Today: Bits, Bytes, and Integers

- **Representing information as bits**
- **Bit-level manipulations**
- **Integers**
  - **Representation: unsigned and signed**
  - Conversion, casting
  - Expanding, truncating
  - Addition, negation, multiplication, shifting
  - Summary
- **Representations in memory, pointers, strings**
- **Summary**

Encoding Integers

Unsigned

$$B2U(X) \;=\; \sum_{i=0}^{w-1} x_i \cdot 2^i$$

Two's Complement

$$B2T(X) \;=\; -x_{w-1} \cdot 2^{w-1} + \sum_{i=0}^{w-2} x_i \cdot 2^i$$

```
short int x =  15213;
short int y = -15213;
```

Sign Bit

- **C short 2 bytes long**

	Decimal	Hex	Binary
x	15213	3B 6D	00111011 01101101
y	-15213	C4 93	11000100 10010011

- **Sign Bit**
  - For 2's complement, most significant bit indicates sign
    - 0 for nonnegative
    - 1 for negative

Two-complement: Simple Example

```
          -16   8    4    2    1
  10 =   0    1    0    1    0       8+2 = 10


          -16   8    4    2    1
 -10 =   1    0    1    1    0      -16+4+2 = -10
```

Two-complement Encoding Example (Cont.)

```
x =        15213:  00111011 01101101
y =       –15213:  11000100 10010011
```

Weight	15213		-15213	
1	1	1	1	1
2	0	0	1	2
4	1	4	0	0
8	1	8	0	0
16	0	0	1	16
32	1	32	0	0
64	1	64	0	0
128	0	0	1	128
256	1	256	0	0
512	1	512	0	0
1024	0	0	1	1024
2048	1	2048	0	0
4096	1	4096	0	0
8192	1	8192	0	0
16384	0	0	1	16384
-32768	0	0	1	-32768
Sum		**15213**		**-15213**

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

Numeric Ranges

- **Unsigned Values**
  - *UMin* = 0

    000...0
  - *UMax* = $2^w - 1$

    111...1

- **Two's Complement Values**
  - *TMin* = -2^{w-1}

    100...0
  - *TMax* = $2^{w-1} - 1$

    011...1
  - Minus 1

    111...1

Values for *W* = 16

	Decimal	Hex	Binary
UMax	**65535**	`FF FF`	`11111111 11111111`
TMax	**32767**	`7F FF`	`01111111 11111111`
TMin	**-32768**	`80 00`	`10000000 00000000`
`-1`	**-1**	`FF FF`	`11111111 11111111`
`0`	**0**	`00 00`	`00000000 00000000`

Values for Different Word Sizes

	W			
	8	**16**	**32**	**64**
UMax	255	65,535	4,294,967,295	18,446,744,073,709,551,615
TMax	127	32,767	2,147,483,647	9,223,372,036,854,775,807
TMin	-128	-32,768	-2,147,483,648	-9,223,372,036,854,775,808

- **Observations**
  - $|TMin|$ = $TMax + 1$
    - Asymmetric range
  - $UMax$ = $2 * TMax + 1$

- **C Programming**
  - #include <limits.h>
  - Declares constants, e.g.,
    - ULONG_MAX
    - LONG_MAX
    - LONG_MIN
  - Values platform specific

Unsigned & Signed Numeric Values

X	B2U(X)	B2T(X)
0000	0	0
0001	1	1
0010	2	2
0011	3	3
0100	4	4
0101	5	5
0110	6	6
0111	7	7
1000	8	−8
1001	9	−7
1010	10	−6
1011	11	−5
1100	12	−4
1101	13	−3
1110	14	−2
1111	15	−1

- **Equivalence**
  - Same encodings for nonnegative values

- **Uniqueness**
  - Every bit pattern represents unique integer value
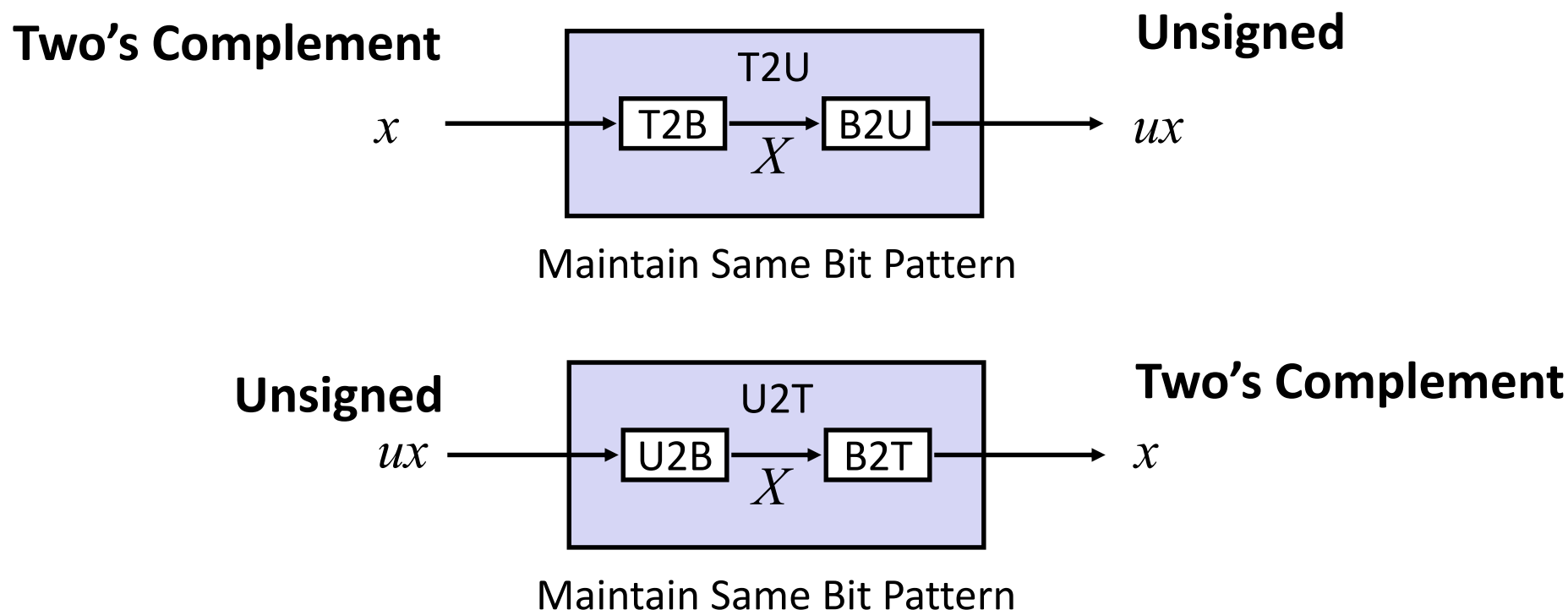 - Each representable integer has unique bit encoding

- $\Rightarrow$ **Can Invert Mappings**
  - U2B(x) = B2U^{-1}(x)
    - Bit pattern for unsigned integer
  - T2B(x) = B2T^{-1}(x)
    - Bit pattern for two's comp integer

Today: Bits, Bytes, and Integers

- **Representing information as bits**
- **Bit-level manipulations**
- **Integers**
  - Representation: unsigned and signed
  - **Conversion, casting**
  - Expanding, truncating
  - Addition, negation, multiplication, shifting
  - Summary
- **Representations in memory, pointers, strings**

Mapping Between Signed & Unsigned

Two's Complement

x

Unsigned

ux

T2U

T2B → X → B2U

Maintain Same Bit Pattern

Unsigned

ux

Two's Complement

x

U2T

U2B → X → B2T

Maintain Same Bit Pattern

- **Mappings between unsigned and two's complement numbers:**
  Keep bit representations and reinterpret

Mapping Signed ↔ Unsigned

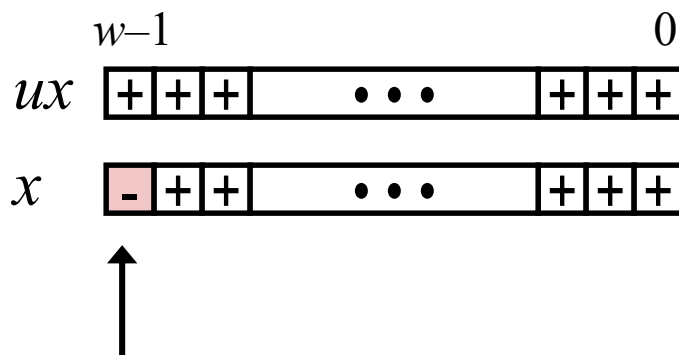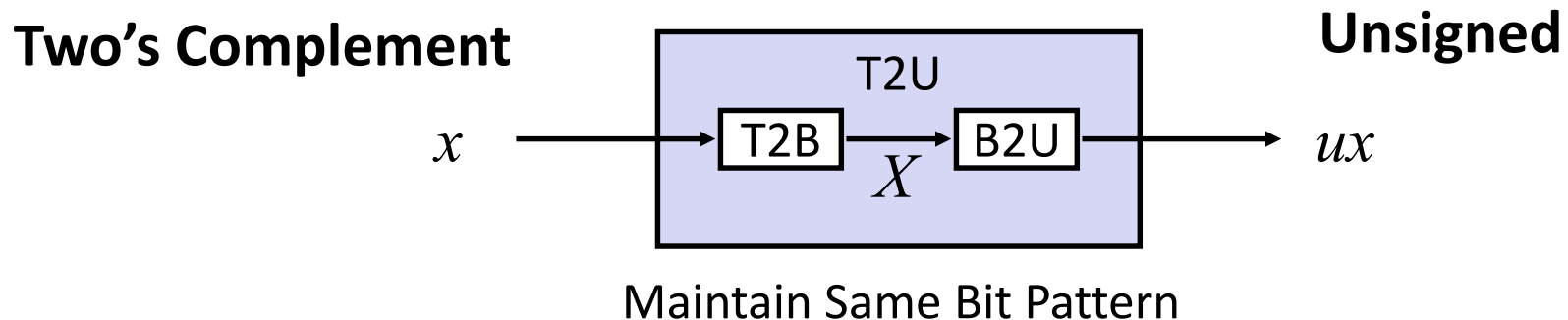| Bits |
|------|
| 0000 |
| 0001 |
| 0010 |
| 0011 |
| 0100 |
| 0101 |
| 0110 |
| 0111 |
| 1000 |
| 1001 |
| 1010 |
| 1011 |
| 1100 |
| 1101 |
| 1110 |
| 1111 |

Signed
0
1
2
3
4
5
6
7
−8
−7
−6
−5
−4
−3
−2
−1

T2U →

U2T ←

Unsigned
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15

Mapping Signed ↔ Unsigned

Bits
0000
0001
0010
0011
0100
0101
0110
0111
1000
1001
1010
1011
1100
1101
1110
1111

Signed
0
1
2
3
4
5
6
7
−8
−7
−6
−5
−4
−3
−2
−1

Unsigned
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15

=

+/- 16

Relation between Signed & Unsigned

Two's Complement   **Unsigned**

x → | T2U |
      | T2B | → X → | B2U | → ux

Maintain Same Bit Pattern

$w{-}1$ 0

ux | + | + | + | • • • | + | + | + |

x | - | + | + | • • • | + | + | + |

Large negative weight
becomes
Large positive weight

Conversion Visualized

- **2's Comp. → Unsigned**
  - Ordering Inversion
  - Negative → Big Positive
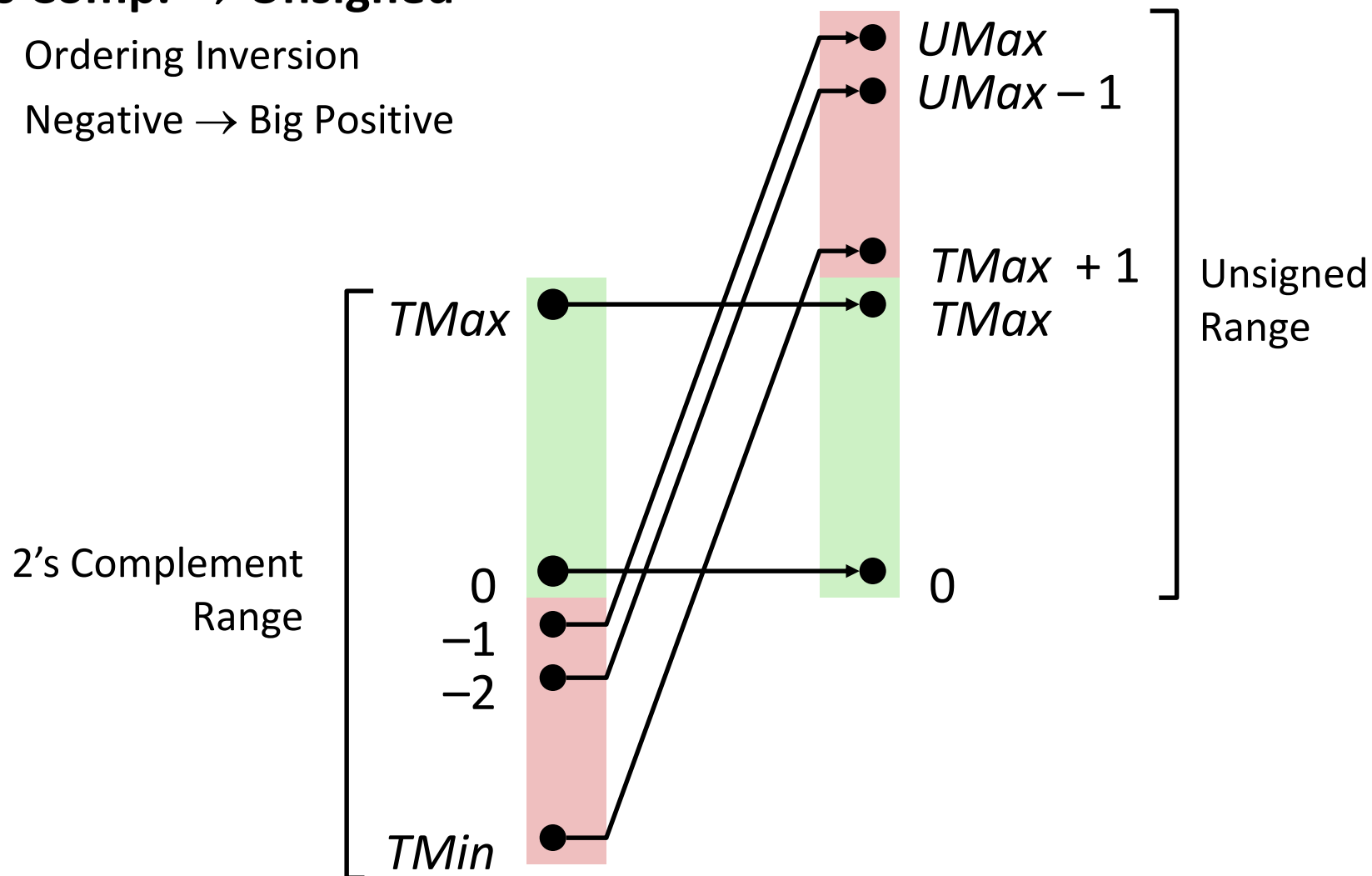
Signed vs. Unsigned in C

- ■ **Constants**
  - ▪ By default are considered to be signed integers
  - ▪ Unsigned if have "U" as suffix

    `0U, 4294967259U`

- ■ **Casting**
  - ▪ Explicit casting between signed & unsigned same as U2T and T2U

    ```
    int tx, ty;
    unsigned ux, uy;
    tx = (int) ux;
    uy = (unsigned) ty;
    ```

  - ▪ Implicit casting also occurs via assignments and procedure calls

    ```
    tx = ux;                     int fun(unsigned u);
    uy = ty;                     uy = fun(tx);
    ```

Casting Surprises

- **Expression Evaluation**
  - If there is a mix of unsigned and signed in single expression, *signed values implicitly cast to unsigned*
  - Including comparison operations **<, >, ==, <=, >=**
  - Examples for W = 32:   **TMIN = -2,147,483,648 ,   TMAX = 2,147,483,647**

- **Constant₁**

Constant$_1$	Constant$_2$	Relation	Evaluation
0	0U	**==**	**unsigned**
-1	0	**<**	**signed**
-1	0U	**>**	**unsigned**
2147483647	-2147483647-1	**>**	**signed**
2147483647U	-2147483647-1	**<**	**unsigned**
-1	-2	**>**	**signed**
(unsigned)-1	-2	**>**	**unsigned**
2147483647	2147483648U	**<**	**unsigned**
2147483647	(int) 2147483648U	**>**	**signed**

Unsigned vs. Signed: Easy to Make Mistakes

```
unsigned i;
for (i = cnt-2; i >= 0; i--)
  a[i] += a[i+1];
```

- Can be very subtle

```
#define DELTA sizeof(int)
int i;
for (i = CNT; i-DELTA >= 0; i-= DELTA)
  . . .
```

Summary
Casting Signed ↔ Unsigned: Basic Rules

- **Bit pattern is maintained**

- **But reinterpreted**

- **Can have unexpected effects: adding or subtracting 2^w**

- **Expression containing signed and unsigned int**
  - `int` is cast to `unsigned`!!