Lecture 2: Number Systems

- Binary numbers
- Base conversion
- Arithmetic
- Number systems
  - Sign and magnitude
  - Ones-complement
  - Twos-complement
- Binary-coded decimal (BCD)

Positional number notation

- Bases we will use
  - Binary: base 2
  - Octal: base 8
  - Decimal: base 10
  - Hexadecimal: base 16
- Positional number system
  - $101_2 = 1\times2^2 + 0\times2^1 + 1\times2^0 = 5_{10}$
  - $63.4_8 = 6\times8^1 + 3\times8^0 + 4\times8^{-1} = 51.5_{10}$
  - $A1_{16} = 10\times16^1 + 1\times16^0 = 161_{10}$

Base conversion from binary

- Conversion to octal / hex
  - Binary: 10011110001
  - Octal: 10 | 011 | 110 | 001 = 2361_8
  - Hex: 100 | 1111 | 0001 = $4F1_{16}$
- Conversion to decimal
  - $101_2 = 1\times2^2 + 0\times2^1 + 1\times2^0 = 5_{10}$

Base conversion from decimal

Binary

	Quotient	Remainder
56÷2=	28	0
28÷2=	14	0
14÷2=	7	0
7÷2=	3	1
3÷2=	1	1
1÷2=	0	1

Octal

	Quotient	Remainder
56÷8=	7	0
7÷8=	0	7

$56_{10}=111000_2$

$56_{10}=70_8$

- Why does this work?

Base conversion from decimal

- N = 56_{10} = 111000_2
- Quotient = N / 2 = 56 / 2 = 111000 / 2 = 11100 remainder 0
- Each successive division "liberates" a least significant bit

Arithmetic

- Decimal

```
  11
  1234        5274
+ 5678      − 1638
  6912        3636
```

- Binary

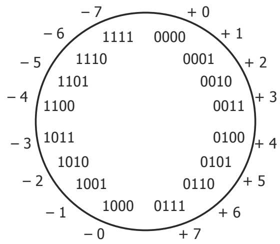
```
 1 1
  1011        1011
+ 1010      − 0110
 10101        0101
```

Negative numbers

- How do we write negative binary numbers?
  - Prefix numbers with minus symbol?
- 3 approaches:
  - Sign and magnitude
  - Ones-complement
  - Twos-complement
- All 3 approaches represent positive numbers in the same way

Sign and magnitude

- Most significant bit (MSB) is the sign bit
  - 0 ≡ positive
  - 1 ≡ negative
- Remaining bits are the number's magnitude



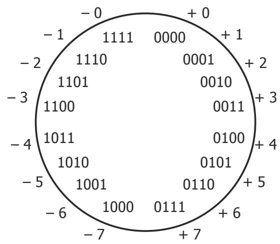
Sign and magnitude

- Problem 1: Two representations of for zero
  - +0 = 0000 and also –0 = 1000
- Problem 2: Arithmetic is cumbersome
  - 4 – 3 != 4 + (-3)

Add		Subtract			Compare and subtract		
4	0100	4	0100	0100	– 4	1100	1100
+ 3	+ 0011	– 3	+ 1011	– 0011	+ 3	+ 0011	– 0011
= 7	= 0111	= 1	≠ 1111	= 0001	– 1	≠ 1111	= 1001

Ones-complement

- Negative number: Bitwise complement of positive number
  - 0111 ≡ 7_{10}
  - 1000 ≡ $–7_{10}$



Ones-complement

- Solves the arithmetic problem

Add		Invert, add, add carry		Invert and add	
4	0100	4	0100	– 4	1011
+ 3	+ 0011	– 3	+ 1100	+ 3	+ 0011
= 7	= 0111	= 1	1 0000	– 1	1110
		add carry:	+1		
			= 0001		

end-around carry

Why ones-complement works

- The ones-complement of an 4-bit positive number y is $1111_2 – y$
  - 0111 ≡ 7_{10}
  - $1111_2 – 0111_2 = 1000_2 ≡ –7_{10}$
- What is 1111_2?
  - 1 less than $10000_2 = 2^4 – 1$
  - $–y$ is represented by $(2^4 – 1) – y$

Why ones-complement works

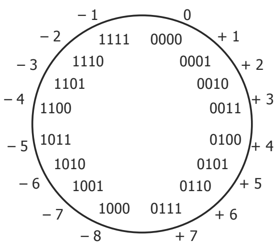
- Adding representations of x and $-y$ where x, y are positive numbers, we get $x + ((2^4 - 1) - y) = (2^4 - 1) + (x - y)$
  - If $x < y$, then $x - y < 0$. There will be no carry from $(2^4 - 1) + (x - y)$. Just add representations to get correct negative number.
  - If $x > y$, then $x - y > 0$. There will be a carry. Performing end-around carry subtracts 2^4 and adds 1, subtracting $(2^4 - 1)$ from $(2^4 - 1) + (x - y)$
  - If $x = y$, then answer should be 0, get $(2^4 - 1) = 1111_2$

So what's wrong?

- Still have two representations for zero!
  - +0 = 0000 and also –0 = 1111

Twos-complement

- Negative number: Bitwise complement **plus one**
  - $0111 \equiv 7_{10}$
  - $1001 \equiv -7_{10}$
- Benefits:
  - Simplifies arithmetic
  - Only one zero!



Twos-complement

Add		Invert and add		Invert and add	
4	0100	4	0100	– 4	1100
+ 3	+ 0011	– 3	+ 1101	+ 3	+ 0011
= 7	= 0111	= 1	1 0001	– 1	1111
		drop carry	= 0001		

Why twos-complement works

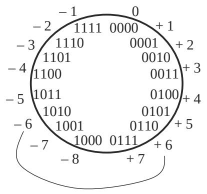
- Recall: The ones-complement of a b-bit positive number y is $(2^b - 1) - y$
- Twos-complement adds one to the bitwise complement, thus, $-y$ is $2^b - y$
  - $-y$ and $2^b - y$ are equal mod 2^b (have the same remainder when divided by 2^b)
  - Ignoring carries is equivalent to doing arithmetic mod 2^b

Why twos-complement works

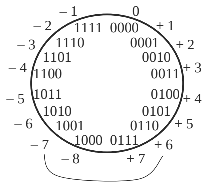
- Adding representations of x and $-y$ where x, y are positive numbers, we get $x + (2^b - y) = 2^b + (x - y)$
  - If there is a carry, that means that $x \geq y$ and dropping the carry yields $x - y$
  - If there is no carry, then $x < y$, then we can think of it as $2^b - (y - x)$, which is the twos-complement representation of the negative number resulting from x – y.

Overflow

- Answers only correct mod 2^b
  - Summing two positive numbers can give a negative result
  - Summing two negative numbers can give a positive result



6 + 4 ⇒ –6



–7 – 3 ⇒ +6

Miscellaneous

- Sign-extension
  - Write +6 and –6 as twos-complement
    - 0110 and 1010
  - Sign-extend to 8-bit bytes
    - **0000**0110 and **1111**1010
- Can't infer a representation from a number
  - 11001 is 25 (unsigned)
  - 11001 is -9 (sign and magnitude)
  - 11001 is -6 (ones complement)
  - 11001 is -7 (twos complement)

BCD (Binary-Coded Decimal)

Decimal Symbols	BCD Code
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001