计算机学院计算机科学与技术专业

《计算机组成原理课程设计》报告

〔2011/2012 学年 第一学期〕

学生姓名： 某某

学生班级： 计算机 092001 班

学生学号： 200920010117

指导教师： 某某某

2012 年 01 月 15 日

目录

1 关于此次课程设计

1.1 课程设计目的

本课程设计是计算机科学与技术专业重要的实践性教学环节之一，是在学生学习《计算机组成原理》课程后进行的一次全面的综合设计。目的是通过一个完整的 8 位指令系统结构〔ISA〕的设计和实现，加深对计算机组成原理课程内容的理解，建立起整机系统的概念，掌握计算机设计的根本方法，培养学生科学的工作作风和分析、解决实际问题的工作能力。

课程设计内容及要求

基于 TDN-CM++计算机组成原理实验教学系统，设计和实现一个 8 位指令系统结构〔ISA〕，通过调试和运行，使设计的计算机系统能够完成指定的功能。

设计过程中要求考虑到以下各方面的问题：

〔1〕指令系统风格〔存放器-存放器，存放器-存储器，存储器-存储器〕；

〔2〕数据类型〔无符号数，有符号数，整型，浮点型〕；

〔3〕存储器划分〔指令，数据〕；

〔4〕寻址方式〔立即数寻址，存放器寻址，直接寻址等〕；

〔5〕指令格式〔单字节，双字节，多字节〕；

〔6〕指令功能类别〔算术 / 逻辑运算，存储器访问，存放器操作，程序流控制，输入 / 输出〕。

〔7〕依据 CPI 值对指令系统进行功能分析。

要求学生综合运用计算机组成原理、数字逻辑和汇编语言等相关课程的知识，理解和熟悉计算机系统的组成原理，掌握计算机主要功能部件的工作原理和设计方法，掌握指令系统结构设计的一般方法，熟练地掌握并运用微程序设计〔Microprogramming〕思想，在设计过程中能够发现、分析和解决各种各样的问题，自行设计自己的指令系统结构〔ISA〕。

2 可行性分析

课程设计要求学生综合运用计算机组成原理、数字逻辑和汇编语言等相关课程的知识，理解和熟悉计算机系统的组成原理，掌握计算机主要功能部件的工作原理和设计方法，掌握指令系统结构设计的一般方法，掌握并运用微程序设计〔Microprogramming〕思想，在设计过程中能够发现、分析和解决问题，自行设计自己的指令系统结构〔ISA〕。

本课程设计要求基于 TDN-CM++计算机组成原理实验教学系统，设计和实现一个 8 位指令系统结构〔ISA〕，通过调试和运行，使设计的计算机系统能够完成指定的功能。

在设计中，我们要考虑到，指令系统风格、数据类型、存储器划分、寻址方式、指令格式指、令功能类别等问题，最后还要对所涉及的指令系统进行 CPI 分析。综合考虑以上诸问题，我组方案设计一个存放器-存放器风格的二进制无符号的整数型的可变字节的体系结构。用到了 RO,R1,R2 三个通用存放器.该体系结构可以实现存储器的访问〔LW、SW〕、程序流控制和简单的逻辑和算术运算。

2.1 指令系统风格

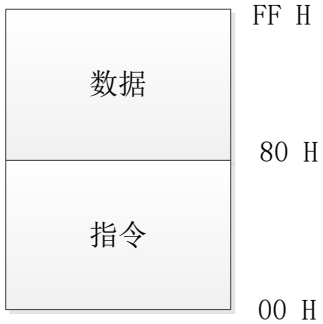
本实验采用存放器——存放器指令系统风格。

2.2 数据类型

本设计涉及整型二进制数，不支持浮点数。

2.3 存储器划分

系统可用存储空间为 256*8 bits=256 B.



2.4 寻址方式

寻址方式	指令	助记符	说明
直 接 寻 址	LB		RD←[addr]
	SB	SB Addr,RS	[addr]←RS
存放器 寻 址	BEQ	BEQ RS，RD Addr	If(RS==RD)Goto[addr]
	SLT	SLT R0,R1，R2	IF R1<R2,R0=FFH ,ELSE R0=00 H

本设计涉及三个通用存放器：R0，R1，R2.

2.5 指令格式

字节类别	指令格式	指令内容
单字节	7 6 5 4 Op / 3 2 Rs / 1 0 Rd	IN OUT ADD HALT NULL MOVE
双字节	7 6 5 4 op / 3 2 Rs / 1 0 Rd / address	LB SB JMP

3 分析阶段

3.1 微指令格式分析

微指令格式如下表：

表 3-1 微代码定义

S3 S2 S1 S0 M CN	WE	A9 A8	A	B	C	μA5～μA0
24 23 22 21 20 19	18	17 16	1514 13	12 11 10	9 8 7	6 5 4 3 2 1

每个字段的具体含义如下：

〔1〕字段 24～19 控制运算器的控制端，通过改变 S3～CN 来决定对数据进行何种算术或逻辑运算。本设计中全部为正逻辑运算。

	算术运算		逻辑运算
	无进位	有进位	
M	0	0	1
CN	1	0	×

〔2〕字段 18 为控制对主存 W/R 的开关

A9	A8	WE	说明
0	1	0	对主存进行对操作
0	1	1	对主存进行写操作

〔3〕字段 17、16 控制 24 译码器的输出端，对 Y0、Y1、Y2 进行选择。

A9	A8	Yi	操作	说明
0	0	Y0	选中 SW-B	INPUT UNIT 的开关
0	1	Y1	选中 CE	MAIN MEN 的控制片选开关
1	0	Y2	选中 LED-B	OUTPUT UNIT 的开关
1	1	×	NULL	

〔4〕字段 15～7 为 A、B、C 三个开关控制端。

A 字段

15	14	13	开关	说明
0	0	1	LDDRi	控制存放器 Ri 的写入
0	1	0	LDDR1	暂存器 DR1 的控制开关
0	1	1	LDDR2	暂存器 DR2 的控制开关
1	0	0	LDIR	指令存放器 IR 的控制开关
1	0	1	LOAD	非自动输入的数据装载入 PC 计数器的控制开关
1	1	0	LDAR	地址存放器 AR 的控制开关
0	0	0	NULL	空操作

B 字段

12	11	10	开关	说明

0	0	1	RS-B	存放器 R0、R1、R2 的输出开关
0	1	0	RD-B	存放器 R0、R1、R2 的输出开关
0	1	1	RI-B	存放器 R0、R1、R2 的输出开关
1	0	1	ALU-B	运算器 ALU 的输出开关
1	1	0	PC-B	PC 计数器的输出开关
1	0	0	299-B	本设计中不涉及
0	0	1	NULL	空操作

C 字段

9	8	7	开关	说明
0	0	1	P(1)	指令译码器中的 P(1)为低电平有效
0	1	0	P(2)	指令译码器中的 P(2)为低电平有效
0	1	1	P(3)	指令译码器中的 P(3)为低电平有效
1	0	0	P(4)	指令译码器中的 P(4)为低电平有效
1	1	0	LDPC	将自动输入的数据加 1 后输入到 PC 计数器中的控制开关
1	0	1	AR	本设计中不涉及
0	0	1	NULL	空操作

〔5〕字段 6～1 为该条微程序的八位二进制后继地址，其决定顺序执行哪条微程序。

指令译码电路分析

指令译码工作原理图如下：

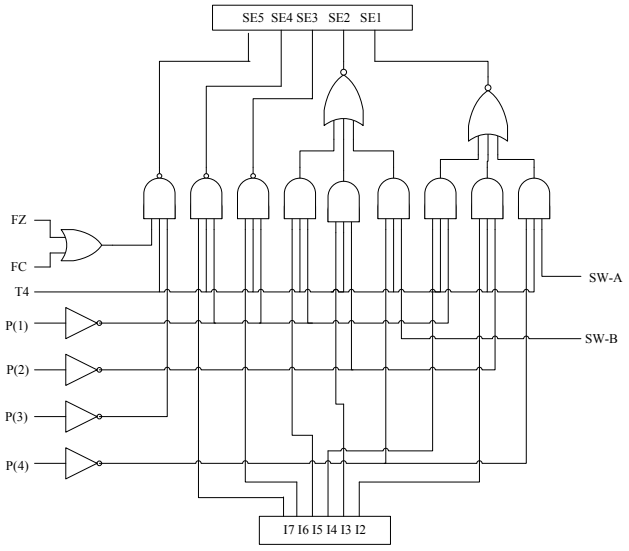


图 3.1 指令译码原理图

P(1) ～P(4)为低电平有效，中选用时该信号为零；I7 ～I2 表示机器指令前六位；SE5 ～SE1 表示能够强置改变入口地址的后五位。

根据上图得出指令译码器的逻辑表达式如下：

$$SE5=\overline{(FC+FZ)*T4*\overline{P(3)}}$$

$$SE4=\overline{I7*T4*\overline{P(1)}}$$

$$SE3=\overline{I6*T4*\overline{P(1)}}$$

$$SE2=\overline{I5*T4*\overline{P(1)}+I3*T4*\overline{P(2)}+T4*\overline{P(4)}*SWB}$$

$$SE1=\overline{I4*T4*\overline{P(1)}+I2*T4*\overline{P(2)}+T4*\overline{P(4)}*SWA}$$

其真值表如下：

	P(1)				P(3)		P(4)	
	$\overline{I7}$	$\overline{I6}$	I5	I4	FC	FZ	SWB	SWA
SE5	1	1	1	1	0	0	1	1

续表：

SE4	0	1	1	1	1	1	1	1
SE3	1	0	1	1	1	1	1	1
SE2	1	1	0	1	1	1	0	1
SE1	1	1	1	0	1	1	1	0

拟定机器指令通过上式即可算出每条子程序的入口地址。

1，各信号有效都需要 T4 可到来。

2，本设计不设计 P(2)。

3.2.1 指令操作码字段与微程序入口地址形成的关系

字段	SE5	SE4	SE3	SE2	SE1
P(1)	1	I7	I6	I5	I4
P(2)	1	1	1	I3	I2
P(3)	FC	1	1	1	1
P(4)	1	1	1	SWB	SWA

1.P(1)

机器指令	SE5 SE4 SE3 SE2 SE1	入口地址
0 0 0 0 × × × ×	1 1 1 1 1	0 0 0 0 0 0
0 0 0 1 × × × ×	1 1 1 1 0	0 0 0 0 0 1
0 0 1 0 × × × ×	1 1 1 0 1	0 0 0 0 1 0
0 0 1 1 × × × ×	1 1 1 0 0	0 0 0 0 1 1
0 1 0 0 × × × ×	1 1 0 1 1	0 0 0 1 0 0

0 1 0 1 × × × ×	1 1 0 1 0	0 0 0 1 0 1
0 1 1 0 × × × ×	1 1 0 0 1	0 0 0 1 1 0
0 1 1 1 × × × ×	1 1 0 0 0	0 0 0 1 1 1
1 0 0 0 × × × ×	1 0 1 1 1	0 0 1 0 0 0
1 0 0 1 × × × ×	1 0 1 1 0	0 0 1 0 0 1
1 0 1 0 × × × ×	1 0 1 0 1	0 0 1 0 1 0
1 0 1 1 × × × ×	1 0 1 0 0	0 0 1 0 1 1
1 1 0 0 × × × ×	1 0 0 1 1	0 0 1 1 0 0
1 1 0 1 × × × ×	1 0 0 1 0	0 0 1 1 0 1
1 1 1 0 × × × ×	1 0 0 0 1	0 0 1 1 1 0
1 1 1 1 × × × ×	1 0 0 0 0	0 0 1 1 1 1

2. P(2)

机器指令	SE5～SE1	入口地址
× × × × 0 0 × ×	1 1 1 1 1	0 0 0 0 0 0
× × × × 0 1 × ×	1 1 1 1 0	0 0 0 0 0 1
× × × × 1 0 × ×	1 1 1 0 1	0 0 0 0 1 0
× × × × 1 1 × ×	1 1 1 0 0	0 0 0 0 1 1

3.P(3)

机器指令	SE5～SE1	入口地址
× × 0 × × × × ×	1 1 1 1 1	0 0 0 0 0 0
× × 1 × × × × ×	0 1 1 1 1	0 1 0 0 0 0

4. P(4)

SWB	SWA	SE5～SE1	入口地址
0	0	1 1 1 1 1	0 0 0 0 0 0
0	1	1 1 1 1 0	0 0 0 0 0 1
1	0	1 1 1 0 1	0 0 0 0 1 0
1	1	1 1 1 0 0	0 0 0 0 1 1

3.3 存放器译码电路分析

存放器译码原理图如下:

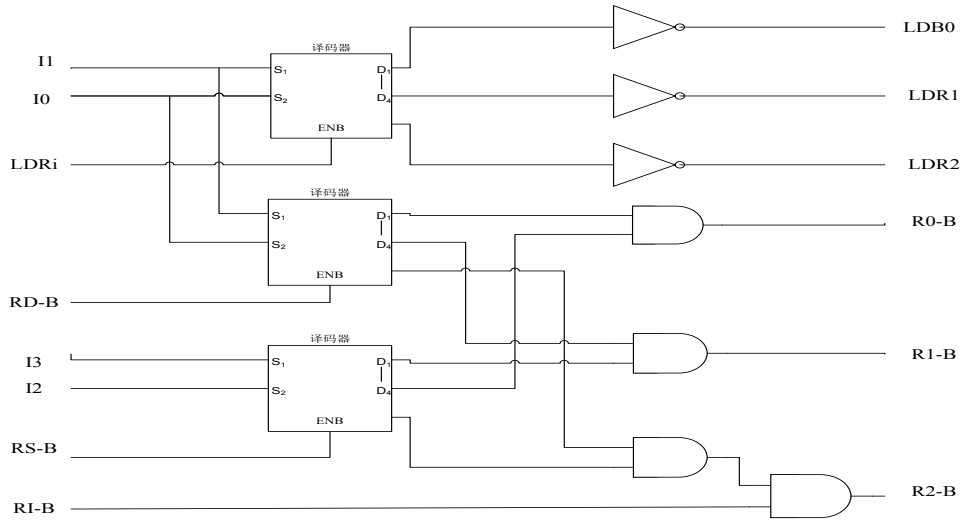


图 3.2 存放器译码原理图

存放器的输入、输出不仅决定于输入、输出开关，还与机器指令的后四位(即 I3～I0)有关，由其决定哪个存放器被选中。

3.3.1 存放器的输入

LDRi 为存放器的输入开关，且为低电平有效(即 LDRi=0)，I1、I0 对存放器进行选择，决定数据进入哪个存放器。

	LDRi	I1	I0
LDR0	0	0	0
LDR1	0	0	1
LDR2	0	1	0

3.3.2 存放器的输出

RS-B、RD-B、RI-B 为存放器的输出开关，且为低电平有效；I3、I2 对存放器进行选择，决定从哪个存放器输出指令；从原理图上可以得出 R2-B 的输出，假设 RI-B 有效那么无需关注 I3、I2 因而 I3、I2 可为任意状态。

	RS-B	RD-B	RI-B	I3	I2
R0-B	0	1	1	0	0
R1-B	0	1	1	0	1
R2-B	0	1	1	1	0
	1	1	0	×	×

3.4 时序分析

T1、T2、T3、T4 为节拍控制端，本设计用了 T4 节拍控制端，当指令通过译码器 P〔1〕时，P〔1〕对操作码进行测试，通过节拍脉冲 T4 的控制，以便识别所要求的操作。

TS1 时进行微程序控制器控制,TS2 时进行微指令存放器控制,TS 时控制 LDIR、LDAR，TS4 时对 P〔1〕、P〔2〕、P〔3〕、P〔4〕、AR、LOPC、LDRi、LDDR1、LDDR2 进行控制。

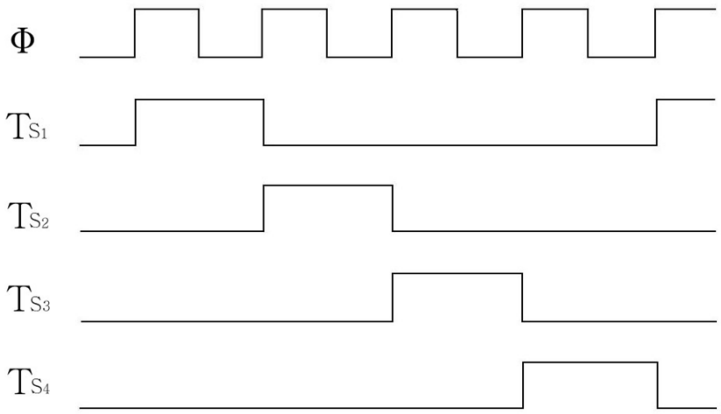


图 2.3 时序信号图

3.4.1 时钟节拍对应的微操作

时钟	有效信号	微操作
T1	微控制器编程单元 PROM	编程
	微控制器编程单元 READ	校验
	微控制器编程单元 RUN	运行
T2	微指令存放器 MIR	MIR 锁存
	微地址存放器 MAR	MAR 锁存
T3	指令存放器 IR、LOIR	指令装入指令存放器
	地址存放器 AR、DDAR	地址装入地址存放器
T4	P(i)有效	选择分支
	AR 有效	算术运算影响进位及清零
	LDPC 有效，PC 计数器	PC+1 输入 PC 计数器
	LDRi 有效	数据装入 Ri

	LDDR1 有效	数据装入暂存器 DR1
	LDDR2 有效	数据装入暂存器 DR2

4 初步设计阶段

4.1 数据格式和机器指令描述

4.1.1 数据格式

本设计中所有需要处理的数据全部采用定点无符号整数表示，8 个 bit 位，格式如下：

7	6	5	4	3	2	1	0
数值							

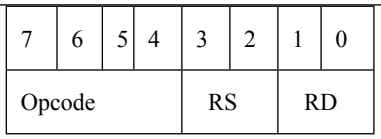
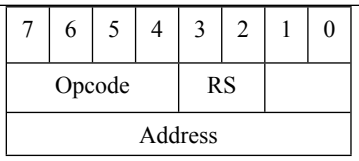
数据的范围是 0～28，即 0～255。

4.1.2 机器指令描述

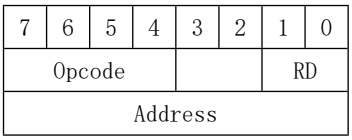
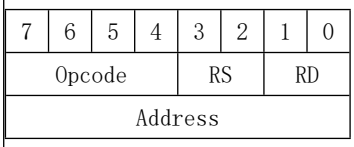
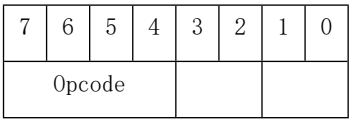
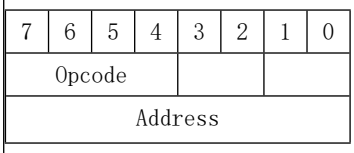
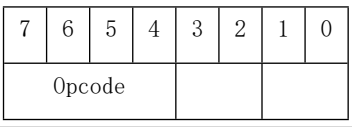
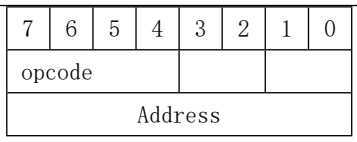
设计用到 11 条机器指令，分别是 IN,LB,SB,OUT,JMP,ADD,BEQ,STOP,MOVE,SLT,HALT。在三个存放器的协调下，分别发挥着不同的作用。下面我们将详细介绍着 11 条指令它们的标识符，指令格式，寻址方式，指令风格，指令功能及作用说明
机器指令描述见下表:

表 4-1　机器指令描述

指令类别	指令名称	标志符	指令格式		寻址方式	指令系统风格	指令功能	说明
I/O	输入	IN RD	单字节	7 6 5 4 3 2 1 0 Opcode　　RD			RD←“INPUT DEVICE”	“INPUT DEVICE”数放入 RI
	输出	OUT RD	单字节	7 6 5 4 3 2 1 0 Opcode　　RD			LED←RD	将存放器中的数据在 LED 中显示

算术逻辑运算	加	ADD RS,RD	单字节		存放器寻址	存放器—存放器	RD←RS+RD	将两存放器的数据相加放入存放器
存储器访问	存字	SB Addr,RS	双字节		直接寻址	存放器－存储器	[addr]←RS	将 RD 中的数据放入存储器

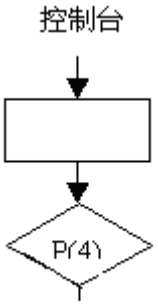
续表:

取字	LB	双字节		直接寻址	存放器－存储器	RD←[addr]	将存储器中的数据取出并放入 RD
相等时跳转	BEQ RS，RD Addr	双字节		直接寻址	存放器－存储器	If(RS==RD)Go to[addr]	假设 RD、RS 相等跳转至指定地址
空操作	NOP	单字节					Do nothing
无条件跳转	JMP Addr	双字节		直接寻址		Go to[addr]	无条件跳转至[addr]
停机	HALT	单字节					停机
比拟大小跳转	SLT R0,R1，R2	双字节		直接寻址		IF R1<R2,R0=FFH ,ELSE R0=00 H	比拟两个数大小，有进位跳转

移动	MOVE	单字节	7 6 5 4 3 2 1 0 opcode / RS / RD			RD←RS	将 RS 中的送入 RD 中

4.2 控制台微程序流程

4.2.1 公操作



对机器进行总清零 CLR 1-0-1。

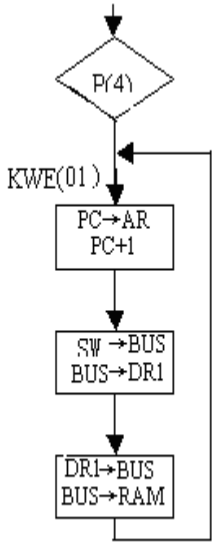
S3 S2 S1 S0 M CN	WE	A9 A8	A	B	C
0 0 0 0 0 0	0	1 1	0 0 0	0 0 0	1 0 0
NULL					P(4)

选中 P(4)，通过译码形成入口地址。

4.2.2 强置写

形成入口地址后，执行写操作。

1.



S3 S2 S1 S0 M CN	WE	A9 A8	A	B	C
0 0 0 0 0 0	0	1 1	1 1 0	1 1 0	1 1 0
NULL			LDAR	PC-B	LDPC

〔1〕翻开 PC-B 将数据送到总线上；
〔2〕翻开 LDAR 将数据从总线流到 AR 中；
〔3〕翻开 LDPC，让自动加 1 的数据进入 PC 中。

2.

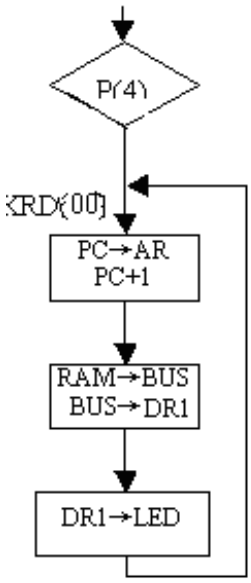
S3 S2 S1 S0 M CN	WE	A9 A8	A	B	C
0 0 0 0 0 0	0	0 0	0 1 0	0 0 0	0 0 0
NULL		SW-B	LDDR1	NULL	

〔1〕翻开 SW-B 将数据送到总线上，
〔2〕翻开 LDDR1 将数据从总线流到 DR1 中。
3.

S3 S2 S1 S0 M CN	WE	A9 A8	A	B	C
0 0 0 0 0 1	1	0 1	0 0 0	1 0 1	0 0 0

〔1〕翻开 ALU-B 将数据送到总线上，其间 DR1 中的数据相当于在 ALU 中做 F=A 的运算；
〔2〕翻开 CE、WE 置成 01 状态，将数据从总线流到主存相应地址单元中，完成数据写操作。

4.2.3 强置读



形成入口地址后，执行写操作。
1.

S3 S2 S1 S0 M CN	WE	A9 A8	A	B	C
0 0 0 0 0 0	0	1 1	1 1 0	1 1 0	1 1 0
NULL			LDAR	PC-B	LDPC

〔1〕翻开 PC-B 将数据送到总线上；
〔2〕翻开 LDAR 将数据从总线流到 AR 中；
〔3〕翻开 LDPC，让自动加 1 的数据进入 PC 中。
2.

S3 S2 S1 S0 M CN	WE	A9 A8	A	B
0 0 0 0 0 0	0	0 1	0 1 0	0 0 0
NULL	WE	CE	LDDR1	NU

〔1〕翻开 CE、WE 置成 00 状态，将数据从主存送到总线上；
〔2〕翻开 LDDR1 将数据从总线流到 DR1 中。
3.

S3 S2 S1 S0 M CN	WE	A9 A8	A	B	C
0 0 0 0 0 1	1	1 0	0 0 0	1 0 1	0 0 0
F=A	NULL	LED-B	NULL	ALU-B	NULL

〔1〕翻开 DR1-B 将数据送到总线上，其间 DR1 中的数据相当于在 ALU 中做 F=A 的运算；
〔2〕翻开 LED-B，数据从总线流到输出单元，在数码管上显示出来，完成数据读操作。