Exercise for Midterm

1. What are the five classic components of computer organization?

2. Consider three different processors P1, P2, and P3 executing the same instruction set with the clock rates and CPIs given in the following table.

	Processor	Clock Rate	CPI
a.	P1	3 GHz	1.5
	P2	2.5 GHz	1.0
	P3	4 GHz	2.2
b.	P1	2 GHz	1.2
	P2	3 GHz	0.8
	P3	4 GHz	2.0

Figure 1(1.3.1-1.3.3)

(1) Which processor has the highest performance expressed instructions per second?

(2) If the processors each execute a program in 10 seconds, find the number of cycles and the number of instructions.

(3) We are trying to reduce the time(10s) by 30% but this leads to an increase of 20% in the CPI. What clock rate should we have to get this time reduction?

3. Suppose a program runs in 120 seconds on a computer, with multiply operations responsible for 80 seconds of this time. We want the program to run 3 times faster. If possible, how much do we have to improve the speed of multiplication? Or explain if it is impossible.

4. True / False

(1) We usually use different compilers for various application programs to optimize performance because compiler determines instruction count, CPI and CPU's clock rate.

(2) Floating point addition is NOT associative.

(3) Increasing the clock rate always results in better CPU performance.

(4) Increasing the clock rate always results in better CPI.

(5) Jump instruction can jump to any legal memory address as it is an unconditional branch.

(6) CPU performance can be improved by reducing both instruction count and CPI.

(7) ISA solely influences instruction count and clock rate.

(8) The organization of the computer will have effect on the instruction count.

(9) The clock rate is the primary factor determining CPU performance.

(10) CPI can be improved by optimizing the CPU's microarchitecture.

5. In this task, we are required to illustrate the principle of building a 1-bit binary full adder. But it can be finished by dividing it into the following subtasks.

(1) Given a half adder first, which executes the addition of two binary numbers. Determine the logic equation of the two outputs – sum and carry. Please give the reason of these two equations.

(2) Why can we use two half adders to build a full adder(use the logic equation to explain), allocating them like the following figure:

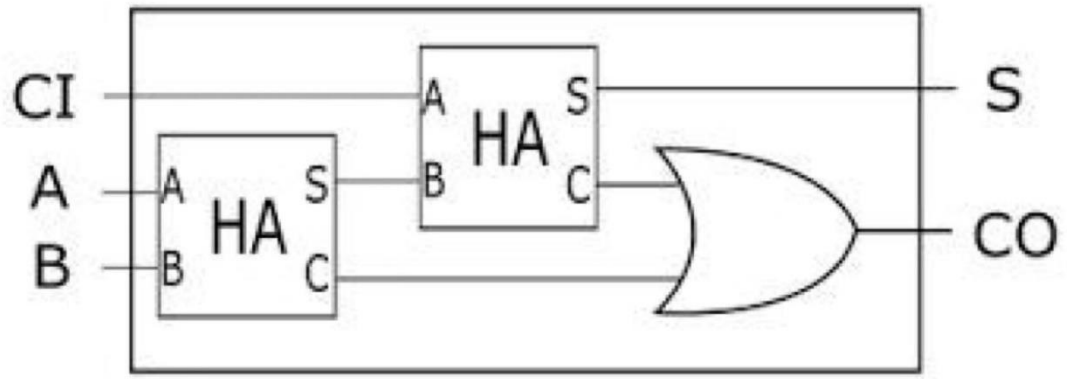


Figure 2(CI represents carry in, the same as CO)

6. Given a 4-bit multiplicand register, a 4-bit ALU, and a 8-bit product register(Multiplier register is encapsulated in product register), indicate the state of each repetition using Booth algorithm executing multiplication.

(Multiplier = -6= 1010, Multiplicand = 0111)

7. Give the advantages and disadvantages between RISC and CISC by stating them in terms of these two instruction sets respectively.(Ease of compilation vs hardware complexity should be included in your answer.)

8. The following problems deal with translating from C to MIPS. Assume that the variables f, g, h, i, and j are assigned to registers $s0, $s1, $s2, $s3 and $s4, respectively. Suppose that the base address of arrays A and B are in registers $s6 and $s7, respectively. For the C statements below, what are the corresponding MIPS assembly codes?

(1) g = I & 0x20171025;
    h = f – g + A[4];

(2) for(i = 0; i < j; i++)
        B[8] = A[i] + A[j+1];

(3) i = 0;
    do{
        B[8] = A[i] + A[i + 1];
    }while(i < 10)

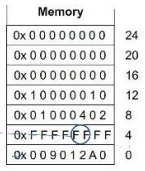
(4) j = 0;
  while(j < 10){
     B[8] = B[j] + B[j + 1];
     j = j + 1;
  }

9. Convert the following MIPS instructions to the corresponding machine codes. Assume the instructions are stored in memory with the address listed left.

start: beq $s0, $a0, next	#memory address [0x00000000]
sub $v0, $a0, $s0	#memory address [0x00000004]
next: addi $s0, $s0, 1	#memory address [0x00000008]
j start	#memory address [0x0000000C]

10. Given following code sequence and memory state, what is the state of the memory after executing the code?

add $s3, $zero, $zero
lb $t0, 1($s3) # lb means ‘load byte’
sb $t0, 6($s3) # sb means ‘store byte’
(The memory address is expressed with big Endian)



(1) Determine the value of $t0 after executing the code.
(2) What is the change of the memory address?
(3) Repeat (1) and (2) for little Endian.

11. Write a procedure, *find_k*, in MIPS assembly language. The procedure should take a single argument that is a pointer to a null-terminated string in register $a0. The *find_k* procedure should locate the first 'k' character in the string and return its address in register $v0. If there is no 'k' in the string, then *find_k* should return a pointer to the null character at the end of the string. For example, if the argument to *find_k* points to the string "hijklmn", then the return value will be a pointer to the fourth character of the string.

12. Write a function '*dsum*' in MIPS assembly language to calculate sum of all the digit value in a string. A valid digit should be between 0 and 9. Your program should expect register $a0 to hold the address of a null-terminated string containing some combination of the digits 0 thorough 9 and sum up all the digits in the string and place the sum in register $v0, skipping all the non-digit character appeared anywhere in the string. If no valid digit in string, your program should return a **zero** in register $v0.
For example: if register $a0 points to a string "23the5i908", then when the procedure returns, $v0 should contain the value 27.
Remark: ASCII of "0" = 48; ASCII of "9" = 57; ASCII of "null" = 0.

13. Given a single-cycle processor, the MIPS instruction lw $rt, offset($rs) sets register $rt to the value at Mem[$rs + offset] where offset is a 16-bit immediate. When executed on the single-cycle datapath shown below, determine the value of control signals for

(1) lw rt, offset(rs);

(2) sw rt, offset(rs);

(3) add rd, rs, rt;

(4) sub rd, rs, rt;

(5) addi rt, rs, imm16;

(6) beq rs, rt, Label;

(Control signals: PCsrc, ALUsrc, ALUOp, MemWrite, MemToReg, RegDst, RegWrite, MemRead)

Extra: Consider a new R-type instruction lwd $rd, $rt($rs), which sets register $rd to the value at Mem[$rs+$rt]. Determine the value of control signals in the datapath.

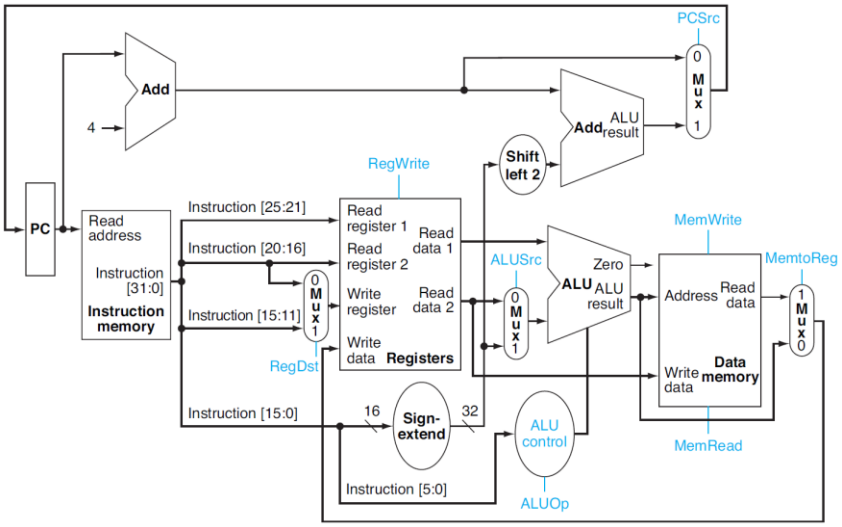


Figure 3

14. A single-cycle processor has a cycle time of 800ps and a multi-cycle processor has $CPI_{R\text{-}type} = 4$, $CPI_{lw} = 5$, $CPI_{beq} = 3$. Cycle time = 200ps. A program has 40% R-type, 20% lw and 40% beq. Totally 10000 instructions. Which one is better?