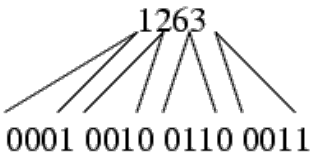
Binary and 8421 Codes

A **BCD** (binary-coded decimal) **code** is a device used to express one of the 10 decimal digits (O through 9) using a nibble of binary digits. Such codes are used extensively in various applications of digital logic, such as encryption de-vices, arithmetic circuits, and so forth, as well as in computer applications that provide error detection and correction. We will briefly explore several of these codes and indicate why one might be preferable over another.

Perhaps the most straightforward method for encoding the decimal digits in binary form is the 8421 BCD code. This is actually the direct representa-tion of the decimal digits in their binary equivalent form and hence is a code for which we know the basis. Note that this is not the same as representing the complete decimal number in its equivalent binary form - far from it! The code translates the individual decimal digits into their binary equivalents, displaying the number in a string of nibbles.

An example of a BCD number is shown below. In the example there are four digits, therefore 16 bits are required. Note that the most significant digit and bits are both on the left hand side. The BCD number is the binary equivalent of each digit.



decimal

BCD

Note: this example shows four digits in two bytes. The hex values would also be 1263.

The representations of the digits are shown below as a reminder: because we are using nibbles to rep-resent the 10 decimal digits and there are $2^4 = 16$ actual nibbles possible, there are six nibbles that are considered invalid in 8421 BCD code.

Decimal Number	8421 BCD

	Representation
1	0000
2	0001
3	0010
4	0011
5	0100
6	0101
7	0110
8	0111
9	1000

8421 BCD equivalents for decimal numbers

Note that when a decimal number is represented in BCD code, the sub-script of 2, denoting the base-2 representation of a decimal number, is not used. That is, if we wanted to represent the decimal number 368 in BCD, we would have the representation as 0011 0110 1000, and there would be no subscript of 2 at the end of the string of nibbles.

A code such as 8421 BCD is referred to as a **weighted code**, which means that the representation of the base-IO digits is performed using a four-bit com-bination for which each position within the nibble carries a particular weight of 8, 4, 2, or 1. That is, when we give a nibble of BCD code, such as 1001, the first digit carries a place weight of 8, the second a place weight of 4, the third a place weight of 2, and the final a place weight of 1. Because this is precisely the man-ner in which the first four places of binary numbers were defined, this terminol-ogy may seem a bit superfluous, but there are other BCD codes, such as 2421 BCD, for which the place weights are not the same as binary representation of decimal numbers. In that code, the nibble 1101 would be the representation of the decimal digit 7 because (in expanded form) it is equivalent to $1 \times 2 + 1 \times 4 + 0 \times 2 + 1 \times 1 = 7$.

In some texts, there is an effort to make explicit the point that a string of nibbles is representative of a decimal number in BCD code by appending a subscript of BCD where one would typically see the base identifier. That is, the BCD representation of the decimal number 368 could be given as $0011\ \ 0110\ \ 1000_{BCD}$, but we will (in this lesson) ensure that the use of BCD representation is clear by context and will not employ that notation

Example: Compare the binary representation of the decimal number 843 to the 8421 BCD code for that number.

Solution:

The binary representation of the decimal number 843 is 11010010112.
The BCD code representation of the decimal number 843 is 1000 0100 0011.

The BCD code representation consists of the coded representations of the individual digits 8, 4, and 3. Referring to our chart or recalling the base-2 representation for each digit, you obtain 1000 0100 0011.

In the following problems, convert the decimal numbers to binary form and then give the 8421 BCD representation of the number.

1. 54

2. 347

3. 6666

4. 23.875

A decimal number encrypted using 8421 BCD code can be decoded into its decimal equivalent by replacing every nibble in the code by the appropriate digit, which we can either obtain from the table given earlier or by translating each nibble individually back to base-10.

Find the decimal number represented by the following BCD codes:

5. 0010 1001 0111 0101.

6. 1001 0111 0101

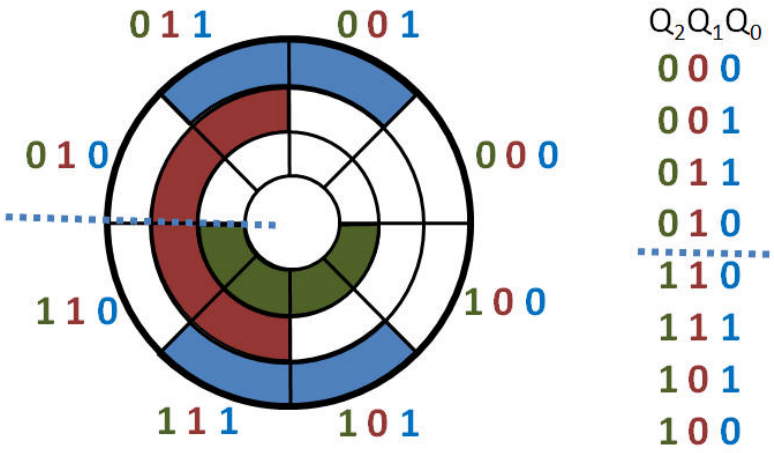
7. 0111 0101 1001 0011 0001

8. 1001.0111 0011

9. 0111 0011.1001

10. Why is the following an invalid BCD code: 0010 1101 0001 0101? (Hint: try converting it to decimal.)

11. Which do you find easier to convert - BCD code or binary? Explain why.



Non-weighted Codes

All the codes discussed so far have been weighted codes, but there certainly are non-weighted codes as well. Two examples of these are Excess-3 (XS-3) code and Gray codes. The former can be used for arithmetic operations, the latter for mechanical switching systems.

Full exploration of these codes would take us yet farther afield, but Gray codes are sufficiently interesting that we will spend a brief time considering them. The code type is named in honor of the creator of its first iteration, **Frank Gray** of Bell Labs. Note that the source of the code type is the name of the inven-tor, and therefore it would be grammatically incorrect to use any of the other possible spellings of the word Gray (such as "grey").

The difficulty with using binary codes applied to switching systems is this: if a device were to indicate position by opening and closing switches, where open and closed are represented by 1 and 0, respectively, then two adjacent positions (O ll and 100, for example) are such that the transition between one position and the next would require the transition of three switches synchronously. In the moment of transition, all three switches will be changing at a slightly dif-ferent rate, and this could spawn misleading readings for an observer. Itwould be highly desirable for successive states to be represented by strings of binary numbers that have minimal differences, such as (in an ideal case) variation of only a single bit. A code capable of

representing a succession of integer values possessing this quality is referred to as a Gray code.

There is no single Gray code; rather, the name applies to any code that represents each number in a sequence of integers (from O through 2n - l, inclusive} as a string of binary digits of length n, such that strings that differ only in one bit represent adjacent integers in the sequence. The rationale for this construction is that advancing through the list of integers requires changing, or flipping, the value of one bit at a time.

One method of constructing a Gray code to represent, for example, the integers from O through 15, inclusive, offers an interesting exercise, which we shall leave to you. One starts with a string of four Os (we use four digits because we know that there are 16 possible strings of four binary digits) and then suc-cessively flip the rightmost bit that produces a completely new string of digits. When we're through, we will (hopefully!) have 16 distinct sets of four binary digits to which we can assign the numbers O through 15. The set will, by con-struction, satisfy the definition of what it means to be a Gray code.

A second method is a process of reflecting and prepending; we will dem-onstrate the process here, since it is less intuitive than the method already de-scribed. The point is that there are many ways of constructing Gray codes rather than a single canonical process.

Our procedure will be to start with the digits O and l, "mirror" those dig- its ("prepend" 0 to the first half and 1 to the last half of digits on the list), and then iterate the process until the desired number of digits (in this case, 16) is obtained. That may sound a bit complicated, but you'll find that it is, in practice, fairly easy.

Step One: Begin with the digits O and 1and mirror that set:

- 0, 1, 1, 0 (note that "mirroring" merely means to repeat the digits in re- verse order)

Step Two: To the first half of strings on that list, prepend O; to the second half of strings on the list, prepend I:

- 00,01, 11, 10

Step Three: Mirror the set once more:

- 00,01, 11, 1 1 11,01,00

Step Fou r : Prepend Os to the first half of strings and ls to the second half of strings:

- 000,001,011,010, 110, 111, 101, 100

Step Five: Mirror the set once more (this will produce a set of 16 strings):

- 000;0010 11,01110,111,101,100,10 101,111, 110,010,011,001,ooo

Step Six: Prepend Os to the first half of strings and ls to the second half of strings:

- 0000, 0001,0011,0010,0110,0111,0101,0100, 1100, 1101, 1111, 1110,

1010, 1011, 1001, 1000

Observe there are no repetitions on the list and that each string differs from its predecessor in a single bit. We could thus make a unique assignment of the decimal numbers from O through 15, inclusive, to these binary strings, in order

1. What is meant by a BCD code?
2. What is meant by a weighted code?

1. What is a Gray code?

In this section, we examined the 8421BCD weighted code, but there exist other weighted codes, such as the 2421BCD code. The development of the code is similar to that of 8421 BCD, but the weights are 2, 4, 2, and 1, respectively. The following are 2421BCD representations of decimal nu:m1ers. Find the decimal numbers with the shown representations.

12. 0001 1011 1110 0011
13. 0101 0001 1111.1100 1101
14. 1111.1110 1011 0010
15. 0000.0011 0100 1110

Write an explanation for the the solution of the following.

16. We have introduced one method for generating a Gray code and sug- gested an alternative approach to represent the integers from O through 15, inclusive. Begin with a string of four Os (we use four digits because we know that there are 16 possible strings of four binary digits) and then successively flip the rightmost bit that produces a completely new string of digits. When we're through, you will have 16 distinct sets of four binary digits to which we can assign the numbers O through 15. Demonstrate this procedure generates a Gray code.