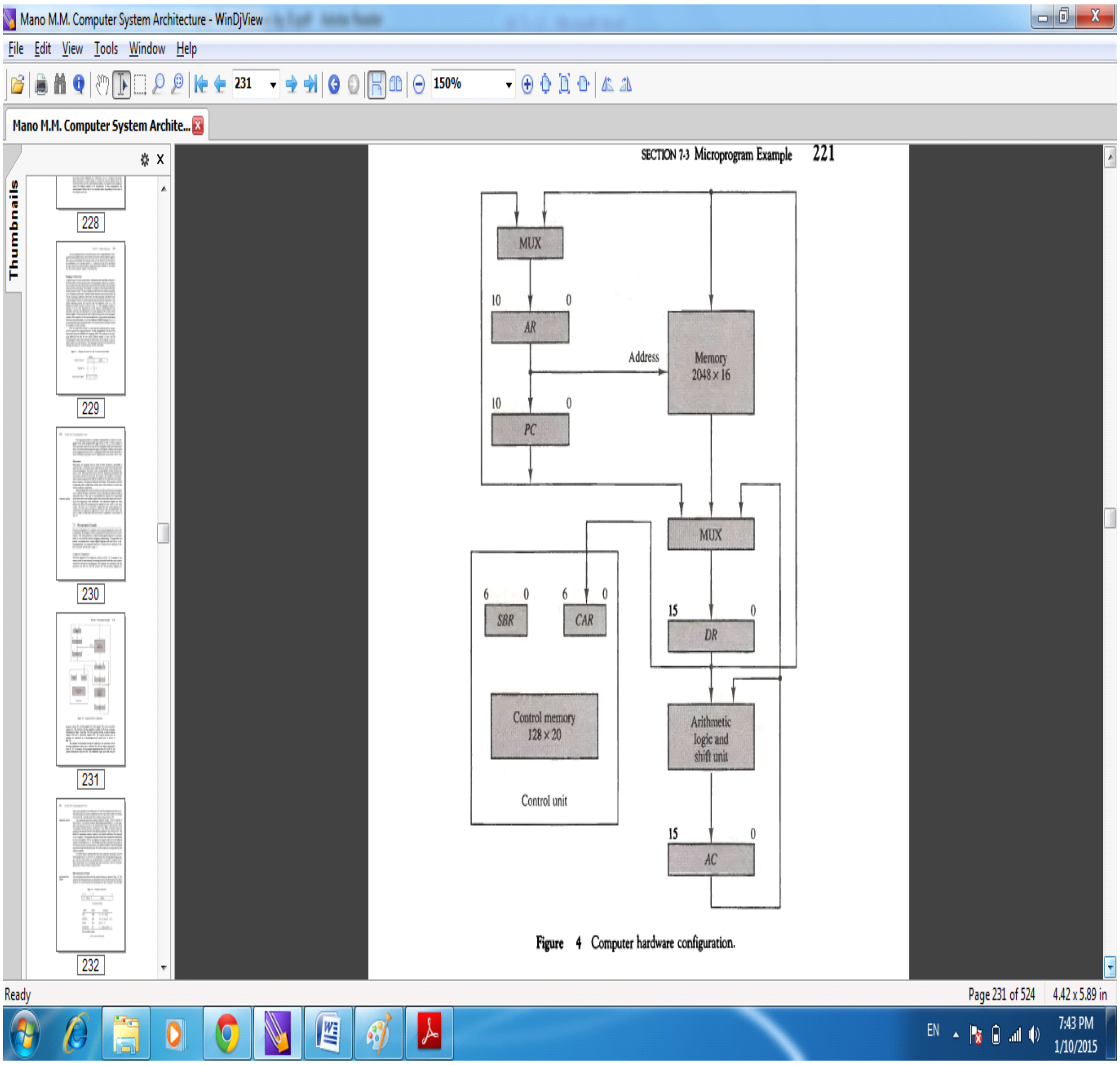
Subroutines

Subroutines are programs that are used by other routines to accomplish a particular task. A subroutine can be called from any point within the main body of the microprogram.

For example, the sequence of microoperations needed to generate the effective address of the operand for an instruction is common to all memory reference instructions. This sequence could be a subroutine that is called from within many other routines to execute the effective address computation.

Microprograms that use subroutines must have a provision احتياط for storing the return address during a subroutine call and restoring the address during a subroutine return. This may be accomplished by placing the incrementedoutput from the control address register into a subroutine register and branching to the beginning of the subroutine. The subroutine register can then become the source for transferring the address for the return to the main routine.



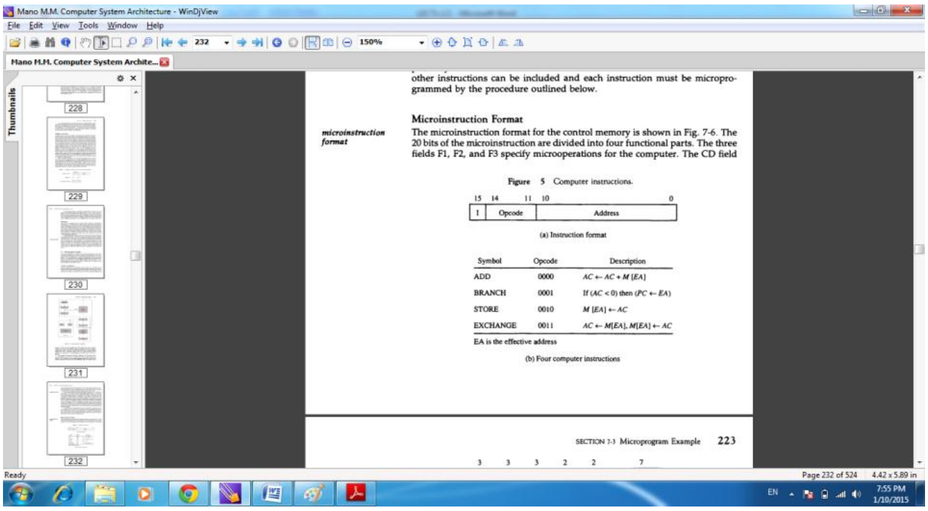
Computer microprogram Example

The block diagram of the computer shown consists of two memory units: a main memory for storing instructions and data, and a control memory for storing the microprogram. Four registers are associated with the processor unit and two with the control unit. The processor registers are program counter PC, address register AR, data register DR, and accumulator register AC.

The control unit has a control address register CAR and a subroutine register SBR.

Computer instruction format

The computer instruction format is shown in fig 5(b), consists of three fields: a 1-bit held for indirect addressing symbolized by (I), a 4-bit operation code (opcode), and an 11-bit address field. Figure 5(b) lists four of the 16 possible memory-reference instructions. The ADD instruction adds the content of the operand found in the effective address to the content of AC. The BRANCH instruction causes a branch to the effective address if the operand in AC is negative. The program proceeds with the next consecutive instruction if AC is not negative. The AC is negative if its sign bit (the bit in the leftmost position of the register) is a 1. The STORE instruction transfers the content of AC into the memory word specified by the effective address. The EXCHANGE instruction swaps the data between AC and the memory word specified by the effective address.


other instructions can be included and each instruction must be microprogrammed by the procedure outlined below.

Microinstruction Format

microinstruction format

The microinstruction format for the control memory is shown in Fig. 7-6. The 20 bits of the microinstruction are divided into four functional parts. The three fields F1, F2, and F3 specify microoperations for the computer. The CD field

Figure 5 Computer instructions.

15	14 – 11	10 – 0
I	Opcode	Address

(a) Instruction format

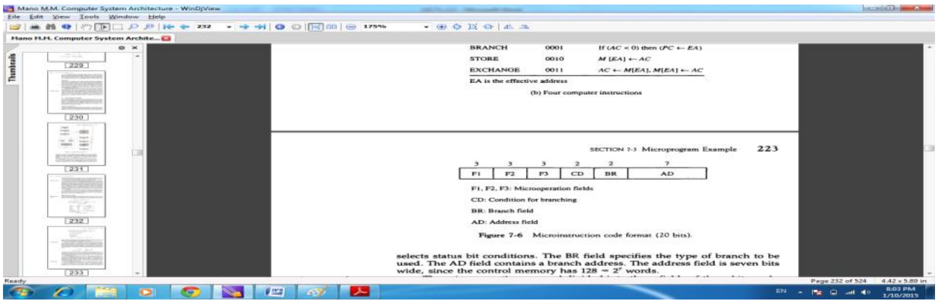
Symbol	Opcode	Description
ADD	0000	$AC \leftarrow AC + M[EA]$
BRANCH	0001	If $(AC < 0)$ then $(PC \leftarrow EA)$
STORE	0010	$M[EA] \leftarrow AC$
EXCHANGE	0011	$AC \leftarrow M[EA], M[EA] \leftarrow AC$

EA is the effective address

(b) Four computer instructions

SECTION 7-3 Microprogram Example 223

Microinstruction Format

The 

BRANCH	0001	If ($AC < 0$) then ($PC \leftarrow EA$)
STORE	0010	$M[EA] \leftarrow AC$
EXCHANGE	0011	$AC \leftarrow M[EA], M[EA] \leftarrow AC$

EA is the effective address

(b) Four computer instructions

SECTION 7-3 Microprogram Example 223

3	3	3	2	2	7
F1	F2	F3	CD	BR	AD

F1, F2, F3: Microoperation fields

CD: Condition for branching

BR: Branch field

AD: Address field

Figure 7-6 Microinstruction code format (20 bits).

selects status bit conditions. The BR field specifies the type of branch to be used. The AD field contains a branch address. The address field is seven bits wide, since the control memory has $128 = 2^7$ words.

microinstruction format for the control memory is shown in Fig. 7-6. The20 bits of the microiristruction are divided into four functional parts. The three fields Fl, F2, and F3 specify microoperations for the computer.

The CD field selects status bit conditions. The BR field specifies the type of branch to be used. The AD field contains a branch address. The address field is seven bits wide, since the control memory has 128 = 27 words.

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	$AC \leftarrow \text{shl } AC$	SHL
100	$AC \leftarrow \text{shr } AC$	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

BR	Symbol	Function
00	JMP	$CAR \leftarrow AD$ if condition = 1
		$CAR \leftarrow CAR + 1$ if condition = 0
01	CALL	$CAR \leftarrow AD$, $SBR \leftarrow CAR + 1$ if condition = 1
		$CAR \leftarrow CAR + 1$ if condition = 0
10	RET	$CAR \leftarrow SBR$ (Return from subroutine)
11	MAP	$CAR(2\text{-}5) \leftarrow DR(11\text{-}14)$, $CAR(0,1,6) \leftarrow 0$

The microoperations are subdivided into three fields of three bits each. The three bits in each field are encoded to specify seven distinct microoperations as listed in Table 7-1. This gives a total of 21 microoperations. No more than three microoperations can be chosen for a microinstruction, one from each field. If fewer than three microoperations are used, one or more of the fields will use the binary code 000 for no operation. As an illustration, a microinstruction can specify two simultaneous microoperations from F2 and F3 and none from Fl.

DR M[AR] with F2 = 100

and PC `PC + 1 with F3 = 101

The nine bits of the microoperation fields will then be 000 100 101.

When condition bits (CD) are used in conjunction with the BR (branch) field, it provides an unconditional branch operation.

The first condition is always a 1, so that a reference to CD = 00 (or the symbol U) will always find the condition to be true. When this condition is used in conjunction with the BR (branch) field, it provides an unconditional branch operation. The indirect bit I is available from bit 15 of DR after an instruction is read from memory. The sign bit of AC provides the next status bit. The zero value, symbolized by Z, is a binary variable whose value is equal to 1 if all the bits in AC are equal to zero.

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. As shown in Table 7-1, when BR = 00, the control performs a jump QMP) operation (which is similar to a branch), and when BR = 01, it performs a call to subroutine (CALL) operation. The two operations are identical except that a call microinstruction stores the return address in the subroutine register SBR. The jump and call operations depend on the value of the CD field. If the status bit condition specified in the CD field is equal to 1, the next address in the AD field is transferred to the control address register CAR. Otherwise, CAR is incremented by 1.

The return from subroutine is accomplished with a BR field equal to 10. This causes the transfer of the return address from SBR to CAR. The mapping from the operation code bits of the instruction to an address for CAR is accomplished when the BR field is equal to 11. This mapping is as depicted in Fig. 7-3. The bits of the operation code are in DR(11-14) after an instruction is read from memory. Note that the last two conditions in the BR field are independent of the values in the CD and AD fields.

NEXT P235 (Symbolic Microinstructions)

*Symbolic Microinstructions

Each symbolic microinstruction is divided into five fields: label, microoperations, CD, BR, and AD.

* The Fetch Routine(and fetch & decode)

* Binary microprogram

Hardwired vs Microprogrammed control

There are two major types of control organization: hardwired control and microprogrammed control. In the hardwired organization, the control logic is implemented with gates, flip-flops, decoders, and other digital circuits. It has the advantage that it can be optimized to produce a fast mode of operation. In the microprogrammed organization, the control information is stored in a control memory. The control memory is programmed to initiate the required sequence of microoperations. A hardwired control, as the name implies, requires changes in the wiring among the various components if the design has to be modified or changed. In the microprogrammed control, any required changes or modifications can be done by updating the microprogram in control memory.