北京航空航天大學
BEIHANG UNIVERSITY

计算机组成原理第五次作业

第六部分 单周期处理器设计

北京航空航天大学

计算机学院

陈麒先

16061160

二〇一七年十一月

郑重声明

关于诚实守信公约：

以下题目标号前注明*号表示本题参考了互联网资料，题目标号为红色表示本题是与同学研究后的结论，其余未注明题目均为翻阅课件、课堂笔记和教材后独立思考的结果。特此声明。

16061160

陈麒先

原创性声明

作业中出现的公式、图片、代码段和图片的文字注释信息，均为作者原创。抄袭行为在任何情况下都被严格禁止(**COPY is strictly prohibited under any circumstances**)！转载或引用须征得作者本人同意，并注明出处！

16061160

陈麒先

作业题目内容

第一题：

答：

① 对于 add 指令，控制单元需要产生 RegWrite 信号。

对于 lw 指令，控制单元需要产生 RegWrite，AluSrc，MemRead，MemToReg 信号。

对于上述两个指令，控制单元产生的 AluOp 信号均为（3’b000）加法信号。

② 对于 add 指令，需要用到 IFU （取指运算单元，包括 IM 和 PC），GPR （寄存器堆），ALU （算术逻辑单元）。

对于 lw 指令，需要用到 IFU （取指运算单元，包括 IM 和 PC），GPR （寄存器堆），ALU （算术逻辑单元），DM（数据存储单元）。

③ NPC 的第二级 ALU 运算结果会产生输出，但由于无条件分支指令，该结果不会被利用。

对于 add 指令，DM 不产生输出，因为 DM 的控制信号 MemRead＝0；而对于 lw 指令，所有部件均产生输出，只不过根据指令行为对输出结果选择性使用。

④ IM → GPR → MUX → ALU → MUX → GPR

⑤ IM → GPR → ALU → DM → MUX → GPR

⑥ IM → GPR → MUX → ALU → ANDGATE → MUX

第二题：

答：

① 可继续使用的部件有 IM，GPR，ALU。（但需要对 GPR 进行改进）。

＊question：add3 指令中 Rx 字段应存储在指令序列中的哪几位？

② 需要新增的功能单元有 ALU’ （增加在 GPR 输出和 ALU 之间），SHIFT（增加在 GPR 的 BusB 输出端口）。

③ 对于 add3 指令，需要新增对于 ALU’的控制信号 ALUOp’，使得 ALU’进行加法运算。

对于 sll 指令，则无需添加控制信号，因为 shift 的值直接由指令中的 shamt 字段决定。

④ 由于关键路径为 lw 指令所流经的数据路径，因此时钟周期由关键路径的最长

延迟决定，即 $T_C = 400 + 200 + 120 + 350 + 30 + 200 = 1300ps$

由于改进只更改了 ADD 和 GPR 的延迟，而 ADD 不在关键路径上，所以改进后的时钟周期为 $T_C' = 400 + 300 + 120 + 350 + 30 + 300 = 1500ps$

⑤ 改进后的平均时间 ： 1500 * 95% = 1425ps > 1300ps，因此无法起到加速的效果。

⑥ 该改进机制没有任何意义，既浪费成本，又降低了性能，是一种十分糟糕的改进方案。

第三题：

答：

① a. $T_C = 100 + 400 = 500ps$

b. $T_C = 500 + 150 = 650ps$

② 写出关键路径如下：IM → EXT → SHIFT → ALU → MUX

a. $T_C = 400 + 20 + 2 + 120 + 30 = 572ps$

b. $T_C = 500 + 90 + 20 + 180 + 100 = 890ps$

③ 此时需要比对关键路径是否发生改变：IM → GPR → MUX → ALU → MUX

a. $T_C = 400 + 200 + 30 + 120 + 30 = 780ps$

b. $T_C = 500 + 220 + 100 + 150 + 100 = 1070ps$

更新后的关键路径延迟变长，时钟周期更新为上述值。

④ 对于 PC 单元，任何指令都需执行 PC = PC + 4 操作

对于 DM 单元，lw 和 sw 指令需要对内存进行访问，会用到该部件。

⑤ 无法确定。这需要视各部件延迟的具体情况而定，关键路径指数据传播延迟的最大值。

⑥ 当 ADD 的延迟在 630 ps 以内变化时，对时钟周期不产生影响，只有当 ADD 延迟超过 630 ps 时，流经 ADD 的数据路径的延迟取代原来最长延迟路径，成为新的关键路径，才会对时钟周期产生影响。

而 DM 不属于两条指令所经过的路径，因此 DM 的任何改变均无法引起时钟周期的变化。

第四题：

答：

① 关键路径 ： IM → GPR → MUX → ALU → MUX → GPR

a. $T_C = 400 + 200 + 30 + 120 + 20 + 200 = 980 ps$

b. $T_C = 500 + 220 + 100 + 180 + 100 + 220 = 1320 ps$

② 关键路径 ： IM → GPR → ALU → DM → MUX → GPR

a. $T_C = 400 + 200 + 120 + 350 + 30 + 200 = 1300 ps$

b. $T_C = 500 + 220 + 180 + 1000 + 100 + 220 = 2220 ps$

③ 多条指令以最长延迟为准计算时钟周期，即以 lw 为准进行计算

a. $T_C = 400 + 200 + 120 + 350 + 30 + 200 = 1300 ps$

b. $T_C = 500 + 220 + 180 + 1000 + 100 + 220 = 2220 ps$

④ 数据存储器仅在 lw ， sw 指令中有所应用，

在 lw 指令中的占比为：

a. 350 / 1300 = 26.92%

b. 1000 / 2220 = 45.05%

在 sw 指令中的占比为：

a. 350 / 1070 = 32.71%

b. 1000 / 1900 = 52.63%

结合上述结果，知 DM 在整个时钟周期中的占比为：

a. 26.92% * 20% + 32.71% * 10% = 8.66%

b. 45.05% * 35% + 52.63% * 15% = 23.66%

⑤ 符号扩展单元需要在 addi ， beq ， lw ， sw 四个指令中使用，

先计算 addi 和 beq 指令的时钟周期。

对于 addi 指令，时钟周期为：

a. $T_C = 400 + 200 + 120 + 30 + 200 = 950 ps$

b. $T_C = 500 + 220 + 180 + 100 + 220 = 1220 ps$

对于 beq 指令，时钟周期为：

a. $T_C = 400 + 200 + 30 + 120 + 30 = 780ps$

b. $T_C = 500 + 220 + 100 + 180 + 100 = 1100ps$

EXT 单元在 addi 指令中时间占比为：

a. 20 / 950 = 2.11%

b. 20 / 1220 = 1.64%

EXT 单元在 beq 指令中时间占比为：

a. 20 / 780 = 2.56%

b. 20 / 1100 = 1.82%

EXT 单元在 lw 指令中时间占比为：

a. 20 / 1300 = 1.54%

b. 20 / 2220 = 0.90%

EXT 单元在 sw 指令中时间占比为：

a. 20 / 1070 = 1.87%

b. 20 / 1900 = 1.05%

综合上述结果，知 EXT 单元在时钟周期中的占比为：

a. 2.11% * 15% + 2.56% * 20% + 1.54% * 20% +1.87% * 10% = 1.32%

b. 1.64% * 5% + 1.82% * 15% + 0.90% * 35% + 1.05% * 15% = 0.83%

在未用到该输入时，符号扩展电路闲置，故造成了单周期处理器的资源浪费。

⑥ 由于 IM 单元在处理器中使用频度高，且延迟基数大，故应着重减小该单元的延迟。

第五题：

答：

① 若要测试 MemToReg 信号有固定为 0 的缺陷则需要使用 lw 指令。测试 IM 输出的某一位，则需要使用 beq（跳转类）指令，通过 NPC 的值来确定是否有缺陷。

② 若要测试 MemtoReg 信号有固定为 1 的缺陷则需要 R 类运算指令，同时在 DM 中预置一个与运算结果不一致的数据，通过执行不同的 R 类运算指令，检测写回 GPR 中的数值若为 DM 中的值，则有固定为 1 的缺陷。测试 IM 同样需要用到 beq （跳转类）指令。

重复几次上述测试，修改数据，即可判断是固定为 1 还是 0 的缺陷。

③可以继续使用，只需将无法正常使用的指令替换为功能相同且不受缺陷影响的指令即可。

④若要测试 MemRead 信号有固定为 1 或 0 的缺陷则需要 sw ， lw 指令,若每次执行玩 sw 指令再执行 lw 指令，观察到取值没有被更新为写入的值，则有固定为 1 的缺陷，若每次 lw 指令的 WD 输出都是 x，则有固定为 0 的缺陷。

⑤若要测试 Jump 信号有固定为 1 或 0 的缺陷则需要用到 j 指令。对 PC 的值进行观察，同时要对指令第[25:0]位预置特定数值，观察 NPC 的数值变化，若在 j 指令控制下恒有 NPC = PC + 4，则有固定为 0 的缺陷，而若无论什么指令输入，NPC 都在 PC + 4 的基准上有一个与指令第[25:0]位相关的地址偏移，则有固定为 1 的缺陷。

⑥根据被测试信号的功能相关性，可以优化测试方案。

第六题：

答：

① a.0x8cc10028；b.0x1422ffff。

② a.$6,被读；b.$1,未被读。

③ a.$1,被写；b.$2，未被写。

④ a.0，1；b.0，0。

⑤ RegDst = add;

MemRead = lw;

RegWrite = add + lw;

第七题：

答：

①原关键路径长度：IM → GPR → ALU → DM → MUX → GPR

T = 400 + 200 + 120 + 30 + 350 = 1100 ps

控制单元路径：IM → CTRL → DM → MUX → GPR

T_MAX = 200 + 120 = 320 ps

②最不关键的信号为 MemToReg ，最多可以有

T_MAX = 200 + 120 + 350 = 670 ps 的延迟。

③最关键的信号为 RegDst ，若想避免其成为关键路径，则其延迟需为 0。

④由于 RegDst 信号产生延迟对关键路径产生影响，故时钟周期更新为：

T = 400 + 720 + 30 + 200 + 120 + 350 + 30 = 1250 ps

⑤若追求性能的最优化，则需以最优时间为基准，去改善对关键路径有影响的控制信号：则需要对 RegDst 信号进行优化，其优化幅度为：RegDst 信号 720 ps。因此总优化成本为 W = 720 / 5 = ￥124

⑥若追求最低成本，则需以关键路径为基准，其余信号均向最长时间方向找齐：则除 RegDst 信号以外，均可进行优化，各个信号的优化幅度为：

Jump = (1250 - 400 - 30) - 730 = 90 ps

MemRead = 400 + 720 + 30 + 200 + 120 - 400 = 1070 ps

MemToReg = 400 + 720 + 30 + 200 + 120 + 350 - 700 = 1120 ps

AluSrc = 400 + 720 + 200 - 200 = 1120 ps

AluOp = (1250 - 400 - 50 - 120 - 250 - 30 - 200) - 200 = 0

MemWrite = 400 + 720 + 30 + 200 + 120 - 710 = 760 ps

RegWrite = 400 + 720 + 30 + 200 + 120 + 350 + 30 - 800 = 1050 ps

综上，共可节约成本为：

S = (90 + 1070 + 1120 + 1120 + 760 + 1050) / 5 = ￥1042

故维持性能不变，通过降低非关键信号性能，可大幅度节约成本。

第八题：

答：

①a. 0x0000_0010, 0x10c_0040;

b. 0x0000_000c, 0x08c_0030。

②a. $2 的值和 0x0000_0010;

b. $1 的值和$3 的值。

③a. PC + 4, PC → ADD → MUX → MUX → PC

b. PC + 4 /PC + 4 + 0x0000_0030, PC →ADD → ALU → MUX → MUX →PC

④自左下起，逆时针：

a. 00011 → 0x0000_0010 → 无法确定（内存初值未知）→ PC +4 → PC +4

b. x → -3 → x →PC + 4 → PC + 4

⑤a. 2, 16; 4, PC; PC + 4, 0x0000_0040

b. -16, -3;PC, 4;PC + 4, 0x0000_0030

⑥a. 00010, 00011, 00011

b. 00001, 00011, xxxxx