计算机体系设计实践报告

2021 学年秋季学期

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **序号** | **学号** | **姓名** | **专业** | **班级** | **成绩** |
| 245 | 20211060245 | 陈俊宏 | 人工智能 | 人工智能 |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

指导教师：武丽雯

软件学院 计算机体系设计实践 王逍

二 O 二二年十二月

# 计算机体系设计实践

**实验目的**

实践课设计 4 个实验，基本涵盖了计算机体系结构与组成课程的重点内容，实验电路的 设计与教材保持一致，实验电路在 FPGA 实验板上实现，并使用调试软件 JULAB 完成实验 的调试与分析。

**实验要求**

实验课前复习相关原理，按实验指导书的要求认真预习。 实验时独立思考，掌握实验设备或软件的构造和操作方法，按实验指导书要求设计或验

证实验内容，测试有关数据，分析相应的问题。

实验课结束时需要整理提交 fpga 工具箱，并检查工具箱附件是否缺失。

由于 fpga 工具箱数量有限，实验以小组形式完成，2 人 1 个小组，提交 1 份实验报告。

**实验资料环境和资料**

Win10+FPGA 设计软件 Quartus II、实验调试软件 Julab、实验相关设计文件（工程模板、 电路设计源文件、虚拟构图文件）等。请找实验知道老师索取。

**实验地点**

计算机硬件实验室：软件学院大楼 1428 室

# 实验 4 微程序控制器

实验操作

1. 下载实验资源

在实验指导页面下载通用文件“DE2-115 工程模板”解压缩到 E 盘或 F 盘。 将通用文件“DE2-115\_proj”解压缩到 E 盘或 F 盘。得到 DE2-115 工程文件夹。将微程

序控制器中文件解压到 DE2-115 工程文件夹中；

将 lab3 中文件解压，将其中的运算器 *ALU.v*、寄存器模块 *R.v*（运算通路中的 A 暂存器 和 PSW 标志寄存器由该模块实例化得到）、通用寄存器组 *GRS.v*、移位寄存器 *Shifter.v*，四 个文件解压到到 DE2-115 工程文件夹中。

实验电路顶层文件 *Lab\_Top.v*、微地址形成模块 *uAG.v*、时序发生器模块 *Sequencr.v*、控 制存储器模块 *ControlMemory.v* 及它的 IP 核文件、寄存器模块 *R.v*（微指令寄存器 uIR、微 地址寄存器 uAR、指令寄存器 IR 都由该模块实例化得到）。

*lab7.vpl* 和 *lab7.bmp* 是留给实验调试软件使用的虚拟面板构图文件。*Lab7\_CM.mif* 是 控制存储器的初始化文件。

2. 实验电路设计与下载

在工程文件夹 DE2-115 中双击工程文件 *DE2\_115\_Lab.qpf* 打开实验电路的 QuartusII 工 程。

对工程进行全编译（Start Compilation）按钮，自动完成分析综合、布局布线、生成编程 文件等整个过程，全编译完成后，点击工具栏中的编程按钮（Programmer），将生成的实验 电路文件 *DE2\_115\_Lab.sof* 下载到实验板。

3. 实验电路功能验证

打开实验调试软件 JULAB3，选择逻辑部件实验类型，在“虚拟实验板”菜单的面板构 图选项下，浏览选择工程文件夹中的 *lab7.vpl* 文件，打开本实验的虚拟面板，根据实验原理， 控制虚拟面板的开关、按键，观察对应的指示灯，在实验报告册中填写实验结果记录和分析。

## 实验记录

1. 取指令微程序设计

取指令是任何指令执行的第一个阶段。实验电路复位时，微指令寄存器 uIR 清零，微地 址形成模块 uAG 输出 00H 给控制存储器的地址，因此第一条微指令要存放在控制存储器的 00H 地址单元，即取指令微程序的入口地址从 00H 开始。

指令寄存器的内容由开关提供，因此取指令微程序只需要设计一条微指令用来产生指令 寄存器的时钟使能信号 IRce，即微指令字段 F1 编码为 100B，同时使用固定转移方式（BM=0）

根据 NA 字段产生下一条微指令的微地址 01H。 针对指令系统中装数和运算两类指令，指令寄存器 IR 取到指令后，指令执行流程应根

据指令操作码决定是否需要取目的操作数实现两分支转移，设计第二条微指令实现两分支转 移转移（BM=1），即 F4 字段为 01B，F5 字段可以任意，考虑到地址的连续性，设置 NA 为 000010B。

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 指令执行 阶段 | 微地址  (H) | 微指令(H) | 微指令字段 | | | | | | 微命令 |
| F0 | F1 | F2 | F3 | F4 | F5 |
| 取指令 | 00 | 10001 | 000 | 100 | 0000 | 00 | 00 | 000001 | IRce |
| 01 | 00042 | 000 | 000 | 0000 | 00 | 01 | 000010 | BM1 |

使用 Quartus II In-System Memory Content Editor 工具将微程序输入到控制存储器中， 具体操作参阅实验指导书第五章 5.1.3 在系统存储器数据编辑器。

将指令寄存器输入端的开关 I9~6 设置为全 0，执行上面的取指令微程序，将结果填入下 表；每条微指令的执行需要 2 个周期，故用两行记录。表中“有效的控制信号”一栏填写点 亮的指示灯所对应的控制信号名称，如 IRce。

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | RESET | Clock | I9~0 | CP1 | CP2 | µAR | CMdata(H) | BM | NA | 有效的控制信号 |
|  |  | —— | 0000000000 | 0 | 0 | —— | —— | —— | —— | —— |
| ① | 0 |  | —— | 0 | 1 | 00H | 10001 | 0 | 0 | —— |
|  | 0 |  | —— | 1 | 0 | —— | —— | —— | —— | IRce |
| ② | 0 |  | —— | 0 | 1 | 01H | 00042 | 1 | 1 | BM1 |
|  | 0 |  | —— | 1 | 0 | —— | —— | —— | —— | —— |
| ③ | 0 |  | —— | 0 | 1 | 02H | —— | —— | —— | —— |

实验结果分析：

复位时，CP2=　0　，CP1=　0　，因此微指令执行过程中，Clock 时钟信号到来后， 首先出现的是　CP2　（CP1/CP2）的上升沿。

第①条微指令执行时，µAR 和控存输出 CMdata 的变化发生在　CP1　（CP1/CP2）变高 的时候，表明　CP2　（CP1/CP2）将微地址打入 µAR，启动从控存读出微指令的操作；控 制信号 IRce 的变化发生在　CP2　（CP1/CP2）变高的时候，表明　CP1　（CP1/CP2）将控 存输出的微指令打入 µIR，开始执行这条微指令。

第②条微指令的 CP1 为 1 时 uAR=　01H　，表明第①条微指令的微地址转移方式为　计数器方式　，微转移地址由　NA　 （NA/NA 及 IR）决定。

表格第③条的设计是为了观察第②条微指令产生的微地址，CP1 为 1 时 uAR=　02H　，表明第②条微指令的微程序转移方式为　分支转移　，微转移地址由　NA及IR　（NA/NA 及 IR）决定，取到的指令是　运算指令　（装数指令/运算指令），微程序将进入　装数指令执行阶段　（取目的操作数阶段/装数指令执行阶段），入口地址为　00042H　。 将指令寄存器输入端的开关 I9~6 设置为不全 0，复位后，重新执行取指令微程序，取到的指令是　运算指令　（装数指令/运算指令），在第③步 CP1 为 1 时 uAR=　02H　，微程序将进入　取目的操作数阶段　（取目的操作数阶段/装数指令执行阶段），入口地址　00042H　。 如果想将装数指令执行阶段和取目的操作数的微指令安排在 08H~09H 地址，01H 地址的微指令的 NA 字段应该改成　08H　。

2. LD R1，#0101B

（1） 指令编码

将指令 LD R1，#0101B 翻译成二进制机器码。根据指令格式和表 3.3 指令操作码编码 表，LD 指令的操作码 OPCODE（ IR9~IR6）是 0000B，INDEX（IR5~IR4）是 01B，DATA

（IR3~IR0）是 0101B，因此翻译出指令机器码是　0000010101　B，使用开关将二进制机器码 送到指令寄存器 IR 的数据输入端。

（2） 微程序设计

取指令微程序已经在前面的任务中完成，下表只包含执行和观察阶段的微指令。设计微

程序并输入到控制存储器中。

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 指令执行 阶段 | 微地址  (H) | 微指令(H) | 微指令字段 | | | | | | 微命令 |
| F0 | F1 | F2 | F3 | F4 | F5 |
| 执行 | 02 | 64006 | 11 | 001 | 0000 | 00 | 00 | 000110 | DATAoe　　GRSce |
| 观察 | 06 | 20000 | 10 | 000 | 0000 | 00 | 00 | 000000 | GRSoe |

（3） 微程序的执行结果记录

复位后运行 LD 指令微程序，将结果填入下表。

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | RESET | Clock | CP1 | CP2 | µAR | CMdata | 有效的控制信号 | BUS | INDEX |
|  |  | —— | 0 | 1 | —— | —— | —— | —— | —— |
| ① | 0 |  | 1 | 0 | 00H | 10001H | —— | —— | —— |
|  | 0 |  | 0 | 1 | —— | —— | IRce | —— | —— |
| ② | 0 |  | 1 | 0 | 01H | 00041H | IRce | —— | 01 |
|  | 0 |  | 0 | 1 | —— | —— | —— | —— | 01 |
| ③ | 0 |  | 1 | 0 | 02H | 64006H | —— | —— | 01 |
|  | 0 |  | 0 | 1 | —— | —— | DATAoe　　GRSce | 5 | 01 |
| ④ | 0 |  | 1 | 0 | 06H | 20000H | GRSce　　DATAoe | 5 | 01 |
|  | 0 |  | 0 | 1 | —— | —— | GRSoe | 5 | 01 |

实验结果分析：

　　第③条微指令的 CP2 为 1 时，INDEX=　01　，DATAoe=　1　，GRSce=　1　，BUS=　5　，也就是将 DATA 的内容送到总线上，寄存器 R 将在　CP2　（CP1/CP2）变高的时候，保存总线上的内容。

3. 改变 LD 指令操作码

将 LD 指令的操作码 OPCODE 改为 1111B，需要将 uAG.v 代码的第 行修改为

。 完成代码修改后，重新编译 QuartusII 工程并下载，试一试修改后的 LD 指令在取指令结束后能否转移到 02H 微地址正确运行。

4. ADD R1，#0111B

（1） 指令编码

将指令 ADD R1，#0111B 翻译成二进制机器码。根据指令格式和表 3.3 指令操作码编 码表，ADD 指令的操作码 OPCODE（ IR9~IR6）是　0001H　，INDEX（IR5~IR4）是　01　， DATA（IR3~IR0）是　0111　，因此翻译出指令机器码是　0001010111　B，使用开关将二进制 机器码送到指令寄存器 IR 的数据输入端。

（2） 取目的操作数的微程序设计

取目的操作数指将寄存器 Ri 的值取出后保存在 A 中，由指令的 INDEX 字段指定寄存 器，因此设计一条微指令产生控制信号 GRSoe 和 Ace。考虑到地址的连续性，下一条微指 令的微地址设计为 04H，即设置 BM 为 00B，NA 为 000100B。

目的操作数取到以后，需要根据指令操作码生成各条运算类指令执行阶段的微程序入口 地址，因此接下来设计的一条微指令，用于实现多分支转移（BM＝2），即 F4＝10，F5 字 段在多分支转移方式下不影响微地址生成，可以为任意值。在 uAG 模块中实现 BM＝2 的代 码是　第35、36行　，据此可计算出各运算类指令执行阶段的微程序地址范围是004H~007H　。

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 指令执行 阶段 | 微地址  (H) | 微指令(H) | 微指令字段 | | | | | | 微命令 |
| F0 | F1 | F2 | F3 | F4 | F5 |
| 取目的 操作数 | 03 | 28004 | 01 | 010 | 0000 | 00 | 00 | 000100 | GRSoe　　Ace |
| 04 | 00080 | 00 | 000 | 0000 | 00 | 10 | 000000 | BM2 |

（3） ADD 指令执行阶段的微程序设计

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 指令执行 阶段 | 微地址  (H) | 微指令(H) | 微指令字段 | | | | | | 微命令 |
| F0 | F1 | F2 | F3 | F4 | F5 |
| 执行 | 11 | 6c705 | 11 | 100 | 0001 | 11 | 00 | 000101 | DATAoe　PSWce　ADD |
| 保存 | 05 | 44006 | 10 | 001 | 0000 | 00 | 00 | 000110 | Soe　　GRSce |
| 观察 | 06 | 20000 | 01 | 000 | 0000 | 00 | 00 | 000000 | GRSoe |

取指令和取目的操作数的微程序在前面的任务中已经输入控制存储器，继续使用

Quartus II In-System Memory Content Editor 工具将后续执行等阶段的微程序输入到控制存 储器中。

（4） 微程序的执行结果记录

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | RESET | Clock | µAR | CMdata | 有效的控制信号 | A | BUS | S\_Q | PSW | INDEX |
|  |  | —— | —— | —— | —— | —— | —— | —— | —— | —— |
| ① | 0 |  | 00 | 10001 | —— | —— | —— | —— | —— | —— |
|  | 0 |  | —— | —— | IRce | —— | —— | —— | —— | —— |
| ② | 0 |  | 01 | 00042 | IRce | —— | —— | —— | —— | 01 |
|  | 0 |  | —— | —— | —— | —— | —— | —— | —— | 01 |
| ③ | 0 |  | 03 | 28004 | —— | —— | —— | —— | —— | 01 |
|  | 0 |  | —— | —— | GRSoe　　Ace | —— |  | —— | —— | 01 |
| ④ | 0 |  | 04 | 00080 | GRSoe　　Ace |  | —— | —— | —— | 01 |
|  | 0 |  | —— | —— | —— |  | —— | —— | —— | 01 |
| ⑤ | 0 |  | 11 | 6c705 | —— |  | —— | —— | —— | 01 |
|  | 0 |  | —— | —— | DATAoe　PSWce |  |  | —— | —— | 01 |
| ⑥ | 0 |  | 05 | 44006 | DATAoe　PSWce |  |  |  |  | 01 |
|  | 0 |  | —— | —— | GRSce　Soe |  |  |  |  | 01 |
| ⑦ | 0 |  | 06 | 20000 | GRSce　Soe |  |  |  |  | 01 |
|  | 0 |  | —— | —— | GRSoe |  |  |  |  | 01 |

实验结果分析：

前面任务的 LD 指令完成后，R1 寄存器中的值为 0101B，微程序执行完后，R1 寄存器 中的值应该是　1100B　。反复调试执行 ADD 指令的过程中，可能使 R1 寄存器的值发生变 化，观察加法指令结果时注意以当次执行过程中从 R1 寄存器取到 A 寄存器中的值为准。

第③条微指令将 R1 的内容送到 A 暂存器，但是 A 暂存器内容的变化发生在 µAR= 时的　CP1　（CP1/CP2）上升沿，说明　下一条　（当前／下一条）微指令地址打入 µAR 的同时，　当前　（当前／下一条）微指令的执行结果打入寄存器保存。PSW 和 SHIFTER 的变化发生在　CP1　（CP1/CP2）变高的时候，表明　CP1　CP1/CP2）将微指令的执行结果打入运算器数据通路中的寄存器保存。

和实验 3.3 手动产生控制信号相比，用微指令产生控制信号更要注意时序，哪些信号应 该在一条微指令中产生、哪些信号不能同时产生。从上面的实验可以看出，完成一次 ALU 运算需要　5　个步骤。

仿照上述步骤，验证其它运算类指令。

5. 修改微地址分配

将 uAG 代码的第 行修改为 ，使运算 类指令执行阶段的微程序安排在 21H~2FH。完成代码修改后，重新编译 QuartusII 工程并下 载；设置指令寄存器 IR 的输入 I9~6 为 ，复位后重新执行，取目的操作数完成后， 微程序转移到地址 。

6. 修改指令系统（选做）

增加寻址方式，使得源操作数不仅可以来自于立即数，也可以来自于寄存器，例如可以 实现指令：

ADD R1，R2

修改指令格式如下：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 10 | 7 | 6 5 | 4 | 3 0 |
| OPCODE |  | INDEX | M | DATA／INDEX2 |

其中 M 用于表明源操作数是来自寄存器（由 IR1~0 指定）还是立即数。

提示：需要修改实验电路硬件，如修改 IR 寄存器，修改 uAG 代码以增加依据源操作数 的两分支微转移方式，增加微命令选择寄存器号来自于 INDEX 或 INDEX2。

重画指令执行流程图，把取立即数从执行阶段分离出来，增加取源操作数阶段。 设计微程序，运行微程序，记录执行结果。

## 实验小结及实验分工

实验分工：单人完成

答：由于疫情返乡的原因，无法到实验室进行完整的实验，但是在对实验原理有一定程度的了解之后，加上由老师提供的实验数据的协助，虽然较为艰难，但仍然可以得出一些实验结论以及数据，总的来说实验不算困难，只要足够细心和耐心即可完成。