

学号：202200130048	姓名：陈静雯	班级：6
实验题目：实验四：结构相关		
实验学时：2	实验日期：2025. 5. 16	
<b>实验目的：</b> 通过本实验，加深对结构相关的理解，了解结构相关对 CPU 性能的影响		
<b>硬件环境：</b> Windows		
<b>软件环境：</b> Otvdm		
<b>实验程序：</b> structure_d.s  <b>代码解读：</b> <ol style="list-style-type: none"> <li>LHI R2, (A&gt;&gt;16)&amp;0xFFFF 将数组 A 的高 16 位地址加载到寄存器 R2 的高 16 位，低 16 位清零。</li> <li>ADDUI R2, R2, A&amp;0xFFFF 将数组 A 的低 16 位地址加到 R2，形成完整的 32 位基地址。</li> <li>LHI R3, (B&gt;&gt;16)&amp;0xFFFF 类似地，加载数组 B 的高 16 位地址到 R3。</li> <li>ADDUI R3, R3, B&amp;0xFFFF 补全 B 的 32 位基地址到 R3。</li> <li>ADDU R4, R0, R3 将 R3 (B 的基地址) 复制到 R4，用于循环终止条件。</li> <li>loop: LD F0, 0(R2) 从 R2 指向的 A 数组加载双精度数到浮点寄存器 F0。</li> <li>LD F4, 0(R3) 从 R3 指向的 B 数组加载双精度数到 F4。</li> <li>ADDD F0, F0, F4 计算 <math>A[i] + B[i]</math>，结果存入 F0。</li> <li>ADDD F2, F0, F2 累加 F0 到 F2 (此处存在结构相关导致的暂停)。</li> <li>ADDI R2, R2, #8 指针 R2 递增 8 字节 (指向 A 的下一个元素)。</li> <li>ADDI R3, R3, #8 指针 R3 递增 8 字节 (指向 B 的下一个元素)。</li> <li>SUB R5, R4, R2 计算剩余元素数 (<math>R4 - R2</math>)。</li> <li>BNEZ R5, loop 若 R5 不为零，继续循环。</li> <li>TRAP #0</li> </ol>		

A: . double

定义数组 A，包含 10 个双精度数。

定义数组 A，包含 10 个双精度数。

定义数组 B，与 A 相同。

定义数组 B，与 A 相同。

(1). 用 WinDLX 模拟器运行程序 `structure_d.s`。

(2). 通过模拟, 找出存在结构相关的指令对以及导致结构相关的部件。

(3). 记录由结构相关引起的暂停时钟周期数, 计算暂停时钟周期数占总执行周期数的百分比。

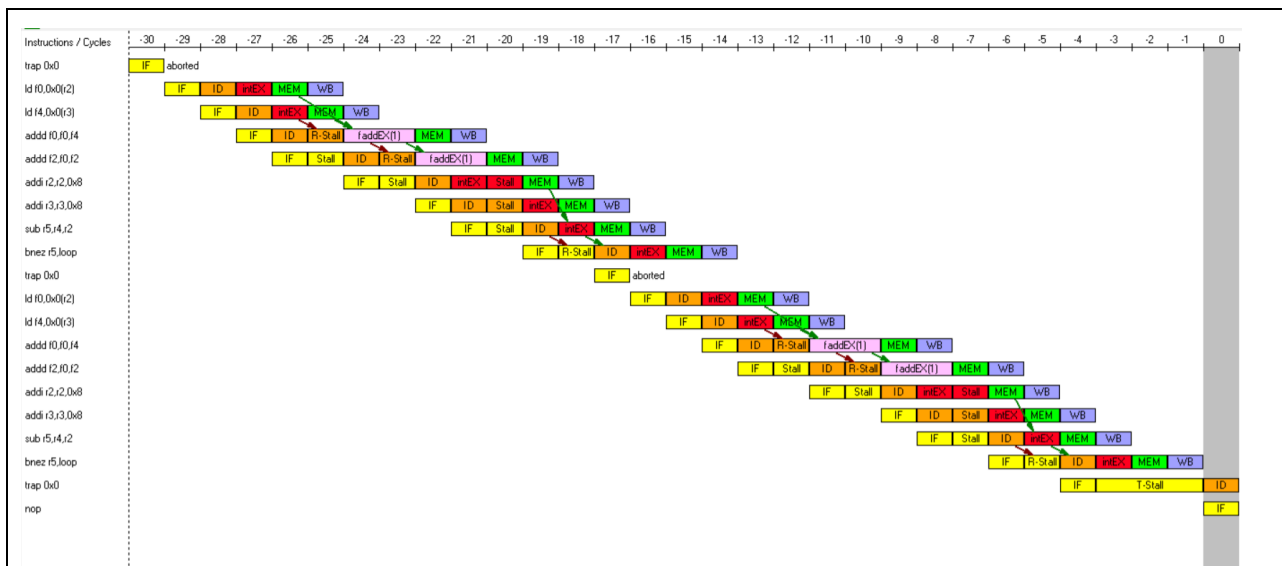
(4). 论述结构相关对 CPU 性能的影响，讨论解决结构相关的方法。

### (1) 实验结果

[illegible]

```
PC= 0x0
IMB= 0x0
```

PC=	0x0000013c	R9=	0x00000000	F2=	0	F27=	0
IMAR=	0x00000138	R10=	0x00000000	F3=	3.4297	F28=	0
IR=	0x00000000	R11=	0x00000000	F4=	0	F29=	0
A=	0x00000000	R12=	0x00000000	F5=	2.5625	F30=	0
AHI=	0x00000000	R13=	0x00000000	F6=	0	F31=	0
B=	0x00000000	R14=	0x00000000	F7=	0	D0=	20
BHI=	0x00000000	R15=	0x00000000	F8=	0	D2=	110
ETA=	0x00000000	R16=	0x00000000	F9=	0	D4=	10
ALU=	0x00000000	R17=	0x00000000	F10=	0	D6=	0
ALUHI=	0x00000000	R18=	0x00000000	F11=	0	D8=	0
FPSR=	0x00000000	R19=	0x00000000	F12=	0	D10=	0
DMAR=	0x00000000	R20=	0x00000000	F13=	0	D12=	0
SDR=	0x00000000	R21=	0x00000000	F14=	0	D14=	0
SDRHI=	0x00000000	R22=	0x00000000	F15=	0	D16=	0
LDR=	0x00000000	R23=	0x00000000	F16=	0	D18=	0
LDRHI=	0x00000000	R24=	0x00000000	F17=	0	D20=	0
R0=	0x00000000	R25=	0x00000000	F18=	0	D22=	0
R1=	0x00000000	R26=	0x00000000	F19=	0	D24=	0
R2=	0x00000188	R27=	0x00000000	F20=	0	D26=	0
R3=	0x000001d8	R28=	0x00000000	F21=	0	D28=	0
R4=	0x00000188	R29=	0x00000000	F22=	0	D30=	0
R5=	0x00000000	R30=	0x00000000	F23=	0		
R6=	0x00000000	R31=	0x00000000	F24=	0		
R7=	0x00000000	F0=	0	F25=	0		
R8=	0x00000000	F1=	2.8125	F26=	0		



## (2) 存在的结构相关指令和部件



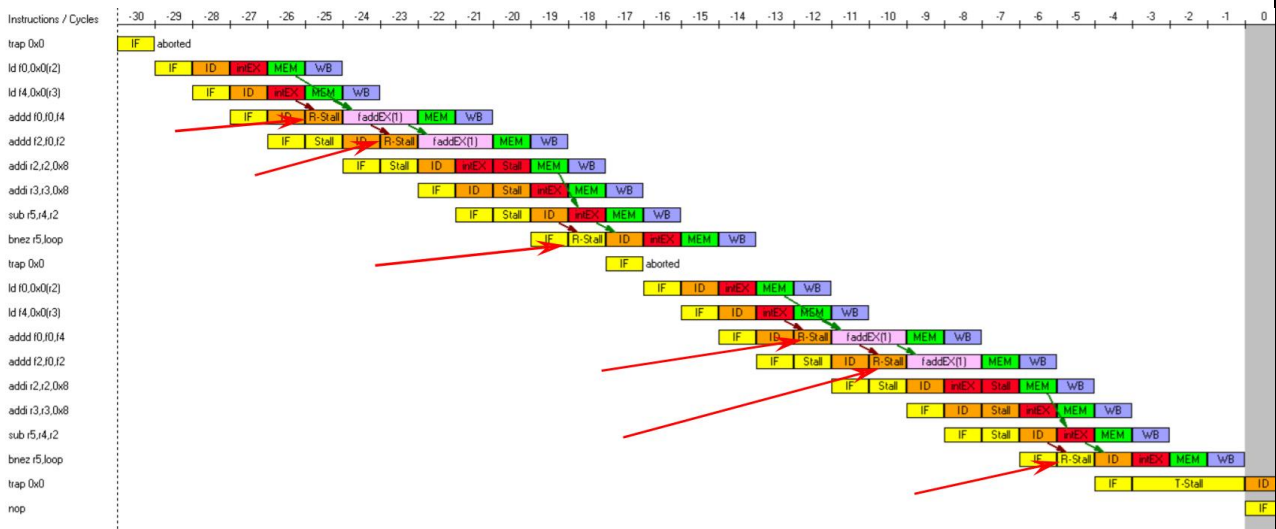
ADD F2, F0, F2 和 ADDI R2, R2, #8 都用到了浮点数加法器，就要等前一条指令用完一条指令才能用。

Information about addi f0,f0,f4		
<b>addi f0,f0,f4</b> Adr.: loop+0x8 Code: 0x04040004 Terminated successfully First Cycle: -14 Last Cycle: -8 Total Cycles: 7	IF	ID
	Cycles: -14(1) Terminated successfully IMAR<PC (=loop+0x8) IR<Mem[IMAR] (=0x04040004) PC<PC+4 (=loop+0xc) No Stalls required.	Cycles: -13(2) Terminated successfully A<D0 (=18) B<D4 (=9) 1 Stall(s) because of RAW-Hazard with ld f4,0x0(r3)
<b>faddEX[1]</b> Cycles: -11(2) Terminated successfully ALU<A+B (=20) [A=10, B=10] No Stalls required. Forwarding applicated: A<-0x0 (ld f0,0x0(r2)) AH<-0x40240000 (ld f0,0x0(r2)) B<-0x0 (ld f4,0x0(r3)) BH<-0x40240000 (ld f4,0x0(r3))	MEM	WB
	Cycles: -9(1) Terminated successfully Nothing to do. No Stalls required.	Cycles: -8(1) Terminated successfully D0<ALU (=20) No Stalls required.

OK

Information about addd f2,f0,f2		
<b>addd f2,f0,f2</b> Adr.: loop+0xc Code: 0x04021004 Terminated successfully First Cycle: -13 Last Cycle: -6 Total Cycles: 8	IF	ID
	Cycles: -13(2) Terminated successfully IMAR<-PC (=loop+0xc) IR<-Mem[IMAR] (=0x04021004) PC<-PC+4 (=loop+0x10) 1 Stall(s) because of structural Hazard!	Cycles: -11(2) Terminated successfully A<-D0 (=10) B<-D2 (=90) 1 Stall(s) because of RAW-Hazard with addd f0,f0,f4
<b>faddEX[1]</b> Cycles: -9(2) Terminated successfully ALU<-A+B (=110) (A=20, B=90) No Stalls required. Forwarding applicated: A<-0x0 (addd f0,f0,f4) AH1<-0x0340000 (addd f0,f0,f4)	MEM	WB
	Cycles: -7(1) Terminated successfully Nothing to do. No Stalls required.	Cycles: -6(1) Terminated successfully D2<-ALU (=110) No Stalls required.
OK		

(3) 记录由结构相关引起的暂停时钟周期数，计算暂停时钟周期数占总执行周期数的百分比。



每次循环，总共有箭头指向的 6 处结构相关，每处暂停了一个时钟周期，总共暂停了 6 个周期，总执行周期 30，百分之 20%。

(4) 论述结构相关对 CPU 性能的影响，讨论解决结构相关的方法。

影响：

- 性能下降：流水线因资源冲突暂停，降低指令吞吐量。
- 资源争用：功能部件（如浮点加法器）成为瓶颈，限制并行性。

解决方法：

- 增加硬件资源：添加多个浮点加法器，支持并发执行。

2. 流水线化部件：将功能部件划分为多级流水，允许指令重叠执行。
3. 指令调度：调整指令顺序，插入无关指令或使用编译器优化减少冲突。

#### 结论分析与体会：

##### 1. 结构相关的本质

结构相关（Structural Hazard）源于硬件资源的冲突。例如，两条连续的浮点加法指令（ADDD F0, F0, F4 和 ADDD F2, F0, F2）争用非流水线化的浮点加法器，导致第二条指令必须等待第一条指令完成才能执行，引发流水线暂停。

##### 2. 现代 CPU 的解决方案

（1）动态调度（Tomasulo 算法）：通过寄存器重命名和保留站（Reservation Station）动态分配功能部件，缓解资源冲突。

（2）超标量架构：支持同时发射多条指令到不同功能单元，结合乱序执行（Out-of-Order Execution）提升资源利用率。

##### 3. 结构相关是流水线设计中不可忽视的性能瓶颈，其影响在资源受限或指令密集的场景下尤为显著。硬件增强（如多部件、流水线化）和软件优化（指令调度、循环展开）的结合是解决结构相关的关键。