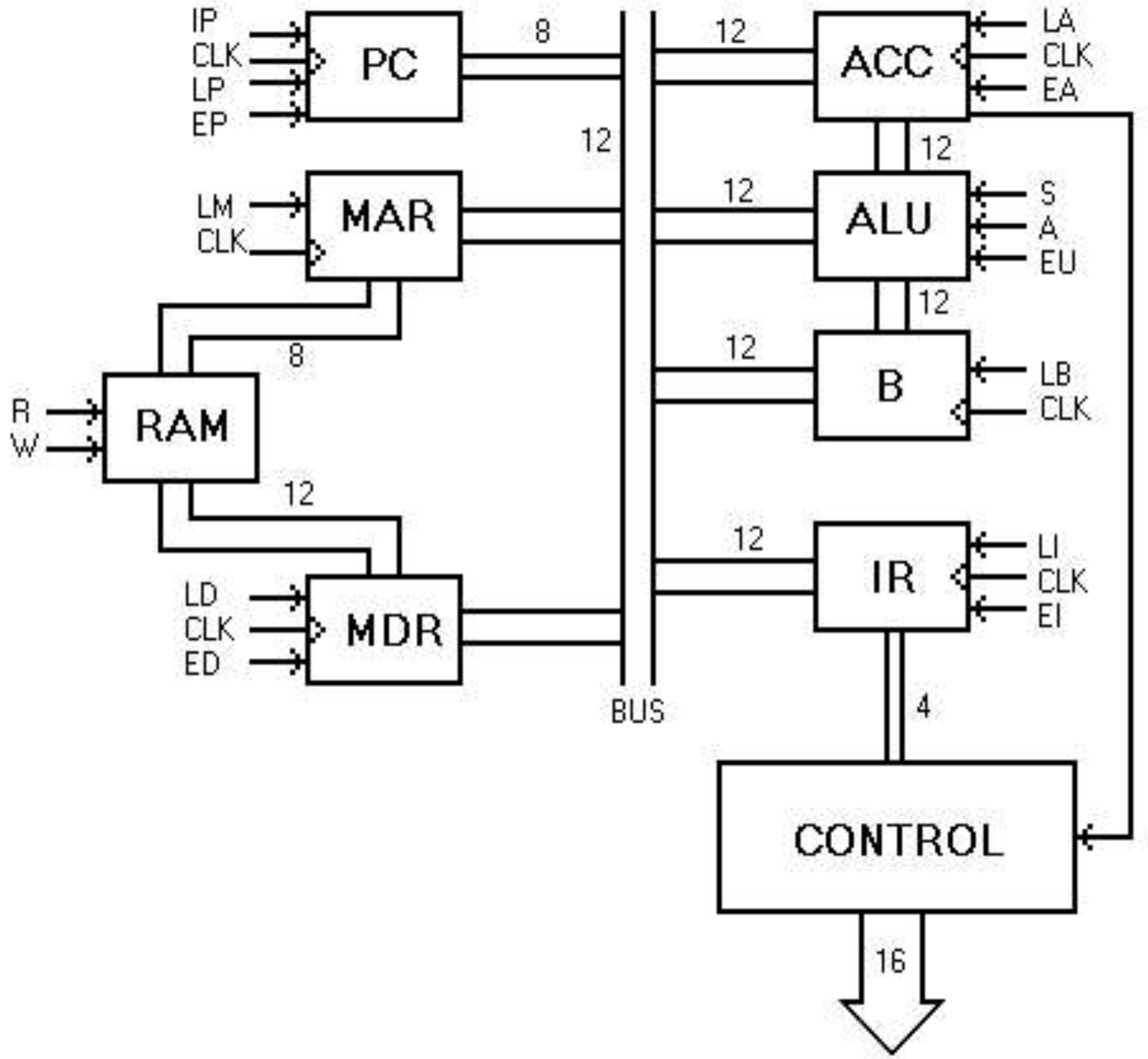
Case Study for Hardwired and Micro programmed Control Units (JC-62)

The Basic Computer

- all traditional digital computers have two principal functional parts
- the **data path section** in which processing occurs
- the **control section** which is responsible for decoding instructions and leaving the correct sequence of control signals to make the processing happen in the data path.

- Basically there are two types of control units:
  - hard-wired controllers
  - micro-programmed controller



IP
CLK
LP
EP
PC
8
LM
CLK
MAR
8
R
W
RAM
12
LD
CLK
ED
MDR
12
BUS
12
ACC
LA
CLK
EA
12
12
ALU
S
A
EU
12
12
B
LB
CLK
12
IR
LI
CLK
EI
4
CONTROL
16

- Data Bus: 12- bits (Data paths)
- Address bus: 8-bits
- Memory Size: 256×12
  - Word size/word length: 12-bit
- Control signals: 16
  - The registers and the 256 X 12 bit RAM memory are controlled by 16 control signals
- Most of the registers have **Load (L)** and **Enabled (E)** signals.
  - An active **L signal** to a register causes the contents of the bus to be clocked into that register on the next rising pulse from the system clock. **(Bus-> register)**
  - An active **E signal** enables the tri-state outputs of the register, thereby making its contents available to the bus. **(register->bus)**
  - **e.g. A->B, EA of A and LB of B**

- Processing of data is done by the Arithmetic-Logic-Unit (ALU), a circuit that is capable of adding or subtracting the 12-bit numbers contained in its two input registers: the accumulator (ACC) and register B.
    - **ACC operation B->ACC**
- The operation performed by the ALU is selected by the **Add (A) or Subtract (S) control signals**.
- The accumulator also contains a single **flip-flop** that is set whenever its contents are **negative.**
- The value of this "**negative flag**" provides input to the controller/sequencer, and permits implementation of **conditional branching instructions.**

The Computer's Instruction Set

- An instruction on computer consists of one **12-bit word.**

Opcode (4-bit)	Operand (memory Address) (8-bit)

- The leading **four bits** form the **operation code** (opcode) which specifies the action to be taken, and the remaining **8 bits**, when used, indicate the **memory address** of **one of the instruction's operands.**
- instructions that have **two operands**, the other operand is always contained **within the accumulator.**

The Computer's Instruction Set

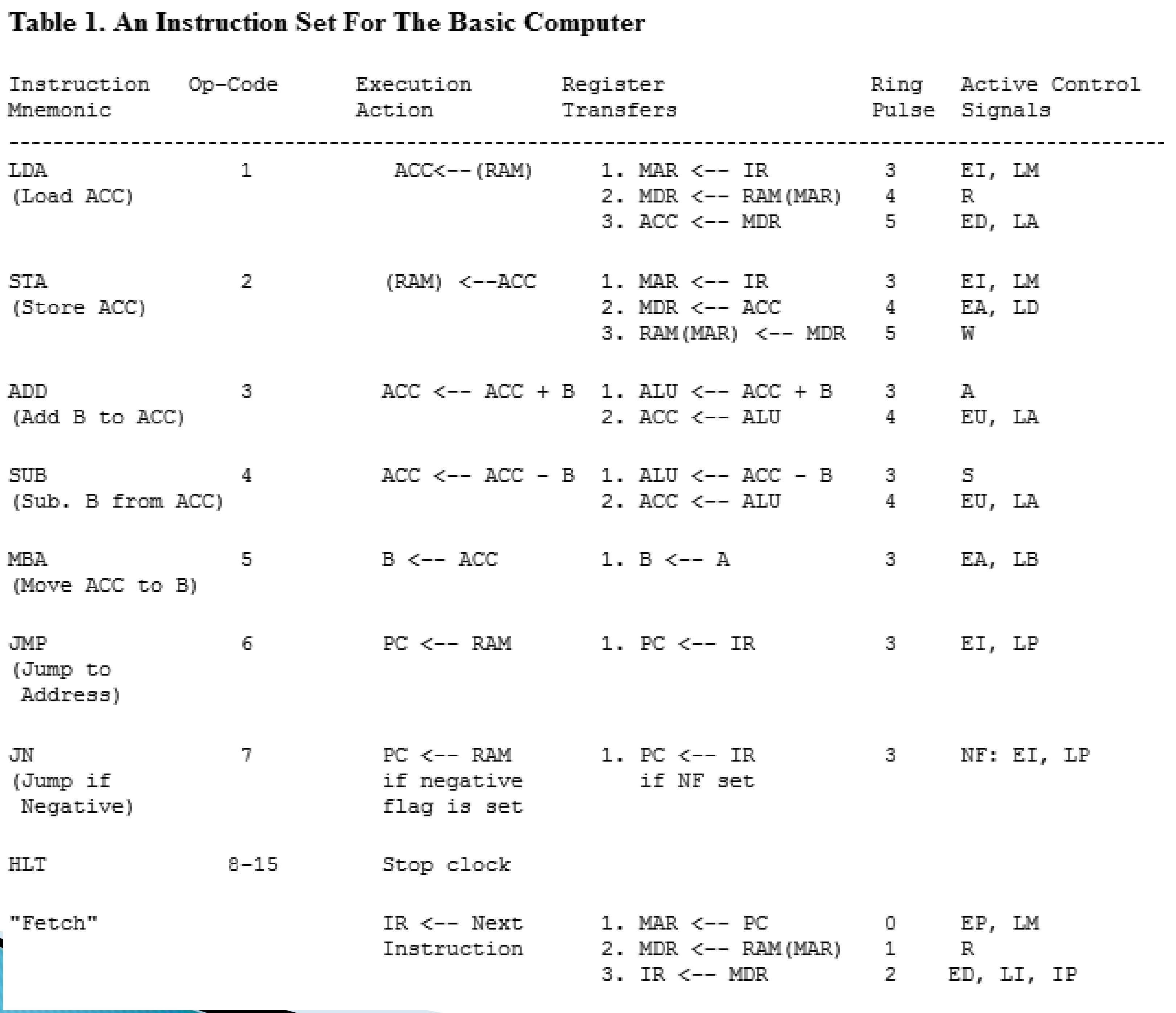


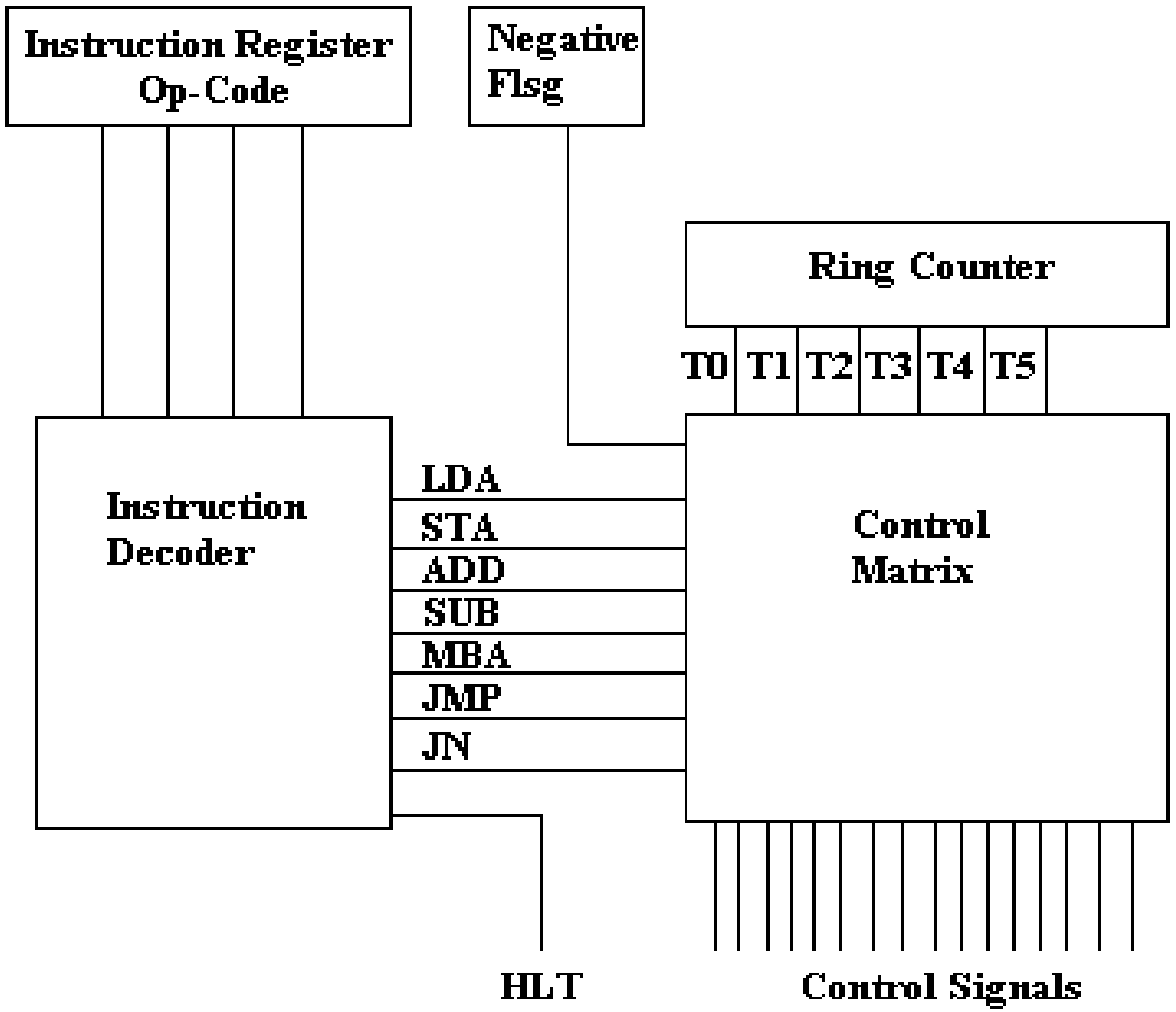
Table 1. An Instruction Set For The Basic Computer

Instruction Mnemonic	Op-Code	Execution Action	Register Transfers	Ring Pulse	Active Control Signals
LDA (Load ACC)	1	ACC<--(RAM)	1. MAR <-- IR	3	EI, LM
			2. MDR <-- RAM(MAR)	4	R
			3. ACC <-- MDR	5	ED, LA
STA (Store ACC)	2	(RAM) <--ACC	1. MAR <-- IR	3	EI, LM
			2. MDR <-- ACC	4	EA, LD
			3. RAM(MAR) <-- MDR	5	W
ADD (Add B to ACC)	3	ACC <-- ACC + B	1. ALU <-- ACC + B	3	A
			2. ACC <-- ALU	4	EU, LA
SUB (Sub. B from ACC)	4	ACC <-- ACC - B	1. ALU <-- ACC - B	3	S
			2. ACC <-- ALU	4	EU, LA
MBA (Move ACC to B)	5	B <-- ACC	1. B <-- A	3	EA, LB
JMP (Jump to Address)	6	PC <-- RAM	1. PC <-- IR	3	EI, LP
JN (Jump if Negative)	7	PC <-- RAM if negative flag is set	1. PC <-- IR if NF set	3	NF: EI, LP
HLT	8-15	Stop clock			
"Fetch"		IR <-- Next Instruction	1. MAR <-- PC	0	EP, LM
			2. MDR <-- RAM(MAR)	1	R
			3. IR <-- MDR	2	ED, LI, IP

The Hard-Wired Control Unit

- Input to the controller consists of the 4-bit opcode of the instruction currently contained in the Instruction Register and the negative flag from the accumulator.
- The controller's output is a set of 16 control signals that go out to the various registers and to the memory of the computer.
- HLT signal that is activated whenever the leading bit of the op-code is one.
- The controller is composed of the following functional units:
- A ring counter, an instruction decoder, and a control matrix.

The Hard-Wired Control Unit



Ring counter

- The ring counter provides a sequence of six consecutive active signals that cycle continuously.

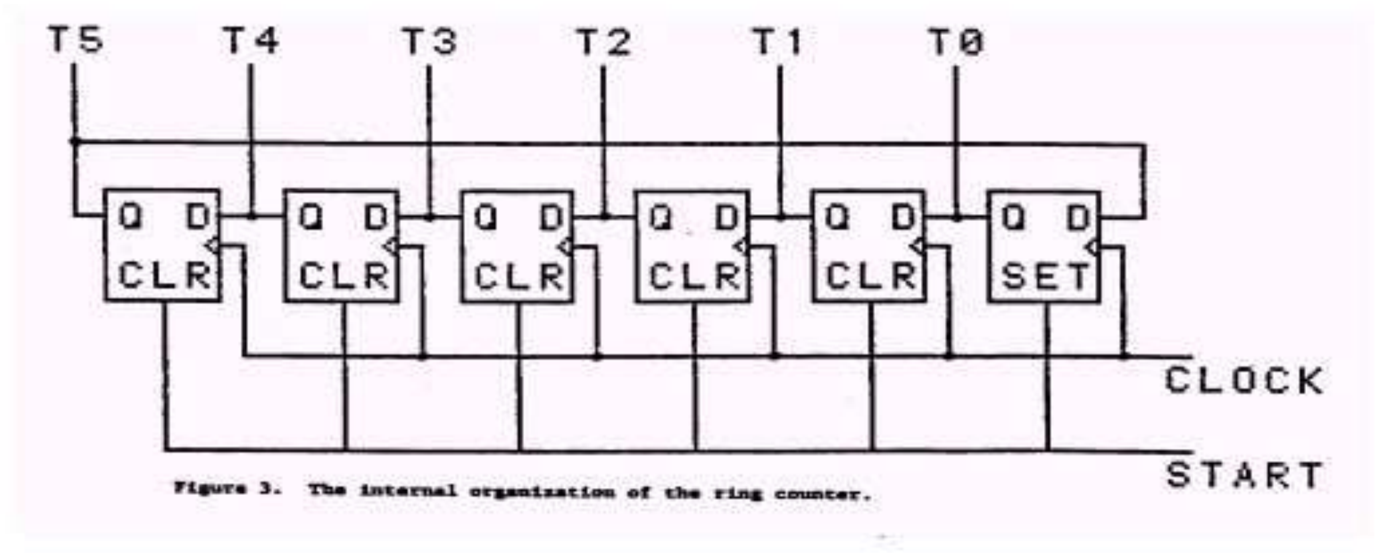


Figure 3. The internal organization of the ring counter.

Decoder

- Bubble means'0'
- Without bubble means '1'

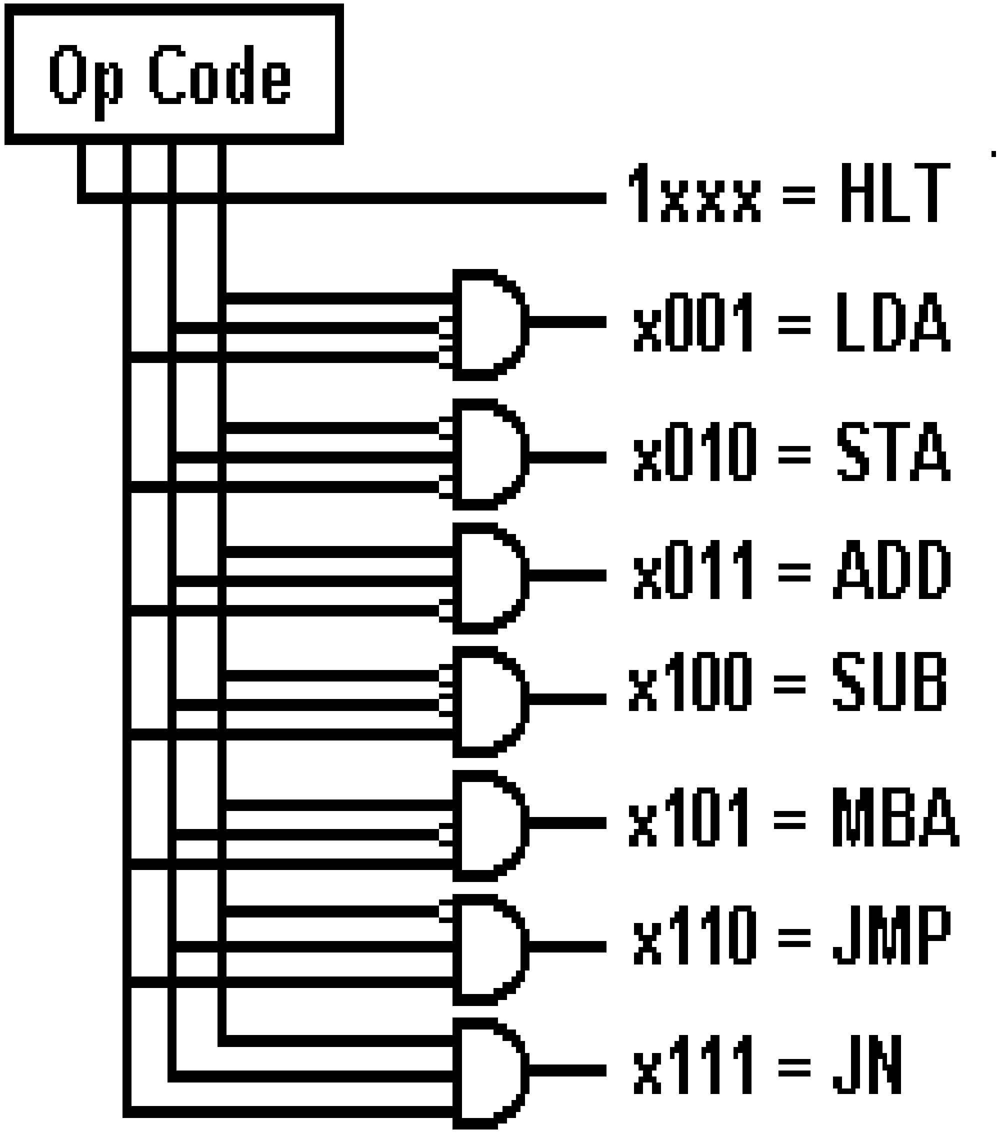


Table 2. A Matrix of Times at which Each Control Signal Must Be Active in Order to Execute the Hard-wired Basic Computer's Instructions

Control Signal: Instruction:	IP	LP	EP	LM	R	W	LD	ED	LI	EI	LA	EA	A	S	EU	LB
"Fetch"	T2		T0	T0	T1			T2	T2							
LDA				T3	T4			T5		T3	T5					
STA				T3		T5	T4			T3		T4				
MBA												T3				T3
ADD											T4		T3		T4	
SUB											T4			T3	T4	
JMP		T3								T3						
JN		T3*NF								T3*NF						

Figure 6. The logical equations required for each of the hardwired control signals on the basic computer. The machine's control matrix is designed from these equations.

IP = T2

W = T5*STA

LP = T3*JMP + T3*NF*JN

LD = T4*STA

LA = T5*LDA + T4*ADD + T4*SUB

EA = T4*STA + T3*MBA

EP = T0

S = T3*SUB

A = T3*ADD

LI = T2

LM = T0 + T3*LDA + T3*STA

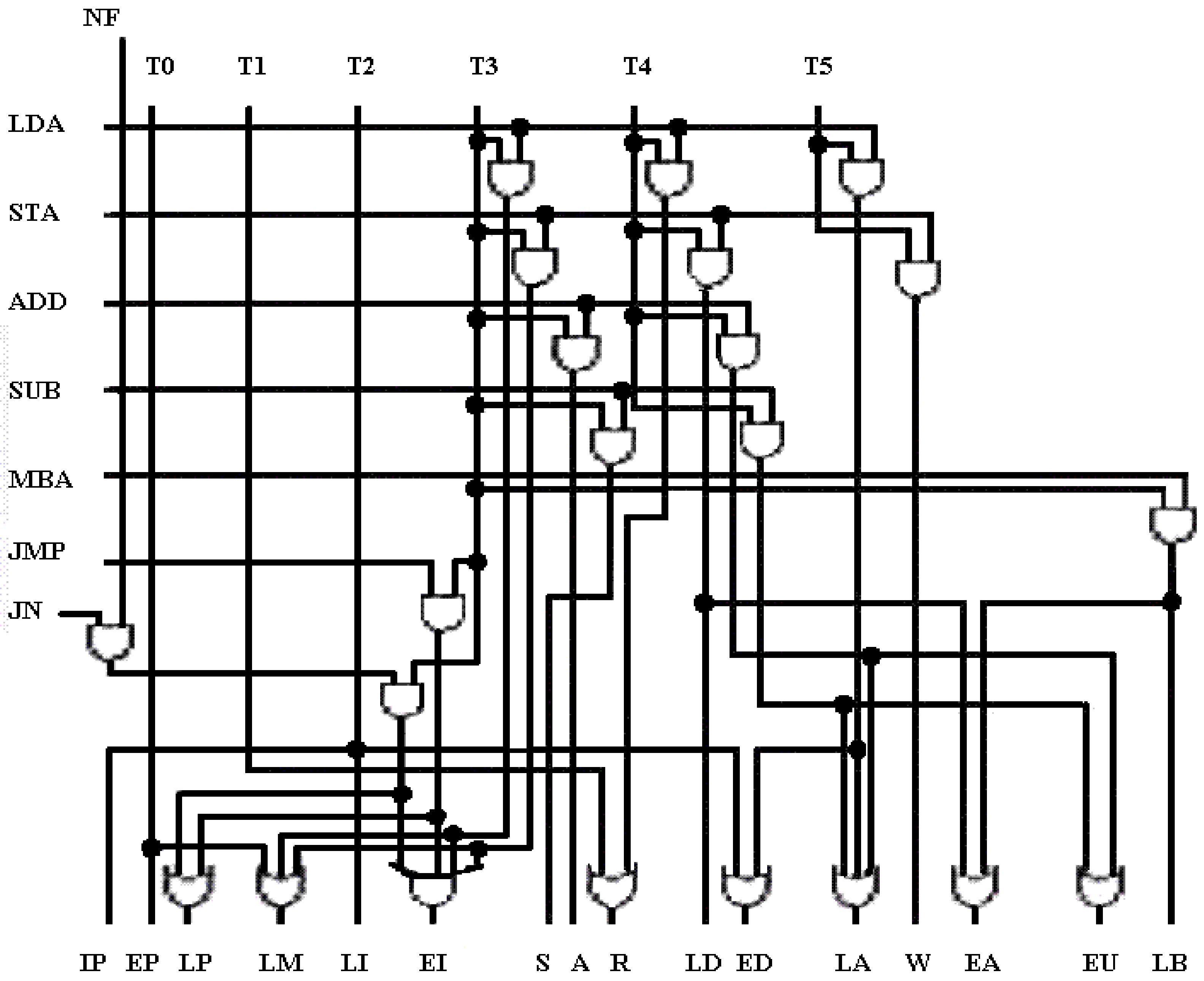
ED = T2 + T5*LDA

R = T1 + T4*LDA

EU = T3*ADD+T3*SUB

EI = T3*LDA + T3*STA + T3*JMP + T3*NF*JN

LB = T3*MBA


NF
T0
T1
T2
T3
T4
T5
LDA
STA
ADD
SUB
MBA
JMP
JN
IP
EP
LP
LM
LI
EI
S
A
R
LD
ED
LA
W
EA
EU
LB

A Micro-programmed Control Unit

- the controller causes instructions to be executed by issuing a specific set of control signals at each beat of the system clock.
- **Each set of control signals** issued causes **one basic operation (micro-operation)**
- **sets of control signals** that cause specific **micro-operations** to occur as being "**microinstructions**" that could be **stored in a memory.**
- **Each bit** of a **microinstruction** might correspond to **one control signal.**
- If the bit is **set** it means that the control signal will be **active**; if **cleared** the signal will be **inactive**.
- **Sequences** of microinstructions could be stored in an **internal "control" memory.**
- A **sequence of microinstructions** that implements **an instruction on the external computer** is known as a **micro-routine.**
- The **instruction set of the computer** is thus determined by **the set of micro-routines**, the **"microprogram,"** stored in the **controller's memory.**
- **The control unit of a microprogram-controlled computer is essentially a computer within a computer.**

- IP, LP, EP, LM, R, W, LD, ED, LI, EI, LA, EA, A, S, EU, LB
- Fetch operation (Control bits)
  - 3000h
  - 0800h
  - 8180h
- LDA
  - 1040
  - 0800
  - 0120

Micro- Instruction(Control Word)

Control Word (16-bit)	CD (1-bit)	MAP (1-bit)	HLT (1-bit)	CRJA (5-bit)

24-bit

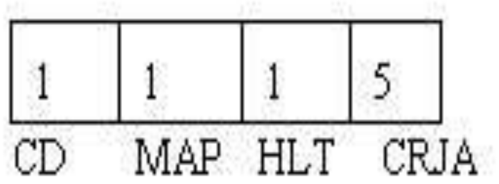
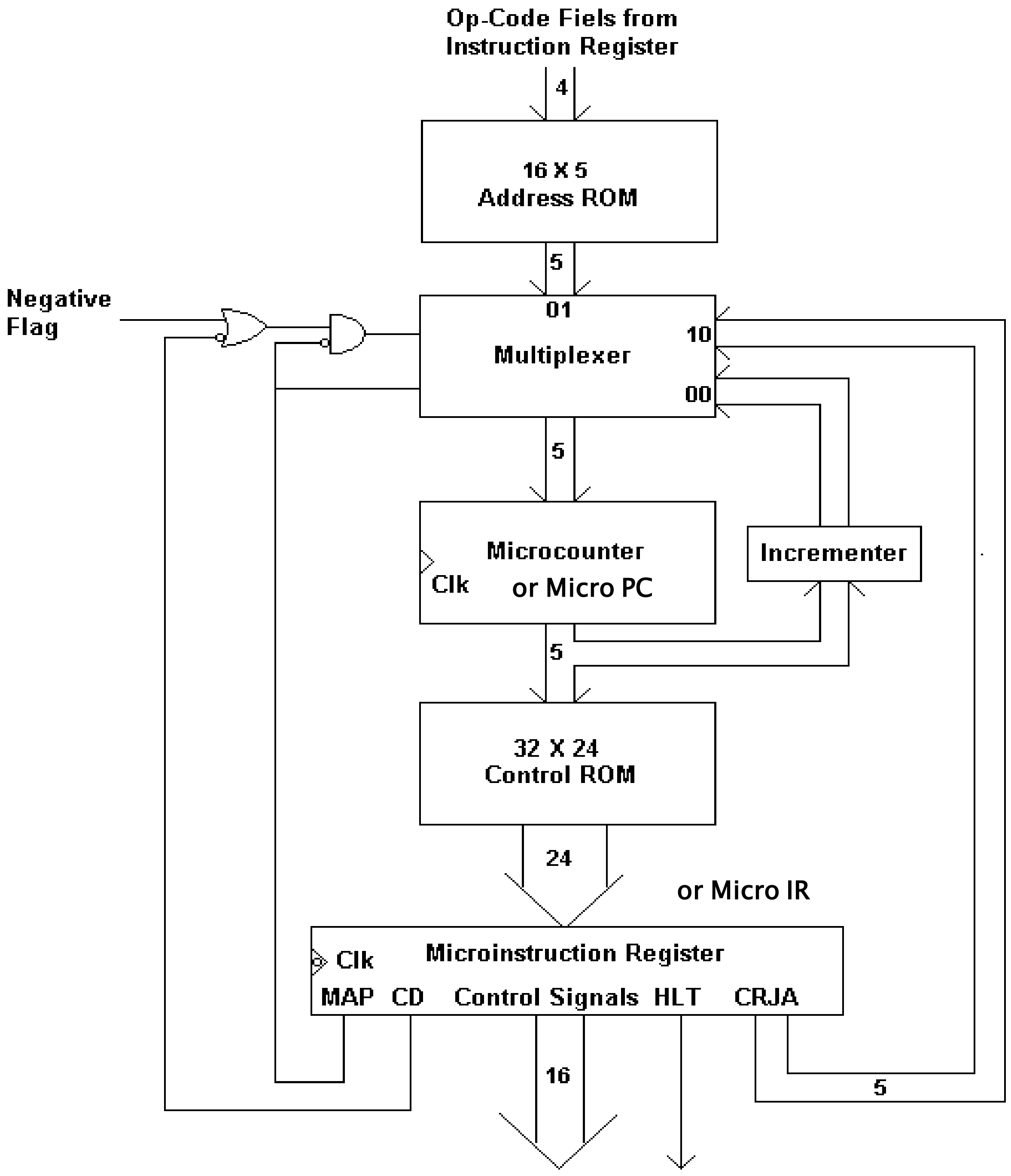


Figure 8. Next address field of the microinstruction register. CD is the condition bit, MAP causes the address of the next microinstruction to be obtained from the address ROM, HLT stops the clock, and CRJA is the control ROM jump address field.

A Microprogrammed Control Unit for the Simple Computer



Control ROM

- Addresses provided to the **control ROM** come from a **micro-counter register**, which is analogous to the external machine's program counter.
- The micro-counter, in turn, receives its input from a multiplexer which selects from :
  - (1) the output of an address ROM,
  - (2) a current-address incrementer, or
  - (3) the address stored in the next-address field of the current microinstruction.

Address ROM

- **Address zero** of the **address ROM** contains the control-ROM address of the **fetch routine**; each other addresses in the address-ROM corresponds to **one of the op-codes** of the computer's instruction set.

Address ROM

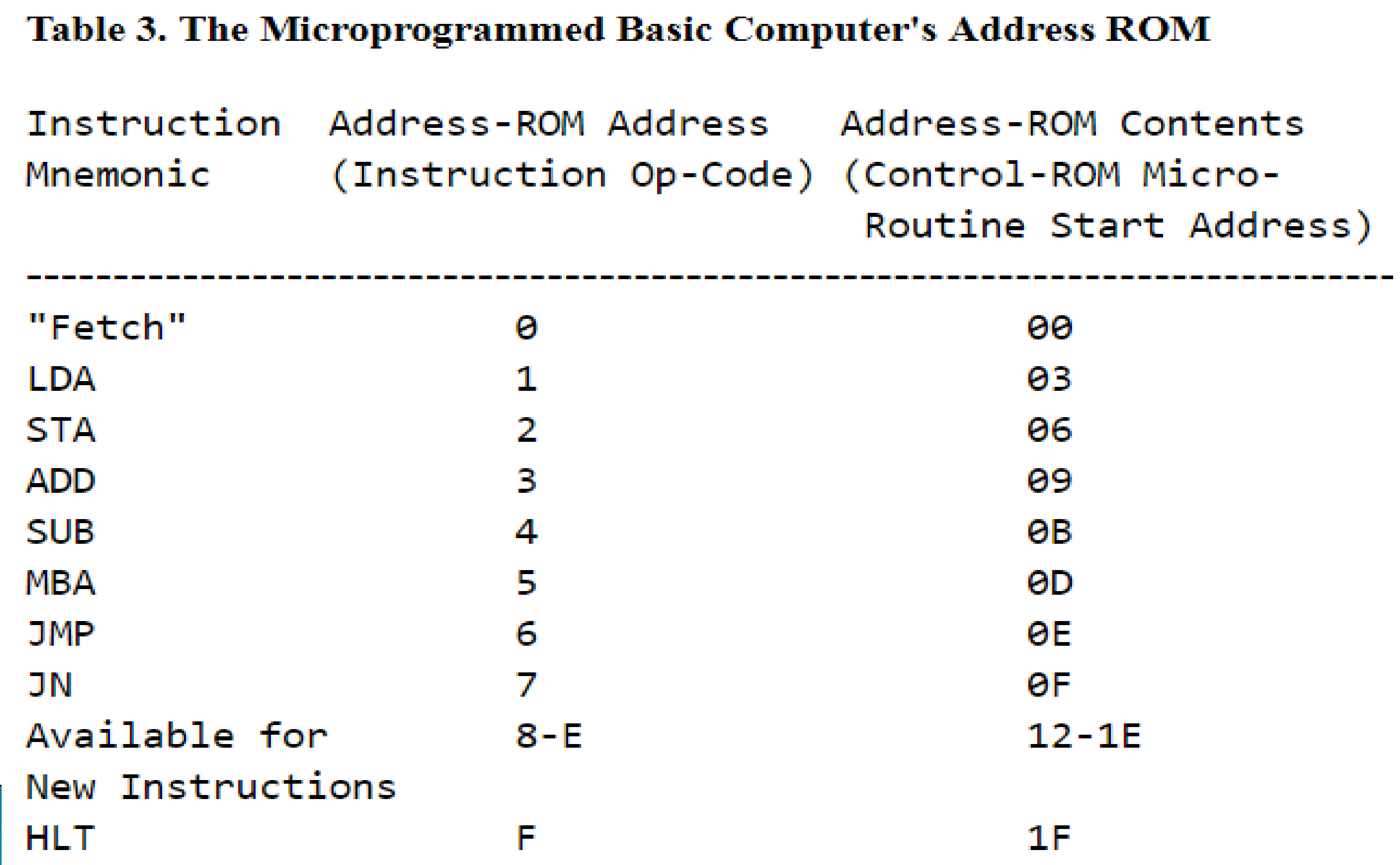


Table 3. The Microprogrammed Basic Computer's Address ROM

Instruction Mnemonic	Address-ROM Address (Instruction Op-Code)	Address-ROM Contents (Control-ROM Micro-Routine Start Address)
"Fetch"	0	00
LDA	1	03
STA	2	06
ADD	3	09
SUB	4	0B
MBA	5	0D
JMP	6	0E
JN	7	0F
Available for New Instructions	8-E	12-1E
HLT	F	1F

- **MAP=1:**Bring address from IR (Select code 01)
- **MAP=0 & CD=0:**Unconditional Branch (Select logic 10) Take CRJA as next address
- **MAP=0 & CD=1 :**Conditional Branch
  - Select logic =00 (if NF=0) load inc. address , no branching
  - Select logic =10 (if NF=1) use CRJA for next micro-instruction, Branching

MAP	CD	Select Logic		
0	0	1	0	Unconditional Branch, Take CRJA as next address
0	1	0	0	If NF=0, Load Incrementer address, No branching
		1	0	If NF=1, use CRJA for next Micro-instruction, Branching
1	X	0	1	Bring Address from IR/ Address ROM

Table 4. The Microprogram that Implements the Basic Computer's Instruction Set

Microroutine	Address-ROM	Micro-	Control Signals:	CD	MAP	HLT	Address of	Comment
Name	Address	Instruction	ILELRWLELELEASEL	bit	bit	bit	Next Micro-	
(Mnemonic)	(Op-code)	Address	PPPM DDIIAA UB				Instruction	
"Fetch"	0	00	0011000000000000	0	0	0	01	Next CR Address = 01
		01	0000100000000000	0	0	0	02	Next CR Address = 02
		02	1000000110000000	0	1	0	XX	Get CR Address from Address ROM
LDA	1	03	0010000001000000	0	0	0	04	Nexr CR Address = 04
		04	0000100000000000	0	0	0	05	Next CR Address = 05
		05	0000000100100000	0	0	0	00	Next CR Address = 00 (Fetch)
STA	2	06	0010000001000000	0	0	0	07	Next CR Address = 07
		07	0000001000010000	0	0	0	08	Next CR Address = 08
		08	0000010000000000	0	0	0	00	Next CR Address = 00 (Fetch)
ADD	3	09	0000000000001000	0	0	0	0A	Next CR Address = 0A
		0A	0000000000100010	0	0	0	00	Next CR Address = 00 (Fetch)
SUB	4	0B	0000000000000100	0	0	0	0C	Next CR Address = 0C
		0C	0000000000100010	0	0	0	00	Next CR Address = 00 (Fetch)
MBA	5	0D	0000000000010001	0	0	0	00	Next CR Address is 00 (Fetch)
JMP	6	0E	0100000001000000	0	0	0	00	Change PC; next CR Address is 00 (Fetch)
JN	7	0F	0000000000000000	1	0	0	11	NF=0: INC CRJA; NF=1: Next CR Address = 11
		10	0000000000000000	0	0	0	00	Next CR Address = 00 (Fetch)
		11	0100000001000000	0	0	0	00	Change PC; next CR Address is 00 (Fetch)
Available for	8-E	12-1E						New microinstructions can be added here
HLT	F	1F	0000000000000000	0	0	1	XX	Stop Clock

Example:

```
LABEL_Z :  ADD
           SUB
           JN LABEL_Z
           MBA
```

Hardwired vs. Micro-programmed Computers

- micro-programmed
  - Flexibility
  - Simplified

- Hardwired
  - Fast