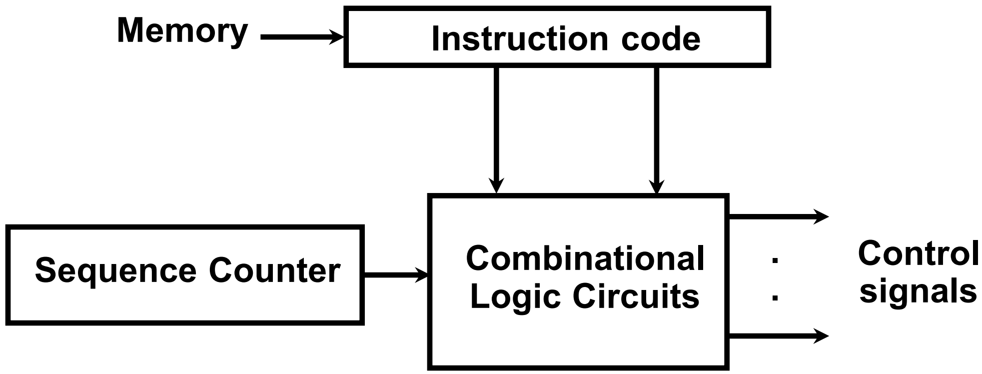
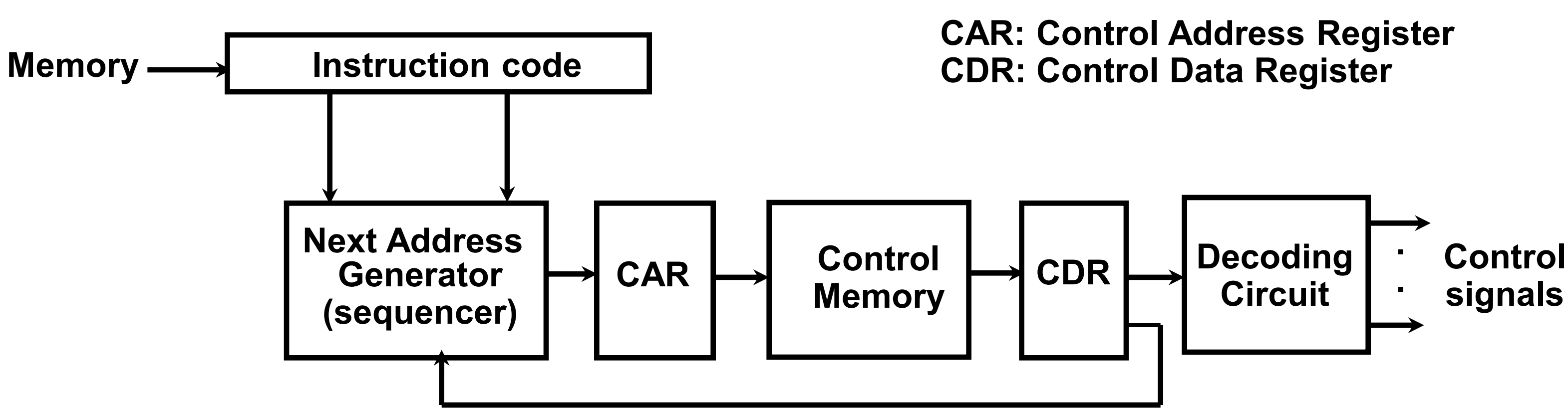
Micro-programmed Control

Control Unit Implementation

- Hardwired



- Microprogrammed



Microprogrammed Control Unit

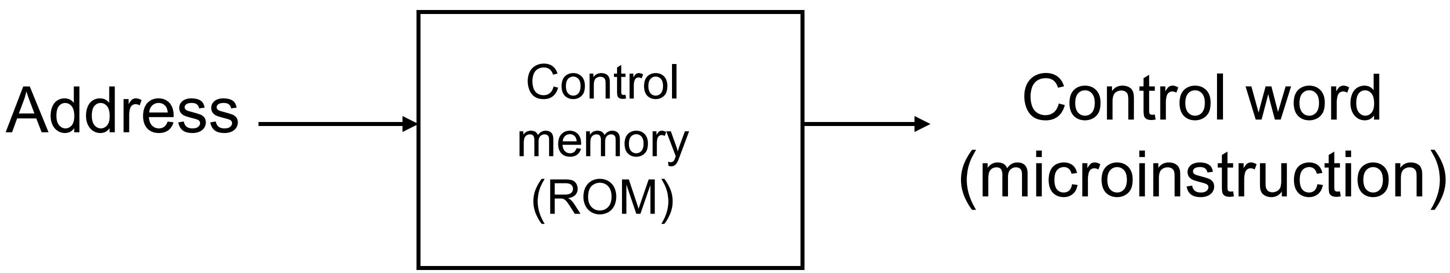
- Control signals
  - Group of bits used to select paths in multiplexers, decoders, arithmetic logic units
- Control variables
  - Binary variables specify microoperations
    - Certain microoperations initiated while others idle
- Control word
  - String of 1's and 0's represent control variables

Micro-programmed Control Unit

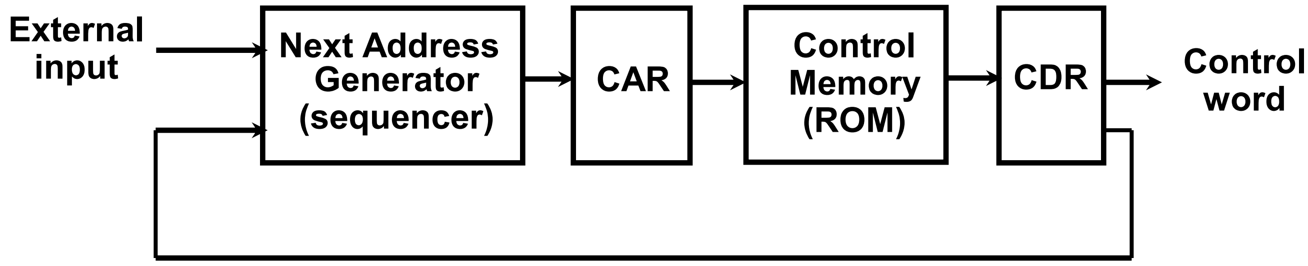
- Control memory
  - Memory contains control words
- Microinstructions
  - Control words stored in control memory
  - Specify control signals for execution of microoperations
- Microprogram
  - Sequence of microinstructions

Control Memory

- Read-only memory (ROM)
- Content of word in ROM at given address specifies microinstruction
- Each computer instruction initiates series of microinstructions (microprogram) in control memory
- These microinstructions generate microoperations to
  - Fetch instruction from main memory
  - Evaluate effective address
  - Execute operation specified by instruction
  - Return control to fetch phase for next instruction



Microprogrammed Control Organization



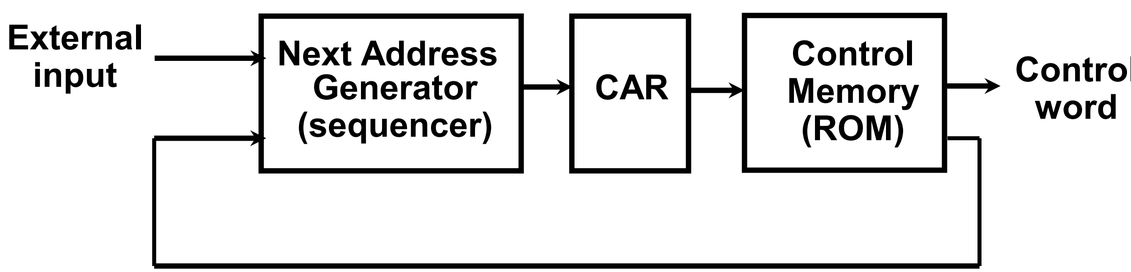
- Control memory
  - Contains microprograms (set of microinstructions)
  - Microinstruction contains
    - Bits initiate microoperations
    - Bits determine address of next microinstruction
- Control address register (CAR)
  - Specifies address of next microinstruction

Microprogrammed Control Organization

- Next address generator (microprogram sequencer)
  - Determines address sequence for control memory
- Microprogram sequencer functions
  - Increment CAR by one
  - Transfer external address into CAR
  - Load initial address into CAR to start control operations

Microprogrammed Control Organization

- Control data register (CDR)- or pipeline register
  - Holds microinstruction read from control memory
  - Allows execution of microoperations specified by control word simultaneously with generation of next microinstruction
- Control unit can operate without CDR



Microprogram Routines

- Routine
  – Group of microinstructions stored in control memory
- Each computer instruction has its own microprogram routine to generate microoperations that execute the instruction

Microprogram Routines

- Subroutine
  - Sequence of microinstructions used by other routines to accomplish particular task
- Example
  - Subroutine to generate effective address of operand for memory reference instruction
- Subroutine register (SBR)
  - Stores return address during subroutine call

Conditional Branching

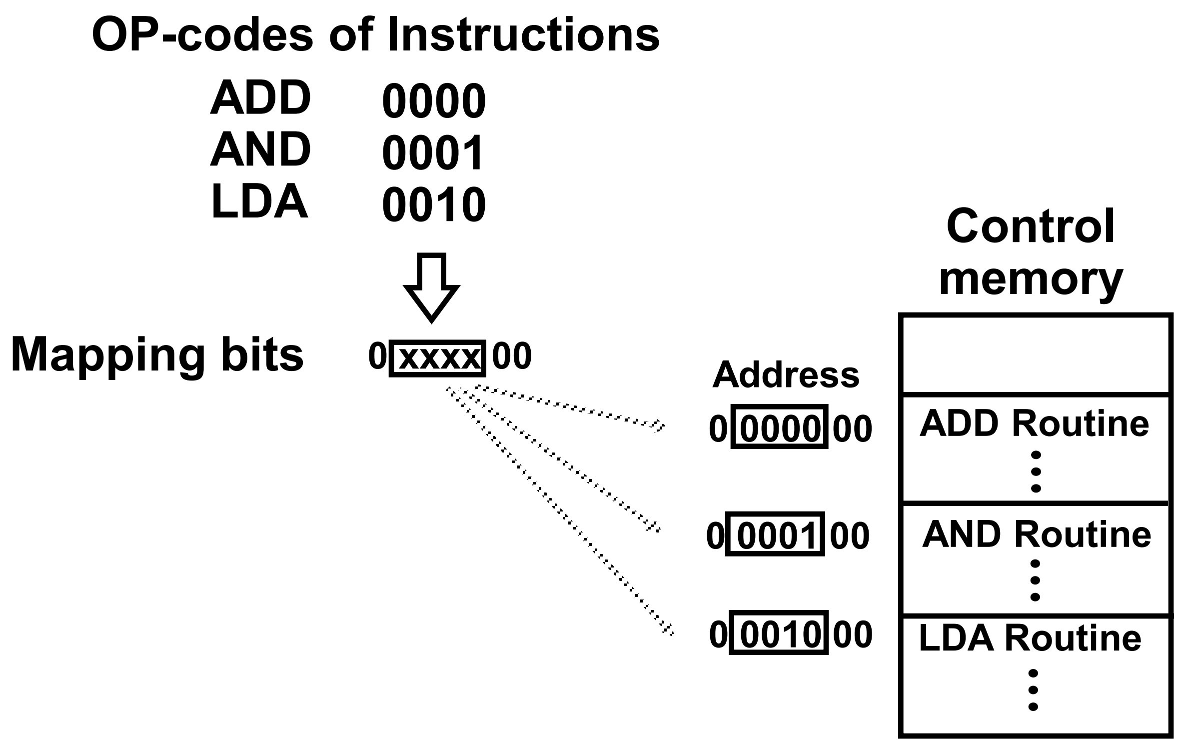
- Branching from one routine to another depends on status bit conditions
- Status bits provide parameter info such as
  - Carry-out of adder
  - Sign bit of number
  - Mode bits of instruction
- Info in status bits can be tested and actions initiated based on their conditions: 1 or 0
- Unconditional branch
  - Fix value of status bit to 1

Mapping of Instruction

- Each computer instruction has its own microprogram routine stored in a given location of the control memory
- Mapping
  – Transformation from instruction code bits to address in control memory where routine is located

Mapping of Instruction

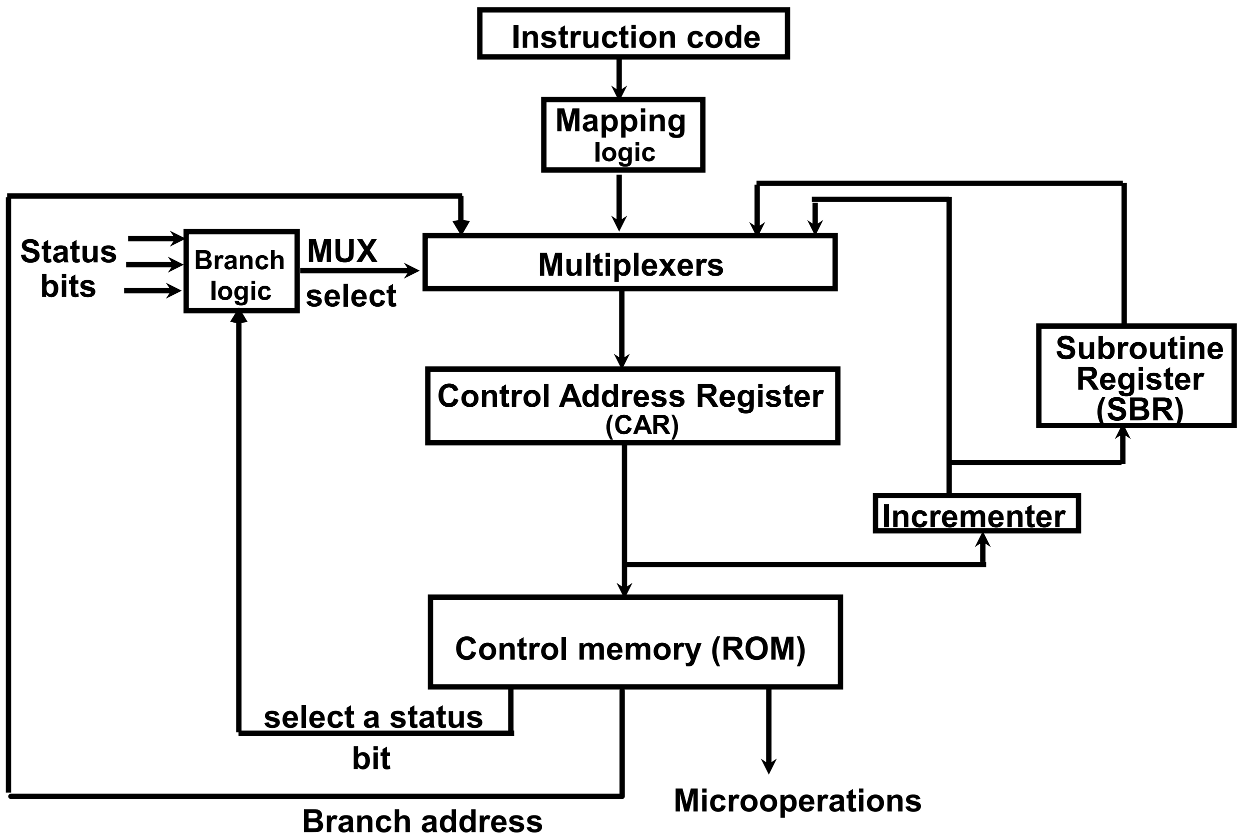
- Example
  - Mapping 4-bit operation code to 7-bit address



Address Sequencing

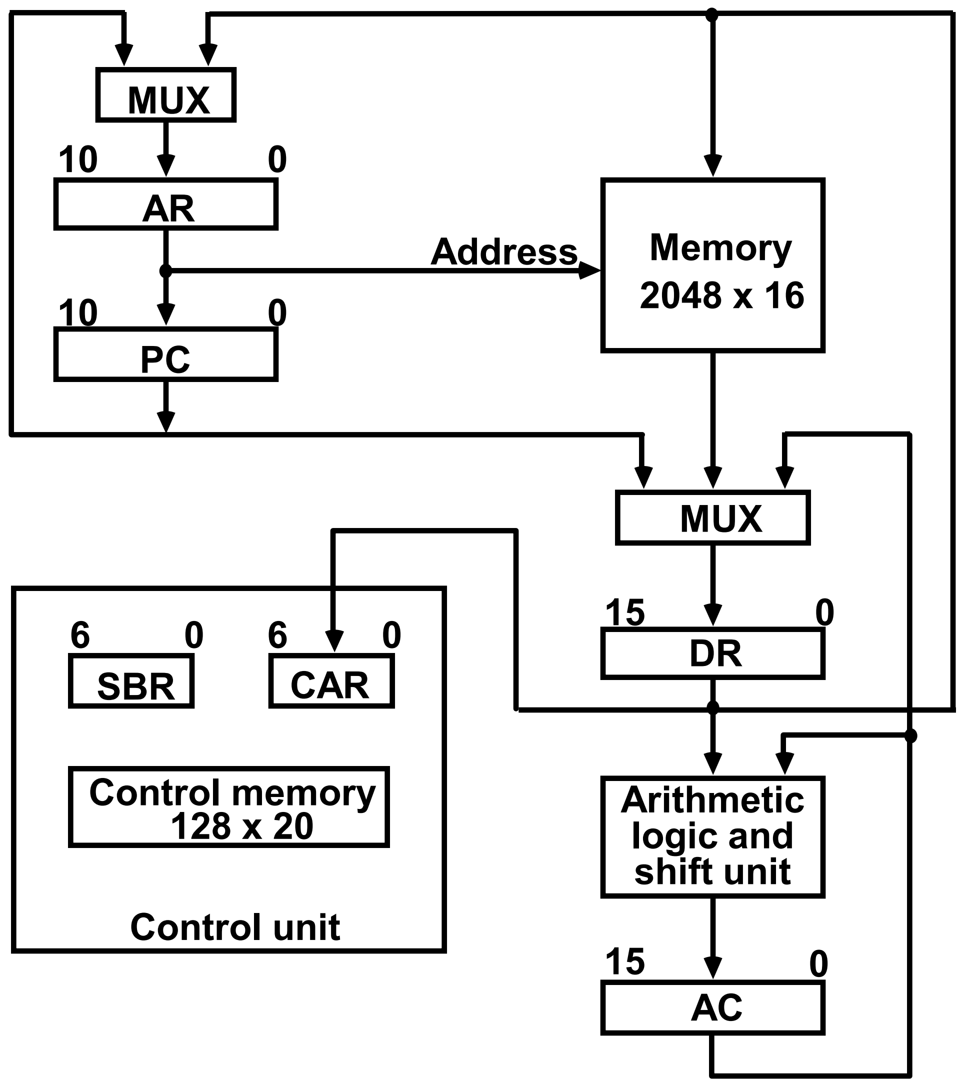
- Address sequencing capabilities required in control unit
  - Incrementing CAR
  - Unconditional or conditional branch, depending on status bit conditions
  - Mapping from bits of instruction to address for control memory
  - Facility for subroutine call and return

Address Sequencing



Micro-program Example

Computer Configuration



Microprogram Example

Computer instruction format

15	14 11	10 0
I	Opcode	Address

Four computer instructions

Symbol	OP-code	Description
ADD	0000	$AC \leftarrow AC + M[EA]$
BRANCH	0001	if $(AC < 0)$ then $(PC \leftarrow EA)$
STORE	0010	$M[EA] \leftarrow AC$
EXCHANGE	0011	$AC \leftarrow M[EA], M[EA] \leftarrow AC$

EA is the effective address

Microinstruction Format

3	3	3	2	2	7
F1	F2	F3	CD	BR	AD

F1, F2, F3: Microoperation fields
CD: Condition for branching
BR: Branch field
AD: Address field

Micro-instruction Fields

F1	Microoperation	Symbol
000	None	NOP
001	$AC \leftarrow AC + DR$	ADD
010	$AC \leftarrow 0$	CLRAC
011	$AC \leftarrow AC + 1$	INCAC
100	$AC \leftarrow DR$	DRTAC
101	$AR \leftarrow DR(0\text{-}10)$	DRTAR
110	$AR \leftarrow PC$	PCTAR
111	$M[AR] \leftarrow DR$	WRITE

F2	Microoperation	Symbol
000	None	NOP
001	$AC \leftarrow AC - DR$	SUB
010	$AC \leftarrow AC \vee DR$	OR
011	$AC \leftarrow AC \wedge DR$	AND
100	$DR \leftarrow M[AR]$	READ
101	$DR \leftarrow AC$	ACTDR
110	$DR \leftarrow DR + 1$	INCDR
111	$DR(0\text{-}10) \leftarrow PC$	PCTDR

F3	Microoperation	Symbol
000	None	NOP
001	$AC \leftarrow AC \oplus DR$	XOR
010	$AC \leftarrow AC'$	COM
011	$AC \leftarrow \text{shl } AC$	SHL
100	$AC \leftarrow \text{shr } AC$	SHR
101	$PC \leftarrow PC + 1$	INCPC
110	$PC \leftarrow AR$	ARTPC
111	Reserved	

Microinstruction Fields

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

Symbolic Microinstruction

- **Sample Format**

Label:	Micro-ops	CD	BR	AD

- Label: may be empty or may specify symbolic address terminated with colon
- Micro-ops: consists of 1, 2, or 3 symbols separated by commas
- CD: one of {U, I, S, Z}
  U: Unconditional Branch
  I: Indirect address bit
  S: Sign of AC
  Z: Zero value in AC
- BR: one of {JMP, CALL, RET, MAP}
- AD: one of {Symbolic address, NEXT, empty}

Fetch Routine

- Fetch routine
  - Read instruction from memory
  - Decode instruction and update PC

Microinstructions for fetch routine:

$AR \leftarrow PC$
$DR \leftarrow M[AR], PC \leftarrow PC + 1$
$AR \leftarrow DR(0\text{-}10), CAR(2\text{-}5) \leftarrow DR(11\text{-}14), CAR(0,1,6) \leftarrow 0$

Symbolic microprogram for fetch routine:

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

Binary microporgram for fetch routine:

Binary address	F1	F2	F3	CD	BR	AD
1000000	110	000	000	00	00	1000001
1000001	000	100	101	00	00	1000010
1000010	101	000	000	00	11	0000000

Symbolic Microprogram

- **Control memory:** 128 20-bit words
- **First 64 words:** Routines for 16 machine instructions
- **Last 64 words:** Used for other purpose (e.g., fetch routine and other subroutines)
- **Mapping:** OP-code XXXX into 0XXXX00, first address for 16 routines are 0(0 0000 00), 4(0 0001 00), 8, 12, 16, 20, ..., 60

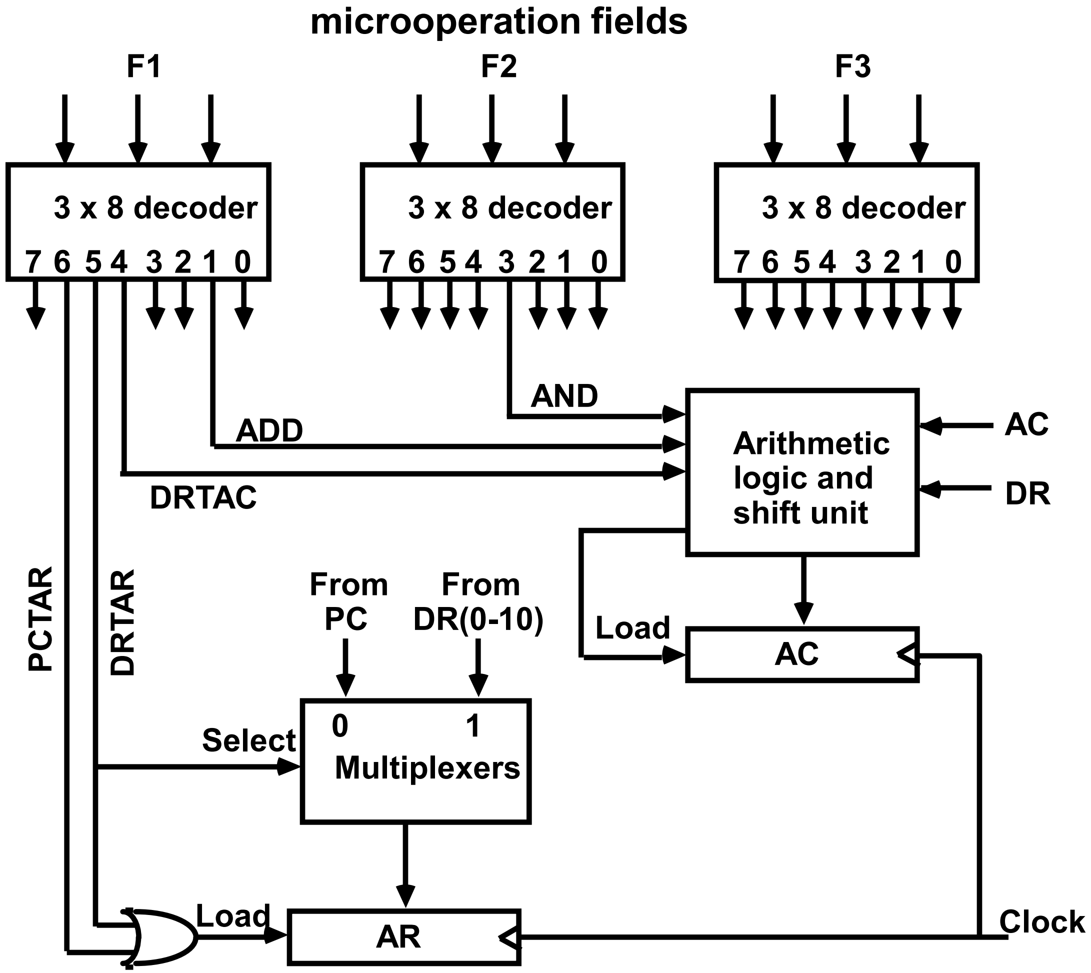
Partial Symbolic Microprogram

Label	Microops	CD	BR	AD
	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
	ORG 8			
STORE:	NOP	I	CALL	INDRCT
	ACTDR	U	JMP	NEXT
	WRITE	U	JMP	FETCH
	ORG 12			
EXCHANGE:	NOP	I	CALL	INDRCT
	READ	U	JMP	NEXT
	ACTDR, DRTAC	U	JMP	NEXT
	WRITE	U	JMP	FETCH
	ORG 64			
FETCH:	PCTAR	U	JMP	NEXT
	READ, INCPC	U	JMP	NEXT
	DRTAR	U	MAP	
INDRCT:	READ	U	JMP	NEXT
	DRTAR	U	RET	

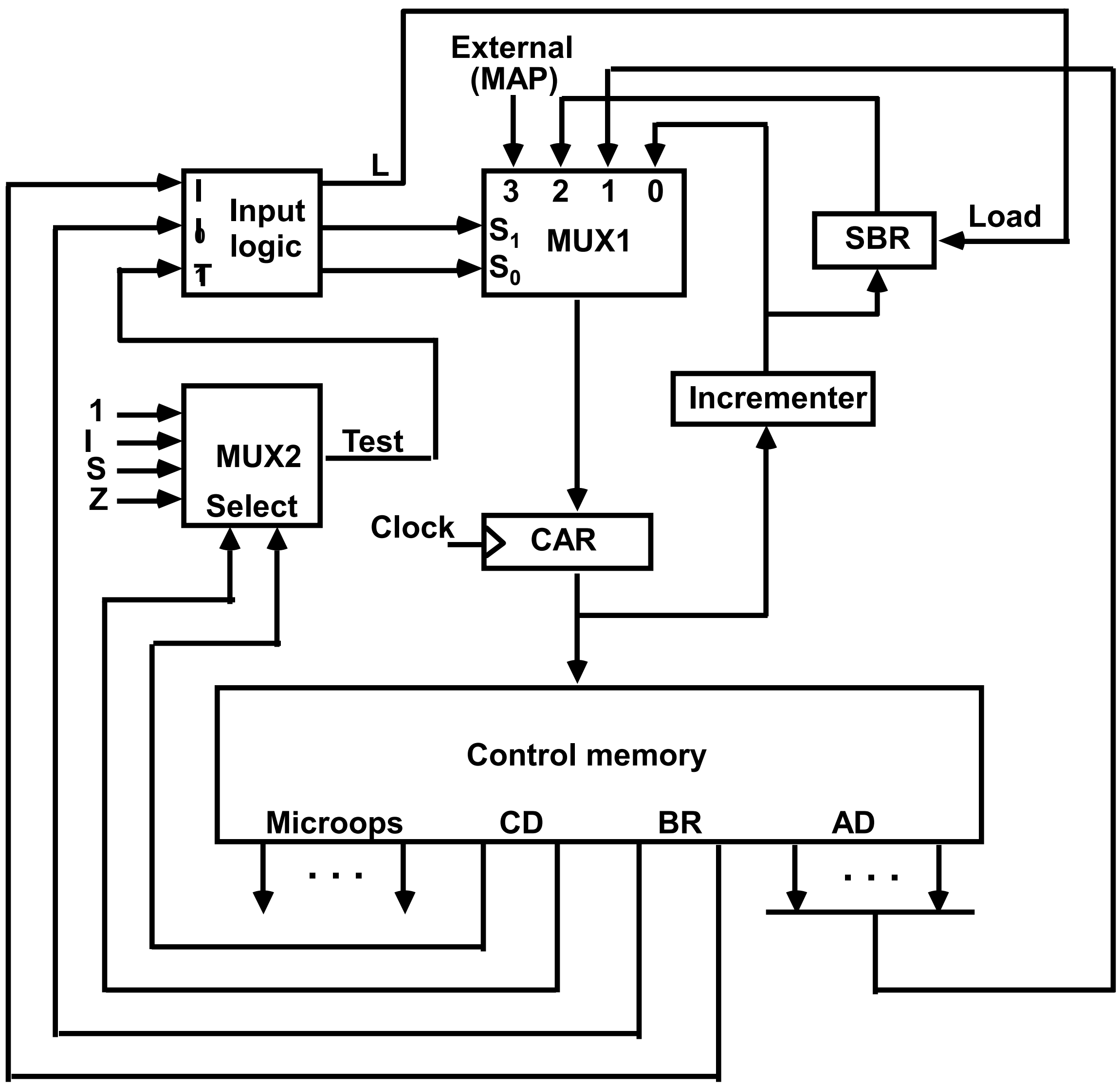
Binary Microprogram

Micro Routine	Address Decimal	Address Binary	Binary Microinstruction F1	F2	F3	CD	BR	AD
ADD	0	0000000	000	000	000	01	01	1000011
	1	0000001	000	100	000	00	00	0000010
	2	0000010	001	000	000	00	00	1000000
	3	0000011	000	000	000	00	00	1000000
BRANCH	4	0000100	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	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	12	0001100	000	000	000	01	01	1000011
	13	0001101	001	000	000	00	00	0001110
	14	0001110	100	101	000	00	00	0001111
	15	0001111	111	000	000	00	00	1000000
FETCH	64	1000000	110	000	000	00	00	1000001
	65	1000001	000	100	101	00	00	1000010
	66	1000010	101	000	000	00	11	0000000
INDRCT	67	1000011	000	100	000	00	00	1000100
	68	1000100	101	000	000	00	10	0000000

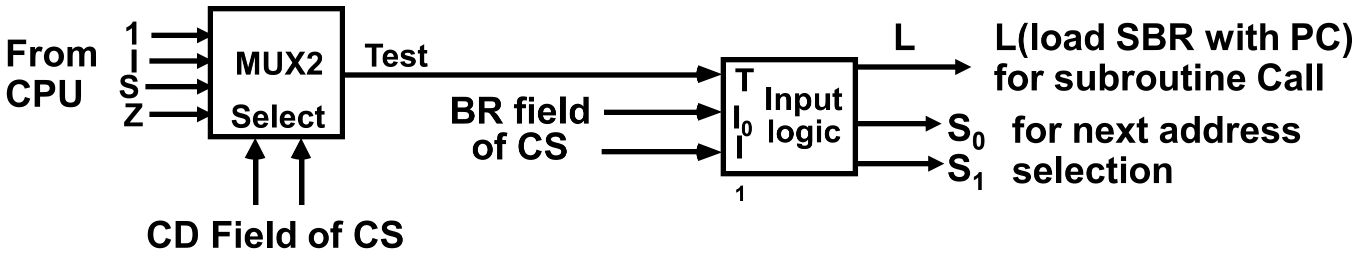
Design of Control Unit



Microprogram Sequencer



Input Logic for Microprogram Sequencer



Input Logic

I1I0T	Meaning	Source of Address	S_1S_0	L
000	In-Line	CAR+1	00	0
001	JMP	CS(AD)	01	0
010	In-Line	CAR+1	00	0
011	CALL	CS(AD) and SBR <- CAR+1	01	1
10x	RET	SBR	10	0
11x	MAP	DR(11-14)	11	0

$S1 = I1$

$S_0 = I_0I_1 + I1'T$

$L = I_1'I0T$