University of Babylon, College of science for women
Dept. of Computer science

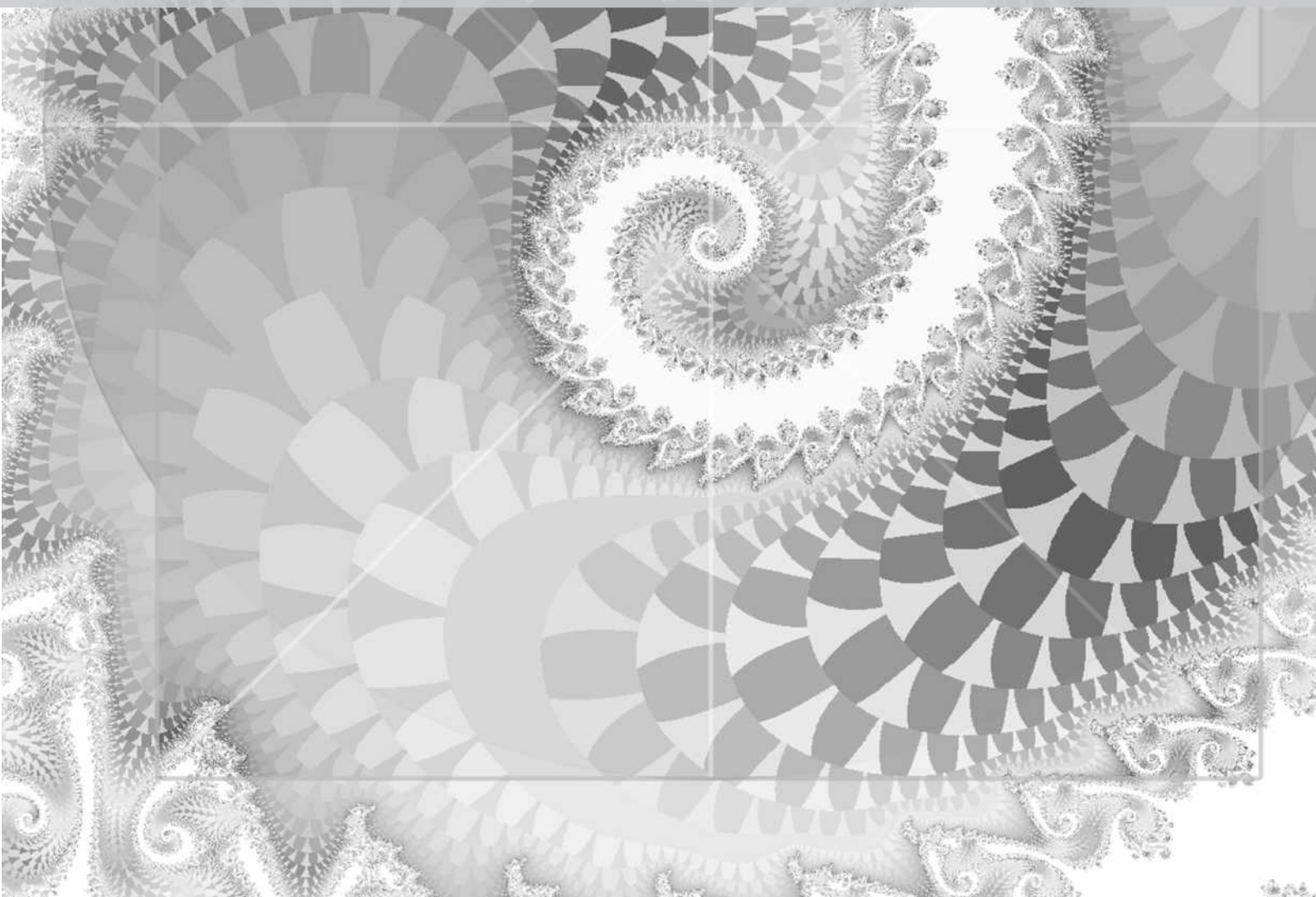
Computer Architecture

Second year

Dr. Salah Al-Obaidi

Lecture #8: The Control Unit

Spring 2024



Contents

9. Design of the control unit

The control unit is one of the major components of the processor. Control unit provides control signals for the operation and coordination of all processor components. Traditionally, a microprogramming implementation has been used, in which major components are control memory, microinstruction sequencing logic, and registers. More recently, microprogramming has been less prominent but remains an important implementation technique. The control unit manages the computer's resources and orchestrates the performance of its functional parts in response to instructions. The control unit controls the movement of data and instructions into and out of the processor and controls the operation of the ALU.

For the control unit to perform its function, it must have inputs that allow it to determine the state of the system and outputs that allow it to control the behavior of the system. These are the external specifications of the control unit. Internally, the control unit must have the logic required to perform its sequencing and execution functions. Figure 9.1 is a general model of the control unit, showing all of its inputs and outputs. The inputs are:

- **Clock:** This is how the control unit "keeps time." The control unit causes one micro-operation (or a set of simultaneous micro-operations) to be performed for each clock pulse. This is sometimes referred to as the processor cycle time, or the clock cycle time.
- **Instruction register:** The opcode and addressing mode of the current instruction are used to determine which micro-operations to perform during the execute cycle.
- **Flags:** These are needed by the control unit to determine the status of the processor

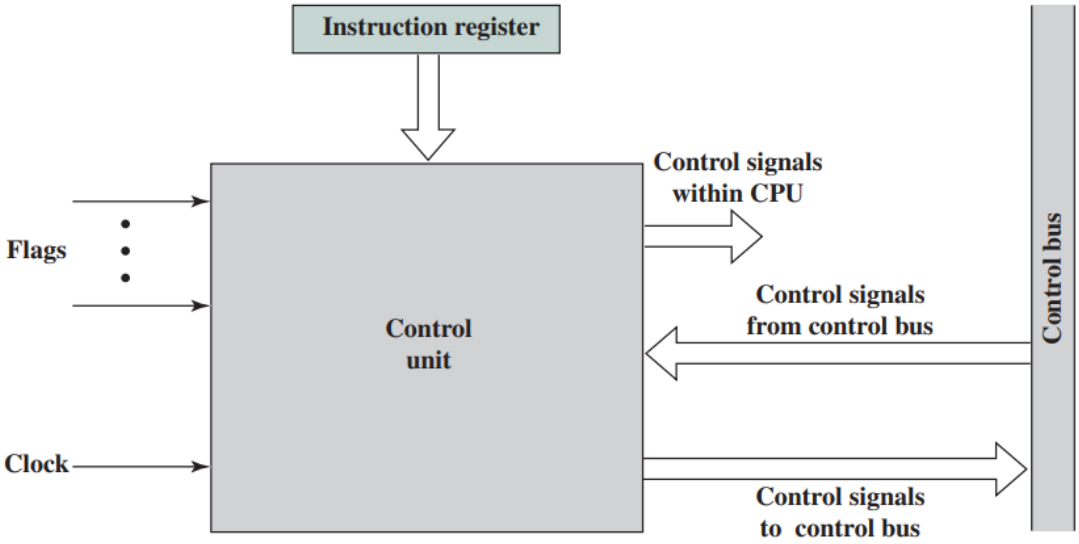


Figure 9.1: Block Diagram of the Control Unit

and the outcome of previous ALU operations. For example, for the increment-and-skip-if-zero (ISZ) instruction, the control unit will increment the PC if the zero flag is set.

- **Control signals from control bus:** The control bus portion of the system bus provides signals to the control unit.

The outputs are as follows:

- **Control signals within the processor:** These are two types: those that cause data to be moved from one register to another, and those that activate specific ALU functions.
- **Control signals to control bus:** These are also of two types: control signals to memory, and control signals to the I/O modules.

Three types of control signals are used:

1. those that activate an ALU function;
2. those that activate a data path;
3. those that are signals on the external system bus or other external interface.

All of these signals are ultimately applied directly as binary inputs to individual logic gates.

A Control Signals Example

To illustrate the functioning of the control unit, let us examine a simple example. Figure 9.2 illustrates the example. This is a simple processor with a single accumulator (AC). The data paths between elements are indicated. The control paths for signals emanating from the control unit are not shown, but the terminations of control signals are labeled C_i and indicated by a circle. The control unit receives inputs from the clock, the IR, and flags. With each clock cycle, the control unit reads all of its inputs and emits a set of control signals. Control signals go to three separate destinations:

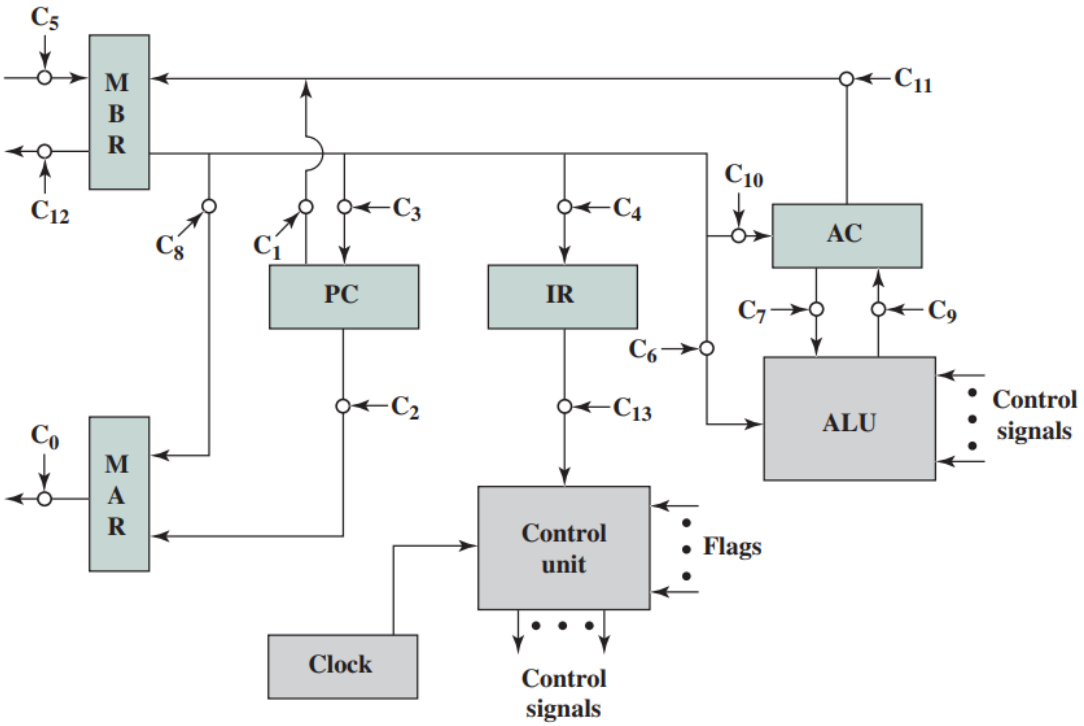


Figure 9.2: Data Paths and Control Signals

- **Data paths:** The control unit controls the internal flow of data. For example, on instruction fetch, the contents of the memory buffer register are transferred to the

IR. For each path to be controlled, there is a switch (indicated by a circle in the figure). A control signal from the control unit temporarily opens the gate to let data pass.

- **ALU:** The control unit controls the operation of the ALU by a set of control signals. These signals activate various logic circuits and gates within the ALU.
- **System bus:** The control unit sends control signals out onto the control lines of the system bus (e.g., memory READ).

The control unit must maintain knowledge of where it is in the instruction cycle. Using this knowledge, and by reading all of its inputs, the control unit emits a sequence of control signals that causes micro-operations to occur. It uses the clock pulses to time the sequence of events, allowing time between events for signal levels to stabilize.

It is worth pondering the minimal nature of the control unit. The control unit is the engine that runs the entire computer. It does this based only on knowing the instructions to be executed and the nature of the results of arithmetic and logical operations (e.g., positive, overflow, etc.). It never gets to see the data being processed or the actual results produced. And it controls everything with a few control signals to points within the processor and a few control signals to the system bus.

9.1 HARDWIRED IMPLEMENTATION

A wide variety of techniques have been used for control unit implementation. Most of these fall into one of two categories:

- **Hardwired implementation**
- **Microprogrammed implementation**

In a **hardwired implementation**, the control unit is essentially a state machine circuit. Its input logic signals are transformed into a set of output logic signals, which are the control signals.

Control Unit Inputs

The key inputs are the IR, the clock, flags, and control bus signals. In the case of the flags and control bus signals, each individual bit typically has some meaning (e.g., overflow).

First consider the IR. The control unit makes use of the opcode and will perform different actions (issue a different combination of control signals) for different instructions. To simplify the control unit logic, there should be a unique logic input for each opcode. This function can be performed by a ***decoder***, which takes an encoded input and produces a single output. In general, a decoder will have n binary inputs and 2^n binary outputs. Each of the 2^n different input patterns will activate a single unique output.

The clock portion of the control unit issues a repetitive sequence of pulses. This is useful for measuring the duration of micro-operations. Essentially, the period of the clock pulses must be long enough to allow the propagation of signals along data paths and through processor circuitry. However, the control unit emits different control signals at different time units within a single instruction cycle. The control unit can be depicted as in Figure 9.3.

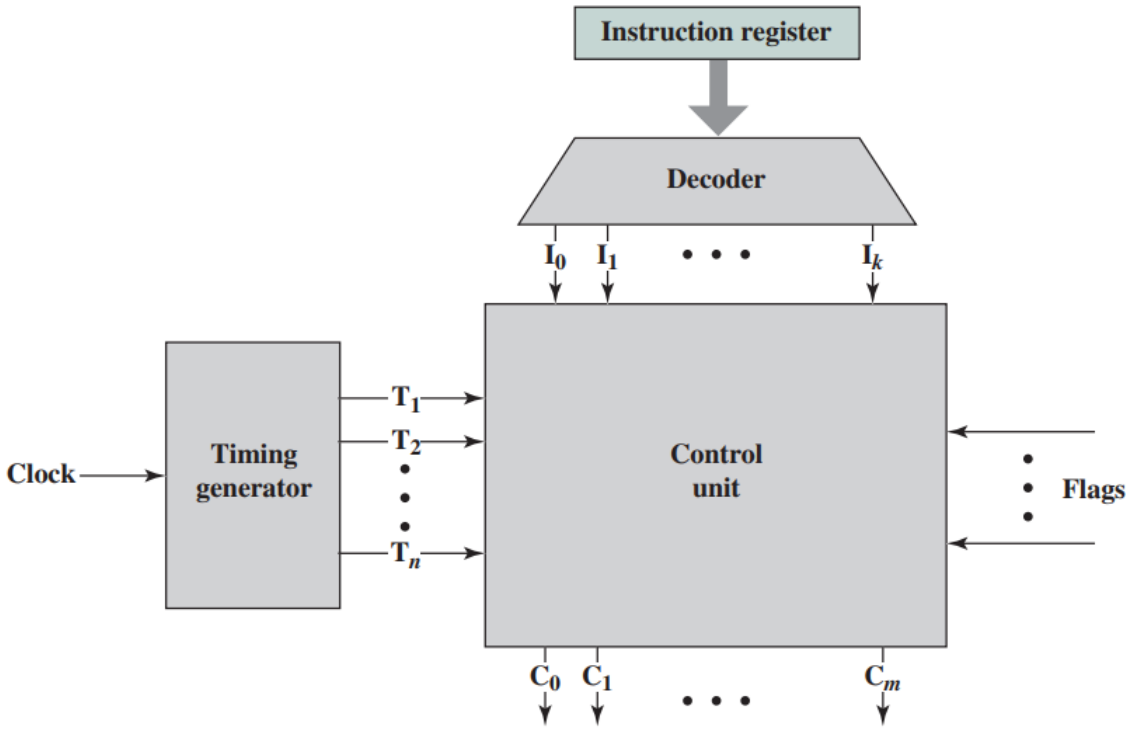


Figure 9.3: Control Unit with Decoded Inputs.

9.2 MICROPROGAMMED CONTROL

The term microprogram was first coined by M. V. Wilkes in the early 1950s. Wilkes proposed an approach to control unit design that was organized and systematic and avoided the complexities of a hardwired implementation. The idea intrigued many researchers but appeared unworkable because it would require a fast, relatively inexpensive control memory. Microprogramming became a popular technique for implementing the control unit of CISC processors. In recent years, microprogramming has become less used but remains a tool available to computer designers. For example, in Pentium 4, machine instructions are converted into a RISC-like format, most of which are executed without the use of microprogramming. However, some of the instructions are executed using microprogramming.

Microinstructions

The control unit seems a reasonably simple device. Nevertheless, to implement a control unit as an interconnection of basic logic elements is no easy task. The design must include logic for sequencing through micro-operations, for executing micro-operations, for interpreting opcodes, and for making decisions based on ALU flags. It is difficult to design and test such a piece of hardware. Furthermore, the design is relatively inflexible. For example, it is difficult to change the design if one wishes to add a new machine instruction.

An alternative, which has been used in many CISC processors, is to implement a **microprogrammed control unit**.

How can we use the concept of microprogramming to implement a control unit?. Consider that for each micro-operation, all that the control unit is allowed to do is generate a set of control signals. Thus, for any micro-operation, each control line emanating from the control unit is either on or off. This condition can, of course, be represented by a binary digit for each control line. So we could construct a ***control word*** in which each bit represents one control line. Then each micro-operation would be represented by a different pattern of 1s and 0s in the control word.

Figure 9.4 shows how these control words or microinstructions could be arranged in a control memory. The microinstructions in each routine are to be executed sequentially. Each routine ends with a branch or jump instruction indicating where to go next. There is a special execute cycle routine whose only purpose is to signify that one of the machine instruction routines (AND, ADD, and so on) is to be executed next, depending on the current opcode

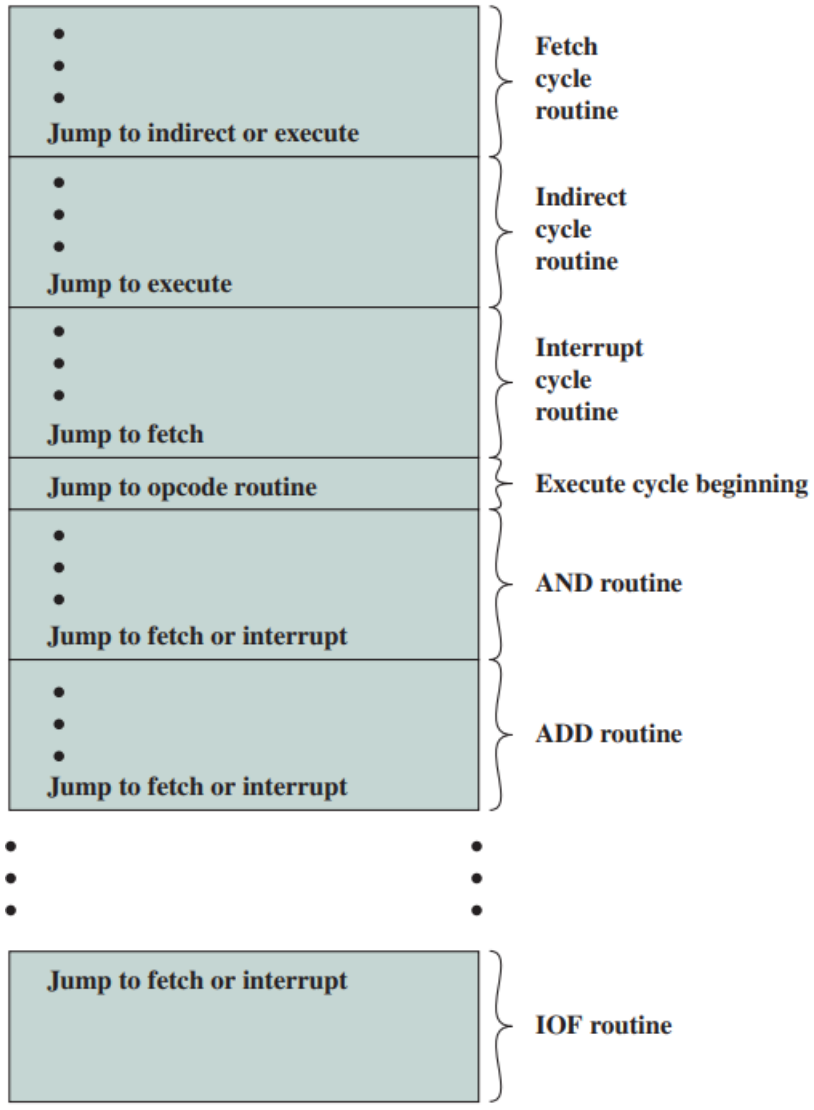


Figure 9.4: Organization of Control Memory.

Microprogrammed Control Unit

The control memory of Figure 9.4 contains a program that describes the behavior of the control unit. It follows that we could implement the control unit by simply executing that program.

Figure 9.5 shows the key elements of such an implementation. The set of microinstructions is stored in the control memory. The ***control address register*** contains the address of the next microinstruction to be read. When a microinstruction is read from the control memory, it is transferred to a ***control buffer register***. The lefthand portion of that register connects to the control lines emanating from the control unit. Thus, reading a microinstruction from the control memory is the same as executing that microinstruction. The third element shown in the figure is a sequencing unit that loads the control address register and issues a read command.

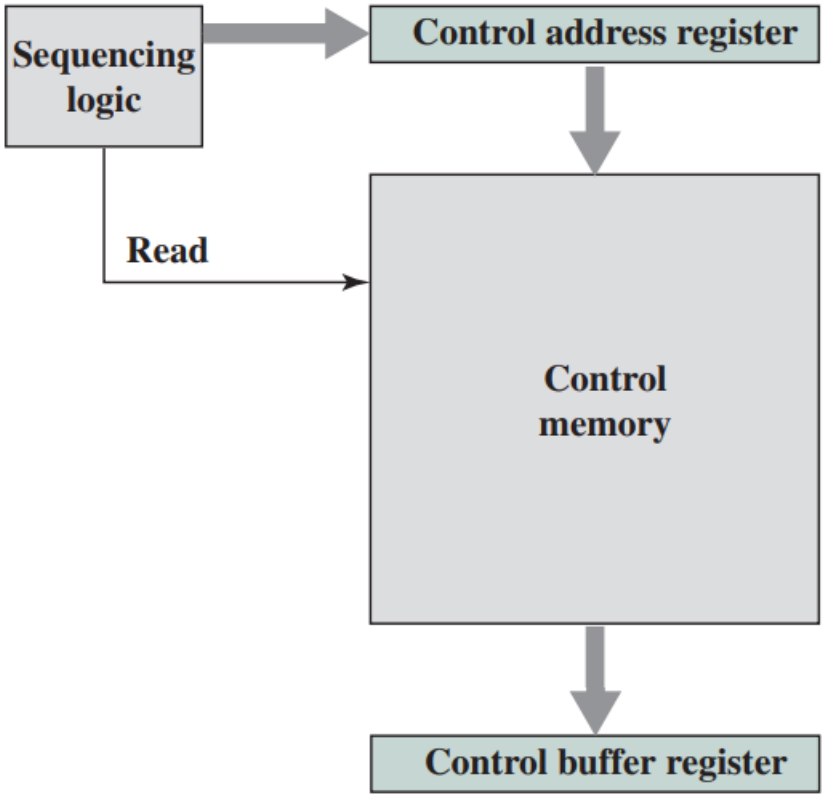


Figure 9.5: Control Unit Microarchitecture.

Let us examine this structure in greater detail, as depicted in Figure 9.6. Comparing this with Figure 9.5, we see that the control unit still has the same inputs (IR, ALU flags,

clock) and outputs (control signals). The control unit functions as follows:

1. To execute an instruction, the sequencing logic unit issues a READ command to the control memory.

2. The word whose address is specified in the control address register is read into the control buffer register.

3. The content of the control buffer register generates control signals and next-address information for the sequencing logic unit.

4. The sequencing logic unit loads a new address into the control address register based on the next-address information from the control buffer register and the ALU flags.

All this happens during one clock pulse.

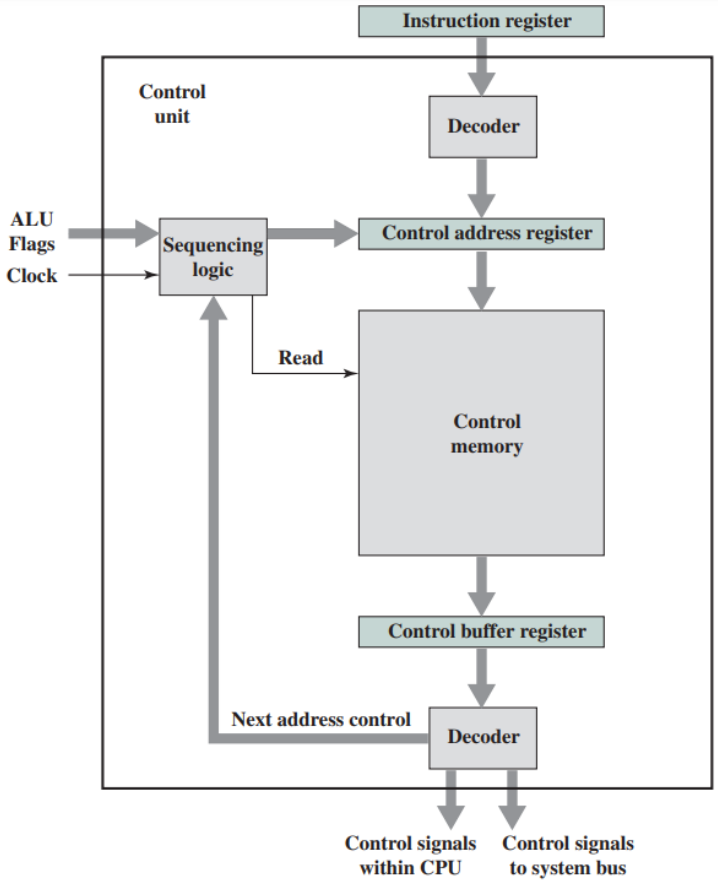


Figure 9.6: Functioning of Microprogrammed Control Unit.

Advantages and Disadvantages

The principal advantage of the use of microprogramming to implement a control unit is that it simplifies the design of the control unit. Thus, it is both cheaper and less error prone to implement. A hardwired control unit must contain complex logic for sequencing through the many micro-operations of the instruction cycle. On the other hand, the decoders and sequencing logic unit of a microprogrammed control unit are very simple pieces of logic.

The principal disadvantage of a microprogrammed unit is that it will be somewhat slower than a hardwired unit of comparable technology. Despite this, microprogramming is the dominant technique for implementing control units in pure CISC architectures, due to its ease of implementation. RISC processors, with their simpler instruction format, typically use hardwired control units.