EECS22 Lab Week4

TA: Weiwei CHEN
Office hour: Mon, 11:00-12:50am EH 1141
weiwei.chen@uci.edu
eecs22@eecs.uci.edu

Bitwise Operators

- Bitwise "and" operator **&**
- Bitwise "or" operator **|**
- Bitwise "exclusive or" operator **^**
- Bitwise "ones complement" operator **~**
- Shift left <<
- Shift right >>

	a	0	0	1	1
	b	0	1	0	1
and	a & b	0	0	0	1
or	a | b	0	1	1	1
exclusive or	a ^ b	0	1	1	0
one's complement	~a	1	1	0	0

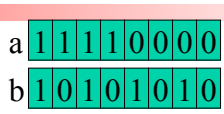
Operation

- Logic Operation
  - Left-shift, right-shift
  - 3 << 2 = 3 * 2 ^2 = 12;
  - 3 >> 2 = 3 / (2 ^2) = 0;
- Evaluation order
  - 3 << 2 * 4 = 768
  - (3 << 2) * 4 = 48
- Unary operation
  - +, -
  - **x = (10 – (3 – (-10 - -20)));**

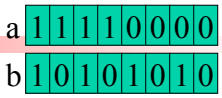
Examples

```
unsigned int c, a, b;
   a =
   b =
   c = a & b;
   c = a | b;
   c = a ^ b;
   c = ~a;

   c = a << 2;
   c = a >> 3;
```



Examples

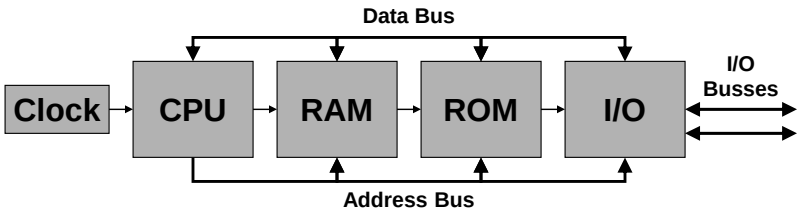



```
unsigned int c, a, b;
   a = (11110000)2 = (240)10;
   b = (10101010)2 = (170)10;
   c = a & b = (10100000)2 = (160)10;
   c = a | b = (11111010)2 = (250)10;
   c = a ^ b = (01011010)2 = (90)10;
   c = ~a = (00001111)2 = (15)10;

   c = a << 2 = (1111000000)2 = (960)10;
   c = a >> 3 = (00011110)2 = (30)10;
```

Basic Computer Architecture

- Essential Computer Components
  - Central Processing Unit (CPU)
    - e.g. Intel Pentium, Motorola PowerPC, Sun SPARC, ...
  - Random Access Memory (RAM)
    - storage for program and data, read and write access
  - Read Only Memory (ROM)
    - fixed storage for basic input/output system (BIOS)
  - I/O Units
    - Input/output units connecting to peripherals



EECS10: Computational Methods in ECE, Lecture 21 (c) 2010 R. Doemer 3

Binary Data Representation

- Programs and data in a computer are represented in binary format
  - 1 *bit* (binary digit), 2 possible values ■ □
    - 0 (false, “no”, power off, “empty”, ...)
    - 1 (true, “yes”, power on, “solid”, ...)
  - 1 *byte* = 8 bits (2^8 = 256 values) ■□■□□□■□
    - in C, type **`char`** equals one byte*
  - 1 *word* = 4 bytes* (2^{32} = 4294967296 values) ■□■□□□■■ □■■□■□■□ ■□■□□■■□ ■□■■□□■■
    - in C, type **`int`** equals one word
- Memory size is measured in Bytes
  - 1 KB = 1024 byte = 1 “kilo byte”
  - 1 MB = 1024*1024 byte = 1 “mega byte”
  - 1 GB = 1024*1024*1024 byte = 1 “giga byte”

*(*architecture dependent!)*

EECS10: Computational Methods in ECE, Lecture 21 (c) 2010 R. Doemer 4

Binary Data Representation

- Memory is composed of addressable bytes
  - Example:
    1 KB of memory
  - What is the value at address 7?

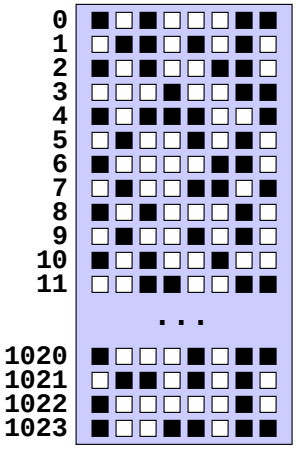
7 □■□□■■□■
7 6 5 4 3 2 1 0

$= 0*2^7 + 1*2^6 + 0*2^5 + 0*2^4 + 1*2^3 + 1*2^2 + 0*2^1 + 1*2^0$

$= 0*128 + 1*64 + 0*32 + 0*16 + 1*8 + 1*4 + 0*2 + 1*1$

$= 64 + 8 + 4 + 1$

$= 77$



Binary Data Representation

- Number Systems
  - DEC: Decimal numbers
    - Base 10, digits 0, 1, 2, 3, ..., 9
    - e.g. $157 = 1*10^2 + 5*10^1 + 7*10^0$
  - BIN: Binary numbers
    - Base 2, digits 0, 1
    - e.g. $10011101_2 = 1*2^7 + 0*2^6 + ... + 1*2^0$
  - OCT: Octal numbers
    - Base 8, digits 0, 1, 2, 3, ..., 7
    - e.g. $235_8 = 2*8^2 + 3*8^1 + 5*8^0$
  - HEX: Hexadecimal numbers
    - Base 16, digits 0, 1, 2, 3, ..., 9, A, B, C, ..., F
    - e.g. $9D_{16} = 9*16^1 + 13*16^0$

Arithmetic Operations in C

- Arithmetic Operators
  - parentheses **(,)**
  - unary plus, minus **+, -**
  - multiplication, division, modulo ***, /, %**
  - addition, subtraction **+, -**
  - shift left, shift right **<<, >>**
- Evaluation order of expressions
  - usually left to right
  - by operator precedence
    - ordered as in table above (higher operators are evaluated first)
- Arithmetic operators are available
  - for integer types: all
  - for floating point types: all except **%**, **<<**, **>>**

Shift Operators

- Left-shift operator: **x << n**
  - shifts *x* in binary representation *n* times to the left
  - multiplies *x* *n* times by 2
  - Examples
    - $2x$ = **x << 1**
    - $4x$ = **x << 2**
    - $x * 2^n$ = **x << n**
    - 2^n = **1 << n**
- Right-shift operator: **x >> n**
  - shifts *x* in binary representation *n* times to the right
  - divides *x* *n* times by 2
  - Examples
    - $x / 2$ = **x >> 1**
    - $x / 4$ = **x >> 2**
    - $x / 2^n$ = **x >> n**

Binary Data Representation

- Number Systems

DEC	BIN	OCT	HEX
0	0000	0	0
1	0001	1	1
2	0010	2	2
3	0011	3	3
4	0100	4	4
5	0101	5	5
6	0110	6	6
7	0111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

Binary Data Representation

- Number Systems (signed vs. unsigned)

SDEC	UDEC	BIN	OCT	HEX
0	0	0000	0	0
1	1	0001	1	1
2	2	0010	2	2
3	3	0011	3	3
4	4	0100	4	4
5	5	0101	5	5
6	6	0110	6	6
7	7	0111	7	7
-8	8	1000	10	8
-7	9	1001	11	9
-6	10	1010	12	A
-5	11	1011	13	B
-4	12	1100	14	C
-3	13	1101	15	D
-2	14	1110	16	E
-1	15	1111	17	F

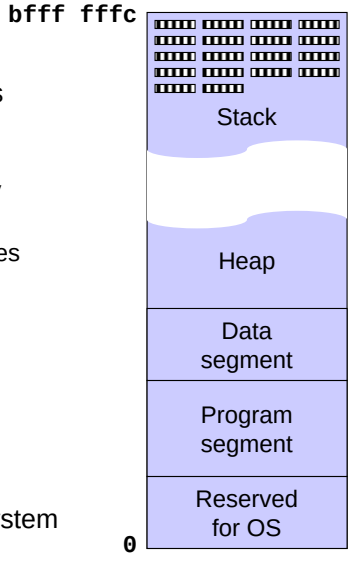
Binary Data Representation

- Number Systems
  - Signed representation: *two's complement*
    - to obtain the negative of any number in binary representation, ...
      - ... invert all bits,
      - ... and add 1
  - Example: 4-bit two's complement

SDEC	UDEC	BIN	OCT	HEX
...	...	...	...	...
7	7	0111	7	7
-8	8	1000	10	8
-7	9	1001	11	9
...	...	...	...	...

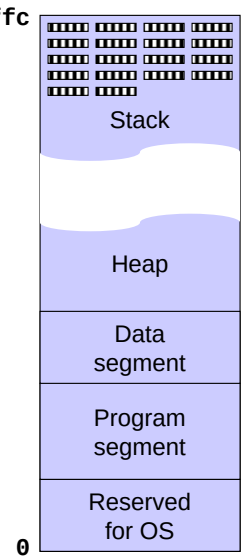
Binary Data Representation

- Memory Segmentation
  - typical (virtual) memory layout on processor with 4-byte words and 1 GB of memory
  - Stack
    - grows and shrinks dynamically
    - function call hierarchy
    - stack frames with local variables
  - Heap
    - "free" storage
    - dynamic allocation by the user
  - Data segment
    - global (and static) variables
  - Program segment
    - stores binary program code
  - Reserved area for operating system



Binary Data Representation

- Memory Segmentation
  - typical (virtual) memory layout on processor with 4-byte words and 1 GB of memory
- Memory errors
  - *Out of memory*
    - Stack and heap collide
  - *Segmentation fault*
    - access outside allocated segments
    - e.g. access to segment reserved for OS
  - *Bus error*
    - mis-aligned word access
    - e.g. word access to an address that is not divisible by 4



EECS10: Computational Methods in ECE, Lecture 21 (c) 2010 R. Doemer 11