ECE 204 - BINARY NUMBERS AND CODES - INVESTIGATION 8
INTRODUCTION TO BINARY NUMBERS

FALL 2003 **A.P. FELZER**

To do "well" on this investigation you must not only get the right answers but must also do neat, complete and concise writeups that make obvious what each problem is, how you're solving the problem and what your answer is. You also need to include drawings of all circuits as well as appropriate graphs and tables.

As we've seen in the previous Investigations the inputs and outputs of digital circuits can only be 0's or 1's. So if we're going to be able to build digital systems to do things like math calculations and word processing as well as control traffic lights and play CD's then we're going to have to somehow code the information in terms of 0's and 1's. The objective of this Investigation is to introduce binary numbers and show how they can be obtained from decimal numbers.

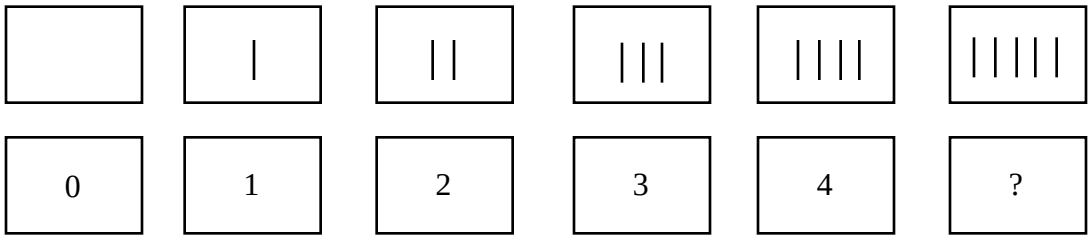
1. Let us as usual begin with a review problem. Use a Karnaugh Map to find the minimum sum of products (SOP) for $F = \sum_{W,X,Y}(1,5) + d(2,4,7)$

2. The genius of our number system is that the value of a number like 273 depends not only on the values of the digits 2, 7 and 3 but also on their positions. This gives us two great results:
   (1) We only need ten symbols 0, 1, . . . , 9 to represent all decimal numbers
   (2) Our numbers are easy to add, subtract, multiply and divide

   Note that we refer to our decimal number system as **base 10** since it contains ten symbols as follows

   0, 1, 2, 3, 4, 5, 6, 7, 8, 9

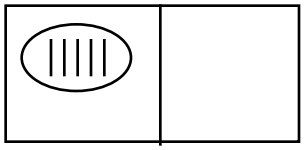
   The objective of this problem is to review the idea of place value by counting sticks in the **base 5** number system as follows



   The "problem" we run into is that after four sticks we run out of symbols because in the base 5 system we only have the five symbols

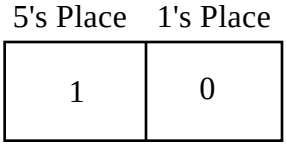
   0, 1, 2, 3, 4

   The way we get out of this dilemna of course is to simply bundle up the five sticks and move them to a new rectangle to the left as follows



   We call the rectangle on the left the 5's place and the rectangle on the right the 1's place. Since we now have one bundle of 5 sticks in the fives place and no sticks in the ones place the

corresponding base 5 number is



$= 10_5$

which we write as 10_5 with a subscript 5 to indicate that the number is base 5. Note that another term for **base** is **radix**.

a. Draw the stick diagram for 20_5
b. Draw the stick diagram for 23_5
c. Draw the stick diagram and write the base 5 number for 8 sticks
d. Draw the stick diagram and write the base 5 number for 12 sticks
e. What's the place value to the left of the 5's place - how many sticks are in each "bundle"

3. From Problem (2) we see how we can use counting to convert base 10 numbers to base 5 numbers. The objective of this problem is to see how to convert base 5 numbers back to base 10. The trick is to simply multiply the digits by their place values as illustrated in the following example.

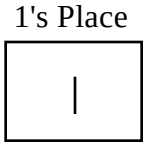
   For the base 5 number 342_5

   3 is in the $5^2 = 25$'s place
   4 is in the $5^1 = 5$'s place
   2 is in the $5^0 = 1$'s place

   and so $342_5 = 3(25) + 4(5) + 2 = 97_{10}$. Now use this algorithm to

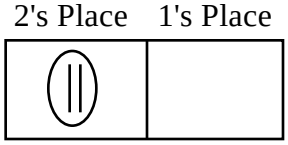
   a. Convert 233_5 to base 10
   b. Convert 412_5 to base 10

4. Now for base 2 binary numbers. Binary numbers have just two symbols 0 and 1. To count sticks in binary we make use of place value just like we do for base 10, base 5 and all other base number systems. After one stick we have



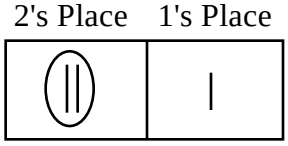
$= 1_2$

After 2 sticks



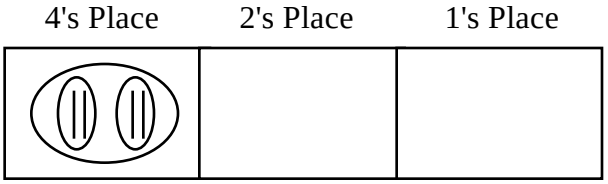
$= 10_2$

After 3 sticks



$= 11_2$

And after 4 sticks we have



$= 100_2$

where the place values from left to right have the values 2^2, 2^1 and 2^0. Now draw the stick diagrams and obtain the corresponding binary numbers for

a. 5 sticks
b. 8 sticks
c. 13 sticks

5. In Problem (4) we saw how to use counting to convert decimal numbers to binary numbers. The objective of this problem is to see how to convert binary numbers back to decimal. But this is straightforward. All we need to do is multiply each binary digit by its place value as follows

$$1101_2 = 1(2^3) + 1(2^2) + 0(2^1) + 1(2^0) = 13_{10}$$

Use this algorithm to convert each of the following binary numbers to decimal

a. 10111_2
b. 11010_2

6. You should be able to convert decimal numbers to binary pretty much by inspection for numbers up to 16. A general algorithm for converting larger decimal numbers to binary involves repeatedly dividing by 2 as indicated in the following example that converts 23 to binary as follows

$$23 = 2\cdot 11 + 1 = 2^1\cdot 11 + 1\cdot 2^0$$
$$2^1\cdot 11 = 2^1(2\cdot 5 + 1) = 2^2\cdot 5 + 1\cdot 2^1$$
$$2^2\cdot 5 = 2^2(2\cdot 2 + 1) = 2^3\cdot 2 + 1\cdot 2^2$$
$$2^3\cdot 2 = 2^3(2\cdot 1 + 0) = 2^4\cdot 1 + 0\cdot 2^3$$
$$2^4\cdot 1 = 2^4(2\cdot 0 + 1) = 2^5\cdot 0 + 1\cdot 2^4$$

And so we have

$$23_{10} = 1\cdot 2^4 + 0\cdot 2^3 + 1\cdot 2^2 + 1\cdot 2^1 + 1\cdot 2^0 = 10111_2$$

More compactly we have

Divide By Two	**Dividend**	**Remainder**
23/2	11	1
11/2	5	1
5/2	2	1
2/2	1	0
1/2	0	1

Memorize this **algorithm**. Then use it to

a. Convert 95 to binary
b. Convert 43 to binary
c. Now use this method to convert 57 to base 5

7. The objective of this problem is to show how to convert decimals to binary. The trick is to simply use negative powers of two for digits to the right of the *binary point* as illustrated in the following example

$$101.11_2 = 1\cdot 2^2 + 0\cdot 2^1 + 1\cdot 2^0 + 1\cdot 2^{-1} + 1\cdot 2^{-2} = 5.75_{10}$$

which is just how negative powers of ten are used for decimal numbers less than one. Now use this result to

   a. Convert 1101.11_2 to base 10
   b. Convert 101.101_2 to base 10

8. To convert a decimal like 0.23 to binary we use the same procedure as in Problem (5) except we repeatedly *Multiply By Two* instead of *Divide By Two* as follows

$$\begin{aligned}
0.23 &= 2^{-1}(2(0.23)) = 2^{-1}(0.46) = 2^{-1}(0.46 + 0) = 2^{-1}(0.46) + 0\cdot 2^{-1}\\
2^{-1}(0.46) &= 2^{-2}(2(0.46)) = 2^{-2}(0.92) = 2^{-2}(0.92 + 0) = 2^{-2}(0.92) + 0\cdot 2^{-2}\\
2^{-2}(0.92) &= 2^{-3}(2(0.92)) = 2^{-3}(1.84) = 2^{-3}(0.84 + 1) = 2^{-3}(0.84) + 1\cdot 2^{-3}\\
2^{-3}(0.84) &= 2^{-4}(2(0.84)) = 2^{-4}(1.68) = 2^{-4}(0.68 + 1) = 2^{-4}(0.68) + 1\cdot 2^{-4}\\
&\ \ \vdots
\end{aligned}$$

   And so we have

$$0.23_{10} = 0\cdot 2^{-1} + 0\cdot 2^{-2} + 1\cdot 2^{-3} + 1\cdot 2^{-4} + \cdots = 0.0011_2\cdots$$

   And so more compactly

Multiply By Two	Integer Part	Decimal
2(0.23)	0	0.46
2(0.46)	0	0.92
2(0.92)	1	0.84
2(0.84)	1	0.68

   Memorize this algorithm. And then use it to

   a. Convert 0.74 to binary to five places past the binary point
   b. Convert 3.38 to five places past the binary point

9. We refer to a single **b**inary dig**it** as a **bit**. And we refer to 8 bits together like the following

$$10110110$$

   as a **byte**. **Memorize** these definitions. Then find

   a. How many bits does it take to represent the decimal 35
   b. How many numbers can be formed with 3 bits
   c. How large a decimal number can be represented by one byte