

# 中国科学院大学

## 《计算机体系结构基础(研讨课)》实验报告

姓名 艾华春,李霄宇,王敬华

学号 2022K8009916011

实验项目编号 3 实验名称 在流水线中添加普通用户态指令

### 一、 逻辑电路结构与仿真波形的截图及说明

#### • 访存指令添加。

##### 1. 添加访存操作类型的数据通路

在 ID(译码)阶段添加判断当前的 load 或 store 操作的类型,包括是否无符号扩展,数据的位宽。并将相应的信号沿着流水级传递到 MEM(访存)阶段。

```
assign id_op_st_ld_b = op_25_22[1:0] == 2'd0; // 位宽为1 byte
assign id_op_st_ld_h = op_25_22[1:0] == 2'd1; // 位宽为2 byte
assign id_op_st_ld_u = op_25_22[3];           // 无符号数扩展
```

##### 2. 添加从 EX 到 MEM 访存地址最低两位传递数据通路

在 EX 到 MEM 的传递的数据中添加访存地址的最低两个 bit,使得 MEM 阶段可以获取到从 sram 中取出来 1 字节的数据内部的偏移。

##### 3. 在 load 操作的结果处添加多路选择器(在访存流水级)

首先,通过操作的数据位宽和访存地址来选择出从 sram 中输出的的数据的有效部分。

```
// mem_data_sram_addr[2] 是上一小节中传递到过mem级的访存地址后两位
assign mem_word_result = data_sram_rdata; // ld.w 的有效部分
assign mem_half_result = mem_data_sram_addr[1] ? data_sram_rdata[31:16]
: data_sram_rdata[15:0]; // ld.h 的有效部分
assign mem_byte_result = ({8{mem_data_sram_addr[1:0] == 2'd0}} & data_sram_rdata[7:0])
| ({8{mem_data_sram_addr[1:0] == 2'd1}} & data_sram_rdata[15:8])
| ({8{mem_data_sram_addr[1:0] == 2'd2}} & data_sram_rdata[23:16])
| ({8{mem_data_sram_addr[1:0] == 2'd3}} & data_sram_rdata[31:24]);
// ld.b 的有效部分
```

然后根据有符号扩展还是无符号数扩展,对结果进行扩展,最终得到可以写入目的寄存器(rd)的结果mem\_result。

```
// mem_op_st_ld_b, mem_op_st_ld_h 分别表示数据位宽为1byte和2byte
// mem_op_st_ld_u 表示数据进行无符号数扩展(即0扩展)
// ~mem_op_st_ld_u & mem_byte_result[7] 或~mem_op_st_ld_u & mem_half_result[15] 表示符号位
assign mem_result
= mem_op_st_ld_b ? ({24{~mem_op_st_ld_u & mem_byte_result[7]}}, mem_byte_result[7:0]) :
mem_op_st_ld_h ? ({16{~mem_op_st_ld_u & mem_half_result[15]}}, mem_half_result[15:0]) :
mem_word_result;
```

#### 4. 在 store 操作中传递给data\_sram的信号处添加选择器

首先,根据访存的位宽和地址,通过多路选择和移位操作确定字节使能信号。

```
// 字节使能
// st.b 时, 根据访存地址的最低两位进行移位操作。有四种不同情况
// st.h 时, 根据访存地址的最低两位添加二选以选择器。有两种不同情况
assign ex_sram_we = ex_op_st_ld_b ? (4'b0001 << ex_data_sram_addr[1:0]) : // st.b
                    ex_op_st_ld_h ? (ex_data_sram_addr[1] ? 4'b1100 : 4'b0011) : // st.h
                    4'b1111; // st.w
```

确定字节使能后,生成写入数据。不需要将数据进行移动,而是仅仅根据数据长度,对有效部分进行复制连接操作,确保数据在写入时能拿到有效的数据。

```
//生成写入数据
//对有效部分进行复制连接操作
assign data_sram_wdata = ex_op_st_ld_b ? {4{ex_rkd_value[7:0]}}:
                    ex_op_st_ld_h ? {2{ex_rkd_value[15:0]}}:
                    ex_rkd_value[31:0];
```

## 二、 实验过程中遇到的问题、对问题的思考过程及解决方法(比如 RTL 代码中出现的逻辑 bug,逻辑仿真和 FPGA 调试过程中的难点等)

- data\_sram 写使能 bug。在仿真中,在 load 指令中,写入寄存器的结果与标准结果不一致。

然后在波形图中查看从 sram 直接返回的数据,发现同样与标准结果不一致,于是排除是新添加的 load 多路选择器的问题。

进而在反汇编代码中,发现在 load 指令前,有一条 store 指令,地址相同。于是在波形图中查看相关波形,发现该指令 ex 阶段的 sram 写使能没有拉高,再回溯到 id 阶段,发现是 mem 写使能中,不同种类的 load 之间的逻辑连接错写成与,导致不能向 sram 写入数据。

```
// assign id_mem_we      = inst_st_w & inst_st_b & inst_st_h & id_valid;
assign id_mem_we      = (inst_st_w | inst_st_b | inst_st_h) & id_valid;
```

- ALU 设计问题。请输入你的实验报告内容。

如果一段话写不下,可以再这里继续新的段落。

插入图片,可以参考如下方法:

- 其他关键模块设计问题。请输入你的实验报告内容。

如果一段话写不下,可以再这里继续新的段落。

插入第二张图片:

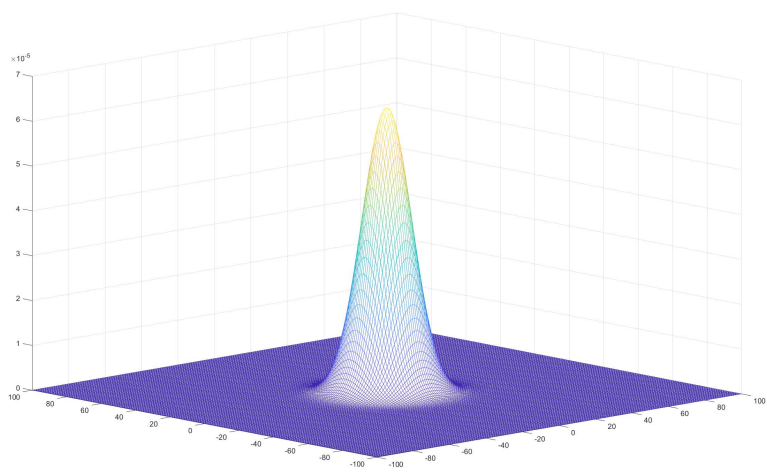


图 1: 这是第一张图

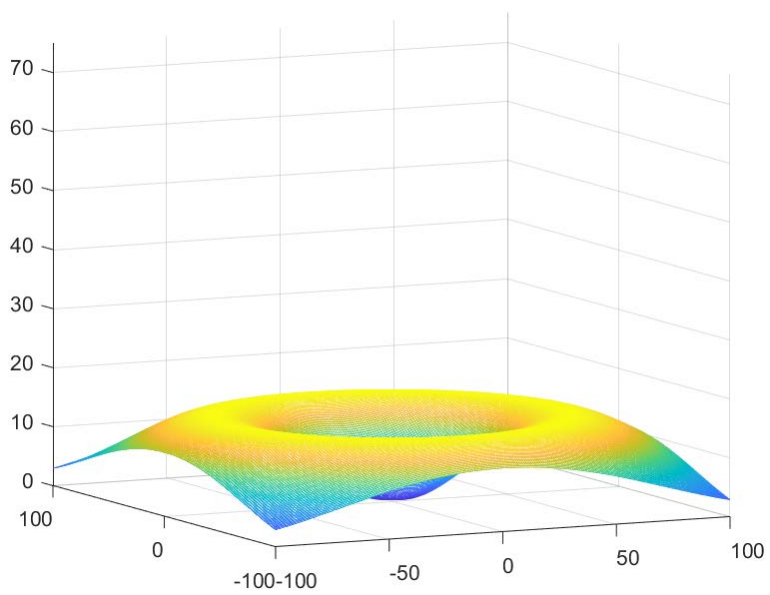


图 2: 这是第二张图