ATLANTIS PRESS
Advances in Computer Science Research, volume 44
3rd International Conference on Wireless Communication and Sensor Network (WCSN 2016)

Simple Structured Data Initial SZG File's Generation Software Design and Implementation

Shuang Li
School of Computer Engineering and Science, Shanghai University,
Shanghai, 200444 China

Yi Jin
School of Computer Engineering and Science, Shanghai University,
Shanghai, 200444 China
yijin@shu.edu.cn

***Abstract*-SZG file is a programming platform for users to use Ternary Optical Computer (TOC) in high-level language program. This paper proposes an initial SZG file, with an extension .SZG, generation technology for structured data. The technology proposed in the paper can implement some functions such as storing the initial SZG file, appending operands to a SZG file, adding data and modifying the initial SZG file in external memory continually. Therefore, in the process of using the technology, users do not need to understand the format of SZG file and the process that the computer system analyzing the SZG files. Furthermore, in application programs, TOC can parallel process large amounts of raw data in rapidly by using the SZG file. In this paper, an experiment of the initial SZG file of contains four simple structured data (SSD) were tested, and the experiment results confirmed the validity and correctness of the technique and software.**

Keywords-ternary optical computer; initial SZG file; simple structured data; module design; software design; software implementation; software test.

I. INTRODUCTION

In 2000, Jin Yi et al. proposed the concept of the ternary optical computer (TOC) and designed its structure [1][2][3][4]. TOC is one of various new type computers. From the application point of view, a ternary optical processor has up to thousands of data bits. Each data bit can be assigned to a single task independently and reconstructed in real-time according to users' demands during running time. Traditional computers do not have these advantages [5][6][7][8][9]. As a result, TOC will be more efficient than several traditional computers when calculating large quantities of data [10][11][12].

The SZG file is divided into two categories: the simple data types and the structure data types. Considering the development trend of TOC, the former first to be studied, and the simple data type generation software has been upgraded to the second version now (CHSH_JD_SZG_SHCH_LSH_2016). With the deepening of research on TOC [13], it can handle the data types and processing methods more and more close to the complex activities of human. So the structure data becomes a current concerning point. There are two types of the structured data that the TOC can handle them. They are the database structured data and the SSD. This paper mainly studies the initial SZG file generation technology of the SSD, and full description of the main results of this study.

II. THE THEORETICAL BASIS OF THE INITIAL SZG FILE'S GENERATING TECHNOLOGY

The SZG file is the only way to exchange information between users and the TOC when processing large quantities of data. The data that recognized and calculated through the TOC must be complied according the protocol requirements of the SZG file strictly. The SZG file format is given in Figure 1.

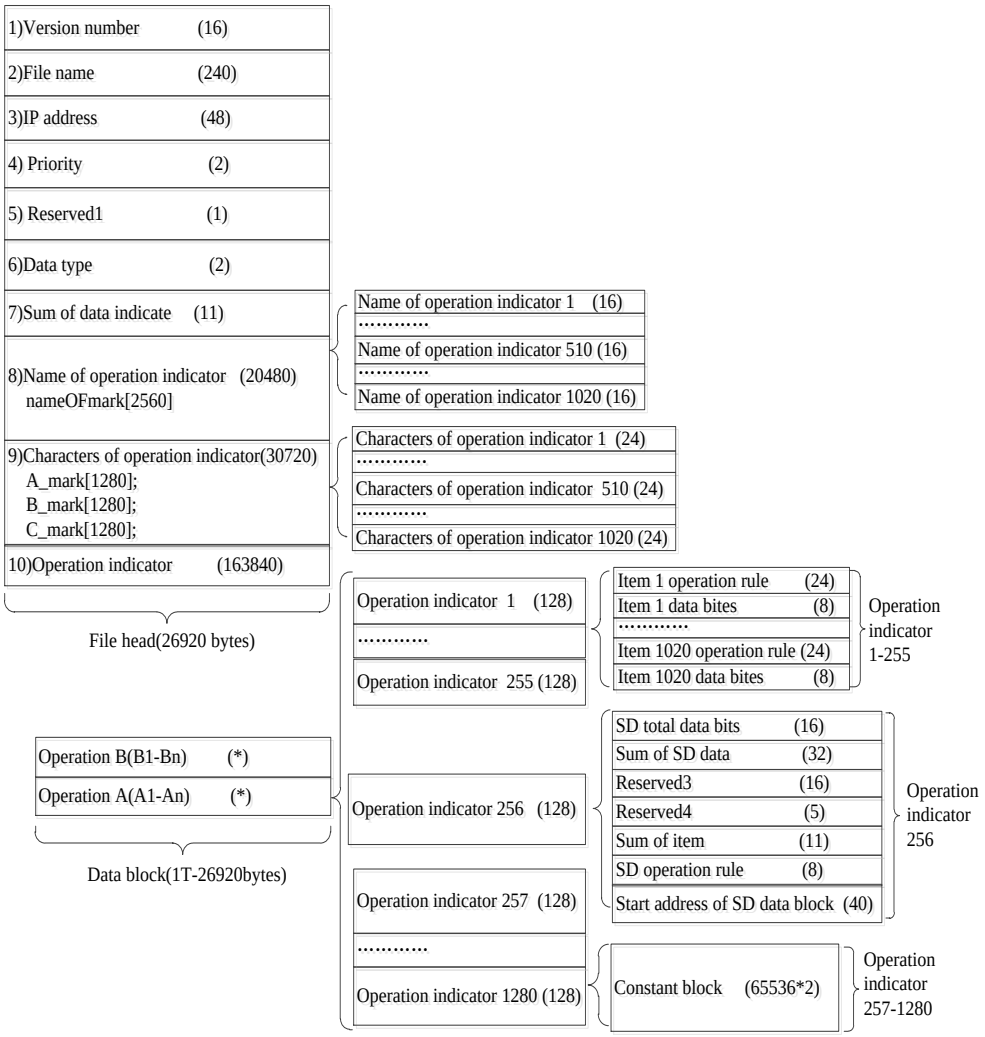

Figure 1. SZG file format of the simple structured data

III. THE DESIGN OF THE INITIAL SZG FILE'S GENERATION TECHNOLOGY

A. *The Overall Design of the Target Software*

The design of the target software will be decomposed into the following three functional modules:

1) Information input module. This module provides users with a user-friendly input interface, which help users to enter all kinds of information. The technology will check the information following the SSD SZG file.

2) Creating buffer module. The module's function is to prepare two buffers in the memory to store SZG header and the original operands.

3) The initial SZG file of SSD generation module. This module's function is to organize the information that meet the requirements on the input interface into SZG file.

B. *Target Software Process Design*

According to the above overall design, the flow chart of the target software is given in Figure 2:

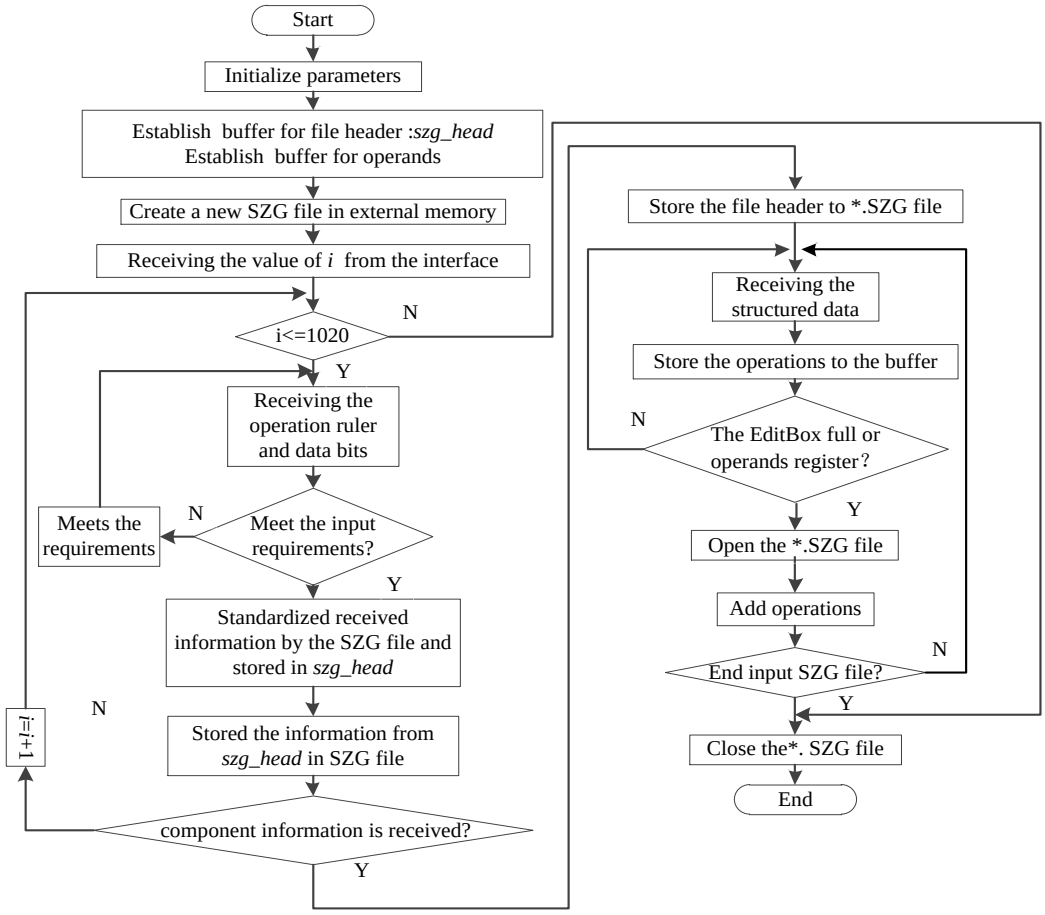

Figure 2. The techinlogy flow chart

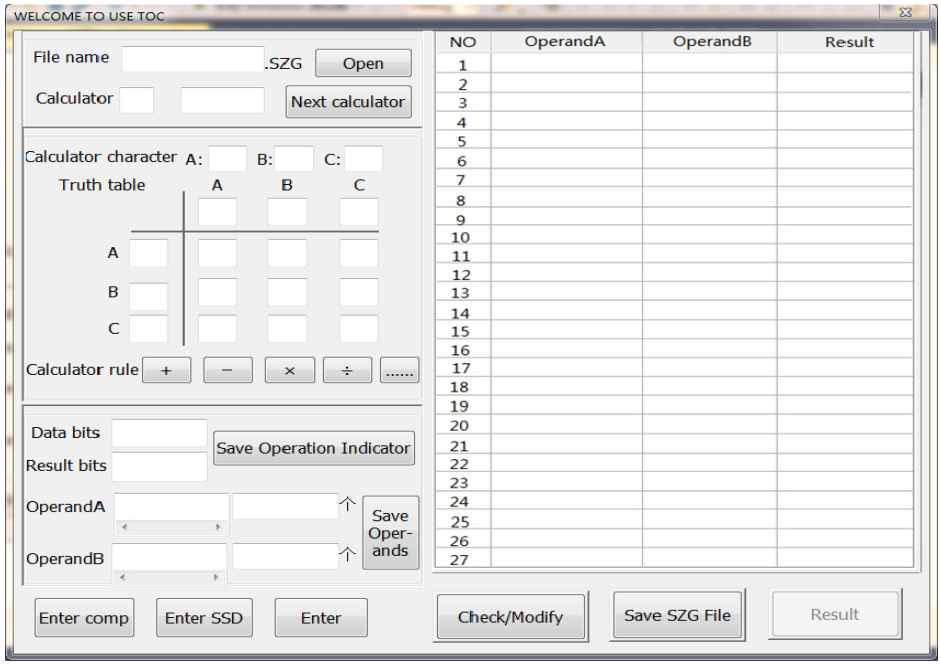

Figure 3. Target Software Interface

IV. THE IMPLEMENTATION OF THE SOFTWARE

Based on the above technology research, we developed the initial SZG file generation software. According to the naming rules of the TOC team, the name of the software is: CHSH_JD_SZG_SHCH_LSH_2016.

A. *The realization of the input interface*

The task of this module is to assist users input the information that constitutes the initial SZG file. The software interface design as follows Figure 3.

B. *The Process of SZG File's Generation*

We give the process of SZG file's generation by reference related content of literature [14]. Because the SSD is defined by user according to the application case, the processor manufactures are impossible to know the details of the SSD before the processor leaving the factory. From the fact, it is required that structured data processor (SDP) must be reconfigurable in hardware at any time. When user asks for reconfiguring SDP, the monitoring system or operating system will reconfigure the optical processor hardware to satisfy the user's requirements. Recently, this process can be completed in TOC via some techniques as follows:

(1) User enters operation rule and corresponding bits for SDP's each component one by one on the Figure 3. After entering one component, press the Enter Comp button. After entering all components, press the Enter SSD button. For tri-valued or two-valued logical the operation rules are given in the "Truth Table", and for addition, subtraction, multiplication and division the operation rules are given by press a corresponding button among the "+", "-", "*" and "\".

(2) User enters all structured data in the software interface, separates components by commas and separates structured data by semicolon. Every data will be displayed on right of interface and can be modified.

(3) By pressing the Enter button, the plug-in creates a TOC data file which conforms to the SZG file format and has .SZG expanded-name from the information inputted by user.

C. *The Realization of the Memory Buffers*

The input box of the operand using EditBox, the maximum capacity of the box is 32KB. It is bound to increase the number of interactions with external memory each time if the data reaching 32k written to SZG file. Therefore, we have created buffers to resolve this problem. Because of the file header size of the simple structured data is fixed, and the size of the data area is variable, it is determined by the original operands, so we create the SZG file header buffer and the SZG file operand buffer separately.

Create the SZG file header buffer:

SSD SZG file header contains nine specific parameters and 1020 operation signs. First we need to declare the structures of the operation sign before the file header buffer is established:

```
struct Op_Mark
{
    bitset<24>    OperateRule;
    bitset<8>     BitsOfItem;
};
```

Then statement the structures of the SZG file header, according to the SZG file format shown in Figure 1, as follows:

```
struct SZG_FILE{
    bitset<16>      Version_ID;
    char            NameOfFile[30];
    bitset<48>      IPAddress;
    bitset<2>       Priority;
    bitset<1>       Reserved1;
    bitset<2>       TypeOfData;
    bitset<11>      AmountOp_Mark;
    char            nameOf mark[2560];
    char            A_ mark [1280];
    char            B_ mark [1280];
    char            C_ mark [1280];
    Op_Mark         Op_mark[1020];
    bitset<16>      BitsOfSD;
    bitset<32>      SumOfSD;
    bitset<16>      Reserve3;
    bitset<5>       Reserve4;
    bitset<11>      NumOfItem;
    bitset<8>       OperateRuleOfSD;
    bitset<40>      DataAddress;
    Cstring         Constant;
} szg _header;
```

Then we have defined a SSD variable named *szg _hearder*. The system allocates memory unit for the *szg _hearder* when the program runs to here, and this memory unit is the SZG file header buffer. Considering the understandability of the program, we agreed to use the abbreviations that consistent with the field name of the file header when we definition the member variable in the definition structure. For example, the version number is represented by *Version_ID*, and the file name is represented by *NameOfFile*.

V. EXPERIMENT AND ANALYSIS

The experiment has a test running in TOC simulation software environment that constituted by two computers. We use a PC to simulate the auxiliary processor of TOC. The PC, called the back-end computer, runs with the TOC's underlying software. Another PC is the programming platform which is called the front-end computer and used by the programmer.

Refer to cases of the experiment in reference [14]: we need to compute $f1=a+b$, $f2=c-d$, $f3=e\wedge f$, the bits of a, b, c, d, e and f i are 8, 8, 5, 5, 3 and 3 respectively, and each variable has 2000 data. As everyone knows, in the electronic computer the technology of solving this problem is to compute $f1$, $f2$ and $f3$ one by one, and then repeat the computation 2000 times. So the operation instruction which includes addition, subtraction, logic AND and logic OR instructions will run 6000 times regardless of CPU, many-core, multi-core or GPU in total.

But, in the TOC, a SSD denoted by *S* will be defined with four components of 8-bit int type, 5-bit int type and 3-bit logic type. And two structured data variables *A* and *B* will be defined with the SSD *S*. *A* includes scalar variables *a*, *c* and *e*, and B includes scalar variables *b*, *d* and *f*. It must be noted that *S*, *A* and *B* are defined only in the mind of programmer and they are never written in program. After defining *S*, *A* and *B*, programmer can ask TOC to configure out a corresponding SSD via inputting the four components of *S*, and input the 2000 pairs structured data of *A* and *B* in Figure 3. After receiving the request of configuring the SSD and the 2000 pair structured data, TOC allots 16 bits of its optical processor to reconfigure the SSD through configuring the first 8 bits into adder, the second 5 bits into subtractor, the next 3 bits into logical AND unit. Then the first pair of structured data in *A* and *B* is sent to the 16-bit SDP and the first group results of *f1*, *f2* and *f3* fare gat in one operation instruction. The TOC only has one operation instruction which controls the optical processor running one time, but the electronic computer has several operation instructions such as ADD, SUB and AND et. And each of them controls itself computing unit running. The TOC only repeats the operation instruction 2000 times, then the 2000 pairs of structured data, i.e. all the 6000 pairs of scalar data, can be computed.

The SZG file was generated in external memory, shown in Figure 4-(a).

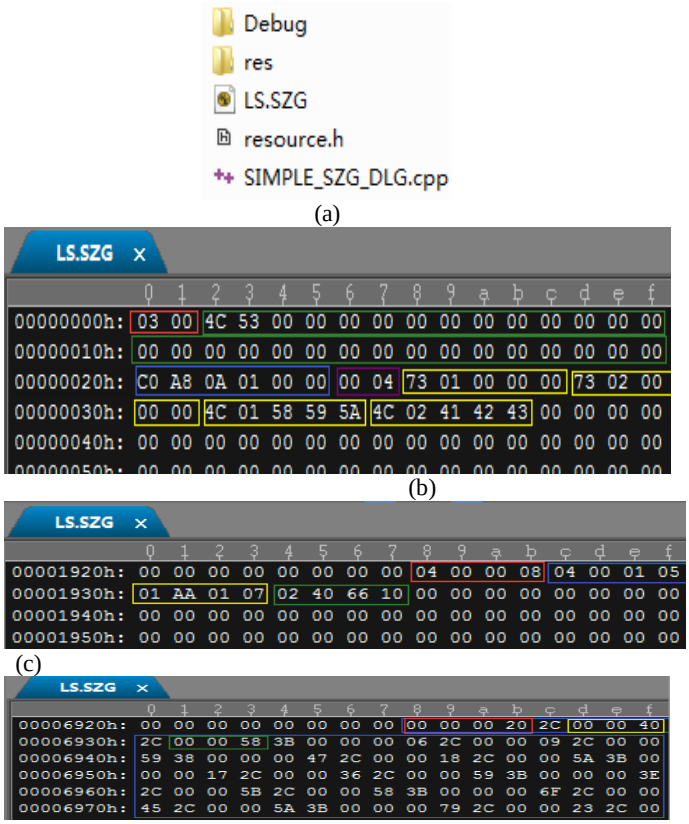

Figure 4. The SZG File

Figure 4-(b) is the parameter area of SZG file header, the first two bytes are given the version number 0300H. 30 bytes in the green box shows the name of the file LS, it corresponds to the ASCII code 4C, 53. Then the blue box gives the IP address 192.168.10.1 of the computer. The purple box shows that the SZG file contains four operation signs. And the next four yellow boxes give the names of the four operation signs: *s1*、*s2*、*L1* and *L2*. Among them, the following bytes of *L1* and *L2* give their operation sign characters. And the following bytes 00 00 00 of *s1* and *s2* indicate that the operation is numerical operations.

In Figure 4-(c), 1928H is the start address of the data, and it is the beginning of the operation signs. The data in red, blue, yellow, green four boxes are the four items of SSD *s1*, *s2*, *L1* and *L2*. The address of the data in the red box is 1928H-192bH that is consistent with the design of Figure 1. The first three bytes 04 00 00 is the arithmetic rule of the addition, and the following byte 08 represent data bits is 8 decimal numbers. The meaning of the rest three items is similar, so we will not repeat them here.

Figure 4-(d) is the operand area of SZG file. Data in the blue box is the first 21 data of the SSD. SZG header has 26920 (0-26919) bytes, the address of the data area starts from 26920 byte which can be represented as a hex- notation "6928H". Take the first number as an example, the first data is 32 hexadecimal representation 20H. On the other hand, we enters all structured data on the software interface, separates components by commas and separates structured data by semicolon. Because *b*, *d* and *f* are 8 decimal numbers, 5 decimal numbers and 3 bit logic type respectively, so, they need 4 bytes binary, 3 bytes binary and 3 bytes binary respectively. Among them, 2C means ","and 3B means ";".

Experimental test illustrates the generated initial SZG file is fully compliant with the latest version of the SZG file format. The validity and correctness of the initial SZG file generating technology of the simple structured data is validated.

VI. Conclusion

This paper describes the initial SZG file generating technology of the simple structured data. It proposes the process technology design and the technology of software implementation. The software can generate the SZG file of SSD accurately, according to the operation rules and raw operands. The analysis of experiment results confirmed its correctness. The study also brings great convenience to compile applications for the TOC.

Acknowledgements

This work is under the support of the National Natural Science Foundation of Chine (Grant No. 61572305), the Shanghai Special Scientific Research Plan (Grant No. 15700500400), the National Natural Science Foundation of Shanghai (Grant No. 15ZR1415400), and the National Natural Science Foundation of Shanghai (Grant No. 13ZR1416000).

REFERENCES

[1] Y. Jin, H. C. He and L. R. Ai.: Lane of parallel through carry in ternary optical adder. J .Science in China (Series F), vol. 48, no. 1, 107-116 (2005)

[2] J. Y. Yan, Y. Jin and K. Z. Zuo.: Decrease-radix design principle for carrying/borrowing free multi-valued and application in ternary optical computer. J. Science in China Series F- Information Sciences, vol. 51, no. 10, 1415-1426 (2008)

[3] Y. Jin.: Draw near optical computer. J. Journal of Shanghai University (Natural Science), vol. 17, no. 4, 401-411 (2011)

[4] J. L. Bao, Y. Jin and C. Cai.: An experiment for ternary optical computer hundred-bit encoder. J. Computer Technology and Development, vol. 17, no. 2, 19-22 (2007)

[5] Y. Jin, S. Oushang, K. Song, Y. F. Shen, J. J. Peng and X. M. Liu.: Management of many data bits in ternary optical computers, Sci China InfSci, vol. 43, no. 3, 361–373 (2013)

[6] Y. Jin, H. J. Wang, S. OUYANG, Y. Zhou and Y. F. Shen.: Principles, structure, and implementation of reconfigurable ternary optical processors. J. Sci China Ser F-InfSci, vol. 54, no. 11, 2236-2246 (2011)

[7] Q. Zhang, Y. Jin, K. Song and H. Gao.: MPI programming based on ternary optical in supercomputer. J. Journal Shanghai University (Nature Science), vol. 20, no. 2, 180-189 (2014)

[8] Y. Jin, Y. Y. Gu and K. Z. Zuo.: Theory, technology and progress of a ternary optical computer's decoder. J. Science China Information Sciences, vol. 43, no. 2, 275-286 (2013)

[9] J. Y. Yan, Y. Jin and K. Z. Zuo.: Nocarry& noborrown-value operate unit. China Patent, ZL200710041144. 1, October 28 (2009)

[10] S. F. Li and Y. Jin.: Mapping technology from components pixels to data-bits of ternary optical computer. J. Computer Engineering and Design, vol. 31, no. 5, 1077-1080 (2010)

[11] X. J. HU, Y. Jin and S. OUYANG.: A 40-Bit Multiplication Routine of Ternary Optical Computer. J. Journal Shanghai University (Nature Science), vol. 20, no. 5, pp. 645-657 (2014)

[12] J. J. Peng, R. Shen, Y. Jin and Y. F. Shen.: Design and Implementation of Modified Signed-Digit Adder. J. IEEE TRANSACTION ON COMPUTER. J. vol. 63, no. 5, 1134-1143 (2014)

[13] Lovkesh.: Realization optical signal processing components using SOA. J. Optik-International Journal for Light and Electron Optics, vol. 122, no. 23,2136-2139 (2011)

[14] Y. Jin, Q. XU, S. OUYANG, Y. X. Han and W. M. Li.: Structured data computer—application characteristics of a ternary optical computer. J. Sci China Ser F-InfSci, vol. 46, no. 3, 311-324 (2016)