MIPS CPU

In this MIPS CPU implementation, Memory devices are simply treated as arrays of registers for simplicity in the interfacing. The memory depth is no more than 32 words while the width is 32 bits. Using the MIPS reference sheet which can be found online as a reference for the opcode and function numbers.

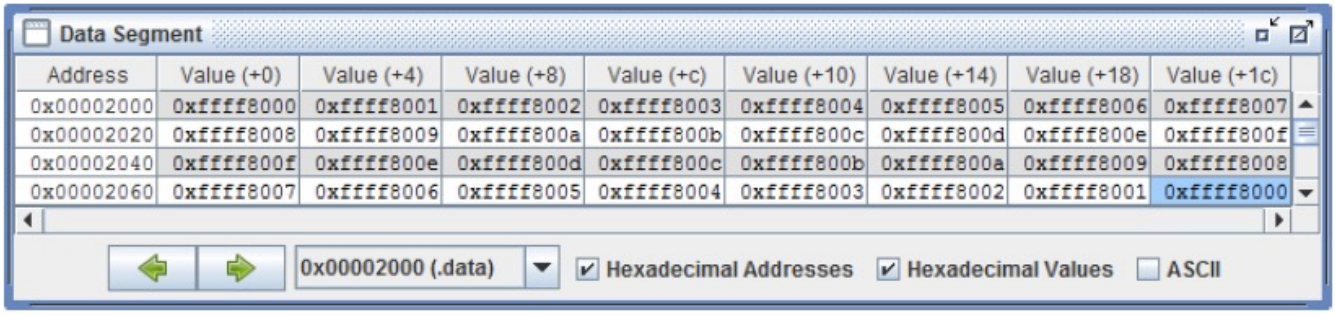
Testing code:

```
start:  addiu   $t0,$0,0        # 0
        addiu   $t1,$0,1        # 1
        addiu   $t2,$0,0        # 2
        addiu   $t3,$0,4        # 3
        addiu   $t4,$0,0x2000   # 4
loop1:  sw      $t2,0($t4)      # 5
        addu    $t2,$t2,$t1     # 6
        addiu   $t4,$t4,4       # 7
        sltiu   $at,$t2,16      # 8
        bne     $at,$0,loop1    # 9
        addiu   $t4,$t4,8       # 10
loop2:  subu    $t2,$t2,$t1     # 11
        sw      $t2,-8($t4)     # 12
        addu    $t4,$t4,$t3     # 13
        beq     $t2,$0,loop3    # 14
        j       loop2           # 15
loop3:  addiu   $t4,$0,0x1ff8   # 16
        addiu   $t3,$0,32       # 17
loop4:  lw      $t5,8($t4)      # 18
        addiu   $t5,$t5,-32768  # 19
        sw      $t5,8($t4)      # 20
        addu    $t2,$t2,$t1     # 21
        addiu   $t4,$t4,4       # 22
        sltu    $at,$t2,$t3     # 23
        bne     $at,$0,loop4    # 24
        addiu   $v0,$v0,10      # 25
        syscall                 # 26
```

The Testing Code shown below will be used for this implementation. Therefore, only the instructions found in this test code are required to be implemented. Other instructions are not being considered.

The first loop fills memory from 0x2000 to 0x202f with values from 0x0 to 0xf (increasing). The second loop fills memory from 0x2040 to 0x206f with values from 0xf to 0x0 (decreasing).

The last loop takes the values from 0x2000 to 0x206f and adds “-32768” to them to produce a large negative number. The sign extension should work such that the upper 16 bits of the results are all 1’s. The expected output is shown below:



In all three loops, memory is accessed using different byte offsets. Since this is MIPS implementation is on a 32-bit system, each word contains 4 bytes and thus memory accesses by word require a 4-byte offset. In this VHDL MIPS implementation, the RAM will hold data from 0x0 to 0x1f, not 0x2000 to 0x206f. Since memory will be accessed by words which will not require any offset in order to grab the next word in memory.