IOSR Journal of Electronics and Communication Engineering (IOSR-JECE)
e-ISSN: 2278-2834,p- ISSN: 2278-8735.Volume 12, Issue 3, Ver. IV (May - June 2017), PP 75-80
www.iosrjournals.org

Implementation of Number Base Conversions using LabVIEW

Mahammad Eliyaz [1], N. Nagaraju[2], N. Madhu[3]
[1,2,3]*(Department of ECE, Vardhaman College of Engineering, India)*

Abstract: *Digital systems have a prominent role in everyday life that we refer to the present technological period as the digital age. The signals in most present day electronic digital systems use just two discrete values and are therefore said to be binary. Discrete elements of information are represented with group of bits called binary codes. The conversion from and to binary, octal, and hexadecimal plays an important role in digital computers. In this paper, an innovative method is introduced for solving number base conversions using LabVIEW. This method mainly uses LabVIEW icons such as For loop, Shift registers and other required icons as per the required base conversion. This method is used to convert the number base systems from any base to any base.*

Keywords: *LabVIEW, Octal, Hexadecimal, For loop, Shift Register*

I. Introduction

Digital devices and digital systems plays very important role in daily life. Digital systems are used in communication, business transactions, traffic control, space guidance, medical treatment, weather monitoring, the internet, and many other commercial, industrial and scientific enterprises. We have digital telephones, digital television, digital versatile discs, digital cameras, hand held devices and of course digital computers. The most striking property of the digital computer is its generality. It can follow a sequence of instructions, called a program that operates on given data. One characteristic of digital systems is their ability to represent and manipulate discrete elements of information called binary data. For a human being it is very difficult to remember the binary data when dealing with large numbers. Hence it is convenient and more efficient for us to write the numbers in octal and hexadecimal rather than binary. However, the digital circuits and systems work strictly in binary, we use octal, hexadecimal for the convenience of the operators of the system. This paper presents an effective way for number base conversions using LabVIEW.

LabVIEW is a graphical programming language that uses icons instead of lines of text to create applications. In contrast to text-based programming languages, where instructions determine program execution, LabVIEW uses dataflow programming, where the flow of data determines execution. In LabVIEW, a user interface is built by using a set of tools and objects. The user interface is known as the front panel. Then code is added using graphical representations of functions to control the front panel objects. The block diagram contains this code. In some ways, the block diagram resembles a flowchart. In this paper, Number base conversions are implemented using LabVIEW. This implementation helps us to perform number base conversions from another base to another base also.

II. Labview Icons Used For Implementation

The different LabVIEW icons which are used to implement the number base conversions are discussed below.

2.1. For loop

For loop repeats part of the block diagram code a predetermined number of times. For loop is a resizable box with two terminals: the count terminal (an input terminal) and the iteration terminal (an output terminal). The count terminal specifies the number of times to execute the loop. The iteration terminal contains the number of times the loop has executed.

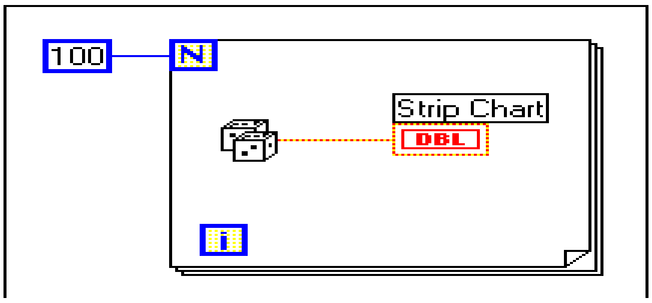


Figure 1: LabVIEW icon for the Structure of For loop

2.2. Shift Register

With While Loops and For Loops, shift registers can be used to transfer values from one iteration to the next. A shift register is created by right-clicking the left or right loop border and selecting Add Shift Register from the shortcut menu. The shift register contains a pair of terminals directly opposite each other on the vertical sides of the loop border.

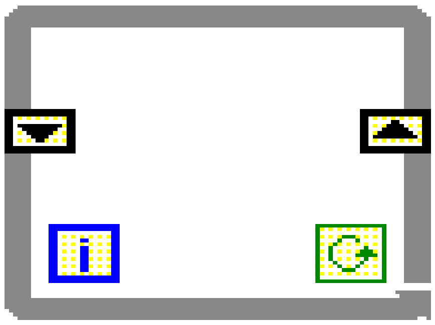


Figure 2: LabVIEW icon for the Structure of Shift Register

The right terminal stores the data on the completion of iteration. Those data are shifted at the end of the iteration and they appear in the left terminal at the beginning of the next iteration. A shift register can hold any data type, such as numeric, boolean, string, array, and so on. The shift register automatically adapts to the data type of the first object wired to the shift register.

2.3. Boolean to (0, 1)

This icon converts a Boolean false or true value to a 16-bit integer with a value of 0 or 1 respectively

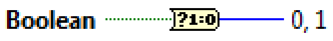


Figure 3: LabVIEW icon for Boolean to (0, 1)

2.4. Scale by power of 2

This icon multiplies X by 2 raised to power of n.

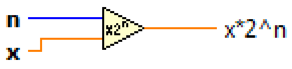


Figure 4: LabVIEW icon for Scale by power of 2

2.5. Power of X

This icon computes X raised to the Y power.

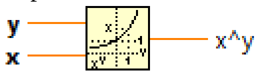


Figure 5: LabVIEW icon for Power of X

2.6. Quotient and Remainder

This icon computes the integer quotient and the remainder of the inputs. This function rounds floor (X/Y) to the nearest integer.

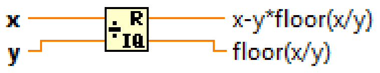


Figure 6: Quotient and Remainder icon

2.7. Reverse 1-D Array

This icon reverses the order of the elements in array, where array is of any type.

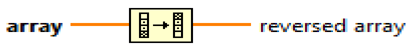


Figure 7: LabVIEW icon for Reverse 1-D array

2.8. Number to Hexadecimal String

This icon converts number to a string of hexadecimal digit at least width characters wide or wider if necessary. The digits A-F always appear in upper case in the output string.

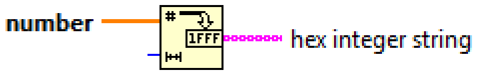
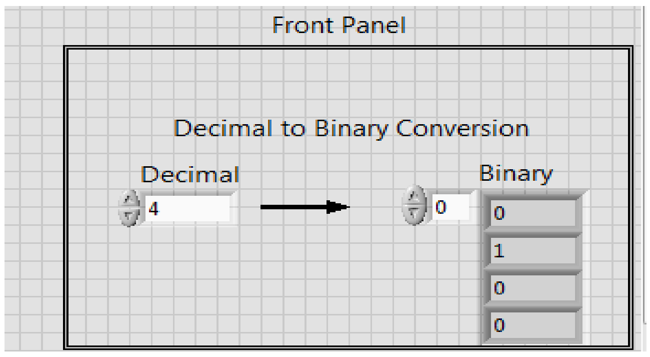


Figure 8: LabVIEW icon for Number to hexadecimal sting

III. Number base conversions

3.1. Decimal to Binary Conversion

In this method, the decimal integer number is converted to the binary integer number by successive division by 2. Here Quotient and remainder icon produces two outputs. One is Quotient and the other is Remainder. The Quotient goes on divided by 2. The Remainder produces the required binary output. The front panel and block diagram of LabVIEW environment is shown in fig.9 below.



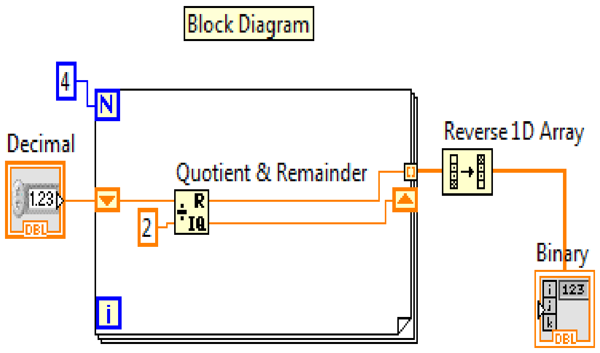
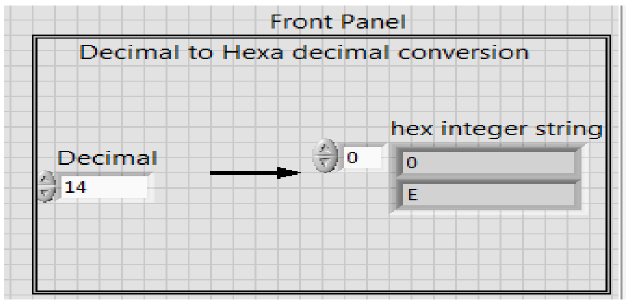


Figure 9: Front panel and block diagram for Decimal to Binary Conversion

3.2. Decimal to Hexadecimal Conversion

In this method, the decimal integer number is converted to the binary integer number by successive division by 16. Here Quotient and remainder icon produces two outputs. One is Quotient and the other is Remainder. The Quotient goes on divided by 16. The Remainder produces the required hexadecimal output. The front panel and block diagram of LabVIEW environment is shown in fig.10 below.



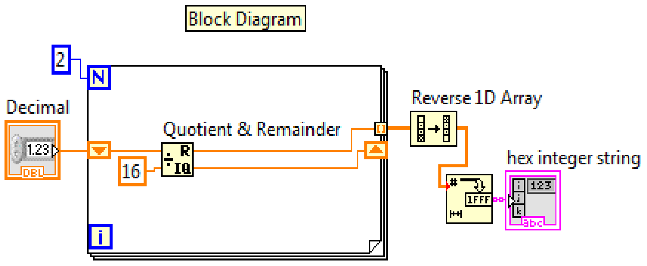
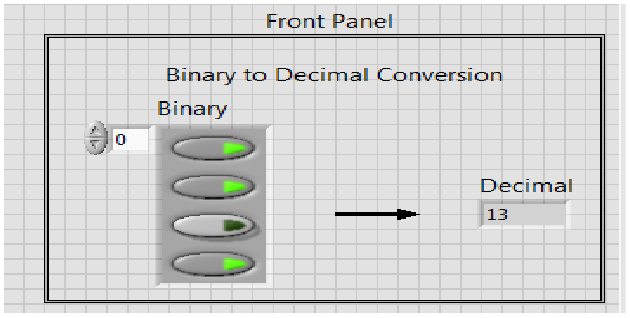


Figure 10: Front panel and block diagram for Decimal to Hexadecimal Conversion

3.3. Binary to Decimal Conversion

Binary numbers may be converted to their decimal equivalents by the positional weights method. In this method, each binary digit of the number is multiplied by its position weight and the product terms are added to obtain the decimal. From the block diagram shown in fig. 11 the positional weights are provided by the scale by power of 2 icon. The front panel and block diagram of LabVIEW environment is shown in fig.11 below.



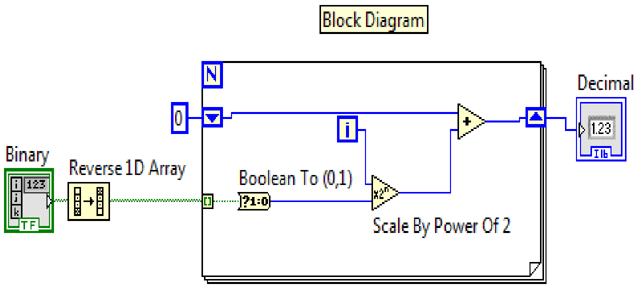


Figure 11: Front panel and block diagram for Binary to Decimal Conversion

3.4. Octal to Decimal Conversion

Octal numbers may be converted to their decimal equivalents by the positional weights method. In this method, each octal digit is multiplied by its position weight and the product terms are added to obtain the decimal. From the block diagram, the positional weights are provided by the x power y icon. The front panel and block diagram of LabVIEW environment is shown in fig.12 below.

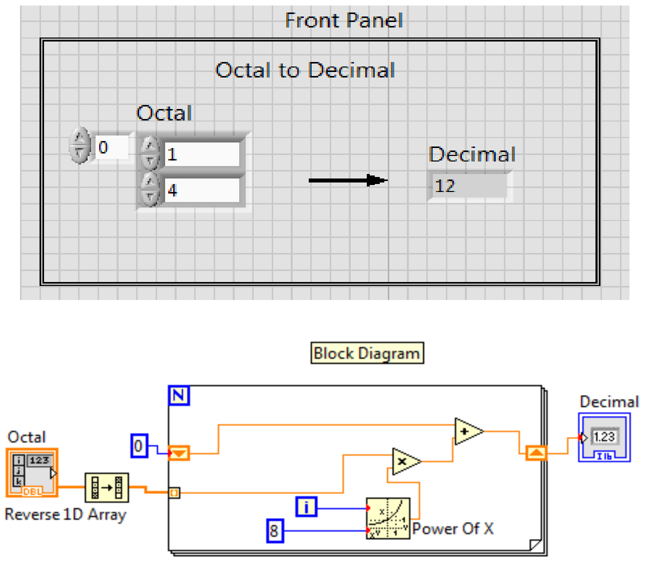


Figure 12: Front panel and block diagram for Octal to Decimal Conversion

3.5. Octal to hexadecimal Conversion

Here Octal to hexadecimal conversion involves two stages. In first stage, octal number is converted to decimal. In the second stage, the obtained decimal number in first stage is converted to hexadecimal. The front panel and block diagram of LabVIEW environment is shown in fig.13 below.

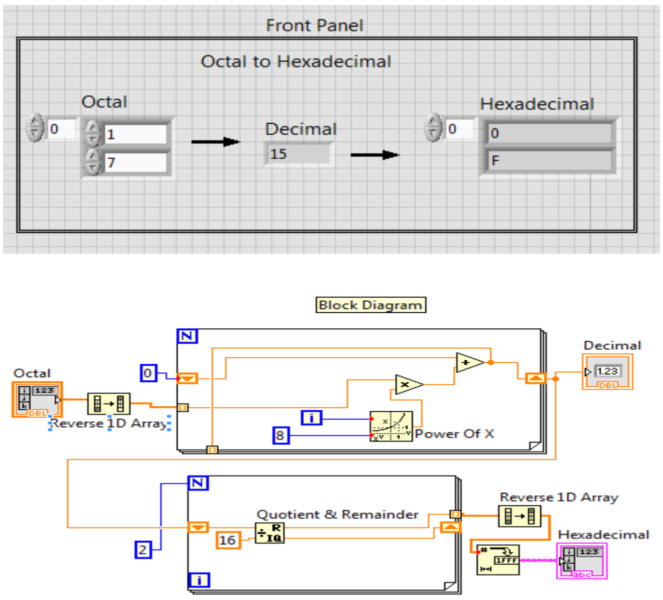


Figure 13: Front panel and block diagram for Octal to Hexadecimal Conversion

3.6. Other Base to other base Conversion

Here other base to other base conversions is also possible. In the following block diagram shown in fig. 14, Base 4 number system is converted to Base 12 number system.

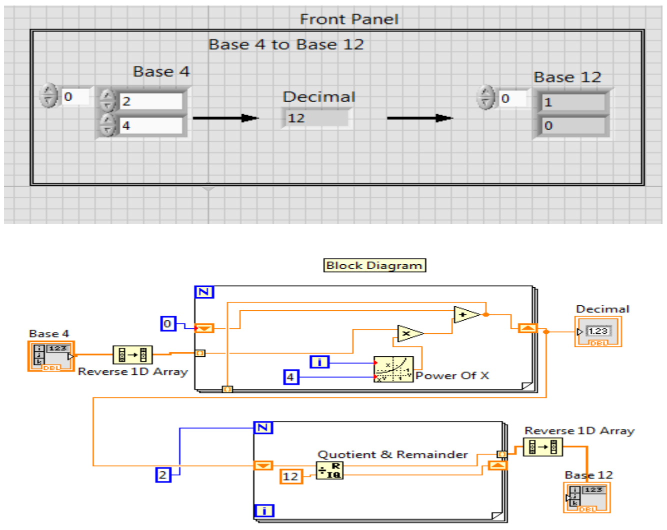


Figure 14: Front panel and block diagram for Base 4 to Base 12 Conversion

IV. Conclusion

Digital systems play very important role in daily life. Since binary numbers used in digital systems are long, it is very difficult for the user to handle. So, there is a need to represent the binary numbers concisely. Hence the number base conversions play a very important role for the convenience of operators. This paper provides an efficient way for basic number system conversions from one base to another base using LabVIEW. This method is also able to perform other base to other base conversions. In this paper six types of conversions have been shown with front panel and block diagram of LabVIEW environment.

References

Journal Papers:

[1]. Nelson Raja joseph, Jaganathan Planichamy, Domnic Sandanam, "Variable Length Integer codes based on radix number system

[2]. conversion", Electronics Letters, Year :2016, Volume :52, Issue :16, pages: 1385-1387.

[3]. Mk. Das, J.N. Roy, T. Chattopadhyay, "Passive all optical tree architecture based scheme for conversion of 2^n radix-based number to

[4]. its binary form", IET Opto Electronics, Year: 2012, Volume :6, Issue :5, pages: 223-229.

[5]. Robert Todd Gregory, David W. Matula, "Base Conversion in residue number systems," 1975 IEEE 3rd symposium on computer

[6]. arithmetic, pp. 117-125.

Books:

[7]. A. Anand Kumar, "*Switching Theory and Logic Design*," 2008 by PHI learning private Limited, New Delhi.