UNIT 4. Microprogrammed Control

Control Memory

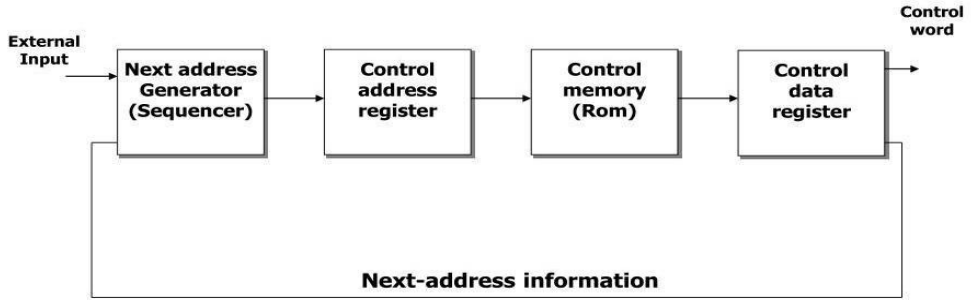
The control function that specifies a microoperation is called as control variable. When control variable is in one binary state, the corresponding microoperation is executed. For the other binary state the state of registers does not change. The active state of a control variable may be either 1 state or the 0 state, depending on the application. Example; For bus-organized systems the control signals that specify microoperations are groups of bits that select the paths in multiplexers, decoders, and arithmetic logic units.

Control Word

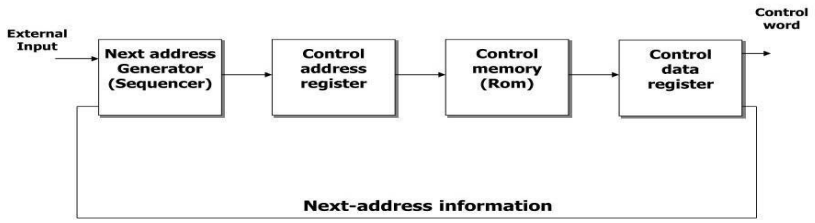
The control variables at any given time can be represented by a string of 1's and 0's called a control word. All control words can be programmed to perform various operations on the components of the system.

Microprogram control unit

A control unit whose binary control variables are stored in memory is called a microprogram control unit. The control word in control memory contains within it a microinstruction. The microinstruction specifies one or more micro-operations for the system. A sequence of microinstructions constitutes a microprogram. The control unit consists of control memory used to store the microprogram. Control memory is a permanent i.e., read only memory (ROM). The general configuration of a micro-programmed control unit organization is shown as block diagram below.



The control memory is ROM so all control information is permanently stored. The control memory address register (CAR) specifies the address of the microinstruction and the control data register (CDR) holds the microinstruction read from memory. The next address generator is sometimes called a microprogram sequencer. It is used to generate the next micro instruction address. The location of the next microinstruction may be the one next in sequence or it may be located somewhere else in the control memory. So it is necessary to use some bits of the present microinstruction to control the generation of the address of the microinstruction. Sometimes the next address may also be a function of external input conditions. The control data register holds the present microinstruction while next address is computed and read from memory. The data register is sometimes called a pipeline register.



A computer with a microprogrammed control unit will have two separate memories:

1. a main memory (RAM)
2. control memory (ROM)

The microprogram consists of microinstructions that specify various internal control signals for execution of register microoperations. These microinstructions generate the microoperations to:

1) fetch the instruction from main memory
2) evaluate the effective address
3) execute the operation return control to the fetch phase for the next instruction

Addressing sequence

Microinstructions are stored in control memory in groups, with each group specifying a routine. Each computer instruction has its own microprogram routine to generate the microoperations. The hardware that controls the address sequencing of the control memory must be capable of sequencing the microinstructions within a routine and be able to branch from one routine to another Steps the control must undergo during the execution of a single computer instruction are as follows. Initial address is loaded into the control address register(CAR) when power is turned on in the computer. This address is usually the address of the first microinstruction that activates the instruction fetch routine. At the end of the fetch routine instruction is placed in the instruction register- IR. The control memory then goes through the routine to determine the effective address of the operand with the help of mode bits and branch micro instructions. At the end of this routine Address register AR holds operand address. The next step is to generate the microoperations that execute the instruction fetched from memory by considering the opcode and applying a mapping process. The transformation of the instruction code bits to an address in control memory where the routine of instruction located is referred to as mapping process. After execution, control must return to the fetch routine by executing an unconditional branch.

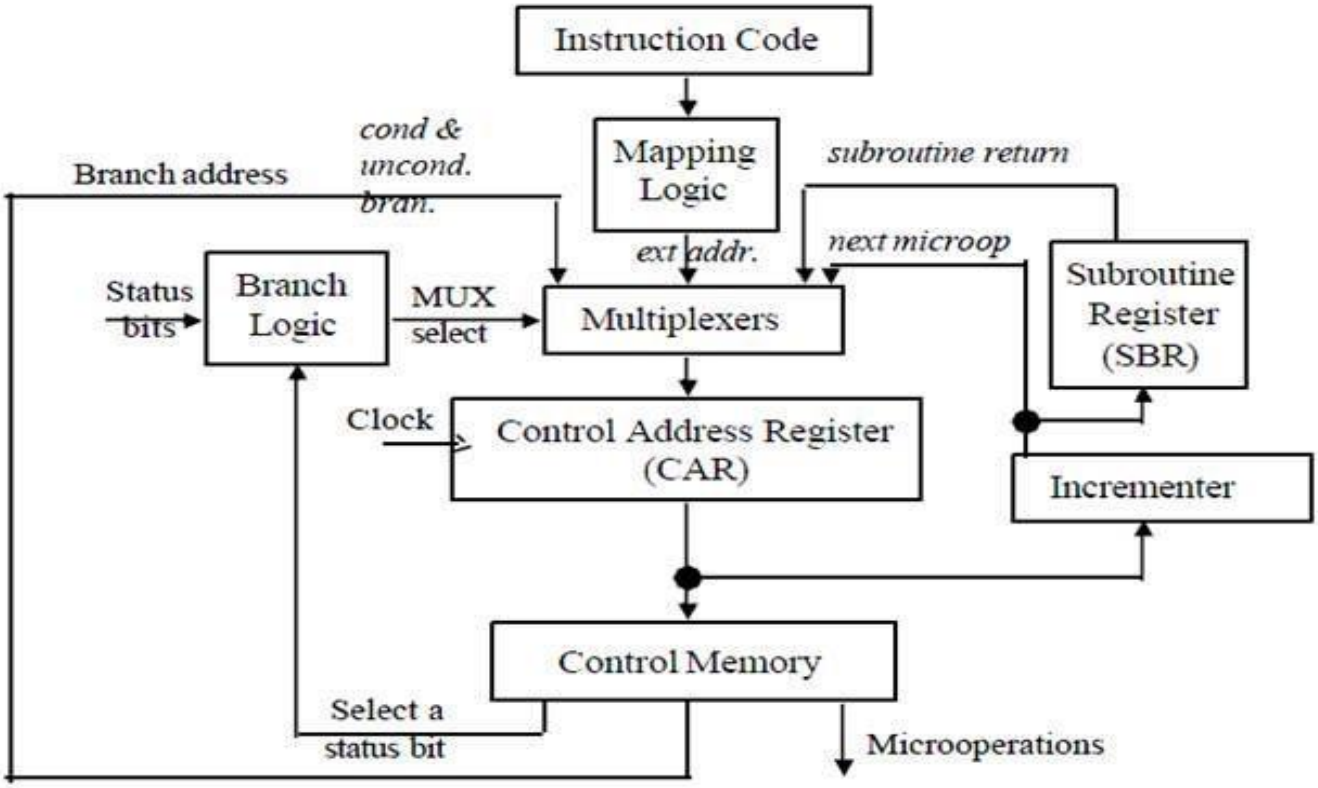
In brief the address sequencing capabilities required in a control memory are:

1. Incrementing of the control address register.
2. Unconditional branch or conditional branch, depending on status bit conditions.
3. A mapping process from the bits of the instruction to an address for control memory.
4. A facility for subroutine call and return.

- In brief the address sequencing capabilities required in a control memory are:
    1. Incrementing of the control address register.
    2. Unconditional branch or conditional branch, depending on status bit conditions.
    3. A mapping process from the bits of the instruction to an address for control memory.
    4. A facility for subroutine call and return.

Selection of address for control memory

The microinstruction in control memory contains a set of bits to initiate microoperations in computer registers and other bits to specify the method by which the next address is obtained.



In the figure four different paths form which the control address register (CAR) receives the address.

✓The incrementer increments the content of the control register address register by one, to select the next microinstruction in sequence.

✓Branching is achieved by specifying the branch address in one of the fields of the microinstruction.

✓Conditional branching is obtained by using part of the microinstruction to select a specific status bit in order to determine its condition.

✓An external address is transferred into control memory via a mapping logic circuit.

✓The return address for a subroutine is stored in a special register, that value is used when the micoprogram wishes to return from the subroutine.

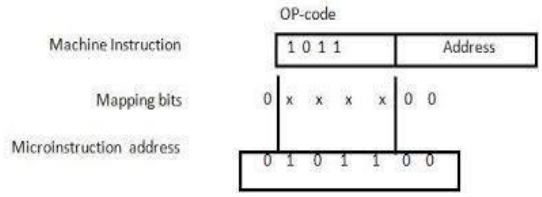
Conditional branching

- Conditional branching is obtained by using part of the microinstruction to select a specific status bit in order to determine its condition.
- The status conditions are special bits in the system that provide parameter information such as the carry-out of an adder, the sign bit of a number, the mode bits of an instruction, and i/o status conditions.
- The status bits, together with the field in the microinstruction that specifies a branch address, control the branch logic.
- The branch logic tests the condition, if met then branches, otherwise, increments the CAR.
- Conditional branching can be implemented with a multiplexer. If there are 8 status bit conditions, then 3 bits in the microinstruction are used to specify any one of the conditions and they provide the selection variables for the multiplexer.
- If the selected status bit is in 1 state, the output of the multiplexer is 1, otherwise it is 0.
- A 1 output in the multiplexer generates a control signal to transfer the branch address from the microinstruction into the control address register.

- A 0 output in the multiplexer causes the address register to be incremented.
- For unconditional branching, fix the value of one status bit to be 1.
- Reference to this bit causes the branch address to be loaded into the control address register unconditionally.

Mapping of instructions

- A special type of branch exists when a microinstruction specifies a branch to the first word in control memory where a microprogram routine is located.
- The status bits for this type of branch are the bits in the opcode.
- Assume an opcode of four bits and a control memory of 128 locations. The mapping process converts the 4bit opcode to a 7-bit address for control memory shown in below figure.
- Mapping consists of placing a 0 in the most significant bit of the address, transferring the four operation code bits, and clearing the two least significant bits of the control address register.
- This provides for each computer instruction a microprogram routine with a capacity of four microinstructions.
- The mapping function is implemented by integrated circuit called programmable logic device



Microprogram example

- The process of code generation for the control memory is called microprogramming.
- The block diagram of the computer configuration is shown in the figure.
- Two memory units:

1. Main memory – stores instructions and data
2. Control memory – stores microprogram

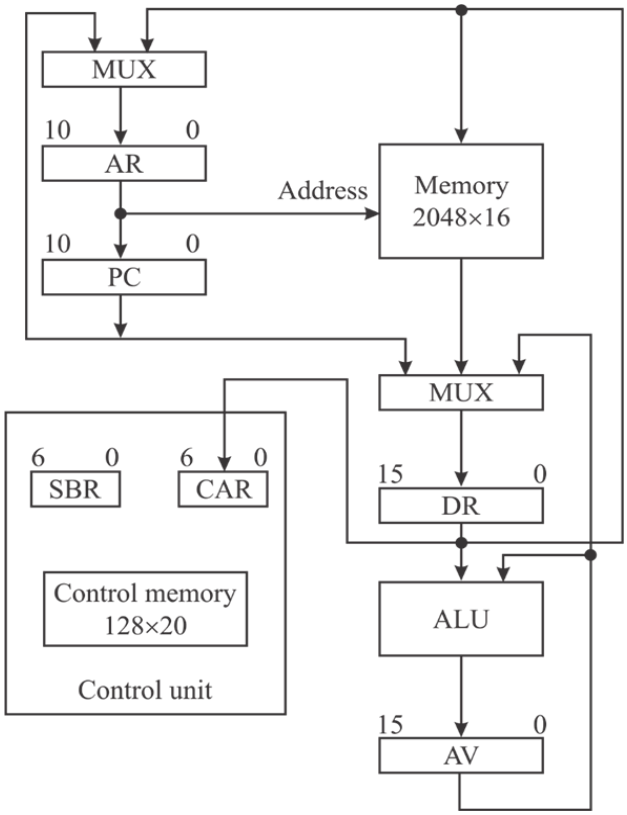
- Four processor registers

1. Program counter – PC
2. Address register – AR
3. Data register – DR
4. Accumulator register - AC

- Two control unit registers

1. Control address register – CAR
2. Subroutine register – SBR

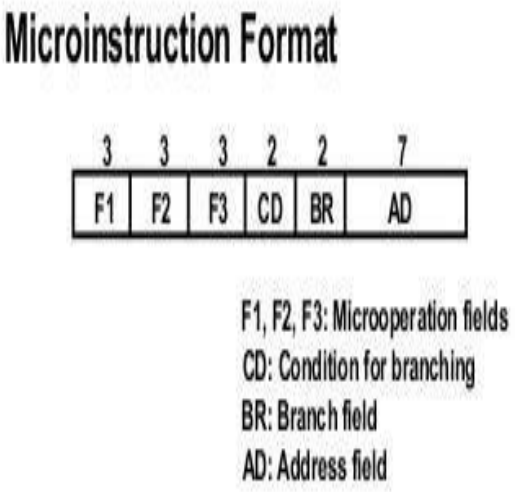
- Transfer of information among registers in the processor is through MUXs rather than a bus.
- DR receives information from AC,PC or memory.
- AR can receive information from PC or DR



Micro instruction format

The microinstruction format is composed of 20 bits divided into four parts

- Three fields F1, F2, and F3 specify microoperations for the computer [3 bits each]
- The CD field selects status bit conditions [2 bits]
- The BR field specifies the type of branch to be used [2 bits]
- The AD field contains a branch address [7 bits] because control memory has 128 words
- Each of the three microoperation fields can specify one of seven possibilities.
- No more than three microoperations can be chosen for a microinstruction.
- If fewer than three are needed, the code 000 = NOP.
- The three bits in each field are encoded to specify seven distinct microoperations listed in below table.
- The condition field (CD) is two bits to specify four status bit conditions .
- The branch field (BR) consists of two bits and is used with the address field to choose the address of the next microinstruction.



Symbols and Binary Code for Microinstruction Fields

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

F2	Microoperation	Symbol
000	None	NOP
001	AC ← AC - DR	SUB
010	AC ← AC ∨ DR	OR
011	AC ← AC ∧ DR	AND
100	DR ← M[AR]	READ
101	DR ← AC	ACTDR
110	DR ← DR + 1	INCDR
111	DR(0-10) ← PC	PCTDR

F3	Microoperation	Symbol
000	None	NOP
001	AC ← AC ⊕ DR	XOR
010	AC ← AC'	COM
011	AC ← shl AC	SHL
100	AC ← shr AC	SHR
101	PC ← PC + 1	INCPC
110	PC ← AR	ARTPC
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

Symbolic microinstructions

- Different symbols can be used to construct the micro instructions in symbolic form.
- Each line of an assembly language microprogram defines a symbolic microinstruction and is divided into five parts

➢ Lable
➢ Microoperations
➢ CD
➢ BR
➢ AD

Label	Microops	CD	BR	AD
	ORG 0			
ADD:	NOP	I	CALL	INDRCT
	READ	U	JMP	NEXT
	ADD	U	JMP	FETCH

1. The label field may be empty or it may specify a symbolic address. Terminate with a colon (:).
2. The microoperations field consists of 1-3 symbols, separated by commas. Only one symbol from each field. If NOP, then translated to 9 zeros
3. The condition field specifies one of the four conditions U,I,S,Z.
4. The branch field has one of the four branch symbols JMP,CALL,RET,MAP
5. The address field has three formats

- a. A symbolic address – must also be a label
- b. The symbol NEXT to designate the next address in sequence
- c. Empty if the branch field is RET or MAP and is converted to 7 zeros
- The symbol ORG defines the origin i;e the first address of a microprogram routine.
- Eg; ORG 64 – places first microinstruction at control memory 1000000 which is equivalent to decimal number 64.

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	

22