Name:	Rainer Angelo Lim
Lab Section:	B03
Course:	ENCM 369
Lab Assignment:	9

Exercise A

For safe operation,

$$T_C \geq t_{pcq} + t_{pd} + t_{setup}$$

Part I

1. Two paths: (1) through Instruction Memory, (2) through Adder

   (1) $T_C \geq t_{pcq_{PC}} + t_{pd_{Instruction\ Memory}} + t_{setup_{Pipeline\ Register}}$
   $T_C \geq 19\ ps + 265\ ps + 28\ ps$
   $T_C \geq 312\ ps$

   (2) $T_C \geq t_{pcq_{PC}} + t_{pd_{Adder}} + t_{setup_{Pipeline\ Register}}$
   $T_C \geq 19\ ps + 90\ ps + 28\ ps$
   $T_C \geq 137\ ps$

   The shortest safe clock period is 312 ps.

2. $f_C = 3.333\ GHz$
   $T_C = \frac{1}{f_C} = \frac{1}{3.333\ GHz} = \frac{1}{3.333 \times 10^9\ Hz} = 3.00 \times 10^{-10}\ s = 300\ ps$
   $T_C = t_{pcq_{PC}} + t_{pd_{Instruction\ Memory}} + t_{setup_{Pipeline\ Register}}$
   $300\ ps = 19\ ps + t_{pd_{Instruction\ Memory}} + 28\ ps$
   $t_{pd_{Instruction\ Memory}} = 253\ ps$

Part II

1. Six paths: (1) through ALU, (2) through Multiplexer + ALU, (3) through Wire, (4) through Multiplexer, (5) through Shift Left 2 + Adder, (6) through Adder

   (1) $t_{pd} = t_{pd_{ALU}}$
   $t_{pd} = 160\ ps$

   (2) $t_{pd} = t_{pd_{Multiplexer}} + t_{pd_{ALU}}$
   $t_{pd} = 36\ ps + 160\ ps$
   $t_{pd} = 196\ ps$

   (3) $t_{pd} = t_{pd_{Wire}}$
   $t_{pd} = 0\ ps$

   (4) $t_{pd} = t_{pd_{Multiplexer}}$
   $t_{pd} = 36\ ps$

   (5) $t_{pd} = t_{pd_{Shift\ Left\ 2}} + t_{pd_{Adder}}$
   $t_{pd} = 0\ ps + 90\ ps$
   $t_{pd} = 90\ ps$

   (6) $t_{pd} = t_{pd_{Adder}}$
   $t_{pd} = 90\ ps$

   $t_{pd_{Overall}} = 196\ ps$

2. $T_C \geq t_{pcq} + t_{pd} + t_{setup}$
   $T_C \geq t_{pcq_{Pipeline\ Register}} + t_{pd_{Overall}} + t_{setup_{Pipeline\ Register}}$
   $T_C \geq 19\ ps + 196 + 28\ ps$
   $T_C \geq 243\ ps$

   It is not necessary to modify any part of the Execute stage to allow a clock frequency of 3.333 GHz (300 ps) because the shortest safe clock period is 243 ps. Since 300 ps is greater than 243 ps, the contents from the D/E register will reach the E/M register safely.

Part III

1. Three paths: (1) from E/M Register, through Multiplexer, to PC Register, (2) from E/M Register, through AND gate, through Multiplexer, to PC register, (3) from PC Register, through Adder, through Multiplexer, to PC Register

   (1) $t_{pd} = t_{pd_{Multiplexer}}$
   $t_{pd} = 36\ ps$

   (2) $t_{pd} = t_{pd_{AND}} + t_{pd_{Multiplexer}}$
   $t_{pd} = 15\ ps + 36\ ps$
   $t_{pd} = 51\ ps$

   (3) $t_{pd} = t_{pd_{Adder}} + t_{pd_{Multiplexer}}$
   $t_{pd} = 90\ ps + 36\ ps$
   $t_{pd} = 126\ ps$

   The longest possible delay from a positive clock edge to the point in time when the input to the PC is stable is:

   $$t_{Longest\ Delay} = t_{pcq_{PC}} + t_{pd_{(3)}}$$

   $$t_{Longest\ Delay} = 19\ ps + 126\ ps$$

   $$t_{Longest\ Delay} = 145\ ps$$

   The minimum clock period for the computer must be greater than or equal to the longest delay plus the setup time of the PC register:

   $$T_C \geq t_{Longest\ Delay} + t_{setup_{PC}}$$

   $$T_C \geq 145\ ps + 28\ ps$$

   $$T_C \geq 173\ ps$$

2. Two paths: (1) through Wire, (2) through Data Memory

   (1) $T_C \geq t_{pcq_{Pipeline\ Register}} + t_{pd_{Wire}} + t_{setup_{Pipeline\ Register}}$
   $T_C \geq 19\ ps + 0\ ps + 28\ ps$
   $T_C \geq 47\ ps$

   (2) $T_C \geq t_{pcq_{Pipeline\ Register}} + t_{pd_{Data\ Memory}} + t_{setup_{Pipeline\ Register}}$
   $T_C \geq 19\ ps + 275\ ps + 28\ ps$
   $T_C \geq 322\ ps$

   The minimum clock period needed to allow safe operation of the Memory stage is 322 ps.

3. $f_C = 3.333\ GHz$
   $T_C = \frac{1}{f_C} = \frac{1}{3.333\ GHz} = \frac{1}{3.333 \times 10^9\ Hz} = 3.00 \times 10^{-10}\ s = 300\ ps$
   $T_C = t_{pcq_{Pipeline\ Register}} + t_{pd_{Data\ Memory}} + t_{setup_{Pipeline\ Register}}$
   $300\ ps = 19\ ps + t_{pd_{Instruction\ Memory}} + 28\ ps$
   $t_{pd_{Instruction\ Memo}} = 253\ ps$

Part IV

1. $0.5T_C \geq t_{pcq_{Pipeline\ Register}} + t_{pd_{Multiplexer}} + t_{setup_{Register\ File}}$
   $0.5T_C \geq 30\ ps + 35\ ps + 31\ ps$
   $0.5T_C \geq 96\ ps$
   $T_C \geq 192\ ps$
2. $0.5T_C \geq t_{Until\ RD1\ and\ RD2\ are\ Ready} + t_{setup_{Pipeline\ Register}}$
   $0.5T_C \geq 117\ ps + 28\ ps$
   $0.5T_C \geq 145\ ps$
   $T_C \geq 290\ ps$
3. A 3.333 GHz clock has a period of 300 ps. Since 192 ps and 290 ps are both less than this, there is nothing about the designs of the Writeback and Decode stages that would prevent the use of such a clock.

Part V

1. Part I: $T_C \geq 312$ ps
   Part II: $T_C \geq 243$ ps
   Part III: $T_C \geq 322$ ps
   Part IV: $T_C \geq 290$ ps

   The minimum clock period to allow reliable operation of all five pipeline stages is 243 ps.
2. If 3.333 GHz (300 ps) is the desired clock frequency, the stage studied in Part III must be modified its clock constraint, 322 ps, is greater than 300 ps. Using the equation from Part III, the parts that need modification can be determined:
   $T_C \geq t_{Longest\ Delay} + t_{setup_{PC}}$
   $T_C \geq t_{pcq_{PC}} + t_{pd_{(3)}} + t_{setup_{PC}}$
   $T_C \geq t_{pcq_{PC}} + t_{pd_{Adder}} + t_{pd_{Multiplexer}} + t_{setup_{PC}}$
   $300\ ps \geq 19\ ps + t_{pd_{Adder}} + t_{pd_{Multiplexer}} + 28\ ps$
   $t_{pd_{Adder}} + t_{pd_{Multiplexer}} \leq 253\ ps$

   Therefore, the Adder and the Multiplexer must be modified to work faster, and the overall t_{pd} of the two components must be less than or equal to 253 ps.

Exercise B

1. At t = 45.5 ns, the address of the and instruction gets written into the PC: 0x0040_00a8.
   Shortly after t = 45.5 ns,
   InstrD is the beq instruction: 0x1240_fffa,
   and PCPlus4D is the address of the and instruction: 0x0040_00a8.

	45.0	45.5
beq	F	D
sub		F

2. At t = 46.0 ns, the address of the slt instruction gets written into the PC: 0x0040_00ac.
   Shortly after t = 46.0 ns,
   InstrD is the sub instruction: 0x0319_4022,
   and PCPlus4E is the address of the and instruction: 0x0040_00a8.

	45.0	45.5	46.0
beq	F	D	E
sub		F	D
slt			F

3. At t = 46.5 ns, the address of the and instruction gets written into the PC: 0x0040_00b0.
   Shortly after t = 46.5 ns,
   InstrD is the slt instruction: 0x0319_482a,
   PCBranchM is the sum of SignImmE << 2 and PCPlus4E:
   = (0xffff_fffa × 4) + 0x0040_00a8
   = 0xffff_ffe8 + 0x0040_00a8
   = 0x0040_0090,
   and ZeroM is 1 since the difference between the contents of $18 and $0 is 0.

	45.0	45.5	46.0	46.5
beq	F	D	E	M
sub		F	D	E
slt			F	D
and				F

4. At t = 47.0 ns, the address of the or instruction gets written into the PC: 0x0040_00b4.
   Shortly after t = 47.0 ns,
   InstrD is the and instruction: 0x0319_5024.

	45.0	45.5	46.0	46.5	47.0
beq	F	D	E	M	W
sub		F	D	E	M
slt			F	D	E
and				F	D
or					F

5. At t = 47.5 ns, the address of the instruction after or gets written into the PC: 0x0040_00b8.
   Shortly after t = 47.5 ns,
   InstrD is the or instruction: 0x0319_5825.

	45.0	45.5	46.0	46.5	47.0	47.5
and					F	D	E
or						F	D
?							F

Exercise C

Part I

(1) Use of first add result as a source in the addi instruction of line 6.

During the Execute stage of addi, the Hazard Unit detects that RsE (11001_{two} for \$25) matches WriteRegW (also 11001_{two} for \$25) and that RegWriteW = 1. So it sets ForwardAE = 01 so that ResultW (the first add result) is passed to the "A" input of the ALU.

(2) Use of first sub result as a source in the lw instruction of line 7.

During the Execute stage of lw, the Hazard Unit detects that RsE (11000_{two} for \$24) matches WriteRegW (also 11000_{two} for \$24) and that RegWriteW = 1. So it sets ForwardAE = 01 so that ResultW (the first sub result) is passed to the "A" input of the ALU.

(3) Use of first lw result as a source in the sw instruction of line 9.

During the Execute stage of sw, the Hazard Unit detects that RtE (01010_{two} for \$10) matches WriteRegW (also 01010two for \$10) and that RegWriteW = 1. So it sets ForwardBE = 01 so that ResultW (the first lw result) is passed onto the E/M pipeline register to be written.

sw	F	D	E	M	W									1
sw		F	D	E	M	W								2
sw			F	D	E	M	W							3
add				F	D	E	M	W						4
sub					F	D	E	M	W					5
addi						F	D	E	M	W				6
lw							F	D	E	M	W			7
addi								F	D	E	M	W		8
sw									F	D	E	M	W	9

Part II

During the Execute stage of sw, the Hazard Unit detects that RtE (01010_{two} for \$10) matches WriteRegM (also 01010_{two} for \$10) and that RegWriteM = 1. So it sets ForwardBE = 10, which results in sw storing the lw instruction's address instead of the contents of \$10.

Therefore, the instruction sequence fails in a different way, storing some other wrong value.

lw	F	D	E	M	W	
sw		F	D	E	M	W

Exercise D

```
Variables          Registers
int sum            $t0
int *p             $t1         <- from $a0
int *past_last     $t2         <- from $a1

L1:   lw    $t1, ($a0)        # $t1 = *p
      addiu $a0, $a0, 4       # p++
      slt   $t3, $zero, $t1   # $t3 = (*p > 0)
      movn  $t3, $t1, $t3     # if ($t3 != 0) $t3 = $t1
      bne   $a0, $t2, L1      # if (p != past_last) goto L1
      addu  $t0, $t0, $t3     # sum += $t3
```