**数字电路与逻辑设计**

**单周期CPU设计**

学院：数据科学与计算机学院

专业：软件工程

班别：教务二班软工4班

姓名：郑佳豪

学号：16305204

时间：2018年7月8号

单周期CPU

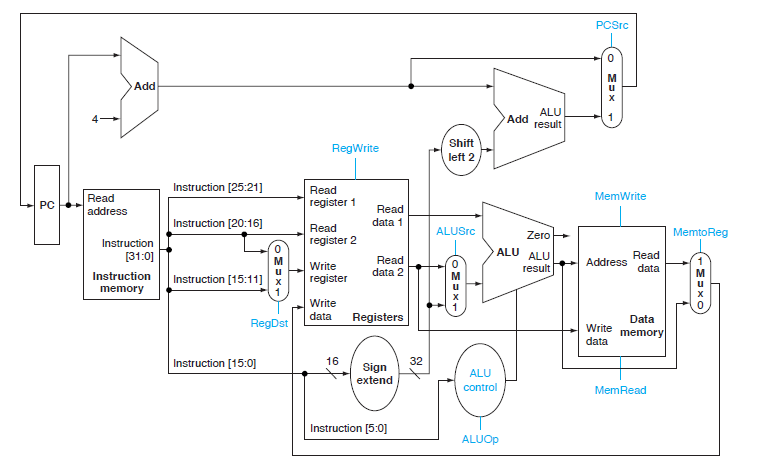
概念

单周期CPU指的是一条指令的执行在一个时钟周期内完成，然后开始下一条指令的执

行，即一条指令用一个时钟周期完成。电平从低到高变化的瞬间称为时钟上升沿，两个相

邻时钟上升沿之间的时间间隔称为一个时钟周期。

工作原理



根据上图，我们知道CPU在处理指令时，一般需要经过以下几个步骤

取指令(IF)：根据程序计数器(PC)中的指令地址，从存储器中取出一条指令，同时PC根据指令字长度自动递增产生下一条指令所需要的指令地址，但遇到地址转移指令时，则控制器把转移地址送入PC，当然得到的地址需要处理才送入PC。

指令译码(ID)：对从PC中取到的指令进行译码，产生相应的操作控制信号。

指令执行(EXE)：根据指令译码得到的操作控制信号，执行相应的动作。

存储器访问(MEM)：所以需要访问存储器的操作都在此进行，把数据写入到存储器中数据地址所指定的存储单元或者从存储器中得到数据地址单元中的数据。

结果写回(WB)：指令执行的结果或者访问存储器中得到的数据写回相应的目的寄存器中。

MIPS指令

该单周期CPU的指令集完全依照MIPS规范，但是只实现了部分指令。关于MIPS指令详细信息请访问：<https://en.wikipedia.org/wiki/MIPS_architecture>

指令类型

MIPS指令有如下三个类型，MIPS使用的是大端存储方式，关于大端存储的详细信息请访问：[https://en.wikipedia.org/wiki/Endianness#Big](https://en.wikipedia.org/wiki/Endianness" \l "Big)

R类型

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Type | -31- Format(bits) -0- | | | | | |
| R | opcode(6) | rs(5) | rt(5) | rd(5) | shamt(5) | funct(6) |

I类型

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Type | -31- Format(bits) -0- | | | |
| I | opcode(6) | rs(5) | rt(5) | immediate(16) |

J类型

|  |  |  |
| --- | --- | --- |
| Type | -31- Format(bits) -0- | |
| J | opcode(6) | address(26) |

字母简写含义

opcode: 6位操作码

rs: 第1个源操作数寄存器，寄存器地址编号为 00000-11111，即 00-1F

rt: 第2个源操作数寄存器或目的操作数寄存器，寄存器地址编号为 00-1F

rd: 目的操作数寄存器，寄存器地址编号为 00-1F

shamt: 位移量(Shift Amount)，在移位指令中用于指定移动的位数

funct: 为功能码，在R类型指令中用来指定指令功能

常见运算指令

1. 算术运算指令

* add rd, rs, rt

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 000000 | Rs(5位) | Rt(5位) | Rd(5位) | Reserved |

功能：rd←rs + rt。reserved为预留部分，即未用，一般填“0”。

* addi rt, rs, immediate

|  |  |  |  |
| --- | --- | --- | --- |
| 000001 | Rs(5位) | Rt(5位) | Immediate(16位) |

功能：rt←rs + (sign-extend)immediate；immediate符号扩展再参加“加”运算。

* sub rd, rs, rt

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 000010 | Rs(5位) | Rt(5位) | Rd(5位) | Reserved |

功能：rt←rs + (sign-extend)immediate；mmediate符号扩展再参加“加”运算。

1. 逻辑运算指令

* ori rt, rs, immediate

|  |  |  |  |
| --- | --- | --- | --- |
| 010000 | Rs(5位) | Rt(5位) | Immediate(16位) |

功能：rt←rs | (zero-extend)immediate；immediate做“0”扩展再参加“或”运算。

* and rd, rs, rt

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 010001 | Rs(5位) | Rt(5位) | Rd(5位) | Reserved |

功能：rd←rs & rt；逻辑与运算。

* or rd, rs, rt

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 010010 | Rs(5位) | Rt(5位) | Rd(5位) | Reserved |

功能：rd←rs | rt；逻辑或运算。

1. 移位指令

* sll rd, rt, sa

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 011000 | 未用 | Rt(5位) | Rd(5位) | Sa | Reserved |

功能：rd<－rt<<(zero-extend)sa，左移sa位 ，(zero-extend)sa

1. 比较指令

* slt rd, rs, rt 带符号数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 011100 | Rs(5位) | Rt(5位) | Rd(5位) | Reserved |

功能：if (rs<rt) rd =1 else rd=0, 具体请看表2 ALU运算功能表，带符号

1. 存储器读/写指令

* sw rt, immediate(rs) 写存储器

|  |  |  |  |
| --- | --- | --- | --- |
| 100110 | Rs(5位) | Rt(5位) | Immediate(16位) |

功能：memory[rs+ (sign-extend)immediate]←rt；immediate符号扩展再相加。

* lw rt, immediate(rs) 读存储器

|  |  |  |  |
| --- | --- | --- | --- |
| 100111 | Rs(5位) | Rt(5位) | Immediate(16位) |

功能：rt ← memory[rs + (sign-extend)immediate]；immediate符号扩展再相加。

1. 分支指令

* beq rs, rt, immediate

|  |  |  |  |
| --- | --- | --- | --- |
| 110000 | Rs(5位) | Rt(5位) | Immediate(偏移量，16位) |

功能：if(rs=rt) pc←pc + 4 + (sign-extend)immediate <<2；

* bne rs, rt, immediate

|  |  |  |  |
| --- | --- | --- | --- |
| 110001 | Rs(5位) | Rt(5位) | Immediate(16位) |

功能：if(rs!=rt) pc←pc + 4 + (sign-extend)immediate <<2 else pc ←pc + 4

* bgtz rs, immediate

|  |  |  |  |
| --- | --- | --- | --- |
| 110010 | Rs(5位) | 00000 | Immediate(16位) |

功能：if(rs>0) pc←pc + 4 + (sign-extend)immediate <<2 else pc ←pc + 4

1. 跳转指令

* j addr

|  |  |
| --- | --- |
| 111000 | Addr[27..2] |

功能：pc <－{(pc+4)[31..28],addr[27..2],0,0}，无条件跳转。

1. 停机指令

* halt

|  |  |
| --- | --- |
| 111111 | 00000000000000000000000000(26位) |

功能：停机；不改变PC的值，PC保持不变。

设计思路

数据通路和控制单元

功能测试

测试文件

功能验证

实验心得