

第 3 章

芯片级实验

3.1 认知主板上元器件布局和开关、指示灯的使用方法

【实验目的】

熟悉主板上元器件布局是开展各项实验的基础,了解教学计算机各部件的组成及其相互连接关系有利于提高实验质量。为此需要较为认真地看一看教学机的照片,粗浅地看一看在图 1.1 中给出的第一个系统的基本硬件组成逻辑框图,结合实验机的照片初步了解主板上元器件布局,特别是开关和指示灯的位置、作用和使用方法。

【实验说明】

电路板下侧设置有 6 组 8 位的拨数开关,开关向上拨输出高电平,表示 1;向下拨为低电平,表示 0。在手动操作实验中,用于向实验电路提供运行数据和控制信号等,需要能够按照使用要求,把开关的输出连接到实验电路。这里为每一位开关的输出留有一个接线插孔,还把每组 8 位开关的输出通过一片 SN74LS244 三态门芯片(用到输出使能信号 G)送到 8 位的接线插针,当 G 为低电平时,芯片输出正常电平,开关所拨数据送到 8 位排针,当 G 为高电平时,芯片输出为高阻态,没有信息送到 8 位排针。

电路板上安排了很多指示灯,当其输入信号为高电平时灯被点亮,为低点平时灯熄灭。这些指示灯可以显示系统或电路运行过程中的状态,帮助实验者了解系统运行的步骤、执行的功能、运行的结果、用到的控制信号的状态等。

电路板上方的中间位置设置有 4 组 8 位的通用指示灯,可用于显示不同电路的信息,取决于把哪个电路的信息接通到这里。出于接线需求考虑,为每组指示灯设置有 8 位的接线排针,还为每一位指示灯安排了一个接线插孔。

电路板上还有另外一些专用的指示灯,如数据总线 DB 的 16 位指示灯、地址总线 AB 的 16 位指示灯、指令寄存器 IR 的 16 位指示灯及电路板最左上角的 8 位指示灯,在教学计算机系统运行时,这些指示灯不能转用于显示其他信息。在完成其他线路或者部件实验时,也可以依据实验者安排,用于显示其他内容。出于接线需求,为 DB、AB、IR 都设置了两组 8 位的接线排针,送到这里的信息(系统运行产生的或者通过接线传送来的)将被直接显示出来。

【实验操作】

- (1) 关闭 Am2910、MACH、Am2901、FPGA 这 4 个芯片的电源,禁止它们运行。
 - (2) 用 8 位的排线连接一组开关的输出到一组通用 8 位的指示灯,向用到的 244 芯片的 G 管脚提供低电平的控制信号,拨动开关,查看指示灯的显示内容。若拨入信息是 00000000,则 8 个指示灯都不亮,之后逐位改变开关输出为 1,指示灯会逐个被点亮,直到全亮。
 - (3) 变 G 信号为高电平,则指示灯全部熄灭。
- 结论,这表明用到的 8 个开关、244 芯片、8 位排线、8 个指示灯可能都正确。
- (4) 按照第(2)步的操作,依次检查另外 3 组通用指示灯,看指示灯是否都正确。
 - (5) 按照第(2)步的操作,依次检查另外的 5 组拨数开关,看开关是否都正确。
 - (6) 按照前(2)步的操作,依次检查 DB、AB、IR 指示灯,看指示灯是否都正确。
- 若在实验过程中发现错误,需要找出原因并设法解决。例如,怀疑是排线有问题,更换一条排线试试;若怀疑是 244 芯片有错,更换另外一片试试;若怀疑是哪位指示灯或哪个开关坏了,用万用量表一下这个灯的输入管脚或开关的输出管脚的电平。若感觉是操作失误,找出是哪里操作出错了,按正确的办法重新操作。

这是熟悉设备组成概况的过程,也为今后的实验做了必要储备,学习判断实验中的正确或错误、排除错误的基本技术。实验操作很简单,但对熟悉设备组成和使用方法很有用。

3.2 单独 RAM6116 或 ROM58C65 芯片的读写实验

【实验目的】

了解 RAM 和 ROM 这两种存储器芯片的特性、功能和执行读写操作的基本方法,为设计和构建主存储器部件奠定基础。

【实验说明】

最简单的方案是把存储器芯片插接到用于扩展的 ROM58C65 芯片的 IC 座,它可以通过直接断开接线的方式,使其与计算机其他部件没有连接关系,确保实验操作仅针对此处的存储器芯片。

该芯片的地址信号、数据信号、读写控制信号都可以直接通过开关提供,芯片的读出数据内容可以通过指示灯予以显示。

此时需要关掉 FPGA 芯片的电源,最好也同时关掉 MACH、Am2901、Am2910 芯片的电源;为 MIO 信号提供高电平,使基本内存和串口芯片不工作,并注意以下 6 点。

(1) 此处的 IC 座既能插接管脚 28 的 ROM 芯片,也能插接管脚 24 的 RAM 芯片,在插接 RAM6116 芯片时,需按照右侧对齐方式插接,要求为 RAM 芯片提供读写命令 MWE 信号,24 脚的 V_{cc} 在电路板上已经连接好。需要注意以下几个管脚的不同用法。

IC 座的管脚号	28	27	26	23	2	1
ROM 芯片	V_{cc}	WE	NC(不用)	A11	A12	—
RAM 芯片	—	—	V_{cc} (24 脚)	WE (21 脚)	—	—

(2) 允许对此处 ROM58C65 芯片执行写操作,需要为 ROM 芯片提供 EAB11 信号。

(3) 实验可以只用单个芯片来进行 8 位数据的读写操作,也可以选用两个芯片来进行 16 位数据的读写操作,原理是一样的;差异表现在是对 8 位数据还是 16 位数据进行读写。

(4) 电路板的扩展 LC 座附近设置了 4 个接线插孔,分别标记如下。

EAB11'	EXWE	EXOE	EXCS
接线排的 AB11	IC 座的 27 脚	IC 座的 22 脚	IC 座的 20 脚
	对应	RAM 芯片的 20 脚	RAM 芯片的 18 脚

(5) 存储器芯片的写入数据和读出数据都要经过数据总线传送,为此必须确保不能出现开关所拨的写入数据和从芯片中读出的数据同时送到数据总线 EDB,即只在执行写操作时才送开关数据到 EDB,而在执行读操作时必须禁止送开关数据到 EDB。最简单的方案是把读写命令信号 EXWE 同时用作拨数开关上方的 74LS244 芯片的输出使能信号 G。

(6) 一次存储器读写要占用一个总线周期完成,包括地址时间和数据时间两个时间段。

【实验内容】

- 完成 RAM 芯片的读写操作,检查运行结果的正确性。
- 完成 ROM 芯片的读写操作,检查运行结果的正确性。

【实验操作】

1. RAM 芯片的读写操作实验

(1) 在实验机关机的状态下完成以下接线操作。

① 使用第一组、第二组拨数开关为芯片提供 16 位的地址信息到 EAB,即用两条 8 位的排线连接这些开关的输出到 IC 座的 16 位地址接线排针。

② 使用第三组拨数开关为芯片提供 8 位的数据信息到 EDB,即用一条排线连接这些开关的输出排针到 IC 座的低 8 位的数据接线排针,并把 EXWE 信号用作相应 244 芯片的 G 命令。

③ 使用第四组的低 3 位开关为芯片提供控制信号:片选 EXCS、使能 EXOE,读写命令 G23(即 RAM 芯片的 21 管脚);可以用 3 根 1 号线连接这 3 个开关的输出到 IC 座附近的接线插孔。

EXCS 是片选信号,ROM58C65 和 RAM6116 芯片使用的管脚位置相同,读或者写存储器芯片时片选信号必须为低电平,为高电平时存储器芯片不工作。

EXOE 是输出使能信号,ROM58C65 和 RAM6116 芯片使用的管脚位置相同,在读或者写 6116 芯片时,可使 EXOE 都为低电平。

EXWE 是读写命令,当其为低电平时执行写操作,为高电平时执行读操作。

④ 用一条排线把数据总线 EXDB 的低 8 位连接到一组 8 位的通用指示灯,用于显示从芯片中读出的数据信息。

⑤ 把一片 RAM6116 芯片插到扩展存储器的低位字节的 IC 座,方向要正确并按照右侧对齐方式插。

(2) 关闭 isp MACH、Am2901、Am2910、FPGA 这些芯片的电源,使这些芯片不工作。

(3) 通过一个开关向电路板上的 MIO 接线插孔送去高电平的控制信号,使基本存储器和串行接口都处于不工作状态。

(4) 打开设备机箱上的电源开关,此时电路板上只有辅助电路和扩展的存储器芯片能够运行,就可以开始 RAM 芯片的数据读写操作了。

请注意,每次内存读写都要用两个时间段(地址时间、数据读写时间)来完成,可以使用片选信号电平的高低状态来区分这两个时间段,在 EXCS 为高电平期间是地址时间,此期间存储器芯片并不执行数据读写,正好用开关向芯片给出存储器的单元地址信息和写入的数据信息;在 EXCS 为低电平期间是数据读写时间,不允许地址和写入数据信息发生变化。

写 RAM6116 芯片:此期间需要向 EXWE 提供低电平。

① 向 EXCS 提供高电平,进入总线的地址时间,用开关拨入写操作的存储器首地址,如十六进制的 0000,接着拨入写操作的数据内容,如[^]h00。

② 变 EXCS 为低电平,进入总线的数据时间,完成存储器的数据写入操作,之后立即恢复 EXCS 为高电平,转回到步骤①,就可以启动下一次的写操作。

重复上述两个操作步骤,依次完成向[^]h0001、[^]h0002、[^]h0003……各存储单元分别写入[^]h11、[^]h22、[^]h33……不同数据。

读 RAM6116 芯片:此期间需要向 EXWE 提供高电平。

① 向 EXCS 提供高电平,进入总线的地址时间,用开关拨入读操作的存储器首地址,如十六进制的 0000。

② 变 EXCS 为低电平,进入总线的数据时间,完成存储器的数据读出操作,此时读出数据的内容就会显示在相关的 8 位指示灯,看它是否是原来的写入内容。

重复上述两个操作步骤,依次检查存储器各单元中的数据内容,若写入和读出操作均无错,则从[^]h0001、[^]h0002、[^]h0003……各地址读出的数据应是[^]h11、[^]h22、[^]h33……内容。

请注意以下 4 点说明。

① 在写 RAM 芯片的整个过程中,只能在 EXCS 为高电平的期间,通过开关拨入芯片的地址和写入数据;否则可能造成写入错误。

② 在读 RAM 芯片的整个过程中,也可使 EXCS 保持低电平不变,通过开关直接拨入芯片的地址并通过指示灯观察读出的数据内容,并且拨入的地址次序可以随意安排。

③ 还可以把 RAM 芯片的写入与读出操作交叉进行,即每写入一个数据后,接着就读出来检查,此时需要在写入数据之后,在回复 EXCS 为高电平之前,把 EXWE 信号从低电平变为高电平,此时芯片进入读操作过程,数据指示灯显示的就不再是开关的拨入数据,而是芯片的读出数据。接着首先恢复 EXWE 信号为高电平,之后立刻恢复 EXCS 为高电平,就可以启动芯片的下一次写入与读出操作。

④ 在关掉电源后, RAM 芯片中的已有信息会丢失。

2. ROM 芯片的读写操作实验

ROM 芯片的读写操作步骤与 RAM 芯片基本相同,差别是实验用到的器件不同。

① ROM 芯片有 28 个管脚,要用到 IC 座的 1、2、27、28 这 4 个管脚。接线过程中,需要把读写命令开关的输出接到 EXWE 插孔(27 管脚),读写命令为高电平时执行读操作,为低电平时执行写操作。

② IC 座的 23 管脚需要连接地址总线的 AB11。

③ 对 ROM 芯片执行读操作时,要求使能信号 EXOE 为低电平,执行写操作时,要求使能信号 EXOE 为高电平,这需要通过一个开关来切换。对 EEPROM 的写操作属于非常规用法,其写操作的速度比较慢。

④ ROM 芯片中的已有信息在关掉电源后不会发生变化。

按上述要求完成接线后,就可以参照 RAM 芯片的实验步骤开始 ROM 芯片的实验。

这项实验中,有两件事情必须处理好。

① 要妥善解决好开关拨入数据与存储器读操作争用数据总线 DB 的矛盾。开关拨数用到的 244 芯片有三态使能控制管脚 G(电路板上对应有一个接线插孔),当 G 信号为低电平时,244 芯片的输出为正常电平;否则 244 芯片输出为高阻态。把存储器的读写命令信号 MWE 用作 G 信号是最便捷的解决方案。写存储器时 MWE 应为低电平,把这个信号送到 244 的 G 管脚,就能把开关所拨的数据送到 DB,用作为存储器的写入数据;读存储器时 244 的 G 管脚和 MWE 应同为高电平,244 芯片的输出为高阻态,数据总线 DB 正好用于显示存储器的读出数据。

② 写存储器的过程中不能出现误操作,要防止破坏已正确写入芯片的内容。关键措施是在用开关拨数、拨地址的时刻,一定要保证提供给存储器芯片的片选信号为高电平,仅在检查数据和地址都正确之后,再把片选信号拨为低电平,进入芯片读写操作过程,之后立即恢复片选信号为高电平状态,转入下一次的读写操作过程。

结论:实验中可以发现,静态存储器芯片的读写操作是用不到时钟信号的,数据和地址有了之后,再给出正确的控制信号(片选、输出使能、读写命令),芯片读写操作就可以直接完成了,表明静态存储器芯片用于保存信息的触发器工作于电平触发方式。

这次实验用到的某些技术和概念,可以延伸到计算机控制器设计过程中的扩展新指令和指令调试的实验中,要求同学们较好掌握。

3.3 单独 MACH 芯片的运算器设计实验

【实验目的】

- 复习先修课程“数字电路和逻辑设计”的基本知识。
- 了解现场可编程的 isp MACH 芯片的内部硬件资源、管脚编号及其对应的信号名称。
- 学习硬件描述语言 ABEL 的程序结构、语法规则和程序设计的基础知识。
- 学习对 ABEL 程序的编译与综合的技术,对 .jed 类型文件的下载操作方法。

【实验内容与要求】

使用 ABEL 语言描述运算器部件的基本组成及其功能,在高集成度的 MACH 芯片中实现出来并完成调试和运行实验。

具体要求:在 MACH 芯片内设计实现一个 8 位的原理性运算器模型,即使用 ABEL 语言设计这个 8 位运算器的电路组成和功能,并把编译与综合的结果下载到 MACH 芯片,得到一个能够调试和运行的计算机运算器部件的雏形。这项实验涉及较宽的知识面,用到较

新的实验技术,要求同学认真完成,这也是开展后续各项实验的必要准备。

对这个运算器模型的功能要求如下。

(1) ALU 能完成加、减、与、或 4 种运算功能,需要使用 2 位的功能选择码 alu_f1、alu_f2 加以选择,由开关提供。ALU 要产生数据运算的值 Y 和结果的特征信息 carry(进位输出)以及 zero(结果为 0)。

(2) ALU 的两路输入数据被命名为 B 和 A,B 路数据可选择累加器 Acc 或常数 0,A 路数据可选择 8 位开关提供的拨入数据 D 或 D 的反码/D,为此 B 和 A 路输入都会用到一个二选一电路,以便确定 B 路数据选择 0(用于向 Acc 赋初值)还是 Acc(实现累加运算),A 路数据可选择 D(用于非减法运算)还是 /D(用于减法运算)。两个二选一电路的选择信号是 b_sel 和 a_sel,b_sel 由开关提供,a_sel 可在 MACH 芯片内直接生成。ALU 的输出可以送指示灯显示,还要连接到累加器 Acc 的输入端以便保存。

(3) 累加器 Acc 只接收 ALU 的计算结果,可以是 D+0 或 D op Acc(op 可以是 4 种运算功能中的任何一种,由 alu_f1 和 alu_f2 两位功能选择信号决定)。Acc 的内容可以经过三态门控制被送到 8 个指示灯予以显示。

依据上述要求,得到图 3.1 所示的该运算器模型的组成框图。

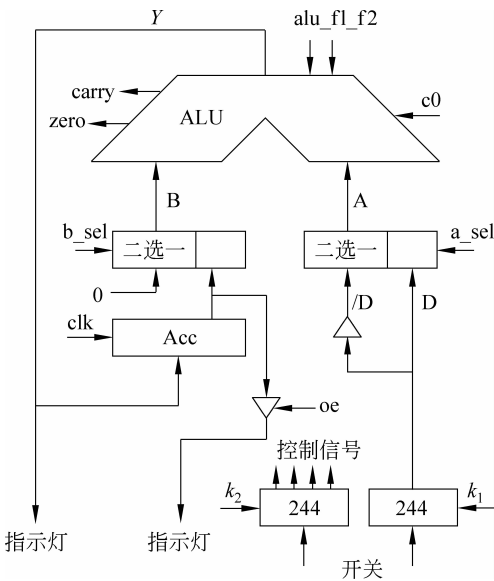


图 3.1 简单运算器模型组成框图

从图 3.1 中可以看到,用 8 个指示灯显示 ALU 的计算结果 Y,用另外 8 个指示灯显示 Acc 的内容,这里对 Acc 的输出显示加了三态控制。当 oe(由开关提供)为高电平时,显示的是 Acc;当 oe 为低电平时输出为高阻态,不再显示 Acc。

ALU 产生的两个标志位信息(carry 和 zero),也可以接到指示灯予以显示。

下面给出的是实现上述功能的 ABEL 语言的程序源码。

```
MODULE ispmach
TITLE 'simple alu'
"program alu8_16.abl 2014/10/06 ALU 串行进位
```

```

DECLARATIONS
clk                pin 68;
alu_f1,alu_f0,b_sel,oe pin 87..84;
D7..D0             pin 24,23,26,25,
                    28,27,30,29;
Y7..Y0             pin 32..39;
carry,zero         pin 80, 81;
Acc_7..Acc_0       pin 77..70;
B7..B0,A7..A0,a_sel node istance 'com';
c8..c1,c0          node istance 'com';
Acc7..Acc0         node istance 'reg,keep';
alu_f=[alu_f1,alu_f0];
A=[A7..A0]; B=[B7..B0]; D=[D7..D0];
Y=[Y7..Y0]; Acc=[Acc7..Acc0];
EQUATIONS
    when b_sel then B=Acc;
        else B=[0,0,0,0,0,0,0,0];
    when alu_f==[0,1] then {c0=1; a_sel=1;}
    when a_sel then A=!D; else A=D;
    Acc:=Y; Acc.CLK=clk;
    [Acc_7..Acc_0]=Acc;
    [Acc_7..Acc_0].oe=oe;

    when(alu_f==[0,0])#(alu_f==[0,1])then
    { Y0=B0&A0&c0 #B0&!A0&!c0 #!B0&A0&!c0 #!B0&!A0&c0
      Y1=B1&A1&c1 #B1&!A1&!c1 #!B1&A1&!c1 #!B1&!A1&c1
      Y2=B2&A2&c2 #B2&!A2&!c2 #!B2&A2&!c2 #!B2&!A2&c2;
      Y3=B3&A3&c3 #B3&!A3&!c3 #!B3&A3&!c3 #!B3&!A3&c3;
      Y4=B4&A4&c4 #B4&!A4&!c4 #!B4&A4&!c4 #!B4&!A4&c4;
      Y5=B5&A5&c5 #B5&!A5&!c5 #!B5&A5&!c5 #!B5&!A5&c5;
      Y6=B6&A6&c6 #B6&!A6&!c6 #!B6&A6&!c6 #!B6&!A6&c6;
      Y7=B7&A7&c7 #B7&!A7&!c7 #!B7&A7&!c7 #!B7&!A7&c7;
      c1=B0&A0 #B0&c0 #A0&c0;
      c2=B1&A1 #B1&c1 #A1&c1;
      c3=B2&A2 #B2&c2 #A2&c2;
      c4=B3&A3 #B3&c3 #A3&c3;
      c5=B4&A4 #B4&c4 #A4&c4;
      c6=B5&A5 #B5&c5 #A5&c5;
      c7=B6&A6 #B6&c6 #A6&c6;
      c8=B7&A7 #B7&c7 #A7&c7;}
    when alu_f==[1,0] then Y=B&A;
    when alu_f==[1,1] then
    { Y0=B0#A0; Y1=B1#A1; Y2=B2#A2; Y3=B3#A3;
      Y4=B4#A4; Y5=B5#A5; Y6=B6#A6; Y7=B7#A7; }

```

"说明入出信号和内部节点信号
"时钟信号
"输入控制信号 AB15~AB12
"输入开关数据 DB15~DB9
"显示 ALU 的结果 DB7~DB0
"显示 2 个特征位 AB9,AB8
"显示累加器内容 AB7~AB0
"ALU 的 2 路输入数据与选择控制
"ALU 每一位的进位入/出信号
;"8 位的累加器
"定义集合,用于简化逻辑方程式
"运算器的功能与电路描述,用了集合
"ALU 的 B 路数据选择
"b_sel=1 选 Acc,否则选 0 值
"执行减运算时 c0 为 1
"A 路数据选择,减运算选 !D,否则选 D
"累加器接收输入
"累加器内容送指示灯显示
"oe=1:显示累加器内容,oe=0:不显示
"ALU 的运算功能描述 00:+, 01:-, 10:&, 11:#
"加减运算
"逐位计算和/差
"并送指示灯显示
"逐位计算进位输出
"可以送指示灯显示
"与运算,用了集合
"或运算,未用集合
"Y 的逻辑方程较多

```
when(alu_f==[0,0])then carry=c8;
when(alu_f==[0,1])then carry=!c8;
when [Y7..Y0]==^h00 then zero=1;
END
```

"得到并显示 ALU 结果的两个特征位信息
"进位标志=1:加运算有进位
"减运算有借位
"零标志=1:ALU 的结果为 0

【实验环境和条件准备】

在实验之前,教师会在课堂上用少量学时讲解以下 3 项内容。

- (1) isp MACH 芯片内部的电路资源、功能特性和芯片的使用方法。
- (2) ABEL-HDL 硬件描述语言的程序结构、语法规则以及程序设计知识(见附录 A)。
- (3) 对 ABEL 程序进行编译综合、对芯片执行下载的方法和操作步骤,这里会用到 ISPLEVER 软件和 ispVM System 软件(在附录 B 中给出了这两个软件的使用方法和操作步骤),同学只需按照工作布置照办即可,要求学生能从实验中学到必要的设计知识和实验技术。

实验室在 PC 上准备好必要的实验环境,建立项目工程文件,安装好编译软件和下载软件,作为学习使用 ABEL 语言的实例,提供本次实验用到的 ABEL 语言源程序。

实验室为每一个实验小组提供一台教学实验箱系统。

【实验操作】

1. 接线操作

MACH 芯片被焊接到一块小的印制电路板上,这块小板是通过 4 个 20 脚的双排插针被插接到实验设备的主板上,使 MACH 芯片的 132 个 IO 管脚被接通到主板上不同接线插孔或插针。主板上是按照实现 16 位的教学计算机系统来布线的,各接线插针、插孔的命名也是依据 16 位机系统的电路功能来选择的。

在描述 8 位运算器电路及其功能的 ABEL 程序中,MACH 芯片的每个管脚是用这个管脚的编号来指定的,这些管脚对应的接插针(孔)在电路板上的位置在相应语句的注释部分给出,举例如下。

```
alu_f1,alu_f0,b_sel,oe pin 64..61;
D7..D0          pin 24,23,26,25,
                28,27,30,29;
Y7..Y0          pin 32..39;
Acc_7..Acc_0    pin 77..70;
carry,zero      pin 58,57;
```

"输入控制信号 AB15~AB12
"开关输入数据 DB15~DB9

"显示 ALU 的结果 DB7~DB0
"显示累加器内容 AB7~AB0
"显示两个特征位 flag_c,flag_z

这样选择是期望能简化接线操作,尽量选用那些本身接有指示灯的管脚,如 16 位的 DB、16 位的 AB,并确保不会与电路板上的已有器件发生冲突,如不能选用 16 位的 IR 管脚,因为用于实现 IR 的两片 377 器件的电源与 MACH 的电源是接在一起的,在为 MACH 芯片供电时,也会接通两片 377 器件的电源。

- (1) 连接 4 位开关输出到 AB 总线的最高 4 位,用于提供运算器的 4 位控制信号。
- (2) 连接 8 位开关输出到 DB 总线的高位字节,用于提供运算器的 8 位输入数据。

(3) 连接 V_{cc} 到 MIO 接线插孔(在 139 芯片的左下角),禁止内存和串行接口运行,为用到的两片 244 芯片的 G 管脚提供低电平的信号,使开关的输出信息能够送出。

2. 调试运行运算器

(1) 初步阅读实验室提供的 ABEL 源程序,试着对其进行编译和综合,产生 .jed 类型的结果文件。

(2) 试着把得到的 .Jed 文件的内容下载到 MACH 芯片中。

(3) 对选用 MACH 芯片实现的运算器进行运行调试。

① 为累加器 Acc 赋初值。

a. 拨入 4 位控制信号为 0、0、0、0,拨入 8 位输入数据为十六进制的 ^h25,观察并记录 ALU 的输出 Y、cy 和 zero 的值;观察累加器 Acc 的输出(应该无显示)。按一次 start 按键,变 oe 信号为高电平,再看各项显示内容是否有变化。

b. 依次变动 alu_f1、alu_f2 为 0、1(减功能),1、0(与功能),1、1(或功能),0、0(加功能),观察每种运算功能的计算结果并记录下来,判断运算结果是否正确,期间不能按 start 按键,只在最后按一次 start 按键,再次观察各项显示内容。

② 执行 Acc 内容和开关输入数据 D 之间的 4 种运算。

a. 变 b_sel 信号为高电平,保持 alu_f1、alu_f2、oe 为 0、0、1 不变,观察并记录各项显示内容的变动情况,按一次 start 按键,再次观察各项显示内容的变动情况。

b. 变 alu_f1、alu_f2 为 0、1,保持 b_sel、oe 为 1、1 不变,观察并记录各项显示内容的变动情况,按一次 start 按键,再观察各项显示内容的变动情况。

c. 参照前面的操作过程,分别变动 alu_f1、alu_f2 为 1、0 和 1、1,保持 b_sel、oe 为 1、1 不变,观察各项显示信息的变动情况,按一次 start 按键,再观察各项显示内容的变动情况。

③ 参照前面的操作过程,改用其他数据,观察 4 种运算功能的执行结果。

④ 综合前面完成的各项实验结果,判断该运算器的设计和运行是否正确。

⑤ 试着使用这个运算器模型计算机整数 1~10 的累加和。

实验之后,应该掌握以下内容。

(1) ALU 是组合逻辑电路,其计算的结果和特征信息直接取决于当前的运算数据和执行的运算功能,可以通过指示灯实时显示出来。请注意,运算器执行加减法运算使用的是数的补码,减法运算是使用加法器完成的,此时需要向 ALU 送减数的反码/D(而不是减数本身 D),并且要向 ALU 最低位的进位输入信号 c0 送入 1 值(而不是 0)。

(2) 累加器是有记忆功能的时序逻辑电路,Acc 在时钟信号的上升沿(时钟信号结束时刻)接收它的输入数据,必须为累加器指定时钟脉冲信号。

(3) 三态门电路有两种输出状态:或是它当前的输入信息(芯片正常工作),或处于高阻态(芯片不工作,不能向外输出信息,在逻辑上等同于其输出与系统断开连接),取决于它的输出使能信号,如 Acc 输出处的三态门的 oe 信号、244 芯片的 G 信号。

(4) 只有算术运算才产生进位输出信号 carry,逻辑运算时,数据的高、低位之间不存在进位关系。算术和逻辑运算都需要处理结果为 0 的特征信息 zero。

(5) 对 ABEL 语言与 MACH 芯片的功能和使用能够有基本了解,学会使用。

第4章

脱机的计算机部件实验

所谓脱机部件实验,指的是把某个部件从计算机整机系统中独立出来,即断开它与其他部件的连接关系,单独针对这个部件本身开展实验,更有利于准确深入地掌握部件本身的组成、运行原理、控制使用方法等内容,避免牵涉其他部件以及部件之间的交互作用。

4.1 脱机的运算器部件实验

【实验目的】

- 了解 Am2901 运算器芯片的内部组成、实现功能。
- 学习选用 4 片 4 位的 Am2901 运算器芯片组成 16 位运算器的知识。
- 掌握控制运算器完成算术和逻辑运算、移位操作的具体方法。

【实验说明】

运算器是承担数据运算、数据和结果暂存功能的部件,主要由算术逻辑运算单元 ALU、寄存器组 REG、标志寄存器 Flag 等 3 个部分组成,当然还有其他一些辅助电路。运算器部件由 4 片 4 位的 Am2901 芯片构成。在脱机部件实验中,需要为其提供输入数据,显示其运算结果(包括标志位信息),要用到通用的拨数开关和指示灯,需要完成必要的接线操作。最重要的是,学习、了解需要向运算器提供哪些和什么电平的控制信号。

本运算器运行要用到 21 位的控制信号,从左到右排列的信号是 A3~A0、B3~B0、I8~I1、I0、oe、RAM15、RAM0、c0,各字段所代表的内容如表 4.1 所示,其中的 I8~I0 实现的控制功能如表 4.2 所示。实验人员需要依据运算器完成的功能决定每一位信号的取值,并通过开关提供给运算器。在表 4.2 后面给出了使用这些控制信号实现 5 项功能的例子。

表 4.1 Am2901 运算器使用的控制信号

A3~A0	B3~B0	I8~I6	I5~I3	I2~I0	oe	RAM15、RAM0	c0
源寄存器	目的寄存器	结果保存	运算功能	数据来源	输出使能	移位输入	低位进位

表 4.2 I8~I0 这 3 组 3 位信号的控制作用

编码	I8~I6 寄存器接收	编码	I5~I3 运算功能	编码	I2~I0 数据选择	
					R	S
000	暂未选用	000	$S + R$	001	A	B
001	寄存器不接收	001	$S - R$	011	0	B
011	$F \rightarrow B$ 接收	011	S 或 R	100	0	A
101	$F/2 \rightarrow B$ 右移位接收	100	S 与 R	111	D	0
111	$F * 2 \rightarrow B$ 左移位接收	110	S 异或 R			

例如,要执行输入数据 $^h0305 \rightarrow R0$ 、 $^h0300 \rightarrow R1$ 、 $R0-R1 \rightarrow R0$ 、 $R1 \# R0 \rightarrow R1$ 、 $R1$ 的内容逻辑右移一位这 5 项功能,需要提供如表 4.3 所示 5 组 21 位的控制信号。

表 4.3 5 组控制信号

A3~A0	B3~B0	I8~I6	I5~I3	I2~I0	oe	RAM15	RAM0	c0	说 明
0000	0000	011	000	111	0	0	0	0	“ $^h0305 \rightarrow R0$ ”
0000	0001	011	000	111	0	0	0	0	“ $^h0300 \rightarrow R1$ ”
0001	0000	011	001	001	0	0	0	1	“ $R0-R1 \rightarrow R0$ ”
0000	0001	011	011	001	0	0	0	0	“ $R1 \# R0 \rightarrow R1$ ”
0000	0001	101	000	011	0	0	0	0	“ $R1$ 右移 $\rightarrow R1$ ”

注: oe 是 ALU 输出的使能控制信号,为低时允许输出,为高时输出为高阻态。 h 后跟的是十六进制数。

【实验内容】

在进行脱机运算器部件实验时,需要用开关向运算器提供运算数据和运行控制信号,使其完成加、减、与、或、寄存器内容右移、左移等功能,并通过指示灯查看运算器的运算结果,有利于加深理解课堂授课内容,掌握使用运算器部件的基础知识与技术,为构建计算机整机系统奠定初步基础。

要求从以下给出的多项操作功能中选择有兴趣的 8 项进行实验,数据内容可以变化,每一项功能可以执行一次或连续执行几次。观察各种情况下的运行结果并判断运行结果的正确性。

$^h1234 \rightarrow R1$

$^h2345 \rightarrow R2$

$R1 + R2 \rightarrow R2$

$R2 - R1 \rightarrow R2$

$R1$ 内容左移一位

$R2$ 内容左移一位

$R1 \rightarrow R3$

$R2$ and $R1 \rightarrow R2$

$R3$ or $R2 \rightarrow R3$

$R1 + 1 \rightarrow R1$

$R2 - 1 \rightarrow R2$

显示一个指定寄存器的内容。

【实验步骤】

实验过程中,只需接通4片运算器芯片的电源,系统要处于单步骤运行方式,每按一次 start 按键,运算器完成一次运算、处理功能。

1. 接线操作

(1) 关闭 MACH、Am2910、FPGA 的电源,禁止这3个芯片运行。

连接 V_{CC} (+5V 电源) 和 MIO, 为 MIO 提供高电平, 禁止存储器和串行口工作。

此时能正常运行的只有运算器部件和电路板上的辅助电路。

(2) 连接两组8位开关的输出到运算器的16个数据输入管脚(即数据总线 DB), 以便为运算器提供输入数据, 并向这两组开关用到的244芯片的输出使能信号 G 管脚提供恰当的电平, 保证开关拨数到 DB 和送 ALU 的输出到 DB 的互斥关系。

在 $G=0$ 、 $oe=1$ 时, 允许用开关拨入数据, 禁止送 ALU 输出到 DB。

在 $G=1$ 、 $oe=0$ 时, 禁止用开关拨入数据, 允许送 ALU 输出到 DB。

(3) 连接24位开关的输出到运算器的21位控制信号管脚, 对应的是设置在电路板上 MACH 芯片下方的24位的接线排针(其中最低3位空闲未用), 各信号的排列次序给出在表4.1中。

2. 控制运算器执行选定的功能

在系统处于单步骤运行的方式下, 每按一次 start 按键, 启停电路将向运算器提供一个时钟信号, 运算器结束一次运算或操作功能, 并转到下一次的功能操作。

Am2901 运算器是可以在一个时钟周期完成一次运算的, 包括取得运算数据(开关送来的或从通用寄存器读出的), 得到运行需要的控制信号, 完成指定的运算功能, 并把结果保存到选中的寄存器中。因此, 这里的运算器脱机实验就是拨动开关并通过指示灯观察运算器的执行结果的过程。实验中的每项操作功能都要通过按 start 键来结束。

4.2 脱机基本存储器部件实验

【实验目的】

- 了解存储器部件的字位扩展原理, 使用存储器芯片构建存储器部件的技术。
- 了解存储器部件的读写过程, 控制其运行的方法和操作步骤。

【实验说明】

存储器是计算机系统中承担存储程序和数据部件, 通过字位扩展技术, 选用两片随机读写的 RAM6116 和两片只读的 ROM58C65 存储芯片, 来构建字长 16bit 的基本存储器部件, 还另外设置了两个 28 脚的 IC 插座, 可以插接 ROM58C65 或 RAM6116 芯片, 用于进一步扩展基本存储器的存储容量。

在完成存储器芯片读写实验的过程中已经看到, 存储器可以执行读、写两种操作, 每次的读、写都要用两段时间完成, 会用到3种控制信号, 即读写命令 MWE、片选信号 MCS、使

【实验操作】

1. 接线和必要说明

(1) 连接 16 个拨数开关到地址总线 AB, 用于提供存储器的单元地址。

(2) 连接 16 个拨数开关到数据总线 DB, 用于提供存储器的写入数据。对于写操作, 输入数据将被写入到存储器的选定单元; 对于读操作, 读出的数据将出现在数据总线 DB 并通过指示灯予以显示。

(3) 连接 3 个开关到 MIO、REQ、WE 接线处(在 139 译码器芯片的下方), 经过 3 片译码器芯片(图 4.1)可以产生存储器和串口芯片读写的片选信号和控制信号。

在进行本实验时, 需要使 REG 保持低电平(使用存储器, 停用串行口), 并使用 MIO 信号的高、低电平两个状态来区分一个存储周期的地址时间和数据时间。通常只能在地址时间使用开关向存储器拨入地址、写入数据和控制信号。存储器芯片的实际读写操作将在数据时间段完成。读存储器的期间必须禁止送开关数据到 DB。

(4) 实验过程中, 需要关闭 MACH 芯片、4 片 Am2901 芯片、Am2910 芯片和 FPGA 芯片的电源, 系统要处于单步骤运行方式。

2. 读出并显示 ROM 存储区的指令内容

ROM 存储区用于存储教学计算机的监控程序, 仅支持读操作功能, 不能执行写操作(芯片的 WE 和 OE 管脚已分别接 +5V 和 GND), 以防止破坏监控程序的原有内容。

为了对 ROM 存储区执行读操作, 需要在地址时间段用开关为芯片提供 16 位的地址和 3 位的控制信号(MIO、REQ、WE=1、0、X), 地址的最高 3 位取 0、0、0 值(以便产生 ROM 芯片的片选信号 mcs0), 再拨入 ROM 芯片的 13 位地址; 再拨开关使 MIO 为低电平则进入数据时间段, 芯片的读出内容就显示在 DB 的指示灯。

接下来可以在数据时间段直接变动低 13 位的地址, 选中单元的内容就会立刻显示在 DB 的指示灯, 这是一种变通的操作方法, 比正规的做法(在地址时间段拨开关, 在数据时间段执行读操作)更快捷。请思考为什么可以这样操作? 在拨入低位地址的过程中, DB 的指示灯会不断变化, 多次的显示信息可能不是你期望要观察的内容。

3. 读写 RAM 存储区的操作

RAM 存储器能支持读和写的两种操作, 为了避免或者减少误操作, 对 RAM 区的每一次读写操作最好都用地址和数据两段时间来完成(MIO 为高电平是地址时间段, 为低电平是数据时间段)。请注意, 在读存储器期间必须禁止送开关数据到数据总线 DB, 这只需把存储器芯片的 WE 信号和拨数据到 DB 用到的两片 244 芯片的 G 信号连接到同一个开关的输出即可, 使这两个信号总保持相同的值。

在读 RAM 存储区时, 需要在地址时间段用开关向地址总线的 AB15~AB13 拨入 0、0、1 值(以便产生 mcs1 片选信号)和 3 位的控制信号(MIO、REQ、WE=1、0、1), 再拨入 RAM 芯片的 11 位地址; 然后切换到数据时间(为 MIO 拨入低电平)就能观察到芯片的读出内容。接下来可直接变动低 11 位地址来观察 RAM 不同存储单元的内容。

在写 RAM 存储区时, 需要把 16 个数据开关的输出连接到数据总线 DB, 用于提供写存储器的写入数据。可靠的办法, 是在地址时间段(MIO 为高电平)拨入存储器的单元地址、写入的数据内容, 以及 3 位的控制信号(MIO、REQ、WE=1、0、0), 之后变更 MIO 为低电平

(进入数据时间段),使存储器完成写操作;接着恢复 MIO 为高电平状态,进入下一次的数据写入过程,多次重复就可以把一批数据写入 RAM 存储区。之后再用读 RAM 存储区的方式检查此前的写入操作和正在执行读操作是否都正确执行。当然也可以通过每写完一个数据之后立即将其读出来的读写交替方式,来检查读、写操作的正确性。

4.3 脱机的串行接口读写与输入输出实验(选做)

【实验目的】

- 了解串行接口芯片的内部组成、功能和使用方法。
- 检查串口芯片是否和仿真终端正确连接,双方能否正常通信并得到正确运行结果。

【实验说明与实验内容】

教学机中的串口是连接 CPU 和仿真终端设备并执行输入输出操作的电路。CPU 和串行接口之间以 8 位并行方式交换信息,而串行接口和设备之间则以逐位串行方式交换信息。在设备上把两片 Intel 8251 芯片用作两路串行接口,都能连接 PC 仿真终端设备。输入输出实验需要把串行口与仿真终端连接起来才能完成,单个的串口芯片实验是没有意义的。串口芯片的 8 位数据管脚连接到系统数据总线 DB 的低位字节,数据可在两个方向进行传送(串口 $\leftrightarrow$ DB)。

通过串口执行输入输出的实验属于选做而不是必做项目,其操作会涉及某些略显复杂的概念,需要初步了解以下有关知识和相关技术。

(1) 输入输出操作不只是教学计算机一方的事情,还要用到 PC 仿真终端,双方之间需要通过串口数据线连接起来,还要求仿真终端一方的接口、键盘输入和屏幕显示都能正常工作,并且与教学机系统保持正确的时序配合关系。

(2) 教学机和仿真终端之间传送信息要求使用电平更高的信号,双方都会用到一个电平转换电路,如一片 MAX202 芯片,该芯片可同时用于两路串行接口。串行口要用到几个不同频率的脉冲信号,已经连接好。

(3) CPU 对串口执行读、写操作时,要求向串口芯片提供 4 个控制信号,即读命令 `iord`、写命令 `iowr`、片选信号 `iocs` 及信息类型信号 `c_d`,前 3 个信号都是低电平有效。

`c_d` 是串口端口地址的最低位,用于区分读写串口的信息类型,为 0 时,读写的内容是数据,为 1 时,读出的是串口的状态信息,写入的是初始化串口的控制信息。

(4) 在教学机和仿真终端双方的接口之间的信息交换是自动执行的,CPU 不能直接控制仿真终端,它能做的只限于写、读自己一方的串行接口芯片。

写串口对应的是数据的输出操作,是把运算器 R0 寄存器中的一个字符的 ASCII 码送到串口的发送数据缓冲器,之后串口会以串行方式将其发送到仿真终端一方。

读串口对应的是数据的输入操作,是把已保存在接收数据缓冲器中的一个字符的 ASCII 码(是仿真终端在此之前以串行方式传送过来的)读出来并写入 R0 寄存器。

(5) 在使用串口之前要完成芯片的初始化操作,确定芯片的运行模式和控制参数,该初始化只能紧跟在系统复位(按 `reset` 键)之后进行,并且仅能执行一次,要求向串口写入初始

化信息[^]h4E 的操作仅被执行一次;否则初始化操作不能正确完成。

(6) 在进行脱机的串口实验时,每次读写串口通常要用两段时间完成,可以用 MIO 信号的高、低电平的两个状态来区分这两个时间段,在地址时间用开关拨入串口的写入数据、端口地址和运行控制信号,在数据时间完成串口芯片的读写操作。

为此可以把系统时钟信号 CLK 用作 MIO(连接 CLK 到 MIO),在单步骤运行方式下,每按一次 start 按键,启停控制电路会发出唯一的一个完整的脉冲信号,其电平变化是从高到低再返回到高的一个过程,能很好地满足实验要求。

(7) 为了确保在串口执行读操作期间不传送数据开关的输出到数据总线 DB,选用同一个开关来提供 WE 信号(串口读写命令)和送开关信息到 DB 用到的两片 244 芯片的 G 信号(输出使能信号)是合理又简便的解决方案。

【实验操作】

1. 接线和启动操作

(1) 用串口数据线连接教学机串口和仿真终端串口(或 USB 口);运行 PC 系统中 pcecl6.com 程序,按两次 Enter 键后,PC 进入仿真终端状态,成为教学机的输入输出设备。

若希望检查仿真终端本身是否运行正常,可以把串口数据线从实验设备上拔下来,再把 DB9 插头的数据发送线插孔和数据接收线插孔用 1 号线短接起来,使仿真终端送出来的信息又被直接接收回去,则按下键盘上的任何一个字符键,相应字符就会被显示到屏幕上,表明仿真终端本身运行正常。

(2) 连接 8 位拨数开关到 DB 的低位字节,为串行口提供写入数据(D7~D0)。

(3) 连接 8 位拨数开关到 IR 输出的低 8 位排针,为串口提供端口地址(IR7~IR4)和 c_d(IR0),对 IR7~IR4 译码就得到串口的片选信号 iocs, c_d 用于确定信息类型。

(4) 连接 CLK1 到 MIO 的接线插孔,连接 2 位开关到 REQ 和 WE 的接线插针,为 139 译码器提供 3 位的输入信号。实验中使 REQ 保持恒高(使用串口,停用存储器)。

(5) 实验开始前,需要关闭 MACH 芯片、4 片 Am2901 芯片、Am2910 芯片和 FPGA 芯片的电源,并设置系统为单步运行方式。

2. 对串口的写操作

用于把 8 位开关数据写入串口芯片。

1) 执行串口芯片初始化

(1) 按 reset 按键,首先对系统进行复位,接下来开始串口芯片的初始化操作。

(2) 拨入端口地址[^]h81 和初始化信息[^]h4E, REQ、WE 选 1、0,按一次 start 按键。

(3) 保持端口地址[^]h81 和 REQ、WE 为 1、0 不变,拨入初始化信息[^]h37,按一次 start 按键;至此已经完成了串口的初始化操作。

2) 执行数据输出操作

(1) 变串口的端口地址为[^]h80,保持开关数据[^]h37 和 REQ、WE 为 1、0 不变,按一次 start 按键,则在仿真终端的屏幕就可以显示出数字 7,一次数据输出已经完成。

(2) 之后可以改变开关的拨入数值(字符的 ASCII 码),每按一次 start 按键,都会在仿真终端的屏幕上显示出对应这个 ASCII 码的字符,新的一次数据输出已完成。重复步骤(2)则可输出一批数据到仿真终端。

3. 对串口的读操作

用于读出串口芯片中的 8 位数据。

执行的是数据输入操作,即把串口中接收数据缓冲器的 8 位数据读出来,经过数据总线 DB 的低 8 位保存到 8 位的寄存器(若不保存则该数据会丢失),此处的难点是暂无这个寄存器可用,只能把读出的数据(一个字符的 ASCII 码)送到 DB 指示灯显示,供实验者查看。出于能看清楚的要求,信息显示的时间必须相对长一些,若此时还把 CLK 用作 MIO 就办不到了(信息显示时间仅半个时钟周期),改用中断按钮的输出(如 INT5)替代 CLK 就能解决此问题,可以通过改接 INT5 到 MIO 来实现。

4. 操作步骤

(1) 在不按下中断按钮时 MIO 为高电平,串口芯片不工作,可拨入 REQ、WE 为 1、1 值。此时可通过敲击键盘上的一个字符键,传送相应字符的 ASCII 码到串口芯片。

(2) 按下并不松开中断按钮时 MIO 维持低电平不变,串口芯片进入读操作时间,读出的一个 ASCII 码被显示在 DB 指示灯,可以查看它的显示内容。

(3) 松开中断按钮后 MIO 回到高电平,就进入下一次的数据输入操作步骤,可以继续输入不同的字符并观察字符的 ASCII 码。