Assembly Language

Tiago Oliveira
Instituto de Matemática e Estatística
Departamento de Ciência da Computação
Universidade Federal da Bahia

Working with Integers

Unsigned integers (which are non-negative) are represented in a very straightforward binary manner. The number 200 as an one byte unsigned integer would be represented as by 11001000 (or C8 in hex);

if it is sign?

imul

The source is either a register or a memory reference;
It can not be an immediate value.
Exactly what multiplication is performed depends on the size of the source operand;
If the operand is byte sized, it is multiplied by the byte in the AL register and the result is stored in the 16 bits of AX;
If the source is 16-bit, it is multiplied by the word in AX and the 32-bit result is stored in DX:AX

imul

The IMUL instruction has the same formats as MUL, but also adds some other instruction formats.
There are two and three operand formats

```
imul    dest, source1
imul    dest, source1, source2
```

imul

dest	source1	source2	Action
	reg/mem8		AX = AL*source1
	reg/mem16		DX:AX = AX*source1
	reg/mem32		EDX:EAX = EAX*source1
reg16	reg/mem16		dest *= source1
reg32	reg/mem32		dest *= source1
reg16	immed8		dest *= immed8
reg32	immed8		dest *= immed8
reg16	immed16		dest *= immed16
reg32	immed32		dest *= immed32
reg16	reg/mem16	immed8	dest = source1*source2
reg32	reg/mem32	immed8	dest = source1*source2
reg16	reg/mem16	immed16	dest = source1*source2
reg32	reg/mem32	immed32	dest = source1*source2

idiv - div

If the source is 8-bit, then AX is divided by the operand. The quotient is stored in AL and the remainder in AH;

If the source is 16-bit, then DX:AX is divided by the operand. The quotient is stored into AX and remainder into DX;

If the source is 32-bit, then EDX:EAX is divided by the operand and the quotient is stored into EAX and the remainder into EDX.

idiv – div

Instruction	Description
IDIV *r/m8*	Signed divide AX by *r/m8*, with result stored in: AL ← Quotient, AH ← Remainder.
IDIV *r/m8**	Signed divide AX by *r/m8*, with result stored in AL ← Quotient, AH ← Remainder.
IDIV *r/m16*	Signed divide DX:AX by *r/m16*, with result stored in AX ← Quotient, DX ← Remainder.
IDIV *r/m32*	Signed divide EDX:EAX by *r/m32*, with result stored in EAX ← Quotient, EDX ← Remainder.
IDIV *r/m64*	Signed divide RDX:RAX by *r/m64*, with result stored in RAX ← Quotient, RDX ← Remainder.

idiv – div

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

Bibliografia

CARTER, Paul A. **PC Assembly Language**. Github, 2004.

Contatos:
tiagocompuesc@gmail.com