

Arithmetic Operations Based on Imitation of Handwriting Calculation

Hsin-Kun Lai

*Depart. of Electronic Engineering, Chin Min Institute of Technology
No.110 Syuefu Rd. Toufen Township, Miaoli County, Taiwan351, R.O.C.
larry@ms.chinmin.edu.tw*

Abstract—Based on the handwriting calculation, this paper realizes the arithmetic operations of fixed point using the 8-bit single chip microcomputer 8051.

Keywords— Microcomputer, 8051, arithmetic, calculation, handwriting

I. INTRODUCTION

An idea may change one's life. Even a device may change the world. In 1980, a hand-hell calculator can solve a Schrödinger equation [1] and one also has used 8051 microcontroller to do numeric processing for instrumentation development [2]. Recently, people designed a calculator with 8051 microcomputer [3]. In addition, implementation of asynchronous 8051 processor has been made to speed up the computation [4]. Furthermore, some people have implemented arithmetic unit focusing on improving number of clock cycles based on cellular automaton [5]. Nevertheless, human nature is used to do calculation in base of 10. However, machines base on 2. To compromise between the two, people have proposed the decimal floating point format specification for easy use in daily life such as money transactions and for compliance with the modern processors [6]. In this paper, a different kind of design concept is introduced and which is a design of calculator based on the behavior of handwriting. The motivation is that many scientific technologies originate from natural phenomenon or mimic the behaviors of creatures on the earth. Based on this kind of philosophy, using an 8-bit adder incorporated with complement one can achieve the arithmetic operations of any numbers with enough memory buffers. Basically, An 8-bit represents a number of 255 at most. If assign each digit of a number an 8-bit memory to store the value of that digit, an 8-bit adder with help of complement can do any number of addition, subtraction, division, and multiplication. In the subsequent sections, a more detail of arithmetic operation imitating handwriting calculation is demonstrated including a photo of implementation using single chip microcomputer 8051 [7].

II. THE PROPOSED ARITHMETIC OPERATION BASED ON HANDWRITING CALCULATION

In this design, the binary coded decimal (BCD) data format is used. However, an 8-bit BCD data format is used instead of 4-bit BCD data format and which may speed up the calculations, particularly for multiplication and division. In this paper, AO-HWC is used hereafter for the acronym of proposed arithmetic operations based on handwriting calculation. While doing an arithmetic operation on an x-digit number and a y-digit number, numbers are assigned with x bytes of memory and y bytes of memory. Additionally, every number needs one more byte of memory for identification of positive or negative. Hence, an N-digit number requires N+1 bytes of memory.

A. Addition and Subtraction

In operations of addition and subtraction, there are three stages which are signs checking, addition or subtraction, and carry checking. In more detail, the steps of calculation are as follows.

Step 1. Check the sign digits of both numbers.

Step 2. If both numbers have the same signs; namely, both are positive numbers or negative numbers, both numbers are added up to the answer and the sign digit of the answer follows that of either number. If numbers have different signs, the two numbers are compared. The sign of answer follows that of the greater number and the answer is equal to the greater one subtracting the other one.

Step 3. Check each digit of answer in memory. If the 10^n digit is greater than or equal to ten, subtract 10 from the nth digit and add 1 to the 10^{n+1} digit.

Example 1 demonstrates how addition and subtraction of numbers N1 and N2 are carried out. Moreover, Table 1 shows two numbers in the memory and Table2 shows the contents of memory after the addition of two numbers. Table 3 and Table 4 list out how numbers are changed in memory after carry checking.

Example 1: For numbers 98 and 76, then it becomes

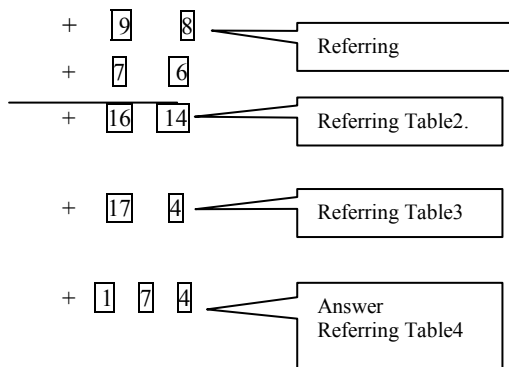


Table 1. Numbers in memory

Address	8-bit Memory
001	+
002	9
003	8
...	
...	
...	
011	+
012	7
013	6

Table 2. Numbers in memory after addition

100	+
101	
102	16
103	14

Table 3. Numbers in memory after carry checking

100	+
101	
102	17
103	4

Table 4. Answer in memory after carry checking

100	+
101	1
102	7
103	4

B. Multiplication

In the operation of multiplication, there are four stages which are sign checking, addition, shifting, and carry checking.

Step 1. Check the sign digits of both numbers.

Step 2. If both numbers have the same signs, the sign of answer is positive, otherwise negative.

Step 3. By programming skill of indirect addressing which mimics the shift of handwriting, multiplication is done by a series of additions one after another.

Step 4. Check each digit of answer in memory. Do the same carry operation as the step 3 in addition.

Example 2 describes how multiplication is done by a series of additions.

Example 2: For multiplication of numbers 98 and 76, it is done traditionally by handwriting in the following way.

$$\begin{array}{r}
 98 \\
 \times 76 \\
 \hline
 16 \\
 18 \\
 24 \\
 + 27 \\
 \hline
 3136
 \end{array}$$

Similar to the calculation of handwriting, it is done by a microcomputer with a series of additions. However, it will not be done by 31 times of additions with number 98. It would be time consuming. In case of 1×10000000000 , it would require at least 1 hour to complete ten billion times of additions with 1 for 8051 microcomputer. To avoid such a situation, the proposed method completes the series of additions in the following way. In that, each number in the box represents a number in a memory.

$$\begin{array}{r}
 98 \\
 \times 32 \\
 \hline
 98 \quad \text{addition: } 8 + 8 \\
 + 98 \quad \text{addition: } 9 + 9 \\
 \hline
 1816 \quad \text{carry-1} \\
 196 \quad \text{carry-2} \\
 \hline
 196 \\
 \times 98 \\
 \hline
 196 \quad \text{additions: } 9 + 8 + 8 + 8 = 33 \\
 98 \quad \text{additions: } 1 + 9 + 9 + 9 = 28 \\
 98 \\
 + 98 \\
 \hline
 2836 \quad \text{carry-3} \\
 2926 \quad \text{carry-4} \\
 3016 \quad \text{carry-5} \\
 3136 \quad \text{carry-6} \\
 112136 \quad \text{carry-7} \\
 + 21136 \quad \text{carry-8} \\
 \hline
 31136
 \end{array}$$

As shown above, there are only $[(3+2)-1] \times 2 = 8$ additions and $2+6=8$ carry operations in the case of 98×32 . In addition, numbers in the memory may be more than 15. That is why

using 8-bit BCD format in stead of convectional 4-bit BCD format. In general, the time complexity of multiplication depends on the number of additions and the number of carry operations. Suppose that an M-digit number X is multiplied by N-digit number Y. The number of additions required are $[(y_0+y_1+...+y_{N-1}) - 1] \times M$, where y_i denotes the number of ith digit of number Y. And the number of carry operations required at most are $[(M+N)-1] \times 9$. For example of 124×76 , it requires $[(6+7)-1] \times 3 = 36$ times of additions and no more than $[(3+2)-1] \times 9 = 36$ times of carry operations.

C. Division

By the same way, division is done by a series of subtractions. In the operation of division, there are three possibilities which are the divisor is greater the dividend, the divisor is less than the dividend, and the divisor is equal to the dividend. To do the division, it requires four stages.

Step 1. Check the sign of both numbers.

Step 2. Compare the two numbers.

Step 3. Mimics padding 0 to the divisor or dividend by shifting the divisor or dividend which depends on which number is less the other. Meanwhile, the quotient is shifted forward or backward by using indirect addressing.

While padding 0 the divisor which mans the dividend is greater than the divisor, divisor is shifted forward. That is equivalent to multiply the divisor by 10. Such a shifting operation may speed up the division in the series of subtraction to avoid the formidably time-consuming situation of $10000000000 \div 1$.

Step 4. Add 1 to the quotient for each subtraction of divisor from the dividend until the remainder is 0 or less than the divisor.

The above description may be kind of wording and uneasy to get the picture. The following example will concentrate on the key point of how division is done by padding 0, shifting, and subtractions.

Example 3: Suppose that the calculation is $999 \div 3$. Division is done in the following way. Assign divisor, dividend, and quotient memory buffers plus a pointer pointed to the least significant digit (LSD) of the quotient. As shown in Figure 1, the divisor is less than the dividend. So, the divisor is shifted and padded with 0 as shown in Figure 2. Meanwhile, the pointer shifts forward indirection of the most significant digit (MSD). Keep padding and shifting until the divisor is greater than the dividend and then shift the divisor backward as shown in Figure 3.

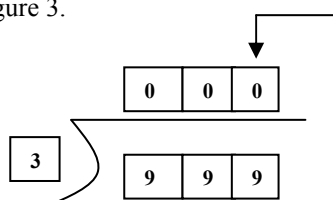


Figure 1. Assign memory buffers

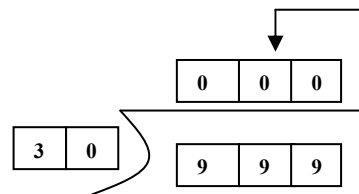


Figure 2. Shift divisor forward and pad with 0

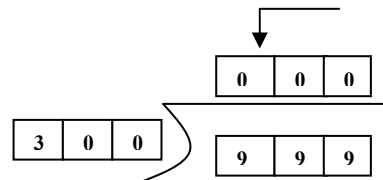


Figure 3. Shift divisor forward, pad with 0, and move pointer forward.

In the next, it subtracts the divisor from the dividend and the digit of quotient indicated by the pointer is increased by one as shown in Figure 4. As shown in Figure 5 followed by Figure 6, this process continues until the dividend is less than the divisor. In turn, the dividend is shifted forward and padded with 0 at LSD again. At the same time, the pointer shifts backward as well. It is a kind of recursive process.

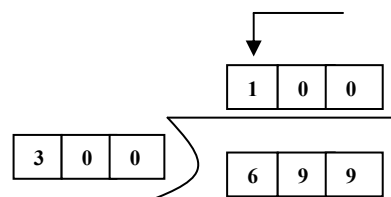


Figure 4. Subtraction of divisor from dividend and the increase of quotient by one

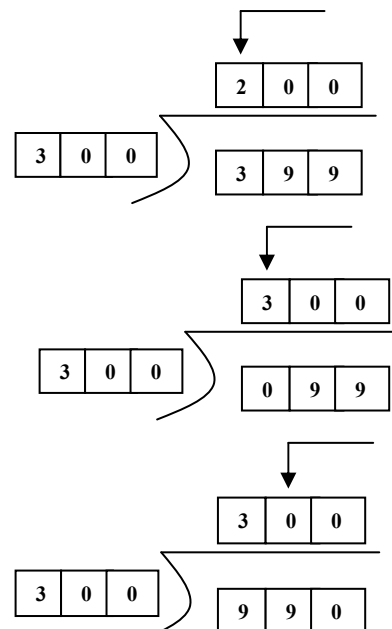
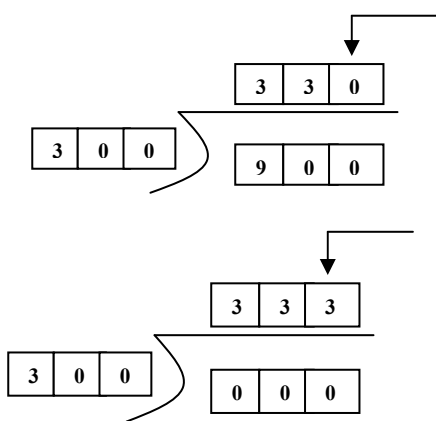
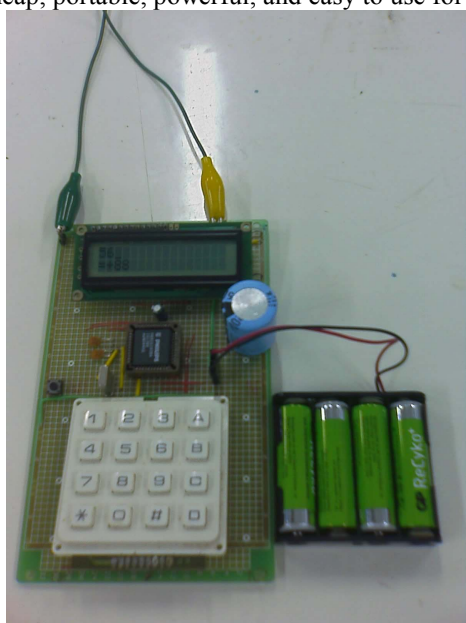


Figure 5. The recursive process of division



III. IMPLEMENTATION OF AO-HWC USING 8051 MICROCOMPUTER



IV. POWER SYSTEM FOR CALCULATOR

In addition to the calculator, a simple, low cost, and effective uninterruptible DC power system (UPS) is designed as Figure 8. Normally, 5V voltage supplies the power of the 8051 and also charges the battery BT1 and the capacitor C1 at the same time. When the power of 5V is out of work, the energy stored in the capacitor C1 keeps the system working at the very moment without interrupt. As long as the energy in the C1 is consumed and results in the voltage on C1 lower than that of the BT1, the BT1 takes the job of supplying power to the 8051. Whenever the power of 5V comes back, the BT1 stops supply power to the 8051 system and is charged for supplying power at next time of 5V power fail to work. In this uninterruptible power system, the backup power BT1 and the main power 5V are isolated by the diodes D1, D2, and D3. So, the main power and the backup power BT1 will not interfere by each other. Further, there is a little bit voltage-drop at diodes. Therefore, the BT1 will not be overcharged.

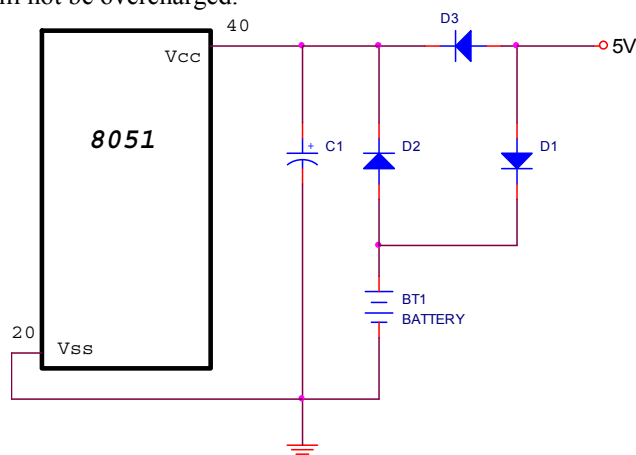


Figure 8. Uninterruptible power system

V. CONCLUSIONS

The design concept of AO-HWC is a trade of time complexity for space complexity. With the help of programming skill, the designed calculator can do a multiplication of result equivalent to a number of 32-bit. The complexity in terms of big **O** is only **O**(n) for addition and subtraction, **O**(n²) for multiplication and division. Such a design concept is only suitable for those cases for which time is not sensitive. In addition, an UPS of DC power system is designed with only three diodes, one capacitor, and rechargeable batteries.

ACKNOWLEDGMENT

I would like to express my gratitude to Chin Min Institute of Technology for offering me the travelling expenses and five days of leaving in order to present my work at "The 12th International Conference on Advanced Communication Technology" at Phoenix park in Korea.

REFERENCES

- [1] G. Faucher, "Solving the Schrödinger Equation with a Hand-Held Calculator", IEEE T. EDUC., 1980.

- [2] V.T. Randal, J.L. Schmalzel, and A.P. Shepherd, "Floating-Point Computation Using a Microcontroller", In *Proc. IEEE Engineering in Medicine & Biology Society of 10th Annual International Conference* 1988. IEEE Press, pp. 1243-1244.
- [3] A. Dottler and D. Hollobaugh, "8051 Calculator", USA: XESS Corp. 2001. Available:<http://www.xess.com/projects/uccalculator1.pdf>.
- [4] J.H. Lee, Y.H. Kim, and K.R. Cho, "A Low-Power Implementation of Asynchronous 8051 Employing Adaptive Pipeline Structure", *IEEE T. CIRCUITS-II*, vol. 55, pp673-677, Jul. 2008.
- [5] P.P. Choudhury, S. Sahoo, and M. Chakraborty, "Implementation of Basic Arithmetic Operations Using Cellular Automaton", in *Proc. ICIT '08*, 2008nIEEE Press, pp.79-80.
- [6] M.F. Cowlishaw, E.M. Schwarz, R.M. Smith, and C.F. Webb, "A Decimal Floating-Point Specification", in *Proc. ARITH-15 '01*, 2001, IEEE Press, pp147-154.
- [7] *User's Manual(MCS-51 Family of Single Chip Microcomputers)*, Intel Corporation, 1976.
- [8] *Flash Magic Software(Flash Magic.exe)*, Embedded Systems Academy.
- [9] "P89C60X2/61X2 Data Sheet", Philips, 2003.