ECS 36A, May 19, 2023

Announcements

1. Still grading the midterms; stay tuned
2. Gradescope for the homework problems is up and running!

Converting Hexadecimal to Binary

bit pattern	hex digit	bit pattern	hex digit	bit pattern	hex digit	bit pattern	hex digit
0000	0	0100	4	1000	8	1100	c *or* C
0001	1	0101	5	1001	9	1101	d *or* D
0010	2	0110	6	1010	a *or* A	1110	e *or* E
0011	3	0111	7	1011	b *or* B	1111	f *or* F

So 0x401d2f1b is 0100 0000 0001 1101 0010 1111 0001 1011
Similarly, 0000 0000 0000 0000 0000 0001 0101 1001 is 0x00000159
Easiest way to do this:

- Binary to hexadecimal: group binary digits in sets of 4, starting at the *end*; then use the table to translate
- Hexadecimal to binary: translate each hexadecimal digit to the 4 corresponding digits in the table above and merge them

Decimal to Binary

- Repeatedly divide by 2, then stop when you get 0
- Record the remainders; those are the binary digits
- Example: 345 in decimal:
  - 345 / 2 = 172 r 1
  - 172 / 2 = 86 r 0
  - 86 / 2 = 43 r 0
  - 43 / 2 = 21 r 1
  - 21 / 2 = 10 r 1
  - 10 / 2 = 5 r 0
  - 5 / 2 = 2 r 1
  - 2 / 2 = 1 r 0
  - 1 / 2 = 0 r 1
- So 345 base 10 is 101011001 in base 2

Binary to Decimal

- Each digit is a power of 2, starting from the right (which is 2^0)
- So 101011001 in base 2 is:
  - $1\times2^8 + 0\times2^7 + 1\times2^6 + 0\times2^5 + 1\times2^4 + 1\times2^3 + 0\times2^2 + 0\times2^1 + 1\times2^0 =$
  - $256 + 0 + 64 + 0 + 16 + 8 + 0 + 0 + 1 = 345$

Dealing with Bits: Operation Tables

and (&)	0	1
0	0	0
1	0	1

Examples:

- 10 & 01 = 00
- 11 & 01 = 11

or (|)	0	1
0	0	0
1	0	1

Examples:

- 10 | 01 = 11
- 11 | 01 = 11

xor (^)	0	1
0	0	1
1	1	0

Examples:

- 10 | 01 = 11
- 11 | 01 = 11

not (~)	0	1
	1	0

Examples:

- ~10 = 01
- ~11 = 00

Dealing with Bits: Shift Operations

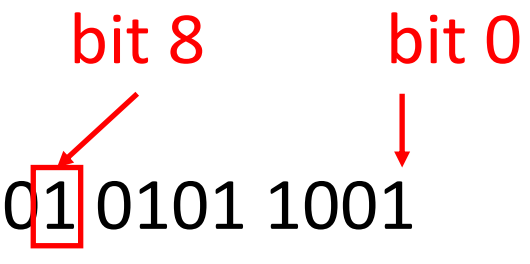
- *b* << *n*: shift *b* left *n* bits
  - Bits shifted beyond the end of the word are discarded
  - 0 bits are inserted at the right
- *b* >> *n*: shift *b* right *n* bits
  - If *b* is signed, the high-order (most significant) bit is propogated
  - If *b* is unsigned, 0 bits are inserted at the left
  - Bits shifted beyond the end of the word are discarded

How to Extract Bits

- Number the bits from 31 to 0
- To get the i-th bit of unsigned int x:

b = (x>>i)&01

Example: 345 in binary:



0000 0000 0000 0000 0000 0001 0101 1001

Extract bit 8:

b = (345>>8)&01 =

(0000 0000 0000 0000 0000 0001 0101 1001>>8)&01 =

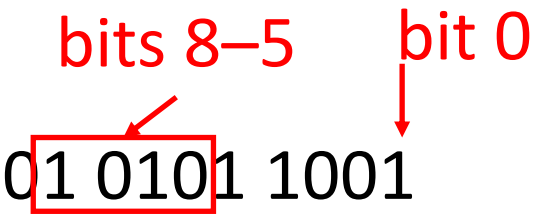
(0000 0000 0000 0000 0000 0000 0000 0001)&01 = 1

How to Extract Groups of Bits

- Number the bits from 31 to 0
- To get the i-th through j-th bits of unsigned int x:

b = (x>>j)&0xZ where Z is the hex representation of i–j bits

Example: 345 in binary:



0000 0000 0000 0000 0000 0001 0101 1001

Extract bits 8 to 5:

b = (345>>5)&0xf = (345>>5)&0xf =

(0000 0000 0000 0000 0000 0001 0101 1001>>5)&0xf =

(0000 0000 0000 0000 0000 0000 0000 0101)&0xf = 0101 = 5

gdb

- To print in binary, use
- `(gdb) print/t x`