CEN 214 Microprocessors

Lab Assignment 3

New Instruction Set Commands:

CMP [destination operand], [source operand]

Description:

Compares the first source operand with the second source operand and sets the status flags in the EFLAGS register according to the results. The comparison is performed by subtracting the second operand from the first operand and then setting the status flags in the same manner as the SUB instruction. When an immediate value is used as an operand, it is sign-extended to the length of the first operand.

Operation

temp ← SRC1 − SignExtend(SRC2);
ModifyStatusFlags; (* Modify status flags in the same manner as the SUB instruction*)

Flags Affected

The CF, OF, SF, ZF, AF, and PF flags are set according to the result.[1]

JMP [label]

Description

Transfers program control to a different point in the instruction stream without recording return information. The destination (target) operand specifies the address of the instruction being jumped to. This operand can be an immediate value, a general-purpose register, or a memory location. [2]

Jcc [label]

Description

Checks the state of one or more of the status flags in the EFLAGS register (CF, OF, PF, SF, and ZF) and, if the flags are in the specified state (condition), performs a jump to the target instruction specified by the destination operand. A condition code (*cc*) is associated with each instruction to indicate the condition being tested for. If the condition is not satisfied, the jump is not performed and execution continues with the instruction following the J *cc* instruction.[3] JA, JAE, JB, JBE, JC, JCXZ, JE, JG, JGE, JL, JLE, JNA, JNAE, JNB, JNBE, JNC, JNE, JNG, JNGE, JNL, JNLE, JNO, JNP, JNS, JNZ, JO, JP, JPE, JPO, JS and JZ can be used as conditional jump instruction.

---

1 IA-32 Intel® Architecture Software Developer's Manual Volume 2: Instruction Set Reference, 2003, (CMP Command, pp. 3-85)

2 ...(JMP Command, pp. 3-366)

3 ...(Jcc Command, pp. 3-363)

Examples

1. Write a program that analysis **0100:1000h** memory address. If value of **0100:1000h** is equals to **C2h** write **FFh** value to **0100:1100h** memory address. If the value is not equals to **C2h** then write **AAh** to **0100:1100h** memory address.

2. There is an array of bytes written on memory beginning from **0100:1000h** with an ending byte value of **00h.** Copy this array to memory beginning from **0100:2000h** memory address. Use the array predefined below:

```
MOV byte ptr [1000h], 12h
MOV byte ptr [1001h], 34h
MOV byte ptr [1002h], 56h
MOV byte ptr [1003h], 78h
MOV byte ptr [1004h], 9Ah
MOV byte ptr [1005h], 0BCh
MOV byte ptr [1006h], 0DEh
MOV byte ptr [1007h], 0F0h
MOV byte ptr [1008h], 00h
```

3. Write to memory addresses between **0100:1500h** and **0100:15FFh,** hex values that begin from **00h** and increase **1** to **FFh** value if corresponding memory byte's value is equals to **00h.**