1. 简介

这是一台简单的PC机。支持键盘输入，字符显示模式和硬盘读写。指令集一共有30条指令，可以灵活的进行算术运算，逻辑运算，内存读写，以及分支循环等控制。结构上主要分为两个部分，一部分为CPU，另一部分为内存及外设。外设与CPU的交互用的寄存器全部与内存统一编址，只需要正常的内存读写指令即可与外设进行交互。而且每一个地址对应一个16位数。

1. 指令集

移位指令：

|  |  |  |  |
| --- | --- | --- | --- |
| **指令** | **功能** | **格式** | **执行** |
| sll | 逻辑左移 | sll rd,rt,sft | rd=rt<<sft |
| srl | 逻辑右移 | srl rd,rt,sft | rd=rt>>sft |
| sra | 算术右移 | sra rd,rt,sft | rd=rt>>>sft |
| sllv | 逻辑左移 | sllv rd,rt,rs | rd=rt<<rs |
| srlv | 逻辑右移 | srlv rd,rt,rs | rd=rt>>rs |
| srav | 算术右移 | srav rd,rt,rs | rd=rt>>>rs |

逻辑的左移右移，算数的左移右移，立即数指定所移位数，通过寄存器指定所移位数，一次移多位，这些全都是支持的。

跳转指令：

|  |  |  |  |
| --- | --- | --- | --- |
| **指令** | **功能** | **格式** | **执行** |
| jr | 寄存器转移 | jr rs | PC=$rs |
| jalr | 寄存器转移 | jalr rs,rd | $rd=PC+4, PC=$rs |
| j | 转移 | j L | PC=PC高4|(L<<2) |
| jal | 调子程序 | jal L | $ra=PC+4, j L |
| beq | 相等转移 | beq rs,rt,L | PC+=4;if(rs==rt)PC+=L |
| bne | 不等于转移 | bne rs,rt,L | PC+=4;if(rs!=rt)PC+=L |

支持跳绝对地址，相对地址，相等时跳转，不相等时跳转，子程序调用，间接地址跳转。

算术运算：

|  |  |  |  |
| --- | --- | --- | --- |
| **指令** | **功能** | **格式** | **执行** |
| add | 加法，有溢出 | add rd,rs,rt | rd=rs+rt |
| addu | 加法，不溢出 | addu rd,rs,rt | rd=rs+rt |
| sub | 减法，有溢出 | sub rd,rs,rt | rd=rs-rt |
| subu | 减法，不溢出 | subu rd,rs,rt | rd=rs-rt |
| mul | 短乘法 | mul rd,rs,rt | rd=rs\*rt |
| addi | 立即数加，有溢出 | addi rt,rs,imm | rt=rs+imm |
| addiu | 立即数加，不溢出 | addiu rt,rs,imm | rt=rs+imm |

支持加减乘三种指令，以及含有立即数的加减。

逻辑运算以及关系运算：

|  |  |  |  |
| --- | --- | --- | --- |
| **指令** | **功能** | **格式** | **执行** |
| and | 与 | and rd,rs,rt | rd=rs&rt |
| or | 或 | or rd,rs,rt | rd=rs|rt |
| xor | 异或 | xor rd,rs,rt | rd=rs^rt |
| nor | 或非 | nor rd,rs,rt | rd=~(rs|rt) |
| slt | 比较 | slt rd,rs,rt | rd=0;if(rs<rt)rd=1; |
| sltu | 小于无符号数 | sltu rd,rs,rt | rd=0;if(rs<rt)rd=1; |
| slti | 小于立即数 | slti rt,rs,imm | rd=0;if(rs<imm)rt=1; |
| sltiu | 小于无符号数 | sltiu rt,rs,imm | rd=0;if(rs<imm)rt=1; |
| andi | 立即数与 | addi rt,rs,imm | rt=rs&imm |
| ori | 立即数或 | ori rt,rs,imm |  |
| xori | 立即数异或 | xori rt,rs,imm | rt=rs^imm |
| lui | 高位立即数 | lui rt,imm | rt高=imm |

逻辑运算支持与、或、异或、或非，以及含有立即数的与或和异或。非运算可以通过或非来实现。

关系运算支持有符号数的小于判断和无符号数的小于判断。其它关系运算均可以转化成小于运算。

最后还支持了高位立即数指令。

协0处理器指令，和系统调用

|  |  |  |  |
| --- | --- | --- | --- |
| **指令** | **功能** | **格式** | **执行** |
| mfc0 | 读协0寄存器 | mfc0 rt,rd | $rt=协0.rd |
| mtc0 | 写协0寄存器 | mtc0 rt,rd | 协0.rd=rd |
| eret | 中断返回 | eret | EXL=0;PC=EPC |
| syscall | 系统调用 | syscall | $v0功能号，$a参数 |

内存读写：

|  |  |  |  |
| --- | --- | --- | --- |
| **指令** | **功能** | **格式** | **执行** |
| lh | 读2字节 | lh rt,D(rs) | rt=Memory[rs+D] |
| lw | 读字(4字节) | lw rt,D(rs) | rt=Memory[rs+D] |
| sh | 写2字节 | sh rt,D(rs) | Memory[D+rs]=rt |
| sw | 写字(4字节) | sw rt,D(rs) | Memory[D+rs]=rt |

1. 模块层级关系
2. CPU模块
   1. 控制模块

总共有35个状态，23个控制信号。

详细的状态转移图请见附件StateGraph，各个控制信号的具体情况也请见ControlUnit.xlsx.

* 1. ALU模块

这里进行各种算术运算，逻辑运算，位移运算。运算类型由ALUop和Func共同决定。

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ALUop** | **0** | **1** | **1x** | **1x** | **1x** | **1x** | **2x** | **3x** | **4x** | **5x** |
| **Func** | **xxxxxx** | **xxxxxx** | **0x0** | **0x2** | **0x3** | **0x4** | **0x6** | **0x7** | **0x20** | **0x21** |
| **OP** | **add** | **sub** | **sll** | **srl** | **sra** | **sllv** | **srlv** | **srav** | **add** | **addu** |
| **ALUop** | **6x** | **7x** | **8x** | **9x** | **10x** | **11x** | **12x** | **13x** | **14x** |  |
| **Func** | **0x22** | **0x23** | **0x24** | **0x25** | **0x26** | **0x27** | **0x2A** | **0x2B** | **0xFF** |  |
| **OP** | **sub** | **subu** | **and** | **or** | **xor** | **nor** | **slt** | **sltu** | **mul** |  |

* 1. 协0寄存器

里面有三个32位寄存器：

0号：EPC

1号：Cause

2号：State

State[0]为1时允许中断，为0时不允许中断。State[1]为1时，表示前一个中断还未处理完成。

中断来时，如果当前State[0]为0，会把中断号存入Cause，并把State[1]置为1。

取指时，如果State[1]为1，且State[0]为1，则会把PC值会存入EPC， 之后跳到0x4F00的中断总入口处。之后中断总入口处的程序可以对中断进行后续的处理。

中断号：

0号中断：Syscall

1号中断：键盘中断

2号中断：时钟中断

1. 存储及外设
   1. 主存0x0000-0x2FFF以及0x33e8-0x4fff
   2. 显存0x3000-0x33e7。

这1000个地址对应屏幕上25\*40的字符界面。向显存内写入浙标码，会在显示器上显示出对应的字符。

* 1. 键盘：0x5000。

每次键盘需要发送扫描码时，都会把扫描码放入0x5000的位置，然后引起一次键盘中断。键盘内维护了一个长度为16的扫描码缓冲区。

* 1. 串口控制的虚拟磁盘：0x50FC-0x50FF。

0x50FC-0x50FD这32位数存的是扇区号。

0x50FE存的是指令。0代表空，1代表读磁盘，2代表写磁盘

0x50FF存的是ready信号。

0x5100-0x5200存的是数据区。

每次操作磁盘时，需要先写0x50FC-0x50FD的扇区号。

之后向0x50FE写入指令。

写入1时，代表读，磁盘控制器会将虚拟磁盘的对应扇区读入数据区。

写入0时，代表写，磁盘控制器会将数据区写回虚拟磁盘的对应扇区。

此时CPU可以去干自己的事情，磁盘控制器的工作不需要CPU暂停等待。

等到读写完成后，磁盘控制会把0x50FE的指令置为0，把0x50FF的ready信号置为1。

如果CPU想要访问数据区，只需判断ready信号是不是1，如果是1，则数据有效，可以进行访问。否则数据区的数据仍在变化当中，此时访问是不安全的。