

# **ICS 51**

# **Introductory Computer Organization**

## **: Data Range and Data Types**

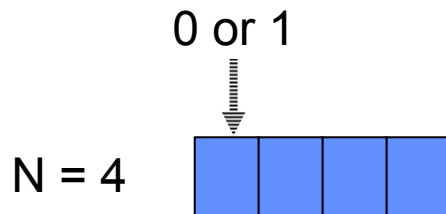
# Today's topics

- **Data range**
- **Data types**
- **How to specify addresses**

# Data range

- How many numbers can be represented with N bits?  $2^N$
- Maximum value N bits can hold?
- **Depends on representation!**
- **Unsigned number:  $2^N - 1$** 
  - If N=4, max number is 15
  - Range is (0,15), therefore 16 numbers are represented!

0000 – 1111



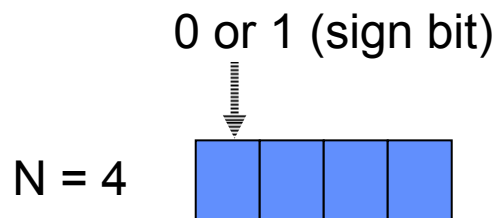
- $2^4 - 1 = 15$
- 4 bits can represent 16 ( $2^4$ ) numbers ranging from 0 to 15 ( $2^4 - 1$ )

Binary	Unsigned
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9
1010	10
1011	11
1100	12
1101	13
1110	14
1111	15

# Data range (cont.)

- **Signed numbers: one bit is used to determine sign!**
- $2^{N-1}-1$
- **2's complement**
  - If  $N=4$ , max number is 7
  - Range is  $(-8, 7)$ , therefore 16 numbers are represented

**1000 – 0111<sub>2</sub>**



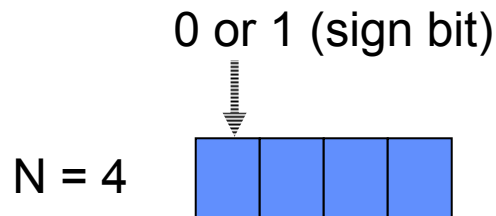
- Max number:  $2^{4-1} - 1 = 7$
- 4 bits can represent 16 ( $2^N$ ) numbers ranging from -8 ( $-2^{N-1}$ ) to 7 ( $2^{N-1}-1$ )

Copyright A. Nicolau & A. Veidenbaum -- do not distribute

Binary	2's Comp
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	-8
1001	-7
1010	-6
1011	-5
1100	-4
1101	-3
1110	-2
1111	-1

# Data range (cont.)

- **1's complement**
  - If  $N=4$ , max number is 7
  - Range is  $(-7, 7)$ , therefore 15 numbers are represented
  - **1000** – **0111** where 0 is 0000 and  $-0$  is 1111

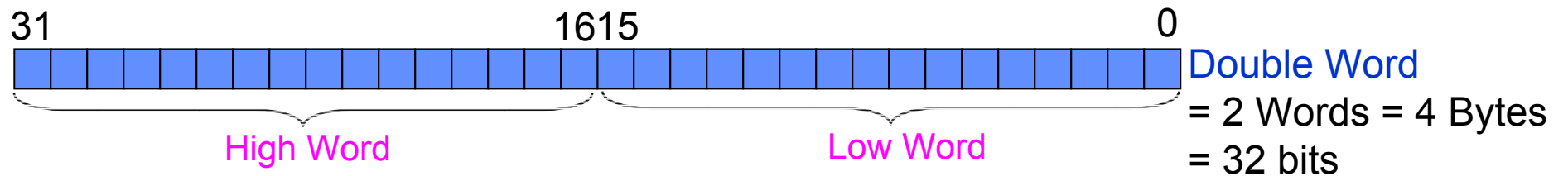
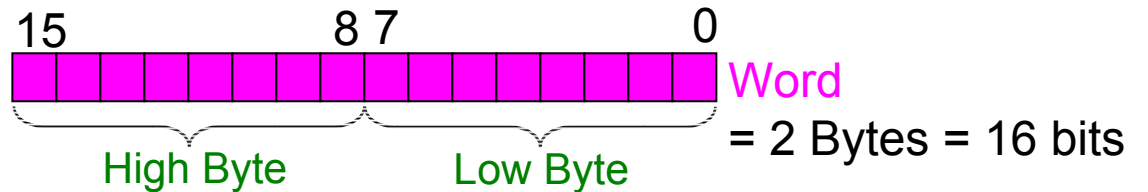
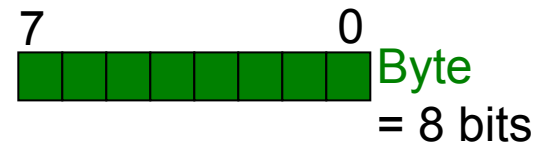


- Max number:  $2^{4-1} - 1 = 7$
- 4 bits can represent 15 numbers ranging from  $-7$  ( $-2^{N-1}+1$ ) to  $7$  ( $2^{N-1}-1$ )

Copyright A. Nicolau & A. Veidenbaum -- do not distribute

Binary	1's Comp
<b>0000</b>	<b>0</b>
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
<b>0111</b>	<b>7</b>
<b>1000</b>	<b>-7</b>
1001	-6
1010	-5
1011	-4
1100	-3
1101	-2
1110	-1
<b>1111</b>	<b>0</b>

# Data Types



# Data Types (cont.)

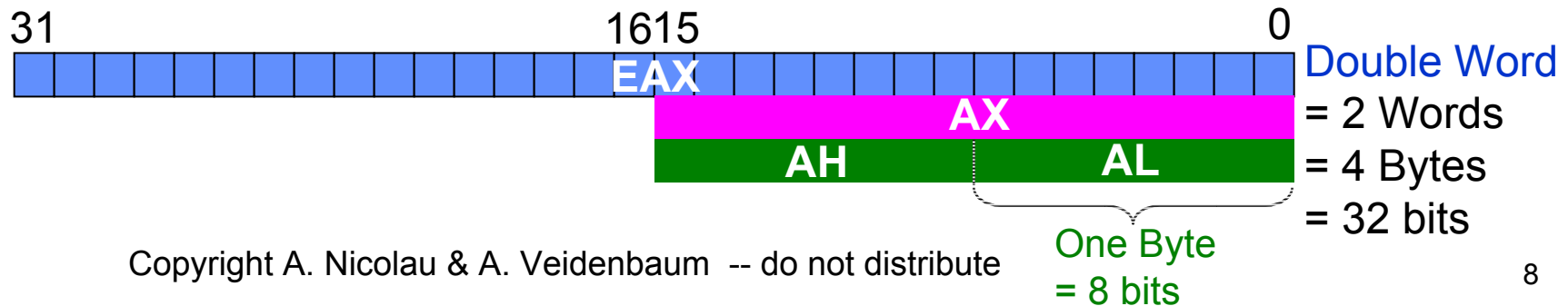
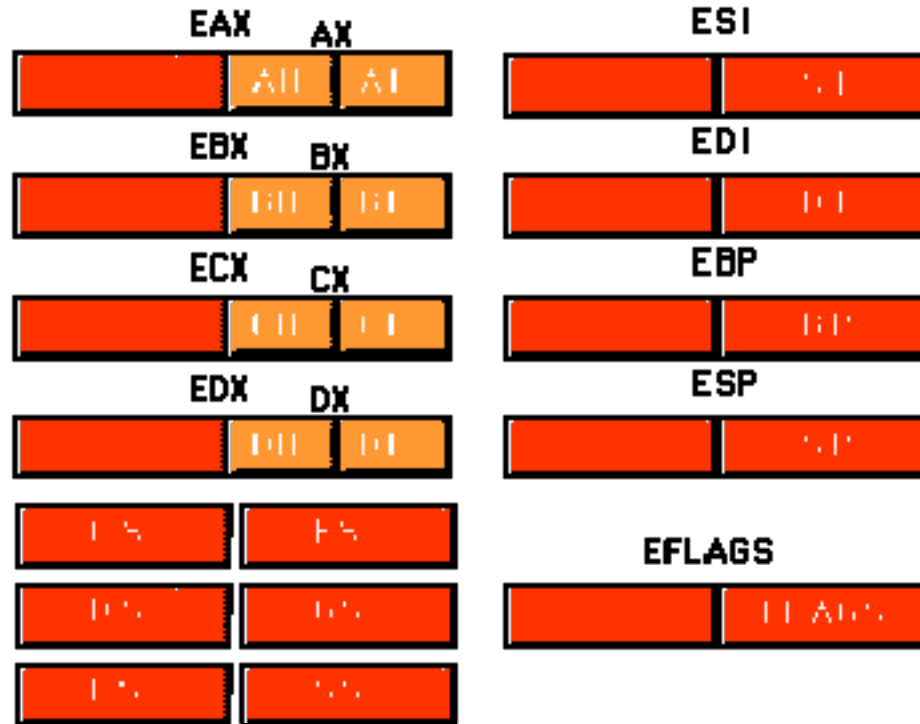
- **Most common error: operand size conflict**

Data type in C/C++ for INTEL!	Size in bytes
char, unsigned char, signed char	1 byte <b>byte</b>
short, unsigned short	2 bytes <b>word</b>
int, unsigned int	4 bytes <b>dword</b>
long, unsigned long	4 bytes <b>dword</b>
float	4 bytes
double	8 bytes
long double <sup>1</sup>	8 bytes

» <sup>1</sup> The representation of long double and double is identical. However, long double and double are separate.

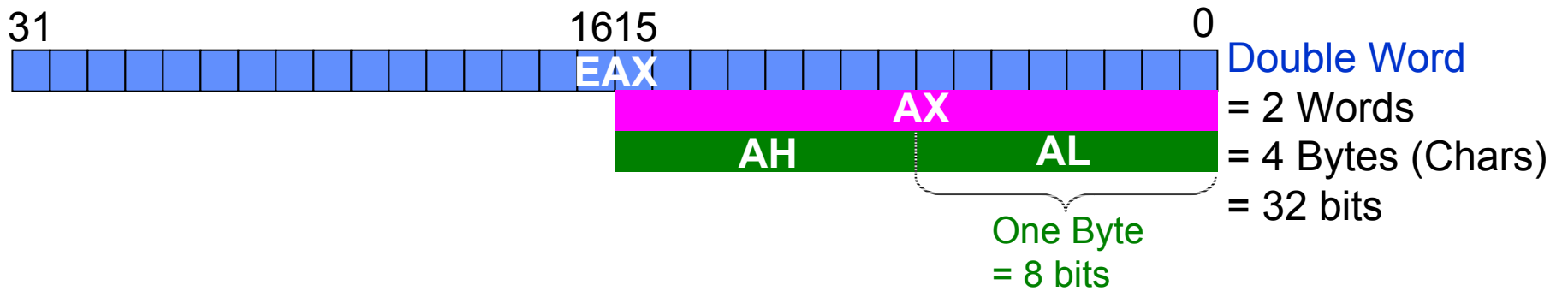
# Data Types (cont.)

- x86 registers

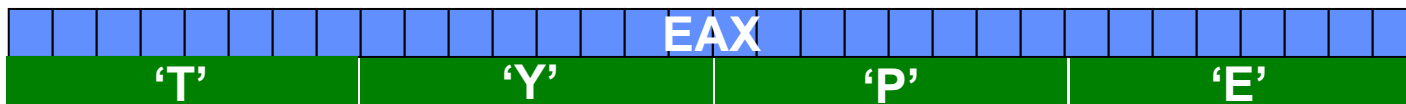


# Data types (cont.)

- **Example**



- **“TYPE” in EAX**

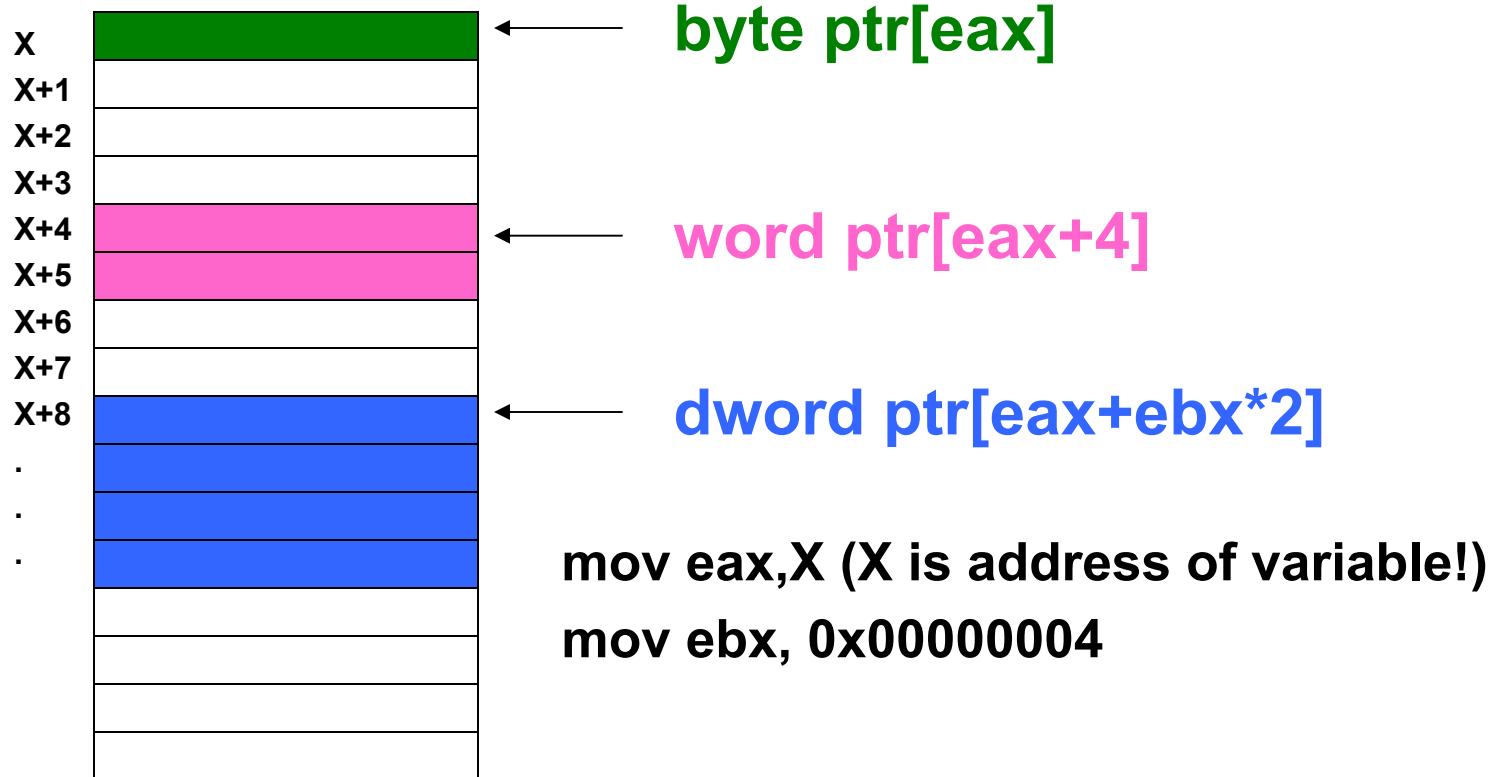


- ✓“PE” in AX
- ✓‘P’ in AH
- ✓‘E’ in AL

# Data Types (cont.)

- Each byte has a unique address in Memory
- All pointers are 32 bits long

## Memory



# Data Types (cont.)

- Example

```

#include <stdio.h>

```

```

char *yourName = "name";

```

```

void main()

```

```

{

```

```

    __asm{

```

```

        mov eax, yourName;

```

```

        mov bx, ax;

```

```

        mov bh, al;

```

```

        mov bl, ah;

```

```

        mov ebx, dword ptr[eax]; ①

```

```

        mov bx, word ptr[eax+2]; ②

```

```

        mov bh, word ptr[eax+3]; ③

```

```

        mov bl, byte ptr[eax+4]; ④

```

```

    }

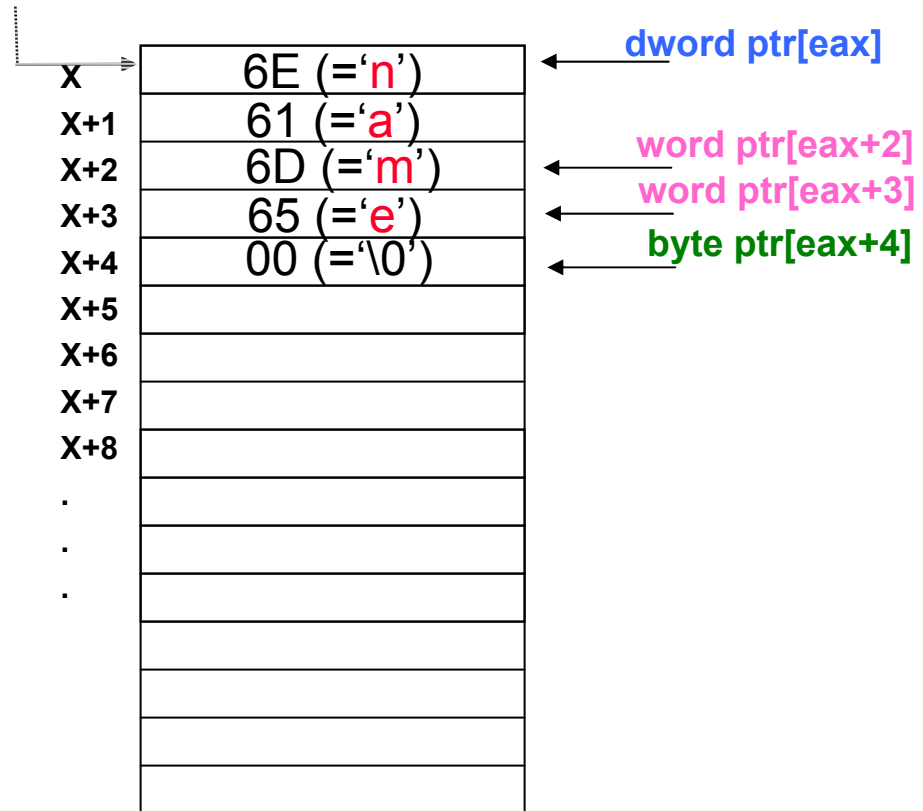
```

```

}

```

yourName = X (32 bit values) => eax



Compile → error C2443: operand size conflict

# Address specification

Base          Index          Scale          Displacement

$$\begin{bmatrix} EAX \\ EBX \\ ECX \\ EDX \\ ESP \\ EBP \\ ESI \\ EDI \end{bmatrix} + \begin{bmatrix} (EAX) \\ (EBX) \\ (ECX) \\ (EDX) \\ (ESP) \\ (EBP) \\ (ESI) \\ (EDI) \end{bmatrix} * \begin{bmatrix} 1 \\ 2 \\ 4 \\ 8 \end{bmatrix} + \begin{bmatrix} \text{None} \\ 8\text{-bit} \\ 16\text{-bit} \\ 32\text{-bit} \end{bmatrix}$$

**Offset = Base + (Index \* Scale) + Displacement**

# Address specification (cont.)

- **Examples:**

- `mov byte ptr[eax], dl`
- `mov cx, word ptr[eax+ebx]`
- `mov edx, dword ptr[eax+ebx*2]`
- `mov edx, dword ptr[eax+ebx*2+8]`