Lecture 2: Number Systems

- Logistics
  - http://www.cs.washington.edu/370
  - HW1 is posted on the web in the calender --- due 1/14 11:30am
  - Email list: please sign up on the web.
- Last lecture
  - Class introduction and overview
- Today
  - Binary numbers
  - Base conversion
  - Number systems
    - Twos-complement
  - A/D and D/A conversion

The "WHY" slide

- Binary numbers
  - All computers work with 0's and 1's so it is like learning alphabets before learning English
- Base conversion
  - For convenience, people use other bases (like decimal, hexdecimal) and we need to know how to convert from one to another.
- Number systems
  - There are more than one way to express a number in binary. So 1010 could be 10, -2, -5 or -6 and need to know which one.
- A/D and D/A conversion
  - Real world signals come in continuous/analog format and it is good to know generally how they become 0's and 1's (and vice versa).

Digital

- Digital = discrete
  - Binary codes (example: BCD)
  - Decimal digits 0-9
- Binary codes
  - Represent symbols using binary digits (bits)
- Digital computers:
  - I/O is digital
    - ASCII, decimal, etc.
  - Internal representation is binary
    - Process information in bits

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

The basics: Binary numbers

- Bases we will use
  - Binary: Base 2
  - Octal: Base 8
  - Decimal: Base 10
  - Hexadecimal: Base 16
- Positional number system
  - $101_2 = 1\times2^2 + 0\times2^1 + 1\times2^0$
  - $63_8 = 6\times8^1 + 3\times8^0$
  - $A1_{16} = 10\times16^1 + 1\times16^0$
- Addition and subtraction

$$\begin{array}{r} 1011 \\ +\ 1010 \\ \hline 10101 \end{array} \qquad \begin{array}{r} 1011 \\ -\ 0110 \\ \hline 0101 \end{array}$$

Binary → hex/decimal/octal conversion

- Conversion from binary to octal/hex
  - Binary: 10011110001
  - Octal: 10 | 011 | 110 | 001=2361_8
  - Hex: 100 | 1111 | 0001=$4F1_{16}$
- Conversion from binary to decimal
  - $101_2 = 1\times2^2 + 0\times2^1 + 1\times2^0 = 5_{10}$
  - $63.4_8 = 6\times8^1 + 3\times8^0 + 4\times8^{-1} = 51.5_{10}$
  - $A1_{16} = 10\times16^1 + 1\times16^0 = 161_{10}$

Decimal→ binary/octal/hex conversion

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?
  - N=56_{10}=111000_2
  - Q=N/2=56/2=111000/2=11100 remainder 0
- Each successive divide liberates an LSB (least significant bit)

Number systems

- How do we write negative binary numbers?
- Historically: 3 approaches
  - Sign-and-magnitude
  - Ones-complement
  - Twos-complement
- For all 3, the most-significant bit (MSB) is the sign digit
  - 0 ≡ positive
  - 1 ≡ negative
- twos-complement is the important one
  - Simplifies arithmetic
  - Used almost universally

Sign-and-magnitude

- The most-significant bit (MSB) is the sign digit
  - 0 ≡ positive
  - 1 ≡ negative
- The remaining bits are the number's magnitude
- Problem 1: Two representations for zero
  - 0 = 0000 and also –0 = 1000
- Problem 2: Arithmetic is cumbersome

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 positive number
  - 0111 ≡ 7_{10}
  - 1000 ≡ $–7_{10}$
- 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		

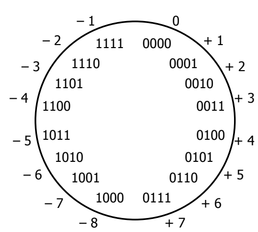
- Remaining problem: Two representations for zero
  - 0 = 0000 and also –0 = 1111

Why ones-complement works

- The ones-complement of an 8-bit positive y is $11111111_2 – y$
- What is 11111111_2 ?
  - 1 less than 1 00000000_2 ≡ 2^8 ≡ 256_{10}
  - So in ones-complement –y is represented by $(2^8 -1) – y$
- Adding representations of x and –y where x, y are positive we get $(2^8 -1) + x – y$
  - If x < y then x - y < 0 there is no carry and get –ve number
    - Just add the representations if no carry
  - If x > y then x - y > 0 there is a carry and get +ve number
    - Need to add 1 and ignore the 2^8, i.e. "add the carry"
  - If x = y then answer should be 0, get 2^8-1 =11111111_2

Twos-complement

- Negative number: Bitwise complement plus one
  - 0111 ≡ 7_{10}
  - 1001 ≡ $–7_{10}$
- Number wheel



- Only one zero!
- MSB is the sign digit
  - 0 ≡ positive
  - 1 ≡ negative

Twos-complement (con't)

- Complementing a complement ➜ the original number
- Arithmetic is easy
  - Subtraction = negation and addition
    - Easy to implement in hardware

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 better

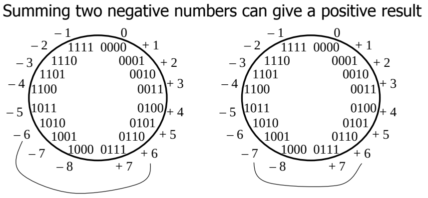
- Recall: The ones-complement of a b-bit positive y is $(2^b-1) - y$
- Adding 1 to get the twos-complement represents –y by $2^b - y$
  - So -y and $2^b - y$ are equal mod 2^b (leave the same remainder when divided by 2^b)
  - Ignoring carries is equivalent to doing arithmetic mod 2^b
- Adding representations of x and –y yields $2^b + x - y$
  - If there is a carry then that means $x \geq y$ and dropping the carry yields x-y
  - If there is no carry then $x < y$ and then we can think of it as $2^b - (y-x)$

Miscellaneous

- Twos-complement of non-integers
  - $1.6875_{10} = 01.1011_2$
  - $-1.6875_{10} = 10.0101_2$
- Sign extension
  - Write +6 and –6 as twos complement
    - 0110 and 1010
  - Sign extend to 8-bit bytes
    - 00000110 and 11111010
- Can't infer a representation from a number
  - 11001 is 25 (unsigned)
  - 11001 is –9 (sign magnitude)
  - 11001 is –6 (ones complement)
  - 11001 is –7 (twos complement)

Twos-complement overflow

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



$6 + 4 \Rightarrow -6$ $\quad$ $-7 - 3 \Rightarrow +6$

- Make sure to have enough bits to handle overflow

BCD (Binary-Coded Decimal) and Gray codes

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

Decimal Symbols	Gray Code
0	0000
1	0001
2	0011
3	0010
4	0110
5	0111
6	0101
7	0100
8	1100
9	1101

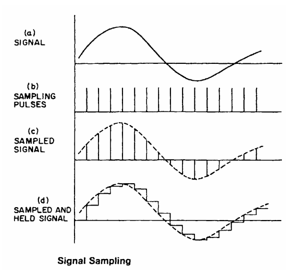
Only one bit changes per step

The physical world is analog

- Digital systems need to
  - Measure analog quantities
    - Speech waveforms, etc
  - Control analog systems
    - Drive motors, etc
- How do we connect the analog and digital domains?
  - Analog-to-digital converter (A/D)
    - Example: CD recording
  - Digital-to-analog converter (D/A)
    - Example: CD playback

Sampling

- **Quantization**
  - Conversion from analog to discrete values
- Quantizing a signal
  - We sample it

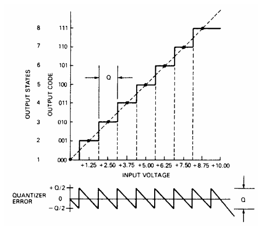


Signal Sampling

Datel Data Acquisition and Conversion Handbook

Conversion

- **Encoding**
  - Assigning a digital word to each discrete value
- Encoding a quantized signal
  - Encode the samples
  - Typically Gray or binary codes



Transfer Function of Ideal 3-Bit Quantizer

Datel Data Acquisition and Conversion Handbook