Chapter 4. Pipelined Processor Design

Problem 4.1

Additional assumptions beyond those stated in the question:

- For Amdahl V-8 and MIPS R2000, the operations have to be aligned to their respective phases (e.g., the IF must start at phase 2 and the Decode must occur at phase 2.)
- The instruction setting the CC is immediately before the branch instruction for the analysis.

For this problem, we need to calculate the penalty cycles for executing

a. Branch instruction.

b. Conditional branch instruction that branched in-line.

c. Conditional branch instruction that branched to the target.

d. The effect of address dependencies.

For branch instruction, the instruction fetch of the instruction following the branch instruction is delayed since it is fetched during the data fetch of the branch instruction. For BC-inline, the decode is delayed until the conditional code is set. For BC-target, we must consider both the instruction immediately following BC-target (BC-target+1) as well as the instruction following it (BC-target+2). The instruction fetch for BC-target+1 is delay till the data fetch of the BC-target instruction and the instruction fetch for the BC-target+2 is delayed till the condition code is set. For the EX/LD instruction sequence, the address calculation of the LD instruction is delayed until the end of the execution cycles. For the LD/LD instruction, the address calculation is delayed until the end of data fetch cycles.

The equation for CPI is:

$$\text{CPI} = \text{BaseCPI} + \text{RunOnCPI} + \text{BrCPI} + \text{BcCPI} + \text{ExLdCPI} + \text{LdLdCPI}.$$

$$\begin{aligned}
\text{BaseCPI} &= 1. \\
\text{RunOnCPI} &= 0.6 \text{ (from study 4.3).} \\
\text{BrCPI} &= (0.05) * \text{BrPenalty}. \\
\text{BcCPI} &= (0.15) * (0.5) * (\text{BC-inlinePenalty} + \text{BC-targetPenalty}). \\
\text{ExLdCPI} &= (0.015) * \text{ExLdPenalty}. \\
\text{LdLdCPI} &= (0.04) * \text{LdLdPenalty}.
\end{aligned}$$

Base Pipeline Template

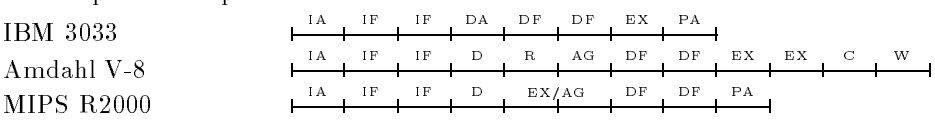


Table 2: Summary of IBM 3033

Instruction	Penalty	Execution Frequency
Br	2	5%
BC-inline	2	7.5%
BC-target	3	7.5%
EX/LD	1.54	1.5%
LD/LD	0.29	4.0%
CPI	2.2	

Instr	IBM 3033
1	Set CC
2	Branch
3	BR+1
3	BC-inline+1
3	BC-target+1
4	BC-target+2
3	Ex/Ld
3	Ld/Ld

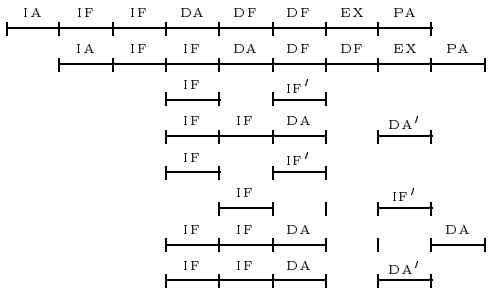


Table 3: Summary of Amdahl V-8

Instruction	Penalty	Execution Frequency
Br	4	5%
BC-inline	4	7.5%
BC-target	4	7.5%
EX/LD	2.17	1.5%
LD/LD	0.28	4.0%
CPI	2.54	

Instr	Amdahl V-8 (Phase)
1	
2	Instr
3	BR+1
3	BC-inline+1
3	BC-target+1
4	BC-target+2
3	Ex/Ld
3	Ld/Ld

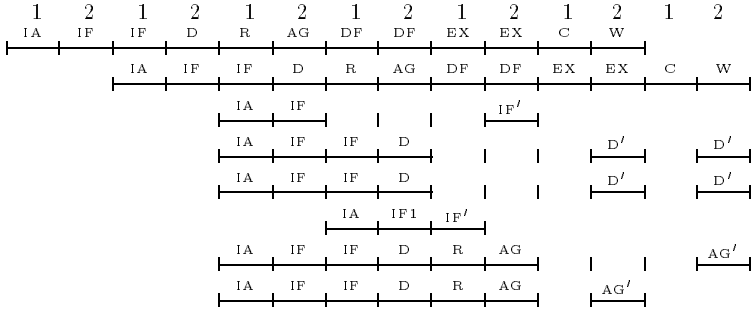
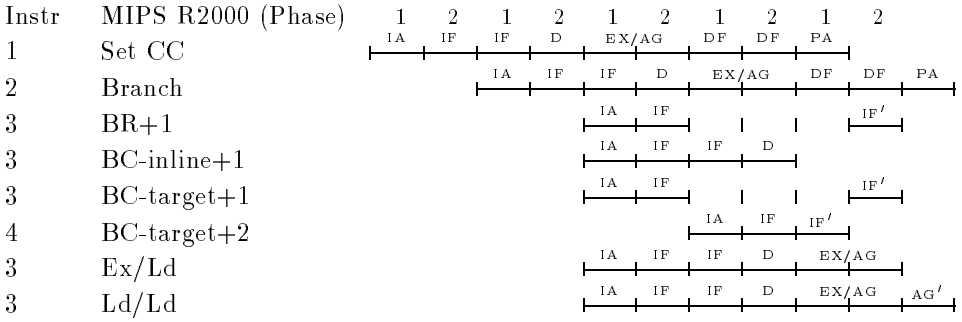


Table 4: Summary of MIPS R2000

Instruction	Penalty	Execution Frequency
Br	4	5%
BC-inline	2	7.5%
BC-target	3	7.5%
EX/LD	1.54	1.5%
LD/LD	0.29	4%
CPI	1.41	



Problem 4.3

For this problem, we need to calculate the following based on data from Chapter 3. Use pipeline template from problem 4.1.

a. EX/LD and LD/LD frequencies.

b. BR, BC-inline, BC target frequencies.

c. Calculate weighted penalty for EX/LD and LD/LD based on Tables 3.19 and 3.20.

a. We need to approximate the possibility of a Load instruction following a instruction that generates its data address. We use the data from Table 3.15.

   The occurrence of EX type instruction followed by a LOAD is $0.5 * 0.4 = 0.2$ for the R/M machines.

   The occurrence of a LOAD followed by a LOAD is 0.4 * 0.4 = 0.16 for the R/M machines.

   The occurrence of EX type instruction followed by a LOAD is $0.5 * 0.5 = 0.25$ for the L/S machines.

   The occurrence of a LOAD followed by a LOAD is $0.5 * 0.5 = 0.25$ for the L/S machines.

b. We use Table 3.10 for this calculation. To calculate the frequency of BR, BC-inline, and BC-target, we need to perform the following calculations. We also use R/M data for the IBM 3033 and AmdahlV-8 and L/S data for the MIPS R2000.

$$\begin{aligned} \text{BrFreq} &= \text{Type1Freq} * \text{AGFreq} * \text{UnconditionalFreq} = 1.8\% \\ \text{BC-inline} &= \text{Type1Freq} * \text{AGFreq} * \text{ConditionalFreq} * \text{InlineFreq} + \text{Type2Freq} * \text{InLineFreq} \\ &= 3.4\% \end{aligned}$$

$$\begin{aligned}\text{BC-target} &= \text{Type1Freq} * \text{AGFreq} * \text{ConditionalFreq} * \text{TargetFreq} \\ &\quad + \text{Type2Freq} * \text{TargetFreq} + \text{Type3Freq} \\ &= 7.9\end{aligned}$$

c. The weighted penalty is simply the sum of Prob (distance $= n$) $*$ penalty (distance $= n$), where n is the number of penalty cycles to 1.

Table 5: Summary of IBM 3033

Instruction	Penalty	Execution Frequency
BR	2	1.8%
BC-inline	2	3.4%
BC-target	3	7.9%
EX/LD	1.54	20%
LD/LD	0.29	16%
CPI	1.7	

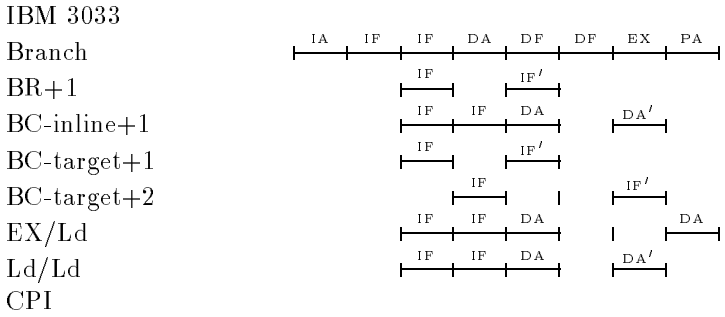


Table 6: Summary of Amdahl V-8

Instruction	Penalty	Execution Frequency
BR	4	1.8%
BC-inline	4	3.4%
BC-target	4	7.9%
EX/LD	2.17	20%
LD/LD	0.28	16%
CPI	2.0	

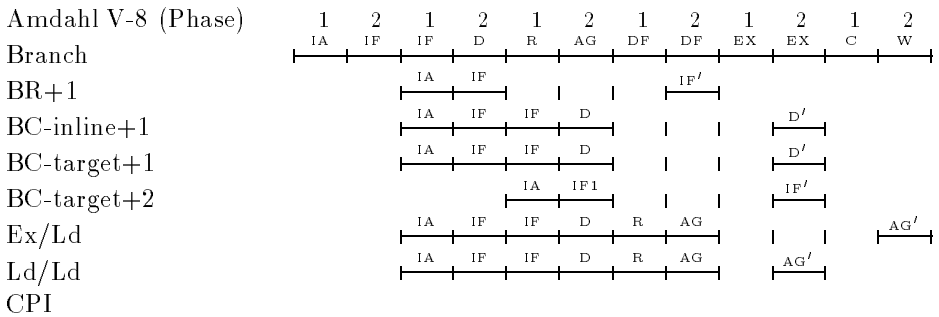
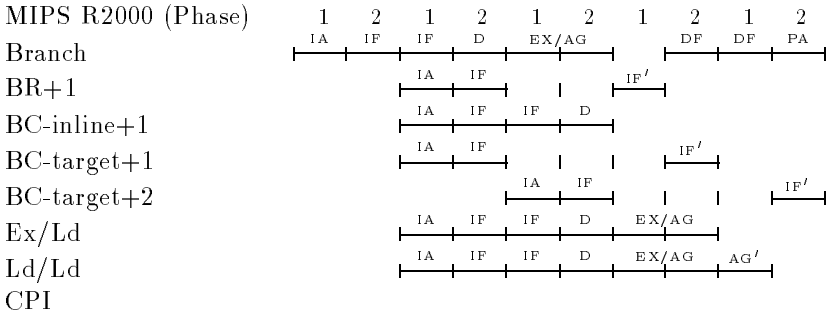


Table 7: Summary of MIPS R2000

Instruction	Penalty	Execution Frequency
BR	4	1.55%
BC-inline	2	3.0%
BC-target	3	6.8%
EX/LD	1.54	25%
LD/LD	0.29	25%
CPI	1.41	



Problem 4.5

This problem follows the same steps described in Study 4.4. The difference between early CC setting and delay branch is:

- Early CC hides the time the CPU is stalled waiting for the condition code to set. As a result, early CC setting makes it possible to hide the delay of BC-inline completely, but it can only hide BC-target up to the BR delay.

- Delay branch hides both the time the CPU is stalled waiting for condition code and the time to fetch the target instruction by inserting the instruction after the BC itself. As a result, it is possible to hide both BC-inline and BC-target delays completely.

To compute the excess CPI due to branch instructions for these two schemes, we follow the procedure of problem 4.1, where

BcCPI = $(0.15) * (0.5) *$ (BC-inline penalty + BC-target penalty).

The effectiveness of either scheme depends on the compiler's ability to insert instructions for both schemes. The results of the analysis are shown in Tables 8–13.

(a) IBM 3033

BaseCpi	1
Run-on	0.6

Table 8: Using Early CC setting for MIPS R2000

BC	$n=0$	$n=1$	$n=2$	$n=3$	$n=4$
BC-inline penalty	4	3	2	1	0
BC-target penalty	4	4	4	4	4
ExcessCPI	0.6	0.525	0.45	0.375	0.3

Table 9: Using delay branch for MIPS R2000

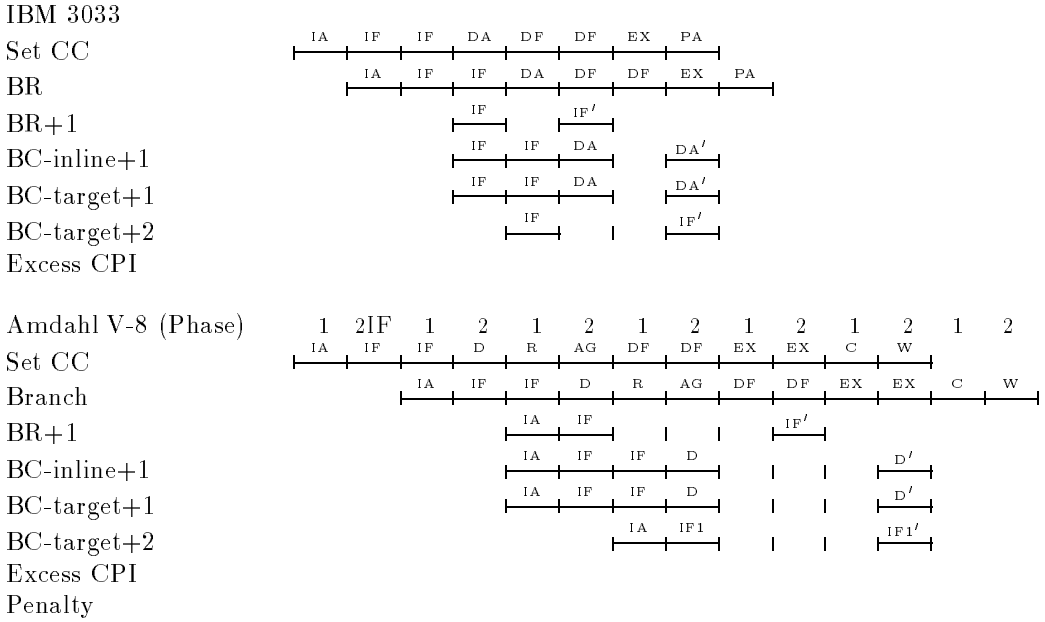
BC	$n=0$	$n=1$	$n=2$	$n=3$	$n=4$
BC-inline penalty	4	3	2	1	0
BC-target penalty	4	3	2	1	0
ExcessCPI	0.6	0.45	0.3	0.15	0

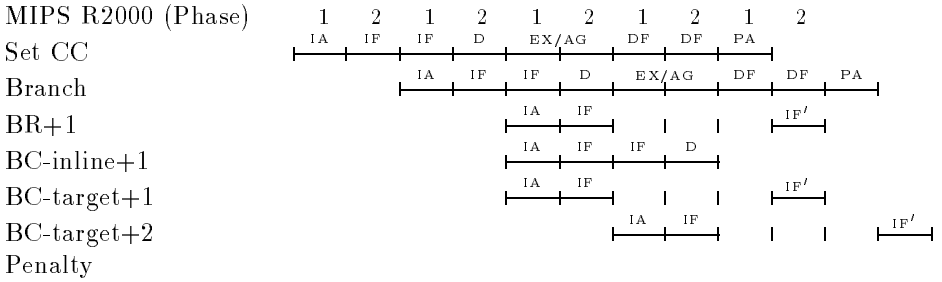
Table 10: Using Early CC setting for IBM 3033

BC	$n=0$	$n=1$	$n=2$	$n=3$
BC-inline penalty	2	1	0	0
BC-target penalty	3	2	2	2
ExcessCPI	0.375	0.225	0.15	0.15

Table 11: Using delay branch for IBM 3033

BC	$n=0$	$n=1$	$n=2$	$n=3$
BC-inline penalty	2	1	0	0
BC-target penalty	3	2	1	0
ExcessCPI	0.375	0.225	0.075	0





(b) Amdahl V-8

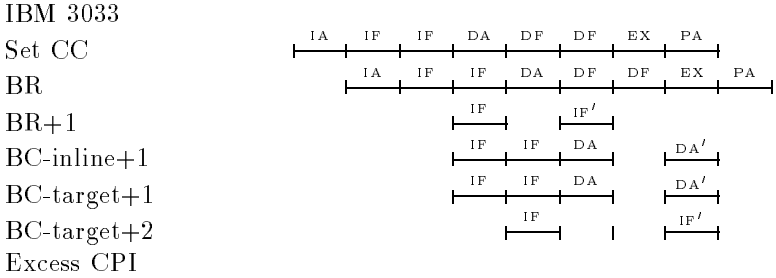
BaseCpi	1
Run-on	0.6

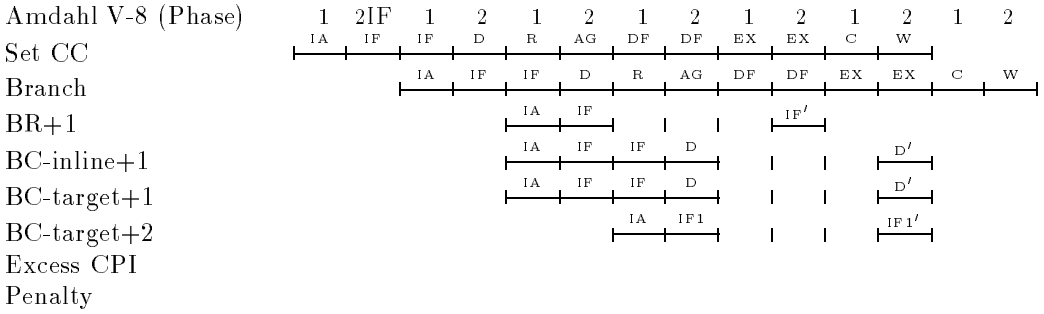
Table 12: Using Early CC setting for Amdahl-V8

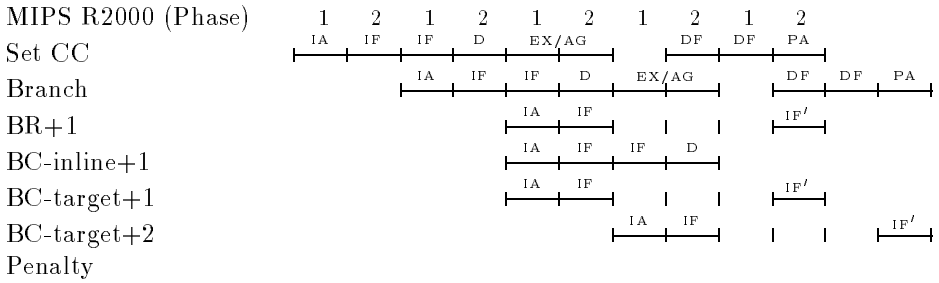
BC	$n=0$	$n=1$	$n=2$	$n=3$	$n=4$
BC-inline penalty	4	3	2	1	0
BC-target penalty	4	4	4	4	4
ExcessCPI	0.6	0.525	0.45	0.375	0.3

Table 13: Using delay branch for Amdahl-V8

BC	$n=0$	$n=1$	$n=2$	$n=3$	$n=4$
BC-inline penalty	4	3	2	1	0
BC-target penalty	4	3	2	1	0
ExcessCPI	0.6	0.45	0.3	0.15	0







Problem 4.7

From section 4.4.5, we have the following equation:

$$p = 1 + \left\lceil \frac{1}{m} \times \frac{\text{IF access time (cycles)}}{\text{instructions/IF}} \right\rceil,$$

where p is the number of in-line buffer words, and an instruction is decoded every m cycles.

a. IBM 3033

$m = 1$

IF access time = 2 cycles (from Figure 4.3)

Average instruction length = $\frac{3.2B+3.8B}{2} = 3.5B$ (from Table 3.2)

Instructions/IF = $\frac{\text{path width } w}{\text{instruction length}} = \frac{4}{3.5} = 1.14$ for $w = 4$

Instructions/IF = $\frac{8}{3.5} = 2.29$ for $w = 8$

$$p(w = 4B) = 1 + \left\lceil \frac{1}{1} \times \frac{2}{1.14} \right\rceil = 3$$

$$p(w = 8B) = 1 + \left\lceil \frac{1}{1} \times \frac{2}{2.29} \right\rceil = 2$$

We assume that we have branch prediction which may sometimes predict the target path. Therefore both the primary and target paths must be the same (minimum) size in order to avoid runout.

b. Amdahl V-8

The Amdahl V-8 is similar to the 3033 (executing the same instruction set), but it only decodes an instruction every other cycle. From Figure 4.4, we have $m = 2$, IF access time = 2 cycles.

$$p(w = 4B) = 1 + \left\lceil \frac{1}{2} \times \frac{2}{1.14} \right\rceil = 2$$

$$p(w = 8B) = 1 + \left\lceil \frac{1}{2} \times \frac{2}{2.29} \right\rceil = 2$$

Once again, we assume branch prediction, so both instruction buffers must be the same size.

c. MIPS R2000

$m = 2$ (Figure 4.5)

IF access time = 2 cycles (Figure 4.5)

Average instruction length = 4B (from Table 3.2)

$$p(w = 4B) = 1 + \left\lceil \frac{1}{2} \times \frac{2}{1} \right\rceil = 2$$
$$p(w = 8B) = 1 + \left\lceil \frac{1}{2} \times \frac{2}{2} \right\rceil = 2$$

Assuming we can sometimes predict taken, both buffers must be the same size. However, if we assume that the R2000 always predicted the in-line path (relying perhaps on delayed branches to reduce branch latency), then the target buffer could perhaps be as small as one entry (as per section 4.4.5.)

Problem 4.8

We use Chebyshev's inequality.

a. Without knowing the variance, we have to use Bound 1:

$$\begin{aligned} \text{Prob (overflow)} &= \text{Prob}\, q > BF = \text{Prob}\, q \geq BF + 1 \rightarrow p \geq \frac{Q}{BF+1} \\ Q &= \text{mean number of requests} = 2 \\ BF &= \text{buffer size} = 4 \\ p &\geq \frac{Q}{BF+1}, \text{ so } p \geq 2/5,\ p \geq 40\% \end{aligned}$$

b. Knowing the variance, we can compare Bound 2 and take the minimum:

s^2 = variance = 0.5

$p \geq \frac{s^2}{(BF+1-Q)^2}$

$p \geq .5/9 = 5.55\%$

This upper bound is lower than the value determined in the first part, so we can say that $\text{Prob}\, q > BF \geq 5.55\%$.