MIPS32指令系统

计组课程设计

设计报告

汇报人：陈扬 网络工程161

时间：2018年09月21日

目录

01 设计目标

- 模拟MIP32系统的运行机制
- 在win系统上模拟运行MIP32汇编指令
- 使用C++对MIP32指令系统进行仿真
- 模拟五段流水线的执行过程

02 主要功能

- 完成MIP32指令的取指、译码、计算、访存和写回五个步骤的软件模拟
- 向系统输入机器语言源程序
- 对内部寄存器进行初始花
- 运行程序
- 查看运行结果，能够反映指令的执行过程
- 模拟五段流水线的执行过程
- 解决数据相关的问题
- 反应流水线的执行过程

03 概要设计&模块划分

• 寄存器模块	register.h、register.cpp
• 内存模块	memory.h、memory.cpp
• 取值模块	fetch.h、fetch.cpp
• 译码模块	decoding.h、decoding.cpp
• 执行模块	execute.h、execute.cpp
• 程序动作记录模块	log.h、log.cpp
• 程序全局数据模块	overalldata.h
• 流水线模块	pipeline.h、pipeline.cpp
• 全局控制模块	control.h、control.cpp

04 关键技术

- 模块划分
  - 用“面向对象”的方法，将每一过程整合为一个类
  - 取值、译码、执行、访存、写回，都有自己对应的模块
- 流水线生成
  - 二维坐标系，过程可视化
  - 基于时间段
- 暂停控制
  - 自动检测数据相关，并利用暂停执行加以解决
  - 利用位运算方式，加快指令的检测速度
- 全局控制
  - 对每一个模块进行全局调动
  - 保证模块可删减
- 流水线绘图
  - 利用graphics.h功能进行绘图
  - 流水线更加形象生动

流水线生成

- 构造一个二维XY坐标系，每一个坐标即对应流水线中每一条指令在某一个时间点所对应的动作

  F-Fetch　　D-Decode　　E-Execute
  M-Memory　W-WriteBack　U-Wait
  B-Ban

- 构造一个一维int（32位）数组
- 每一个数组的int值有32位，即可对应32个寄存器

  利用位运算，可对int数据进行读取、存储操作
  写1：int |= (1<<address)
  写0：int &= ~(1<<address)
  读：　return (int>>address)&1

流水线生成

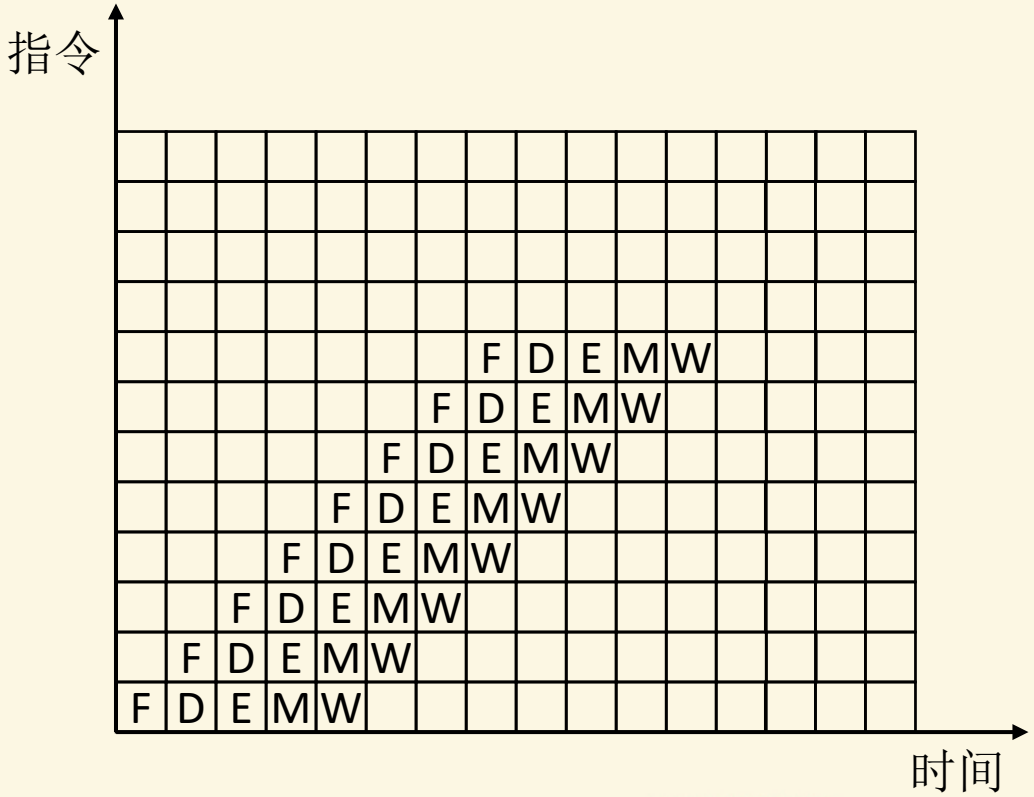
- **取值：**根据PC的值，从存储器中取指令并装入到IR中，同时PC+4
- **译码/读寄存器：**译码，读寄存器堆，判断是否为转移指令，计算下条地址
- **地址/执行周期：**对上一周期准备好的操作数进行运算
- **访问存储器：**load根据上一周期的有效地址进行读；store根据上一周期的有效地址进行写
- **写回：**将结果写入寄存器堆

流水线生成

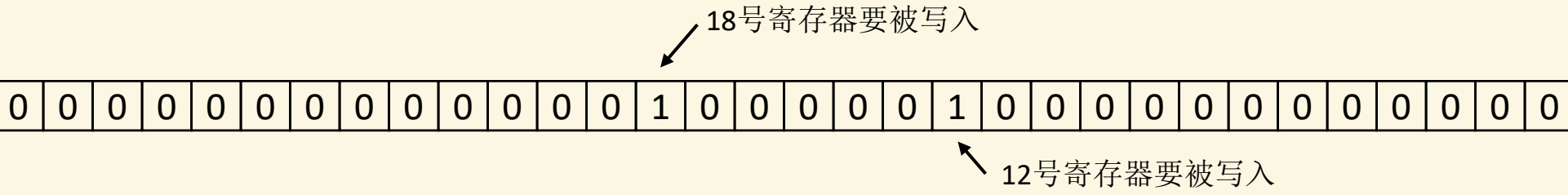
for循环：第一条指令开始
得到指令起始时间位置address
执行Fetch操作，获取指令编码
address地址+1，查看下一时间点
Decode解码，得到该寄存器的类型，需要的读/写的寄存器
如果要读的寄存器被占用要写
等待，直到该寄存器被释放
否则，要读的寄存器不被占用
执行Execute操作
执行Memory操作
判断，该条指令是否先于上一条指令完成
如果先于上一条指令完成，则等待
否则，执行WriteBack操作
在下一时间点，对本条指令所用的寄存器进行释放
结束

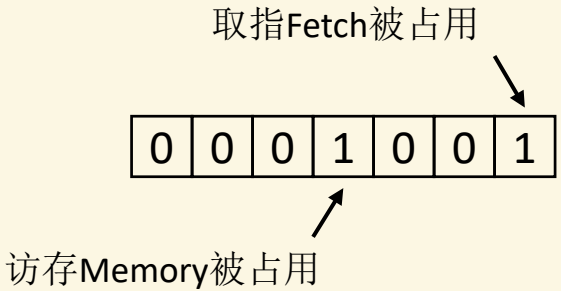


流水线生成



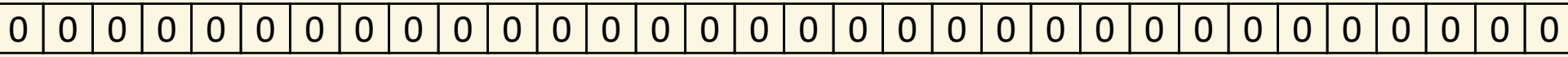
流水线生成


18号寄存器要被写入
0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0
12号寄存器要被写入


取指Fetch被占用
0 0 0 1 0 0 1
访存Memory被占用

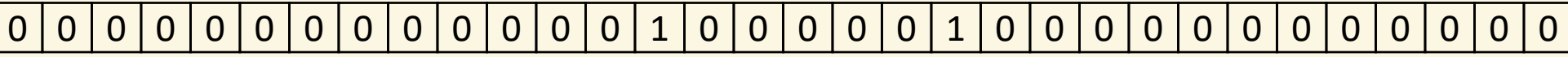
流水线生成

某时间点的寄存器状态标识（int）



该指令需要的寄存器标识（int）

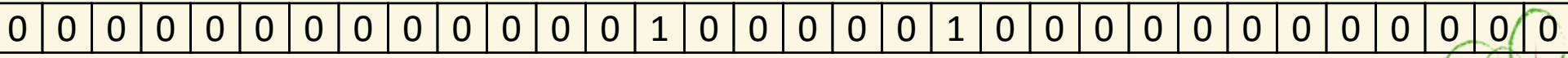
18号寄存器要被写入



12号寄存器要被写入

两数据进行或运算，得：

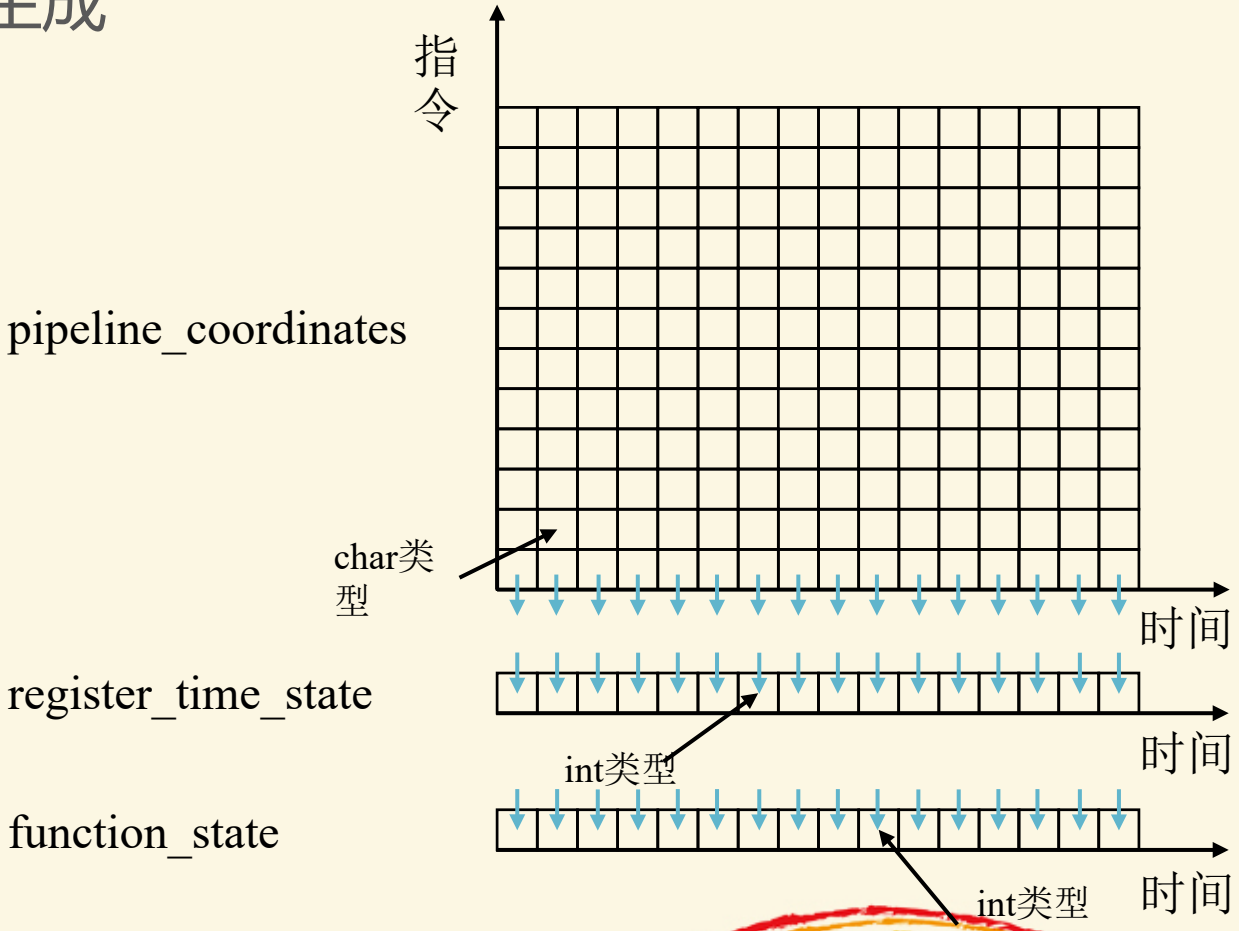
18号寄存器在该时间点被占用



12号寄存器在该时间点被占用

在该时间点所有被占用的寄存器标识

流水线生成


指令
pipeline_coordinates
char类型
时间
register_time_state
int类型
时间
function_state
int类型
时间

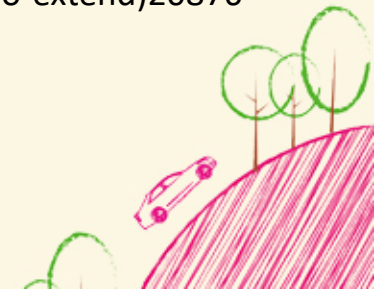
05 实验设计及实验结果

- 全部流通的指令
- 试图写入$zero的指令
- 存在数据相关的指令
- 既存在试图写$zero寄存器的指令，又存在数据相关的指令

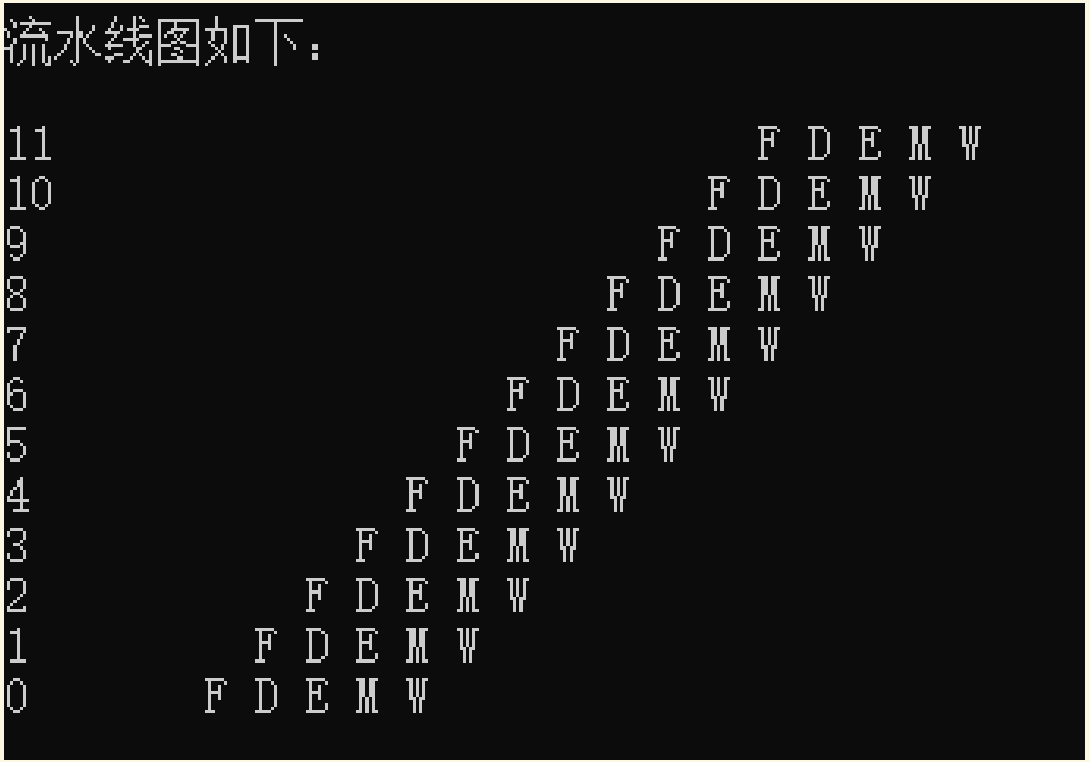
全部流通的指令

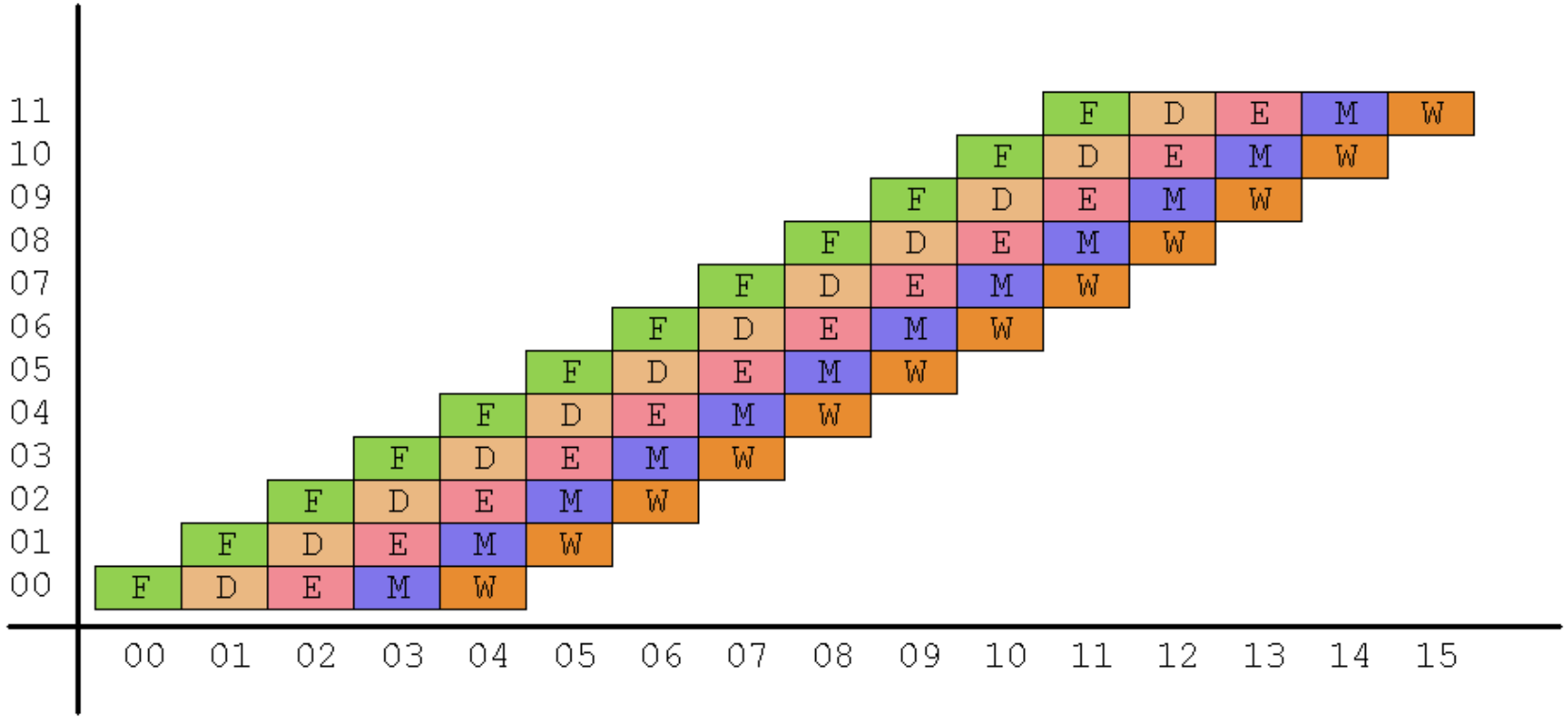
00000000001000100001100000100001	addu $v1, $at, $v0	$r3 <- $r1 + $r2
00000000100001010011000000100011	subu $a2, $a0, $a1	$r6 <- $r4 - $r5
00000000111010000100100000100100	and $t1, $a3, $t0	$r9 <- $r7 & $r8
00000001010010110110000000100101	or $t4, $t2, $t3	$r12 <- $r10 | $r11
00000001101011100111100000100110	xor $t7, $t5, $t6	$r15 <- $r13 ^ $r14
00000010000100011001000000100111	nor $s2, $s0, $s1	$r18 <- $r16 NOR $17
00000000000100111010000010000000	sll $s4, $s3, 2	$r20 <- $r19 << 2
00000000000101011011000100000010	srl $s6, $s5, 4	$r22 <- $r21 >> 4
00100010111110000110101001010010	addi $t8, $s7, 27218	$r24 <- $r23 +(sign-extend)27218
00110011001110101010101101010100	andi $k0, $t9, 43860	$r26 <- $r25 & (zero-extend)43860
00110111011111000101010001100110	ori $gp, $k1, 21606	$r28 <- $r27 | (zero-extend)21606
00111011101111100101000110001100	xori $fp, $sp, 20876	$r30 <- $r29 ^ (zero-extend)20876





流水线图如下：

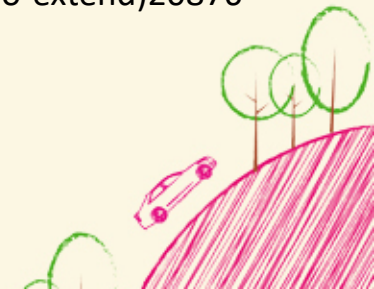

11 F D E M W
10 F D E M W
9 F D E M W
8 F D E M W
7 F D E M W
6 F D E M W
5 F D E M W
4 F D E M W
3 F D E M W
2 F D E M W
1 F D E M W
0 F D E M W


11
10
09
08
07
06
05
04
03
02
01
00
F D E M W
00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15

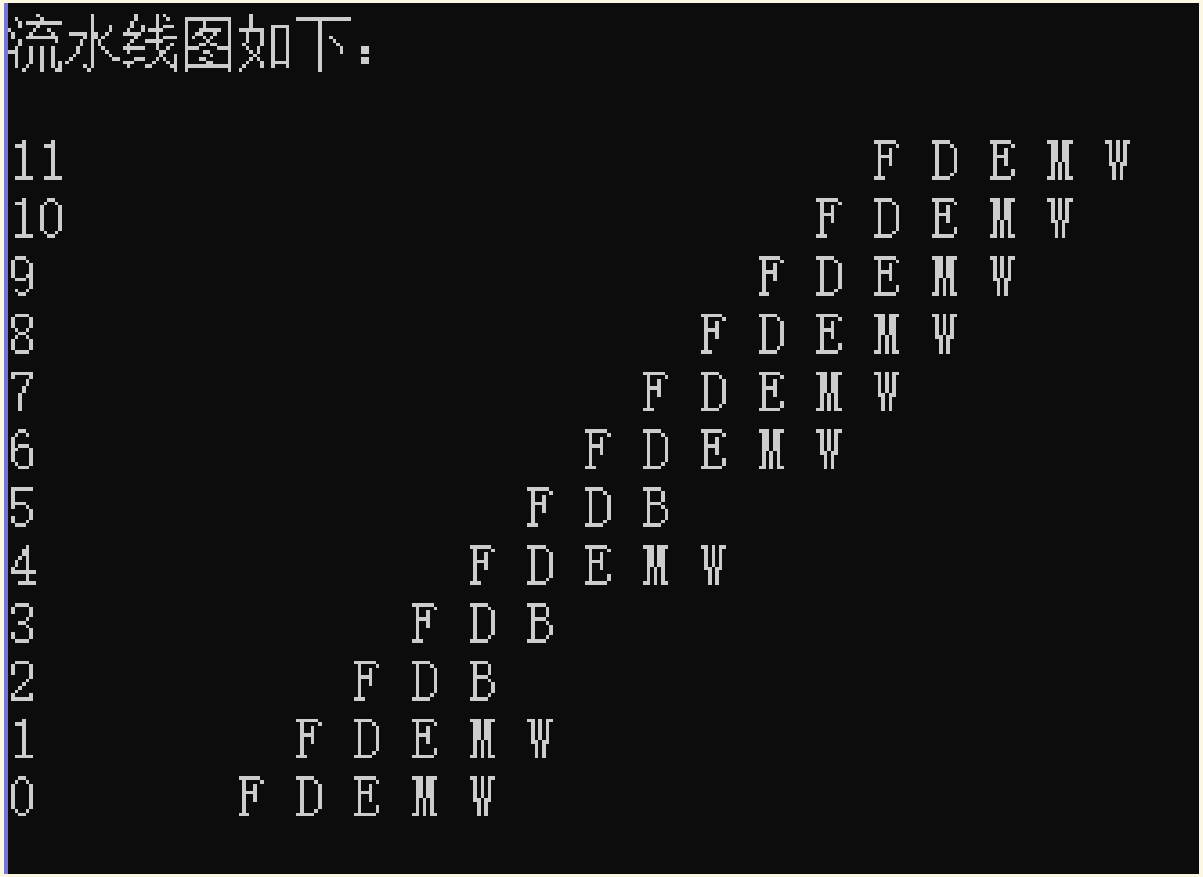
试图写入$zero的指令

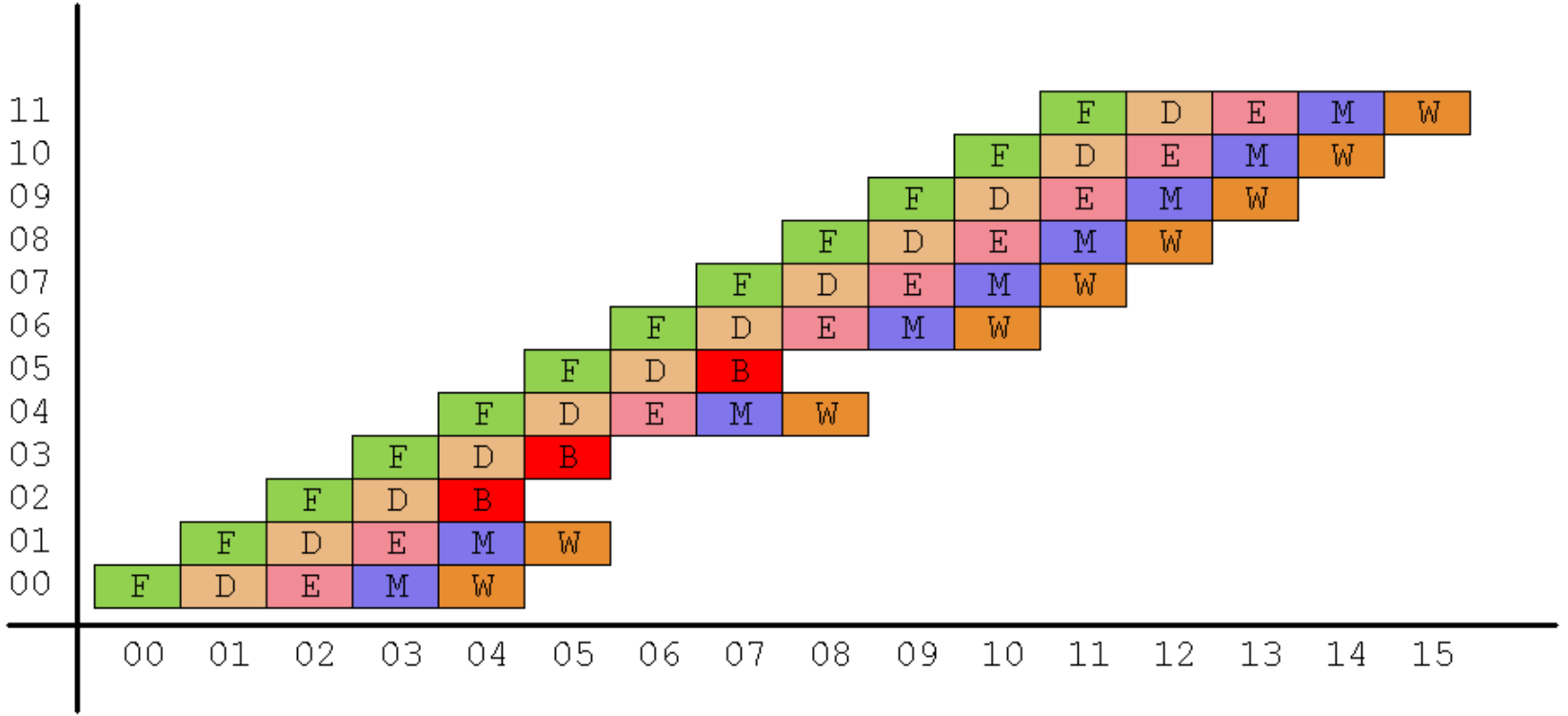
00000000001000100001100000100001	addu $v1, $at, $v0	$r3 <- $r1 + $r2
00000000100001010011000000100011	subu $a2, $a0, $a1	$r6 <- $r4 - $r5
00000000111010000000000000100100	and $zero, $a3, $t0	$zero <- $r7 & $r8
00000001010010110000000000100101	or $zero, $t2, $t3	$zero <- $r10 | $r11
00000001101011100111100000100110	xor $t7, $t5, $t6	$r15 <- $r13 ^ $r14
00000010000100010000000000100111	nor $zero, $s0, $s1	$zero <- $r16 NOR $17
00000000000100111010000010000000	sll $s4, $s3, 2	$r20 <- $r19 << 2
00000000000101011011000100000010	srl $s6, $s5, 4	$r22 <- $r21 >> 4
00100010111110000110101001010010	addi $t8, $s7, 27218	$r24 <- $r23 +(sign-extend)27218
00110011001110101010101101010100	andi $k0, $t9, 43860	$r26 <- $r25 & (zero-extend)43860
00110111011111000101010001100110	ori $gp, $k1, 21606	$r28 <- $r27 | (zero-extend)21606
00111011101111100101000110001100	xori $fp, $sp, 20876	$r30 <- $r29 ^ (zero-extend)20876





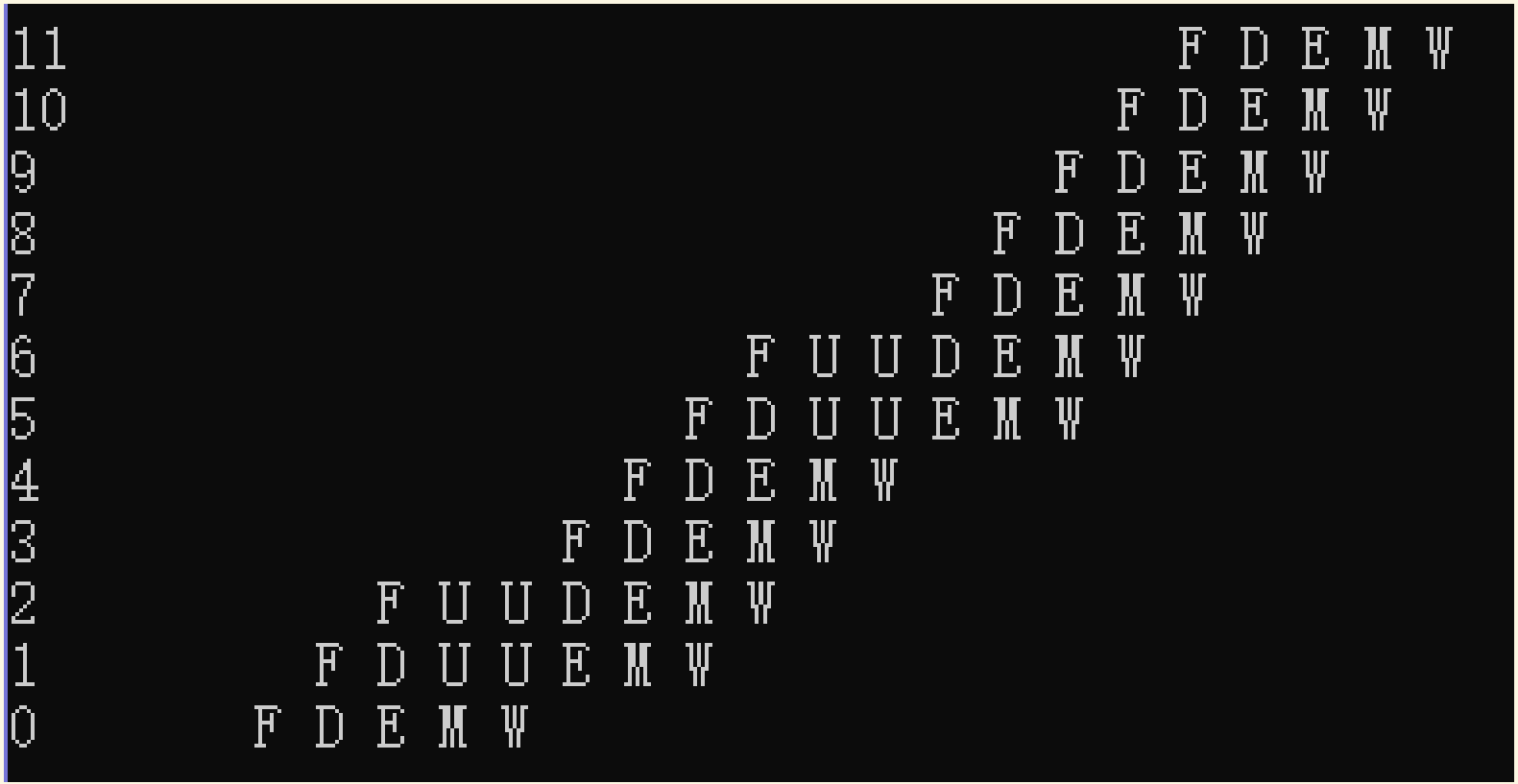
流水线图如下：

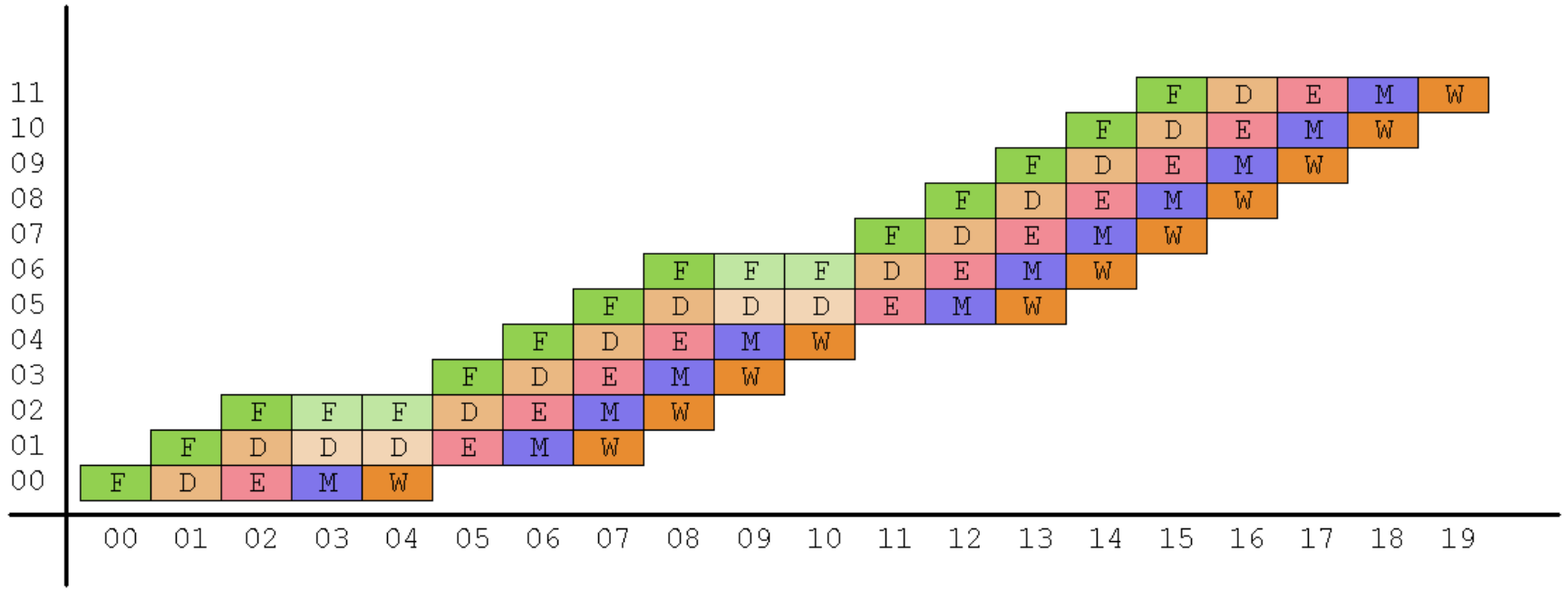

11 F D E M W
10 F D E M W
9 F D E M W
8 F D E M W
7 F D E M W
6 F D E M W
5 F D B
4 F D E M W
3 F D B
2 F D B
1 F D E M W
0 F D E M W


11
10
09
08
07
06
05
04
03
02
01
00
F D E M W
F D E M W
F D E M W
F D E M W
F D E M W
F D E M W
F D B
F D E M W
F D B
F D B
F D E M W
00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15

存在数据相关的指令

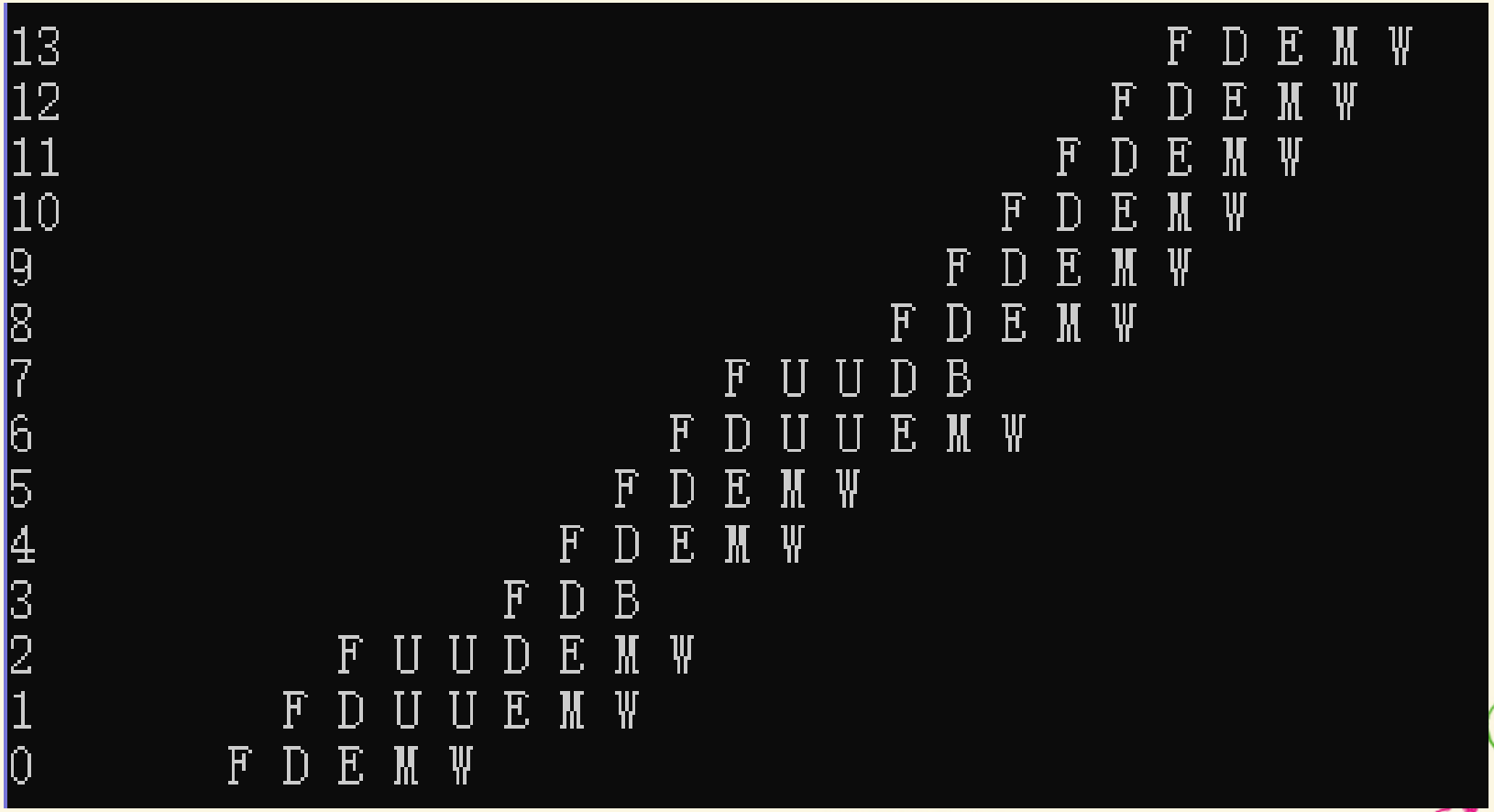
00000000001000100001100000100001	addu $v1, $at, $v0	$r3 <- $r1 + $r2
00000000011001010011000000100011	subu $a2, $v1, $a1	$r6 <- $r3 - $r5
00000000111010000100100000100100	and $t1, $a3, $t0	$r9 <- $r7 & $r8
00000001010010110110000000100101	or $t4, $t2, $t3	$r12 <- $r10 | $r11
00000001101011100111100000100110	xor $t7, $t5, $t6	$r15 <- $r13 ^ $r14
00000001111100011001000000100111	nor $s2, $t7, $s1	$r18 <- $r15 NOR $17
00000000000100111010000010000000	sll $s4, $s3, 2	$r20 <- $r19 << 2
00000000000101011011000100000010	srl $s6, $s5, 4	$r22 <- $r21 >> 4
00100010111110000110101001010010	addi $t8, $s7, 27218	$r24 <- $r23 +(sign-extend)27218
00110011001110101010101101010100	andi $k0, $t9, 43860	$r26 <- $r25 & (zero-extend)43860
00110111011111000101010001100110	ori $gp, $k1, 21606	$r28 <- $r27 | (zero-extend)21606
00111011101111100101000110001100	xori $fp, $sp, 20876	$r30 <- $r29 ^ (zero-extend)20876

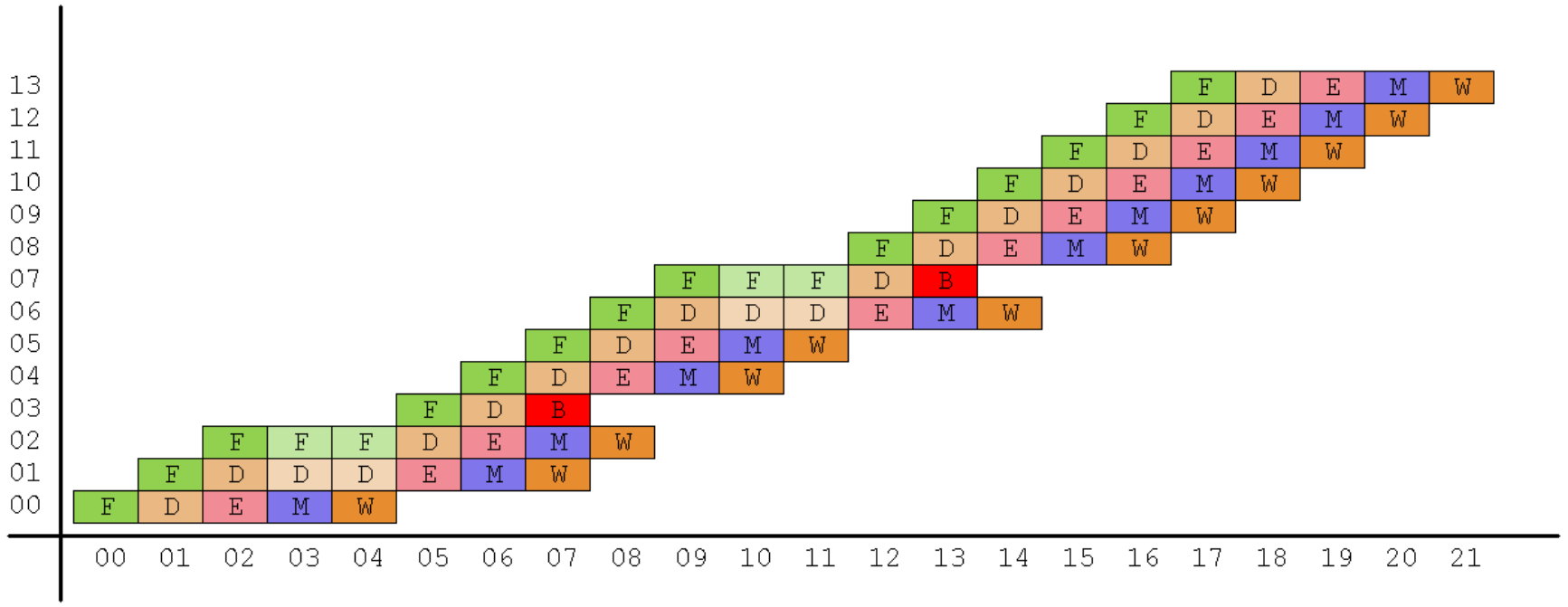

11 F D E M W
10 F D E M W
9 F D E M W
8 F D E M W
7 F D E M W
6 F U U D E M W
5 F D U U E M W
4 F D E M W
3 F D E M W
2 F U U D E M W
1 F D U U E M W
0 F D E M W


11
10
09
08
07
06
05
04
03
02
01
00
F D E M W
00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19

试图写$zero & 数据相关

00000000001000100001100000100001	addu $v1, $at, $v0	$r3 <- $r1 + $r2
00000000011001010011000000100011	subu $a2, $v1, $a1	$r6 <- $r3 - $r5
00000000111010000100100000100100	and $t1, $a3, $t0	$r9 <- $r7 & $r8
00000001010010110000000000100101	or $zero, $t2, $t3	$zero <- $r10 | $r11
00000001010010110110000000100101	or $t4, $t2, $t3	$r12 <- $r10 | $r11
00000001101011100111100000100110	xor $t7, $t5, $t6	$r15 <- $r13 ^ $r14
00000001111100011001000000100111	nor $s2, $t7, $s1	$r18 <- $r15 NOR $17
00000000111010000000000000100100	and $zero, $a3, $t0	$zero <- $r7 & $r8
00000000000010011010000010000000	sll $s4, $s3, 2	$r20 <- $r19 << 2
00000000000101011011000100000010	srl $s6, $s5, 4	$r22 <- $r21 >> 4
00100010111110000110101001010010	addi $t8, $s7, 27218	$r24 <- $r23 +(sign-extend)27218
00110011001110101010101101010100	andi $k0, $t9, 43860	$r26 <- $r25 & (zero-extend)43860
00110111011111000101010001100110	ori $gp, $k1, 21606	$r28 <- $r27 | (zero-extend)21606
00111011101111100101000110001100	xori $fp, $sp, 20876	$r30 <- $r29 ^ (zero-extend)20876


13 F D E M W
12 F D E M W
11 F D E M W
10 F D E M W
9 F D E M W
8 F D E M W
7 F U U D B
6 F D U U E M W
5 F D E M W
4 F D E M W
3 F D B
2 F U U D E M W
1 F D U U E M W
0 F D E M W


13
12
11
10
09
08
07
06
05
04
03
02
01
00
00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21
F D E M W
F D E M W
F D E M W
F D E M W
F D E M W
F D E M W
F D D D E M W
F D E M W
F D E M W
F D E M W
F D B
F F F D E M W
F D E M W
F D E M W
F D B
F F F D E M W
F D D D E M W
F D E M W
F D E M W
F D E M W

06 特色之处

- 清晰的模块划分
- 可重新组装的模块、框架
- 自行设计的流水线生成算法
- 在流水线生成过程中使用位运算进行优化
- 流水线的绘图
- 流水线的单步模拟生成

07 小组成员&任务分配

独自承担并完成所有
开发工作

08 心得体会

- 对“面向对象”概念的更深理解
- 程序架构的设计经验与体会
- 对“流水线”技术的更深理解
- MIP32指令集的运行原理
- 位运算作用的理解

谢谢！

汇报人：陈扬 网络工程161　　时间：2018年09月21日