计算机体系结构实验

实验报告　（实验一）

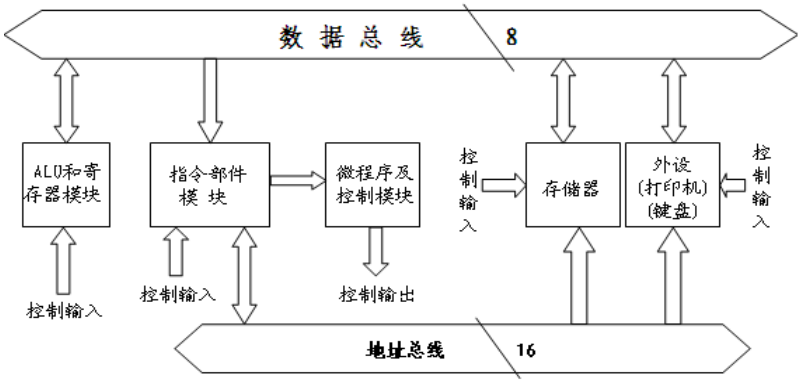
李秦琦
11300720096

目录

实验一　8 位微程序控制计算机设计

实验内容

1.1　8 位微程序控制计算机基本结构



8位单累加器计算机基本结构

总线：该计算机采用单总线，即CPU的内部总线和外部总线均为一条总线。它的结构简单，实现较方便。但它的速度较慢，无法实现流水线和并行操作。

CPU：它的CPU由ALU和寄存器模块、指令部件模块及微程序控制模块组成。

寄存器：ALU和寄存器模块包括通用寄存器（含累加器）、ALU（包括暂存器）、状态寄存器等，它可采用单累加器多寄存器或多累加器结构（见第一章）。

ALU：ALU可完成各种算术、逻辑运算功能，如加、减、与、或、取反、取负、移位等。状态寄存器可包括进位位、全零标志位、负数标志位、溢出标志位等。

指令部件：指令部件模块包括程序计数器（PC）及它的控制电路（加1、接数等）、指令寄存器（它一般包括一至三个8位寄存器，与最长的指令相对应）等。

微程序控制部件：微程序控制摸块包括微程序存贮器（现为1Kx32）、微程序计数器（MPC）及它的控制电路（加1、接数等）、微指令寄存器、时序控制电路、微操作信号发生电路等。

存贮器：存贮器为外接的RAM存贮器（现为32Kx8）

外设：包括打印机输出设备、键盘输入设备。它们均接于总线上，有分别的读、写控制信号。

1.2 指令系统和 CPU 结构

表格 1　　定义指令系统　(No. 109)

编号	汇编码	操作	指令码
1	MOV A, Ri	(Ri) → A	00000iii
2	MOV Ri, A	(A) → Ri	00001iii
3	MOV A, @Ri	(7EH[Ri]) → A	00010iii
4	MOV @Ri, A	(A) → 7EH[Ri]	00011iii
5	ADD A, Ri	(A) + (Ri) → A	00100iii
6	SUB A, Ri	(A) – (Ri) → A	00101iii
7	MOV A, #data	date → A	00110000 dddddddd
8	MOV Ri, #data	data → Ri	00111iii dddddddd
9	LDA addr	(addr) → A	01000000 addrh addrl
10	STA addr	(A) → addr	01001000 addrh addrl
11	JC addr	if CY=1 then addr→PC else (PC)+3→PC	01010000 addrh addrl
12	JMP addr	addr → PC	01011000 addrh addrl
13	JNKB addr	if KRIX=0 then addr→PC else (PC)+3→PC	01100000 addrh addrl
14	JNPB addr	if PRIX=0 then addr→PC else (PC)+3→PC	01101000 addrh addrl
15	CALL add	保存 PC 进栈，按 addr 转到子程序	01110000 addrh addrl
16	RET	PC 退栈，子程序返回	01111000
17	RSP	7fffH → SP	10000000
18	SUB A, @Ri	(A) – (7EH[Ri]) → A	10001iii
19	ASR A	(A) 算术右移一位	10010000
20	CLR addr	0 → (addr)	10011000
21	PUSH Ri	Ri → 堆栈，SP → 1	10100iii
22	POP Ri	SP+1，堆栈 → Ri	10101iii
23	JZ addr	if Z=1 then addr → PC else (PC)+3 → PC	10110000 addrh addrl
24	ADC A, Ri	if CY=1 then (A)+(Ri)+1→A else (A)+(Ri)→A	10111iii

注: 对寄存器间接寻址指令，如 MOV @Ri, A，由于 Ri 为 8 位，而存储器地址为 16 位，故取 Ri 为地址低 8 位，高 8 位固定为 7EH。

CPU 结构框图

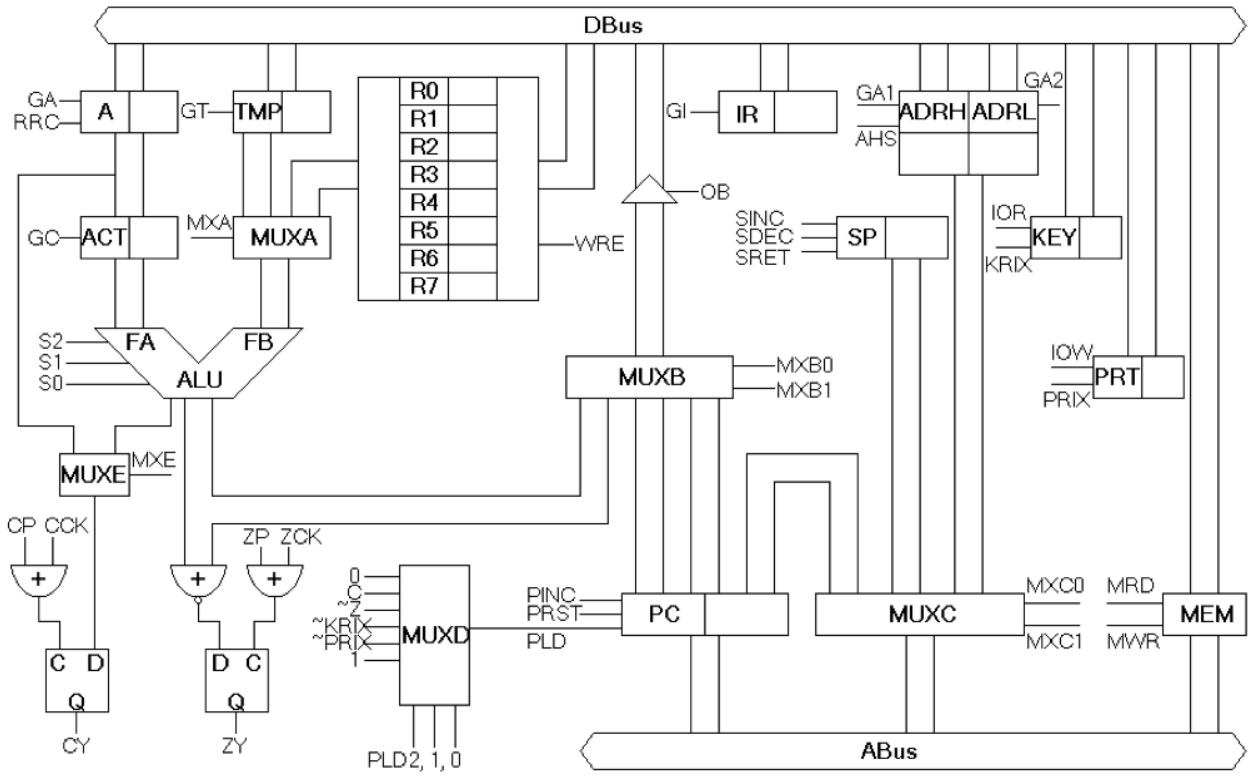


图1 CPU结构框图

1.3 设计指令流程

表格 2 指令流程定义

编号	汇编代码	指令流程
0	取指令（所有指令的最后一步）	T0: (PC) $\xrightarrow{MC=00}$ AB; (M) $\xrightarrow{CRDX=0}$ DB $\xrightarrow{GI=0}$ IR; (PC)+1 $\xrightarrow{PINC=0}$ PC; (A) $\xrightarrow{GC=0}$ ACT; MPLD=0;
1	MOV A, Ri	T1: (Ri) $\xrightarrow{MA=0\ S=011}$ ALU $\xrightarrow{MB=00}$ DB $\xrightarrow{GA=0}$ A;
2	MOV Ri, A	T1: (ACT) $\xrightarrow{S=010}$ ALU $\xrightarrow{MB=00}$ DB $\xrightarrow{WRE=0}$ Ri;
3	MOV A, @Ri	T1: (Ri) $\xrightarrow{MA=0\ S=011}$ ALU $\xrightarrow{MB=00}$ DB $\xrightarrow{GA2=0}$ ADRL; 7EH $\xrightarrow{AHS=0}$ ADRH; T2: (ADR) $\xrightarrow{MC=01}$ AB; (M) $\xrightarrow{CRDX=0}$ DB $\xrightarrow{GA=0}$ A;
4	MOV @Ri, A	T1: (Ri) $\xrightarrow{MA=0\ S=011}$ ALU $\xrightarrow{MB=00}$ DB $\xrightarrow{GA2=0}$ ADRL; 7EH $\xrightarrow{AHS=0}$ ADRH; T2: (ADR) $\xrightarrow{MC=01}$ AB; (ACT) $\xrightarrow{S=010}$ ALU $\xrightarrow{MB=00}$ DB $\xrightarrow{CWRX=0}$ M;
5	ADD A, Ri	T1: (ACT)+(Ri) $\xrightarrow{MA=0\ S=000}$ ALU $\xrightarrow{MB=00}$ DB $\xrightarrow{GA=0}$ A; COUT $\xrightarrow{CP=0\ ME=0}$ CY;

编号	汇编代码	指令流程
6	`SUB A, Ri`	T1: (ACT)-(Ri) $\xrightarrow{\text{MA=0 S=001}}$ ALU $\xrightarrow{\text{MB=00}}$ DB $\xrightarrow{\text{GA=0}}$ A; COUT $\xrightarrow{\text{CP=0 ME=0}}$ CY;
7	`MOV A, #data`	T1: (PC) $\xrightarrow{\text{MC=00}}$ AB; (M) $\xrightarrow{\text{CRDX=0}}$ DB $\xrightarrow{\text{GA=0}}$ A; (PC)+1 $\xrightarrow{\text{PINC=0}}$ PC;
8	`MOV Ri, #data`	T1: (PC) $\xrightarrow{\text{MC=00}}$ AB; (M) $\xrightarrow{\text{CRDX=0}}$ DB $\xrightarrow{\text{WRE=0}}$ Ri; (PC)+1 $\xrightarrow{\text{PINC=0}}$ PC;
9	`LDA addr`	T1: (PC) $\xrightarrow{\text{MC=00}}$ AB; (M) $\xrightarrow{\text{CRDX=0}}$ DB $\xrightarrow{\text{GA1=0}}$ ADRH; (PC)+1 $\xrightarrow{\text{PINC=0}}$ PC; T2: (PC) $\xrightarrow{\text{MC=00}}$ AB; (M) $\xrightarrow{\text{CRDX=0}}$ DB $\xrightarrow{\text{GA2=0}}$ ADRL; (PC)+1 $\xrightarrow{\text{PINC=0}}$ PC; T3: (ADR) $\xrightarrow{\text{MC=01}}$ AB; (M) $\xrightarrow{\text{CRDX=0}}$ DB $\xrightarrow{\text{GA=0}}$ A;
10	`STA addr`	T1: (PC) $\xrightarrow{\text{MC=00}}$ AB; (M) $\xrightarrow{\text{CRDX=0}}$ DB $\xrightarrow{\text{GA1=0}}$ ADRH; (PC)+1 $\xrightarrow{\text{PINC=0}}$ PC; T2: (PC) $\xrightarrow{\text{MC=00}}$ AB; (M) $\xrightarrow{\text{CRDX=0}}$ DB $\xrightarrow{\text{GA2=0}}$ ADRL; (PC)+1 $\xrightarrow{\text{PINC=0}}$ PC; T3: (ADR) $\xrightarrow{\text{MC=01}}$ AB; (ACT) $\xrightarrow{\text{S=010}}$ ALU $\xrightarrow{\text{MB=00}}$ DB $\xrightarrow{\text{CWRX=0}}$ M;
11	`JC addr`	T1: (PC) $\xrightarrow{\text{MC=00}}$ AB; (M) $\xrightarrow{\text{CRDX=0}}$ DB $\xrightarrow{\text{GA1=0}}$ ADRH; (PC)+1 $\xrightarrow{\text{PINC=0}}$ PC; T2: (PC) $\xrightarrow{\text{MC=00}}$ AB; (M) $\xrightarrow{\text{CRDX=0}}$ DB $\xrightarrow{\text{GA2=0}}$ ADRL; (PC)+1 $\xrightarrow{\text{PINC=0}}$ PC; T3: (ADR) $\xrightarrow{\text{MC=01}}$ AB $\xrightarrow{\text{PLD2,1,0=001}}$ PC;
12	`JMP addr`	T1: (PC) $\xrightarrow{\text{MC=00}}$ AB; (M) $\xrightarrow{\text{CRDX=0}}$ DB $\xrightarrow{\text{GA1=0}}$ ADRH; (PC)+1 $\xrightarrow{\text{PINC=0}}$ PC; T2: (PC) $\xrightarrow{\text{MC=00}}$ AB; (M) $\xrightarrow{\text{CRDX=0}}$ DB $\xrightarrow{\text{GA2=0}}$ ADRL; (PC)+1 $\xrightarrow{\text{PINC=0}}$ PC; T3: (ADR) $\xrightarrow{\text{MC=01}}$ AB $\xrightarrow{\text{PLD2,1,0=101}}$ PC;
13	`JNKB addr`	T1: (PC) $\xrightarrow{\text{MC=00}}$ AB; (M) $\xrightarrow{\text{CRDX=0}}$ DB $\xrightarrow{\text{GA1=0}}$ ADRH; (PC)+1 $\xrightarrow{\text{PINC=0}}$ PC; T2: (PC) $\xrightarrow{\text{MC=00}}$ AB; (M) $\xrightarrow{\text{CRDX=0}}$ DB $\xrightarrow{\text{GA2=0}}$ ADRL; (PC)+1 $\xrightarrow{\text{PINC=0}}$ PC; T3: (ADR) $\xrightarrow{\text{MC=01}}$ AB $\xrightarrow{\text{PLD2,1,0=011}}$ PC;
14	`JNPB addr`	T1: (PC) $\xrightarrow{\text{MC=00}}$ AB; (M) $\xrightarrow{\text{CRDX=0}}$ DB $\xrightarrow{\text{GA1=0}}$ ADRH; (PC)+1 $\xrightarrow{\text{PINC=0}}$ PC; T2: (PC) $\xrightarrow{\text{MC=00}}$ AB; (M) $\xrightarrow{\text{CRDX=0}}$ DB $\xrightarrow{\text{GA2=0}}$ ADRL; (PC)+1 $\xrightarrow{\text{PINC=0}}$ PC; T3: (ADR) $\xrightarrow{\text{MC=01}}$ AB $\xrightarrow{\text{PLD2,1,0=100}}$ PC;
15	`CALL add`	T1: (PC) $\xrightarrow{\text{MC=00}}$ AB; (M) $\xrightarrow{\text{CRDX=0}}$ DB $\xrightarrow{\text{GA1=0}}$ ADRH; (PC)+1 $\xrightarrow{\text{PINC=0}}$ PC; T2: (PC) $\xrightarrow{\text{MC=00}}$ AB; (M) $\xrightarrow{\text{CRDX=0}}$ DB $\xrightarrow{\text{GA2=0}}$ ADRL; (PC)+1 $\xrightarrow{\text{PINC=0}}$ PC; T3: (SP) $\xrightarrow{\text{MC=10}}$ AB; (PCH) $\xrightarrow{\text{MB=01}}$ DB $\xrightarrow{\text{CWRX=0}}$ M; (SP)-1 $\xrightarrow{\text{SSP=01}}$ SP; T4: (SP) $\xrightarrow{\text{MC=10}}$ AB; (PCL) $\xrightarrow{\text{MB=10}}$ DB $\xrightarrow{\text{CWRX=0}}$ M; (SP)-1 $\xrightarrow{\text{SSP=01}}$ SP; T5: (ADR) $\xrightarrow{\text{MC=01}}$ AB $\xrightarrow{\text{PLD2,1,0=101}}$ PC;
16	`RET`	T1: (SP)+1 $\xrightarrow{\text{SSP=10}}$ SP; T2: (SP) $\xrightarrow{\text{MC=10}}$ AB; (M) $\xrightarrow{\text{CRDX=0}}$ DB $\xrightarrow{\text{GA2=0}}$ ADRL; (SP)+1 $\xrightarrow{\text{SSP=10}}$ SP; T3: (SP) $\xrightarrow{\text{MC=10}}$ AB; (M) $\xrightarrow{\text{CRDX=0}}$ DB $\xrightarrow{\text{GA1=0}}$ ADRH; T4: (ADR) $\xrightarrow{\text{MC=01}}$ AB $\xrightarrow{\text{PLD2,1,0=101}}$ PC;
17	`RSP`	T1: 7FFFH $\xrightarrow{\text{SSP=11}}$ SP;

编号	汇编代码	指令流程
18	`SUB A, @Ri`	T1: (Ri) $\xrightarrow{\text{MA=0, S=011}}$ ALU $\xrightarrow{\text{MB=00}}$ DB $\xrightarrow{\text{GA2=0}}$ ADRL; 7EH $\xrightarrow{\text{AHS=0}}$ ADRH; T2: (ADR) $\xrightarrow{\text{MC=01}}$ AB; (M) $\xrightarrow{\text{CRDX=0}}$ DB $\xrightarrow{\text{GT=0}}$ TMP; T3: (PC) $\xrightarrow{\text{MC=00}}$ AB; (ACT)-(TMP) $\xrightarrow{\text{MA=1 S=001}}$ ALU $\xrightarrow{\text{MB=00}}$ DB $\xrightarrow{\text{GA=0}}$ A; COUT $\xrightarrow{\text{CP=0 ME=0}}$ CY;
19	`ASR A`	T1: (A) $\xrightarrow{\text{MXE=0 ASR=0}}$ CY;
20	`CLR addr`	T1: (PC) $\xrightarrow{\text{MC=00}}$ AB; (M) $\xrightarrow{\text{CRDX=0}}$ DB $\xrightarrow{\text{GA1=0}}$ ADRH; (PC)+1 $\xrightarrow{\text{PINC=0}}$ PC; T2: (PC) $\xrightarrow{\text{MC=00}}$ AB; (M) $\xrightarrow{\text{CRDX=0}}$ DB $\xrightarrow{\text{GA2=0}}$ ADRL; (PC)+1 $\xrightarrow{\text{PINC=0}}$ PC; T3: (ADR) $\xrightarrow{\text{MC=01}}$ AB; 0 $\xrightarrow{\text{S=110}}$ ALU $\xrightarrow{\text{MB=00}}$ DB $\xrightarrow{\text{CWRX=0}}$ M;
21	`PUSH Ri`	T1: (SP) $\xrightarrow{\text{MC=10}}$ AB; (Ri) $\xrightarrow{\text{MA=0 S=011}}$ ALU $\xrightarrow{\text{MB=00}}$ DB $\xrightarrow{\text{CWRX=0}}$ M; (SP)-1 $\xrightarrow{\text{SSP=01}}$ SP;
22	`POP Ri`	T1: (PC) $\xrightarrow{\text{MC=00}}$ AB; (SP)+1 $\xrightarrow{\text{SSP=10}}$ SP; T2: (SP) $\xrightarrow{\text{MC=10}}$ AB; (M) $\xrightarrow{\text{CRDX=0}}$ DB $\xrightarrow{\text{WRE=0}}$ Ri;
23	`JZ addr`	T1: (PC) $\xrightarrow{\text{MC=00}}$ AB; (M) $\xrightarrow{\text{CRDX=0}}$ DB $\xrightarrow{\text{GA1=0}}$ ADRH; (PC)+1 $\xrightarrow{\text{PINC=0}}$ PC; T2: (PC) $\xrightarrow{\text{MC=00}}$ AB; (M) $\xrightarrow{\text{CRDX=0}}$ DB $\xrightarrow{\text{GA2=0}}$ ADRL; (PC)+1 $\xrightarrow{\text{PINC=0}}$ PC; T3: (ADR) $\xrightarrow{\text{MC=01}}$ AB $\xrightarrow{\text{PLD2,1,0=110}}$ PC;
24	`ADC A, Ri`	T1: (ACT)+(Ri) $\xrightarrow{\text{MA=0 S=000}}$ ALU $\xrightarrow{\text{MB=00}}$ DB $\xrightarrow{\text{GA=0}}$ A; COUT $\xrightarrow{\text{CWRX=0}}$ CY; T2: (A) $\xrightarrow{\text{PLD2,1,0=001 S=111}}$ ALU $\xrightarrow{\text{MB=00}}$ DB $\xrightarrow{\text{GA=0}}$ A; COUT $\xrightarrow{\text{CWRX=0}}$ CY;

注1: 除箭头上标注的信号, 其它信号皆为无效状态。

注2：MA、MB、MC等分别为MXA、MXB、MXC的缩写。

表格 3　　微操作信号表

部件	信号	功能	有效位
累加器 **A**	GA	A 功能	GA
		累加器 A 接数允许	0
		禁止（无操作）	1
	ASR	算术右移	0
暂存器 **TMP**	GT	TMP 接数允许	0
暂存器 **ACT**	GC	暂存器接数允许	0
寄存器 **Ri**	WRE	寄存器写入允许	0
MUXA (FB 的多路选择)	MXA	ALU 的第二个操作数多路开关选择	0: 寄存器内容送 ALU 1: 暂存器 TMP 送ALU
算数逻辑单元 **ALU**	S2, S1, S0	ALU 功能	S2 S1 S0
		F = A + B	0 0 0
		F = A − B	0 0 1
		F = A	0 1 0
		F = B	0 1 1
		F = ¬B	1 0 0
		备用	1 0 1
		F = 0	1 1 0
		F = A + 1	1 1 1
CY	MXE	进位位 CY 接数 COUT	0
		CY 接数移位末位输出	1
	CP	CY 接数允许	0
指令寄存器 **IR**	GI	指令寄存器 IR 接数允许	0
地址寄存器 **ADR**	GA1	高位地址寄存器 ADRH 接数允许	0
	AHS	高位地址寄存器 ADRH 置 7EH 允许	0
	GA2	低位地址寄存器 ADRL 接数允许	0
MUXC (地址 **AB** 多路开关)	MXC1,0	MUXC 功能	MC1 MC0
		PC 送至地址总线 AB	0 0
		ADR 送至地址总线 AB	0 1
		SP 送至地址总线 AB	1 0

部件	信号	功能	有效位
		备用	1 1
程序计数器 **PC**	PINC	PC 加 1 信号	0
	PRST	PC 清 0 信号	0
MUXD (控制 **PC** 接数的多路选择**)**	PLD2,1,0	PC 接数控制	PLD2 PLD1 PLD0
		禁止	0 0 0
		CY=1 转移	0 0 1
		ZY=0	0 1 0
		KB=0	0 1 1
		PB=0	1 0 0
		必转	1 0 1
		ZY=1	1 1 0
		备用	其余
微程序计数器 **MPC**	MPLD	MPC 接数允许	0
存储器 **M**	CRDX	读存储器	0
	CWRX	写存储器	0
堆栈 **SP**	SSP	SP 减 1 信号	01
		SP 增 1 信号	10
		SP 初始化为 7FFFH	11
MUXB (数据总线 **DB** 多路开关选择**)**	MXB1,0	MUXB 功能	MXDB1 MXDB2
		ALU 结束送至数据总线 DB	0 0
		PCH 送至数据总线 DB	0 1
		PCL 送至数据总线 DB	1 0
		三态	1 1

注:

① 图1中的寄存器选择信号（RC、RB、RA）不属于微指令编码范围，应直接来自指令码（I2、I1、I0）。

② 以上的各个微操作信号，除了已标明的，可统一取 0 有效。

表格 4　　微指令控制微操作对应表

微指令的控制位	微操作	有效状态	微操作功能描述
0	WRE	0	寄存器写入允许
1	GA	0	A 功能
2	ASR	0	累加器 A 算术右移
3	GC	0	暂存器接数允许
4	GT	0	TMP 接数允许
5	GI	0	指令寄存器 IR 接数允许
6	GA1	0	高位地址寄存器 ADRH 接数允许
7	AHS	0	高位地址寄存器 ADRH 置 7EH 允许
8	GA2	0	低位地址寄存器 ADRL 接数允许
9	OB	0	MUXB 送至 DB 允许
10	MXB0	X	数据总线 DB 多路开关选择
11	MXB1	X	
12	ZP	0	ZY 接数允许
13	CP	0	CY 接数允许
14	MXE	X	进位位 CY 接数 COUT 或移位输出
15	S0	X	ALU 功能选择
16	S1	X	
17	S2	X	
18	MXA	0	ALU 的第二个操作数多路开关选择
19	PLD0	X	控制 PC 接数的多路开关选择
20	PLD1	X	
21	PLD2	X	
22	PINC	0	PC 加 1 信号
23	SSP0	X	堆栈 SP 控制信号
24	SSP1	X	
25	MXC0	X	地址 AB 多路开关选择
26	MXC1	X	
27	MPLD	0	MPC 接数允许
28	CRDX	0	读存储器 M
29	CWRX	0	写存储器 M

对应的 VHDL 硬件描述如下：

```
CWRX<=MIR(29);
CRDX<=MIR(28);
MPLD<=MIR(27);
MXC(1)<=MIR(26); MXC(0)<=MIR(25);
SSP(1)<=MIR(24); SSP(0)<=MIR(23);
PINC<=MIR(22);
PLD(2)<=MIR(21); PLD(1)<=MIR(20); PLD(0)<=MIR(19); MXA<=MIR(18);
S(2)<=MIR(17);
S(1)<=MIR(16); S(0)<=MIR(15);
MXE<=MIR(14);
CP<=MIR(13);
ZP<=MIR(12);
MXB(1)<=MIR(11); MXB(0)<=MIR(10);
OB<=MIR(9); GA2<=MIR(8); AHS<=MIR(7); GA1<=MIR(6); GI<=MIR(5);
GT<=MIR(4); GC<=MIR(3); ASR<=MIR(2); GA<=MIR(1); WRE<=MIR(0);
```

表格 5　　微指令控制微操作对应表

指令助记符	微指令	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	信号	CWRX	CRDX	MPLD	MXC1	MXC0	SSP1	SSP0	PINC	PLD2	PLD1	PLD0	MXA	S2	S1	S0	MXE	CP	ZP	MXB1	MXB0	OB	GA2	AHS	GA1	GI	GT	GC	ASR	GA	WRE
有效状态		0	0	0	X	X	X	X	0	X	X	X	0	X	X	X	X	0	0	X	X	0	0	0	0	0	0	0	0	0	0
无效状态		1	1	1	X	X	X	X	1	X	X	X	1	X	X	X	X	1	1	X	X	1	1	1	1	1	1	1	1	1	1
取指令	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	1	0	1	1	1
	001-6																														
MOV A, Ri	007	1	1	1	0	0	0	0	1	0	0	0	0	0	1	1	0	1	1	0	0	0	1	1	1	1	1	1	1	0	1
	008	取指令																													
MOV Ri, A	00F	1	1	1	0	0	0	0	1	0	0	0	0	0	1	0	0	1	1	0	0	0	1	1	1	1	1	1	1	1	0
	010	取指令																													
MOV A, @Ri	017	1	1	1	0	0	0	0	1	0	0	0	0	0	1	1	0	1	1	0	0	0	0	0	1	1	1	1	1	1	1
	018	1	0	1	0	1	0	0	1	0	0	0	0	0	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	0	1
	019	取指令																													
MOV @Ri, A	01F	1	1	1	0	0	0	0	1	0	0	0	0	0	1	1	0	1	1	0	0	0	0	0	1	1	1	1	1	1	1
	020	0	1	1	0	1	0	0	1	0	0	0	0	0	1	0	0	1	1	0	0	0	1	1	1	1	1	1	1	1	1
	021	取指令																													
ADD A, Ri	027	1	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	0	1
	028	取指令																													
SUB A, Ri	02F	1	1	1	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1	1	1	1	1	1	1	0	1
	030	取指令																													

指令助记符	微指令	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	信号	CWRX	CRDX	MPLD	MXC1	MXC0	SSP1	SSP0	PINC	PLD2	PLD1	PLD0	MXA	S2	S1	S0	MXE	CP	ZP	MXB1	MXB0	OB	GA2	AHS	GA1	GI	GT	GC	ASR	GA	WRE
MOV A, #data	037	1	0	1	0	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	0	1
	038	取指令																													
MOV Ri, #data	03F	1	0	1	0	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0
	040	取指令																													
LDA addr	047	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	0	1	1	1	1	1	1
	048	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	1	1	1	1	1	1	1	1
	049	1	0	1	0	1	0	0	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	0	1
	04A	取指令																													
STA addr	04F	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	0	1	1	1	1	1	1
	050	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	1	1	1	1	1	1	1	1
	051	0	1	1	0	1	0	0	1	0	0	0	0	0	1	0	0	1	1	0	0	0	1	1	1	1	1	1	1	1	1
	052	取指令																													
JC addr	057	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	0	1	1	1	1	1	1
	058	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	1	1	1	1	1	1	1	1
	059	1	1	1	0	1	0	0	1	0	0	1	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	05A	取指令																													
JMP addr	05F	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	0	1	1	1	1	1	1
	060	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	1	1	1	1	1	1	1	1
	061	1	1	1	0	1	0	0	1	1	0	1	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	062	取指令																													
JNKB addr	067	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	0	1	1	1	1	1	1
	068	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	1	1	1	1	1	1	1	1
	069	1	1	1	0	1	0	0	1	0	1	1	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	06A	取指令																													
JNPB addr	06F	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	0	1	1	1	1	1	1
	070	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	1	1	1	1	1	1	1	1
	071	1	1	1	0	1	0	0	1	1	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	072	取指令																													
CALL	077	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	0	1	1	1	1	1	1
	078	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	1	1	1	1	1	1	1	1
	079	0	1	1	1	0	0	1	1	0	0	0	0	0	0	0	0	1	1	0	1	0	1	1	1	1	1	1	1	1	1

指令助记符	微指令	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	信号	CWRX	CRDX	MPLD	MXC1	MXC0	SSP1	SSP0	PINC	PLD2	PLD1	PLD0	MXA	S2	S1	S0	MXE	CP	ZP	MXB1	MXB0	OB	GA2	AHS	GA1	GI	GT	GC	ASR	GA	WRE
addr	07A	0	1	1	1	0	0	1	1	0	0	0	0	0	0	0	0	1	1	1	0	0	1	1	1	1	1	1	1	1	1
	07B	1	1	1	0	1	0	0	1	1	0	1	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	07C	取指令																													
RET	07F	1	1	1	0	0	1	0	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	080	1	0	1	1	0	1	0	1	0	0	0	0	0	0	0	0	1	1	1	1	1	0	1	1	1	1	1	1	1	1
	081	1	0	1	1	0	0	0	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	0	1	1	1	1	1	1
	082	1	1	1	0	1	0	0	1	1	0	1	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	083	取指令																													
RSP	087	1	1	1	0	0	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	088	取指令																													
SUB A, @Ri	08F	1	1	1	0	0	0	0	1	0	0	0	0	0	1	1	0	1	1	0	0	1	0	0	1	1	1	1	1	1	1
	090	1	0	1	0	1	0	0	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	0	1	1	1	1
	091	1	1	1	0	0	X	X	1	X	X	X	1	0	0	1	0	0	1	0	0	1	1	1	1	1	1	1	1	0	1
	092	取指令																													
ASR A	097	1	1	1	0	0	0	0	1	0	0	0	0	1	0	1	1	1	1	0	0	1	1	1	1	1	1	1	0	0	1
	098	取指令																													
CLR addr	09F	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	0	1	1	1	1	1	1
	0A0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	1	1	1	1	1	1	1	1
	0A1	0	1	1	0	1	0	0	1	0	0	0	0	1	1	0	0	1	1	0	0	1	1	1	1	1	1	1	1	1	1
	0A2	取指令																													
PUSH Ri	0A7	0	1	1	1	0	0	1	1	0	0	0	0	0	1	1	0	1	1	0	0	1	1	1	1	1	1	1	1	1	1
	0A8	取指令																													
POP Ri	0AF	1	1	1	0	0	1	0	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	0B0	1	0	1	1	0	0	0	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0
	0B1	取指令																													

指令助记符	微指令	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	信号	CWRX	CRDX	MPLD	MXC1	MXC0	SSP1	SSP0	PINC	PLD2	PLD1	PLD0	MXA	S2	S1	S0	MXE	CP	ZP	MXB1	MXB0	OB	GA2	AHS	GA1	GI	GT	GC	ASR	GA	WRE
JZ addr	0B7	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	0	1	1	1	1	1	1
	0B8	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	1	1	1	1	1	1	1	1
	0B9	1	1	1	0	1	0	0	1	1	1	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	0BA	取指令																													
ADC A, Ri	0BF	0	1	1	0	1	0	0	1	1	1	0	0	0	0	0	0	1	1	0	0	1	1	1	1	1	1	1	1	0	1
	0C0	0	1	1	0	1	0	0	1	0	0	1	0	1	1	1	0	1	1	0	0	1	1	1	1	1	1	1	1	0	1
	0C1	取指令																													

1.4 VHDL 设计

```
--时钟信号与复位信号

---时钟信号
pMCLK:
process(MCLK, CLK, RESET, RUN)
begin
    if((RUN = '0') or (RESET = '0')) then MCLK <= '0';
        elsif(CLK'event and CLK = '0') then MCLK <= not MCLK;
    end if;
end process pMCLK;

MPCK    <= not MCLK and CLK;
MICK    <= not MPCK;

WRC     <= MCLK;        --寄存器 Ri 时钟
PCK     <= MCLK;        --PC 时钟
CA      <= MCLK;        --A 时钟
CT      <= MCLK;        --TMP 时钟
CC      <= MCLK;        --ACT 时钟
CCI     <= MCLK;        --IR 时钟
CA1     <= MCLK;        --ADRH 时钟
CA2     <= MCLK;        --ADRL 时钟
CCK     <= MCLK;        --CY 时钟
SCK     <= MCLK;        --SP 时钟
ZCK     <= MCLK;        --ZY 时钟


----复位信号
PREST   <= RESET;

pMCLR:
process(MCLK, RESET)
begin
    if(RESET = '0') then MCLR <= '0';
        elsif(MCLK'event and MCLK = '1') then MCLR <= RUN;
    end if;
```

```
end process pMCLR;

CWR     <= CWRX or not MCLK;
CRD     <= CRDX or not MCLK;

MRD     <= CRD or AB(15);
MWR     <= CWR or AB(15) or not CLK;
IOW     <= not AB(15) or not AB(1) or CWR or not CLK;
IOR     <= not AB(15) or not AB(0) or CRD;

--功能部件

----MPC的定义
pMPC:
process(MPLD, MPCK, MCLR)
begin
    if(MCLR = '0') then MPC <= "0000000000";
        elsif(MPCK'event and MPCK = '1') then
            if(MPLD = '0') then MPC <= MD;
                else MPC <= MPC + 1;
            end if;
    end if;
end process pMPC;

CI(9 downto 0) <= MPC;

----为程序计数器入口MD的定义
MD(0)           <= '1';
MD(1)           <= '1';
MD(2)           <= '1';
MD(7 downto 3)  <= IR(7 downto 3);
MD(9 downto 8)  <= "00";

----MIR的定义
pMIR:
process(MICK)
begin
    if(MICK'event and MICK = '1') then
        MIR <= CO;
    end if;
end process pMIR;

CWRX    <= MIR(29);
CRDX    <= MIR(28);
MPLD    <= MIR(27);
MXC(1)  <= MIR(26);
MXC(0)  <= MIR(25);
SSP(1)  <= MIR(24);
SSP(0)  <= MIR(23);
PINC    <= MIR(22);
PLD(2)  <= MIR(21);
PLD(1)  <= MIR(20);
PLD(0)  <= MIR(19);
MXA     <= MIR(18);
S(2)    <= MIR(17);
S(1)    <= MIR(16);
S(0)    <= MIR(15);
MXE     <= MIR(14);
CP      <= MIR(13);
ZP      <= MIR(12);
MXB(1)  <= MIR(11);
MXB(0)  <= MIR(10);
OB      <= MIR(9);
```

```
GA2     <= MIR(8);
AHS     <= MIR(7);
GA1     <= MIR(6);
GI      <= MIR(5);
GT      <= MIR(4);
GC      <= MIR(3);
ASR     <= MIR(2);
GA      <= MIR(1);
WRE     <= MIR(0);

----寄存器Ri的定义
pRi:
process(WRC, WRE, RS)
begin
    if (WRC'event and WRC ='0') then
        if (WRE = '0') then
            case RS is
                when "000" => R0 <= DBL;
                when "001" => R1 <= DBL;
                when "010" => R2 <= DBL;
                when "011" => R3 <= DBL;
                when "100" => R4 <= DBL;
                when "101" => R5 <= DBL;
                when "110" => R6 <= DBL;
                when "111" => R7 <= DBL;
            end case;
        end if;
    end if;
end process pRi;

----寄存器输出ROUT的定义
case RS is
    when "000" => ROUT <= R0;
    when "001" => ROUT <= R1;
    when "010" => ROUT <= R2;
    when "011" => ROUT <= R3;
    when "100" => ROUT <= R4;
    when "101" => ROUT <= R5;
    when "110" => ROUT <= R6;
    when "111" => ROUT <= R7;
end case;

----MUXA的定义
if(MXA = '0') then
    FB <= ROUT;
else
    FB <= TMP;
end if;

----MUXB的定义
case MXB is
    when "00" => DB <= FF(7 downto 0);
    when "01" => DB <= PC(15 downto 8);
    when "10" => DB <= PC(7 downto 0);
    when others => DB <= ROUT;
end case;

----MUXC的定义
case MXC is
    when "00" => AB <= PC;
    when "01" => AB <= ADRH&ADRL;
    when "10" => AB <= SP;
    when others => NULL;
```

```
end case;

----MUXD的定义
case PLD is
    when "000" => PCADD <= '0';
    when "001" => PCADD <= CY;
    when "010" => PCADD <= not ZY;
    when "011" => PCADD <= not KRIX;
    when "100" => PCADD <= not PRIX;
    when "101" => PCADD <='1';
    when "110" => PCADD <= ZY;
    when others => NULL;
end case;

----TMP的定义
pTMP:
process(CT, GT)
begin
    if(CT'event and CT = '0') then
        if(GT = '0') then
            TMP <= DB;
        end if;
    end if;
end process pTMP;

----A的定义
pA1:
process(CA, GA)
begin
    if(CA'event and CA = '0') then
        if(GA = '0') then
            A <= DB;
        end if;
    end if;
end process pA1;

pA2:
process(CA, ASR)
variable POOPOO:std_logic_vector(7 downto 0);
begin
    if(CA'event and CA = '0') then
        if(ASR = '0' and MXE = '0') then
            POOPOO := A;
            CYA  <= POOPOO(0);      --移出的末位
            A   <= POOPOO(7) & POOPOO(7 downto 1);
        end if;
    end if;
end pA2;

----MUXE的定义
if(MXE = '0') then
    CY <= FF(8);
else
    CY <= CYA;
end if;


----ACT的定义
pACT:
process(CC, GC)
begin
    if(CC'event and CC = '0') then
        if(GC = '0') then
```

```
            ACT <= DB;
        end if;
    end if;
end process pACT;

FA <= '0'&ACT; --扩展成九位
FB <= '0'&TMP;

----ALU的定义
case S is
    when "000" => FF <= FA + FB;
    when "001" => FF <= FA - FB;
    when "010" => FF <= FA;
    when "011" => FF <= FB;
    when "100" => FF <= not FB;
    when "101" => FF <= "000000000";
    when "110" => FF <= "000000000";
    when "111" => FF <= A + '1';
end case;

COUT <= FF(8);

----CY的定义
pCY:
process(CCK, CP, FF)
begin
    if(CCK'event and CCK = '0') then
        if(CP = '0') then
            CY <= FF(8);
        end if;
    end if;
end process pCY;

----ZY的定义
pZY:
process(ZCK, ZP, FF)
begin
    if(ZCK'event and ZCK = '0') then
        if(ZP = '0') then
            if(FF = "000000000") then
                ZY <= '1';
            else
                ZY <= '0';
            end if;
        end if;
    end if;
end process pZY;

----PC的定义
PLDR <= '0' when PLD = "000" else
        CY when PLD = "001" else
        not ZY when PLD = "010" else
        NVY when PLD = "011" else
        '1' when PLD = "100" else
        '0';
TREL(7 downto 0) <= TMP;
TREL(15 downto 8) <= "00000000" when TMP(7) = '0' else "11111111";

pPC:
process(PCK, PRST, PLDR)
begin
    if(PRST = '0') then
        PC <= "0000000000000000";
```

```
    elsif(PCK'event and PCK = '0') then
        if(PLDR = '1') then PC <= AB;
        elsif(PINC = '0') then
            PC <= PC + 1;
        end if;
    end if;
end process pPC;

----IR的定义
pIR:
process(CCI, GI, DB)
begin
    if(CCI'event and CCI = '0') then
        if(GI = '0') then
            IR <= DB;
        end if;
    end if;
end process pIR;

----ADRH的定义
pADRH:
process(CA1, GA1, AHS, DB)
begin
    if(CA1'event and CA1 = '0') then
        if(AHS = '0') then ADRH <= "01111111";
            elsif(GA1 = '0') then
                ADRH <= DB;
        end if;
    end if;
end process pADRH;

----ADRL的定义
pADRL:
process(CA2, GA2, DB)
begin
    if(CA2'event and CA2 = '0') then
        if(GA2 = '0') then
            ADRL <= DB;
        end if;
    end if;
end process pADRL;

----SP的定义
pSP:
process(SCK, SSP)
begin
    if(SCK'event and SCK = '0') then
        case SSP is
            when "01" => SP <= SP - 1;
            when "10" => SP <= SP + 1;
            when "11" => SP <= "0111111111111111";
            when others => SP <= "0000000000000000";
        end case;
    end if;
end process pSP;
```