UNIT 11 THE CONTROL UNIT

Structure **Page No.**

11.0 INTRODUCTION

In the previous units of this block, instruction set architecture and concept of micro-operations were discussed. The micro-operations are used to provide execution environment for the instruction set in a computer system. The micro-operations are implemented as a part of the ALU and processor internal BUS.

The control unit is responsible for issuing the control signals, as per the micro-operationsrequirements of the instructions of a computer. In addition, it controls all the other units of a computer system. In this unit we are going to discuss the functions of a control unit and its implementation mechanisms like hardwired control unit and micro-programmed control unit. The micro-programmed control unit is popular amongst the Intel computer architecture due to its flexibility and legacy requirements. The hardwired control unit and other computer logic circuitrycan be designed using Hardware Description Languages (HDL). The input to these languages are the electronic circuit structure and expected behaviour.Some of the specialized HDLprogramming languages areVHDL, Verilog etc. Discussion on these languages is beyond the scope of this course.
The unit discusses the basic requirements of a control unit, followed by the hardwired control unit andWilkes control unit. Finally, we will discuss the micro-programmed control.

11.1 OBJECTIVES

After going through this unit, you will be able to:

(a) define what is a control unit and its function;
(b) describe a simple control unit organization;
(c) define a hardwired control unit;
(d) define the micro-programmed control unit;
(e) define the term micro-instruction; and
(f) identify types and formats of micro-instruction.

11.2 THE CONTROL UNIT

The processor of a computer consists of three basic components – a set of registers, the arithmetic logic unitand the control unit. All these components are connected through an internal BUS. The role of a control unit is to ensure that an instruction gets executed, as per the sequence of micro-operations for that instruction. This process also involves decoding the instruction that is to be executed. The control unit issues control signals so that these instructions are executed correctly. The basic responsibilities of the control unit are to control:

a) the data exchange of the processor with the memory or I/O modules or interfaces.
b) the internal operations in the processor, which may include:

- moving data between registers using internal BUS or moving data from/to memory location using system BUS (register transfer micro-operations)
- making ALU to perform a particular operation on the data
- regulating other internal operations.

But how does a control unit control the above operations? What are the functional requirements of the control unit? What is its structure? Let us explore answers to these questions in the next sections.

Functional Requirements of Control Unit

Let us first try to define the functions, which a control unit must perform tocause instructions execution. But to define the functions of a control unit, one must know what resources and means it has at its disposal. A control unit must know about the:

(a) Basic components of the processor
(b) Micro-operation this processorcan perform

The processor of a computer consists of the following basic functional components:

- **The Arithmetic Logic Unit (ALU),** which performs the basic arithmetic and logical operations.
- **Registers** which are used for information storage within the processor.
- **Internal Data Paths:** These paths are useful for moving the data between two registers or between a register and ALU.
- **External Data Paths:** The roles of these data paths are normally to link the processor registers with the memory or I/O interfaces. This role is normally fulfilled by the system bus.
- **The Control Unit:** This causes all the operations to happen in the processor.

The micro-operations performed by the processor can be classified as:

- Micro-operations for data transfer from register-register, register-memory, I/O-register etc.
- Micro-operations for performing arithmetic, logic and shift operations. These micro-operations involve use of registers for input and output.

The basic responsibility of the control unit lies in the fact that the control unit must be able to guide various components of processor to perform a specific sequence of micro-operations to achieve the execution of an instruction.

What are the functions, which a control unit performs to make an instruction execution feasible? The instruction execution is achieved by executing micro-operations in a specific sequence. For different instructions this sequence may be different. Thus, the control unit must perform two basic functions:

- Cause the execution of a micro-operation.
- Enable the processor to execute a proper sequence of micro-operations, which is determined by the instruction to be executed.

But how are these two tasks achieved? The control unit generates control signals, which in turn are responsible for achieving the two tasks stated above. But how are these control signals generated? We will answer this question in later sections. First let us discuss a simple structure of control unit.

Structure of Control Unit

A control unit has a set of input values based on which it produces an output control signal, which in turn performs micro-operations. These output signals control the execution of a program. A general model of control unit is shown in Figure 11.1.

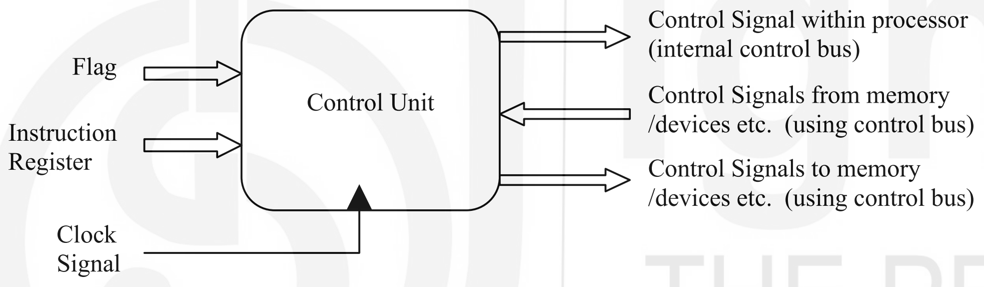

Flag
Instruction Register
Clock Signal
Control Unit
Control Signal within processor (internal control bus)
Control Signals from memory /devices etc. (using control bus)
Control Signals to memory /devices etc. (using control bus)

Figure 11.1: A General Model of Control Unit

In the model given above the control unit is a black box, which has certain inputs and outputs.

The inputs to the control unit are:

- **The Master Clock Signal:** This signal causes micro-operations to be performed in a sequence. In a single clock cycle either a single or a set of simultaneous micro-operations can be performed. The time taken in performing a single micro-operation is also termed as processor cycle time or the clock cycle time in some machines.

- **The Instruction Register:** It contains the operation code (opcode) and addressing mode bits of the instruction. It helps in determining various instruction cycles, as given in Unit 10, that are to be performed for a specific instruction and hence determines the related micro-operations.

- **Flags:** Flag represent the conditional codes that can be used in decision making. Flags are set by the ALU operations.For example, a zero flag, if set,communicates to the control unit that the result of last ALU operations was zero. Thus, if processor wasexecuting theISZ instruction(skip the next instruction if zero flag is set), the next instruction should be skipped. This action is initiated by the control unit, whichwouldincrementPC by one program instruction length, thus skipping the next instruction.

- **Control Signals from Control Bus:** Some of the control signals are provided to the control unit through the control bus. These signals are issued from outside the processor. Some of these signals are interrupt signals and acknowledgement signals.

Based on the input signals the control unit activates certain output control signals, which in turn are responsible for execution of an instruction. These output control signals are:

- **Control signals, which are required within the processor:** These control signals cause two types of micro-operations, viz., for data transfer from one register to another; and for performing an arithmetic, logic and shift operation using ALU.

- **Control signals to control bus:** These control signals transfer data from or to processor register to or from memory or I/O interface. These control signals are issued on the control bus to activate a data path on the data / address bus etc.

A control unit must know how all the instructions would be executed. It should also know about the nature of the results and the indication of possible errors. All this is achieved with the help of flags, opcodes, clock andcontrol signals.

A control unit contains a clock portion that provides clock-pulses. This clock signal is used for determining the timing sequence of the micro-operations. In general, the timing signals from control unit are kept sufficiently long to accommodate the propagational delays of signals within the processor along various data paths. Since within the same instruction cycle different control signals are generated at different times for performing different micro-operations, therefore a counter can be utilised with the clock to keep the count. However, at the end of each instruction cycle the counter should be reset to the initial condition. Thus, the clock to the control unit must provide counted timing signals. Examples, of the functionality of control units along with timing diagrams are given in further readings.

How are these control signals applied to realisemicro-operations? *The control signals are applied directly as the binary inputs to the logic gates of the logic circuits that are responsible for implementing micro-operations.* All these inputs are the control signals, which are applied to select a circuit (for example, select or enable input) or a path (for example, multiplexers) or any other operation in the logic circuits.

11.3 THE HARDWIRED CONTROL

In the last section, we have discussed the control unit in terms of its inputs, output and functions. A variety of techniques have been used to organize a control unit. Most of them fall into two major categories:

1. Hardwired control organization
2. Microprogrammed control organization.

In the hardwired organization, the control unit is designed as a combinational circuit or a sequential circuit. In other words, control units is implemented using logic gates, flip-flops, counter circuits, decoder circuit, select and enable input to various logic circuits etc. Since the hardwired control unit is made up of fast logic circuits, it can be optimised for fast operations.
The block diagram of a hardwired control unit is shown in Figure 11.2. The major input to the circuit are instruction register, the clock, and the flags. The control unit uses the opcode of instruction stored in the IR register to perform different actions. The opcode can be used to decode a typical sequence of control signals that are

responsible for execution of that instruction. Selecting separate instruction line for each instruction simplifies the control logic. This control line selection can be performed by a decoder. Decoder are explained in the Unit 3, it uses n input line to select one output line out of 2^n lines.

The clocksignal is input to sequencing logic andforms one of the input of control unit. The sequencing logic issues a repetitive sequence of pulses for the execution of micro-operation(s). These timing signals control the sequence of execution of instruction and determine what control signal needs to be applied at what time for instruction execution. Please note a typical example, of timing sequence for execution of micro-operations of different sub-cyclesof an instruction are given in Unit 10

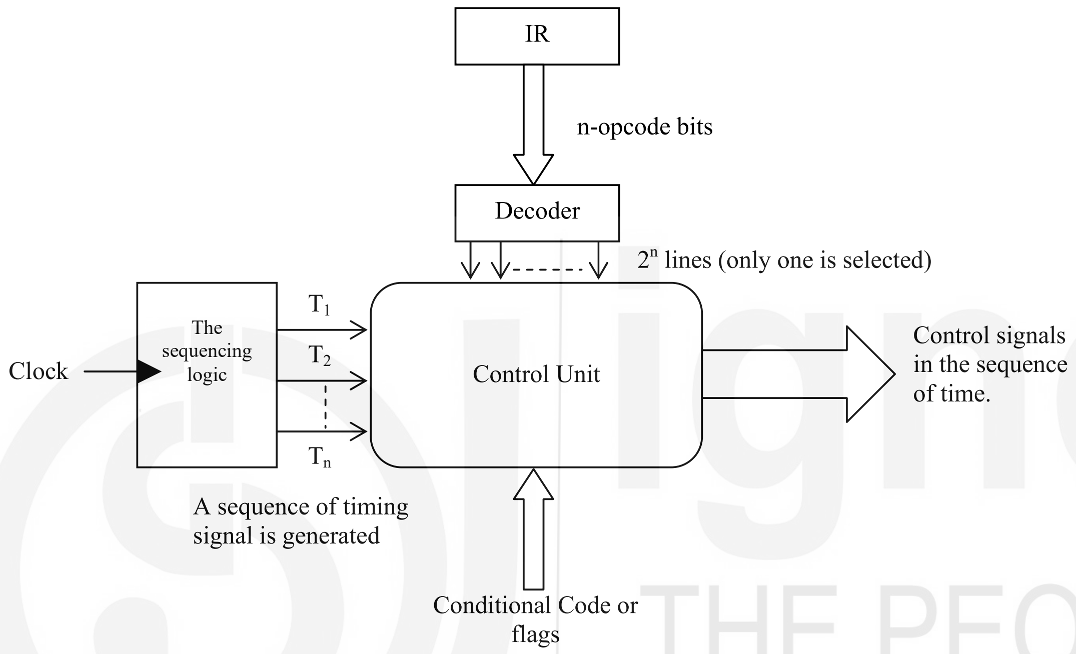

IR
n-opcode bits
Decoder
2^n lines (only one is selected)
Clock
The sequencing logic
T_1
T_2
T_n
Control Unit
Control signals in the sequence of time.
A sequence of timing signal is generated
Conditional Code or flags

Figure 11.2: Block Diagram of Hardwired Control Unit

☞Check Your Progress 1

1. What are the inputs to control unit?

....
....
....

2. How does a control unit control the instruction cycle?

....
....
....

3. What is a hardwired control unit?

....
....
....

11.4 WILKES CONTROL

Prof. M. V. Wilkes of the Cambridge University Mathematical Laboratory coined the term microprogramming in 1951. He provided a systematic alternative procedure for designing the control unit of a digital computer. During instruction executing a machine instruction, a sequence of data transformations due to arithmetic, logic and shirt micro-operations and transfer of information from one register in the processor due to register transfer micro-operations take place. **Because of the analogy between the execution of individual steps in a machine instruction to the execution of the individual instruction in a program, Wilkes introduced the concept of microprogramming.** The Wilkes control unit replaces the sequential and combinational circuits of hardwired control unit by a simple control unit in conjunction with a storage unit that stores the sequence of steps of instruction that is a micro-program.

In Wilkes microinstruction has two major components:

a) Control field which indicates the control lines which are to be activated and
b) Address field, which provides the address of the next microinstruction to be executed.

Figure 11.3 is an example of Wilkes control unit design.

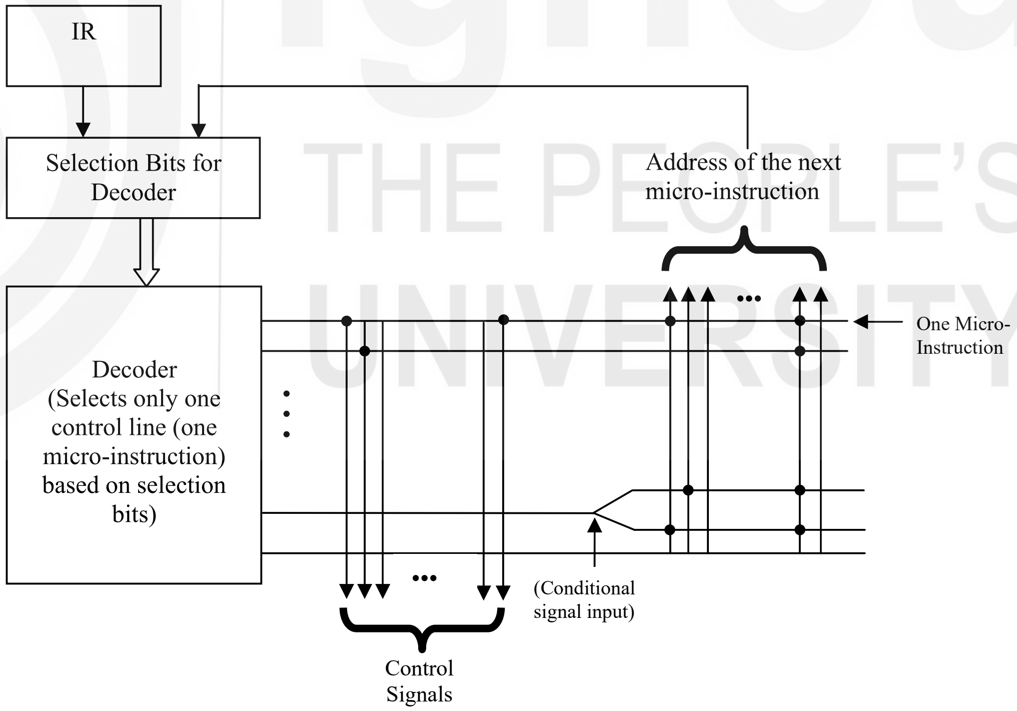

IR
Selection Bits for Decoder
Address of the next micro-instruction
Decoder (Selects only one control line (one micro-instruction) based on selection bits)
One Micro-Instruction
(Conditional signal input)
Control Signals

Figure 11.3: Wilkes Control Unit

The control memory in Wilkes control was organized, as a PLA's like matrix made of diodes. Each horizontal line on this matrix consists of two components, viz. the control signals and the address of the next micro-instruction. The next micro-instruction register stores the address of the next micro-instruction to be loaded. Please note that this register should be loaded at the falling edge of clock, that is once the previous micro-instruction completes its execution. The next micro-instruction to be executed is either specified by IR, after the instruction has been decoded or by the address of the next micro-instruction as specified in the micro-instruction itself. The register input passes through the address decoder and the decoded line of the control matrix is selected for generating the control signals for the processor.The Wilkes control unit also provides handling of conditions. For example, a condition like zero flag can be attached to the conditional signal input, which determines the micro-instruction to be executed next. More details on the Wilkes control unit may be studied from the further readings.

11.5 THE MICRO-PROGRAMMED CONTROL

An alternative to a hardwired control unit is a micro-programmed control unit. Wilke's control unit is one of the examples of micro-program control unit. A micro-program is also called firmware (midway between the hardware and the software). It consists of:

(a) One or more micro-operations to be executed; and
(b) The information about the micro-instruction to be executed next.

The general configuration of a micro-programmed control unit is shown in Figure 11.4.

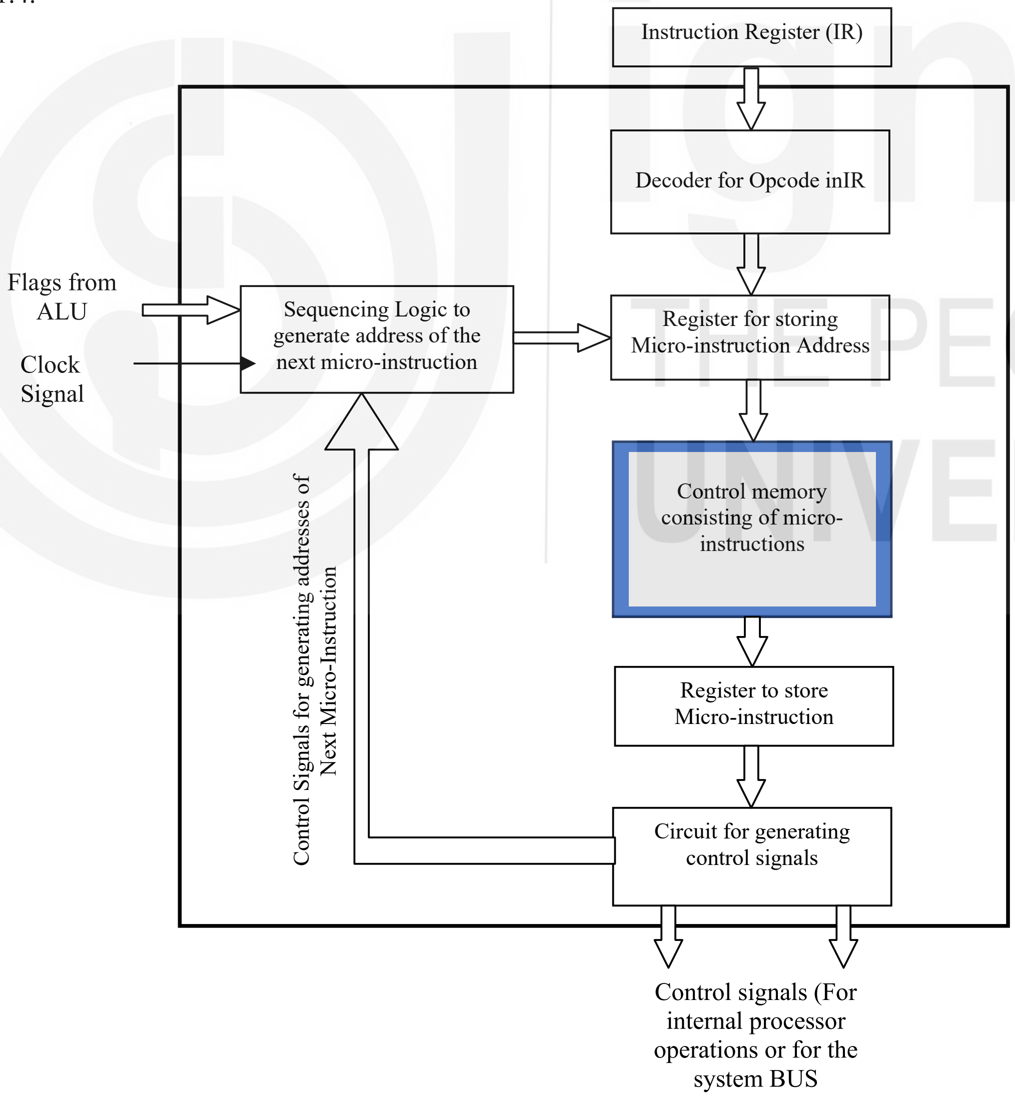

Instruction Register (IR)
Decoder for Opcode inIR
Flags from ALU
Clock Signal
Sequencing Logic to generate address of the next micro-instruction
Register for storing Micro-instruction Address
Control memory consisting of micro-instructions
Register to store Micro-instruction
Control Signals for generating addresses of Next Micro-Instruction
Circuit for generating control signals
Control signals (For internal processor operations or for the system BUS

Figure 11.4: Operation of Micro-Programmed Control Unit

The control memory of the micro-programmed control unit stores the micro-instructions. The micro-instruction address register stores the address of the micro-instruction, which should be used to generate the control signal and the address of next micro-instruction.The micro-instruction register stores the last read micro-instruction, which is used to generate the control signals for performing micro-operations and control signals for generating address of the next micro-instruction. . A micro-instruction execution primarily involves the generation of desired control signals and signals used to determine the next micro-instruction to be executed. The sequencing logic of this control unit loads the micro-instruction address register. It also issues a read command to control memory, which stores the micro-instrucitons. The following functions are performed by the micro-programmed control unit:

1. The sequence logic unit specifies the address of the control memory word which contains the micro-instruction that is to be read, in the micro-instruction address register of the Control Memory. It also issues the READ signal to the control memory, so that the desired micro-instruction can be read.
2. The desired control memory word containing the desired micro-instruction is read into the micro-instruction register.
3. The micro-instruction register forms the input to the logic circuit that generates the control signals based on the current micro-instruction. Further, this circuitry also generates the control signals that can be used by sequencing logic to generate the address of micro-instructionin the control memory that is to be executed next.
4. The sequencing logic uses the control signals, as stated above, and flag register to computethe address of the micro-instruction that is to be executed next.

As we have discussed earlier, the execute cycle steps of micro-operations are different for all instructions in addition the addressing mode may be different. All such information generally is stored is coded in the instruction, which is stored in the instruction Register (IR). The IR input to Micro-Instruction Address Register for Control Memory is used for determining the micro-instruction, which performs the execute cycle of the instruction. The decoder after IR uses the IR register to generate the address of the first micro-instruction in control memory for the specified instruction in IR (Refer to Figure 11.4).

☞Check Your Progress 2

1. What is firmware? How is it different from software?
   ……………………………………………………………………………………..
2. **State True or False**
   - (a) A micro-instruction can initiate only one micro-operation at a time.
   - (b) A control word is equal to a memory word.
   - (c) Micro-programmed control is faster than hardwired control.
   - (d) Wilkes's control does not provide a branching micro-instruction.
3. What will be the control signals and address of the next micro-instruction in the Wilkes control example of Figure 11.3, if the entry address for a machine instruction selects the branching control lineand the conditional bit value for branch is true (assume that out of the two branching lines the bottom line is selected when condition is true)?
   ....
   ....………
   ……………………………………………………………………………………..

11.6 THE MICRO-INSTRUCTIONS

A micro-instruction, as defined earlier, is an instruction of a micro-program. It specifies one or more micro-operations, which can be executed simultaneously. On executing a micro-instruction, a set of control signals are generated which in turn cause the desired micro-operation to happen.

11.6.1 Types of Micro-instructions

In general, the micro-instruction can be categorised into two general types. These are branching and non-branching. After execution of a non-branching micro-instruction the next micro-instruction is the one following the current micro-instruction.However, the sequences of micro-instructions are relatively small and last only for 3 or 4 micro-instructions.
A conditional branching micro-instruction tests conditional variable, or a flag generated by an ALU operation. Normally, the branch address is contained in the micro-instruction itself.

11.6.2 Control Memory Organisation

The next important question about the micro-instruction is: how are they organized in the control memory? One of the simplest ways to organize control memory is to arrange micro-instructions for various sub cycles of the machine instruction in the memory. The Figure 11.5 shows such an organisation.

00h	Micro-instructions for fetch cycle	Fetch cycle
01h	···	
02h	···	
03h	Jump to Indirect or Execute Cycle using the addressing mode bits of the instruction	
04h	Micro-instruction for Indirect cycle	Indirect cycle
05h	···	
06h	···	
07h	Jump to Execute cycle	
08h	Micro-instruction for interrupt initiation	Interrupt cycle
09h	···	
0Ah	···	
0Bh	Jump to fetch cycle	
0Ch	Compute address of micro-instruction based on opcode	Execute cycle
0Dh	···	
0Eh	…	
0Fh	Jump to the micro-instruction of opcode	
10h	Micro-instructions for opcode 0 (Let us say LOAD)	Micro-instructions for opcode 0
11h	···	
12h	···	
13h	Jump to fetch or interrupt cycle	
14h	Micro-instruction for opcode 1 (Let us say ADD)	Micro-instructions for opcode 1
15h	···	
16h	···	
17h	Jump to fetch or interrupt cycle	
…	…	…
F8h	Micro-instruction for last opcode (Let us say ISZ)	Micro-instructions for opcode ISZ
F9h	···	
FAh	···	
FBh	···	
FCh	···	
FDh	···	
FEh	···	
FFh	Jump to fetch or interrupt cycle	

Figure 11.5: An example of Control Memory Organisation

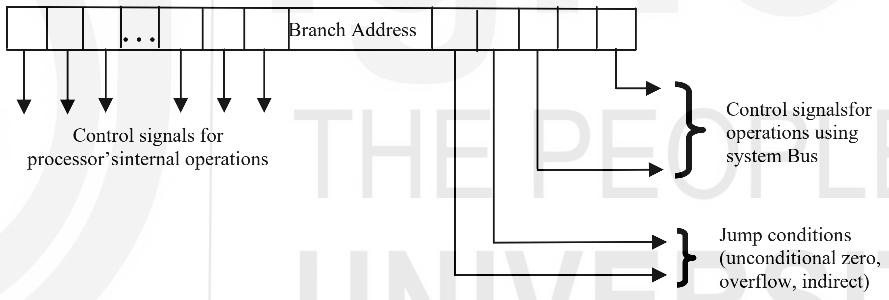
Let us give an example of control memory organization. Let us take a machine instruction: Branch on zero. This instruction causes a branch to a specified main memory address in case the result of the last ALU operation is zero, that is, the zero flag is set. The pseudocode of the micro-program for this instruction may be written as:

```
            Test "Zero flag";if SET branch to micro-code at label ZERO
            Unconditional branch to micro-code of label NON-ZERO
ZERO:       Microcode ofsequence of micro-operations required to be executed
            to replace the PC by the effective address of the instruction
            operand.
NON-ZERO:   Branch to Interrupt or Fetch cycle.
```

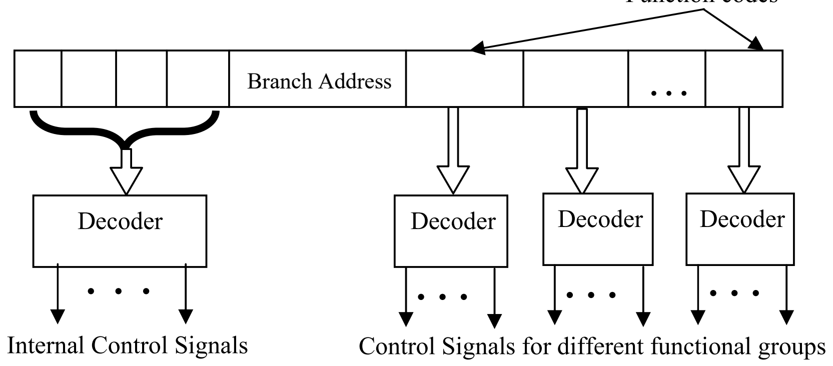
Please note that in case the Zero flag is not SET, then no operation is needed, as next instruction in sequence is to be executed, whose address is already in PC. Thus, next instruction should be fetched.

11.6.3 Micro-instruction Formats

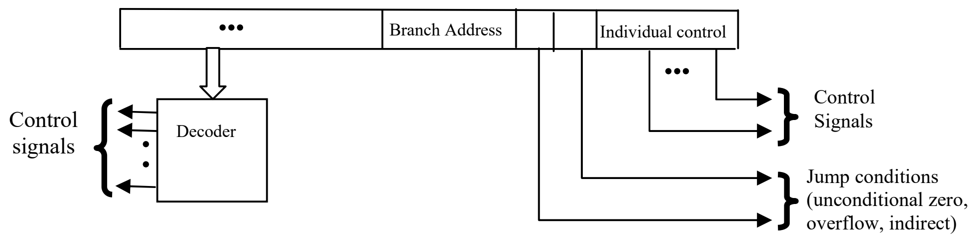
Now let us focus on the format of a micro-instruction. The two widely used formats used for micro-instructions are horizontal and vertical. In the horizontal micro-instruction, each bit of the micro-instruction represents a control signal, which directly controls a single bus line or sometimes a gate in the machine. However, the length of such a micro-instruction may be hundreds of bits. A typical horizontal micro-instruction with its related fields is shown in Figure 11(a).


. . .
Branch Address
Control signals for processor'sinternal operations
Control signalsfor operations using system Bus
Jump conditions (unconditional zero, overflow, indirect)

(a) Horizontal Micro-instruction



(b) Vertical Micro-instruction



(c) A Realistic Micro-instruction
Figure 11.6: Micro-instruction Formats

In a vertical micro-instruction, many similar control signals can be encoded into a few micro-instruction bits. For example, for 16 ALU operations, which may require 16 individual control bits in horizontal micro-instruction, only 4 encoded bits are needed in vertical micro-instruction. Similarly, in a vertical micro-instruction only 3 bits are needed to select one of the eight registers. However, these encoded bits need to be passed from the respective decoders to get the individual control signals. This is shown in Figure 11.6(b).

In general, a horizontal control unit is faster, yet requires wider instruction words, whereas vertical control units, although require a decoder, are shorter in length. Most of the systems use neither purely horizontal nor purely vertical micro-instructions Figure 11.6(c).

11.7 THE EXECUTION OF MICRO-PROGRAM

The micro-instruction cycle can consist of two basic cycles: the fetch and the execute. Here, in the fetch cycle the address of the micro-instruction is generated and this micro-instruction is put in a register used for the address of a micro-instruction for execution. The execution of a micro-instruction simply means generation of control signals. These control signals may drive the processor (internal control signals) or the system bus. The format of micro-instruction and its contents determine the complexity of a logic module, which executes a micro-instruction.These logic module depends on the encoding of micro-instructions, which is discussed next.

11.7.1 Micro-instruction Encoding

One of the key features incorporated in a micro-instruction is the encoding of micro-instructions. What is encoding of micro-instruction? For answering this question let us recall the Wilkes control unit. In Wilkes control unit, each bit of information either generates a control signal or form a bit of next instruction address. Now, let us assume that a machine needs N total number of control signals. If youare using the Wilkes scheme you require N bits for the control signals, one for each control signal in the control unit. In addition, Wilkes control unit also stores the address of the next micro-instruction using address bits.

Since we are dealing with binary control signals, therefore, a ‘N’ bit micro-instruction can represent 2^N combinations of control signals.

The question is do we need all these 2^N combinations?

No, some of these 2^N combinations are not used because:

1. Two sources may be connected by respective control signals to a single destination; however, only one of these sources can be used at a time. Thus, the

combinations where both these control signals are active for the same destination are redundant.

2. A register cannot act as a source and a destination at the same time. Thus, such a combination of control signals is redundant.
3. You can provide only one pattern of control signals at a time to ALU, making some of the combinations redundant.
4. You can provide only one pattern of control signals at a time to the external control bus also.

Therefore, you do not need 2^N combinations. Supposeyou only need 2^K (where $K< N$) combinations, then you need only K encoded bits instead of N control signals. The K bit micro-instruction is an extreme encoded micro-instruction. Let us touch upon the characteristics of the extreme encoded and unencoded micro-instructions:

Unencoded micro-instructions

- One bit is needed for each control signal; therefore, the number of bits required in a micro-instruction is high.
- It presents a detailed hardware view, as control signal need can be determined.
- Since each of the control signals can be controlled individually, therefore these micro-instructions are difficult to program. However, concurrency can be exploited easily.
- Almost no control logic is needed to decode the instruction as there is one to one mapping of control signals to a bit of micro-instruction. Thus, execution of micro-instruction and hence the micro-program is faster.
- The unencoded micro-instruction aims at optimising the performance of a machine.

Highly Encoded micro-instructions

- The encoded bits needed in micro-instructions are smaller in size than that ofunencoded micro-instructions.
- It provided an aggregated view that is a higher view of the processor as only an encoded sequence can be used for micro-programming.
- The encoding helps in reduction in programming burden; however, the concurrency may not be exploited to the fullest.
- Complex control logic is needed, as decoding is a must. Thus, the execution of a micro-instruction can have propagation delay through gates. Therefore, the execution of micro-program takes longer time than that of an unencoded micro-instruction.
- The highly encoded micro-instructions are aimed at optimizing programming effort.

In general, the micro-programmed control unit designs are neither completely unencoded nor highly encoded. They are slightly coded. This reduces the width of control memory and micro-programming efforts. As shown in Figure 11.7, some of the bits of micro-instructions are unencoded, therefore, can be used to generate the control signals directly. Some of the control bits are organised as fields and can be directly used as input to decoder to generate control signals. Further, a combination of decoded control signals can be passed through a second decoder to generate control signals. These decoding operations are shown in Figure 11.7.

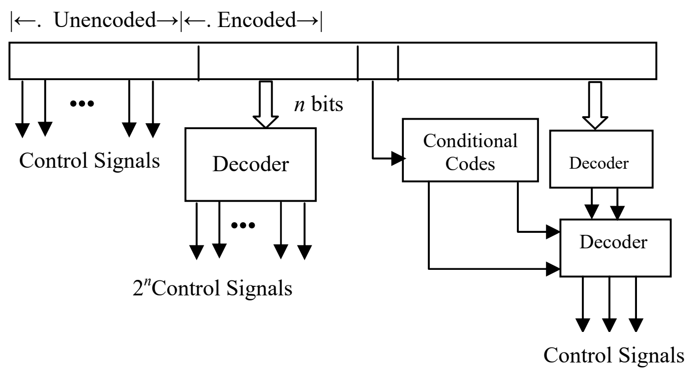


Figure 11.7: Decoding of Micro-instructions

11.7.2 Micro-instruction Sequencing

Another aspect of micro-instruction execution is the micro-instruction sequencing that involves address calculation of the next micro-instruction. In general, the next micro-instruction can be one of the following (refer Figure 11.5):

- Next micro-instruction in sequence
- Calculated on the basis of opcode
- Branch address (conditional or unconditional).

Next instruction in sequence: Figure 11.5 shows one example of control memory. This control organisation has micro-instructions for fetch, indirect and interrupt cycles followed by the execute cycle. You may recall the instruction cycle, as given in Unit 10, the fetchinstruction cycle consists of the following sequence of micro-operations:

T1: MAR ← PC
T2: DR ← (MAR)
T3: PC←PC +1; IR ← DR

One micro-instruction can be created for each timing sequence. For example, for the stated sequence of micro-operations, three micro-instructions, one each for timing T1, T2 and T3 would be created. These micro-instructions would be stored at address 00h, 01h and 02h. The micro-instruction at 03h would be a conditional branch instruction to the indirect or execute cycle. Please also note that at time T3, two micro-operations are to be performed in parallel. Thus, micro-instruction at 02h should generate control signals, which result in the related units of the processor to perform the increment operation on PC and transfer of DR to IR simultaneously.
The micro-instruction at 00h to 03h are to be executed in a sequence to perform the desired operation of instruction fetch.

Branch address (conditional or unconditional): You may please note that the last micro-instruction for instruction fetch is the conditional branch instruction. Please also note that in Figure 11.5, the indirect cycle starts at an address 04h and the execution cycle starts at 0Ch. Thus, the micro-instruction at address 03h would be a conditional branch instruction, having the condition, if the indirect bit is set of not. This conditional branch would be taken to address 0Ch (the start of execution cycle) in case the indirect bit is CLEAR.Thus, if indirect bit is SET, then the next micro-instruction in sequence would be executed, which will be the starting micro-instruction of the indirect cycle that will convert the indirect operand to direct operand. Please also note in the Figure 11.5, that the last micro-instruction of the indirect cycle is an unconditional jump instruction to the execute cycle.

Calculated on the basis of opcode: The opcode of the Instruction Register is used to decode the operation that is to be performed on the operands. The control unit supports this decode operation. In the case of micro-programmed control unit, this opcode can be used to compute the address of the first micro-instruction to be executed to perform the operation. In Figure 11.5, the execute cycle contains the

micro-instructions that perform jump to the micro-instruction address of the desired operation.
We will explain it with the help of an example, assume that in Figure 11.5, the micro-instructions related to opcodes start from micro-instruction address 00h instead of 10h and the micro-instructions of fetch, indirect, interrupt and execute cycles starts at micro-instruction address F0h, F4h, F8h and FCh respectively.Further, we assume that operation of each opcode is performed using just four micro-instructions. Since the control memory has addresses from 00h to FFh, out of which 00h to EFh(a total of F0h) addresses are for storing micro-instructions of various opcodes. Therefore, this control memory can contain micro-instructions for F0h/4h = 3Ch opcodes. Thus, the possible opcodes for such a machine would be $00000000_2$to 00111011_2, or 000000_2 to 111011_2.How these opcodes would be mapped to the related micro-instruction start address. The following table shows these mapping:

Opcode	Starting address of related Micro-instructions	
Binary	Binary	Hexadecimal
0000 00	0000 0000	00
0000 01	0000 0100	04
0000 10	0000 1000	08
0000 11	0000 1000	0C
0001 00	0001 0000	10
0001 01	0001 0100	14
0001 10	0001 1000	18
0001 11	0001 1000	1C
…	…	
1001 00	1001 0000	90
1001 00	1001 0100	94
…	…	
1110 00	1110 0000	E0
1110 01	1110 0100	E4
1110 10	1110 1000	E8
1110 11	1110 1000	EC

Figure 11.8: Mapping of opcode to micro-instruction address

The interesting part of this example is the mapping from the opcode to the micro-instruction address. In Figure 11.8, an opcode is of 6-bit length. To map an opcode to the first related micro-instruction, you just need to append two 0 bits at the least significant position.For example, an opcode 1001 01 will be mapped to a micro-instruction address 1001 0100 or 94h.

Please note different kind of control memory and opcode organisation will make this computational logic a complex one. In any case, you can design the related logic circuit for calculating the micro-instruction address for a given opcode.

You must refer to further readings for more detailed information on Micro-programmed Control Unit Design.

☞Check Your Progress 3

1. **State True or False**

   a) A branch micro-instruction can have only an unconditional jump.

   b) Control store stores opcode-based micro-programs.

   c) A true horizontal micro-instruction requires one bit for every control signal.

d) A decoder is needed to find a branch address in the vertical micro-instruction.

e) One of the responsibilities of sequencing logic (Refer Figure 11.4) is to cause reading of micro-instruction addressed by a micro-program address register. ☐

f) Status bits supplied from ALU to sequencing logic have no role to play with the sequencing of micro-instruction.

2. What art the possibilities for the next instruction address?
....
....
....
....
....

3. How many address fields are there in Wilkes Control Unit?
....
....
....

4. Compare and contrast unencoded and highly encoded micro-instructions.
....
....
....

11.8 SUMMARY

In this unit we have discussed the organization of control units. Hardwired, Wilkes and micro-programmed control units are also discussed. The key to such control units are micro-instruction, which can be briefly (that is types and formats) described in this unit. The function of a micro-programmed unit, that is, micro-programmed execution, has also been discussed. The control unit is the key for the optimised performance of a computer. The information given in this unit can be further appended by going through further readings.

11.9 SOLUTIONS/ANSWERS

Check Your Progress 1

1. IR, Timing Signal, Flags Register
2. The control unit issues control signals that cause execution of micro-operations in a pre-determined sequence. This enables execution sequence of an instruction.
3. A logic circuit-based implementation of control unit.

Check Your Progress 2

1. Firmware is basically micro-programs, which are used in a micro-programmed control unit. Firmwareare more difficult to write than software.

2. (a) False (b) False (C) False (d) False

3. Please check the Figure 11.3 from left to right and select the bottom branch line.
   The control signals would be 000…00
   Address of next micro-instruction would be: 100…10

Check Your Progress 3

1. (a) False (b) False (c) True (d) False (e) True (f) False.

2 The address of the next micro-instruction can be one of the following:

   - the address of the next micro-instruction in sequence.
   - determined by opcode using mapping or any other method.
   - branch address supplied on the internal address bus.

3. Wilkes control typically has one address field. However, for a conditional branching micro-instruction, it contains two addresses. The Wilkes control, in fact, is a hardware representation of a micro-programmed control unit.

4.

Unencoded Micro instructions	Highly encoded
• Large number of bits • Difficult to program • No decoding logic • Optimizes machine performances • Detailed hardware view	Relatively less bits Easy to program Need decoding logic Optimizes programming effort Aggregated view