Module 3

Control Unit Design

Motivation:

The central processing unit is termed as the brain of the computer. It involves millions of data transfers and a number of activities lined up for any particular processing. In order to manage the flow of data and sequence all the activities, a controller is required. Hence the control unit is designed to coordinate the various activities of the computer.

Syllabus:

Lecture no	Content	Duration (Hr)	Self-Study (Hrs)
1	Control Unit: Soft wired (Micro-programmed) and Hardwired control unit design methods	1	2
2	Address sequencing	1	2
3	Microprogram Sequencer, Micro operation	1	2
4	Micro instruction Format, Control Memory	1	2
5	Concepts of nano programming	1	2
6	Introduction to RISC and CISC architectures and design issues	1	2

Learning Objectives:

Learners shall be able to:

1. To explain the design of hardwired and microprogrammed control unit.
2. To describe the process of address sequencing in micropragrammed control unit
3. To explain about micro-operations
4. To identify the fields in a microinstruction
5. To describe the concept of nano programming
6. To compare RISC and CISC Architectures

Theoretical Background:

In a microcomputer the entire CPU is contained on a tiny chip called a microprocessor. Every CPU has three main components. i) Control Unit ii) Arithmetic Logic Unit iii) Register Registers are used for storing intermediate results during program execution, registers are high speed memory unit, and size of register is one of the important factors to measure performance of computer system. Size of register is also called word length or word size, word can be 8bits ,16 bits, 32 bits, 64 bits. More the word size

more number of addressable location it has, if word size is k bits, 2k addresses constitute the address space of the computer. Register can be of many types:

- General purpose Register
- Special purpose Register
- Data Register
- Address Register
- Instruction Register
- Program Counter

To execute any program the PC (program Counter) must have the address of instruction which is to be executed next, and this address is transfer to the MAR (Memory Address Register), by referring this address actual data is fetch from Main Memory of computer and stored in MDR (Memory Data Register). MDR transfer the data to IR for decoding, and PC is updated with next address for next instruction cycle. The Control Unit along with the Instruction Register interprets the machine Language instruction and issues the control signals to make the CPU execute that instruction. The ALU (Arithmetic Logic Unit) does the arithmetic and logic and the communication between different components takes place through internal bus system.

Key Definitions:

Central Processing Unit (CPU): It is the main part of the computer, its 'brain', consisting of the central memory, arithmetic logic unit and control unit. It is also called the central processor.

Hardwired: To connect (electronic components, for example) by electrical wires or cables.

Hardwired Control Unit: When the control signals are generated by hardware using conventional logic design techniques, the control unit is said to be hardwired.

Control Unit: A control unit is the part of a CPU or other device that directs its operation. The outputs of the unit control the activity of the rest of the device. A control unit can be thought of as a finite state machine.

Microprogrammed Control Unit: A control unit whose binary control variables are stored in memory is called a micro programmed control unit.

Control word: A word with each bit for one of the control signals. Each step of the instruction execution is represented by a control word with all of the bits corresponding to the control signals needed for the step set to one.

Microcode: Microprogramming (i.e. writing microcode) is a method that can be employed to implement machine instructions in a CPU relatively easily, often using less hardware than with other methods.

Micro sequencer: In computer architecture and engineering, a sequencer or microsequencer is a part of the control unit of a CPU. It generates the addresses used to step through the microprogram of a control store.

Control Store: A control store is the part of a CPU's control unit that stores the CPU's microprogram. It is usually accessed by a microsequencer.

Micro operations: In computer central processing units, micro-operations (also known as a micro-ops or μops) are detailed low-level instructions used in some designs to implement complex machine instructions (sometimes termed macro-instructions in this context).

Microprocessor: A microprocessor incorporates the functions of a computer's central processing unit (CPU) on a single integrated circuit (IC). The microprocessor is a multipurpose, programmable device that accepts digital data as input, processes it according to instructions stored in its memory, and provides results as output.

Course Content:

Lecture 1

Control Unit: Soft wired (Micro-programmed) and Hardwired control unit design methods:

For each instruction, the control unit causes the CPU to execute a sequence of steps correctly. In reality, there must be control signals to assert lines on various digital components to make things happen. For example, when we perform an Add instruction in assembly language, we assume the addition takes place because the control signals for the ALU are set to "add" and the result is put into the AC. The ALU has various control lines that determine which operation to perform.

We can take one of two approaches to ensure control lines are set properly. The first approach is to physically connect all of the control lines to the actual machine instructions. The instructions are divided up into fields, and different bits in the instruction are combined through various digital logic components to drive the control lines. **This is called hardwired control**, and is illustrated in figure (3.1)

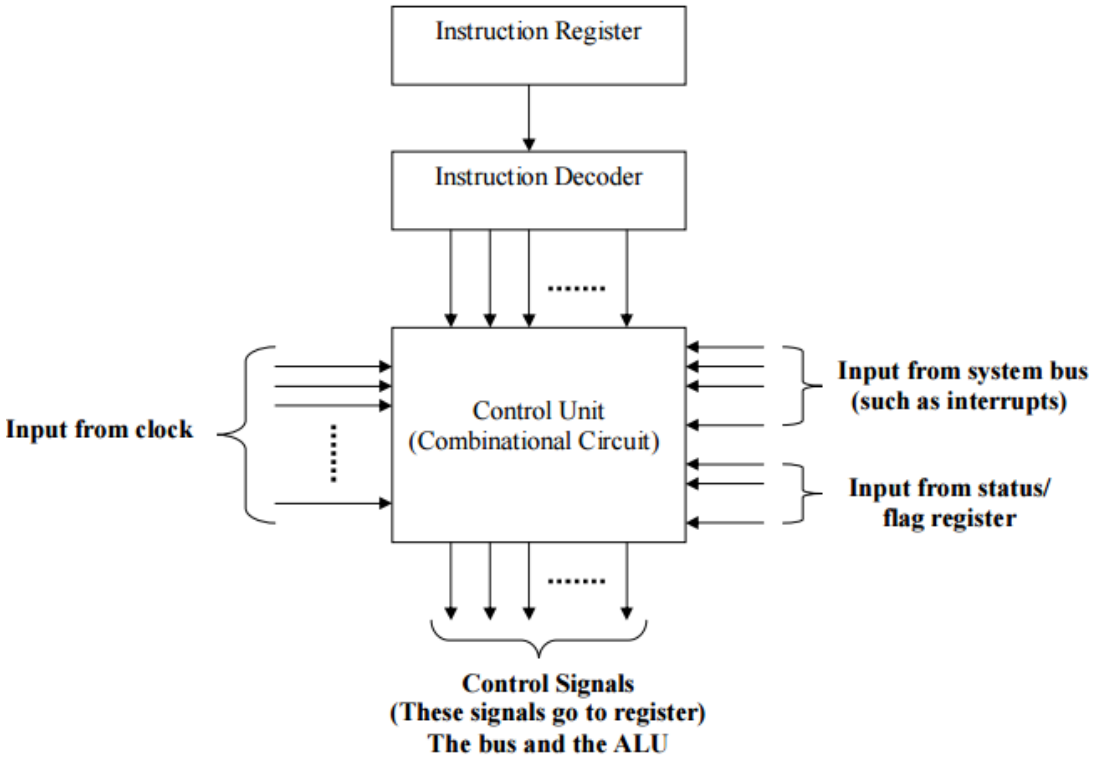


Figure 3.1 Hardwired Control Unit Organization

The control unit is implemented using hardware (for example: NAND gates, flip-flops, and counters).We need a special digital circuit that uses , as inputs, the bits from the Opcode field in our instructions, bits from the flag (or status) register, signals from the bus, and signals from the clock. It should produce, as outputs, the control signals to drive the various components in the computer.

The **advantage of hardwired control** is that is very fast. The **disadvantage** is that the instruction set and the control logic are directly tied together by special circuits that are complex and difficult to design or modify. If someone designs a hardwired computer and later decides to extend the instruction set, the physical components in the computer must be changed. This is prohibitively expensive, because not only must new chips be fabricated but also the old ones must be located and replaced.

Microprogramming is a second alternative for designing control unit of digital computer (uses software for control). A control unit whose binary control variables are stored in memory is called a microprogrammed control unit. The control variables at any given time can be represented by a string of 1's and 0's called a control word (which can be programmed to perform various operations on the component of the system). Each word in control memory contains within it a microinstruction. The microinstruction specifies one or more microoperatiotins for the system. A

sequence of microinstructions constitutes a microprogram. A memory that is part of a control unit is referred to as a control memory.

A more advanced development known as dynamic microprogramming permits a microprogram to be loaded initially from an auxiliary memory such as a magnetic disk. Control units that use dynamic microprogramming employ a writable control memory; this type of memory can be used for writing (to change the microprogram) but is used mostly for reading.

The general configuration of a microprogrammed control unit is demonstrated in the block diagram of Figure (3.2). The control memory is assumed to be a ROM, within which all control information is permanently stored.

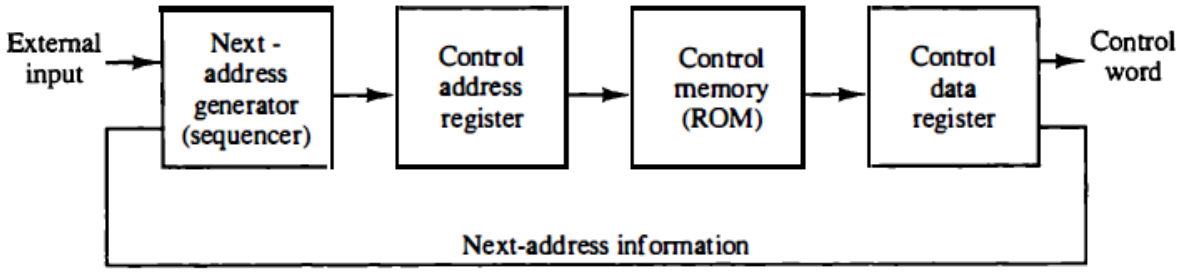


Figure 3.2 Microprogrammed Control Unit Organization

The control memory address register specifies the address of the microinstruction, and the control data register holds the microinstruction read from memory. The microinstruction contains a control word that specifies one or more micro-operations for the data processor. Once these operations are executed, the control must determine the next 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.

While the micro-operations are being executed, the next address is computed in the next address generator circuit and then transferred into the control address register to read the next microinstruction. Thus a microinstruction contains bits for initiating micro-operations in the data processor part and bits that determine the address sequence for the control memory. The next address generator is sometimes called a micro-program sequencer, as it determines the address sequence that is read from control memory. Typical functions of a micro-program sequencer are incrementing the control address register by one, loading into the control address register an address from control memory, transferring an external address, or loading an initial address to start the control operations. The control data register holds the present microinstruction while the next address is computed and read from memory. The data register is sometimes called a pipeline register. It allows the execution of the micro-operations specified by the control word simultaneously with the generation of the next microinstruction. This configuration requires a two-phase clock, with one clock applied to the address register and the other to the data register. The main advantage of the micro programmed control is the fact that once the hardware configuration is established; there should be no need for further hardware or wiring changes. If we

want to establish a different control sequence for the system, all we need to do is specify a different set of microinstructions for control memory.

Let's check the take away from this lecture

1) _____ are the different type/s of generating control signals.

a) Micro-programmed

b) Hardwired

c) Micro-instruction

d) Both Micro-programmed and Hardwired

2) If the control signals are generated by combinational logic, then they are generated by a type of _____ controlled unit.

a) Micro programmed

b) Software

c) Logic

d) Hardwired

3) The type of control signal is generated based on _____

a) Clock

b) Contents of IR

c) Contents of condition flags

d) All of the mentioned

Exercise

Q.1 Draw the block diagram of hardwired control unit and explain.

Q.2 Explain the Microprogrammed Control Unit Organization.

Q.3 State the disadvantages of hardwired control unit.

Learning from this lecture: Learners will be able to understand the working of hardwired and microprogrammed control unit.

Lecture 2

Address sequencing

Microinstructions are stored in control memory in groups, with each group specifying a routine. Each computer instruction has its own micro-program routine to generate the micro-operations. 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. The following are the steps the microprogrammed control unit must undergo during the execution of a single computer instruction:

- Load an initial address into the 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 – IR holds instruction
- The control memory then goes through the routine to determine the effective address of the operand – AR holds operand address
- The next step is to generate the micro-operations that execute the instruction by considering the opcode and applying a mapping
- After execution, control must return to the fetch routine by executing an unconditional branch

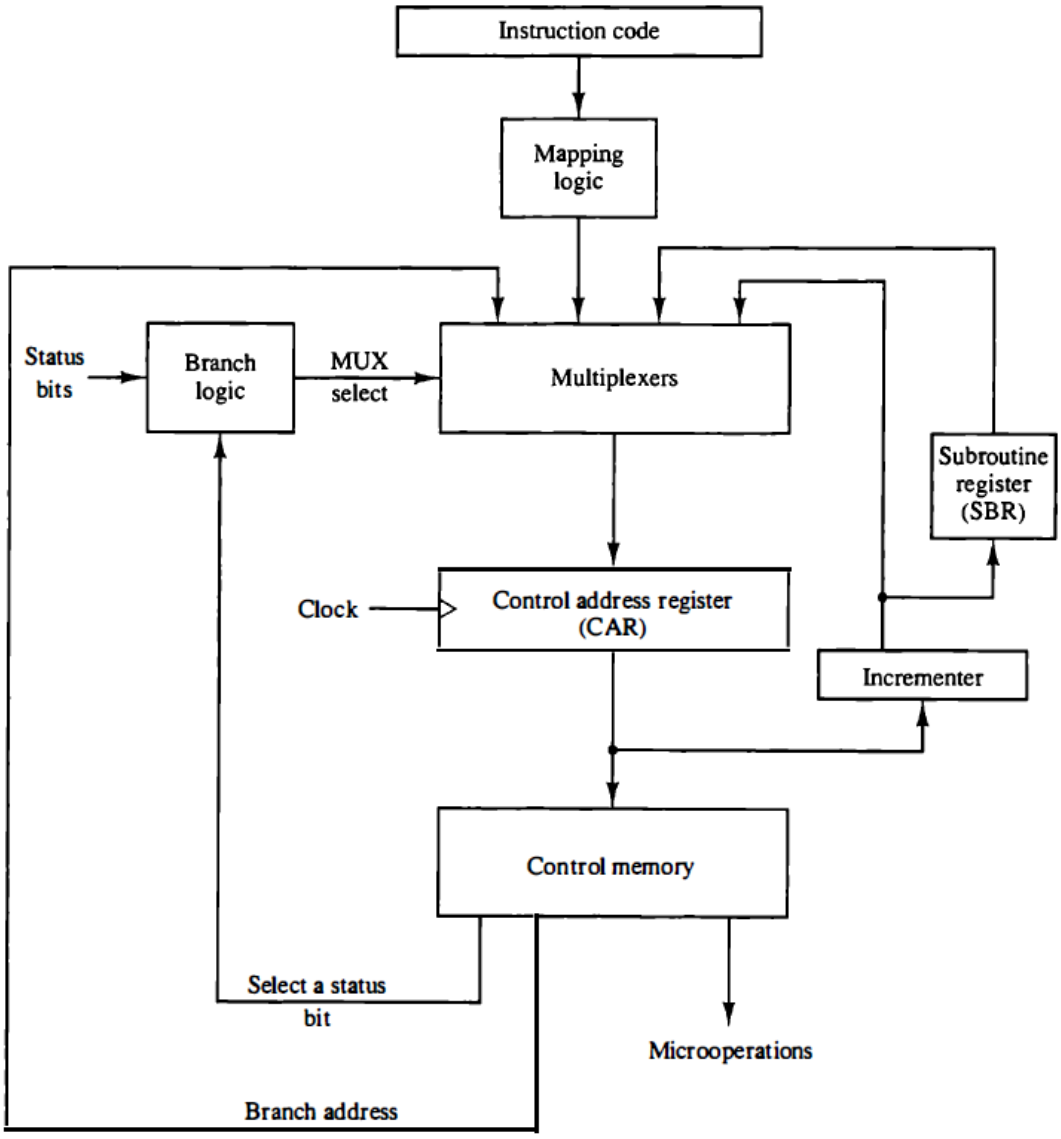


Figure 3.3 Selection of address for a control memory

Figure 3.3 shows a block diagram of a control memory and the associated hardware needed for selecting the next microinstruction address. The microinstruction in control memory contains a set of bits to initiate micro-operations in computer registers and other bits to specify the method by which the next address is obtained. 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. If there are 8 status bit conditions, then 3 bits in the microinstruction are used to specify the condition and provide the selection variables for the multiplexer. For unconditional branching, fix the value of one status bit to be one and load the branch address from control memory into the CAR. A special type of branch exists when a microinstruction specifies a branch to the first word in control memory where a micro-program 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 4-bit opcode to a 7-bit address for control memory. This provides for each computer instruction a micro-program routine with a capacity of four microinstructions.

Subroutines are programs that are used by other routines to accomplish a particular task and can be called from any point within the main body of the micro-program. Frequently many micro-programs contain identical section of code. Microinstructions can be saved by employing subroutines that use common sections of microcode. Micro-programs that use subroutines must have provisions for storing the return address during a subroutine call and restoring the address during a subroutine return. A subroutine register is used as the source and destination for the addresses

Let's check the take away from this lecture

1) A set of microinstructions for a single machine instruction is called _____

a) Program

b) Command

c) Micro program

d) Micro command

2) In a program using subroutine call instruction, it is necessary

a) initialise program counter

b) Clear the accumulator

c) Reset the microprocessor

d) Clear the instruction register

3) Microinstructions are stored in control memory groups, with each group specifying a

a) Routine

b) Subroutine

c) Vector

d) Address

Exercise

Q.1 Explain the process of address sequencing with the help of block diagram.

Q.2 Summarize the steps undergone by microprogrammed control unit during the execution of an instruction.

Q.3 Explain the branch logic in selection of address for a control memory.

.

Learning from this lecture: Learners will be able to understand the process of address sequencing in microprogrammed control unit.

Lecture 3

Microprogram Sequencer, Micro operation

Microprogram Sequencer

The basic components of a microprogrammed control unit are the control memory and the circuits that select the next address. The address selection part is called a microprogram sequencer. A microprogram sequencer can be constructed with digital functions to suit a particular application. To guarantee a wide range of acceptability, an integrated circuit sequencer must provide an internal organization that can be adapted to a wide range of applications.

The purpose of a microprogram sequencer is to present an address to the control memory so that a microinstruction may be read and executed. Commercial sequencers include within the unit an internal register stack used for temporary storage of addresses during microprogram looping and subroutine calls. Some sequencers provide an output register which can function as the address register for the control memory.

The block diagram of the microprogram sequencer is shown in figure 3.4. There are two multiplexers in the circuit. The first multiplexer selects an address from one of four sources and routes it into a control address register CAR. The second multiplexer tests the value of a selected status bit and the result of the test is applied to an input logic circuit. The output from CAR provides the address for the control memory. The content of CAR is incremented and applied to one of the multiplexer inputs and to the subroutine registers SBR. The other three inputs to multiplexer 1 come from the address field of the present microinstruction, from the output of SBR, and from an external source that maps the instruction. Although the figure 3.4 shows a single subroutine register, a typical sequencer will have a register stack about four to eight levels deep. In this way, a number of subroutines can be active at the same time. The CD (condition) field of the microinstruction selects one of the status bits in the second multiplexer. If the bit selected is equal to 1, the T (test) variable is equal to 1; otherwise, it is equal to

0. The T value together with the two bits from the BR (branch) field goes to an input logic circuit. The input logic in a particular sequencer will determine the type of operations that are available in the unit.

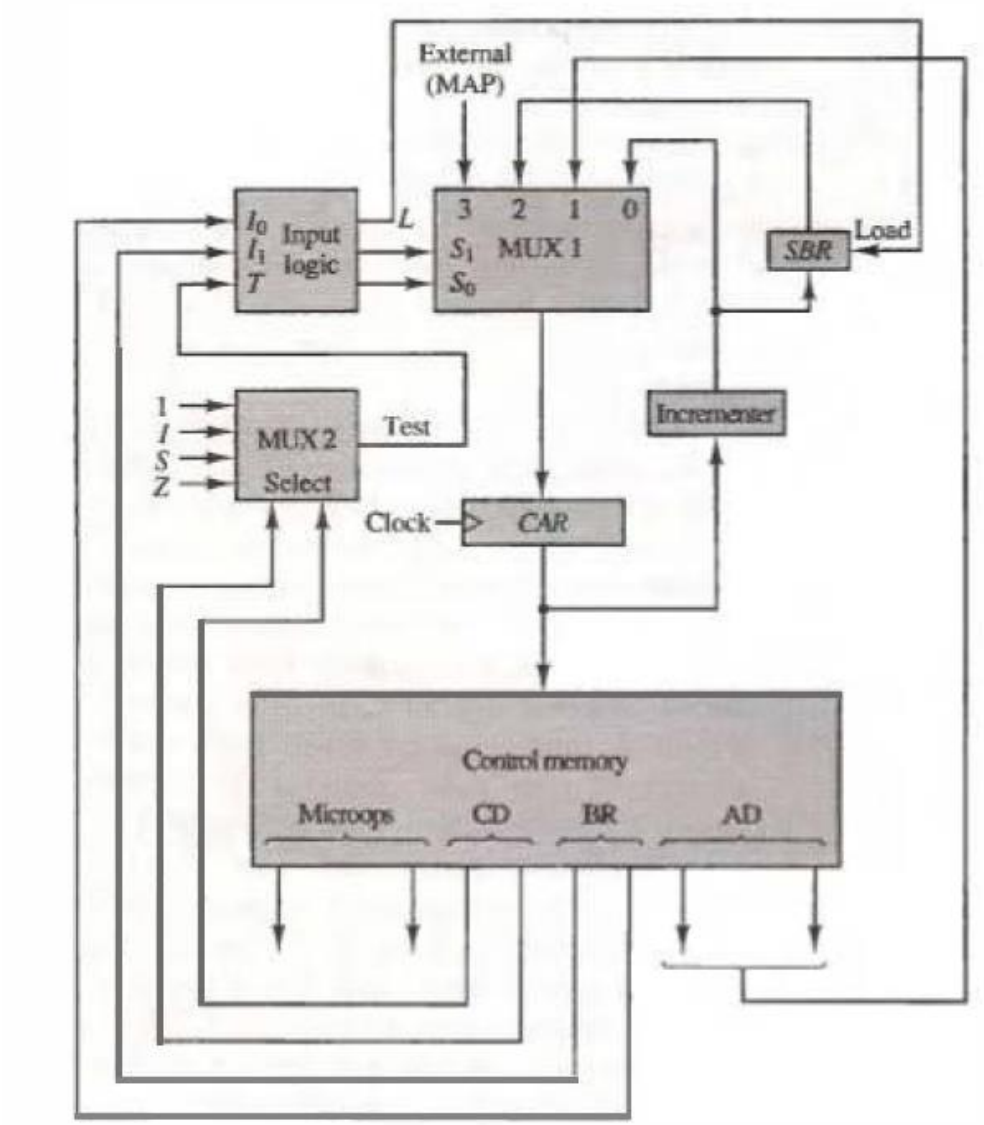


Figure 3.4: Microprogram Sequencer for a control memory

Typical sequencer operations are: increment, branch or jump, call and return from subroutine, load an external address, push or pop the stack, and other address sequencing operations. With three inputs, the sequencer can provide up to eight address sequencing operations. Some commercial sequencers have three or four inputs in addition to the T input and thus provide a wider range of operations.

The truth table for the input logic circuit is shown in the Table below. Inputs I_1 and I_0 are identical to the bit values in the BR field. The bit values for S_1 and S_0 are determined from the stated function and the path in the multiplexer that establishes the required transfer. The subroutine register is loaded with the incremented value of CAR during a call microinstruction (BR=01) provided that the status bit condition

is satisfied (T=1). The truth table can be used to obtain the simplified Boolean functions for the input logic circuit:

$S_1 = I_1$

$S_0 = I_1 I_0 + I_1'T$

$L = I_1' I_0 T$

Input Logic : Truth Table

BR	Input			MUX 1		Load SBR
	I1	I0	T	S1	S0	L
0 0	0	0	0	0	0	0
0 0	0	0	1	0	1	0
0 1	0	1	0	0	0	0
0 1	0	1	1	0	1	1
1 0	1	0	X	1	0	0
1 1	1	1	X	1	1	0

Note that the incrementer circuit in the sequencer of Fig. 3.4 is not a counter constructed with flip-flops but rather a combinational circuit constructed with gates. A combinational circuit incrementer can be designed by cascading a series of half-adder circuits. The output carry from one stage must be applied to the input of the next stage. One input in the first least significant stage must be equal to 1 to provide the increment-by-one operation.

Micro operation

To design a control unit, however, we need to break down the instruction cycle further. In fact, it can be observed that each of the smaller cycles involves a series of steps, each of which involves the processor registers. These steps are referred to as **micro-operations**. Micro-operations are the functional, or atomic, operations of a processor.

The Fetch cycle

The simple fetch cycle actually consists of three steps and four micro operations. Each micro-operation involves the movement of data into or out of a register. So long as these movements do not interfere with one another, several of them can take place during one step, saving time. Symbolically, we can write this sequence of events as follows:

t1: MAR ← (PC)

t2: MBR ← Memory

PC ← (PC) + I

t3: IR ← (MBR)

where I is the instruction length.

The Indirect cycle

Once an instruction is fetched, the next step is to fetch source operands. If the instruction specifies an indirect address, then an indirect cycle must precede the execute cycle. The sequences of events are

t1: MAR ← (IR(Address))

t2: MBR ← Memory

t3: IR(Address) ← (MBR(Address))

The Interrupt cycle

At the completion of the execute cycle, a test is made to determine whether any enabled interrupts have occurred. If so, the interrupt cycle occurs. The nature of this cycle varies greatly from one machine to another. A simple sequence of events is presented below:

t1: MBR ← (PC)

t2: MAR ← Save_Address

PC ← Routine_Address

t3: Memory ← (MBR)

The Execute cycle

Because of the variety opcodes, there are a number of different sequences of micro-operations that can occur. The following instruction is considered

ADD R1, X

The following sequence of micro-operations might occur for the above instruction:

t1: MAR ← (IR(address))

t2: MBR ← Memory

t3: R1 ← (R1) + (MBR)

Let's check the take away from this lecture

1) A microprogram sequencer

a) generates the address of next micro instruction to be executed.

b) generates the control signals to execute a microinstruction.

c) sequentially averages all microinstructions in the control memory.

d) enables the efficient handling of a micro program subroutine.

2) The operation executed on data stored in registers is called

a) Macro-operation

b) Micro-operation

c) Bit-operation

d) Byte-operation

3) Sequencer is another name of?

a) Control address register

b) Control data register

c) Next address generator

d) Control memory

Exercise
Q.1 What is a micro operation?
Q.2 Explain the working of microprogram sequencer with diagram.
Q.3 Write the micro operations for fetch sequence.

Learning from this lecture: Learners will be able to understand the working of microprogram sequencer and micro operations.

Lecture 4

Micro instruction Format, Control Memory

Micro instruction Format

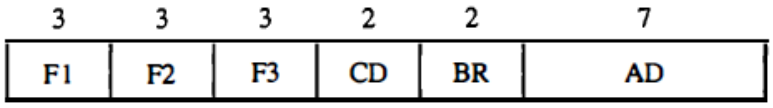
The microinstruction format for the control memory is shown in figure 3.5. The 20 bits of the microinstruction are divided into four functional parts as follows:

1. The three fields F1, F2, and F3 specify micro-operations for the computer. The micro-operations are subdivided into three fields of three bits each. The three bits in each field are encoded to specify seven distinct micro-operations. This gives a total of 21 micro-operations.

2. The CD field selects status bit conditions.

3. The BR field specifies the type of branch to be used.

4. The AD field contains a branch address. The address field is seven bits wide, since the control memory has 128 = 27 words.



F1, F2, F3: Microoperation fields

CD: Condition for branching

BR: Branch field

AD: Address field

Figure 3.5: Microinstruction code format (20 bits)

As an example, a microinstruction can specify two simultaneous micro-operations from F2 and F3 and none from F1.

DR ← M [AR] with F2 = 100

and PC ← PC + 1 with F3 = 101

The nine bits of the micro-operation fields will then be 000 100 101. The CD (condition) field consists of two bits which are encoded to specify four status bit conditions as listed in the Table 3.1.

Symbols and Binary Code for Microinstruction 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	$AC \leftarrow \overline{AC}$	COM
011	$AC \leftarrow$ shl AC	SHL
100	$AC \leftarrow$ shr AC	SHR
101	$PC \leftarrow PC + 1$	INCPC
110	$PC \leftarrow AR$	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

Table 3.1: Condition Field

The BR (branch) field consists of two bits. It is used, in conjunction with the address field AD, to choose the address of the next microinstruction shown in Table 3.2.

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

Table 3.2: Branch Field

Control Memory

The control unit in a digital computer initiates sequences of micro-operations. The complexity of the digital system is derived from the number of sequences that are performed. When the control signals are generated by hardware, it is hardwired. In a bus-oriented system, the control signals that specify micro-operations are groups of bits that select the paths in multiplexers, decoders, and ALUs. The control unit initiates a series of sequential steps of micro-operations. The control variables can be represented by a string of 1's and 0's called a control word. A micro-programmed control unit is a control unit whose binary control variables are stored in memory. A sequence of microinstructions constitutes a microprogram. The control memory can be a read-only memory. Dynamic microprogramming permits a microprogram to be loaded and uses a writable control memory. A computer with a micro-programmed control unit will have two separate memories: a main memory and a control memory. The micro-program consists of microinstructions that specify various internal control signals for execution of register micro-operations. These microinstructions generate the micro-operations to:

- fetch the instruction from main memory
- evaluate the effective address
- execute the operation
- return control to the fetch phase for the next instruction

The control memory address register specifies the address of the microinstruction. The control data register holds the microinstruction read from memory. The microinstruction contains a control word that specifies one or more micro-operations for the data processor. The location for the next microinstruction may, or may not be the next in sequence. Some bits of the present microinstruction control the generation of the address of the next microinstruction. The next address may also be a function of external input conditions. While the micro-operations are being executed, the next address is computed in the next address generator circuit (sequencer) and then transferred into the CAR to read the next microinstructions Typical functions of a sequencer are:

- incrementing the CAR by one
- loading into the CAR and address from control memory
- transferring an external address
- loading an initial address to start the control operations

A clock is applied to the CAR and the control word and next-address information are taken directly from the control memory. The address value is the input for the ROM and the control work is the output. No read signal is required for the ROM as in a RAM. The main advantage of the microprogrammed control is that once the hardware configuration is established, there should be no need for hardware or wiring changes. To establish a different control sequence, specify a different set of microinstructions for control memory.

Let's check the take away from this lecture

1) A group of bits that tell the computer to perform a specific operation is known as____.

a) Instruction code

b) Micro-operation

c) Accumulator

d) Register

2) Microinstructions are stored in control memory in groups, with each group specifying _____.

a) microprogram

b) mapping

c) routine

d) none

3) The transformation from the instruction code bits to an address in control memory where the routine is located is referred to as _____.

a) mapping

b) routine

c) scaling

d) converting

Exercise
Q.1 Explain the fields in a microinstruction format.
Q.2 What is microinstruction?
Q.3 Describe about the control memory.

Learning from this lecture: Learners will be able to understand the microinstruction format and control memory.

Lecture 5

Concepts of nano programming

In microprogrammed processors, an instruction fetched from memory is interpreted by a micro program stored in a single control memory CM; whereas in other microprogrammed processors, the micro instructions are not directly used by the decoder to generate control signals.

This is achieved by the use of a second control memory called a Nano control memory (nCM).

So now there are two levels of control memories, a higher level control memory is known as micro control memory (μCM) and a lower level control memory is known as Nano control memory (nCM). This is shown in Figure 3.6.

Thus a microinstruction is in primary control-store memory, it then has the control signals generated for each microinstruction using a secondary control store memory The output word from the secondary memory is called Nano instruction.

The μCM stores micro instructions whereas nCM stores nano instructions. The decoder uses Nano instructions from nCM to generate control signals. Thus Nano programming gives an alternative strategy to generate control signals. The process of generation of control signals using nano instructions is shown in Figure 3.6.

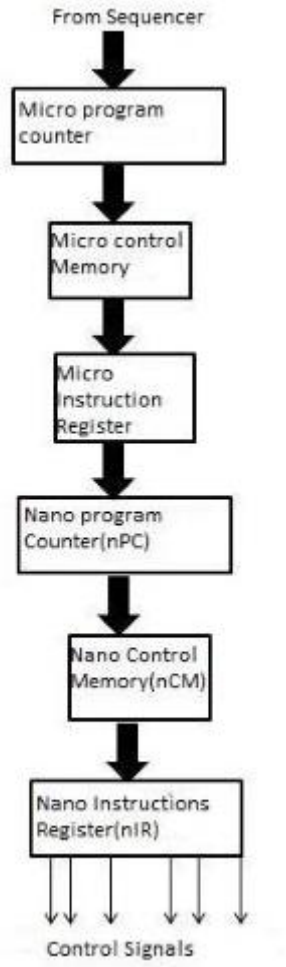


Figure 3.6: Nanoprogramming

Nano instruction addresses are generated by a nano program counter and nano instructions are placed in a register nIR. The next address of nIR is directly obtained. The next address is generated by either incrementing the nano program counter or loading it from external source (branch field or address from micro instruction opcode)

Advantages of Nano programming:

1. Reduces total size of required control memory

In two level control design technique, the total control memory size S2 can be calculated as

S2=Hm×Wm+Hn×Wn;

where Hm represents the number of words in the high level memory

Wm represents the size of word in the high level memory

Hn represents the number of words in the low level memory

Wn represents the size of word in the low level memory

Usually, the micro programs are vertically organized so Hm is large and Wm is small. In Nano programming, we have a highly parallel horizontal organization, which makes Wn large and Hn is small. This gives the compatible size for single level control unit as

S1=Hm×Wn which is larger than S2. The reduced size of control memory reduces the total chip area.

2. Greater design flexibility

Because of two level memories organization more design flexibility exists between instructions and hardware.

Disadvantage of Nano programming

1. Increased memory access time:

The main disadvantage of the two level memory approaches is the loss of speed due to the extra memory access required for Nano control memory.

Let's check the take away from this lecture

1) Which of the following is not part of microinstruction code format?

a) SB

b) BR

c) CD

d) AD

2) Nano instruction addresses are generated by ----------

a) micro program counter

b) nano program counter

c) instruction program counter

d) step counter

3) In general, control memory can be?

a) ROM

b) RAM

c) Both A & B

d) None

Exercise

Q.1 Define nanoprogramming.

Q.2 Explain the process of generation of control signals using nano instructions.

Q.3 State the advantages of nanoprogramming.

Learning from this lecture: Learners will be able to understand the concept of nanoprogramming.

Lecture 6

Introduction to RISC and CISC architectures and design issues

CISC

Pronounced *sisk,* and stands for **C**omplex **I**nstruction **S**et **C**omputer. Most PC's use CPU based on this architecture. For instance Intel and AMD CPU's are based on CISC architectures. Typically CISC chips have a large amount of different and complex instructions. The philosophy behind it is that hardware is always faster than software, therefore one should make a powerful instruction set, which provides programmers with assembly instructions to do a lot with short programs. In common CISC chips are relatively slow (compared to RISC chips) per instruction, but use little (less than RISC) instructions.

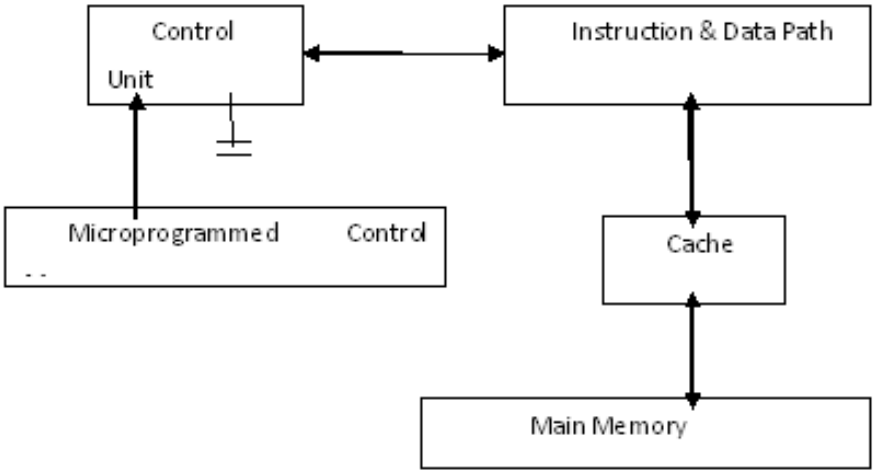


Figure 3.7: CISC Architecture

RISC

Pronounced *risk,* and stands for **R**educed **I**nstruction **S**et **C**omputer. RISC chips evolved around the mid-1980 as a reaction at CISC chips. The philosophy behind it is that almost no one uses complex assembly language instructions as used by CISC, and people mostly use compilers which never use complex instructions. Apple for instance uses RISC chips. Therefore fewer, simpler and faster instructions would be better, than the large, complex and slower CISC instructions. However, more instructions are needed to accomplish a task. Another advantage of RISC is that - in theory - because of the more simple instructions,

RISC chips require fewer transistors, which makes them easier to design and cheaper to produce. Finally, it's easier to write powerful optimized compilers, since fewer instructions exist.

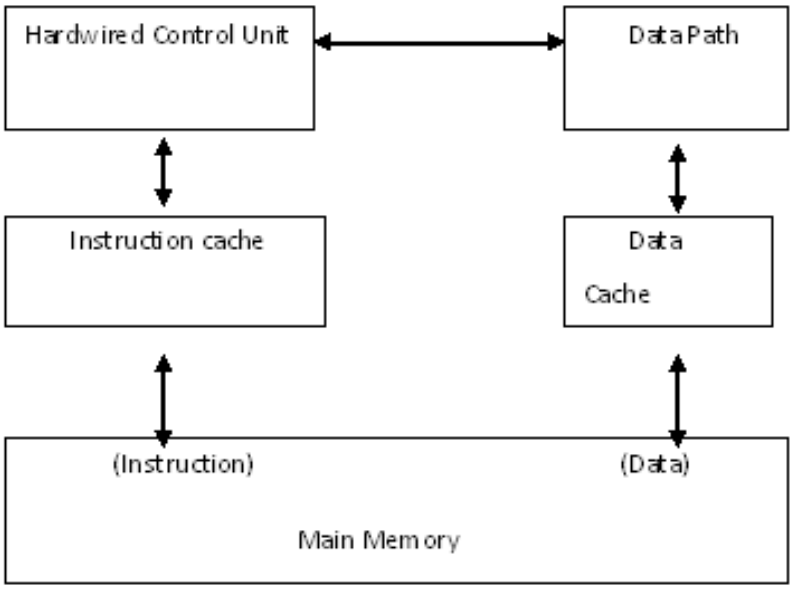


Figure 3.8: RISC Architecture

RISC vs CISC

There is still considerable controversy among experts about which architecture is better. Some say that RISC is cheaper and faster and therefore the architecture of the future. Others note that by making the hardware simpler, RISC puts a greater burden on the software. Software needs to become more complex. Software developers need to write more lines for the same tasks. Therefore they argue that RISC is not the architecture of the future, since conventional CISC chips are becoming faster and cheaper anyway. RISC has now existed more than 10 years and hasn't been able to kick CISC out of the market. If we forget about the embedded market and mainly look at the market for PC's, workstations and servers I guess a least 75% of the processors are based on the CISC architecture. Most of them the x86 standard (Intel, AMD, etc.), but even in the mainframe territory CISC is dominant via the IBM/390 chip. RISC and CISC architectures are becoming more and more alike. Many of today's RISC chips support just as many instructions as yesterday's CISC chips. The PowerPC 601, for example, supports *more* instructions than the Pentium. Yet the 601 is considered a RISC chip, while the Pentium is definitely CISC. Furthermore today's CISC chips use many techniques formerly associated with RISC chips.

Let's check the take away from this lecture

1) The computer architecture aimed at reducing the time of execution of instructions is _____

a) CISC

b) RISC

c) ISA

d) ANNA

2) In CISC architecture most of the complex instructions are stored in _____

a) Register

b) Diodes

c) CMOS

d) Transistors

3) Which of the architecture is power efficient?

a) CISC

b) RISC

c) ISA

d) IANA

Exercise
Q.1 Explain CISC architecture.
Q.2 Explain RISC architecture.
Q.3 Compare RISC and CISC architectures.

Learning from this lecture: Learners will be able to understand the features of RISC and CISC architectures.

Conclusion

The study of Control Unit design helps to understand the importance of control unit in the central processing unit of a computer. By knowing the details of hardwired and microprogrammed control unit, the student will be able to identify the microinstructions for the sequence of operations performed by the processor.

Short Answer Questions:

1. What are the basic operations performed by processor?

Ans: -Fetch instruction

-Interpret instruction

-Fetch data

-Process data

-Write data

2. What is hardwired control unit?

Ans: The control hardware can be viewed as a state machine that changes from one state to another in every clock cycle, depending on the contents of the instruction register, the condition codes and the external inputs. The outputs of the state machine are the control signals. The sequence of the operation

carried out by this machine is determined by the wiring of the logic elements and hence named as "hardwired".

3. What is micro programming and microprogrammed control unit?

Ans: The micro-operations are described in symbolic notations which looks like a programming language and that is known as micro programming language.

When the control unit is implemented using this micro programming language it is known as microprogrammed control unit.

4. What are the basic tasks performed by a micro-programmed control unit?

Ans: i) Microinstruction sequencing: Get the next instruction from the control memory

ii) Microinstruction execution: Generate the control signals needed to execute the microinstruction

5. What is Microinstruction sequencing ?

Ans: The basic components of microprogrammed control unit are the control memory and the circuits that select the next address. The task of Microinstruction sequencing is done by Microprogram sequencer. The address selection part is called as microprogram sequencer. Microprogram sequencer can be constructed with digital functions to suit a particular application.

6. Define Control memory.

Ans: A control unit whose binary control variables are stored in memory is called a microprogrammed control unit. A memory that is part of a control unit is referred to as a control memory. Each word in control memory contains within it a microinstruction.

Long Answer Questions:

1. Explain hardwired control unit with block diagram.

Ans: Refer the section 'Hardwired control unit' under lecture 1

2. Explain microprogrammed control unit.

Ans: Refer the section 'Microprogrammed control unit' under lecture 1

3. Explain the process of address sequencing with diagram.

Ans: Refer the contents under lecture 2

4. Explain Microprogram Sequencer.

Ans: Refer the section 'Microprogram Sequencer' under lecture 3

5. Write the micro operations for fetch stage of an instruction execution.

Ans: Refer the section 'Micro-operation' under lecture 3

6. Describe the different fields in a microinstruction

Ans. Refer the section 'Microinstruction Format' under lecture 4

7. Explain the significance of control memory in a control unit.

Ans. Refer the section 'Control Memory' under lecture 4

8. Explain the concept of Nano programming.

Ans. Refer the contents under lecture 5

9. Compare RISC and CISC architectures.

Ans. Refer the contents under lecture 6

Set of Questions for FA/IA/ESE

1. Explain hardwired control unit with block diagram. (10M)
2. Define the following terms microinstruction, micro program & micro operation. (5M)
3. State the disadvantages of hardwired control unit. (5 M)
4. Explain microprogrammed control unit in detail. (10 M)
5. Distinguish hardwired and microprogrammed control unit (5 M)
6. Write short note on Microprogram sequencer. (10 M)
7. Write down the microinstruction format and define the different fields (6 M)
8. Explain the role of control memory in microprogrammed control unit. (8 M)
9. Explain the process of address sequencing with diagram. (10 M)
10. Explain the concept of Nano programming. (6 M)
11. Compare RISC and CISC architectures. (8 M)

References:

1.William Stallings, “Computer Organization and Architecture: Designing for Performance”, Eighth Edition, Pearson.

2. Carl Hamacher, Zvonko Vrasenic and Safwat Zaky, “Computer Organization”, Fifth edition, Tata McGraw-Hill.

3. Morris Mano, “Computer Architecture and Organization”, Third Edition, McGraw Hill.

Self-assessment

Q.1) What is the role of control unit in computer?

Q.2) Distinguish between hardwired and microprogrammed control unit.

Q.3) Explain the concept of nano programming.

Q.4) Explain the features of RISC and CISC architecture and compare them.

Self-evaluation

Name of Student		
Class		
Roll No.		
Subject		
Module No.		
S.No		Tick Your choice
1.	Do you understand the the role of control unit in a computer?	o Yes o No
2.	Do you understand the difference between hardwired and micro programmed control unit?	o Yes o No
3.	Do you understand the microinstruction format?	o Yes o No
4.	Do you understand the concept of nano programming?	o Yes o No
5.	Do you understand the features of RISC and CISC architectures?	o Yes o No
6.	Do you understand module 3 ?	o Yes, Completely. o Partialy. o No, Not at all.