|  |
| --- |
| 清华大学 计算机科学与技术系 |
| 计算机组成原理大作业 |
| 实验报告 |

|  |
| --- |
| 计34 何钦尧 2012010548  计34 王凯 2013XXXXXX  计34 黄科 2012XXXXXX  2015-12-10 |

目录

[1. 概述 2](#_Toc437593491)

[2. 系统整体设计 2](#_Toc437593492)

[2.1. CPU 2](#_Toc437593493)

[2.2. IOBridge 4](#_Toc437593494)

[3. CPU具体设计 4](#_Toc437593495)

[4. IOBridge具体设计 4](#_Toc437593496)

[5. VGAdapter具体设计 4](#_Toc437593497)

[6. KeyboardAdapter具体设计 4](#_Toc437593498)

[7. 最终成果展示 4](#_Toc437593499)

[8. 实验心得与体会 4](#_Toc437593500)

[8.1. 良好的文档的重要性 4](#_Toc437593501)

# 概述

在本实验中，我们在THINPAD教学计算机的硬件平台上，实现了能够执行THCO-MIPS指令集（MIPS16e的改编实现）的CPU。本组实现的指令为30条，包括25条基础指令和5条扩展指令。

CPU支持指令流水，能够以的主频运行。妥善的处理了流水线中可能有的各种结构冲突，数据冲突，控制冲突。

能够从Flash中加载程序到内存当中，完成自启动。支持串口读写，从PS/2键盘中输入数据，以及分辨率为的VGA外接显示。

此外实现了一个写字板的软件程序用于演示键盘和VGA的功能。实现了字符从键盘中的输入，回车换行，通过上下左右键移动光标等功能。

# 系统整体设计

系统的总体设计分为几个方面。一个是CPU；一个是CPU与各种IO外设（包括内存）交互的一个总控单元（因为所有对外设的访问都通过对某个特定的内存地址的访问来进行），这个有点类似于一般个人电脑的硬件结构中的北桥芯片。另外就是其他的外设，如VGA控制单元，键盘控制单元，Flash控制单元等。

各个组件在VHDL程序中的命名为：

* CPU
* IOBridge
* VGAdapter
* KeyboardAdapter
* FlashAdapter

在后文中，我们也将直接使用这些名字。

CPU本身只直接连接IOBridge（这里我们的“北桥芯片”），由IOBridge来完成对包括内存的其他外设的访问。同时，CPU所用的时钟信号是由IOBridge产生的。这样的实现方式是由于，IOBridge控制内存的访问，需要一个状态机来控制。而这个状态机的状态变换必须要和CPU的操作（也即时钟上升沿产生的时机）保持一个同步，这样才能保证CPU的执行中在需要的时候获取到了需要的信息。

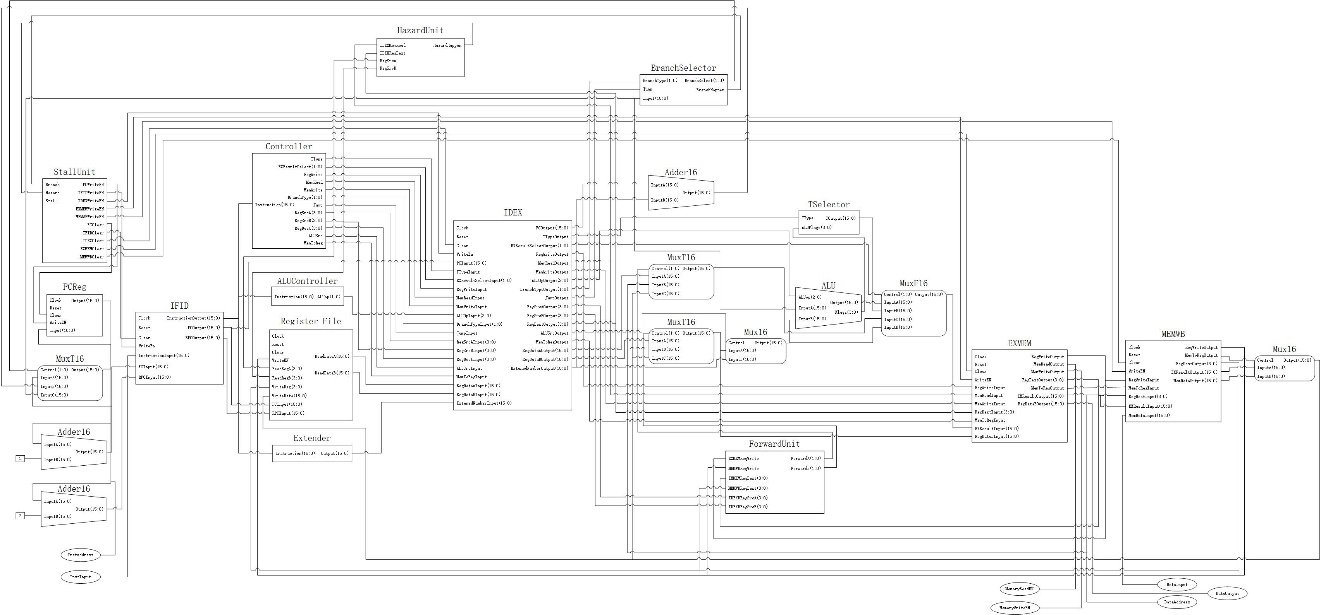
于是CPU在这里仅仅完成指令执行的流水线，而不含外设的访问逻辑。这样使得CPU的整个设计比较规整，易于实现和优化。

由于监控程序的实际需求（写指令内存），我们这里不将指令和数据内存分离，统一都放置在一块内存上。这里我们使用了Ram2，而没有使用Ram1。

## CPU

CPU采用MIPS通用的五段流水线的实现方式，五段流水线分别实现，IF(Instruction Fetch)，ID(Instruction Decode)，EXE(Execution)，MEM(Memory)，WB(Write Back)的功能。

以下为详细的CPU设计数据通路图：



也可以查看该图的源文件：datapath.vsdx, datapath.jpg。在图中，椭圆形标注的信号为外部的输入或者输出。另外所有组件的Clock（时钟信号）和Reset（复位信号）都接自同一来源，因此为了清晰没有在图中画出。

图中所有的矩形或者梯形框为组成CPU的各个组件（具体的也对应到实现中的一个VHDL entity）。约定所有的元件，左边标注的为输入，右边标注的为输出。图中所有信号的标注名称，都是在实现中VHDL内部使用的名称。不引起混淆的情况下，下文中也将使用这些名称。

CPU作为一个整体的元件，其输入输出信号有：

* Clock，输入，时钟信号
* Reset，输入，复位信号
* InstAddress，输出，指令地址
* InstInput，输入，取得的指令的输入
* DataAddress，输出，数据地址
* DataInput，输入，访存获得的内存的输入
* DataOutput，输出，需要往内存中写的数据
* MemoryReadEN，输出，读使能。仅用于控制数据的读。
* MemoryWriteEN，输出，写使能。仅用于控制数据的写。

以上的所有输入输出信号都和IOBridge相连接。也可以认为CPU仅直接和IOBridge通信。在只考虑CPU时，可以认为IF和MEM阶段的内存访问都是完全的组合逻辑操作。因而在CPU中，只有各寄存器用到了时序逻辑。

CPU使用五级流水线的实现，并且处理了流水线中的各种冲突。使用旁路单元（ForwardUnit）和冒险检测单元（HazardUnit）处理了数据冲突。也用StallUnit和BranchSelector的配合处理了跳转控制冲突。而IF和MEM段同时需要访问内存的结构冲突，留给IOBridge进行处理。

## IOBridge

IOBridge这里设计为几乎是整个计算机系统的最中央的控制单元，控制给CPU的时钟信号的发生，以及与各种IO设备的交互。

由于内存读写的时序的复杂性，以及指令和数据同时访存的结构冲突，IOBridge这里必须要运行在比CPU主频更高的频率上。

IOBridge运行一个状态机模型。在系统加电（或者Reset之后），首先从Flash中读取运行的程序并加载到内存当中，这一过程完成之后，开始给CPU产生时钟信号，开始运行。

通过对内存地址进行划分的方式，来处理不同外设的访问。0xBF00设定为串口的读写地址，0xBF01为串口状态信息地址；0xBF02为键盘数据地址，0xBF03为键盘输入状态信息地址；0xF800到0xFFFF划分出来作为显存的地址。对这些特殊地址进行的Load或Store指令就可以完成对IO外设的访问。

# CPU具体设计

## CPU控制信号设计

## 各主要组件实现详解

### ALU

### ALUController

### BranchSelector

### Controller

### Extender

### ForwardUnit

### HazardUnit

### Register

### StallUnit

### TSelector

# IOBridge具体设计

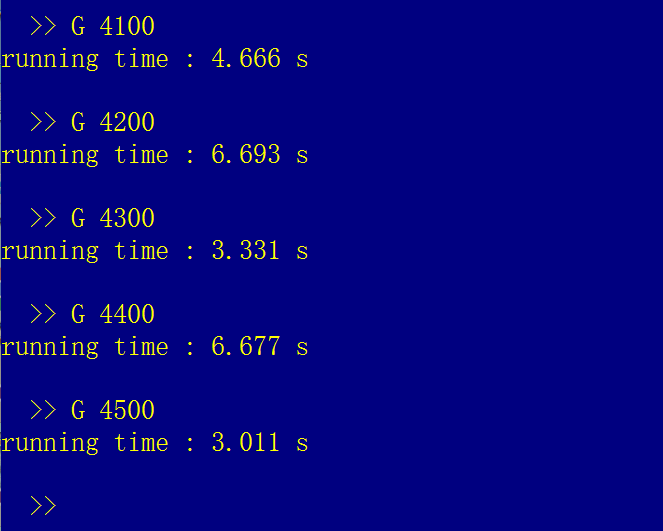
# VGAdapter具体设计

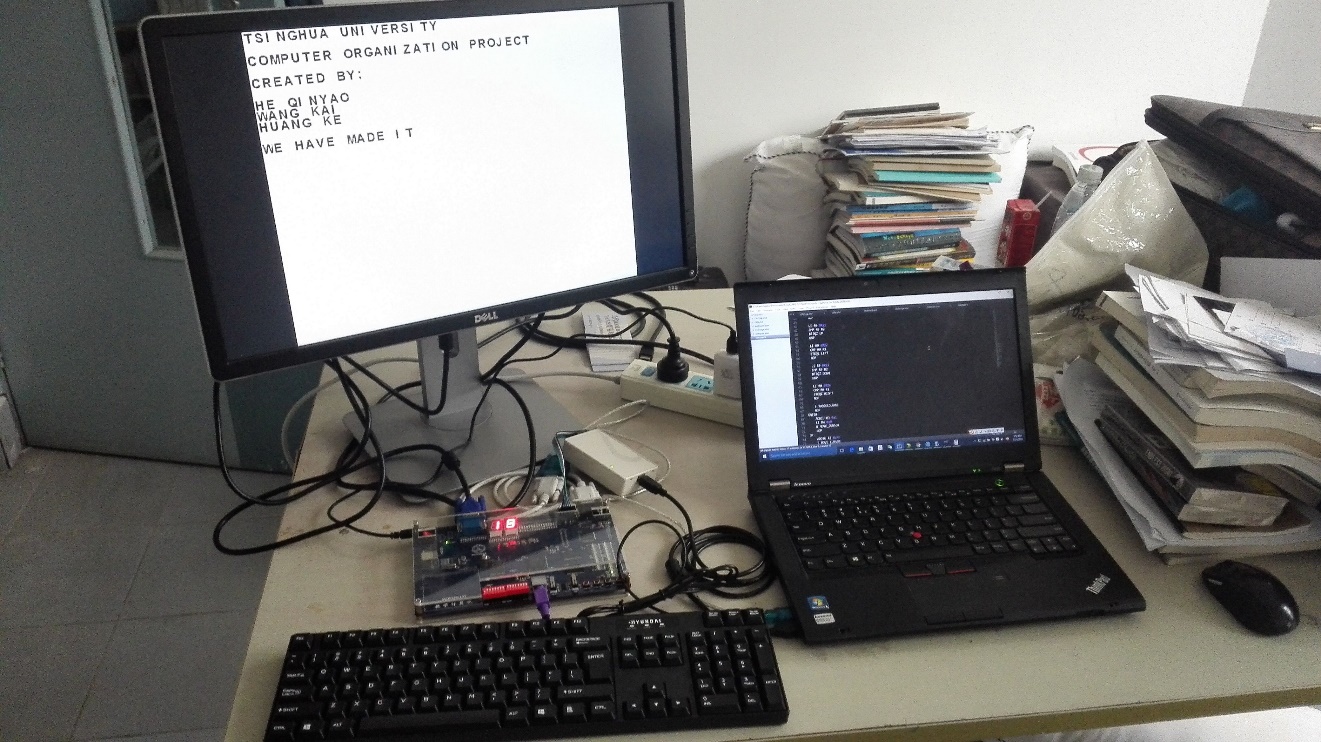
# KeyboardAdapter具体设计

# 写字板软件实现

# 最终成果展示

我们能够最终运行在，最高37.5MHz，最低（在有数据内存访问时）25MHz的可变频率上。下图为运行给出的5个检查使用的程序的结果，运行时间基本符合使用主频计算出来的理论值：





# 实验心得与体会

## 良好的设计的重要性

在“造台计算机”的三个星期的进程当中，我们花了前两个周用来形成一个较为完整，没有问题的设计。