SUB—Subtract

Opcode	Instruction	64-Bit Mode	Compat/ Leg Mode	Description
2C *ib*	SUB AL, i*mm8*	Valid	Valid	Subtract *imm8* from AL.
2D *iw*	SUB AX, i*mm16*	Valid	Valid	Subtract *imm16* from AX.
2D *id*	SUB EAX, i*mm32*	Valid	Valid	Subtract *imm32* from EAX.
REX.W + 2D *id*	SUB RAX, i*mm32*	Valid	N.E.	Subtract *imm32* sign-extended to 64-bits from RAX.
80 /5 *ib*	SUB *r/m8, imm8*	Valid	Valid	Subtract *imm8* from *r/m8.*
REX + 80 /5 *ib*	SUB *r/m8*, imm8*	Valid	N.E.	Subtract *imm8* from *r/m8.*
81 /5 *iw*	SUB *r/m16, imm16*	Valid	Valid	Subtract *imm16* from *r/m16.*
81 /5 *id*	SUB *r/m32, imm32*	Valid	Valid	Subtract *imm32* from *r/m32.*
REX.W + 81 /5 *id*	SUB *r/m64, imm32*	Valid	N.E.	Subtract *imm32* sign-extended to 64-bits from *r/m64.*
83 /5 *ib*	SUB *r/m16, imm8*	Valid	Valid	Subtract sign-extended *imm8* from *r/m16.*
83 /5 *ib*	SUB *r/m32, imm8*	Valid	Valid	Subtract sign-extended *imm8* from *r/m32.*
REX.W + 83 /5 *ib*	SUB *r/m64, imm8*	Valid	N.E.	Subtract sign-extended *imm8* from *r/m64.*
28 /*r*	SUB *r/m8, r8*	Valid	Valid	Subtract *r8* from *r/m8.*
REX + 28 /*r*	SUB *r/m8*, r8**	Valid	N.E.	Subtract *r8* from *r/m8.*
29 /*r*	SUB *r/m16, r16*	Valid	Valid	Subtract *r16* from *r/m16.*
29 /*r*	SUB *r/m32, r32*	Valid	Valid	Subtract *r32* from *r/m32.*
REX.W + 29 /*r*	SUB *r/m64, r32*	Valid	N.E.	Subtract *r64* from *r/m64.*
2A /*r*	SUB *r8, r/m8*	Valid	Valid	Subtract *r/m8* from *r8.*
REX + 2A /*r*	SUB *r8*, r/m8**	Valid	N.E.	Subtract *r/m8* from *r8.*
2B /*r*	SUB *r16, r/m16*	Valid	Valid	Subtract *r/m16* from *r16.*
2B /*r*	SUB *r32, r/m32*	Valid	Valid	Subtract *r/m32* from *r32.*
REX.W + 2B /*r*	SUB *r64, r/m64*	Valid	N.E.	Subtract *r/m64* from *r64.*

NOTES:

* 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.

Description

Subtracts the second operand (source operand) from the first operand (destination operand) and stores the result in the destination operand. The destination operand can be a register or a memory location; the source operand can be an immediate, register, or memory location. (However, two memory operands cannot be used in one instruction.) When an immediate value is used as an operand, it is sign-extended to the length of the destination operand format.

The SUB instruction performs integer subtraction. It evaluates the result for both signed and unsigned integer operands and sets the OF and CF flags to indicate an overflow in the signed or unsigned result, respectively. The SF flag indicates the sign of the signed result.

In 64-bit mode, the instruction's default operation size is 32 bits. Using a REX prefix in the form of REX.R permits access to additional registers (R8-R15). Using a REX prefix in the form of REX.W promotes operation to 64 bits. See the summary chart at the beginning of this section for encoding data and limits.

This instruction can be used with a LOCK prefix to allow the instruction to be executed atomically.

Operation

```
DEST ← (DEST – SRC);
```

Flags Affected

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

Protected Mode Exceptions

#GP(0)	If the destination is located in a non-writable segment. 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 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 but the destination is not a memory operand.

Real-Address Mode Exceptions

#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 but the destination is not a memory operand.

Virtual-8086 Mode Exceptions

#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 but the destination is not a memory operand.

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.
#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 but the destination is not a memory operand.