← CAR + 1 If Condition = 0
10	RET	CAR ← SBR (Return from Subroutine)
11	MAP	CAR(2-5) ← DR(11-14), CAR(0,1,6) ← 0

Micro-Instruction Format :→

* The 20-bits of the micro-instructions are divided into four functional Parts.
* The micro-operation fields F_1, F_2 & F_3 specify the micro-operations. there are 3 bits in each field to specify 7 distinict micro-operation equations.
  ∴ A total of 21 micro-operations.
  No more than 3 micro-operations can be shown for a micro-instruction i.e one from each field.
* If less than three micro-operations are used, we will use the binary code 000 for no-operations.
* The CD (Condition for branching) field select status bit conditions.
* The BR (Branch field) field specifies the type of branch to be used.
* The AD (Address field) field Contains the branch address. The AD field is of 7-bits Since the Control memory has 2^7 words ($128 = 2^7$ words)

Micro instruction for Fetch Routine :→

AR ← PC
DR ← M[AR], PC ← PC+1
AR ← DR(0–10), CAR(2–5) ← DR(11–14), CAR(0,1,6) ← 0

We can write the Symbolic Microprogram for the above fetch routine as follows:

```
        ORG  64
FETCH:  PCTAR          U     JMP   NEXT
        READ, INCPC    U     JMP   NEXT
        DRTAR          U     MAP
```

The translation of the Symbolic microprogram results in the following binary Code microprogram.

Fetch:

Binary Address	F_1	F_2	F_3	CD	BR	AD
1000000	110	000	000	00	00	1000001
1000001	000	100	101	00	00	1000010
1000010	101	000	000	00	11	0000000

* Indirect Subroutine Micro instructions

Indirect : READ U JMP NEXT
ADTAR U RET

Binary Address	F_1	F_2	F_3	CD	BR	AD
1000001	000	100	000	00	00	1000100
1000100	101	000	000	00	10	0000000

* Symbolic Microprogram :-

LABEL	Micro-operation	CD	BR	AD
ADD	ORG 0			
	NOP	I	CALL	INDIRECT
	READ	U	JMP	NEXT
	ADD	U	JMP	FETCH

Binary Address	F_1	F_2	F_3	CD	BR	AD
0000000	000	000	000	01	01	1000011
0000001	000	100	000	00	00	0000010
0000010	001	000	000	00	00	1000000
0000011	000	000	000	00	00	1000000

```
BRANCH:  ORG 4
         NOP      S    JMP    OVER
         NOP      U    JMP    FETCH
OVER:    NOP      I    CALL   INDRCT
         ARTPC    U    JMP    FETCH
```

Decimal	Binary	F_1	F_2	F_3	CD	BR	AD
4	0000010	000	000	000	10	00	0000110
5	0000101	000	000	000	00	00	1000000
6	0000110	000	000	000	01	01	1000011
7	0000111	000	000	110	00	00	1000000

```
STORE:   ORG 8
         NOP      I    CALL   INDRCT
         ACTOR    U    JMP    NEXT
         INRITE   U    JMP    FETCH
```

Decimal	Binary	F_1	F_2	F_3	CD	BR	AD
8	0001000	000	000	000	01	01	1000011
9	0001001	000	101	000	00	00	0001010
10	0001010	111	000	000	00	00	1000000
11	0001011	000	000	000	00	00	1000000

```
EXCHANGE: ORG 12
          NOP              I    CALL   INDRCT
          READ             U    JMP    NEXT
          ACTOR, ARTAC     U    JMP    NEXT
          WRITE            U    JMP    FETCH
```

Decimal	Binary	F_1	F_2	F_3	CD	BR	AD
12	0001100	000	000	000	01	01	1000011
13	0001101	000	100	000	00	00	0001110
14	0001110	100	101	000	00	00	0001111
15	0001111	111	000	000	00	00	1000000