Exercise 4.1

Different instructions utilize different hardware blocks in the basic single-cycle implementation. The next three problems in this exercise refer to the following instruction:

	Instruction	Interpretation
a.	`AND Rd,Rs,Rt`	`Reg[Rd]=Reg[Rs] AND Reg[Rt]`
b.	`SW Rt,Offs(Rs)`	`Mem[Reg[Rs]+Offs]=Reg[Rt]`

- What are the values of control signals generated by the control in the data path diagram given in lecture for these two instructions?
- What are the hardware functional units utilized to perform the above two instructions?

Solution:
Control Signals

Operation	RegDst	RegWrite	ALUSrc	ALUOp	MemWrite	MemRead	MemReg	PCSrc
and	1	1	0	000	0	0	0	0
sw	0 or don't care X	0	1	010	1	0	0 or don't care X	0

Hardware Functional Units
`and`: I-Mem, Reg-File, ALU
sw: I-Mem, Reg-File, ALU,D-Mem

Different execution units and blocks of digital logic have different latencies (time needed to do their work). In Figure 4.2 there are seven kinds of major blocks. Latencies of blocks along the critical (longest-latency) path for an instruction determine the minimum latency of that instruction. For the remaining three problems in this exercise, assume the following resource latencies:

	I-Mem	Add	Mux	ALU	Regs	D-Mem	Control
a.	200ps	70ps	20ps	90ps	90ps	250ps	40ps
b.	750ps	200ps	50ps	250ps	300ps	500ps	300ps

4.1.4 [5] <4.1> What is the critical path for an MIPS AND instruction?

4.1.5 [5] <4.1> What is the critical path for an MIPS load (LD) instruction?

4.1.6 [10] <4.1> What is the critical path for an MIPS BEQ instruction?

Solution:

Part a. Ignore MUXs and Control delay times:

Delay time for **`and`**: Reading I-Mem then Reading Reg then computing in ALU then Writing on Reg
Delay time = 200+90+90+90 = 470 ps

Delay time for **`lw`**: Reading I-Mem then Reading Reg then computing in ALU then Reading D-Mem then Writing on Reg
Delay time = 200+90+90+250+90 = 720 ps

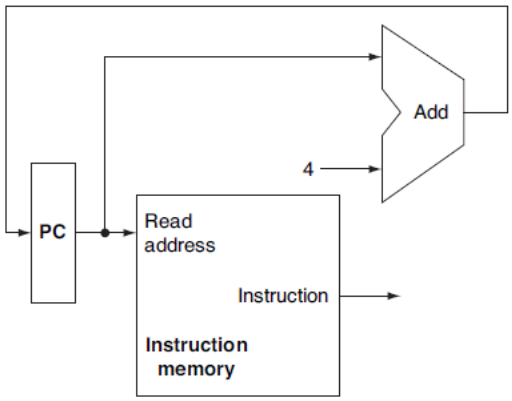
Delay time for **`beq`**: Reading I-Mem then Reading Reg then computing in ALU (No D-Mem & No WB to Reg)
Delay time = 200+90+90= 380 ps

- Which kind of instructions require the unit block 'Shift-left-2'?

Shift left 2 is needed to multiply offset by 4
This implies all branch instructions that contain PC+4+offset*4 calculation.

- Which kind of instructions require the unit block 'Regs': for register file?

All instructions **except** those that use **only labels** example b label, j label



Problems in this exercise assume that logic blocks needed to implement a processor's datapath have the following latencies:

	I-Mem	Add	Mux	ALU	Regs	D-Mem	Sign-Extend	Shift-Left-2
a.	200ps	70ps	20ps	90ps	90ps	250ps	15ps	10ps
b.	750ps	200ps	50ps	250ps	300ps	500ps	100ps	0ps

- If the only thing we need to do in a processor is fetch consecutive instructions (Check figure above), what would the cycle time be?

Solution:

Fetch instruction occurs simultaneously with PC increment to prepare for executing next instruction.
This gives a cycle time of 200 ps (I-Mem delay only)

- Consider a datapath similar to the one given in lecture slides, but for a processor that only has one type of instruction: unconditional PC-relative branch. What would the cycle time be for this datapath?

Unconditional means no comparison, no need for ALU.
Sign extender to make the 16-bit immediate value 32 bits

Cycle time = I-Mem+sign Ext +shiftleft = 200+15+10 = 225 ps (Ignore MUXs and adder delay time).

- Repeat the above exercise but this time we need to support only *conditional* PC-relative branches.

Same as unconditional but this time we need the Reg and ALU to compare registers
Cycle time = I-Mem + sign Ext+ shiftleft + Reg +ALU = 200+15+10+90+90 = 405 ps (Ignore MUXs delay time).

In this exercise we examine how latencies of individual components of the datapath affect the clock cycle time of the entire datapath, and how these components are utilized by instructions. For problems in this exercise, assume the following latencies for logic blocks in the datapath:

	I-Mem	Add	Mux	ALU	Regs	D-Mem	Sign-Extend	Shift-Left-2
a.	200ps	70ps	20ps	90ps	90ps	250ps	15ps	10ps
b.	750ps	200ps	50ps	250ps	300ps	500ps	100ps	5ps

4.7.1 [10] <4.3> What is the clock cycle time if the only types of instructions we need to support are ALU instructions (ADD, AND, etc.)?

4.7.2 [10] <4.3> What is the clock cycle time if we only have to support LW instructions?

4.7.3 [20] <4.3> What is the clock cycle time if we must support ADD, BEQ, LW, and SW instructions?

Solution Part a:

Cycle time if we support R-type instructions:
Reading I-Mem then Reading Reg then computing in ALU then Writing on Reg
Delay time = 200+90+90+90 = 470 ps

Cycle time if we support LW instructions:
Delay time for **`lw`**: Reading I-Mem then Reading Reg then computing in ALU then Reading D-Mem then Writing on Reg
Delay time = 200+90+90+250+90 = 720 ps

Cycle time if we support ADD, BEQ, LW, & SW instructions:
Time for slowest instruction will be the cycle time i.e. time for lw which is 720 ps

Exercise 4.12

In this exercise, we examine how pipelining affects the clock cycle time of the processor. Problems in this exercise assume that individual stages of the datapath have the following latencies:

	IF	ID	EX	MEM	WB
a.	250ps	350ps	150ps	300ps	200ps
b.	200ps	170ps	220ps	210ps	150ps

- What is the clock cycle time in a pipelined and non-pipelined processor?
- What is the total latency of an LW instruction in a pipelined and non-pipelined processor?

Solution Part a:

In a pipelined processor: Take the delay time of the slowest stage: ID 350 ps.

In a single cycle:
Full cycle times of all instructions:
ALU 950ps
BEQ 750ps
LW 1250ps
SW 1050ps
Take the cycle time of the longest instruction LW
250+350+150+300+200 = 1250 ps.

Total Latency of LW in non pipelined is 1250ps
Total latency of LW in a pipelined processor is cycle time of pipelined* number of stages of lw instruction= 350* 5 = 1750 ps.

The remaining problems in this exercise assume that instructions executed by the processor are broken down as follows:

	ALU	BEQ	LW	SW
a.	45%	20%	20%	15%
b.	55%	15%	15%	15%

- Assuming there are no stalls or hazards, what is the utilization of the data memory?

- Assuming there are no stalls or hazards, what is the utilization of the write-register port of the "Registers" unit?
- Instead of a single-cycle organization, we can use a multicycle organization where each instruction takes multiple cycles but one instruction finishes before another is fetched. In this organization, an instruction only goes through stages it actually needs (e.g., ST only takes 4 cycles because it does not need the WB stage).
Compare clock cycle times and execution times with single-cycle, multi-cycle, and pipelined organization.

Solution:

The utilization of the data memory is 20%+15% = 35%
The utilization of the write register ports is 45% + 20% = 65%

Multicycle:
Average CPI = 950ps * 45% + 750ps *20%+1250ps * 20% +1050ps*15% = 985 ps.

We have the speedup of pipeline: 1250/350 = 3.571

Exercise

In this exercise, we examine how data dependences affect execution in the basic 5-stage pipeline described in lecture slides. Problems in this exercise refer to the following sequence of instructions:

	Instruction Sequence

a.	SW R16,-100(R6)
	LW R4,8(R16)
	ADD R5,R4,R4

b.	OR R1,R2,R3 OR R2,R1,R4 OR R1,R1,R2

Indicate dependencies. Assume that the second half of the decode stage performs a read of source registers, and that the first half of the write back stage writes to the register file

Complete the pipeline diagram below (instructions on the left, cycles on top). Insert the characters IF, ID, EX, MEM, WB for each instruction in the boxes. Label wait cycles (Draw an X in the box).

Solution For part a:

The add instruction uses the value of R4 which should be written back to register file in the first half of the WB stage of the second instruction. And the value of R4 is fetched in the second half of the ID of the third instruction. So we need two NOP instructions or we need two extra wait cycles before performing ID of Add.

	T1	T2	T3	T4	T5	T6		T7	T8	T9	T10	T11	T12	T13	T14	T15
I0	IF	ID	EX	MEM	NOP											
I1		IF	ID	EX	MEM	WB										
I2			IF	X	X		ID	EX	NOP	WB						