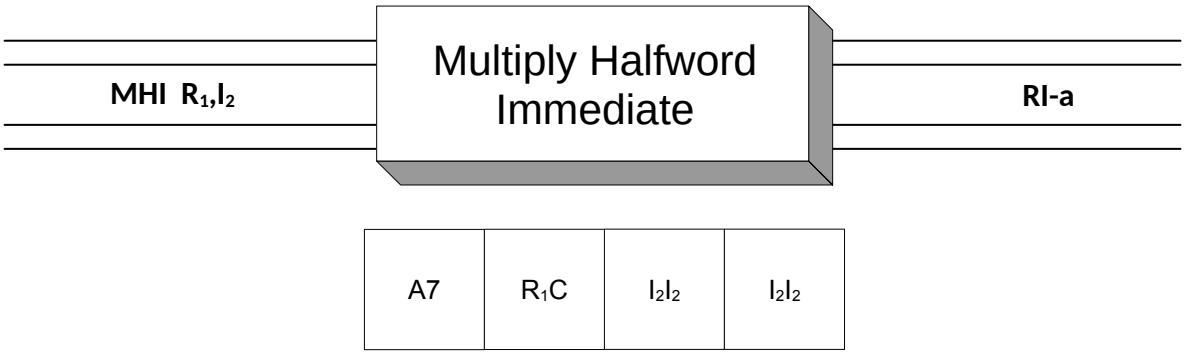
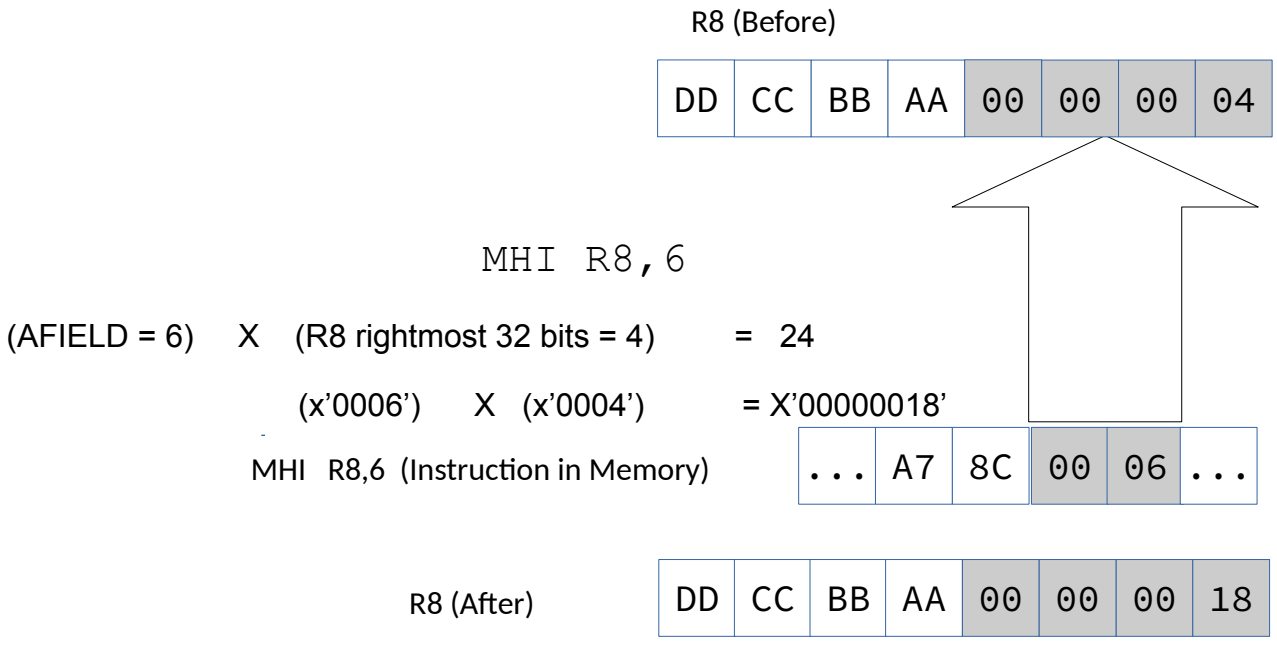


MHI (Multiply Halfword Immediate) computes the product of a halfword immediate constant contained in the rightmost two bytes of the instruction (treated as a 16-bit signed integer) with a fullword in the bits 32-63 of R_1. The rightmost 32-bits of the signed-binary product replace bits 32-63 of R1. The bits to the left of the 32 rightmost bits of the product are ignored, and significance is not tested. Bits 0-31 of R1 are unchanged by this operation.

The sign of the 32-bit product is determined by the rules of algebra. A zero product produces a positive sign. Overflows are not indicated. The condition code is unchanged by this operation.

Products that are outside the range -2,147,483,648 to +2,148364747,483,647 will be incorrect due to leftmost truncation back to 32 bits. Since we can't detect this situation by testing the condition code, we must insure this doesn't occur in other ways. One simple method (but not comprehensive) is insuring the multiplicand and the multiplier are in the range -46,000 to +46,000. If there is any question about possible overflow, choose a different multiply instruction like MFY or MGH.

Consider the following example,



Examples

Some Unrelated Multiply Halfword Immediates

```
R4 =  X'12121212_00000006'
R5 =  X'FFFFFFFF_FFFFFFFF'
R6 =  X'024C6DE5_01234567'
R7 =  X'98FAC346_7FFFFFFF'     RIGHTMOST 32 BITS = MAX POSITIVE VALUE
```

		REGISTER AFTER OPERATION (_ FOR READABILITY)
MHI	R4,4	R4 = X'12121212_00000018'
MHI	R4,B'00000100'	R4 = X'12121212_00000018'
MHI	R4,X'04'	R4 = X'12121212_00000018'
MHI	R4,H'4'	ASSEMBLY ERROR, TERM MUST BE DEC, CHAR, BIN OR HEX
MHI	R5,-1	R5 = X'FFFFFFFF_00000001
MHI	R5,X'FFFF'	R5 = X'FFFFFFFF_00000001'
MHI	R5,3	R5 = X'FFFFFFFF_FFFFFFFD'
MHI	R6,5	R6 = X'024C6DE5_05B05B03'
MHI	R7,2	R7 = X'98FAC346_FFFFFFFE' OVERFLOW NOT INDICATED

Tips

1) Immediate constants must be coded in decimal, binary, character, or hexadecimal.

2) Up to 47 bits may be generated as a result of the multiplication. Only 32 bits are stored in rightmost 32 bits of the register. As a result, the sign bit of the result may differ from the actual sign bit of the product in the case of an overflow.

3) This instruction doesn't set the condition code.