目录

[1、实验目的、重点、难点 2](#_Toc118750561)

[2、 实验任务实现 2](#_Toc118750562)

[2.1 实现指令存储器 inst\_rom 2](#_Toc118750563)

[2.2 取指模块实现 3](#_Toc118750564)

[3、 mips相关知识 4](#_Toc118750565)

**实验四 指令存储器rom**

## 1、实验目的、重点、难点

实验目的：

1. 了解指令存储器原理以及取指过程。
2. 熟悉14条MIPS32指令的格式。

实验重点：

熟悉使用verilog语言进行设计，仿真的过程。

实验难点：

观察仿真波形，确认电路设计是否正确。观察指令存储器的时序图。

## 实验任务实现

### 2.1 实现指令存储器 inst\_rom

inst\_rom设计图如图2.1所示，端口说明如表2.1所示。

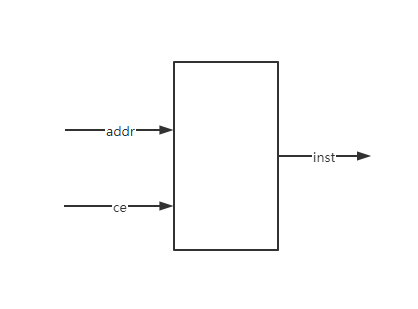


图2.1 inst\_rom设计框图

表2.1 inst\_rom 端口说明

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 接口名 | 宽度（bit） | 输入/输出 | 作用 |
| 1 | ce | 1 | 输入 | 使能信号 |
| 2 | addr | 32 | 输入 | 要读取的指令地址 |
| 3 | inst | 32 | 输出 | 读出的指令 |

指令的地址由程序计数器PC给出，PC中的地址对应的基本存储单元大小是一个字节。也就是指令存储器本质上是按照字节进行寻址的。本次实验中设计的指令存储器实现的功能是给出一个地址，同时能读出4字节的内容，从而作为一条指令的内容。本次实验一共设计了128\*4B指令存储器。指令存储器定义时依然使用memory类型进行定义,即reg[31:0] inst\_mem[0:127]。现在我们要考虑的是怎么将PC的地址转换为inst\_mem的地址。图2.2对这个地址转换进行了描述。

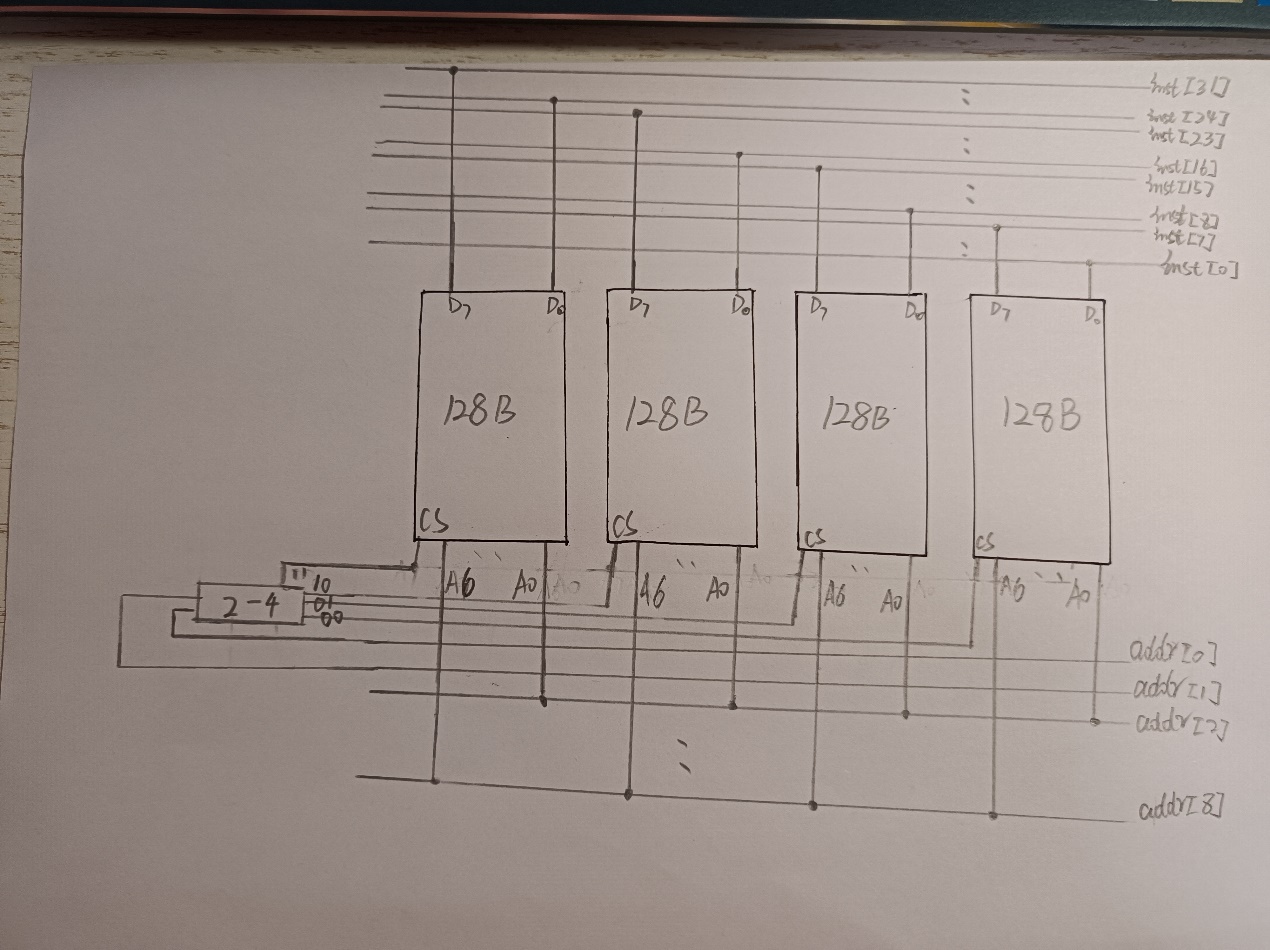


图2.2 pc 地址和指令存储器地址关系

### 2.2 取指模块实现

取指模块将pc模块和inst\_rom模块连成一个大的电路模块，实现框图如图2.3所示。自己写出端口描述。功能为每个时钟周期从指令存储器中取出一条指令。

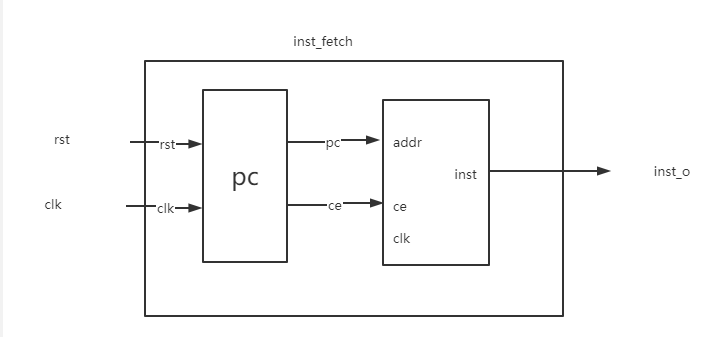


图2.3 inst\_fetch实现框图

## mips相关知识

本次实验中用到的测试的mips代码为以下14条MIPS指令，请将其翻译成机器指令。

Add $3,$2,$1 00411820

Addu $4,$2,$1

Sub $5,$6,$7

Subu $8,$9,$10

Slt $11,$12,$13

Sltu $14,$15,$16

And $17,$18,$19

Or $20,$21,$22

Xor $23,$24,$25

Nor $26,$27,$28

Sll $29,$29,31

Srl $2,$2,8

Sra $4,$4,12

Lui $5,0x0101

翻译出来的机器指令以16进制形式放在inst\_rom.data中。用来初始化inst\_rom.

初始化rom用到的verilog指令为$readmemh。

$readmemh("<数据文件名（路径地址和文件名）>",<存储器名>);

$readmemh("D:/FPGAHUE/myFirstCpu/inst\_rom.data",inst\_mem);