Lecture 1

<2016-03-28 Mon>

Contents

1 Bits, Bytes, and Integers

- Bit is 0 or 1
- Electronic Implementation
  - High voltage: 1, low voltage: 0
- Byte is 8 bits
  - hex: 00 - FF
  - write 0x123f in C for hex number

1.1 Boolean Algebra

- And &
- Or |
- Not ~
- Xor ^

1.1.1 example

```
76543210
01101001:  {0, 3, 5, 6} in bit set
01010101:  {0, 2, 4, 6} in bit set
```

- operate on bit vectors
- Representing and Manipulation of Sets
  - & intersection `{0, 6}`
  - | union `{0, 2, 3, 4, 5, 6}`
  - ^ symmetric difference `{2, 3, 4, 5}`
  - ~ complement

1.1.2 Contrast to Logic Operation

- logical operator && and, || or, ! nor
  - 0 as False
  - return 0 or 1
  - early termination
  - example: `p && *p` avoids null pointer access

1.1.3 Shift Operation

- left shift `x << y`
- right shift `x >> y`
  - logical shift: pad with 0
  - arithmetic shift: pad with left most bit

- undefined behavior
  - shift amount `< 0` or `>=` word size

`11100010`	operation
`00010000`	`« 3`
`00111000`	`Log.  » 2`
`11111000`	`Arith.  » 2`

1.2 Integer

- Unsigned
- Signed
  - 2's complement
  - sign bit: most significant bit
    - 0: nonnegative
    - 1: negative

1.2.1 example

```
15213
00111011 01101101
-15213
11000100 10010011
```

-15213 in binary	weight	value	
1	1	1	least significant bit
1	2	2	
0	4	0	
0	8	0	
1	16	16	
0	32	0	
0	64	0	
1	128	128	
0	256	0	
0	512	0	
1	1024	1024	
0	2048	0	
0	4096	0	
0	8192	0	
1	16384	16384	
1	-32768	-32768	most significant bit
		-15213	

1.2.2 Numerical Range

- Numerical Range
  - Unsigned
    - umin: 0
    - umax: 2^{w} - 1
  - Signed
    - tmin: -2^{w-1}
    - tmax: 2^{w-1} - 1

X	B2U(X) unsigned	B2T(X) signed
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

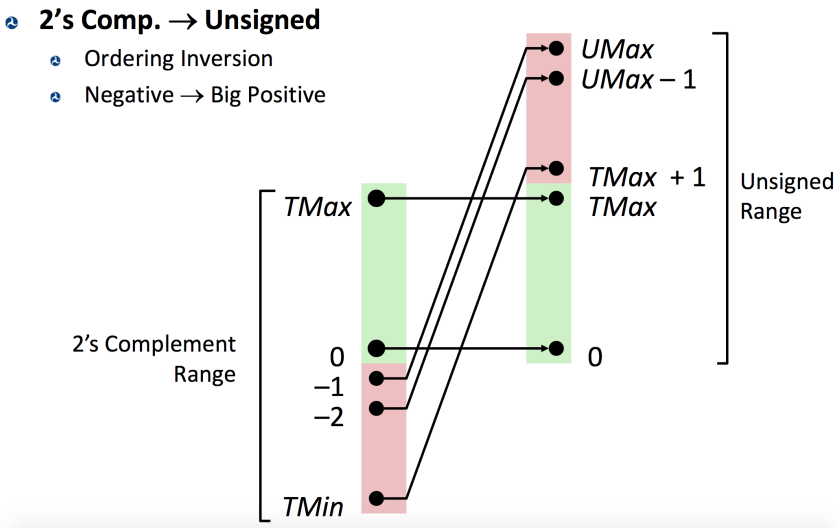

Figure 1: 2's complement

1.2.3 Signed vs Unsigned in C

- Constants

  - defaults to signed
  - unsigned: 4243653U

- Casting
  - mix of unsigned and signed: signed are implicitly cast to unsigned

1. Observation
   - `| TMIN | = TMAX + 1`
   - `UMAX = 2 * TMAX + 1`
   - For C programming

```
#include <limits.h>

#define ULONG_MAX
#define LONG_MIN
```

1.2.4 example

```
W = 32 (word size)
TMIN = -2147483648
TMAX = 2147483647
```

const1	const2	result	
-1	0	<	signed
-1	0U	>	unsigned
2147483647	-2147483647-1	>	signed
2147483647U	-2147483647-1	<	unsigned
(unsigned)-1	-2	>	unsigned
2147483647	2147483648U	<	unsigned
2147483647	(int)2147483648U	>	signed

1.2.5 Sign Extension

- w-bit signed integer
- convert to w+k-bit integer with same value
- make k copies of signed bit

word size	decimal	hex	bin
2	-15213	C4 93	11000100 10010011
4	-15213	FF FF C4 93	11111111 11111111 11000100 10010011

- Make *k* copies of sign bit:
- $X' = x_{w-1}, \ldots, x_{w-1}, x_{w-1}, x_{w-2}, \ldots, x_0$

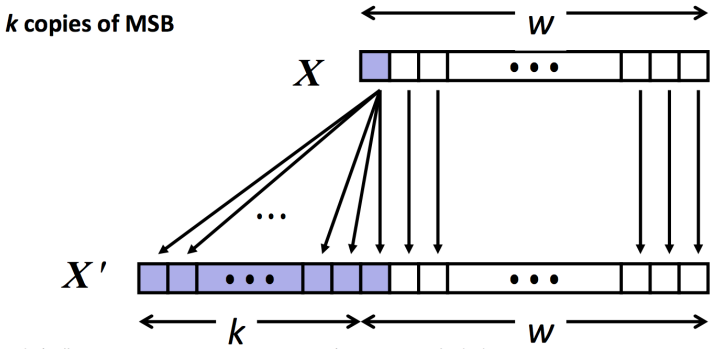

Figure 2: sign extension