CPU 设计: 8008

luoyin

目录

Chapter 1

微程序设计

1.1 子程序设计

- IF (Instruction Fetch): T1-T2-T3 (PCI)
- MR (Memory Read): T1-T2-T3 (PCR)
- MW (Memory Write): T1-T2-T3 (PCW)
- RR (Register Read): T4
- RW (Register Write): T5
- PCU (PC Update): T4-T5
- IOR (I/O Read): T3-T4-T5

1.2 指令组成

表 1.1: 指令组成

指令	指令码	组成
Lrr	11DDDSSS	PCO(PCL-PCH)-IF-rR-rW
LrM	11DDD111	PCO(PCL-PCH)-IF-MA(rLO-rMO)-MR-X1-rW
LMr	11111SSS	PCO(PCL-PCH)-IF-rR-MA(rLO-rMO)-MW
LrI	00DDD110	PCO(PCL-PCH)-IF-PCO(PCL-PCH)-IMMb-X1-rW
INr/DCr	00DDD00V	PCO(PCL-PCH)-IF-X1-rW
ALU OP r	10PPPSSS	PCO(PCL-PCH)-IF-rR-rW
ALU OP M	10PPP111	PCO(PCL-PCH)-IF-MA(rLO-rMO)-MR-X1-rW
ALU OP I	00PPP100	PCO(PCL-PCH)-IF-PCO(PCL-PCH)-IMMb-X1-rW
ROT	000VV010	PCO(PCL-PCH)-IF-X1-rW
JMP	01XXX100	PCO(PCL-PCH)-IF-PCO(PCL-PCH)-IMMb-PCO(PCL-PCH)-IMMa1-PCU(PCHU-PCLU)
JFc/JTc	01VCC000	PCO(PCL-PCH)-IF-PCO(PCL-PCH)-IMMb-PCO(PCL-PCH)-IMMa2-PCU(PCHU-PCLU)
CAL	01XXX110	PCO(PCL-PCH)-IF-PCO(PCL-PCH)-IMMb-PCO(PCL-PCH)-IMMa3-PCU(PCHU-PCLU)
CFc/CTc	01VCC010	PCO(PCL-PCH)-IF-PCO(PCL-PCH)-IMMb-PCO(PCL-PCH)-IMMa4-PCU(PCHU-PCLU)
RET	00XXX111	PCO(PCL-PCH)-IF-POP-X2
RFc/RTc	00VCC011	PCO(PCL-PCH)-IF-POPc(c)-X2
INP	0100MMM1	PCO(PCL-PCH)-IF-IO(rAO-rBO)-IOb-CO-rW
OUT	01RRMMM1	PCO(PCL-PCH)-IF-IO(rAO-rBO)-X0
HLT	0000000X	PCO(PCL-PCH)-IF
HLT	11111111	PCO(PCL-PCH)-IF

1.3 指令执行状态变化

1.4 微指令设计

1.4.1 微指令组成

- 状态码 (3 位)
- 寄存器组操作 (2 位): 输出使能, 写使能
-

1.4.2 指令分类

- $D_7D_6 = 00$: 特殊指令
  - $D_2D_1D_0 = 000$: HLT
  - $D_2D_1D_0 = 001$: HLT
  - $D_2D_1D_0 = 010$: ROT
  - $D_2D_1D_0 = 011$: RFc/RTc
  - $D_2D_1D_0 = 100$: ALU OP I
  - $D_2D_1D_0 = 101$: RST
  - $D_2D_1D_0 = 110$: LrM/LrI
  - $D_2D_1D_0 = 111$: RET
- $D_7D_6 = 01$: 跳转指令
  - $D_2D_1D_0 = 000$: JFc/JTc
  - $D_2D_1D_0 = 010$: CFc/CTc
  - $D_2D_1D_0 = 100$: JMP
  - $D_2D_1D_0 = 110$: CAL
  - $D_2D_1D_0 = XX1$: INP/OUT
- $D_7D_6 = 10$: 算术指令
- $D_7D_6 = 11$: 寄存器指令

1.4.3 微程序分类与跳转

使用 D_7D_6 进行一次分组, 使用 $D_2D_1D_0$ 进行二次分组

1.4.4 微指令转移

微指令转移按照如下计算规则:

$$A_i = \mu A_i + \sum P_i^I I_i + \sum P_i^S S_i + \sum P_i^C C_i \tag{1.1}$$

其中, μA_i 为微指令中的下一指令段, P 为微指令中的控制段, 按作用类型不同分为指令控制段 P_i^I, 状态控制段 P_i^S, 和条件控制段 P_i^C, I_i 为指令寄存器的位段, S_i 为状态寄存器的位段, C_i 为条件判定寄存器的位段.

1.4.5 微指令转移方式

- 直接转移: 微指令中的控制段均为 0, 微指令运行下一指令直接由微指令中的 μA_i 段决定.
- 按指令转移: 微指令中的指令控制段 P_i^I 不为 0, 此时, 微指令中的 μA_i 段决定跳转时的基址, $\sum P_i^I I_i$ 决定偏移量.
- 按状态转移: 微指令中的状态控制段 P_i^S 不为 0, 此时, 微指令中的 μA_i 段决定跳转时的基址, $\sum P_i^S S_i$ 决定偏移量.
- 按条件转移: 微指令中的条件控制段 P_i^C 不为 0, 此时, 微指令中的 μA_i 段决定跳转时的基址, $\sum P_i^C C_i$ 决定偏移量.
- 复合转移: 微指令中的 μA_i 段决定跳转时的基址, 结合指令控制段 P_i^I, 状态控制段 P_i^S, 和条件控制段 P_i^C 综合决定偏移量.

1.4.6 微指令转移方式

表 1.2: 微程序表

地址	微指令	状态	功能	下一微指令	下一状态	转移类型
0	PCL	T1	PCL 输出	PCH	T2	直接转移
1	PCH	T2	PCH 输出	IF, IMMa, IMMb	T3, WAIT	状态转移
2	IF	T3	DATA to IR and regB	rR, rLO, PCL, POP, POPc, rAO, X1	T4, T1, HLT	指令转移, 状态转移, 条件转移
	rR	T4	reg Read	rW, rLO	T5, T1, HLT	指令转移, 状态转移
	rW	T5	reg Write	PCL	T1, INT	指令转移, 状态转移
	rLO	T1	reg L Out	rHO	T2	直接转移
	rHO	T2	reg H Out	MR, MW	T3, WAIT	指令转移, 状态转移
	MR	T3	Memory Read	X1	T4	直接转移
	MW	T3	Memory Write	PCL	T1, INT	状态转移

1.4.7 跳转设计

1.4.7.1 IF 跳出

跳出指向

- rR: Lrr+LMr (11VVVSSS), ALU op r (10PPPSSS), 合并 (1XXXXSSS, SSS<>111)
- rLO: LrM (11DDD111, DDD<>111), ALU op M (10PPP111)
- rAO: INP+OUT (01XXXXX1)
- POP: RETURN (00XXXX11)
- PCL: JUMP, CALL (01XXXXX0)
- PCL(next): HLT, INT, NORMAL
- X1: INr/DCr

1.4.7.2 指令前缀 00

指令通过 $D_2D_1D_0$ 进行分类

- 000: HLT/INr (通过 $D_5D_4D_3$ 进行分类). IF-STOP/IF-X1
- 001: HLT/DCr (通过 $D_5D_4D_3$ 进行分类). IF-STOP/IF-X1
- 010: ROT (RLC, RRC, RAL, RAR, 通过 $D_5D_4D_3$ 进行分类). IF-X1
- 011: RFc/RTc. IF-POP
- 100: ALU op I. IF-PCL
- 101: RST. IF-PCLU2-PCHU2
- 110: LrI/LMI (通过 $D_5D_4D_3$ 进行分类). IF-PCL
- 111: RET. IF-POP

将微程序地址 10000-10111 与上面 8 个指令对应. 指令跳转表达式为

$$J_C = \bar{D}_7\bar{D}_6(\overline{\bar{D}_2D_1D0} + \bar{D}_2D_1D_0J) \tag{1.2}$$

$$A_4 = J_C1 \tag{1.3}$$

$$A_3 = J_CP_0(D_7 + D_6) \tag{1.4}$$

$$A_2 = J_C(\mu A_2 + P_0\bar{D}_7\bar{D}_6I_2) \tag{1.5}$$

$$A_1 = J_C(\mu A_1 + P_0\bar{D}_7\bar{D}_6I_1) \tag{1.6}$$

$$A_0 = J_C(\mu A_0 + P_0\bar{D}_7\bar{D}_6I_0) \tag{1.7}$$

1.4.7.3 指令前缀 01

指令通过 $D_2D_1D_0$ 进行分类

- 000: JFc/JTc
- 010: CFc/CTc
- 100: JMP
- 110: CAL
- XX1: INP/OUT

将微程序地址 11000-11111 与上面 8 个指令对应. 指令跳转表达式为

$$\begin{aligned}
J_C &= \bar{D}_7D_6(\overline{\bar{D}_2\bar{D}_0} + \bar{D}_2\bar{D}_0J) && (1.8)\\
A_4 &= \bar{J}_C1 && (1.9)\\
A_3 &= \bar{J}_C(P_0\bar{D}_7D_6) && (1.10)\\
A_2 &= \bar{J}_C(\mu A_2 + P_1\bar{D}_7D_6I_0) && (1.11)\\
A_1 &= \bar{J}_C(\mu A_2 + P_1\bar{D}_7D_6I_2) && (1.12)\\
A_0 &= \bar{J}_C(\mu A_2 + P_1\bar{D}_7D_6I_1\bar{I}_0 + P_1\bar{D}_7D_6\bar{I}_5\bar{I}_4I_0) && (1.13)
\end{aligned}$$

1.4.7.4 指令前缀 10

指令通过 $D_2D_1D_0$ 进行分类

- SSS: ALU op r
- 111: ALU op M

将微程序地址 100000-100001 与上面 2 个指令对应. 指令跳转表达式为

$$\begin{aligned}
A_5 &= D_7\bar{D}_6 && (1.14)\\
A_4 &= \overline{D_7\bar{D}_6} && (1.15)\\
A_3 &= \overline{D_7\bar{D}_6} && (1.16)\\
A_2 &= \overline{D_7\bar{D}_6} && (1.17)\\
A_1 &= \overline{D_7\bar{D}_6} && (1.18)\\
A_0 &= D_7\bar{D}_6I_2I_1I_0 && (1.19)
\end{aligned}$$

1.4.7.5 指令前缀 11

指令通过 $D_5D_4D_3D_2D_1D_0$ 进行分类

- DDDSSS: Lrr
- DDD111: LrM
- 111SSS: LMr
- 111111: HLT

将微程序地址 100100-100111 与上面 4 个指令对应. 指令跳转表达式为

$$A_5 = D_7 D_6 \tag{1.20}$$
$$A_4 = \overline{D_7 D_6} \tag{1.21}$$
$$A_3 = \overline{D_7 D_6} \tag{1.22}$$
$$A_2 = D_7 D_6 \tag{1.23}$$
$$A_1 = D_7 D_6 I_5 I_4 I_3 \tag{1.24}$$
$$A_0 = D_7 D_6 I_2 I_1 I_0 \tag{1.25}$$

1.4.7.6 条件跳转

适用指令: JFc/JTc, CFc/CTc, RFc/RTc, 引入条件判定变量 J, 当条件成立时 $J = 1$, 否则 $J = 0$, 指令跳转表达式为

$$A_i = J() \tag{1.26}$$

1.4.8 微指令组合逻辑

- srcM: $D_2 D_1 D_0$
- dstM: $D_5 D_4 D_3$
- JUMP:

1.4.9 微指令表

表 1.3: 微指令表

地址	指令	微指令	S			P			μA				
			2	1	0	2	1	0	4	3	2	1	0
000000		PCL	0	1	0	0	0	0	0	0	0	0	1
000001		PCH	1	0	0	0	0	0	x	x	x	x	x
000010		IF	0	0	1	x	x	x	0	1	x	x	x
001000		rR	1	1	1	x	x	x	x	x	x	x	x
001001		POP	1	1	1	0	0	0	x	x	x	x	x
001010		X1	1	1	1	0	0	0	x	x	x	x	x
001100		rLO	0	1	0	0	0	0	x	x	x	x	x
001101		rAO	0	1	0	0	0	0	x	x	x	x	x
001110		PCL2	0	1	0	0	0	0	x	x	x	x	x
010000	INr		1	1	1				x	x	x	x	x
010001	DCr		1	1	1				x	x	x	x	x
010010	ROT		1	1	1				x	x	x	x	x
010011	RETc		1	1	1				x	x	x	x	x
010100	ALU op I		1	1	1				x	x	x	x	x
010101	RST		1	1	1				x	x	x	x	x
010110	LrI/LMI		1	1	1				x	x	x	x	x
010111	RET		1	1	1				x	x	x	x	x

表 1.3: 微指令表 (续)

地址	指令	微指令	S			P			μA				
			2	1	0	2	1	0	4	3	2	1	0
011000	JMPc		1	1	1				x	x	x	x	x
011001	CALc		1	1	1				x	x	x	x	x
011010	JMP		1	1	1				x	x	x	x	x
011011	CAL		1	1	1				x	x	x	x	x
011100	INP		1	1	1				x	x	x	x	x
011101	OUT		1	1	1				x	x	x	x	x
100000	ALU op r		1	1	1				x	x	x	x	x
100001	ALU op M		1	1	1				x	x	x	x	x
100100	Lrr		1	1	1				x	x	x	x	x
100101	LrM		1	1	1				x	x	x	x	x
100110	LMr		1	1	1				x	x	x	x	x
100111	HLT		1	1	1				x	x	x	x	x

1.5 微指令设计

1.5.1 指令译码

$$
\begin{aligned}
Lrr &= D_7D_6\overline{D_5D_4D_3}\ \overline{D_2D_1D_0} && (1.27)\\
LrM &= D_7D_6\overline{D_5D_4D_3}D_2D_1D_0 && (1.28)\\
LMr &= D_7D_6D_5D_4D_3\overline{D_2D_1D_0} && (1.29)\\
LrI &= \bar{D}_7\bar{D}_6\overline{D_5D_4D_3}D_2D_1\bar{D}_0 && (1.30)\\
LMI &= \bar{D}_7\bar{D}_6D_5D_4D_3D_2D_1\bar{D}_0 && (1.31)\\
INr &= \bar{D}_7\bar{D}_6\overline{D_5D_4D_3}\bar{D}_2\bar{D}_1\bar{D}_0 && (1.32)\\
DCr &= \bar{D}_7\bar{D}_6\overline{D_5D_4D_3}\bar{D}_2\bar{D}_1D_0 && (1.33)\\
ALUopR &= D_7\bar{D}_6\overline{D_2D_1D_0} && (1.34)\\
ALUopM &= D_7\bar{D}_6D_2D_1D_0 && (1.35)\\
ALUopI &= \bar{D}_7\bar{D}_6D_2\bar{D}_1\bar{D}_0 && (1.36)\\
ROT &= \bar{D}_7\bar{D}_6\bar{D}_2D_1\bar{D}_0 && (1.37)\\
JMP &= \bar{D}_7D_6D_2\bar{D}_1\bar{D}_0 && (1.38)\\
JMPc &= \bar{D}_7D_6\bar{D}_2\bar{D}_1\bar{D}_0 && (1.39)\\
CAL &= \bar{D}_7D_6D_2D_1\bar{D}_0 && (1.40)\\
CALc &= \bar{D}_7D_6\bar{D}_2D_1\bar{D}_0 && (1.41)\\
RET &= \bar{D}_7\bar{D}_6D_2D_1D_0 && (1.42)\\
RETc &= \bar{D}_7\bar{D}_6\bar{D}_2D_1D_0 && (1.43)\\
RST &= \bar{D}_7\bar{D}_6D_2\bar{D}_1D_0 && (1.44)\\
INP &= \bar{D}_7D_6\bar{D}_5\bar{D}_4D_0 && (1.45)\\
OUT &= \bar{D}_7D_6(\bar{D}_5+\bar{D}_4)D_0 && (1.46)\\
HLT &= \bar{D}_7\bar{D}_6\bar{D}_5\bar{D}_4\bar{D}_3\bar{D}_2\bar{D}_1+D_7D_6D_5D_4D_3D_2D_1D_0 && (1.47)
\end{aligned}
$$

将指令分段

$$
\begin{aligned}
M_s &= D_2D_1D_0 && (1.48)\\
M_d &= D_5D_4D_3 && (1.49)
\end{aligned}
$$

指令编码公式如下

$$I_{Lrr} = D_7D_6\overline{M_d}\,\overline{M_s} \tag{1.50}$$
$$I_{LrM} = D_7D_6\overline{M_d}M_s \tag{1.51}$$
$$I_{LMr} = D_7D_6M_d\overline{M_s} \tag{1.52}$$
$$I_{LrI} = \bar{D}_7\bar{D}_6\overline{M_d}D_2D_1\bar{D}_0 \tag{1.53}$$
$$I_{LMI} = \bar{D}_7\bar{D}_6M_dD_2D_1\bar{D}_0 \tag{1.54}$$
$$I_{INr} = \bar{D}_7\bar{D}_6\overline{M_d}\bar{D}_2\bar{D}_1\bar{D}_0 \tag{1.55}$$
$$I_{DCr} = \bar{D}_7\bar{D}_6\overline{M_d}\bar{D}_2\bar{D}_1D_0 \tag{1.56}$$
$$I_{ALUopR} = D_7\bar{D}_6\overline{M_s} \tag{1.57}$$
$$I_{ALUopM} = D_7\bar{D}_6M_s \tag{1.58}$$
$$I_{ALUopI} = \bar{D}_7\bar{D}_6D_2\bar{D}_1\bar{D}_0 \tag{1.59}$$
$$I_{ROT} = \bar{D}_7\bar{D}_6\bar{D}_2D_1\bar{D}_0 \tag{1.60}$$
$$I_{JMP} = \bar{D}_7D_6D_2\bar{D}_1\bar{D}_0 \tag{1.61}$$
$$I_{JMPc} = \bar{D}_7D_6\bar{D}_2\bar{D}_1\bar{D}_0 \tag{1.62}$$
$$I_{CAL} = \bar{D}_7D_6D_2D_1\bar{D}_0 \tag{1.63}$$
$$I_{CALc} = \bar{D}_7D_6\bar{D}_2D_1\bar{D}_0 \tag{1.64}$$
$$I_{RET} = \bar{D}_7\bar{D}_6D_2D_1D_0 \tag{1.65}$$
$$I_{RETc} = \bar{D}_7\bar{D}_6\bar{D}_2D_1D_0 \tag{1.66}$$
$$I_{RST} = \bar{D}_7\bar{D}_6D_2\bar{D}_1D_0 \tag{1.67}$$
$$I_{INP} = \bar{D}_7D_6\bar{D}_5\bar{D}_4D_0 \tag{1.68}$$
$$I_{OUT} = \bar{D}_7D_6(D_5+D_4)D_0 \tag{1.69}$$
$$I_{HLT} = \bar{D}_7\bar{D}_6\bar{D}_5\bar{D}_4\bar{D}_3\bar{D}_2\bar{D}_1 + D_7D_6M_dM_s \tag{1.70}$$

1.5.2 IF 转出设计

IF 转出到微地址 01000-01111, 转出编码

$$
\begin{aligned}
F_0 &= I_{HLT} \\
F_1 &= I_{Lrr} + I_{ALUopR} + I_{LMr} = D_7\bar{M}_s \\
F_2 &= I_{JMP} + I_{JMPc} + I_{CAL} + I_{CALc} + I_{LrI} + I_{LMI} + I_{ALUopI} = \bar{D}_7\bar{D}_0(D_6 + \bar{D}_6D_2) \\
F_3 &= I_{LrM} + I_{ALUopM} = D_7M_s(D_6\bar{M}_d + \bar{D}_6) \\
F_4 &= I_{INP} + I_{OUT} = D_7\bar{D}_6D_0 \\
F_5 &= I_{INr} + I_{DCr} + I_{ROT} = \bar{D}_7\bar{D}_6\bar{D}_2(\bar{M}_d\bar{D}_1 + D_1\bar{D}_0) \\
F_6 &= I_{RET} + I_{RETc} = \bar{D}_7\bar{D}_6D_1D_0 \\
F_7 &= I_{RST} = \bar{D}_7\bar{D}_6D_2\bar{D}_1D_0
\end{aligned}
$$

使用 8-3 编码器

$$
\begin{aligned}
Y_0^0 &= F_1 + F_3 + F_5 + F_7 \\
Y_1^0 &= F_2 + F_3 + F_6 + F_7 \\
Y_2^0 &= F_4 + F_5 + F_6 + F_7
\end{aligned}
$$

微地址转换公式如下

$$
\begin{aligned}
A_4 &= \mu A_4 \\
A_3 &= \mu A_3 \\
A_2 &= P_0(Y_2^0) + \mu A_2 \\
A_1 &= P_0(Y_1^0) + \mu A_1 \\
A_0 &= P_0(Y_0^0) + \mu A_0
\end{aligned}
$$

1.5.3 1 级转入设计

微地址编码

- 10000: rW
- 10001: ALU
- 10010: rL
- 10011: PCL
- 10100: IFb
- 10101: rB
- 10110: X3

Chapter 2

CPU 模块设计

2.1 模块组成

- Reg
- RegBank
- Stack

Chapter 3

微程序设计 2

3.1 指令跳转

$$
\begin{aligned}
A_5 &= \mu A_5 && (3.1)\\
A_4 &= \mu A_4 + P_4 \cdot D_7 && (3.2)\\
A_3 &= \mu A_3 + P_4 \cdot D_6 && (3.3)\\
A_2 &= \mu A_2 + P_0 \cdot D_2 && (3.4)\\
A_1 &= \mu A_1 + P_0 \cdot D_1 + P_3 \cdot D_5 D_4 D_3 && (3.5)\\
A_0 &= \mu A_0 + P_0 \cdot D_0 + P_1 \cdot D_0 + P_2 \cdot D_2 D_1 D_0 + P_3 \cdot D_2 D_1 D_0 && (3.6)\\
&+ P_5 \cdot D_5 D_4 D_3 + P_6 \cdot (D_5 + D_4) + P_7 \cdot D_4 && (3.7)
\end{aligned}
$$

简化表达式, 如下

$$
\begin{aligned}
A_5 &= \mu A_5 && (3.8)\\
A_4 &= \mu A_4 + P_6 \cdot D_7 && (3.9)\\
A_3 &= \mu A_3 + P_6 \cdot D_6 && (3.10)\\
A_2 &= \mu A_2 + P_0 \cdot D_2 && (3.11)\\
A_1 &= \mu A_1 + P_0 \cdot D_1 + P_5 \cdot D_5 D_4 D_3 && (3.12)\\
A_0 &= \mu A_0 + P_0 \cdot D_0 + P_1 \cdot D_2 D_1 D_0 + P_2 \cdot D_5 D_4 D_3 + P_3 \cdot (D_5 + D_4) + P_4 \cdot D_4 + P_7 \cdot D_1 && (3.13)
\end{aligned}
$$

进一步简化表达式, 如下

$$
\begin{aligned}
A_5 &= \mu A_5 && (3.14)\\
A_4 &= \mu A_4 + P_0 \cdot D_7 && (3.15)\\
A_3 &= \mu A_3 + P_0 \cdot D_6 && (3.16)\\
A_2 &= \mu A_2 + P_1 \cdot D_2 && (3.17)\\
A_1 &= \mu A_1 + P_1 \cdot D_1 + P_2 \cdot D_5 D_4 D_3 && (3.18)\\
A_0 &= \mu A_0 + P_1 \cdot D_0 + P_3 \cdot D_2 D_1 D_0 + P_3 \cdot D_5 + P_4 \cdot D_4 + P_5 \cdot D_1 && (3.19)
\end{aligned}
$$

3.1.1 分支设计

$$
\begin{aligned}
S_M &= D_2D_1D_0 && (3.20)\\
D_M &= D_5D_4D_3 && (3.21)\\
A_5 &= \mu A_5 && (3.22)\\
A_4 &= \mu A_4 + Q_0D_7 && (3.23)\\
A_3 &= \mu A_3 + Q_0D_6 && (3.24)\\
A_2 &= \mu A_2 + Q_0(\bar{D}_7\bar{D}_6D_2) && (3.25)\\
A_1 &= \mu A_1 + Q_0(\bar{D}_7\bar{D}_6D_1 + D_7D_6D_M) && (3.26)\\
A_0 &= \mu A_0 + Q_0(\bar{D}_7D_0 + D_7S_M) + Q_1D_M + Q_2(D_5 + D_4) && (3.27)
\end{aligned}
$$

表 3.1: 微指令表

微地址	微子程序	微指令	Q			μA					
			2	1	0	5	4	3	2	1	0
000000	-	X1	0	0	0	0	1	0	1	1	0
000001	-	X2	0	0	0	0	1	0	1	1	1
000010	-	X3	0	0	0	0	1	1	1	1	1
000011	-	POP(c)	0	0	0	1	0	0	0	0	0
000100	P_IMMB1	PCL_BUS	0	0	0	0	0	1	1	1	0
000101	P_PCU	rA_PCH	0	0	0	1	0	0	1	0	0
000110	P_IMMB2	PCL_BUS	0	0	0	0	1	0	0	1	0
000111	-	POP	0	0	0	1	0	0	0	0	0
001000	P_IMMB3	PCL_BUS	0	0	0	0	1	0	1	0	0
001001	-	rA_BUS	0	0	0	1	0	1	0	1	1
001010	-	X4									
001011	P_MEM_W	rL_BUS									
001100	-	BUS_rB	0	0	0	1	0	1	1	0	0
001101	-	X5	0	0	0	1	1	1	1	1	1
001110	P_IMMB1	PCH_BUS	0	0	0	0	0	1	1	1	1
001111	P_IMMB1	BUS_rB									
010000	-	r[src]_rB	0	0	0	1	0	0	0	0	1
010001	P_MEM_R	rL_BUS									
010010	P_IMMB2	PCH_BUS	0	0	0	0	1	0	0	1	1
010011	P_IMMB2	BUS_rB	0	1	0	0	0	1	0	1	0
010100	P_IMMB3	PCH_BUS	0	0	0	0	1	0	1	0	1
010101	P_IMMB3	BUS_rB	0	0	0	0	1	1	1	0	0
010110	-	ALU(+1)	0	0	0	1	1	1	1	1	1
010111	-	ALU(-1)	0	0	0	1	1	1	1	1	1
011000	-	r[src]_rB									
011001	P_MEM_R	rL_BUS									
011010	-	r[src]_rB									

表 3.1: 微指令表 (续)

微地址	微子程序	微指令	Q			μA					
			2	1	0	5	4	3	2	1	0
011011	-	-									
011100	P_IMMA	PCL_BUS	0	0	0	0	1	1	1	0	1
011101	P_IMMA	PCH_BUS	0	0	0	0	1	1	1	1	0
011110	P_IMMA	BUS_rA									
011111	-	ALU(ROT)									
100000	-	X6									
100001	-	ALU(op)	0	0	0	1	1	1	1	1	1
100010	-	rB_r[D]									
100011	-	X7									
100100	-	rH_BUS									
100101	-	rB_BUS									
100110	-	rH_BUS									
100111	-	BUS_rB									
101000	-	rH_BUS									
101001	-	BUS_rB									
101010	-	X8									
101011	-	rB_BUS									
101100	-	COND_BUS	0	0	0	1	0	1	1	0	1
101101	-	rB_rA	0	0	0	1	1	1	1	1	1
101110	-	BUS_rB	0	0	0	1	0	1	1	1	0
101111	-	rB_PCL	0	0	0	1	1	1	1	1	1
111101	-	BUS_rB/IR	0	0	1	0	0	0	0	0	0
111110	-	PCH_BUS	0	0	0	1	1	1	1	0	1
111111	-	PCL_BUS	0	0	0	1	1	1	1	1	0

表 3.2: 微指令表

微地址	微指令	P					μA					
		4	3	2	1	0	4	3	2	1	0	
000000	X-INr											
000001	X-DCr											
000010	X-ALUop											
000011	POP(c)											
000100	PCL_Imm											
000101	PCHa											
000110	PCL_Imm											
000111	POP											
001000	PCL_ImmB											
001001	rA_o											
001010	ImmA											
001011	ImmA-PUSH											
010000	rR											
010001	rL_o											
010010	ALUop											
010011	rH_o											
010100	rB_i											
010101	X											
011000	rR											
011001	rR											
011010	rL_o											
011011	HLT											
100000	X	0	0	0	0	0	1	1	1	1	1	1
100001	rW	0	0	0	0	0	1	1	1	1	1	1
111101	IF	0	0	0	0	0	0	0	0	0	0	0
111110	PCH_I	0	0	0	0	0	1	1	1	1	0	1
111111	PCL_I	0	0	0	0	0	1	1	1	1	1	0

Chapter 4

CPU 设计 (微程序版本)

4.1 微指令

4.1.1 微指令转移

$$
\begin{aligned}
S_M &= D_2D_1D_0 && (4.1)\\
D_M &= D_5D_4D_3 && (4.2)\\
A_5 &= \mu A_5 && (4.3)\\
A_4 &= \mu A_4 + Q_0D_7 && (4.4)\\
A_3 &= \mu A_3 + Q_0D_6 && (4.5)\\
A_2 &= \mu A_2 + Q_0(\bar{D}_7\bar{D}_6D_2) && (4.6)\\
A_1 &= \mu A_1 + Q_0(\bar{D}_7\bar{D}_6D_1 + D_7D_6D_M) && (4.7)\\
A_0 &= \mu A_0 + Q_0(\bar{D}_7D_0 + D_7S_M) + Q_1D_M + Q_2(D_5 + D_4) && (4.8)
\end{aligned}
$$

4.1.2 微指令表

表 4.1: 微指令表 2

微地址	微子程序	微指令	时序	S			Q			μA					
				2	1	0	2	1	0	5	4	3	2	1	0
000000	-	X1-T4	T4	1	1	1	0	0	0	1	1	0	0	0	1
000001	-	X2-T4	T4	1	1	1	0	0	0	1	1	0	0	1	0
000010	-	X3-T4	T4	1	1	1	0	0	0	1	1	0	0	1	1
000011	-	POP	T4	1	1	1	0	0	0	1	1	0	1	0	1
000100	IMMB1	PCL-BUS	T1	0	1	0	0	0	0	0	1	0	0	1	0
000101	PCupdate	rA-PCH	T4	1	1	1	0	0	0	1	1	0	1	0	0
000110	IMMB2	PCL-BUS	T1	0	1	0	0	0	0	0	1	0	1	0	0
000111	-	POP	T4	1	1	1	0	0	0	1	1	0	1	1	0
001000	IMMB3	PCL-BUS	T1	0	1	0	0	0	0	0	1	0	1	1	0
001001	-	rA-BUS	T4	1	1	1	0	0	0	1	0	0	1	0	1

表 4.1: 微指令表 2(续)

微地址	微子程序	微指令	时序	S			Q			μA					
				2	1	0	2	1	0	5	4	3	2	1	0
001010	-	X4-T4	T4	1	1	1	0	0	0	1	1	1	0	1	1
001011	-	rL-BUS	T4	1	1	1	0	0	0	0	1	1	1	1	1
001100	-	BUS-rB	T4	1	1	1	0	0	0	1	0	0	1	1	0
001101	-	X1-T3	T3	0	0	1	0	0	0	1	1	1	1	1	1
001110	-	X	T5	1	0	1	0	0	0	1	1	1	1	1	1
001111	-	X	T5	1	0	1	0	0	0	1	1	1	1	1	1
010000	-	r[S]-BUS	T4	1	1	1	0	0	0	1	1	1	0	0	1
010001	-	rL-BUS	T1	0	1	0	0	0	0	0	1	1	1	0	0
010010	IMMB1	PCH-BUS	T2	1	0	0	0	0	0	0	1	0	0	1	1
010011	IMMB1	BUS-rB	T3	0	0	1	0	0	0	0	1	1	1	1	0
010100	IMMB2	PCH-BUS	T2	1	0	0	0	0	0	0	1	0	1	0	0
010101	IMMB2	BUS-rB	T3	0	0	1	0	1	0	0	0	1	0	1	0
010110	IMMB3	PCH-BUS	T2	1	0	0	0	0	0	0	1	0	1	1	1
010111	IMMB3	BUS-rB	T3	0	0	1	0	0	0	1	0	0	1	1	1
011000	-	r[S]-rB	T4	1	1	1	0	0	0	1	1	1	0	1	1
011001	MEMR	rL-BUS	T1	0	1	0	0	0	0	1	0	0	0	0	0
011010	-	r[S]-rB	T4	1	1	1	0	0	0	0	0	1	0	1	1
011011	-	X	T5	1	0	1	0	0	0	1	1	1	1	1	1
011100	MEMR	rH-BUS	T2	1	0	0	0	0	0	0	1	1	1	0	1
011101	MEMR	BUS-rB	T3	0	0	1	0	0	0	0	1	1	1	1	0
011110	-	X5-T4	T4	1	1	1	0	0	0	1	1	1	0	0	1
011111	MEMW	rH-BUS	T2	1	0	0	0	0	0	1	1	1	0	1	0
100000	MEMR	rH-BUS	T2	1	0	0	0	0	0	1	0	0	0	0	1
100001	-	BUS-rB	T3	0	0	1	0	0	0	1	0	0	0	1	0
100010	-	X6-T4	T4	1	1	1	0	0	0	1	1	1	0	1	1
100011	IMMA	PCH-BUS	T2	1	0	0	0	0	0	1	0	0	0	1	1
100100	IMMA	BUS-rA	T3	0	0	1	0	0	0	0	0	0	1	0	1
100101	-	rB-BUS	T3	0	0	1	1	0	0	0	0	1	1	0	0
100110	-	COND-BUS	T4	1	1	1	0	0	0	1	1	0	1	1	1
100111	-	PCL-BUS	T1	0	1	0	0	0	0	1	0	0	0	1	1
110001	-	ALU(+1)	T5	1	0	1	0	0	0	1	1	1	1	1	1
110010	-	ALU(-1)	T5	1	0	1	0	0	0	1	1	1	1	1	1
110011	-	ALU(ROT)	T5	1	0	1	0	0	0	1	1	1	1	1	1
110100	-	rB-PCL	T5	1	0	1	0	0	0	1	1	1	1	1	1
110101	-	X-T5	T5	1	0	1	0	0	0	1	1	1	1	1	1
110110	-	rB-PCL	T5	1	0	1	0	0	0	1	1	1	1	1	1
110111	-	rB-rA	T5	1	0	1	0	0	0	1	1	1	1	1	1
111000	-	X	T5	1	0	1	0	0	0	1	1	1	1	1	1
111001	-	ALU(op)	T5	1	0	1	0	0	0	1	1	1	1	1	1

表 4.1: 微指令表 2(续)

微地址	微子程序	微指令	时序	S			Q			μA					
				2	1	0	2	1	0	5	4	3	2	1	0
111010	-	rB-BUS	T3	0	0	1	0	0	0	1	1	1	1	1	1
111011	-	rB-r[D]	T5	1	0	1	0	0	0	1	1	1	1	1	1
111100	-	X	T5	1	0	1	0	0	0	1	1	1	1	1	1
111101	IF-2	BUS-rB/IF	T3	0	0	1	0	0	1	0	0	0	0	0	0
111110	IF-1	PCH-BUS	T2	1	0	0	0	0	0	1	1	1	1	0	1
111111	IF-0	PCL-BUS	T1	0	1	0	0	0	0	1	1	1	1	1	0

4.1.3 状态转换

Chapter 5

CPU 设计 (状态机版本)

5.1 状态机切换

5.1.1 状态变量

- 指令周期: C1, C2, C3
- 执行周期: T1, T2, T3, T4, T5

5.1.2 状态切换

- C1/T1: next=C1/T2
- C1/T2: next=C1/T3
- C1/T3
  - 0000000x/11111111: next=STOP
  - 00xxx011: next=T1 (cond)
  -
- C1/T4
- C1/T5

5.2 操作码编码

表 5.1: 操作码编码

指令	指令码	MR	MW	ALU	ROT	INr	DCr	regR	regW	J/C	RET	INP	OUT	RST
INr	00DDD000	0	0	0	0	1	0	0	0	0	0	0	0	0
DCr	00DDD001	0	0	0	0	0	1	0	0	0	0	0	0	0
ROT	00PPP010	0	0	1	1	0	0	0	0	0	0	0	0	0
RFc/RTc	00XCC011	0	0	0	0	0	0	0	0	0	1	0	0	0
ALU op I	00PPP100	0	0	1	0	0	0	0	0	0	0	0	0	0

表 5.1: 操作码编码 (续)

指令	指令码	MR	MW	ALU	ROT	INr	DCr	regR	regW	J/C	RET	INP	OUT	RST
RST	00AAA101	0	0	0	0	0	0	0	0	0	0	0	0	1
LrI	00DDD110	0	0	0	0	0	0	0	1	0	0	0	0	0
LMI	00111110	0	1	0	0	0	0	0	0	0	0	0	0	0
RET	00XXX111	0	0	0	0	0	0	0	0	0	1	0	0	0

5.2.1 操作编码

- Rs, Ms, Is
- Rd, Md, ALU

表 5.2: 操作编码

指令	指令码	T4/src				T5/dst				misc					
		X	reg	Imm	Mem	X	reg	Mem	ALU	ROT	I/D	PC	IO	POP	RST
INr	00DDD000	0	0	0	0	0	0	0	0	0	1	0	0	0	0
DCr	00DDD001	0	0	0	0	0	0	0	0	0	1	0	0	0	0
ROT	000RR010	0	0	0	0	0	0	0	0	1	0	0	0	0	0
RFc/RTc	00XCC011	0	0	0	0	0	0	0	0	0	0	0	0	1	0
ALU op I	00PPP100	0	0	1	0	0	0	0	1	0	0	0	0	0	0
RST	00AAA101	0	0	0	0	0	0	0	0	0	0	0	0	0	1
LrI	00DDD110	0	0	1	0	0	1	0	0	0	0	0	0	0	0
LMI	00111110	0	0	1	0	0	0	1	0	0	0	0	0	0	0
RET	00XXX111	0	0	0	0	0	0	0	0	0	0	0	0	1	0
JFc/JTc	01XCC000	0	0	0	0	0	0	0	0	0	0	1	0	0	0
CFc/CTc	01XCC010	0	0	0	0	0	0	0	0	0	0	1	0	0	0
JMP	01XXX100	0	0	0	0	0	0	0	0	0	0	1	0	0	0
CAL	01XXX110	0	0	0	0	0	0	0	0	0	0	1	0	0	0
INP	0100MMM1	0	0	0	0	0	0	0	0	0	0	0	1	0	0
OUT	01RRMMM1	0	0	0	0	0	0	0	0	0	0	0	1	0	0
ALU op r	10PPPSSS	0	1	0	0	0	0	0	0	0	0	0	0	0	0
ALU op M	10PPP111	0	0	0	1	0	0	0	0	0	0	0	0	0	0
Lrr	11DDDSSS	0	1	0	0	0	1	0	0	0	0	0	0	0	0
LrM	11DDD111	0	0	0	1	0	1	0	0	0	0	0	0	0	0
LMr	11111SSS	0	1	0	0	0	0	1	0	0	0	0	0	0	0

5.3 总线设计

5.3.1 CPU 内总线

CPU 内总线使用与总线 (wand), 使用以下逻辑关系

$$\begin{aligned} DB &= \overline{D \cdot OE} \\ IN &= \overline{DB} \end{aligned}$$

Chapter 6

CPU 设计

6.1 指令译码设计

6.1.1 指令分类

- 数据转移指令: Lrr, LrM, LMr, LMI
- 算术指令: ALU op r, ALU op M, ALU op I
- 跳转指令: JMP, JFc/JTc, CAL, CFc/CTc, RET, RFc/RTc
- 特殊指令: INr/DCr, ROT, NOP, HLT
- I/O 指令: INP, OUT

6.1.2 模块组成

- rA
  - 源: I(BUS), rB
- rB
  - 源: r[SRC], M(BUS), I(BUS)
- regBank
  - 源: rB
- stack
  - 源: rA, rB
  - 操作: POP, PUSH
- ALU
  - 源: rA, rB
  - 操作: OP
- ACC (r[0])

  - 源: ALU
  - 操作: ROT, INr, DCr

6.1.3 指令译码

Chapter 7

基本逻辑

7.1 编码器

7.1.1 8-3 编码器

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	A_2	A_1	A_0
0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	1	0	0	0	1	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	0
0	0	1	0	0	0	0	0	1	0	1
0	1	0	0	0	0	0	0	1	1	0
1	0	0	0	0	0	0	0	1	1	1

$$A_0 = D_1 + D_3 + D_5 + D_7 \tag{7.1}$$

$$A_1 = D_2 + D_3 + D_6 + D_7 \tag{7.2}$$

$$A_2 = D_4 + D_5 + D_6 + D_7 \tag{7.3}$$