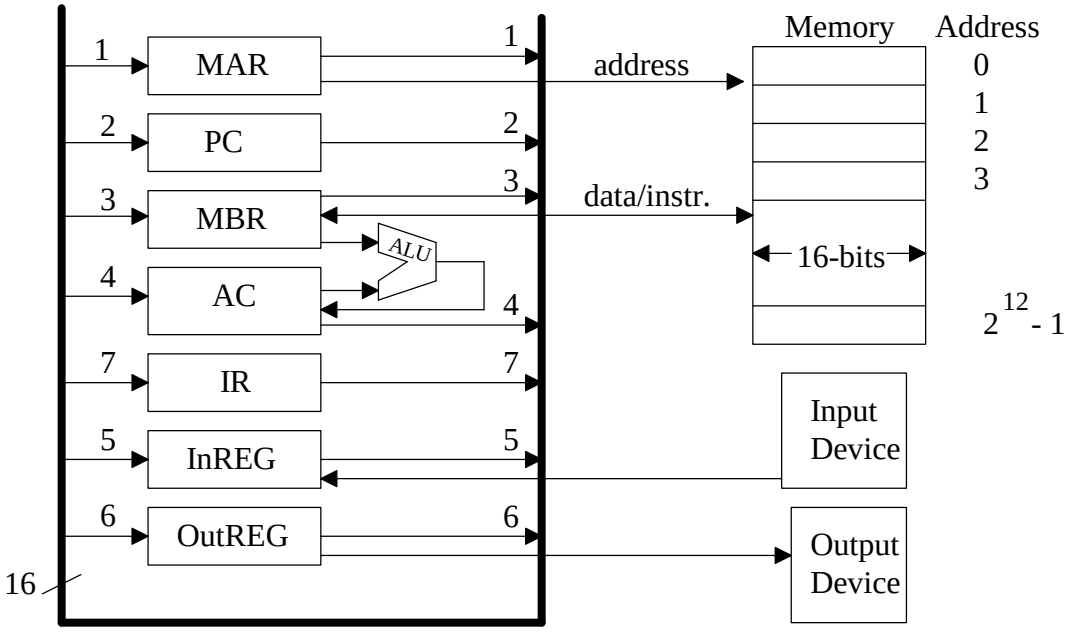
MARIE Assembly Language

Type of Instructions	Mnemonic	Hex Opcode	Description
Arithmetic	ADD X	3	Add the contents of address X to AC
	SUBT X	4	Subtract the contents of address X from the AC
	ADDI X	B	Add Indirect: Use the value at X as the actual address of the data operand to add to AC
	CLEAR	A	Put all zeros in the AC
Data Transfer	LOAD X	1	Load the contents of address X into AC
	STORE X	2	Store the contents of AC at address X
I/O	INPUT	5	Input a value from the keyboard into AC
	OUTPUT	6	Output the value in AC to the display
Branch	JUMP X	9	Unconditional branch to X by loading the value of X into PC
	SKIPCOND C	8	Skip the next instruction based on the condition, C: $C = 000_{16}$: skip if AC is negative ($b_{11}b_{10} = 00_2$) $C = 400_{16}$: skip if the AC = 0 ($b_{11}b_{10} = 01_2$) $C = 800_{16}$: skip if the AC is positive ($b_{11}b_{10} = 10_2$)
Subroutine call and return	JNS X	0	Jump-and-Store: Store the PC at address X and jump to X+1
	JUMPI X	C	Use the value **at** X as the address to jump to
	HALT	7	Terminate the program

MARIE Instruction Format

Revised Figure 4.9 Datapath in MARIE



MARIE Assembly Language Example 1: RESULT = X + Y - Z

Address	Label	Assembly Language	Machine Language
0		LOAD X	1006_{16}
1		ADD Y	3007_{16}
2		SUBT Z	4008_{16}
3		STORE RESULT	2009_{16}
4		OUTPUT	6000_{16}
5		HALT	7000_{16}
6	X,	DEC 10	$000A_{16}$
7	Y,	DEC 20	0014_{16}
8	Z,	DEC 5	0005_{16}
9	RESULT,	DEC 0	0000_{16}

MARIE Assembly Language Example 2: Print null terminated string to output

HLL:
```
index = 0
while str[index] != 0 do
    output str[index]
    index = index + 1
end while
```

Address	Label	Assembly Language	Machine Language
0		CLEAR	$A000_{16}$
1		STORE INDEX	2011_{16}
2	WHILE,	LOAD STR_BASE	1013_{16}
3		ADD INDEX	3011_{16}
4		STORE ADDR	2012_{16}
5		CLEAR	$A000_{16}$
6		ADDI ADDR	$B012_{16}$
7		SKIPCOND 400	8400_{16}
8		JUMP DO	$900A_{16}$
9		JUMP END_WHILE	$900A_{16}$
A	DO,	OUTPUT	6000_{16}
B		LOAD INDEX	$100D_{16}$
C		ADD ONE	$300B_{16}$
D		STORE INDEX	2011_{16}
E		JUMP WHILE	9002_{16}
F	END_WHILE,	HALT	7000_{16}
10	ONE,	DEC 1	0001_{16}
11	INDEX,	DEC 0	0000_{16}
12	ADDR,	HEX 0	0000_{16}
13	STR_BASE,	HEX 14	0014_{16}
14	STR,	DEC 72 / H	0048_{16}
15		DEC 69 / E	0045_{16}
16		DEC 76 / L	$004C_{16}$
17		DEC 76 / L	$004C_{16}$
18		DEC 79 / O	$004F_{16}$
19		DEC 13 /carriage return	$000D_{16}$
1A		DEC 87 / W	0057_{16}
1B		DEC 79 / O	$004F_{16}$
1C		DEC 82 / R	0052_{16}
1D		DEC 76 / L	$004C_{16}$
1		DEC 68 / D	0044_{16}
1F	NULL,	DEC 0 / NULL CHAR	0000_{16}