一、术语解释

**单功能性流水线:** 只能完成一种功能的流水线。在计算机中要实现多个功能，都采用多个单功能流水线。

**多功能流水线：**同一个流水线可有多种连接方式来实现多种功能。

**静态流水线：**同一时间内，流水线的各段只能按同一种连接方式工作。

**动态流水线：**在同一时间内，流水线的各段可按不同运算的连接方式工作。

**线性流水线：**流水线各段串行连接，没有反馈回路。

**非线性流水线：** 流水线中除了串行连接通路，还有反馈回路。在流水过程中，有些段要反复多次使用，它常用于递归或组成多功能流水线。

**超标量处理机：**超标量处理机中使用了多指令流水线，每个时钟周期同时发射多条指令并产生多个结果。

**超流水线处理机：**超流水线处理机是通过各部分硬件的充分重叠工作来提高处理机性能。一台并行度ILP为*n*的超流水线处理机，在一个时钟周期内能发射*n*条指令，每隔1/*n*个时钟周期发射一条指令。

**控制相关：**当流水线遇到转移指令或其他改变PC值的指令而造成断流时，会引起控制相关。

**数据相关：**在一个程序中，存在必须等前一条指令执行完才能获得需要的数据，然后执行下一条指令的情况，那么这两条指令数据相关。

**先行控制方式：**在执行部件执行第*k*条指令的同时，指令控制部件对其后继的*k*+1，*k*+2…条指令进行预取和预处理，为执行部件执行新的指令做好必要和充分的前期准备。这样，就能使指令分析部件和指令执行部件连续、流畅地工作。流程中指令分析部件和执行部件之间有等待的时间间隔Δ*t*，但它们各自的流程却是连续的。这种方式称为先行控制方式。

**写后读相关：**假设指令j是在指令i后面执行的指令，RAW表示指令i将数据写入寄存器后，指令j才能从这个寄存器读取数据。如果指令j在指令i写入寄存器前尝试读出该寄存器的内容，将得到不正确的数据。

**写后写相关：**假设指令j是在指令i后面执行的指令，WAW表示指令i将数据写入寄存器后，指令j才能将数据写入这个寄存器。如果指令j在指令i之前写该寄存器，将使得该寄存器的值不是最新值。

二、问答题

1. 答：一条指令的解释过程可以通过多个独立的部件完成。当一条指令的分析子过程在指令分析器中结束，并将结果送入执行部件去实现执行子过程时，指令分析器不必等本指令在执行部件中完成后再对下一条指令进行分析子过程，而是同时进行。

**流水线有如下特点：**（1）可以划分为若干互有联系的子过程（功能段）。每个功能段由专用功能部件实现。（2）实现功能段所需的时间应尽可能相等，避免因不等产生处理瓶颈，造成流水线“断流”。（3）形成流水线处理，需要一段准备时间，称为“通过时间”。只有在此之后流水过程才能够稳定。（4）指令流不能顺序执行时，会使流水过程中断；再形成流水过程，则需经过一段时间。不应经常“断流”，否则效率不会很高。（5）流水线技术适用于大量重复的程序过程，只有输入端能连续地提供服务，流水线效率才能够得到充分发挥。

1. 答：不是越多越好，增加流水线的段数，流水线的吞吐率和加速比都会提高。由于在每段的输出端都必须设置一个锁存器（或称缓冲寄存器），因此段数增多时各锁存器的延迟时间累加值也将增加，甚至有可能超出流水线各段的延迟时间的累加值。同时，增加锁存器也增加了流水线硬件价格。所以，在设计流水线时，要综合各方面的因素，根据最佳性能价格比的要求，选择流水线的最佳段数。选取时可以参考流水的加速比、吞吐量和效率等参数。
2. 答：指令流水线中的某个过程段，有时会因为阻塞，无法在规定的时间内完成它的任务，致使流水线断流。造成流水线断流的阻塞有三种，即结构阻塞、数据阻塞、控制阻塞。解决结构阻塞可以使相关的指令暂停一个或多个时钟周期执行。解决数据阻塞的方式指令暂停执行或者设置相关的专用数据通如。解决控制阻塞的方式是分支预测技术和预先取值成功和不成功两个控制流方向上的目标指令

三、填空题

1. 10、2.4、7
2. 2n+1、n+2
3. 先行读数栈、先行操作栈和后行写数栈

四、选择题

1. A
2. C
3. A

五、计算题

(1) 100ns，对齐最长一步耗时

(2) 写后读相关，下表相比未发生相关，延迟200ns

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| ADD | 取指 | 译码取数 | 运算 | 写回 |  |  |  |
| SUB |  | 取值 |  |  | 译码取数 | 运算 |  |
|  |  |  |  |  |  |  | 写回 |

1. 添加数据旁路，第一条指令运算完后，无需写回，第二条指令就可译码取数，还需延迟100ns
2. 指令如下：(中间结果寄存器R,源操作数寄存器A,最后结果F)

I1: R1 ← A1+A2

I2: R2←A3+A4

I3: R3←A5+A6

I4: R4←A7+A8

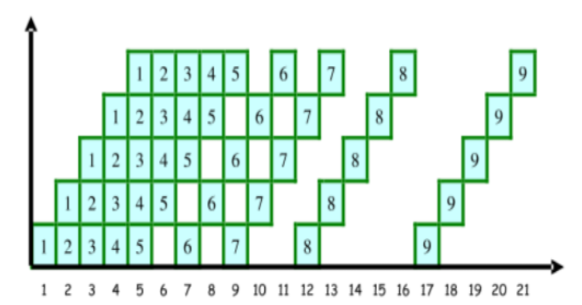
I5: R5←A9+A10

I6: R6←R1+R2

I7: R7←R3+R4

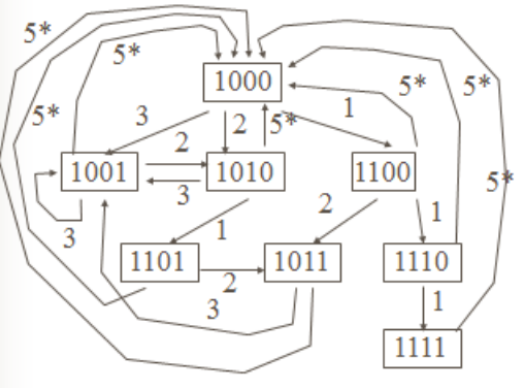
I8: R8←R5+R6

I9： F←R7+R8



吞吐率 9/(21\*t) 加速比 (9\*5)/21=2.1429 效率 (9\*5)/(21\*5)=3/7=0.43

1. 禁止向量(4),初始冲突向量(1000)
2. 如图所示



1. 如表格所示

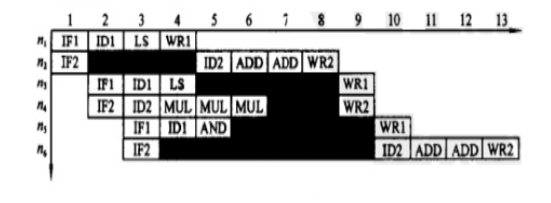
|  |  |  |  |
| --- | --- | --- | --- |
| 1,5 | 6/2 | 2,1,2,5 | 10/4 |
| 1,1,5 | 7/3 | 2,3,5 | 10/3 |
| 1,1,1,5 | 2 | 3,5 | 8/2 |
| 1,2,5 | 8/3 | 3 | 3 |
| 1,2,3,5 | 11/4 | 3,2,5 | 10/3 |
| 2,5 | 7/3 | 3,2,1,5 | 11/4 |
| 2,1,5 | 8/3 | 2,3 | 5/2 |

1. (1,1,1,5) 最小等待为2
2. n1和n2关于寄存器R0的写读数据相关 n3和n6关于寄存器R2的写读数据相关

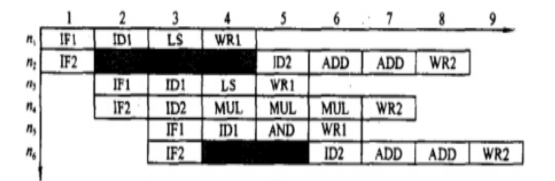
n4和n5关于寄存器R4的读写数据相关 n3和n6关于寄存器R2的写写数据相关

(2)

顺序发射顺序完成



顺序发射乱序完成



采用数据通路的顺序发射乱序完成

