ECE 238L
Arithmetic Operations and Codes

August 30, 2006

Binary Addition

$0 + 0 = 0$
$0 + 1 = 1$
$1 + 0 = 1$
$1 + 1 = 0$ *and carry 1 to the next column.*

Examples:

$$\begin{array}{rr} & 0101 \\ + & 0010 \\ \hline & 0111 \end{array} \qquad \begin{array}{r} 5_{10} \\ 2_{10} \\ 7_{10} \end{array} \qquad \begin{array}{rr} & {}^{1\,1\,1} \\ & 0101 \\ + & 0011 \\ \hline & 1000 \end{array} \qquad \begin{array}{r} 5_{10} \\ 3_{10} \\ 8_{10} \end{array}$$

Binary Addition with Overflow

Add 45_{10} and 44_{10} in binary:

```
  1 11   <------- Carries
   101101   (45)
 + 101100   (44)
----------
  1011001
```

If the operands are unsigned, you can use the final carry-out as the MSB (Most Significant Bit) of the result. Adding 2 k-bit numbers may result in a k+1 bit result.

More Binary Addition with Overflow

$$\begin{array}{rrr} & {\scriptstyle 1\,1\,1} & \\ & 1111 & 15_{10} \\ + & 0001 & 1_{10} \\ \hline & 0000 & 0_{10} \end{array}$$

If you don't want a 5-bit result, just keep the lower 4 bits.

Four bits are insufficient to hold the result (16).

So, it rolls back to 0.

Binary Subtraction

Borrows:		00000		00110
Minuend:		10110		10110
Subtrahend:	-	10010	-	10011
Difference:		00100		00011

Again the same as decimal. If the subtrahend is larger than the minuend reverse the two operands and put a minus sign on the result.

	0110			1000
-	1000	$\longrightarrow$	-	0110
				-0010

Multiplication

Multiplicand:		1011		11
Multiplier:	$\times$	101	$\times$	5
		1011		55
		0000		
		1011		
Product:		110111		

$$110111 = 32 + 16 + 4 + 2 + 1 = 55$$

These same approaches apply to hexadecimal as well as octal, you just need to be more careful with your digits.

Binary Arithmetic

- Arithmetic with binary numbers is just like any other numbers
  - digit-by-digit addition
  - carries propagate to next column to left
- Overflow can occur
  - keep the extra bits

    or
  - just keep k bits (roll-over will occur)

Binary Coded Decimal (BCD)

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

Convert 2496_{10} to BCD code:

$$\begin{array}{cccc} 2 & 4 & 9 & 6 \\ \downarrow & \downarrow & \downarrow & \downarrow \\ 0010 & 0100 & 1001 & 0110 \end{array}$$

Not this is very different from converting to binary which yields:

$$100111000000_2$$

In BCD ...

$$0010010010010110$$

BCD Addition

	BCD carry	1	1	
448		0100	0100	1000
+ 489		+ 0100	+ 1000	+ 1001
937	Binary sum	1001	1101	10001
	Add 6		+ 0110	+ 0110
	BCD sum		10011	10111
	BCD result	1001	0011	0111

Add each digit. If the result is greater than 9, add 6 and carry any overflow to the next digit. Repeat.

Why BCD?

- BCD is common in electronic systems which display numeric values.
- Using BCD simplifies manipulation of numerical data for display.
- Each digit is a separate subcircuit, which will be true for a 7 segment display.
- Binary requires a conversion to decimal to determine the digits.

Disadvantages are 1) that addition is more difficult and requires extra circuitry. 10-15%. And 2), using 4 bits to represent 10 values can be expensive.

Binary Codes - ASCII

Character	ASCII Code
c	1 1 0 0 0 1 1
d	1 1 0 0 1 0 0
e	1 1 0 0 1 0 1
f	1 1 0 0 1 1 0
g	1 1 0 0 1 1 1
h	1 1 0 1 0 0 0
I	1 1 0 1 0 0 1
j	1 1 0 1 0 1 0
k	1 1 0 1 0 1 1
l	1 1 0 1 1 0 0
m	1 1 0 1 1 0 1
n	1 1 0 1 1 1 0
o	1 1 0 1 1 1 1
p	1 1 1 0 0 0 0
q	1 1 1 0 0 0 1

Convert "help" to ASCII

h	e	l	p
110100	1100101	1101100	1111000

Gray Codes

Number	Binary	Gray Code
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110
5	0101	0111
6	0110	0101
7	0111	0100
8	1000	1100
9	1001	1101
10	1010	1111
11	1011	1110
12	1100	1010
13	1101	1011
14	1110	1001
15	1111	1000

- Only one bit changes with each number increment
- Not a weighted code
- Useful for interfacing to some physical systems

Gray Codes are Not Unique

Number	Gray Code
0	000
1	001
2	011
3	010
4	110
5	111
6	101
7	100

Number	Gray Code
0	000
1	010
2	110
3	111
4	011
5	001
6	101
7	100

Parity Bits

Word	Even Parity	Odd Parity
1000001	01000001	11000001
1010100	11010100	01010100

Even Parity - number of 1 bits should be even.

Odd Parity - number of 1 bits should odd.

Parity can detect any number of odd errors: 1,3,5,... Parity is also one of the simplest ways to detect errors. Communication protocols commonly include error detection and even correction.

Codes - Summary

- Bits are bits
  - Modern digital devices represent everything as collections of bits
  - A computer is one such digital device
- You can encode anything with sufficient 1's and 0's
  - Text (ASCII)
  - Computer programs (C code, assembly code, machine code)
  - Sound (.wav, .mp3, ...)
  - Pictures (.jpg, .gif, .tiff)