**CSCI 360 Logical Instructions**

The following instructions are going to perform **logical operations** between two fullword operands, a byte in storage and an immediate byte, or between two fields in storage.

A logical operation works on a bitwise level. Starting on the left, a bit from each operand has the logical operation performed and results in a boolean (true/false) value. This process proceeds until all of the bits have had the logical operation performed.

The possible logical operations are AND, OR, and EXCLUSIVE OR. Since these instructions work on a bit-wise level, a value of 0 is considered false and a value of 1 is considered true.

**AND operation**

**AND Truth Table**

**Oper 1 Oper 2**

**Bit Bit Result**

0 0 0 When an AND operation is performed,

0 1 0 the result is 1 ONLY if both bits

1 0 0 are 1; otherwise the result is 0.

1 1 1

The AND operation is used to set bits to 0

AND a bit with 1, it stays the same. AND a bit with 0, it becomes 0

RX Format: label N R,D(X,B)

The fullword in the register and the fullword at D(X,B) are ANDed together. The result of the operation is placed in the register.

RR Format: label NR R1,R2

The fullword in the R1 and the fullword in R2 are ANDed together. The result of the operation is placed in R1.

SI Format: label NI D(B),byte

The byte located at D(B) and the immediate byte specified by byte are ANDed together. The result of the operation is placed at D(B).

SS Format: label NC D1(L,B1),D2(B2)

The L byte storage areas at D1(B1) and D2(B2) are ANDed together. The result of the operation is placed starting at D1(B1).

For all formats, if every bit of the result is 0, the condition code is set to 0; otherwise the condition code is set to 1.

**AND Examples:**

Suppose that a storage area called BYTE contains X'C5'. In binary:

BYTE 1 1 0 0 0 1 0 1

bit number 0 1 2 3 4 5 6 7

Execution of the instruction: NI BYTE,B'11111010'

BYTE 1 1 0 0 0 1 0 1

1 1 1 1 1 0 1 0

RESULT 1 1 0 0 0 0 0 0

will change or "turn off" bits 5 and 7 in BYTE. BYTE now

contains X'C0'.

Suppose that register 5 contains 00 00 0A 07. Then execution of

the instruction:

N R5,=X'00000000'

Register 5 0000 0000 0000 0000 0000 1010 0000 0111

X'00000000' 0000 0000 0000 0000 0000 0000 0000 0000

0000 0000 0000 0000 0000 0000 0000 0000

will zero out register 5.

While execution of N R5,=X'FFFFFFFF'

Register 5 0000 0000 0000 0000 0000 1010 0000 0111

X'FFFFFFFF' 1111 1111 1111 1111 1111 1111 1111 1111

0000 0000 0000 0000 0000 1010 0000 0111

will do nothing.

**OR operation**

**OR Truth Table**

**Oper 1 Oper 2**

**Bit Bit Result**

0 0 0 When an OR operation is performed,

0 1 1 the result is 0 ONLY if both bits

1 0 1 are otherwise the result is 1.

1 1 1

The OR operation is used to set bits to 1

OR a bit with 0, it stays the same. OR a bit with 1, it becomes 1

RX Format: label O R,D(X,B)

The fullword in the register and the fullword at D(X,B) are ORed together. The result of the operation is placed in the register.

RR Format: label OR R1,R2

The fullword in the R1 and the fullword in R2 are ORed together. The result of the operation is placed in R1.

SI Format: label OI D(B),byte

The byte located at D(B) and the immediate byte specified by byte are ORed together. The result of the operation is placed at D(B).

SS Format: label OC D1(L,B1),D2(B2)

The L byte storage areas at D1(B1) and D2(B2) are ORed together. The result of the operation is placed starting at D1(B1).

For all formats, if every bit of the result is 0, the condition code is set to 0; otherwise the condition code is set to 1.

**OR Examples:**

Suppose that a storage area called BYTE contains X'05'. In binary:

BYTE 0 0 0 0 0 1 0 1

Execution of the instruction: OI BYTE,X'F0'

BYTE 0 0 0 0 0 1 0 1

1 1 1 1 0 0 0 0

RESULT 1 1 1 1 0 1 0 1

will change or "turn on" bits 0 through 3 in BYTE. So BYTE

now contains X'F5' (the character representation of 5).

Suppose that NUM contains X'00378C'. Then execution of the

instructions:

UNPK NUMOUT(5),NUM(3) results in NUMOUT F0 F0 F3 F7 C8

OI NUMOUT+4,X'F0'

NUMOUT+4 = C8 1100 1000

X'F0' 1111 0000

1111 1000

The contents of NUMOUT is now F0 F0 F3 F7 F8.

Execution of O R5,=X'FFFFFFFF'

Register 5 0000 0000 0000 0000 0000 1010 0000 0111

X'FFFFFFFF' 1111 1111 1111 1111 1111 1111 1111 1111

1111 1111 1111 1111 1111 1111 1111 1111

will change the contents of register 5 to all Fs.

While execution of O R5,=X'00000000'

Register 5 0000 0000 0000 0000 0000 1010 0000 0111

X'00000000' 0000 0000 0000 0000 0000 0000 0000 0000

0000 0000 0000 0000 0000 1010 0000 0111

will do nothing.

**EXCLUSIVE OR (XOR) operation**

**XOR Truth Table**

**Oper 1 Oper 2**

**Bit Bit Result**

0 0 0 When an EXCLUSIVE OR operation is

0 1 1 performed, the result is 1 if both

1 0 1 bits are different; the result is 0

1 1 0 if both bits are the same.

The XOR operation is used to set bits to the opposite value.

XOR a bit with 0, it stays the same. XOR a bit with 1, it becomes the opposite value

RX Format: label X R,D(X,B)

The fullword in the register and the fullword at D(X,B) are XORed together. The result of the operation is placed in the register.

RR Format: label XR R1,R2

The fullword in the R1 and the fullword in R2 are XORed together. The result of the operation is placed in R1.

SI Format: label XI D(B),byte

The byte located at D(B) and the immediate byte specified by byte are XORed together. The result of the operation is placed at D(B).

SS Format: label XC D1(L,B1),D2(B2)

The L byte storage areas at D1(B1) and D2(B2) are XORed together. The result of the operation is placed starting at D1(B1).

For all formats, if every bit of the result is 0, the condition code is set to 0; otherwise the condition code is set to 1.

**XOR Examples:**

Suppose that a storage area called FLD1 contains X'C2C3'. In binary:

FLD1 1100 0010 1100 0011

Execution of the instruction: XC FLD1(2),FLD1

FLD1 1100 0010 1100 0011

FLD1 1100 0010 1100 0011

FLD1 RESULT 0000 0000 0000 0000

will set the field to all zeros. So, XORing a field against itself

will set the field to all zeros.

Suppose that R5 contains 00 00 0A 07 and R7 contains 00 34 00 67. Then

execution of the instruction:

XR R5,R7

Register 5 0000 0000 0000 0000 0000 1010 0000 0111

Register 7 0000 0000 0011 0100 0000 0000 0110 0111

0000 0000 0011 0100 0000 1010 0110 0000

Now R5 contains 00 34 0A 60 and R7 contains 00 34 00 67

Execution of:

XR R7,R5

Register 7 0000 0000 0011 0100 0000 0000 0110 0111

Register 5 0000 0000 0011 0100 0000 1010 0110 0000

0000 0000 0000 0000 0000 1010 0000 0111

Now R5 contains 00 34 0A 60 and R7 contains 00 00 0A 07

Execution of:

XR R5,R7

Register 5 0000 0000 0011 0100 0000 1010 0110 0000

Register 7 0000 0000 0000 0000 0000 1010 0000 0111

0000 0000 0011 0100 0000 0000 0110 0111

Now R5 contains 00 34 00 67 and R7 contains 00 00 0A 07

The three XR instructions swapped the contents of the two registers.