Problem EX-2 (5 parts) **MIPS Assembly Expressions**

Part A: Write a MIPS program fragment that computes " $-17 \cdot (B-C)$ "and puts the result in register $6. Assume B and C are in registers $1 and $2, respectively. Use a minimum number of instructions and registers. You may reuse registers $1 and $2.

```
sub   $1, $1, $2        # B = B – C
addi  $6, $0, -17       # $6 = -17
mult  $1, $6            # Lo = -17 * (B – C)
mflo  $6                # $6 = result
```

Part B: Suppose A is stored in memory location 1020 and B is stored in memory location 1024. Write a MIPS program fragment that computes " $256 \cdot (A+B/16)$ " and stores the result at memory location 1028. Use a minimum number of instructions and registers.

```
lw    $1, 1020($0)      # $1 = mem[1020]
lw    $2, 1024($0)      # $2 = mem[1024]
sra   $2, $2, 4         # $2 = $2 / 16
add   $2, $2, $1        # $2 = A+B/16
sll   $2, $2, 8         # $2 = 256*(A+B/16)
sw    $2, 1028($0)      # mem[1028] = $2
```

Part C: Write a MIPS program fragment to put the value 0xABCD1234 into register $1.

```
lui   $1, 0xABCD        # load upper word
ori   $1, $1, 0x1234    # combine with lower word
```

Part D: Suppose an image processing system stores a 512x256 pixel image in memory. Each pixel is represented by 8 bits and they are store contiguously in memory. How much memory (in kilobytes) does this require? How many bits are needed to address 1 pixel?

512 x 256 pixels x 1 byte/pixel = 2^9 x 2^8 = 2^{17} = 128 Kbytes
To address 128 Kbytes, an address would require at least 17 bits.

Part E: Write a MIPS fragment that exchanges two registers ($1 and $2) without using any other registers or memory. (hint: think xor).

```
xor   $1, $1, $2        # $1 = $1 xor $2
xor   $2, $1, $2        # $2 = $1 xor $2 xor $2 = $1
xor   $1, $1, $2        # $1 = $1 xor $2 xor $1 = $2
```