Computer Architecture ECS-355

Mr. Abdul Rehman **email id: abdul@northern.edu.pk** **Whatsapp# 0308-7792217**

(Week 7) Lecture 13-14

Learning objectives:

- Review of the last lecture
- Understand the architecture of a very important computer system component i.e. Control Unit
- Types of control Units
- Understand that how a microprogrammed control unit works?
- Identification and understanding of the working of the different components of microprogrammed control unit.
- Knowledgeability of Address Sequencing
- Understanding of instruction mapping

(Note: It is important to note that we are considering "Basic computer architecture" processor which we were discussing in the class and it's the continuation of the same concepts. Another important point is that brief part of the last lecture is repeated. Audio clip will be sent separately for every diagram.)

Resources: Beside these lecture handouts, this lesson will draw from the following
Text Book: Computer System Architecture by Morris Mano (3rd Edition) and
Reference book:Computer Architecture, by William Stallings (4th Edition).

Lecture:

Terminology

Microprogram

- Program stored in memory that generates all the control signals required to execute the instruction set correctly
- Consists of microinstructions

Microinstruction

- Contains a control word and a sequencing word
    - Control Word - All the control information required for one clock cycle

Mr. Abdul Rehman **email id: abdul@northern.edu.pk** **Whatsapp# 0308-7792217**

    - Sequencing Word - Information needed to decide the next microinstruction address

- Vocabulary to write a microprogram

Control Memory (Control Storage: CS)

- Storage in the microprogrammed control unit to store the microprogram

Writeable Control Memory (Writeable Control Storage: WCS)

- CS whose contents can be modified
    - Allows the microprogram can be changed
    - Instruction set can be changed or modified

Dynamic Microprogramming

- Computer system whose control unit is implemented with a microprogram in WCS
- Microprogram can be changed by a systems programmer or a user

Sequencer (Microprogram Sequencer)

A Microprogram Control Unit that determines the Microinstruction Address to be executed in the next clock cycle

- In-line Sequencing
- Branch
- Conditional Branch
- Subroutine
- Loop
- Instruction OP-code mapping

Control Unit

The function of the control unit in a digital computer is to initiate sequences of microoperations. The number of different types of microoperations that are available in a given system is finite

Mr. Abdul Rehman **email id: abdul@northern.edu.pk** **Whatsapp# 0308-7792217**

(fixed). The complexity of the digital system is derived from the number of sequences of microoperations that are performed. When the control signals are generated by hardware using conventional logic design techniques, the control unit is said to be hardwired. Microprogramming is a second alternative for designing the control unit of a digital computer. The principle of microprogramming is an elegant and systematic method for con· trolling the microoperation sequences in a digital computer.

A computer that employs a microprogrammed control unit will have two separate memories: a main memory and a control memory. The main memory is available to the user for storing the programs. The contents of main memory may alter when the data are manipulated and every time that the program is changed. The user's program in main memory consists of machine instructions and data. In contrast, the control memory holds a fixed microprogram that cannot be altered by the occasional user. The microprogram consists of microinstructions that specify various internal control signals for execution of register microoperations. Each machine instruction initiates a series of microinstructions in control memory. These microinstructions generate the microoperations to fetch the instruction from main memory; to evaluate the effective address, to execute the operation specified by the instruction, and to return control to the fetch phase in order to repeat the cycle for the next instruction.

The general configuration of a microprogrammed control unit is demonstrated in the block diagram of Fig. 1. The control memory is assumed to be a ROM, within which all control information is permanently stored. 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 microoperations 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. For this reason it is necessary to use some bits of the present microinstruction to control the generation of the address of the next microinstruction. The next address may also be a function of external input conditions. While the microoperations 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

Mr. Abdul Rehman **email id: abdul@northern.edu.pk** **Whatsapp# 0308-7792217**

microinstruction contains bits for initiating microoperations in the data processor part and bits that determine the address sequence for the control memory.

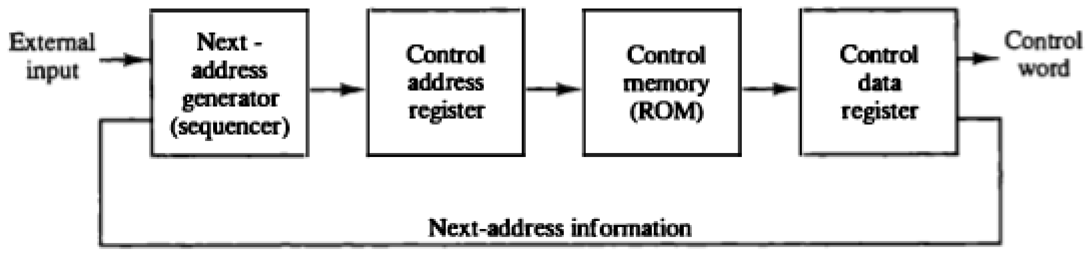


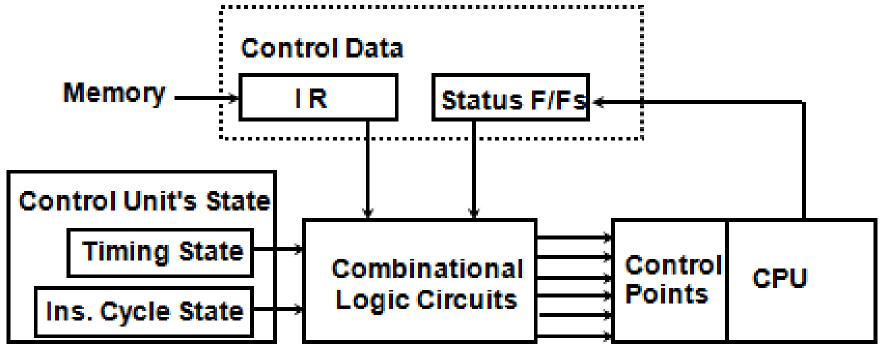
Figure 1: Microprogrammed control organization.

Comparison of Control Unit Implementation

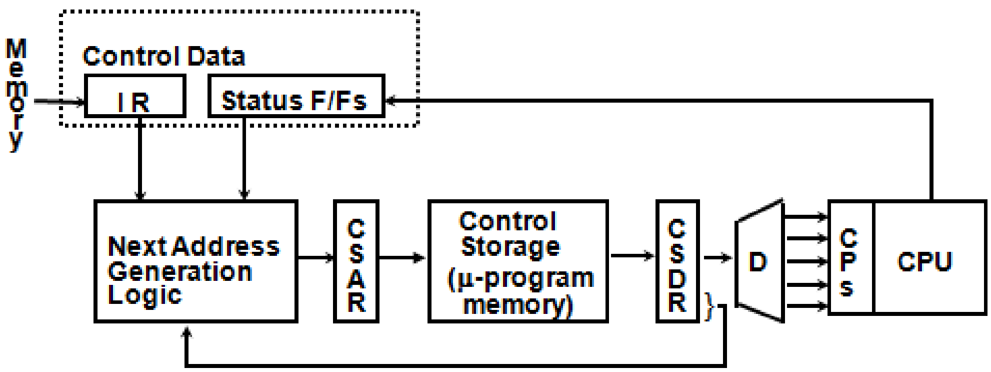
Mr. Abdul Rehman **email id: abdul@northern.edu.pk** **Whatsapp# 0308-7792217**

Control Unit Implementation

Combinational Logic Circuits (Hard-wired)



Microprogram



Microinstruction sequencing

Fig 2 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 microoperations in computer registers and other bits to specify the method by which the next address is obtained. The diagram shows four different paths from which the control address register (CAR) receives the address. The incrementer increments the content of the control 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 whose value is then used when the microprogram wishes to return from the subroutine.

Mr. Abdul Rehman **email id: abdul@northern.edu.pk** **Whatsapp# 0308-7792217**

Sequencing Capabilities Required in a Control Storage.

- Incrementing of the control address register
- Unconditional and conditional branches
- A mapping process from the bits of the machine instruction to an address for control memory
- A facility for subroutine call and return

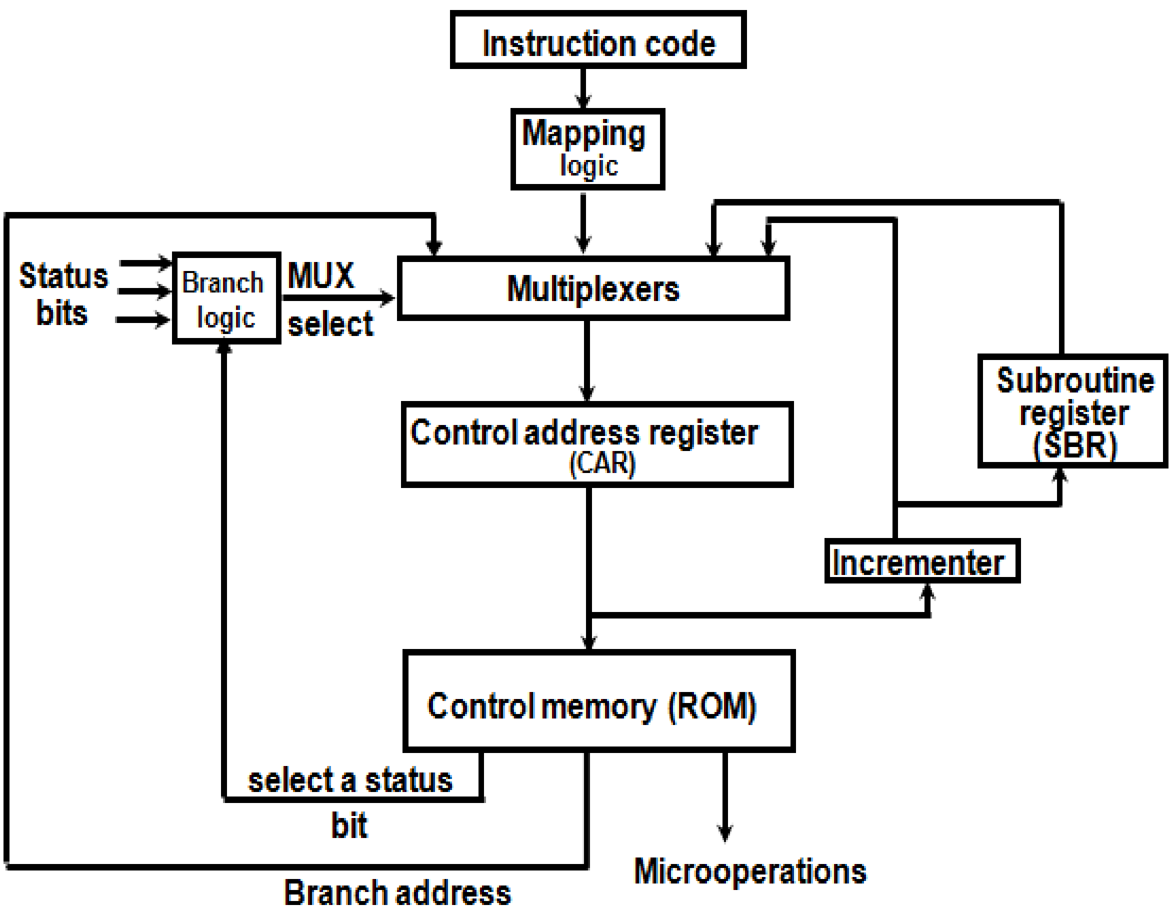


Figure 2: Selection of address of control memory.

Instruction Mapping

Microinstructions are stored in control memory in groups, with each group specifying a routine. Each computer instruction has its own microprogram routine in control memory to generate the

Mr. Abdul Rehman **email id: abdul@northern.edu.pk** **Whatsapp# 0308-7792217**

microoperations that execute the instruction. 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. To appreciate the address sequencing in a microprogram control unit, let us enumerate the steps that the control must undergo during the execution of a single computer instruction. An initial address is loaded into the control address register when power is turned on in the computer. This address is usually the address of the first microinstruction that activates the instruction fetch routine. The fetch routine may be sequenced by incrementing the control address register through the rest of its microinstructions. At the end of the fetch routine, the instruction is in the instruction register of the computer. This includes the mapping of machine instructions with micro-instructions stored in control memory.

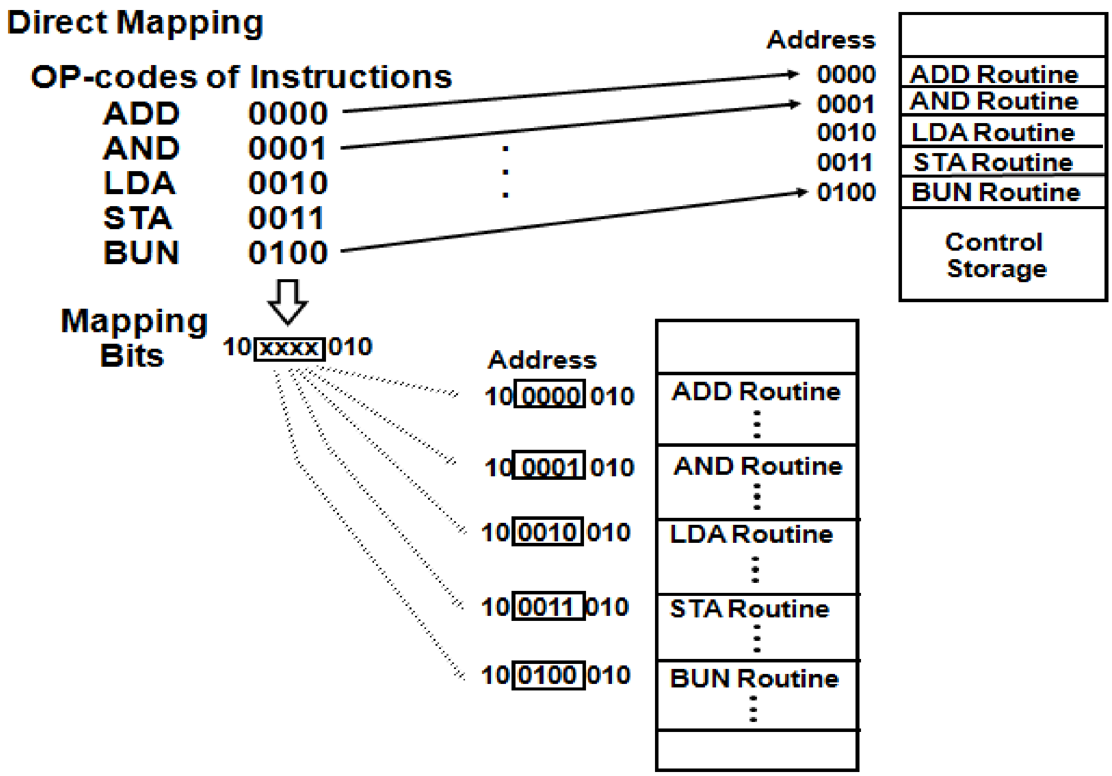


Figure 3: Control memory subroutine.

Mapping to Micro-routine

A special type of branch exists when a microinstruction specifies a branch to the first word in control memory where a microprogram routine for an instruction is located. The status bits for

Mr. Abdul Rehman **email id: abdul@northern.edu.pk** **Whatsapp# 0308-7792217**

this type of branch are the bits in the operation code part of the instruction. For example, a computer with a simple instruction format as shown in Fig. 4 has an operation code of four bits which can specify up to 16 distinct instructions. Assume further that the control memory has 128 words, requiring an address of seven bits. For each operation code there exists a microprogram routine in control memory that executes the instruction. One simple mapping process that converts the 4-bit operation code to a 7-bit address for control memory is shown in Fig. 4. This 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.

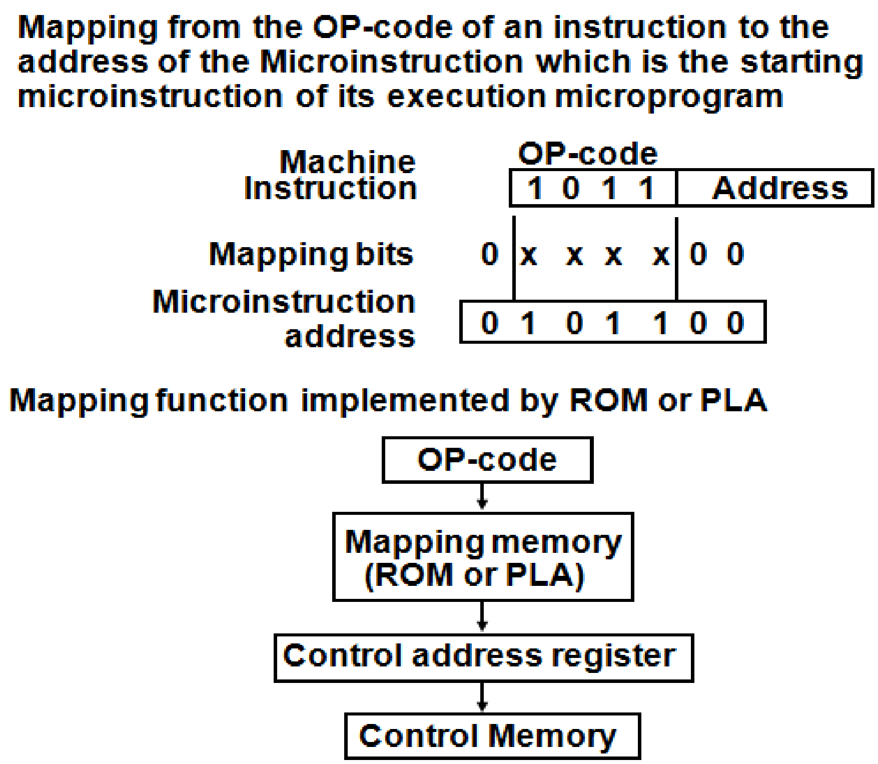


Figure 4: Mapping procedure.

Subroutines

A set of common instructions that can be used in a program many times is called a subroutine, Each time that a subroutine is used in the main part of the program, a branch is executed to the

Mr. Abdul Rehman **email id: abdul@northern.edu.pk** **Whatsapp# 0308-7792217**

beginning of the- subroutine. After the subroutine has been executed, a branch is made back to the main program.

A subroutine is a self-contained sequence of instructions that performs a specific task. However, since various parts of the main program can call the same subroutine, it's crucial for the subroutine to know where to return after execution. This requires storing the return address in the computer. To simplify this process, all computers include special instructions for entering and exiting subroutines.