数字逻辑与处理器基础实验

MIPS处理器实验

实验报告

**胡钰彬 2016011201 SZ1**

**王振宇 2016011169 SZ3**

**于晨昕 2016011037 SZ2**

**2018年7月**

1 课程任务和实验要求

1.1实验目的

MIPS 处理器是一种较为经典的处理器架构，之前在《数字逻辑与处理器基础》理论课上围绕着单周期、多周期、流水线三种模式进行了初步的学习。在《数字逻辑与处理器基础实验》 夏季学期的课程中，我们将围绕理论课学习内容，使用 Verilog 来实现一种 32 位的 MIPS 处理器设计， 并实现单周期和流水线两种模式。同时，我们也将实现 MIPS 指令编译器，将MIPS汇编指令编译为机器代码，从而使得处理器能够进行处理。在接下来的内容中，我们将重述课程的任务及要求、介绍我们小组的分工、介绍汇编代码的设计和编译器的设计、ALU 的设计、单周期 MIPS 处理器的设计、流水线 MIPS 处理器的设计。并在原理介绍之后介绍该处理器的仿真及硬件调试情况。最后，我们将对我们的实验进行总结。相关源代码在文件中给出。

1.2 实验要求

I. 设计一个 32 位的 ALU，实现基本的算术、逻辑、关系、位与移位运算；

II. 设计一个单周期 MIPS 处理器，要求完成以下 指令：

a) 空指令：nop (0x00000000, 即 sll $0,$0,0)

b) 存储访问指令：lw, sw, lui；

c) 算术逻辑指令：add, addu, sub, subu, addi, addiu, and, or, xor, nor, andi, sll, srl, sra, slt, slti, sltiu；

d) 分支和跳转指令 branch(beq, bne, blez, bgtz, bltz) 和 jump(j, jal, jr, jalr)；

e) 其他指令可以根据情况自行添加。

III. 在单周期 MIPS 处理器的基础上，设计一个 5 级流水线的 MIPS 处理器。

2 小组成员分工

|  |  |
| --- | --- |
| 胡钰彬 | ALU设计  五级流水线处理器阻塞、转发、冒险检测单元设计 |
| 王振宇 | 汇编指令编写  编译器设计  外设模块及中断处理的设计 |
| 于晨昕 | 单周期处理器设计  五级流水线处理器段间寄存器设计 |

3 实验设计

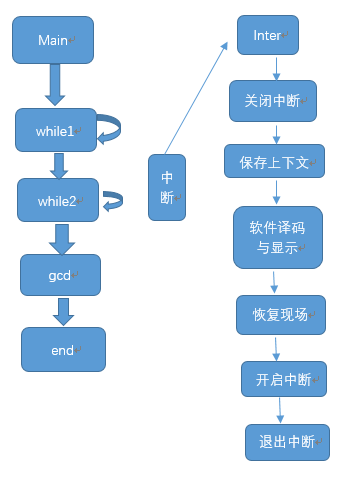
3.1 总体方案设计

整个实验分成如下几个部分：汇编程序和编译器，完成处理器共用汇编代码的编写；32bit ALU设计，完成算术运算功能，之后整合到处理器设计中；单周期MIPS处理器；流水线MIPS处理器。

3.2 汇编程序和编译器

3.2.1 汇编程序

汇编程序的基本结构如下图所示：



汇编程序完成的功能为通过 UART 输入两个 8bits 的操作数并计算它们的最大公约数。采用软件译码方式通过七段数码管显示十六进制的操作数，通过 8 个 LEDs 显示最大公约数结果并通过 UART 输出。

具体来讲，首先，进入 Main，在while1与while2两个部分通过轮询的方式利用UART读取两个操作数，读取完毕后设置定时器周期并启动定时器。之后跳转到gcd处进行最大公约数的计算。计算结束后进入end部分，将结果通过UART返回，并通过二极管显示。

当定时器发送中断信号时，程序会进入Inter部分，开始执行中断处理代码。依次完成以下流程：

I.定时器中断禁止，同时中断状态清零，TCON的 1-2bit 清零，TCON&=0xfffffff9

II.保存现场

III.软件译码并显示

IV. 恢复现场

V.使能中断，TCON的 1bit 置 1，TCON |= 0x00000002

VI.退出中断服务程序，跳转到中断发生时保存 的断点地址处继续执行（$26）

3.2.2 编译器

我们的编译器使用 C++语言，对 txt 格式的文 本进行处理，最终生成 txt 文件，文件中为汇编代 码的对应的机器码。

编译的过程是对汇编指令所在文件的两次扫描。第一次扫描，主要读取汇编代码中的所有标签，并将标签对应的地址计算出来，存入一个vector变量中。第二次扫描，则逐条处理汇编代码，将每一条汇编代码都转换为相应的机器码。

在逐条处理的过程中，考虑到十进制立即数、寄存器、OpCode和Func的转换，设计了dec2bin函数，实现十进制数转换为二进制比特串的功能，输入为需要转换的十进制数和输出二进制比特串的位数，输出为对应的二进制比特串。具体转换时，首先使用函数ins2Opcode识别OpCode对应的十进制数，之后根据识别到的指令种类不同，进行不同的处理。根据本实验的要求，在编译过程中，我们将指令分为了一般R型指令、内存读取型指令（lw和sw）、lui指令、一般I型指令、左移右移型指令、分支跳转型（其中beq和bne为一类、blez, bgtz和bltz为一类）、j和jal、jr、jalr、nop这几类，根据每一类中指令的特点，进行相应的转换。

3.3 外设

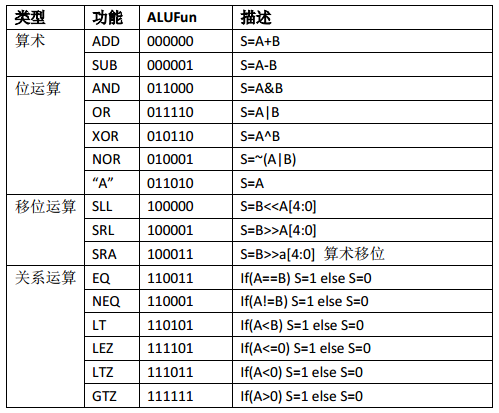
外设模块块其实与数据存储是统一的， 根据地址范围对二者进行区分。0x00000000～0x3FFFFFFF（字节地址）为数据 RAM，可以提供数据存储功能；0x40000000～ 0x7FFFFFFF（字节地址）为外设地址空间，对其地址的读写对应到相应的外设资源。

外设中定时器模块已经由老师完成，因此这里主要介绍一下 UART 外设。UART外设模块根据春季学期设计的UART 进行改造。基本结构包括UART时钟产生模块（产生9600波特率串口对应的合适时钟频率）、接收器和发送器三个模块。与春季学期不同的是由于我们使用的CPU周期大于UART使用的时钟周期，因此接收器中的rx\_status需要适当地加长从而使得CPU能够检测到。

3.4 ALU

本次要设计的32bit ALU主要功能是实现基本的算术、逻辑、关系、位与移位运算。

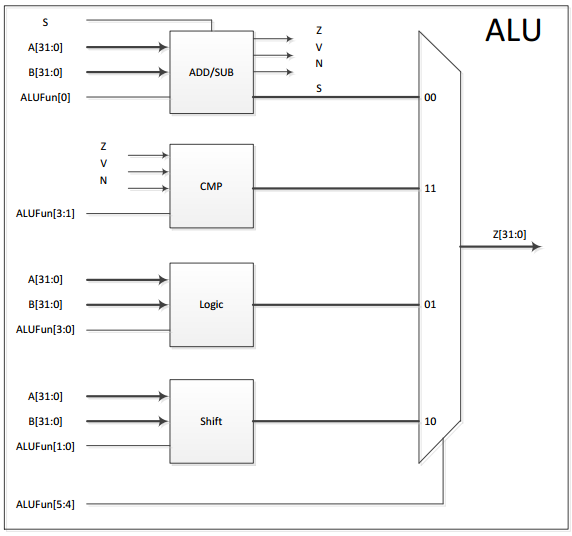
ALU功能表如下所示：



ALU端口如下所示：



该ALU整体按照下图所示结构拼装完成：



因此，按照上图所示结构，ALU主要有以下几个部分：

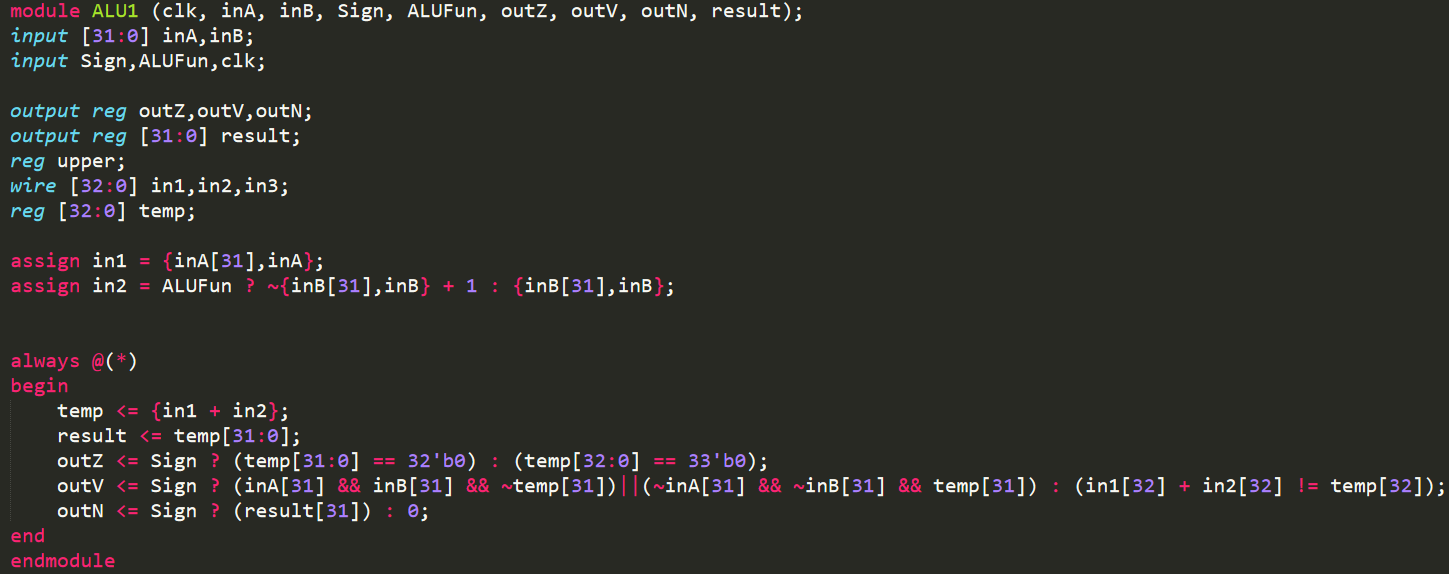
I. 算术运算部分：

这一部分接收两个32位数，同时收到符号位和判断运算方法的ALUFun[0]，其主要功能是按照这两个指示位完成相应的运算并输出结果，同时还要输出结果是否为零、是否溢出和是否为负这三个指示位，供后续模块完成功能。

在进行运算之前，要对输入的两个数进行符号扩展，A直接用符号为扩展即可，B则需要根据运算方法进行符号扩展或是扩展后取其相反数的二补码。

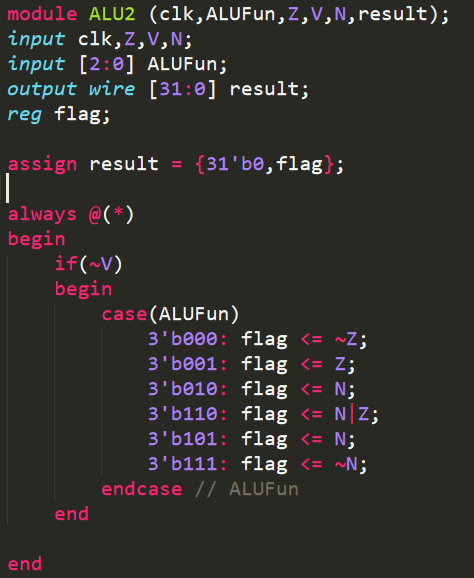
而符号位只影响到做比较时的结果，因此在输出三个指示位的时候，要根据符号为的不同选择不同的逻辑来计算。具体到溢出的判断方法：若有符号，则溢出只可能是同号相加的上溢出或下溢出，这就只需要考虑两个加数与和的符号之间是否一致即可；若无符号，则只需判断最高位相加有无进位，若有进位则溢出，这又可以通过比较加数与和的扩展位得到。

这部分代码如下：



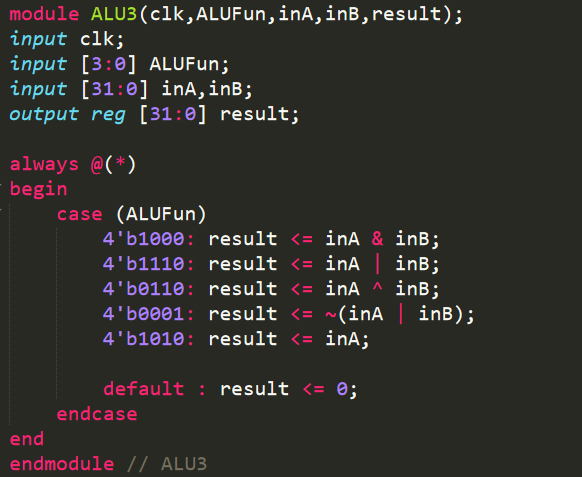
II. 比较运算部分：

这一部分接收的是I中输出的三个指示位和功能选择位ALUFun[3:1]，只需根据功能选择位，构建起输出和三个指示位之间的逻辑关系即可，这部分代码如下：



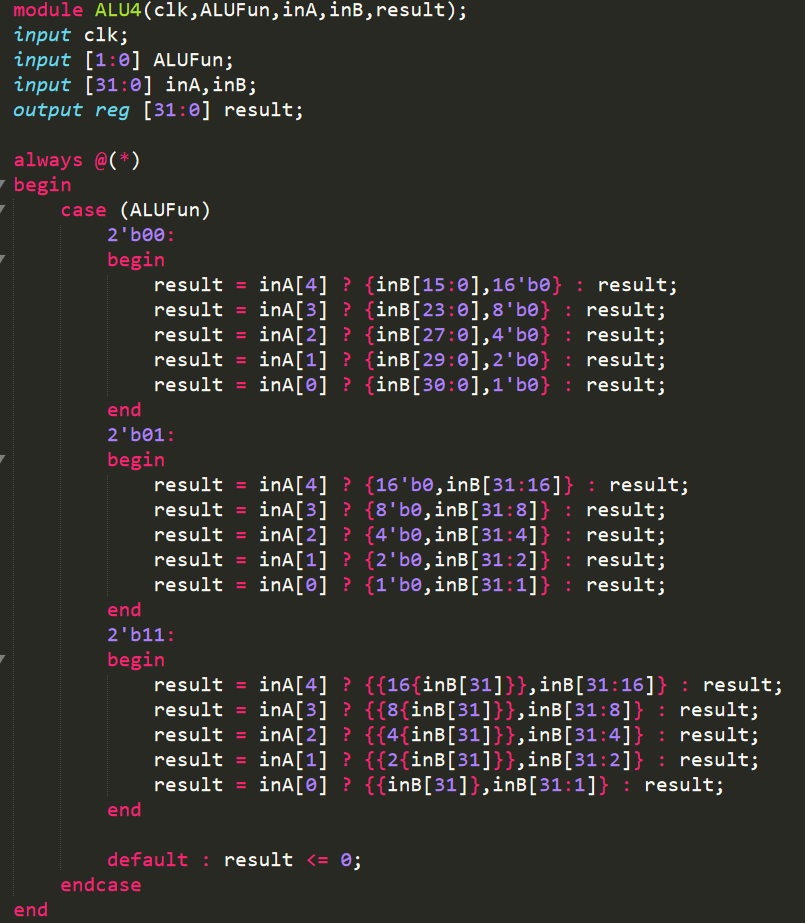
III. 逻辑运算部分：

这部分接收的是两个32位数和3位的功能选择位，只需按照功能选择位对输入的两个数字按位做逻辑运算即可，代码实现如下：



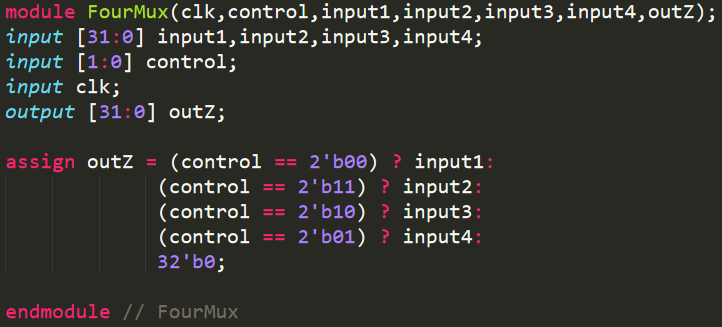
IV. 移位运算部分：

这部分接收两个32位数和2位的功能选择位，其中A[4:0]这五位提供了移位的数量。而功能选择位则提供了一味的方式，只需要根据这两部分进行正确的移位即可。我们采取了级联的移位操作，即逐个检查A[4:0]的每一位，并移位相应的数量。代码实现如下：

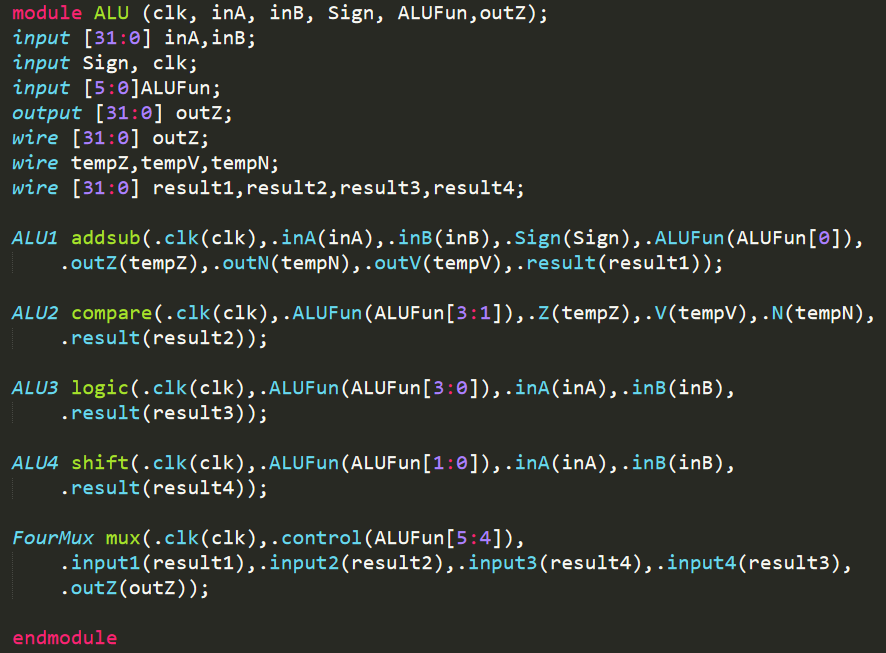


V. 多路选择器部分：

这部分主要是根据功能选择位，选择以上四个模块中的一个作为ALU的输出，是一个纯组合逻辑部分，代码实现如下：



ALU部分的顶层模块代码实现如下：



3.5 单周期MIPS处理器

3.5.1 指令存储器

3.5.2 控制器

3.5.3 寄存器

3.5.4 数据存储器

3.6 五级流水线MIPS处理器

3.6.1 冒险和转发

3.6.2 取指令（Instruction Fetch）

3.6.3 解析指令（Instruction Decompose）

3.6.4 执行阶段（Execute）

3.6.5 内存读写（Memory Operation）

3.6.6 写回寄存器（Write Back）

4 仿真情况

4.1 单周期MIPS处理器仿真情况

4.2 流水线MIPS处理器仿真情况

5 综合情况

5.1 单周期MIPS处理器综合情况

5.1.1 时序分析

5.1.2 面积分析

5.2 流水线MIPS处理器综合情况

6 总结