

# 测试文档

## 目录

### 一、内容概要

### 二、测试结果

#### （一）基本功能实现

#### （二）Hazard 处理

##### 1) Data Hazard

##### 2) Control Hazard

##### a) Load/Use

##### b) Misprediction

##### c) Return

##### d) Combine

##### e) Invalid

#### （三）其他功能

### 三、测试总结

## 一、内容概要

本测试文档通过在 y86 模拟器上运行了一系列代码，与 CMU 提供的 psim 所执行得到的结果进行比较，验证了其在以下几个方面的正确性：基本功能的实现，有 nop, halt, rrmovl, irmovl, rmmovl, mrmovl, addl, subl, andl, xorl, jmp, jle, jl, je, jne, jge, jg, call, ret, pushl, popl；实现流水线控制逻辑，处理各种 hazard；前进/回退功能；按照规定格式输出运行结果。

## 二、测试结果

### （一）基本功能实现：

#### 1) test1.yo

测试 rrmovl, irmovl, rmmovl, mrmovl, call, ret, pushl, popl 等功能。本用例为 archlab 中 copy.yo 的简化版本。

```

irmovl: 0x000 W init: irmovl Stack, %esp      # Set %esp 0x00000100
call: 0x00c W      call Main                  %esp 0x000000fc

```

The content of address: 0x000000fc  
is 0x00000011.

```

pushl: 0x026 W      pushl %eax %eax 0x00000001

```

The content of address: 0x000000f4  
is 0x00000001.

```

rrmovl: 0x01e W      rrmovl %esp,%ebp %esp 0x000000f8 %ebp 0x000000f8

```

```

mrmovl: 0x04c W      mrmovl 8(%ebp), %ecx %ecx 0x00000014 %ebp 0x000000e4

```

The content of address: 0x000000ec  
is 0x00000014.

```

rmmovl: 0x075 W      rmmovl %esi, (%edx)

```

The content of address: 0x00000018  
is 0x0000000a.

#### 2) test2.yo

测试 op 与 jxx。

xorl: 0x016 W	xorl %eax, %eax	<table><tr><td>%eax</td></tr><tr><td>0x00000000</td></tr></table>	%eax	0x00000000
%eax				
0x00000000				
addl: 0x01e W	addl %ebx, %eax	<table><tr><td>%eax</td></tr><tr><td>0x00000004</td></tr></table>	%eax	0x00000004
%eax				
0x00000004				
andl: 0x020 W	andl %ebx, %eax	<table><tr><td>%eax</td></tr><tr><td>0x00000004</td></tr></table>	%eax	0x00000004
%eax				
0x00000004				
subl: 0x028 W	subl %ebx, %eax	<table><tr><td>%eax</td></tr><tr><td>0x00000000</td></tr></table>	%eax	0x00000000
%eax				
0x00000000				

jxx: 在 test2.yo 中修改 subl 时 %ebx 的值, 就会进行不同跳转, 测试发现符合情况。本

例中, %eax = %edx = 4, 相等, 最终

%eax
0x00000009

## (二) Hazard 处理:

test1.yo 和 test2.yo 中已经包含了三种基本 hazard, 这里只测试 CombinationB 和 Invalid 的情况。

- 1) CombinationB: D、F 暂停, E 气泡, M、W 正常执行

test3.yo

```

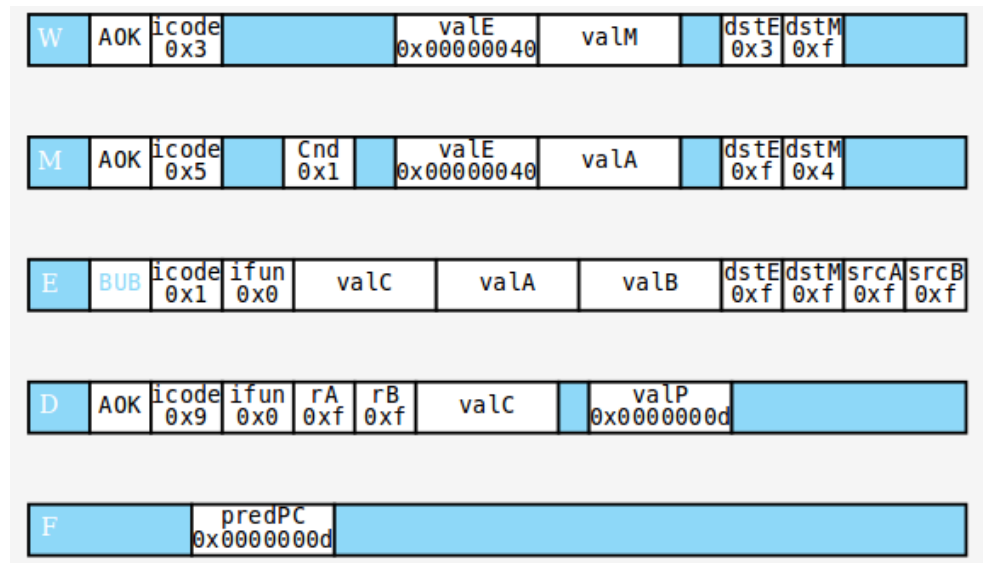
0x000 D  irmovl mem, %ebx
0x006 F  mrmovl 0(%ebx), %esp
0x00c    ret
0x00d    halt
0x00e    irmovl $5, %esi
0x014    halt
0x040    .long stack
0x050    .long rtnpt

```

前一步:

W	BUB	icode 0x1			valE	valM		dstE 0xf	dstM 0xf			
M	AOK	icode 0x3		Cnd 0x1		valE 0x00000040	valA		dstE 0x3	dstM 0xf		
E	AOK	icode 0x5	ifun 0x0		valC 0x00000000	valA		valB 0x00000040	dstE 0xf	dstM 0x4	srcA 0xf	srcB 0x3
D	AOK	icode 0x9	ifun 0x0	rA 0xf	rB 0xf	valC		valP 0x0000000d				
F			predPC 0x0000000d									



后一步:

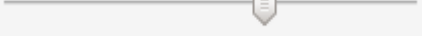


- 2) Invalid: Decode 阶段即可发现不合法的 Icode, Ifunction

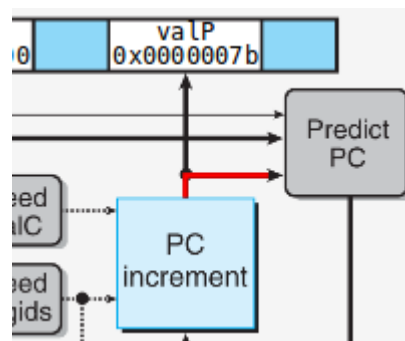


### (三) 其他功能:

- 1) 可以通过   实现前进/回退功能;

- 2) 可以调节运行速度  ;

- 3) 相关数据线显示为红色:



- 4) 将鼠标移到部分元件上可显示当前值:

`d_valA = 0x00000004`

### 三、测试总结

1. 通过全面测试基本功能, 才能确保调试完整程序时的正确性。

2. 在测试的过程中发现了一些 bug 并及时进行了修正，其中也有令人啼笑皆非的，比如误将 `irmovl` 看作 `rmmovl`，导致 `rmmovl` 不在合法的 `IcodeIfunction` 中，使程序遇到 `INS` 而错误停止。

3. 测试的时候不但要测试合法情况，也要对非法和边界问题进行重点考量，才能确保程序在各种情况下的可靠性。