Computer Architecture & Organization

BCA 2^{nd} Sem Study Material
Paper: C4T

Control Unit

Anupam Pattanayak[1]
Assistant Professor,
Department of Computer Science,
Raja N. L. Khan Women's College (Autonomous),
Midnapore, West Bengal

April 9, 2020

[1] anupam.pk@gmail.com

Contents

1

Control Unit

In class, so far, we have seen signed number representation, von Neumann architecture and Harvard architecture, instruction execution cycle, addressing modes - register addressing, immediate addressing, direct addressing, and indirect addressing mode, instruction length, 3-address instructions, 2-address, instructions, 1-address instructions, 0-address instructions, CPU registers - PC, IR (IR or instruction register is part of control unit) , AC, SP, CISC and RISC architectures, design of simple 1-bit ALU and 2-bit ALU by bit-slice design paradigm. So these are mainly from the CPU and ALU. Here, we will have discussion on *control unit.*

1.1 Revisit Block Diagram of Computer

Let us revisit the block diagram of a computer system again. Following figure 1.1 shows a simple block diagram of a computer system:

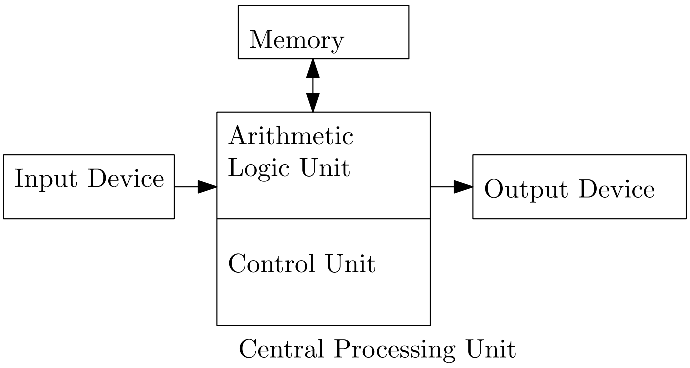

Figure 1.1: Block Diagram of a Computer System

Control unit (CU) is like nervous system of human. In our body, nervous system is responsible for receiving and transmitting neuro signals between different components of our body. Similarly, control unit is responsible for generation, transmission, and reception of control signals of a computer system. Just like CPU controls the different building blocks a computer system, CU controls the CPU.

we will refer the books by Hayes[1], by Sajjan Shiva[3], and by Dumas[4].

Remember that, we mentioned the fact that *ALU is a combinational circuit.* In contrast, *control unit is sequential circuit.* This concept is very much foundational when we attempt to design a simple control unit using hardware description language packages such as VHDL or Verilog.

1.2 Control Unit

We will now slightly modify the basic block diagram of computer that we have seen in figure 1.1. Following figure 1.2 shows a simple block diagram of a computer system:

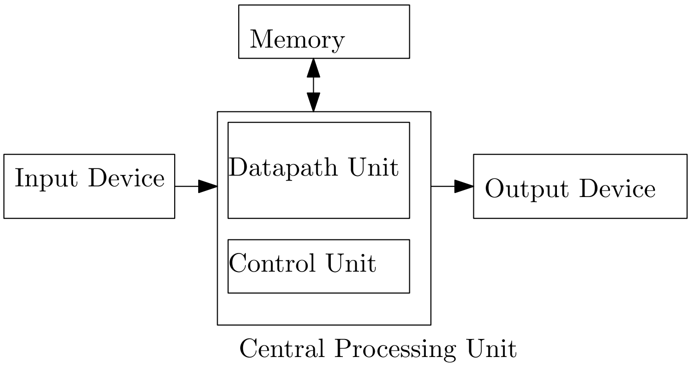

Figure 1.2: Revised Block Diagram of a Computer System

In the relevant literature, we often find the statement like *CPU consists of datapath and control unit.* Datapath is the portion that is capable of performing microoperations on data words. Control unit issues control signals to the datapath.

The function of CU is to generate appropriate control signals to fecilitate every stages of instruction execution cycle. That is, it fetches the instruction

[1] Computer Architecture and Organization by J. P. Hayes, by Mano[2], McGraw Hill

[3] computer Organization Design and Architecture by Sajjan G Shiva, CRC Press

[4] Computer Architecture by Joseph Dumas, CRC Press

opcode from memory and place the opcode into instruction register (IR), decode the content of IR to find out what are to be done, determine the number of operands required and source of these operands, extract and place the operands to the ALU, generate appropriate signals for carrying out the ALU operation, copy the result of operation to the appropriate destination. To accomplish all these, CU sends all the necessary control signals at the appropriate timings so that all these steps are done in proper sequence.

At any point of time, state of the sequential circuit dictates activation of appropriaate set of micro-operations. As CU is also a sequential circuit, so depending on the *present state* of CU,it generates appropriate control signals for enabling micro-operations and then CU changes it's state to *next state.*

There are two approaches to design a control unit:

I. by using hardware, and CU designed so is known as *hardiwred control unit*,

II. by using software, and CU designed so is known as *microprogrammed control unit.*

Note the spelling of *hardwired.* Many students mistakenly spells it as hardware.

1.3 Hardwired Control Unit

In hardwired control unit, we view CU as sequential logic circuit or *finite state machine*[5] to issue specific sequence of control signals against the externally initialed instruction execution. Because it uses direct logic circuit for building the CU, it attempts to minimize number of components used to build the CU and maximizes operational speed of CU. Once the CU is designed in this way, the only way to modify changes in CU operations is to redisgn the entire circuit. That is why the term *hardwired* is used here.

Following figure 1.3 shows a simplified diagram of a hardwired control unit.

Processors that follow RISC architecture uses hardwired control unit to make the control unit faster. Two approaches can be used here. In *one-hot* design method, a flip flop used per states. So, if a control unit can have total p diffferent states, we need to use p flip flops. This straight forward decision simplifies the design. The second method, *classical* design method attempts to minimize number of flip flops by using only $\log_2 p$ flip flops for p states.

[5] Switching and Finite Automata Theory by Kohavi, Jha, CU Press

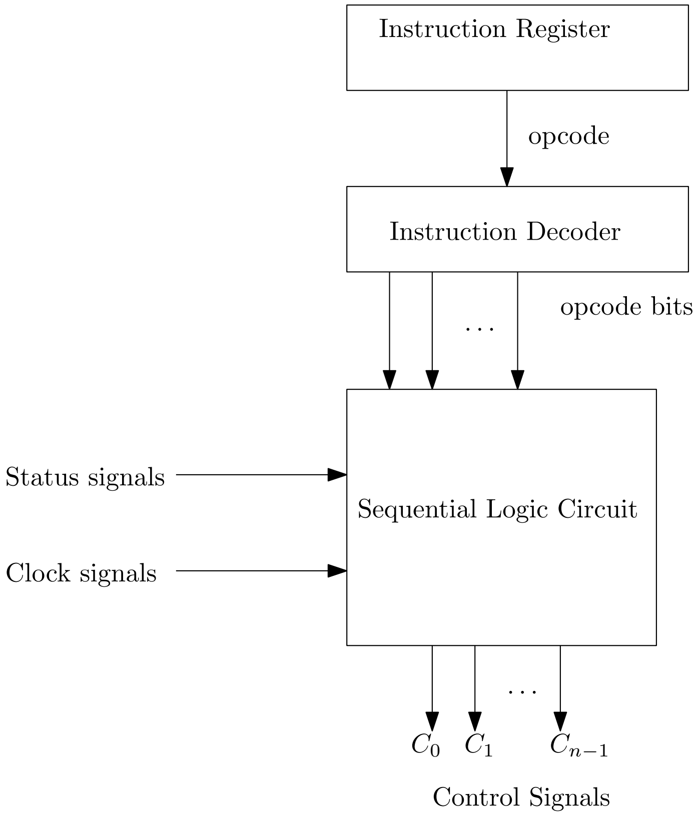

Figure 1.3: Hardwired Control Unit

In hardwired control unit, we view CU as sequential logic circuit or *finite state machine* to issue specific sequence of control signals against the externally initialed instruction execution. Because it uses direct logic circuit for building the CU, it attempts to minimize number of components used to build the CU and maximizes operational speed of CU. Once the CU is designed in this way, the only way to modify changes in CU operations is to redisgn the entire circuit. That is why the term *hardwired* is used here.

Hardwire control unit was good choice for CPU of 1950s and 1960s. But the enlarged complexity of CPUs of 1970s and early 1980s demanded a more robust software oriented control unit which was reconfigurable. However, the massive advancements in hardware design coupled with establishment of RISC architecture dominance over CISC architecture from late 1990s paved the way of comeback for hardwired control unit.

1.4 Microprogrammed Control Unit

The path breaking microprogramming paradigm for control unit design was proposed by M. Wilkes in 1951. It was so popular that many famous systems such as IBM 360, IBM 370, VAX machines, DEC-PDP1 - all employed microprogrammed control unit.

The *microprogramming* philosophy states that any instruction of machine level language (such as MOV A, B) can be decomposed into further granular hardware-level *microinstructions.* To keep parity with programming language terminology, the set of microinstruction that implements a machine-level instruction is called *microprogram.* The set of microinstructions that realize constitutent individual tasks such as fetching instruction, reading a register is called *microroutine.* As the CPU fetches opcodes from primary memory, microprogrammed control unit fetches the microinstructions from *control memory* or *microprogram memory.* As once configured, this memory is generally need not be altered, ROM is used as control memory. If any application requires some change in microprogramming, a change in the microprogram in writable ROM allows the modification in microprogrammed control unit. Like program counter points to the next instruction to be executed in a program, *microprogram counter* points to the next microinstruction in cotrol memory to be issued by control unit. Similar to branching or jumping in general program, *microprogram sequencer* enables change of sequence in issuing microinstructions. Microprogram sequencer and microaddress generator decides the control memory address of next microinstruction to be issued and loads this address in microprogram counter. The microinstruction being fetched is often referred as *control word.* This control word dictates what different control signals to be generated and it also sends next address information to the microaddress generator.

Following figure 1.4 shows a simplified diagram of a microprogrammed control unit.

1.4.1 Different Microinstruction Formats

Microinstructions can be designed in two different formats:

I. Horizontal microinstruction,

II. Vertical microinstruction.

The simplest approach of representing microinstruction is to use a distinct bit of control word for every control signal. This approah is used in *horizontal microinstruction.* Advantage of this approach is faster speed as the control

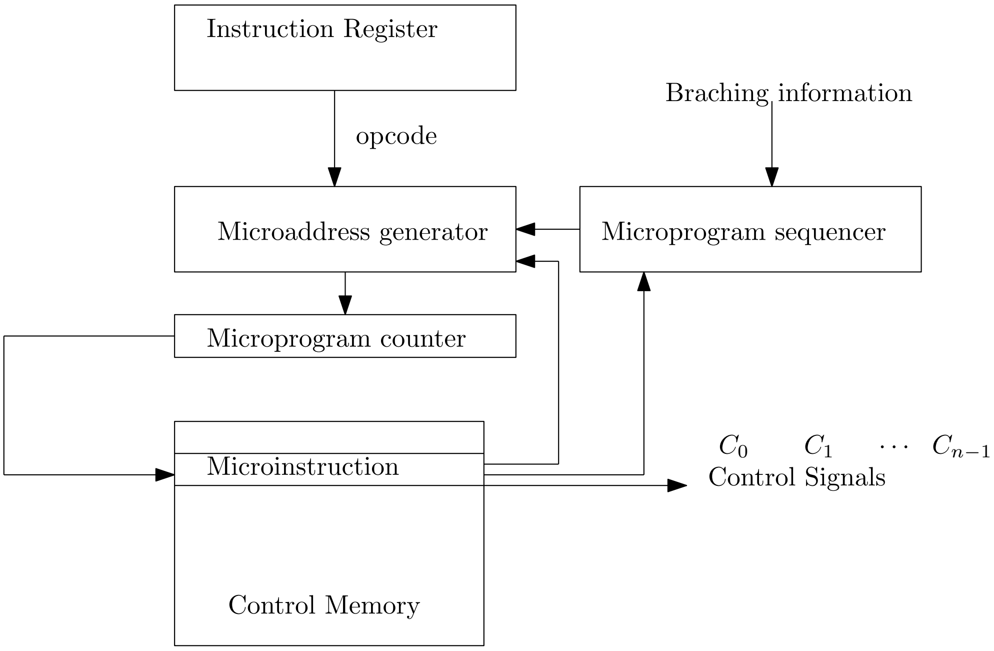

Figure 1.4: Microprogrammed Control Unit

signals to be generated is knwon as soon as microinstruction is available from control memory.

However, it's disadvantage is that the resultant width of microinstruction is very high. For example, if there are 32 different control signals, then the microinstruction has to be of 32-bit wide. If we observe the control signals minutely, then lot of mutually exclusive signals are found. For example, memory operation and I/O operation are never performed at the same time. So, we can use 1-bit instead of 2-bits for this. Say value 1 represents memory and value 0 represents I/O. Similarly, read signal and write signal can be denoted by use of just one control signal. Similarly, suppose there are 8-registers. So instead of using eight separate bits for these registers, we can use just 3-bits to encode the register. This is the approach employed in *vertical microinstruction*. Vertical microistruction reduces the width of microinstructions. As a result the amount of control memory storage requirment gets reduced.