CDA 3200 Digital Systems

Instructor: Dr. Janusz Zalewski

Developed by: Dr. Dahai Guo

Spring 2012

Outline

- Data Representation
- Binary Codes
- Why 6-3-1-1 and Excess-3?

Data Representation (1/2)

- Each numbering format, or system, has a base, or maximum number of symbols that can be assigned to a single digit.

System	Base	Possible Digits
Binary	2	0, 1
Octal	8	0, 1, 2, 3, 4, 5, 6, 7
Decimal	10	0, 1, 2, 3, 4, 5, 6, 7, 8, 9
Hexadecimal	16	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

Data Representation (2/2)

- Binary: 1 1 1 1 0 1 0 1
- Octal: 365
- Decimal: 245
- Hexadecimal: F5

Binary Numbers (1/7)

- A computer stores instructions and data in memory as collections of electronic charges.
  - 1 = "on" ➔ voltage at output of electronic device is high (saturated).
  - 0 = "off" ➔ voltage at output of electronic device is zero.

Binary Numbers (2/7)

- Each digit (strictly, position of a digit) in a binary number is called a bit.
- In a binary number, bits are <u>usually</u> numbered starting at zero on the right side, and increasing toward the left.
- The bit on the left is called the most significant bit (MSB), and the bit on the right is the least significant bit (LSB).

Binary Numbers (3/7)

1 0 1 1 0 0 1 0 1 0 0 1 1 1 0 0

MSB (points to the leftmost bit) — LSB (points to the rightmost bit)

Binary Numbers (4/7)

- **Unsigned** binary integers
  - Can only be positive or zero
- Translating unsigned binary integers to decimal
  - $dec=(D_{n-1}*2^{n-1})+(D_{n-2}*2^{n-2})+.....+(D_1*2^1)+(D_0*2^0)$
  - 1 1 1 1 0 1 0 1 (n=8)
  - $D_7=1, D_6=1, D_5=1, D_4=1, D_3=0, D_2=1, D_1=0, D_0=1$
  - $dec=2^7+2^6+2^5+2^4+2^2+2^0=128+64+32+16+4+1=245$

Binary Numbers (5/7)

- Translating unsigned decimal integers to binary
  - Example: translating 37 to binary

Division	Quotient	Remainder
37/2	18	1
18/2	9	0
9/2	4	1
4/2	2	0
2/2	1	0
1/2	0	1

The result is 100101.

Binary Numbers (6/7)

- Binary Addition
  - Beginning with the lowest (rightmost) order pair of bits.
  - Proceed bit by bit
  - For each bit pair.

0+0=0	0+1=1
1+0=1	1+1=10

Binary Numbers (7/7)

- Binary addition (cont)

Carry: 1

		0	0	0	0	0	1	0	0	(4)
+		0	0	0	0	0	1	1	1	(7)
		0	0	0	0	1	0	1	1	(11)
Bit position:		7	6	5	4	3	2	1	0	

Value Range

- For an n-bit unsigned binary number, the range is $0 \sim 2^n-1$: 2^n different values.
  - 2 bits: 4 values (0 – 3)
  - 3 bits: 8 values (0 – 7)
  - 4 bits: 16 values (0 – 15)
  - ...
  - 8 bits (a byte): 256 values (0 –255)
  - …
  - 10 bits: 1024 values (0 – 1023)

Hexadecimal Integers (1/6)

- hexadecimal numbers are often used to represent computer memory address and instructions.
- A hexadecimal digit ranges from 0 to 15 (total of sixteen).
- The letters of the alphabet are used to represent 10 through 15.
  - where A=10, B=11, C=12, D=13, E=14, and F=15

Hexadecimal Integers (2/6)

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

- Each hexadecimal digit means a four binary bit string.
- Every four binary bit string can be mapped to a hexadecimal digit.

Hexadecimal Integers (3/6)

- If we can break up a byte (8 bits) into halves, the *upper* and *lower halves*, each half can be represented by a hexadecimal digit.
- A byte could then be represented by two hexadecimal digits, rather than 8 bits.
- In general, any binary number can be split into four-bit groups, starting from right. Each such a group can be translated into hexadecimal digit.
- The result hexadecimal is much shorter than the binary equivalent.

Hexadecimal Integers (4/6)

101101010011110010100

0001	0110	1010	0111	1001	0100
1	6	A	7	9	4

- 101101010011110010100
- 16A794

Note when you are finding four bit groups, begin from the right.

Hexadecimal Integers (5/6)

- Converting unsigned hexadecimal to decimal.
  - $dec=(D_{n-1}*16^{n-1})+(D_{n-2}*16^{n-2})+.....+(D_1*16^1)+(D_0*16^0)$
  - F5➔245
  - $D_1=F, D_0=5$
  - $dec=F*16^1+5*16^0$

    =15*16+5*1

    =240+5

    =245

Hexadecimal Integers (6/6)

- Converting unsigned decimal to hexadecimal

Division	Quotient	Remainder
422/16	26	6
26/16	1	A
1/16	0	1

The result is 1A6.

Signed Integers (1/9)

- Signed integers can negative, zero and positive.
- The most significant bit in binary numbers indicates the number’s sign.
  - 0 means positive or zero
  - 1 means negative
- When you are using signed binary numbers, the number of bits must be specified.

Signed Integers (2/9)

- When a signed binary is positive, it can be used as if it was an unsigned binary.
- When it is negative, two's complement is used the most often.
  - Two's complement (TC) notation works like the negating operation
    - TC(TC(number)) = number, [-(-number)=number]
    - TC(number)+number=0, [-number+number=0]

Signed Integers (3/9)

- Given an <u>eight-bit</u> number 0000 0001, its two's complement is 1111 1111

Starting value	0000 0001
Step1: reverse the bits:	1111 1110
Step 2: add 1	1111 1111

The two's complement representation of -1.

Signed Integers (4/9)

- Two's complement of hexadecimal
  - Reversing a hexadecimal digit is subtracting the digit from 15
  - 6A3D→95C2+1→95C3
  - 95C3→6A3C+1→6A3D

Signed Integers (5/9)

- For a signed hexadecimal number, it is negative if its most significant digit is greater than 7. Otherwise it is zero or positive.

Signed Integers (5/9)

- Converting signed binary to decimal
  - MSB=1, this binary is in two’s complement notation.
    - Get its two’s complement (positive equivalent).
    - Convert to decimal.
    - Make the decimal negative.
  - MSB=0, this binary can be treated as an unsigned binary.

Signed Integers (6/9)

Starting value:	1111 0000
Step1: reverse the bits:	0000 1111
Step2: add 1	0000 1111+1
Step3: its two's complement:	0001 0000
Step4: convert to decimal	16
Step5: make the decimal neg:	-16

Signed Integers (8/9)

- How to
  - Convert signed decimal to binary?
  - Convert signed decimal to hexadecimal?
  - Convert signed hexadecimal to decimal?

Signed Integers (9/9)

- Maximum and Minimum Values:
  - For an n-bit signed binary number, the range is -2^{n-1} – $2^{n-1}-1$

Addition of Signed Binary Numbers (1/3)

- 13-12 (use five bits)
  - =13+(-12)
  - =01101+TC(01100)
  - =01101+(10011+1)
  - =01101+10100
  - =1 00001
  - =00001

The carry from the MSB is discarded. in signed binary number addition.

Addition of Signed Binary Numbers (2/3)

- 13+12 (We still use 5 bits)
  - =01101+01100
  - =11001 The result is negative!! It is an overflow.

Addition of Signed Binary Numbers (3/3)

- -12-13 (Still 5 bits)
  - =TC(12)+TC(13)
  - =TC(01100)+TC(01101)
  - =(10011+1)+(10010+1)
  - =10100+10011
  - =1 00111 The carry from the MSB is discarded.
  - =00111 The result is positive!! It is an overflow.

Binary Codes (1/2)

- Binary codes: how to represent decimal digits.
- Weighted codes
  - BCD codes (8-4-2-1): each decimal digit is represented by its four-bit binary equivalent.
    - 937: <u>1001</u> <u>0011</u> <u>0111</u>
  - 6-3-1-1 codes: weights are 6, 3, 1, 1
    - 937: <u>1100</u> <u>0100</u> <u>1001</u>

Binary Codes (2/2)

- Non-weighted codes
  - Excess 3: obtained from the 8-4-2-1 code by adding 3 (0011) to each the codes.
    - 937: 1100 0110 1010
  - 2-out-of-5: exactly 2 out of 5 bits are 1, has error-checking properties.
    - 937: 11000 01001 10010
  - Gray code: the codes for successive decimal digits differ in exactly one bit
    - 456: 0110 1110 1010

Why Excess-3?

- Excess-3 codes
  - 0 0011
  - 1 0100
  - 2 0101
  - 3 0110
  - 4 0111
  - 5 1000
  - 6 1001
  - 7 1010
  - 8 1011
  - 9 1100

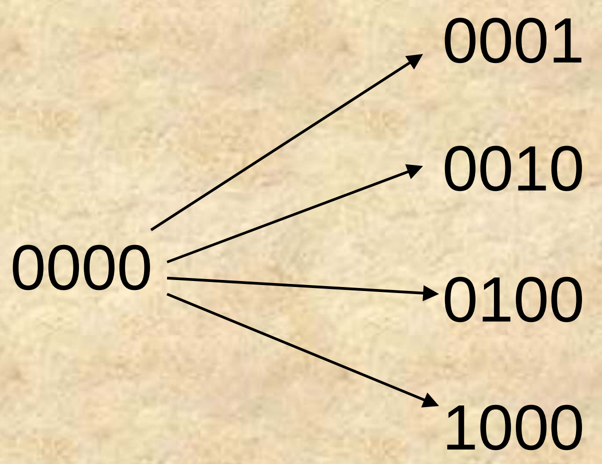
For a decimal digit D, complement its code results in the code of 9-D.

Why 6-3-1-1? (1/3)

- 8-4-2-1 codes
  - 0 0000
  - 1 0001
  - 2 0010
  - 3 0011
  - 4 0100
  - 5 0101
  - 6 0110
  - 7 0111
  - 8 1000
  - 9 1001
- 6-3-1-1 codes
  - 0 0000
  - 1 0001
  - 2 0011
  - 3 0100
  - 4 0101
  - 5 0111
  - 6 1000
  - 7 1001
  - 8 1011
  - 9 1100

Why 6-3-1-1? (2/3)

- Lets consider the situations of 1 bit corrupted.



In 8-4-2-1 coding method, all 0001, 0010, 0100, and 1000 are valid codes.

In 6-3-1-1 coding method, only 0001, 0100, and 1000 are valid codes.

Why 6-3-1-1? (3/3)

- Lets define the concept of error rate at 1 bit corrupted to be the number of possible valid codes after being corrupted divided by 4.
- For example, for 0000, the error rate at 1 bit corrupted is <u>100%</u> when using 8-4-2-1 codes and 7<u>5%</u> when using 6-3-1-1 codes.