x86 Instruction Set Reference

DIV

Unsigned Divide

Opcode	Mnemonic	Description
F6 /6	DIV r/m8	Unsigned divide AX by r/m8, with result stored in AL = Quotient, AH = Remainder.
F7 /6	DIV r/m16	Unsigned divide DX:AX by r/m16, with result stored in AX = Quotient, DX = Remainder.
F7 /6	DIV r/m32	Unsigned divide EDX:EAX by r/m32, with result stored in EAX = Quotient, EDX = Remainder.

Description

Divides (unsigned) the value in the AX, DX:AX, or EDX:EAX registers (dividend) by the source operand (divisor) and stores the result i
DX:AX, or EDX:EAX registers.

DIV Action

Operand Size	Dividend	Divisor	Quotient	Remainder	Maximum Quotient
Word/byte	AX	r/m8	AL	AH	2^8 - 1
Doubleword/word	DX:AX	r/m16	AX	DX	2^16 - 1
Quadword/doubleword	EDX:EAX	r/m32	EAX	EDX	2^32 - 1

The source operand can be a general-purpose register or a memory location. The action of this instruction depends on the operand size
See the table above.

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

Operation

```
if(Source == 0) Exception(DE); //divide error

if(OperandSize == 8) { //word/byte operation
        Temporary = AX / Source;
        if(Temporary > 0xFF) Exception(DE); //divide error
        else {
                AL = Temporary;
                AH = AX % Source;
        }
}
else if(OperandSize == 16) { //doubleword/word operation
        Temporary = DX:AX / Source;
        if(Temporary > 0xFFFF) Exception(DE); //divide error
        else {
                AX = Temporary;
                DX = DX:AX % Source;
        }
}
else { //quadword/doubleword operation
        Temporary = EDX:EAX / Source;
        if(Temporary > 0xFFFFFFFF) Exception(DE); //divide error
        else {
                EAX = Temporary;
                EDX = EDX:EAX % Source;
        }
}
```

Flags affected

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

Protected Mode Exceptions

#DE	If the source operand (divisor) is 0 If the quotient is too large for the designated register.
#DE	If the source operand (divisor) is 0 If the quotient is too large for the designated register.
#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 regi 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.

Real-Address Mode Exceptions

#DE	If the source operand (divisor) is 0. If the quotient is too large for the designated register.
#DE	If the source operand (divisor) is 0. If the quotient is too large for the designated register.
#GP	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 contains selector.

Virtual-8086 Mode Exceptions

#DE	If the source operand (divisor) is 0. If the quotient is too large for the designated register.
#DE	If the source operand (divisor) is 0. If the quotient is too large for the designated register.
#GP(0)	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.
#PF(fault-code)	If a page fault occurs.

Instruction	Latency	Throughput	Execution Unit
CPUID	0F3n/0F2n	0F3n/0F2n	0F2n
DIV	66-80/56-70	30/23	-