NAME: Steven Reeves

CST 131 HW#2

1. Provide the hex result of the proper logical, arithmetic shift of hex number 81

81_{16} = 1000 0001

Logical shift right =	0100 0000	Bit to the right falls off…
Logical shift left =	0000 0010	Bit to left falls off…
Arithmetic shift right =	1100 0000	Keep leading bit with arithmetic shift
Arithmetic shift left =	0000 0010	Bits just fall off left

2. Provide the hex result of the proper rotate of hex number 81

Rotate right =	1100 0000	Right most bit jumps to front
Rotate left =	0000 0011	Left most bit jumps to end

3. Convert the following expressions from infix to reverse Polish (postfix) notation.
a) (8 – 6) / 2
b) (2 + 3) * 8 / 10
c) (5×(4 + 3) × 2 – 6)

a) 8 6 – 2 /	8-6 first, then that / 2
b) 2 3 + 8 * 10 /	2+3 first, then *8, then / 10
c) 5 4 3 + * 2* 6 -	5* (4+3) first, then , then *2, then -6

Watch spaces! 5 * before the first move, OK.

4. Convert the following expressions from reverse Polish notation to infix notation.
a. 12 8 3 1 + – /
b. 5 2 + 2× 1 + 2×
c. 3 5 7 + 2 1 –× 1 + +

a) 12 / (8 - (3+1))	12 /, 8 -, (3+1) 8-… in ()
b) (((5+2) * 2) + 1) * 2	(5+2), +2, +1, *2
c) 3 + (((5+7) * (2 -1)) +1)	3 , 5+7, 2-1,*, +1, +,

Watch order of operations, and add parenthesis when needed. Note: operators always call the TWO operands before them.

5. a. Write the following expression in postfix (Reverse Polish) notation. Remember the rules of precedence for arithmetic operators!

$$X = \frac{A - B + C * (D * E - F)}{G + H * K}$$

X = Num / dem
X = Num dem /

Num = C D E * F - * A B - +
= D * E – F * C + (a – b)

Dem = G H K * +
= G + H * K

X = C D E * F - * A B - + G H K * + /

b. Write a program to evaluate the above arithmetic statement using a stack organized computer with zero-address instructions (so only pop and push can access memory).

Pseudo-code:

- Define pop and push
  - Push → Move stack pointer "up" one and store value there
  - Pop → Get value from location and move stack pointer "down" one.

- Store input in an array-like structure (block of addresses is okay)
- Store operators as unique values

In "main":

- Load operators for comparison

Loop_begin:
- Load value of first array index
- if operator, do that operation
- if not operator, push digit onto stack
- return to "Loop_begin"

"+" operation:
- pop digit_1
- pop digit_2
- add digit_2 to digit_1 ← note backwards for stack
- move array index ← move stack pointer
- return to "Loop_begin"

In MIPS Assembly, (without divide): see attached lab3.S file.

6. A nonpipelined system takes 200ns to process a task. The same task can be processed in a 5-segment pipeline with a clock cycle of 40ns. Determine the speedup ratio of the pipeline for 200 tasks. What is the maximum speedup that could be achieved with the pipeline unit over the nonpipelined unit?

Speedup $S = k * t_p / t_p = 40*5 / 40 = 5$

Where k is number of segments and t_p is time per segment

7. In a certain computer architecture, the content of some registers and memory content currently is as follows.

		Memory Address	Data
R1	FA001234		
		0000FFFE	00349876
R2	0000FFFE	F000128A	0000FFFE

Next, this computer execute the following instruction.

MOVEI R1, R2

Immediately afterward, the contents are as follows.

		Memory Address	Data
R1	00349876		
		0000FFFE	00349876
R2	0000FFFE	F000128A	0000FFFE

a) What is the destination operand?

R1 ← This is the register that changed

b) What is the source operand?

R2 ← This is the register with the *address* to the contents

c) What type of addressing mode is used?

Indirect register addressing ← Address in register was used to find "value"

NOTE: Not in homework but be expected to be tested in quizzes and tests on ALL possible addressing modes (Immediate, Direct, Indirect, Indexed) for 1 operand, 2 operand, 3 operand machine

Immediate → The number X itself

Direct → The contents at X

Indirect → The contents at the address at X

Indexed → The contents of the offset address