MICROPROGRAMMED CONTROL

• Control-signals are generated by a program similar to machine language programs.

• Control word(CW) is a word whose individual bits represent various control-signals(like Add, End, Zin). {Each of the control-steps in control sequence of an instruction defines a unique combination of 1s & 0s in the CW}.

• Individual control-words in microroutine are referred to as microinstructions.

• A sequence of CWs corresponding to control-sequence of a machine instruction constitutes the microroutine.

• The microroutines for all instructions in the instruction-set of a computer are stored in a special memory called the control store(CS).

• Control-unit generates control-signals for any instruction by sequentially reading CWs of corresponding microroutine from CS.

• Microprogram counter(µPC) is used to read CWs sequentially from CS.

• Every time a new instruction is loaded into IR, output of "starting address generator" is loaded into µPC.

• Then, µPC is automatically incremented by clock, causing successive microinstructions to be read from CS. Hence, control-signals are delivered to various parts of processor in correct sequence.

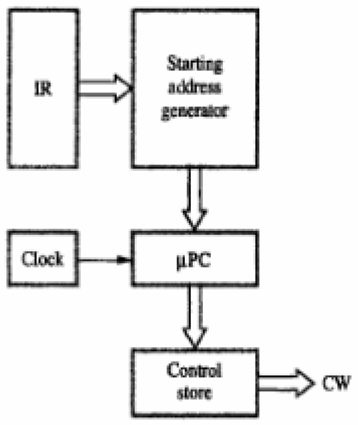


Figure 7.16 Basic organization of a microprogrammed control unit.

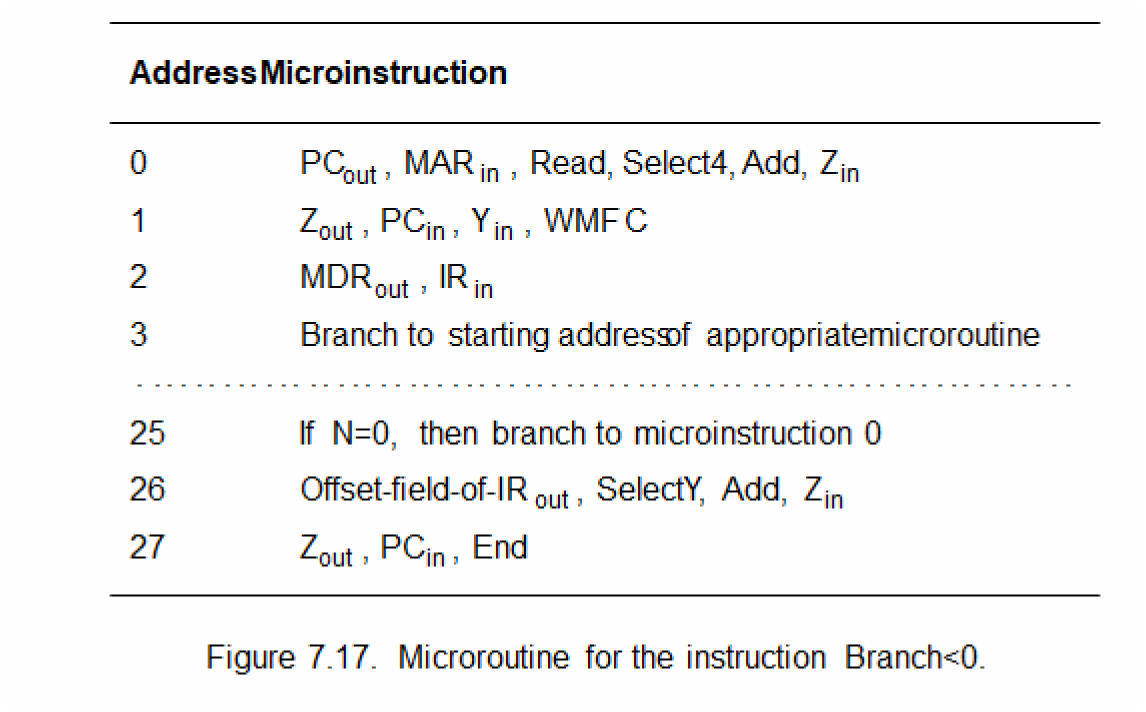
Step	Action
1	PC_{out} , MAR_{in} , Read, Select4,Add, Z_{in}
2	Z_{out} , PC_{in} , Y_{in} , WMF C
3	MDR_{out} , IR_{in}
4	$R3_{out}$, MAR_{in} , Read
5	$R1_{out}$, Y_{in} , WMF C
6	MDR_{out} , SelectY, Add, Z_{in}
7	Z_{out} , $R1_{in}$, End

Figure 7.6. Control sequence for execution of the instruction Add (R3),R1.

Micro - instruction	…	PC_{in}	PC_{out}	MAR_{in}	Read	MDR_{out}	IR_{in}	Y_{in}	Select	Add	Z_{in}	Z_{out}	$R1_{out}$	$R1_{in}$	$R3_{out}$	WMFC	End	…
1		0	1	1	1	0	0	0	1	1	1	0	0	0	0	0	0	
2		1	0	0	0	0	0	1	0	0	0	1	0	0	0	1	0	
3		0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	
4		0	0	1	1	0	0	0	0	0	0	0	0	0	1	0	0	
5		0	0	0	0	0	0	1	0	0	0	0	1	0	0	1	0	
6		0	0	0	0	1	0	0	0	1	1	0	0	0	0	0	0	
7		0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	1	

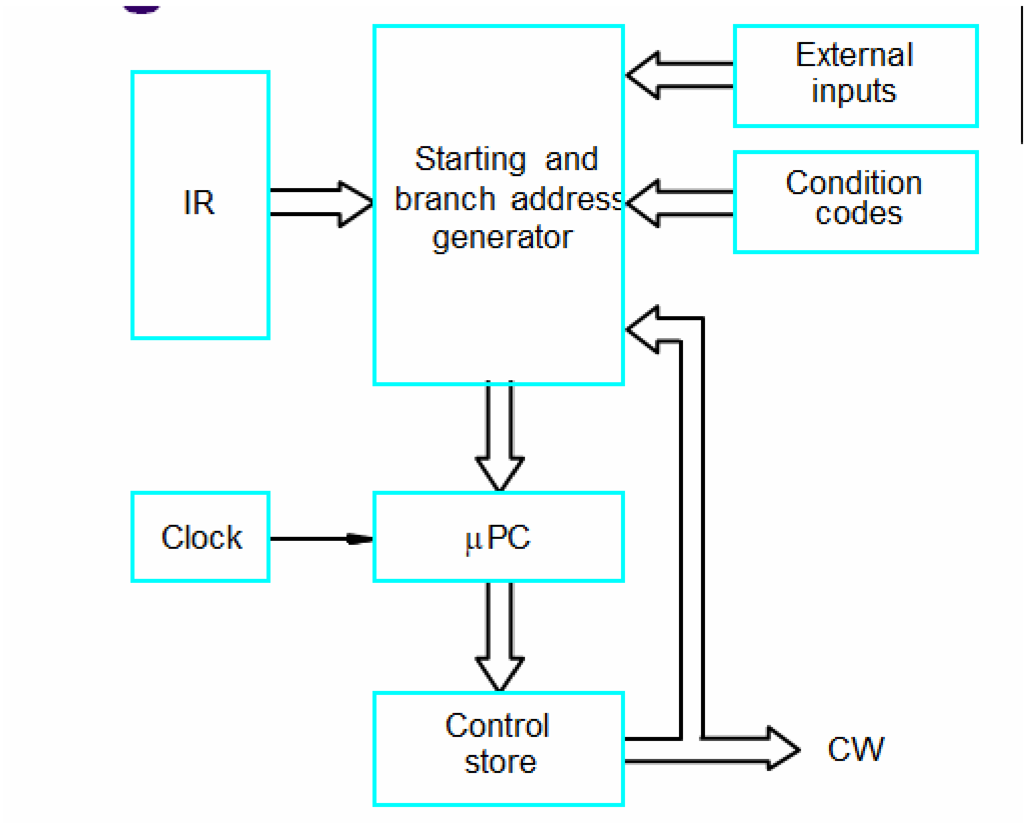
Figure 7.15 An example of microinstructions for Figure 7.6.

- The previous organization cannot handle the situation when the control unit is required to check the status of the condition codes or external inputs to choose between alternative courses of action.
- Use conditional branch microinstruction.



Address	Microinstruction
0	PC_{out} , MAR_{in} , Read, Select4, Add, Z_{in}
1	Z_{out} , PC_{in} , Y_{in} , WMFC
2	MDR_{out} , IR_{in}
3	Branch to starting addressof appropriatemicroroutine
25	If N=0, then branch to microinstruction 0
26	Offset-field-of-IR_{out} , SelectY, Add, Z_{in}
27	Z_{out} , PC_{in} , End

Figure 7.17. Microroutine for the instruction Branch<0.



MICROINSTRUCTIONS •

Drawbacks of micro programmed control:

1) Assigning individual bits to each control-signal results in long microinstructions because the number of required signals is usually large.

2) Available bit-space is poorly used because only a few bits are set to 1 in any given microinstruction.

• Solution: Signals can be grouped because

1) Most signals are not needed simultaneously.

2) Many signals are mutually exclusive.

• Grouping control-signals into fields requires a little more hardware because decoding-circuits must be used to decode bit patterns of each field into individual control signals.

Grouping of control signals

This CPU contains four general purpose registers *R0*, *R1*, *R2* and *R3* . In addition there are three other register called SOURCES, DESTIN and TEMP. These are used for temporary storage within the CPU and completely transparent to the programmer. A computer programmer cannot use these three registers.

For the proper functioning of this CPU, we need all together 24 gating signals for the transfer of information between internal CPU bus and other resources like registers.

In addition to these register gating signals, we need some other control signals which include the Read, Write, Clear Y, set carry in, WMFC, and End signal. (Here we are restricting the control signal for the case of discussion in reality, the number of signals are more).

It is also necessary to specify the function to be performed by ALU. Depending on the power of ALU, we need several control lines, one control signal for each function. Assume that the ALU that is used in the design can perform 16 different operation such as add, subtract, AND, OR etc. So we need 16 different control lines.

The above discussion indicates that 46(24+6+16) distinct signals are required. This indicates that we need 46 bits in each micro instructions, therefore the size of control word is 46.

Consider the microprogram that is shown for the add instruction. On an average 4 to 5 bits are set to 1 in each micro instruction and rest of the bits are 0. Therefore, the bit utilization is poor, and there is a scope to the utilization of bit.

If is observed that most signals are not needed simultaneously and many signals are mutually exclusive.

As for example, only one function of the ALU can be activated at a time. In out case we are considering 16 ALU operations. Instead of using 16 different signal for ALU operation, we can group them together and reduce the number of control signal. From digital logic circuit, it is obvious that instead of 16 different signal, we can use only 4 control signal for ALU operation and them use a 4 $\times$ 16 decoder to generate 16 different ALU signals. Due to the use of a decoder, there is a reduction in the size of control word.

Another possibilities of grouping control signal is: A sources for data transfer must be unique, which means that it is not possible to gate the contents of two different registers onto the bus at the same time. Similarly Read Write signals to the memory cannot be activated simultaneously.This observation suggests the possibilities of grouping the signals so that all signals that are mutually exclusive are placed in the same group. Thus a group can specify one micro operation at a time.At that point we have to use a binary coding scheme to represent a given signal within a group. As for example for 16 ALU function, four bits are enough to decode the appropriate function.

A possible grouping of the 46 control signals that are required for the above mention CPU is given in the table.

A possible grouping of signal is shown here. There may be some other grouping of signal possible. Here all out- gating of registers are grouped into one group, because the contents of only one bus is allowed to go the internal bus, otherwise there will be a conflict of data.

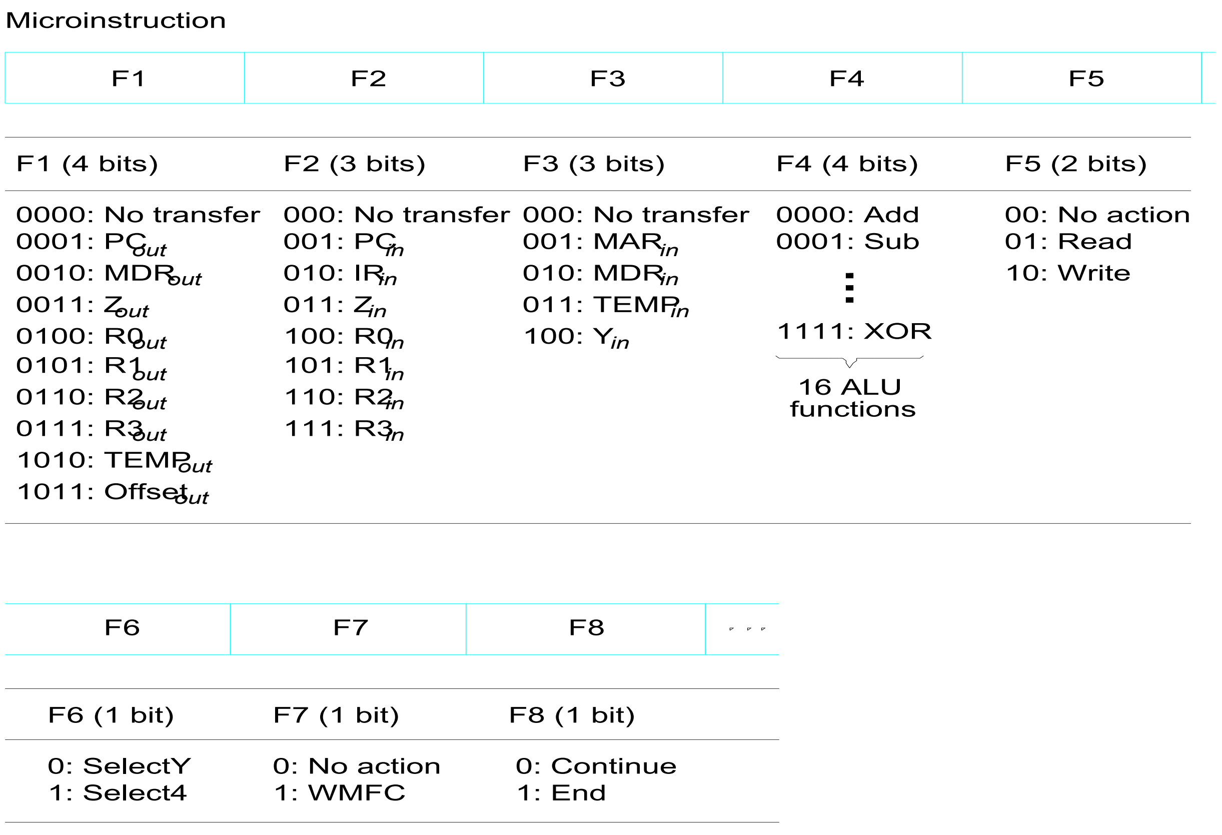
But the in-gating of registers are grouped into three different group. It implies that the contents of the bus may be stored into three different registers simultaneously transfer to MAR and Z. Due to this grouping, we are using 7 bits (3+2+2) for the in-gating signal. If we would have grouped then in one group, then only 4 bits would have been enough; but it will take more time during execution. In this situation, two clock cycles would have been required to transfer the contents of PC to MAR and Z.

Therefore, the grouping of signal is a critical design parameter. If speed of operation is also a design parameter, then compression of control word will be less.

In this grouping, 46 control signals are grouped into 10 different groups (*F1*, *F2*,………., *F10*) and the size of control word is 21. So, the size of control word is reduced from 46 to 21, which is more than 50%.

Following figure 7.19 shows the grouping of control signals

- Advantage: This method results in a smaller control-store .



Microinstruction

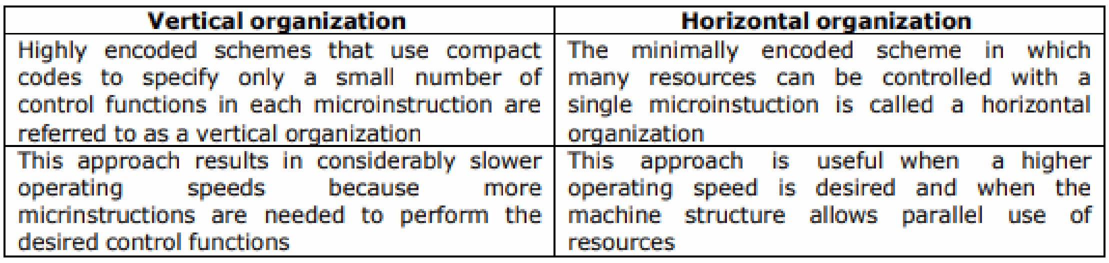
F1	F2	F3	F4	F5

F1 (4 bits)	F2 (3 bits)	F3 (3 bits)	F4 (4 bits)	F5 (2 bits)
0000: No transfer	000: No transfer	000: No transfer	0000: Add	00: No action
0001: PC_{out}	001: PC_{in}	001: MAR_{in}	0001: Sub	01: Read
0010: MDR_{out}	010: IR_{in}	010: MDR_{in}	⋮	10: Write
0011: Z_{out}	011: Z_{in}	011: TEMP_{in}	1111: XOR	
0100: R0_{out}	100: R0_{in}	100: Y_{in}	16 ALU functions	
0101: R1_{out}	101: R1_{in}			
0110: R2_{out}	110: R2_{in}			
0111: R3_{out}	111: R3_{in}			
1010: TEMP_{out}				
1011: Offset_{out}				

F6	F7	F8	. . .

F6 (1 bit)	F7 (1 bit)	F8 (1 bit)
0: SelectY	0: No action	0: Continue
1: Select4	1: WMFC	1: End

Figure 7.19. An example of a partial format for field-encoded microinstruction



Vertical organization	Horizontal organization
Highly encoded schemes that use compact codes to specify only a small number of control functions in each microinstruction are referred to as a vertical organization	The minimally encoded scheme in which many resources can be controlled with a single microinstuction is called a horizontal organization
This approach results in considerably slower operating speeds because more micrinstructions are needed to perform the desired control functions	This approach is useful when a higher operating speed is desired and when the machine structure allows parallel use of resources

Microprogram Sequencing

- If all microprograms require only straightforward sequential execution of microinstructions except for branches, letting a µPC governs the sequencing would be efficient.

- However, two disadvantages:
  - Having a separate microroutine for each machine instruction results in a large total number of microinstructions and a large control store.
  - Longer execution time because it takes more time to carry out the required branches.
  - Example: Add src, Rdst
  - Four addressing modes: register, autoincrement, autodecrement, and indexed (with indirect forms).

Microprogram Sequencing

In microprogrammed controlled CU,

- Each machine instruction can be implemented by a microroutine.
- Each microroutine can be accessed initially by decoding the machine instruction into the starting address to be loaded into the μ PC.

Writing a microprogram for each machine instruction is a simple solution, but it will increase the size of control store.

The most machine instructions can operate in several addressing modes. If we write different microroutine for each addressing mode, then most of the cases, we are repeating some part of the microroutine.

The common part of the microroutine can be shared by several microroutine, which will reduse the size of control store. This results in a considerable number of branch microinstructions being needed to transfer control among various parts. So, it introduces branching capabilities within the microinstruction.

This indicates that the microprogrammed control unit has to perform two basic tasks:

- Microinstruction sequencing: Get the next microinstruction from the control memory.
- Microinstruction execution: Generate the control signals needed to execute the microinstruction.

In designing a control unit, these tasks musts be considered together, because both affect the format of the microinstruction and the timing of control unit.

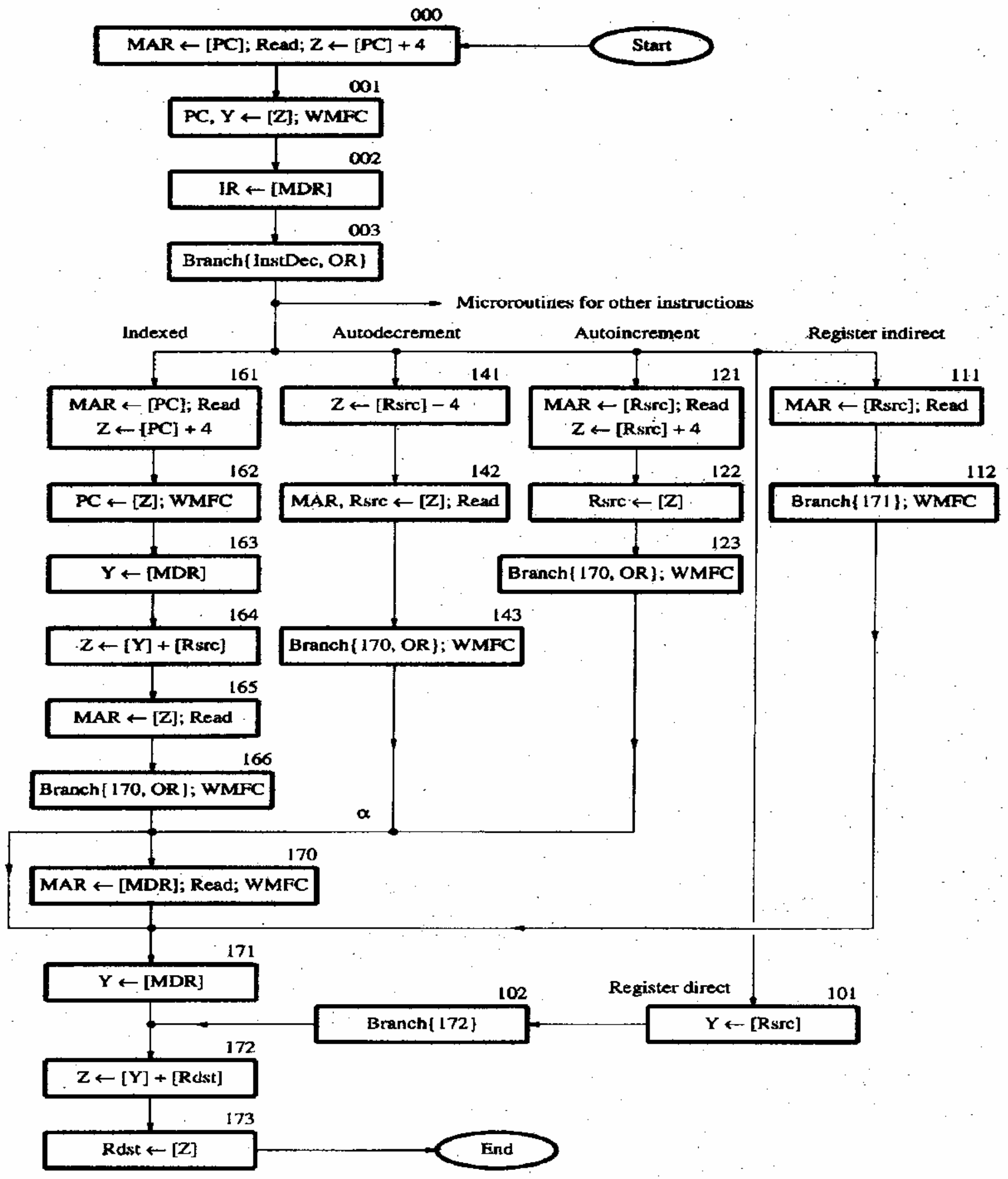
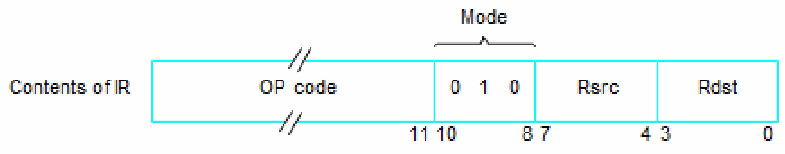


Figure 7.20. Flowchart of a microprogram for the Add src,Rdst instruction.



Textbook page 439

Address (octal)	Microinstruction
000	PC_{out}, MAR_{in}, Read, Select4, Add, Z_{in}
001	Z_{out}, PC_{in}, Y_{in}, WMFC
002	MDR_{out}, IR_{in}
003	μBranch $\{\mu PC \leftarrow 101$ (from Instruction decoder);
	$\mu PC_{5,4} \leftarrow [IR_{10,9}]$; $\mu PC_3 \leftarrow [\overline{IR_{10}}] \cdot [\overline{IR_9}] \cdot [IR_8]\}$
121	$Rsrc_{out}$, MAR_{in}, Read, Select4, Add, Z_{in}
122	Z_{out}, $Rsrc_{in}$
123	μBranch $\{\mu PC \leftarrow 170; \mu PC_0 \leftarrow [\overline{IR_8}]\}$, WMFC
170	MDR_{out}, MAR_{in}, Read, WMFC
171	MDR_{out}, Y_{in}
172	$Rdst_{out}$, SelectY, Add, Z_{in}
173	Z_{out}, $Rdst_{in}$, End

Figure 7.21. Microinstruction for Add (Rsrc)+,Rdst.
Note: Microinstruction at location 170 is not executed for this addressing mode.

MICROINSTRUCTIONS WITH NEXT-ADDRESS FIELDS

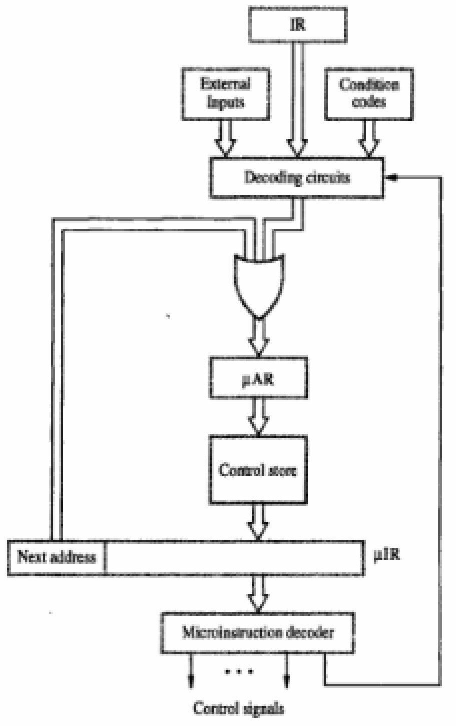


Figure 7.22 Microinstruction-sequencing organization.

The microprogram requires several branch microinstructions which perform no useful operation. Thus, they detract from the operating speed of the computer.

• Solution: Include an address-field as a part of every microinstruction to indicate the location of the next microinstruction to be fetched. (This means every microinstruction becomes a branch microinstruction).

• The flexibility of this approach comes at the expense of additional bits for the address-field. • Advantage: Separate branch microinstructions are virtually eliminated. There are few limitations in assigning addresses to microinstructions. There is no need for a counter to keep track of sequential addresse.

Hence, the µPC is replaced with a µAR (Microinstruction Address Register). {which is loaded from the next-address field in each microinstruction}.

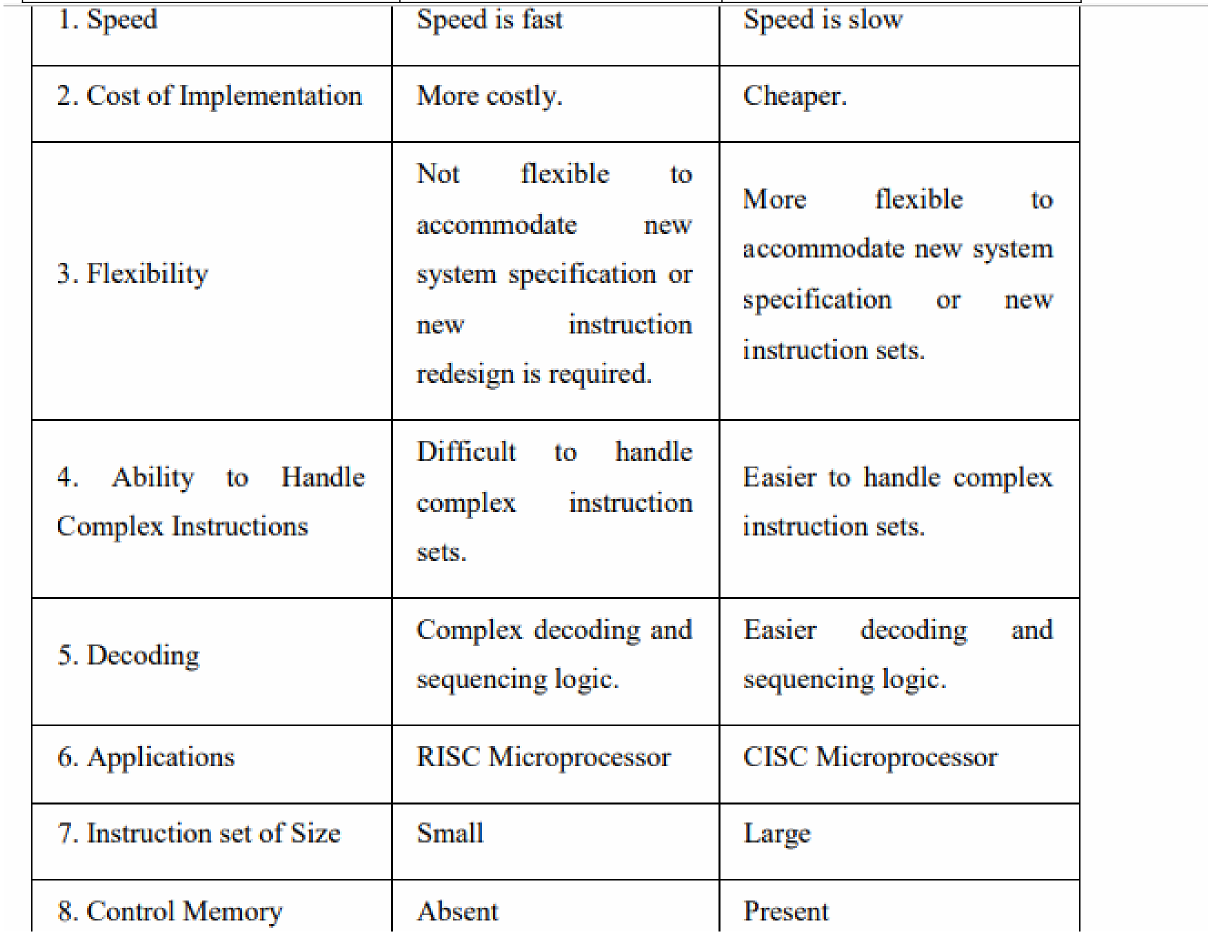
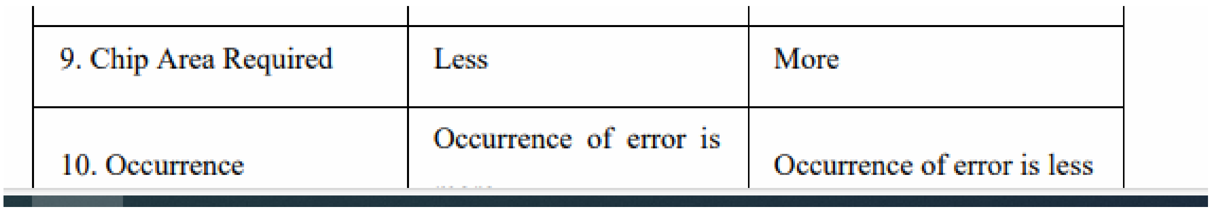
• The next-address bits are fed through the OR gate to the µAR, so that the address can be modified on the basis of the data in the IR, external inputs and condition-codes. • The decoding circuits generate the starting-address of a given microroutine on the basis of the opcode in the IR.

PREFETCHING MICROINSTRUCTIONS •

Drawback of microprogrammed control: Slower operating speed because of the time it takes to fetch microinstructions from the control-store.

• Solution: Faster operation is achieved if the next microinstruction is pre-fetched while the current one is being executed.

Difference between hardwired and microprogrammed control

Attributes	Hardwired Control	Microprogramm control
1. Speed	Speed is fast	Speed is slow
2. Cost of Implementation	More costly.	Cheaper.
3. Flexibility	Not flexible to accommodate new system specification or new instruction redesign is required.	More flexible to accommodate new system specification or new instruction sets.
4. Ability to Handle Complex Instructions	Difficult to handle complex instruction sets.	Easier to handle complex instruction sets.
5. Decoding	Complex decoding and sequencing logic.	Easier decoding and sequencing logic.
6. Applications	RISC Microprocessor	CISC Microprocessor
7. Instruction set of Size	Small	Large
8. Control Memory	Absent	Present
9. Chip Area Required	Less	More
10. Occurrence	Occurrence of error is	Occurrence of error is less

HORIZONTAL M-PROGRAMMED CU	VERTICAL M-PROGRAMMED CU
It supports longer control word.	It supports shorter control word.
It allows higher degree of parallelism. If degree is n, then n Control Signals are enabled at a time.	It allows low degree of parallelism i.e., degree of parallelism is either 0 or 1.
No additional hardware is required.	Additional hardware in the form of decoders are required to generate control signals.
It is faster than Vertical micro-programmed control unit.	it is slower than Horizontal micro-programmed control unit.
It is less flexible than Vertical micro-programmed control unit.	It is more flexible than Horizontal micro-programmed control unit.
Horizontal micro-programmed control unit uses horizontal microinstruction, where every bit in the control field attaches to a control line.	Vertical micro-programmed control unit uses vertical microinstruction, where a code is used for each action to be performed and the decoder translates this code into individual control signals.
Horizontal micro-programmed control unit makes less use of ROM encoding than vertical micro-programmed control unit.	Vertical micro-programmed control unit makes more use of ROM encoding to reduce the length of the control word.

Emulation:-

Emulation is the process of imitating a hardware/software program/platform on another program or platform. This makes it possible to run programs on systems not designed for them. Emulators, as the name implies, emulate the functions of one system on another. Thus, the second system behaves like the original system, attempting to exactly reproduce the external behaviors of the first system. It is a process of enabling one computer system to have like another computer system.

• Suppose we add to the instruction-repository of a given computer M1, an entirely new set of instructions that is in fact the instruction-set of a different computer M2.

• Programs written in the machine language of M2 can be then be run on computer M1 i.e. M1 emulates M2. • Emulation allows us to replace obsolete equipment with more up-to-date machines. • If the replacement computer fully emulates the original one, then no software changes have to be made to run existing programs. • Emulation is easiest when the machines involved have similar architectures

Emulators are used in

Cross-Platform Development: Software developers use emulators to test and debug programs on different platforms without needing to own the actual hardware.

Gaming: Emulators are frequently used to run games designed for older gaming consoles or arcade machines on modern PCs or mobile devices.

Security and Forensics: Emulation can be used in computer security for analyzing and understanding the behavior of malware and for forensic investigations.

Education and Research: Emulation is also a valuable tool for computer science and engineering students and researchers who want to experiment with different architectures and systems.

Examples:- QEMU, VirtualBox

Bit Slicing :-

Bit slicing is a technique for constructing a processor from modules of processors of smaller bit width, for the purpose of increasing the word length; in theory to make an arbitrary n-bit central processing unit (CPU). Each of these component modules processes one bit field or "slice" of an operand. The grouped processing components would then have the capability to process the chosen full word-length of a given software design. Bit slicing was particularly popular in the 1970s and early 1980s but has largely been replaced by more efficient and versatile design approaches.

In a bit-sliced architecture, data, such as binary numbers, is divided into individual bits. Each bit is processed separately by its corresponding slice. For example, a 16-bit ALU in a bit-sliced design would consist of 16 slices, each handling one bit of data.

Modularity: Bit-sliced processors are highly modular. Each slice is a self-contained unit that can be duplicated to handle wider data paths. This modularity makes it relatively easy to extend the processor to handle larger data widths.

Simplification: Each slice typically contains simple logic gates and components to perform basic operations like addition, subtraction, and logical functions (AND, OR, XOR, etc.). This simplicity allows for easy replication of slices.

Limitations

1.More complexity and overhead

2.Cost is more

Question Bank:-

What do mean by emulation .
Define the following terms. • Micro programmed control.

• Control word • Control store • Microprogramm counter • microinstruction
State the meaning of fallowing terms. • Micro routine • Micro instructions
What are the different ways to reduce the number of bits in microinstructions.
What do you mean by bit slicing.
What are the various factors that reduces the performance of pipelined processor.
What are the different situations when microprogram counter is not get increment with next address field?
Draw and explain the basic organization of a microprogrammed control unit.
How to reduce the length of microinstruction explain with example.
Explain microinstruction with next address field.
Explain prefetching of microinstructions. and state its advantages.
With neat diagram explain the basic organization of microprogramed control unit also draw and explain the flowchart for microprogram for add scr, Rdest for index addressing mode.
Explain microprogram sequencing.
Draw and explain the organization of the control unit to allow conditional branching in micro program.
Explain the features of bit slicing.
Draw and explain the architecture for hardwired control unit.
Differentiate between horizontal and vertical organization.
Differentiate between hardwired and microprogramed control.
State advantages and disadvantages of microprogrammed control.
Why there is need of grouping of control signals explain with example.
Write an micro instruction for Add(Rsrc)+,Rdst.
For a single bus organization of data paths inside CPU ,write a micro program of micro instructions and draw flowchart of a microprogram for the following instruction.MOV(Rscr)+,Rdst.
Write the control sequence for the instruction add[r5],r6 and also write the microinstructions for the same.
Draw and explain the organization of the control unit to allow conditional branching in microprogram.
Differentiate between hardwired and microprogrammed control.
What do mean by micro program sequencing.

For a single bus organization of data paths inside CPU ,write a micro program of micro instructions and draw flowchart of a microprogram for the following instruction. add(Rscr),Rdst.
For a single bus organization of data paths inside CPU, write a micro program of micro instructions and draw flowchart of a microprogram for the following instruction. sub Rscr,Rdst.
For a single bus organization of data paths inside CPU ,write a micro program of micro instructions and draw flowchart of a microprogram for the following instruction.add -(Rscr),Rdst.