DIGITAL TECHNIQUES

BARRY G. WOOLLARD

Digital techniques

Barry G. Woollard M Phil, C Eng, MIERE, M Inst MC
Technical Director, Beal-Davis Electronics Limited

McGraw-Hill Book Company (UK) Limited

London · New York · St Louis · San Francisco · Auckland · Bogotá
Guatemala · Hamburg · Johannesburg · Lisbon · Madrid · Mexico · Montreal
New Delhi · Panama · San Juan · São Paulo · Singapore · Sydney · Tokyo
Toronto

Published by
McGRAW-HILL Book Company (UK) Limited
MAIDENHEAD • BERKSHIRE • ENGLAND

British Library Cataloguing in Publication Data

Woollard, Barry G.
Digital techniques.
1. Digital electronics
I. Title
621.3815 TK7868.D5

ISBN 0-07-084669-3

Library of Congress Cataloging in Publication Data

Woollard, Barry G.
Digital techniques.

Includes index.
1. Digital electronics. I. Title.
TK7868.D5W63 1983 621.3819'58 82-20909
ISBN 0-07-084669-3

12345 JWA 86543

Printed in Great Britain at the Alden Press, Oxford

Preface

Developments in the technology of integrated circuits during the last decade have caused major changes to be made in the field of digital electronics and microcomputing. These advances have led to the widespread use of electronic calculators, digital watches, and electronic games in the social and domestic field, and to digital multimeters (DMMs), digital frequency counters, digital panel meters (DPMs), direct digital control of industrial processes, programmable logic control (PLC), and the microcomputer—enabling industrial control applications to be solved at costs which can no longer be ignored.

This text is written to meet all of the objectives of the TEC Units: Digital Techniques 2, Digital Techniques 3A, and Digital Techniques 3B. In addition to the TEC requirements, the text is useful for a wide range of technicians and engineers who are interested in gaining some expertise in the use of digital electronics.

In the field of Digital Techniques, and in Microcomputing, it is *essential* that the normal learning process is complemented by 'hands-on' practical development. This text is prepared so that the diagrams can be interpreted as 'practical circuits'; a Digital Trainer is manufactured by Beal-Davis Electronics Ltd., 18A Friar Street, Worcester. This equipment is designed to provide all the necessary power supplies, binary signals, pulse generators, and monitoring capability required to satisfy all of the objectives in this text.

Many examples are included in the text, each chapter concluding with exercises. Answers to numerical exercises are included in the Appendices.

I sincerely acknowledge my gratitude to all those who have contributed to the realization of this book, in particular to my wife, for her patience and efforts during the preparation and typing of the manuscript.

Barry G. Woollard
Longdon, Staffs.

Contents

1 Binary arithmetic

1.1 Numbering systems

Many different numbering systems are in common use. These have evolved out of the need to simplify the human operator's task in reading and writing data (information) and coded instructions to digital electronic control equipment and to computer systems. A *digital code* is a system of symbols that represent data values and make up a special 'language' that permits information to be communicated, stored, and manipulated by digital circuits and computers.

The number of characters (digits, symbols) used in a system is known as its *base*, or *radix*. Other numbers are 'constructed' by giving different 'weights' to the position of the digit relative to the 'decimal' point. The 'weights' of the different positions are given by powers of the radix, which, in general is:

$$R^4 \quad R^3 \quad R^2 \quad R^1 \quad R^0 \quad \cdot \quad R^{-1} \quad R^{-2} \quad R^{-3} \quad R^{-4}$$

(a) *Denary numbering system*. Although the denary system is very widely understood, it is helpful to examine it here to appreciate the 'construction' technique used, so that numbering systems which are unfamiliar can be better understood.

The radix is 10, i.e., *ten symbols* are used to represent the quantities 0 (zero) through 9 (nine). Positional weights are powers of 10:

$$10^4 \quad 10^3 \quad 10^2 \quad 10^1 \quad 10^0 \quad \cdot \quad 10^{-1} \quad 10^{-2}$$

EXAMPLE 1.1

$$\begin{aligned}2345.63_{10} &= 2\times10^3+3\times10^2+4\times10^1+5\times10^0+6\times10^{-1}+3\times10^{-2}\\ &= 2\times1000+3\times100+4\times10+5\times1+6\times0.1+3\times0.01\\ &= 2000+300+40+5+0.6+0.03\\ &= 2345.63_{10}\end{aligned}$$

(b) *Binary numbering system*. This system is very widely used in logic and computing, and is the basis of machine code language used with computers.

The radix is 2, i.e., *two symbols* are used to represent the quantities 0 and 1. Positional weights are powers of 2:

$$2^7 \quad 2^6 \quad 2^5 \quad 2^4 \quad 2^3 \quad 2^2 \quad 2^1 \quad 2^0 \quad \cdot \quad 2^{-1} \quad 2^{-2} \quad 2^{-3} \quad 2^{-4}$$

EXAMPLE 1.2

$$\begin{aligned} 11101011\cdot101_2 &= 1 \times 2^7 + 1 \times 2^6 + 1 \times 2^5 + 1 \times 2^3 + 1 \times 2^1 + 1 \times 2^0 \\ &\quad + 1 \times 2^{-1} + 1 \times 2^{-3} \\ &= 128 + 64 + 32 + 8 + 2 + 1 + 0.5 + 0.125 \\ &= 235.625_{10} \end{aligned}$$

(c) *Octal numbering system.* This system has been widely used in computer systems, due to the ease with which it can be converted to binary and vice versa. Octal was used in preference to binary, since a given number can be represented by fewer digits than are required in binary.

The radix is 8, i.e., *eight symbols* are used to represent the quantities 0 through 7. Positional weights are powers of 8:

$$8^4 \quad 8^3 \quad 8^2 \quad 8^1 \quad 8^0 \quad \cdot \quad 8^{-1} \quad 8^{-2} \quad 8^{-3}$$

EXAMPLE 1.3

$$\begin{aligned} 2345.63_8 &= 2 \times 8^3 + 3 \times 8^2 + 4 \times 8^1 + 5 \times 8^0 + 6 \times 8^{-1} + 3 \times 8^{-2} \\ &= 2 \times 512 + 3 \times 64 + 4 \times 8 + 5 \times 1 + 6 \times 0.125 + 3 \times 0.015625 \\ &= 1024 + 192 + 32 + 5 + 0.75 + 0.046875 \\ &= 1253.796875_{10} \end{aligned}$$

(d) *Hexadecimal (hex) numbering system.* This system has become much more widely used than octal, due to the advances made in the development of microcomputer systems. *Hex* has replaced the use of octal in most microcomputers, due to the reduced number of digits necessary to represent any particular binary number.

The radix is 16, i.e., *sixteen symbols* are used to represent the quantities 0, 1, 2, 3, 4, 5, 6. 7, 8, 9, A, B, C, D, E, F:

Decimal	*Hexadecimal*
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
10	A

11	B
12	C
13	D
14	E
15	F

In general:

$$16^4 \quad 16^3 \quad 16^2 \quad 16^1 \quad 16^0 \quad \cdot \quad 16^{-1} \quad 16^{-2} \quad 16^{-3}$$

EXAMPLE 1.4

$$\begin{aligned}
\text{B6EF}\cdot\text{4AH} &= \text{B} \times 16^3 + 6 \times 16^2 + \text{E} \times 16^1 + \text{F} \times 16^0 + 4 \times 16^{-1} + \text{A} \times 16^{-2} \\
&= 11 \times 4096 + 6 \times 256 + 4 \times 16 + 15 \times 1 + 4 \times 0.0625 \\
&\quad + 10 \times 0.00390625 \\
&= 45056 + 1536 + 224 + 15 + 0.25 + 0.0390625 \\
&= 46831.2890625_{10}
\end{aligned}$$

Hex-
radix 16_{10}

(e) *Binary coded decimal (BCD).* The binary number system is the simplest and best system for logic circuits and digital computers. However, the denary number system is the most familiar world-wide. Hence, for digital logic circuits and computers to be able to work in the binary system we must have a simple method of converting binary to denary and vice versa. The conventional method using powers of 2 is awkward, and although computers can be instructed to perform the conversions, human operators find it time-consuming to convert long strings of binary digits into a denary number.

The octal and hexadecimal systems are shorthand methods of writing binaries—but they are not a great deal of help in converting them to denary.

To overcome these problems, several binary codes have been devised to *translate each* denary digit into an *equivalent 4-bit binary code* and vice versa, some of which are shown in Fig. 1.1.

After a count of 9_{10}, the BCD *weights* change by a factor equivalent to denary 10.

Denary	Excess-3 (XS-3)	8421 BCD	2421 BCD	7421 BCD
0	0011	0000	0000	0000
1	0100	0001	0001	0001
2	0101	0010	0010	0010
3	0110	0011	0011	0011
4	0111	0100	0100	0100
5	1000	0101	0101	0101
6	1001	0110	0110	0110
7	1010	0111	0111	1000
8	1011	1000	1110	1001
9	1100	1001	1111	1010

Fig. 1.1 Binary coded decimal (BCD) systems.

EXAMPLE 1.5

The denary number 79_{10} may be represented in 8421 BCD as follows:

```
                    0111    1001

that is,       80  40  20  10  8  4  2  1

                0   1   1   1  1  0  0  1
```

1.2 Conversions between numbering systems

Digital electronic circuits, microprocessor-based systems, and computers work in binary numbers and binary digits (*bits*). Information is often fed into the above systems in several different forms–including binary, octal, hexadecimal, BCD, alphanumeric, etc.–and it is therefore necessary to be able to convert from one numbering system to another.

Many rules have been established for these conversions, but the following simple procedures can easily be mastered with a little practice:

(a) *Denary-to-binary conversion*. The binary equivalent of a denary number may be found simply by subtracting the value of the highest power of 2–the *most significant digit*, or *bit* (MSB)–from the denary number and writing a 1 in that position. The procedure is repeated on the remainder, progressively, until all binary digits have been formed.

EXAMPLE 1.6

Convert 157_{10} to binary.

```
157 –
128     = highest power of 2 ∴ MSB  = 1
 29 –     no 64, next significant bit = 0
          no 32, next significant bit = 0
 16 –        16, next significant bit = 1
 13 –
  8           8, next significant bit = 1
  5 –
  4           4, next significant bit = 1
  1           2, next significant bit = 0
  1           1, least significant bit = 1
  0
```

The binary equivalent of 157_{10} is therefore 10011101_2.

When large denary numbers are to be converted, the above method can lead to errors, and an alternative is continuously to divide by 2 while noting the remainder.

EXAMPLE 1.7

Convert 157_{10} to binary.

2)157		
2) 78	remainder 1	(LSB)
2) 39	remainder 0	
2) 19	remainder 1	
2) 9	remainder 1	
2) 4	remainder 1	
2) 2	remainder 0	
2) 1	remainder 0	
0	remainder 1	(MSB)

The binary equivalent of 157_{10} is therefore 10011101_2.

The above examples show how whole numbers (*integers*) may be converted. When the number to be converted is a *compound* number, i.e., it has a fractional part as well as an integer part, the conversion must be performed in two stages. The integer part is converted separately, as above. Then the fractional part is converted by continuously multiplying the fractional part by 2, the radix of the binary system; the integer part of each multiplication gives the binary digits of the solution.

EXAMPLE 1.8

Converting 25.875 to binary.

Dealing with integer part (25_{10}) first:

2)25		
2)12	remainder 1	(LSB)
2) 6	remainder 0	
2) 3	remainder 0	
2) 1	remainder 1	
0	remainder 1	(MSB)

$$\therefore 25_{10} = 11001_2$$

Secondly, deal with the fractional part (0.875_{10})

$$\begin{tikzcd}
\begin{array}{l} 0.875 \times \\ \underline{2\quad\;\;} \\ 1.750 \end{array} \arrow[r] \arrow[d] & \begin{array}{l} 0.75 \times \\ \underline{2\quad\;\;} \\ 1.50 \end{array} \arrow[r] \arrow[d] & \begin{array}{l} 0.5 \times \\ \underline{2\;\;} \\ 1.0 \end{array} \arrow[d] \\
1\ (\text{MSB}) & 1 & 1\ (\text{LSB})
\end{tikzcd}$$

$\therefore 0.875_{10} = 0.111_2$

Thus, $25.875_{10} = 11001.111_2$.

EXAMPLE 1.9

Convert 158.85_{10} to binary.

Considering the integer part (158_{10}) first:

$$\begin{array}{rl}
2\,\underline{)158} & \\
2\,\underline{)\ 79} & \text{remainder } 0 \ = \ \text{LSB} \\
2\,\underline{)\ 39} & \text{remainder } 1 \\
2\,\underline{)\ 19} & \text{remainder } 1 \\
2\,\underline{)\ \ 9} & \text{remainder } 1 \\
2\,\underline{)\ \ 4} & \text{remainder } 1 \\
2\,\underline{)\ \ 2} & \text{remainder } 0 \\
2\,\underline{)\ \ 1} & \text{remainder } 0 \\
0 & \text{remainder } 1 \ = \ \text{MSB}
\end{array}$$

∴ the binary equivalent of $158_{10} = 10011110_2$.

Secondly, dealing with the fractional part:

$$\begin{array}{ccccc}
0.85\times & 0.7\times & 0.4\times & 0.8\times & 0.6\times \\
2 & 2 & 2 & 2 & 2 \\
\hline
1.70 & 1.4 & 0.8 & 1.6 & 1.2 \\
\downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\
1\ (\text{MSB}) & 1 & 0 & 1 & 1\ (\text{LSB})
\end{array}$$

∴ the binary equivalent of 0.85_{10} is 0.11011_2 to five binary places.

Thus, the binary equivalent of 158.85_{10} is 10011110.11011_2, which is accurate to *five* binary places.

An error is obviously produced in this conversion. This is, of course, always possible and in this case *a true conversion cannot be achieved.* Greater accuracy can be obtained by extending the number of binary places.

(b) *Binary-to-denary conversion.* The denary equivalent of a binary number is most easily determined by adding the equivalent values (or weights) of those digits where a 1 exists:

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	•	2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5}
128	64	32	16	8	4	2	1		0.5	0.25	0.125	0.0625	0.03125

EXAMPLE 1.10

Convert 10011101_2 to denary.

$$10011101_2 = 128 + 16 + 8 + 4 + 1$$

$$= 157_{10}$$

Check with Example 1.7.

EXAMPLE 1.11

Convert 11001.111_2 to denary.

$$\begin{aligned} 11001.111_2 &= 16+8+1+0.5+0.25+0.125 \\ &= 25.875_{10} \end{aligned}$$

Check with Example 1.8.

EXAMPLE 1.12

Convert 10011110.11011_2 to denary

$$\begin{aligned} 10011110.11011_2 &= 128+16+8+4+2+0.5+0.25+0.0625+0.03125 \\ &= 158.84375_{10} \end{aligned}$$

Note: If this is checked with Example 1.9 it is possible to determine the error in the denary-to-binary conversion.

(c) *Denary-to-octal conversion*. Similar techniques to denary-to-binary conversion can be applied here. For this conversion, however, we must continuously divide by 8 and note the remainder.

EXAMPLE 1.13

Convert 157_{10} to octal.

$$\begin{array}{r l} 8\,\underline{)157} & \\ 8\,\underline{)\ 19} & \text{remainder } 5 = \text{LSD} \\ 8\,\underline{)\ \ 2} & \text{remainder } 3 \\ 0 & \text{remainder } 2 = \text{MSD} \end{array}$$

∴ the octal equivalent of 157_{10} is 235_8.

EXAMPLE 1.14

Convert 25.875_{10} to octal.

Converting the whole number (integer) part first:

$$\begin{array}{r l} 8\,\underline{)25} & \\ 8\,\underline{)\ 3} & \text{remainder } 1 \ \text{LSD} \\ 0 & \text{remainder } 3 \ \text{MSD} \end{array}$$

∴ the octal equivalent of 25_{10} is 31_8.

Converting the fractional part secondly:

$$\begin{array}{l l} 0.875\ \times & \\ \underline{8\qquad} & \\ 7.000 & \text{no further remainder} \\ \downarrow & \\ \text{MSD} & \end{array}$$

∴ the octal equivalent of 0.875_{10} is 0.7_8 and the octal equivalent of 25.875_{10} is 31.7_8.

EXAMPLE 1.15

Convert 158.85_{10} to octal.

Converting the integer part first:

$$\begin{array}{rl}
8\,\underline{)158} & \\
8\,\underline{)\;\;19} & \text{remainder } 6 = \text{LSD} \\
8\,\underline{)\;\;\;\;2} & \text{remainder } 3 \\
0 & \text{remainder } 2 = \text{MSD}
\end{array}$$

∴ the octal equivalent of 158_{10} is 236_8.

Considering the fraction part:

$$\begin{tikzcd}
0.85 \times & 0.8 \times & 0.4 \times & 0.2 \times & 0.6 \times & 0.8 \times \\
\underline{8} & \underline{8} & \underline{8} & \underline{8} & \underline{8} & \underline{8} \\
6.80 \arrow[d] \arrow[ur] & 6.4 \arrow[d] \arrow[ur] & 3.2 \arrow[d] \arrow[ur] & 1.6 \arrow[d] \arrow[ur] & 4.8 \arrow[d] \arrow[ur] & 6.4 \arrow[d] \\
6\ (\text{MSD}) & 6 & 3 & 1 & 4 & 6\ (\text{LSD})
\end{tikzcd}$$

Therefore, the octal equivalent of 0.85_{10} is 0.663146_8 to six octal places. Note again that a true conversion cannot be achieved. Thus, the octal equivalent of 158.85_{10} is 236.663146_8 to six octal places.

(d) *Octal-to-denary conversion*. The denary equivalent of an octal number is most easily determined by multiplying each digit by its value of power of 8, and then adding:

8^3	8^2	8^1	8^0	·	8^{-1}	8^{-2}	8^{-3}	8^{-4}	8^{-5}
512	64	8	1		0.125	0.015625	0.0019531	0.0002441	0.00003051

EXAMPLE 1.16

Convert 31.7_8 to denary.

$$\begin{aligned}
31.7_8 &= 3 \times 8^1 + 1 \times 8^0 + 7 \times 8^{-1} \\
&= 3 \times 8 + 1 \times 1 + 7 \times 0.125 \\
&= 24 + 1 + 0.875 \\
&= 25.875_{10}
\end{aligned}$$

(e) *Octal-to-binary conversion*. Since 8 (the octal radix) = 2^3, each single octal digit can be represented by a 3-bit binary group. The conversion of an octal number to a binary number is achieved by writing the 3-bit binary equivalent of each octal digit.

EXAMPLE 1.17

Convert 235_8 to binary.

2	3	5_8
010	011	101_2

Thus, the binary equivalent of 235_8 is 10011101_2.

EXAMPLE 1.18

Convert 31.7_8 to binary.

3	1.	7_8
011	001.	111_2

Thus, the binary equivalent of 31.7_8 is 11001.111_2.

(f) *Binary-to-octal conversion*. The binary number is divided into 3-bit groups, working from the binary point (either to the left or to the right).

EXAMPLE 1.19

Convert 10011101_2 to octal.

10	011	101_2
2	3	5_8

Thus, the octal equivalent of 10011101_2 is 235_8.

EXAMPLE 1.20

Convert 11001.111_2 to octal.

11	001.	111_2
3	1.	7_8

Thus, the octal equivalent of 11001.111_2 is 31.7_8.

(g) *Denary-to-hex conversion*. This is achieved by continuously dividing by 16 (the radix of the hexadecimal system) and noting the remainder.

EXAMPLE 1.21

Convert 157_{10} to hexadecimal.

```
16 )157
16 )  9   remainder D =  LSD
      0   remainder 9 =  MSD
```

Thus, the hex equivalent of 157_{10} is 9DH.

EXAMPLE 1.22

Convert 25.875_{10} to hexadecimal.

Converting the integer part first:

```
16 )25
16 ) 1    remainder 9 =  LSD
     0    remainder 1 =  MSD
```

The hex equivalent of 25_{10} is 19H.

Converting the fractional part:

$$\begin{array}{l} 0.875 \times \\ \underline{16} \\ 14.000 \quad \text{no further digits} \\ \downarrow \\ \text{E} \quad \text{(MSD)} \end{array}$$

The hex equivalent of 0.875_{10}, is 0.EH and the hex equivalent of 25.875_{10} is 19.EH.

(h) *Hex-to-denary conversion*. The denary equivalent of a hex number is most easily determined by multiplying each digit by its value of power of 16 and then adding:

16^1	16^0 •	16^{-1}	16^{-2}
16	1	0.0625	0.00390625

EXAMPLE 1.23

Convert 9DH to denary.

$$\begin{aligned} 9\text{DH} &= 9 \times 16^1 + \text{D} \times 16^0 \\ &= 9 \times 16 + 13 \times 1 \\ &= 144 + 13 \\ &= 157_{10} \end{aligned}$$

EXAMPLE 1.24

Convert 19.EH to denary.

$$\begin{aligned} 19.\mathrm{EH} &= 1 \times 16^1 + 9 \times 16^0 + \mathrm{E} \times 16^{-1} \\ &= 1 \times 16 + 9 \times 1 + 14 \times 0.0625 \\ &= 25.875_{10} \end{aligned}$$

(i) *Hex-to-binary conversion*. Since 16 (the hex radix) = 2^4, each single hex digit can be represented by a 4-bit binary group. The conversion of a hex number to a binary number is achieved by writing the 4-bit binary equivalent of each hex digit.

EXAMPLE 1.25

Convert 9DH to binary.

9	D	H
1001	1101_2	

Thus, the binary equivalent of 9DH is 10011101_2.

EXAMPLE 1.26

Convert 19.EH to binary.

1	9.	E	H
0001	1001.	1110_2	

Thus, the binary equivalent of 19.EH is 11001.111_2.

(j) *Binary-to-hex conversion*. The binary number is divided into 4-bit groups, working from the binary point (either to the left or to the right).

EXAMPLE 1.27

Convert 10011101_2 to hex.

1001	1101_2	
9	D	H

Thus, the hex equivalent of 10011101_2 is 9DH.

EXAMPLE 1.28

Convert 11001.111_2 to hex.

0001	1001	.	1110_2	
1	9	.	E	H

Thus, the hex equivalent of 11001.111_2 is 19.EH.