

IDIV—Signed Divide

Opcode	Instruction	Op/En	64-Bit Mode	Compat/Leg Mode	Description
F6 /7	IDIV r/m8	M	Valid	Valid	Signed divide AX by r/m8, with result stored in: AL := Quotient, AH := Remainder.
REX + F6 /7	IDIV r/m8 ¹	M	Valid	N.E.	Signed divide AX by r/m8, with result stored in AL := Quotient, AH := Remainder.
F7 /7	IDIV r/m16	M	Valid	Valid	Signed divide DX:AX by r/m16, with result stored in AX := Quotient, DX := Remainder.
F7 /7	IDIV r/m32	M	Valid	Valid	Signed divide EDX:EAX by r/m32, with result stored in EAX := Quotient, EDX := Remainder.
REX.W + F7 /7	IDIV r/m64	M	Valid	N.E.	Signed divide RDX:RAX by r/m64, with result stored in RAX := Quotient, RDX := Remainder.

NOTES:

1. In 64-bit mode, r/m8 can not be encoded to access the following byte registers if a REX prefix is used: AH, BH, CH, DH.

Instruction Operand Encoding

Op/En	Operand 1	Operand 2	Operand 3	Operand 4
M	ModRM:r/m (r)	N/A	N/A	N/A

Description

Divides the (signed) value in the AX, DX:AX, or EDX:EAX (dividend) by the source operand (divisor) and stores the result in the AX (AH:AL), DX:AX, or EDX:EAX registers. The source operand can be a general-purpose register or a memory location. The action of this instruction depends on the operand size (dividend/divisor).

Non-integral results are truncated (chopped) towards 0. The remainder is always less than the divisor in magnitude. Overflow is indicated with the #DE (divide error) exception rather than with the CF flag.

In 64-bit mode, the instruction's default operation size is 32 bits. Use of the REX.R prefix permits access to additional registers (R8-R15). Use of the REX.W prefix promotes operation to 64 bits. In 64-bit mode when REX.W is applied, the instruction divides the signed value in RDX:RAX by the source operand. RAX contains a 64-bit quotient; RDX contains a 64-bit remainder.

See the summary chart at the beginning of this section for encoding data and limits. See Table 3-51.

Table 3-51. IDIV Results

Operand Size	Dividend	Divisor	Quotient	Remainder	Quotient Range
Word/byte	AX	r/m8	AL	AH	-128 to +127
Doubleword/word	DX:AX	r/m16	AX	DX	-32,768 to +32,767
Quadword/doubleword	EDX:EAX	r/m32	EAX	EDX	-2 ³¹ to 2 ³¹ - 1
Doublequadword/quadword	RDX:RAX	r/m64	RAX	RDX	-2 ⁶³ to 2 ⁶³ - 1

Operation

```

IF SRC = 0
    THEN #DE; (* Divide error *)
FI;
IF OperandSize = 8 (* Word/byte operation *)
    THEN
        temp := AX / SRC; (* Signed division *)
        IF (temp > 7FH) or (temp < 80H)
            (* If a positive result is greater than 7FH or a negative result is less than 80H *)
            THEN #DE; (* Divide error *)
            ELSE
                AL := temp;
                AH := AX SignedModulus SRC;
        FI;
    ELSE IF OperandSize = 16 (* Doubleword/word operation *)
        THEN
            temp := DX:AX / SRC; (* Signed division *)
            IF (temp > 7FFFH) or (temp < 8000H)
                (* If a positive result is greater than 7FFFH
                or a negative result is less than 8000H *)
                THEN
                    #DE; (* Divide error *)
                ELSE
                    AX := temp;
                    DX := DX:AX SignedModulus SRC;
            FI;
        ELSE IF OperandSize = 32 (* Quadword/doubleword operation *)
            THEN
                temp := EDX:EAX / SRC; (* Signed division *)
                IF (temp > 7FFFFFFFFFH) or (temp < 80000000H)
                    (* If a positive result is greater than 7FFFFFFFFFH
                    or a negative result is less than 80000000H *)
                    THEN
                        #DE; (* Divide error *)
                    ELSE
                        EAX := temp;
                        EDX := EDX:EAX SignedModulus SRC;
                FI;
            ELSE IF OperandSize = 64 (* Doublequadword/quadword operation *)
                THEN
                    temp := RDX:RAX / SRC; (* Signed division *)
                    IF (temp > 7FFFFFFFFFFFFFFFFFH) or (temp < 8000000000000000H)
                        (* If a positive result is greater than 7FFFFFFFFFFFFFFFFFH
                        or a negative result is less than 8000000000000000H *)
                        THEN
                            #DE; (* Divide error *)
                        ELSE
                            RAX := temp;
                            RDX := RDX:RAX SignedModulus SRC;
                    FI;
                FI;
            FI;
        FI;
    FI;

```

Flags Affected

The CF, OF, SF, ZF, AF, and PF flags are undefined.

Protected Mode Exceptions

#DE	If the source operand (divisor) is 0. The signed result (quotient) is too large for the destination.
#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit. If the DS, ES, FS, or GS register is used to access memory and it contains a NULL segment selector.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.
#UD	If the LOCK prefix is used.

Real-Address Mode Exceptions

#DE	If the source operand (divisor) is 0. The signed result (quotient) is too large for the destination.
#GP	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS	If a memory operand effective address is outside the SS segment limit.
#UD	If the LOCK prefix is used.

Virtual-8086 Mode Exceptions

#DE	If the source operand (divisor) is 0. The signed result (quotient) is too large for the destination.
#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.
#UD	If the LOCK prefix is used.

Compatibility Mode Exceptions

Same exceptions as in protected mode.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#DE	If the source operand (divisor) is 0 If the quotient is too large for the designated register.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.
#UD	If the LOCK prefix is used.