Module 2 (Lectures 3-10) Computer organization

1. Write a C program that can be used to identify whether the machine it is running on is using the LittleEndian or BigEndian byte ordering convention.
2. Write down the value of the operand specified in each of the following cases given the following values in registers and memory locations. All values are shown in decimal as signed 32bit integers.

Registers	Memory address: Contents
R2: 76	68: -5
R3: 80	72: 78
R4: 88	76: 94
R5: 96	80: 100

   72 in absolute addressing mode
   (R3) in register indirect addressing mode
   R4 in register addressing mode
   68 in immediate addressing mode
3. Write MIPS I assembly language code for stack push and pop operations assuming the following: (a) all stack elements are of size 1 halfword, (b) stack grows from low memory to high memory addresses, (c) register R29 is used as the stack pointer and always contains the memory address just after the element which is on top of the stack, (d) for a push operation, register R1 holds the data to be pushed, (e) for a pop operation, the element from top of stack is to be popped into register R1.
4. Show how a rotate left operation, **R1 ← R2 rotate left by 7 bits,** can be implemented using the MIPS 1 instruction set discussed in the lectures. Note that we require the value present in general purpose register R2 to be rotated left by 7 bits, with source register R2 not being modified, but the rotated value to be available in the destination register R1.
5. Show a sequence of MIPS 1 instructions that can be used to implement the multiplication operation **R1 ← R2 * R3**, where we require the destination to contain the value of the product if it fits in 32bits, or 0 if the product does not fit in 32 bits.