International Journal of Advanced Engineering and Technology
ISSN: 2456-7655; Impact Factor: RJIF 5.54
Received: 26-04-2020; Accepted: 13-05-2020; Published: 20-05-2020
www.newengineeringjournal.com
Volume 4; Issue 2; 2020; Page No. 24-26

Analysis and discussion of problems in the multiplication of different types of data of single-chip microcomputer

Hanyu Yang
School of Physics and Electronic Engineering, Qilu Normal University, Jinan, China

Abstract
When C51 single-chip microcomputer performs multiplication operation with the command *, sometimes the result of the calculation is error, especially when the data types are different and the data is relatively large. This situation is mainly due to the multiplication operation missed one step of operation after data type conversion. In this case, we can use introducing intermediate variables, distribution operation, or cast data type to correct it.

Keywords: single-chip microcomputer, multiplication, different types of data, C51

1. Introduction

Simply speaking, a single-chip microcomputer is an integrated circuit chip. In detail, it is a small and perfect microcomputer system which uses VLSI technology to integrate CPU, Random-Access Memory (RAM), Read-Only Memory (ROM), I/O Port, Interrupt System, Timer/Counter and other functions on a silicon chip [1].

The technology of Single-chip microcomputer started from the 1990s. At present, with the continuous development of science and technology, the practice and application of single-chip microcomputer technology are becoming more and more mature. People pay more and more attention to the development and application of single-chip microcomputers in intelligent electronic technology. Nowadays, the single-chip microcomputer has been widely used in intelligent instruments, communication equipment, a navigation system, household appliances, and other fields [2].

The built-in arithmetic unit of the single-chip microcomputer is composed of an arithmetic logic unit, accumulator and register. The arithmetic logic unit is to carry out arithmetic or logic operation on the data transmitted. Its input source must be two 8-bit data, which come from accumulator and data memory respectively. The arithmetic logic unit can add, subtract, and, or, compare the two data, and finally store the result in the accumulator [3, 4].

The multiplication in the single-chip microcomputer can be realized by MUL AB. The high octave of A * B result is B, and the low octet is A. Or the multiplication can be realized by using the method of the left shift and the operation of * in the single-chip microcomputer [5, 6].

But, when the multiplication operator * is used, the two data must be of the same data type and the operation value is relatively small. When the two data types are different or the operation value is large, the operation result is often different from the actual operation value [5].

However, in practice, the single-chip microcomputer often has the task to do some multiplication works with the different data types and large numbers. So, how to use the single-chip microcomputer to complete the above tasks and give accurate results? This paper intends to analyze and discuss this problem.

2. Problems and analysis

There are many types of data that can be processed by the single-chip microcomputer. When doing multiplication, we often encounter data of type int, char, and long. Therefore, this paper discusses the multiplication of different data by taking the data of type unsigned char and data of type unsigned long as examples.

The data of type unsigned char and data of type unsigned long are two different types of data. When they do multiplication, the result of the calculation is often inconsistent with the actual result.

Example 1. Calculate 9 times 9000.

Solution: the following program was compiled to solve the problem:

```
#include<reg51.h>
Main ()
{
Unsigned char a;
Unsigned long b;
a=9;
b=a*9000;
}
```

Calculation results: 15464

However, the actual result is 81000. From this, it can be seen that the calculation results of the single-chip microcomputer are obviously different from the actual calculation results.

What are the reasons for this? We take C51 single-chip microcomputer as an example and analyze it from its principle.

According to the operation rules of the C51 single-chip microcomputer, the two types of data must be the same when data operation is carried out [1~4]. When the data types are different, one of the data types needs to be converted to the same data type as the other.

At the same time, in the above problem, the data of the type of unsigned long greater than 256 is stored in the form of two bytes of unsigned char, so the data of unsigned char needs to be expanded to two bytes. In this way, the expanded operation becomes the multiplication of two double-byte numbers [7].

In this way, according to the operation rules of the C51 single-chip microcomputer, the above calculation is less than

one multiplication operation in actual operation. In other words, some of the data do not perform multiplication. To be exact, it is the first 8 bits that do not perform multiplication. Therefore, in the operation process, there is a lack of high 8-bit operation results. At the same time, the operation results of the last 8 bits also occupy the position of the operation results of the first 8 bits, so that the operation results are not correct.

3. The method of solution

Therefore, we think that there are three methods that can be used to solve the above problems.

3.1. Introducing intermediate variables

That is, by adding an intermediate variable to the calculation to solve the above problem. This intermediate variable added here needs to be able to store the result of the operation.

According to the relevant operation rules of the C51. First, the data of type unsigned char changes to the same data type as the intermediate variable, then the multiplication is performed. At the same time, the C51 can store the operation results of the same type of data. In this way, the first and second 8-digit numbers can do multiplication.

Example 2. Calculate 9 times 9000.

Analysis: For this problem, we can add an intermediate variable C to get the correct value. The specific increase method is as follows.

```
Solution: #include<reg51.h>
Main ()
{
Unsigned char a;
Unsigned long b, c;
a=9;
c=9000;
b= (a*c);
}
```

In this operation, the type of data A is converted to the type of unsigned long consistent with C. The operation result is also of type unsigned long. The C51 can automatically store the data of 9 * 9000, in this way that the calculation result will not be wrong.

3.2. Distribution operation

This method is to divide the data larger than 256 into the product of two or more data, and then perform separate operations. Because the separated data are 8 digits, the result of multiplication can be stored correctly, so that there will be no omission in the end.

Example 3. Calculate 9 times 9000。

Analysis: For this problem, we can decompose 9000 into 90 times 100, and then calculate the product respectively to get the correct result. The details are as follows.

```
Solution: #include<reg51.h>
Main ()
{
Unsigned char a;
Unsigned long b;
a=9;
b= (a*90);
b= (b*100);
}
```

In this operation, the first operation a * 90 is the operation of two unsigned char data, and the result is stored in B of unsigned long. The overall operation result is of the type of unsigned long, and the number of 9 * 9000 can be stored at B of unsigned long, so the operation result is correct.

3.3. Casting data type

The casting data type is to use "()" to cast the data of type unsigned char, and then perform the operation. There are two conversion methods in the C51: implicit conversion and display conversion. The above two solutions are implicit conversion, that is, the C51 processes and transforms data by itself. In this method, the display of data type is converted, that is, the data type is converted manually. When both data are of the type of unsigned long, the operation results are stored in the storage unit of the type of unsigned long that can be stored.

Example 4. Calculate 9 times 9000.

Analysis: We can first force A into the data of type unsigned long, and then perform the multiplication operation. The specific method is as follows.

```
Solution: #include<reg51.h>
Main ()
{
Unsigned char a;
Unsigned long b;
a=9;
b= ((unsigned long) a)*9000;
}
```

In this operation, A is first cast to the type of unsigned long, which can store the data of 9 * 9000. So the result is correct.

4. Conclusion

When the single-chip microcomputer performs multiplication operation on two kinds of large and different types of data, the result is often incorrect. This situation is mainly due to missing one step of operation after data type conversion. Therefore, we think that we can get accurate results by introducing intermediate variables, distribution operations, or casting data type.

However, these three methods have their own characteristics, so from the convenience of operation, each has its own advantages and disadvantages. From the discussion above, the first and second methods can be applied to the operations with different data types, larger data but less data amounts. When the data type is different, the data is large, but the amount of data is also large, the third method can more effectively prevent operation errors.

5. References

1. Li QL. Principle and application of single chip microcomputer. Beijing: Tsinghua University Press, 2014.
2. Fan LN. Principle and application of single chip microcomputer. Beijing: China Machine Press, 2019.
3. Huang FZ. Principle and application of single chip microcomputer. Beijing: China Machine Press, 2019.
4. Wang HW. Principle and application of single chip microcomputer. Beijing: Tsinghua University Press, 2019.
5. Liu GY. Analysis of multi bit multiplication operation in C language programming of single chip microcomputer. Computer Knowledge and Technology. 2019; 15(24):242- 255.
6. Jiang H. Correction of double byte multiplication subroutine in PIC16 single-chip microcomputer

Microcontrollers & Embedded Systems, 2007; (09):70-71.
7. Yan KJ, Zhang S, and Huang XW. Discussion on multi bit multiplication operation in C language programming of single chip microcomputer. Journal of Guangdong University of Technology, 2006; (04):23-26.