三值光学计算机数据处理机制关键技术研究

Research on key technologies of data processing mechanisms in ternary optical computer

Shuang Li1, 2, Shanchuan Liao1, Wen Yang3, Wenjing Li4,5\*, Luqun Li1\*, Dazhi Li1

1College of Information, Mechanical and Electrical Engineering, Shanghai Normal University, Shanghai 200234, China

2Shanghai Engineering Research Center of Intelligent Education and Bigdata, Shanghai Normal University, Shanghai 200234, China

3Shanghai Miyu Network Technology Co., Ltd. Shanghai 200234,China

4Guangxi Key Lab of Human-machine Interaction and Intelligent Decision, Nanning Normal University，Nanning 530100, China

5School of Logistics Management and Engineering, Nanning Normal University, Nanning 530001, China

\*Corresponding Author: Wenjing Li. Email: lslsshuang@126.com

Xxxx. Email: xxxx

abstract: This paper introduces the arithmetic data file, a key technology for data processing in a ternary optical computer. The physical form of the three-value optical processor and its data processing characteristics are analyzed. Based on this analysis, the compution-data is constructed, and research is carried out on the format of the compution-data, its generation method, and the expansion of the programming language transmitted to the three-value optical processor. The operation rules for the three-value optical computer and the raw data are organized into a file that conforms to the calculation characteristics of the computer. A data processing method based on the compution-data is proposed. Finally, the experimental test was conducted on the platform of a ternary optical computer using specific examples. The results showed that by organizing and transmitting data through the compution-data, the ternary optical computer can fully utilize its computational advantages in data processing while shielding the underlying complex hardware processing. This makes it convenient for users to apply this new type of computer. This data processing mechanism can offer a novel perspective for other heterogeneous systems in data processing.

KeyWords :ternary optical computer; data processing; compution-data file; heterogeneous system

# introduction

With the increasing richness in theory, structure, and experimental systems of the ternary optical computer (TOC), its computational advantages are attributed to three main characteristics[1-5]: numerous processor bits, allocatable processor bits, and the runtime reconfigurability of each processor bit’s computational function. In contrast, electronic computers have a fixed number of data bits, non-allocatable data bits among different users, and immutable computational functions of processors. Common data processing methods designed entirely based on the characteristics of electronic processors cannot exploit the advantages of TOC[6-8]. Therefore, new data processing mechanisms tailored to the TOC’s features of numerous data bits, allocatable data bits, and reconfigurable computational functions of data bits need to be explored[9]. The TOC is divided into an upper machine and a lower machine, with the upper machine being a conventional PC and the lower machine being the ternary optical processor[10]. A key issue that must be addressed at this stage is how to conveniently transfer data requiring TOC processing to the lower machine, fully leverage the computational advantages of the TOC, and allow people to have two types of computers that work collaboratively rather than in opposition.

The most significant difference between the TOC processor and traditional electronic processors is that it allows users to reconfigure the computational functions of each processor bit at any time, and the vast number of processor bits can be independently grouped and used[11]. Additionally, the TOC’s memory has a data bus width that varies with the total number of processor bits on the side facing the processor, but this characteristic is not apparent at the application program level[12-15]. However, various traditional data processing mechanisms established around electronic CPUs, GPUs, or MICs, which lack these three application characteristics, are not suitable for the TOC’s data processing methods.

This paper proposes a data processing method for the TOC, including the computation-data file, the generation of the computation-data file, and the transmission of the computation-data file. The computation-data file organizes the user-submitted computational rules and raw data into a specialized file designed according to the computational characteristics of the TOC. The computation-data file generation software runs on an auxiliary PC to assist users in generating the file. The transmission of the computation-data file involves extending a high-level language with a set of instructions to send the file to the TOC[16-17]. By parsing the information carried in the computation-data file, the optical processor is reconfigured, and the reconfigured calculator is used to process the data[18]. This data processing mechanism can fully exploit the computational advantages of the TOC, is fully compatible with traditional applications, shields users from the complex processing at the TOC processor end, facilitates the use of this new type of computer, and is expected to provide new ideas for data processing as other new types of computers enter the application stage.

# Characteristics Analysis of Ternary Optical Computers

## The physical form of ternary optical processors

In the various components of the ternary optical computer (TOC), processor bits exhibit different physical forms. The abstract structure of n processor bits is shown in Figure 1. The input and output memories are located in the storage space of the PC facing the user, forming a channel for data interaction between the TOC and the user. The input data buffers Ra and Rb, output data buffer Rc, reconstruction command buffer Rm, bit latches A, B, C, and the reconstruction circuit constitute the circuit part of the optical processor module[19-21]. Ra and Rb store the n-bit numbers of the two input data a and b for the current computation, Rc stores the n-bit number of the result c, and Rm forms the image of the optical calculator to be reconstructed in the form of a command set. According to the processor bit allocation strategy, the input data a and b are sent to the data buffers Ra and Rb, respectively. In Ra and Rb, each bit of data is parsed into an encoding form recognized by the encoder through a task analysis strategy. The encoding is then sent to latches A and B, passes through the encoding, control information channel, optical signal converter, and decoder enter latch C, and finally, the data is stored in the output register, completing a computation of n-bit data.

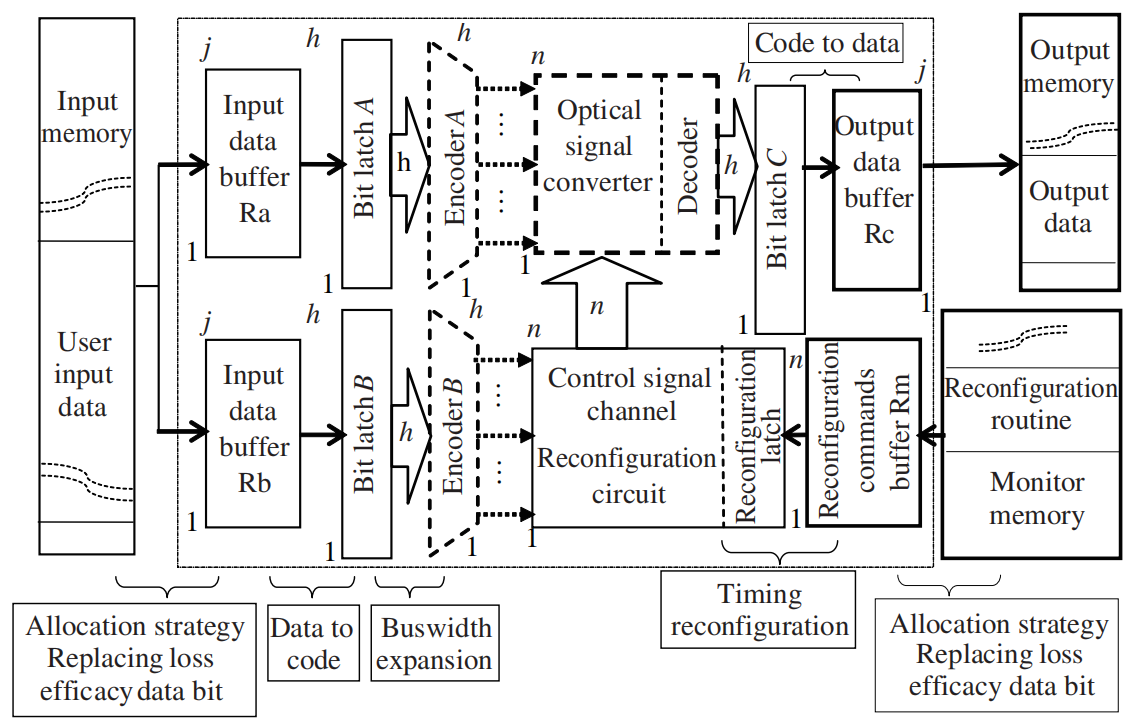


Figure 1: Abstract Diagram of the Physical Form of Ternary Optical Computer Processor Bits

## Ternary Optical Computer Computational Characteristics

Light possesses three physical states suitable for computational use (horizontal, vertical, and no light), which establishes the basis for the ternary optical computer (TOC) having three values[22-24].. The non-interfering nature of light during transmission determines that a TOC can have a multitude of processor bits. Furthermore, the substantial pixel count of liquid crystal devices enables the TOC to possess a vast number of processor bits in reality. [25,26]The natural laws inherent in the reduction design theory dictate the reconfigurability of the TOC’s processors[27-30]. The ternary nature and the reconfigurability of processors also make the TOC well-suited for binary Most Significant Digit (MSD) parallel adders. [31-33]

The characteristic of “numerous data bits” stems from the non-interference of light beams. This physical property allows multiple beams of light to cross paths within the same space without interference, enabling many signal light beams to be arranged together without isolation to form planar images. Consequently, the processing of a large amount of optical information can be conducted in the manner of planar image transformations. [34, 35].

The “on-demand reconfigurability” refers to the optical processor’s ability to dynamically adjust the functions of processor bits according to the actual requirements of the user [36]. Since the optical processor cannot anticipate computational demands in advance, the hardware must be reconfigurable at any moment. Considering system security, the reconfiguration requests are submitted by the user, and the processor is restructured by the system’s underlying modules as per these requests, specifically through the computation-data file.

# Computation-Data Files

## Principles of Computation-Data File Construction

The computation-data file is the sole method of information exchange between users and the ternary optical computer (TOC) when processing large volumes of data. The data entered by users must strictly adhere to the protocol requirements of the computation-data file to be recognized and computed by the TOC. The computation-data file consists of two parts: the file header and the data section. The file header records the overall information and the user’s computational demands, and the TOC’s task management system will allocate processor bit resources and reconfigure the ternary optical processor based on the information in the file header. The data section stores all the raw data input by the user and also places the computed results returned by the TOC to the user. In the latest version of the computation-data file, the file header occupies 26,920 bytes, and the data section ranges from 0 to (1T - 26,920) bytes. The principles for defining the format of the computation-data file are as follows:

(1) File Header: The file header contains comprehensive information about the task and the computational unit requested by the user. It includes all the information needed to reconfigure the processor, as well as the user’s priority, the number of computational units, and the total amount of data to be computed.

(2) Ease of Parsing: Since the task management software that parses the computation-data file runs on an auxiliary PC, which typically processes data in 32-bit or 64-bit binary units, the design of the computation-data file header must be compatible with the current processing requirements of PCs.

(3) Reserved Expansion Bits: As the ternary optical computer (TOC) continues to evolve and the problems it can handle become increasingly complex, the parameters and functions contained in the computation-data file will also become more enriched and varied. To accommodate these changes, it is necessary to reserve some bits in the appropriate positions of the computation-data file header to facilitate future upgrades of the computation-data file version.

## Computation-Data File Format

Constructed according to the principles as mentioned above, the computation-data file’s header contains information primarily to provide the necessary parameters to reconfigure the optical processor. In contrast, the data section includes the operands, including operand b and operand a. As shown in Figure 2, following the numbering provided in Figure 2, the construction of the computation-data file is detailed as follows:

1. Version Number: The version number of the computation-data file format. The version number enables the computation-data file parsing software to recognize the characteristics of the current computation-data file and ensures backward compatibility. This field is set to 16 bits, with the first byte designating the main version number and the subsequent byte indicating the revision number to the main version.
2. File (Structure) Name: This field records the name of the computation-data file. Users may submit multiple computation-data files to the ternary optical computer (TOC), and each file may be renamed, split, or merged according to the TOC’s computational rules during transmission and use. Therefore, to ensure that the computation results files returned by the TOC correctly correspond to the submitted files, this field has been established.
3. IP Address: This field records the IP address of the user’s end that submits the computation-data file, facilitating the return of the computation results file. Currently, this field is set to 48 bits to record an IPV4 address. In subsequent versions, if an IPV6 address is adopted, this field must be expanded to 128 bits.
4. Priority: Indicates the priority of the computation-data file. The specific priority strategy is designed by the system administrator, and this field is reserved for technical support for priority management.
5. Reserved 1: This bit is reserved to facilitate the addition of priority bits or data type bits in future versions. It is set to 1 bit to make the fields 3), 4), 5), and 6) form two bytes, which are the lowest two bytes in 32 bits or 64 bits.
6. Data Type: Indicates the type of data contained in the computation-data file. Currently, there are only two types: structured data type and simple data type. Therefore, this field is currently set to 2 bits, with 00 representing simple data type, 01 representing structured data type, and 10 and 11 reserved for future use.
7. Computation Identifier Count: The number of valid computation identifiers contained in this computation-data file. This field is set to enable the ternary optical computer’s task management software to correctly interpret all computation identifiers, and it is stipulated that computation identifiers must be used sequentially. This version of the computation-data file has set 1280 computation identifiers. Hence, the field is set to 11 bits.
8. Computation Identifier (Item) 1 Name: This field is set to facilitate the referencing of the original data and computation results corresponding to this computation identifier through instructions.
9. Computation Identifier (Item) 1 Character: Stores the values of the three English characters used in the ternary logic operation of computation identifier (item) 1. Therefore, this field is set to 24 bits. To adapt to the underlying operations of the ternary optical computer, it is stipulated that the internal code of the first character is 00, the second character is 01, and the third character is 10. The character meanings for the remaining 1279 computation identifiers (items) are the same.
10. Computation Identifiers 1-1280: This field provides the operational rules of each calculator to be reconstructed by the ternary optical computer and the characteristics of the corresponding original data. Each computation identifier is set to 128 bits.

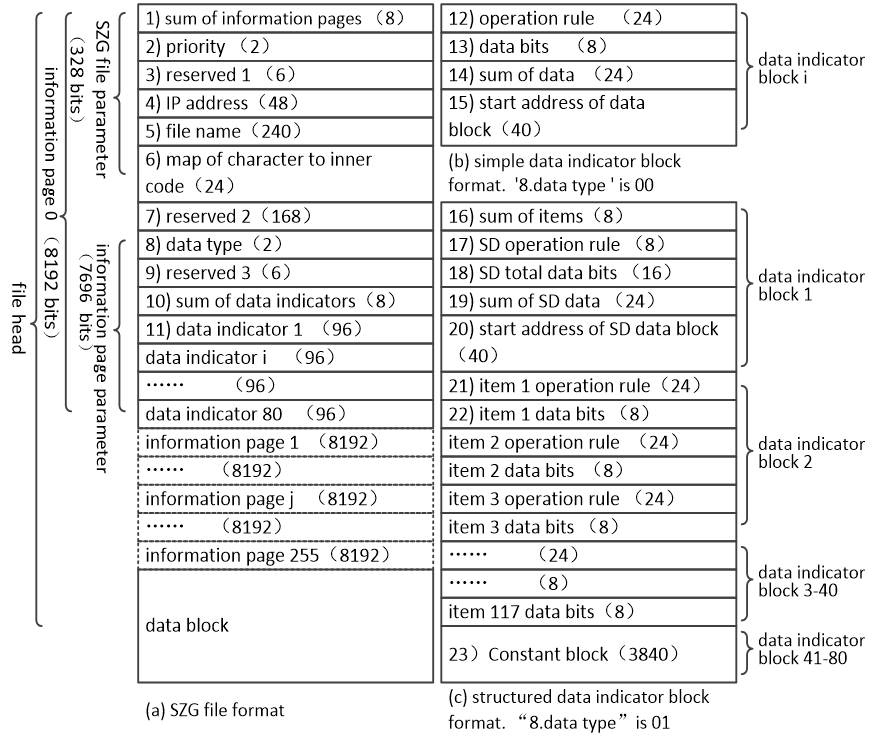


Figure 2: Schematic Diagram of the Computation-Data File Format

## Ternary Optical Computer Data Processing Method

Based on the hardware structure and computational characteristics of the ternary optical computer (TOC), a data processing method for the TOC is proposed, utilizing the computation-data file. The flowchart is shown in Figure 3.flowchart is shown in Figure 3.

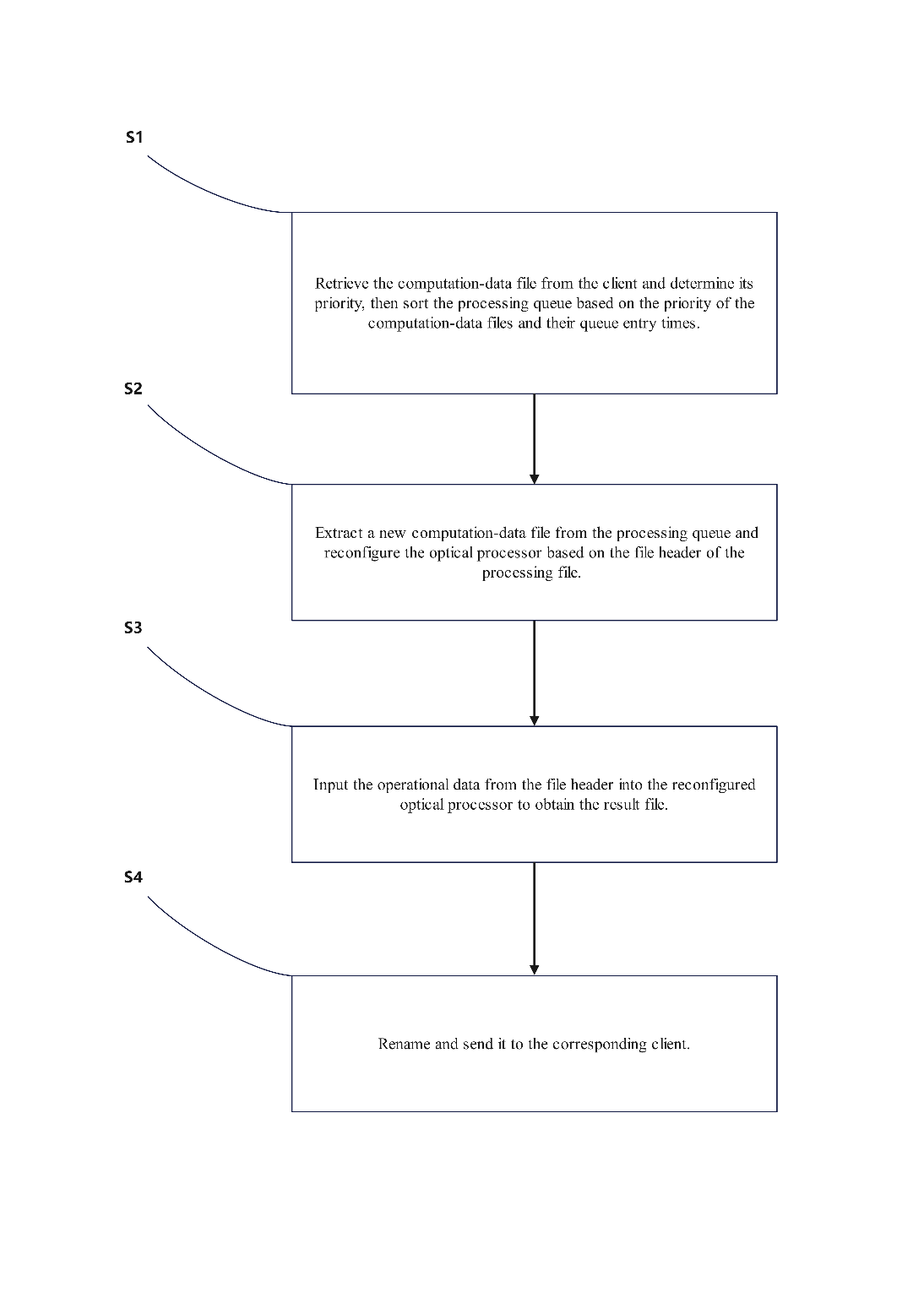
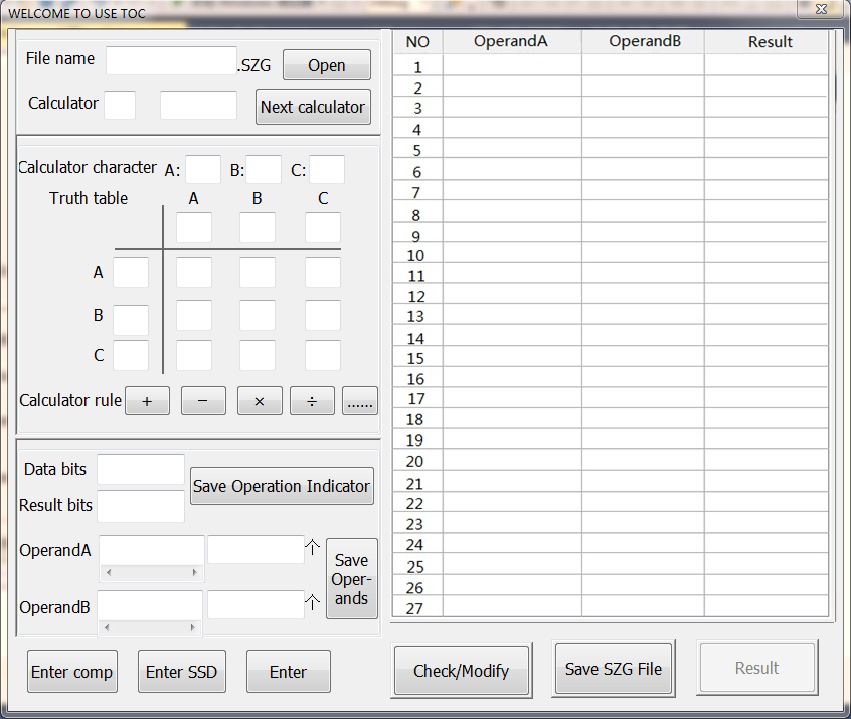


Figure 3: Flowchart of Ternary Optical Computer Data Processing Method

Based on the data processing method flowchart, the following processing scheme is implemented:

1. Users’ computational rules and original data are organized into a computation-data file, which is specifically formatted and constructed based on the operational characteristics of the ternary optical computer (TOC).
2. Specialized software assists users in creating the computation-data file. As depicted in Figure 4, this software provides a straightforward interface, requiring users to input only their computational requirements and original operands. The software then generates the corresponding computation-data file in line with the users’ intentions, operating on an auxiliary PC.
3. The computation-data file is transmitted to the TOC by extending high-level language instructions, allowing users to employ the TOC through applications written in high-level languages.

Figure 4: Computation-Data File Generation Software

**Efficacy Experiment of Data Processing in Ternary Optical Computers**

## Experimental Cases

The upper machine uses an electronic computer to generate computation-data files, while the slave machine uses SD16 to process these files, as shown in Figure 5. The experimental case includes calculations of four simple data types, which are ( f1 = a + b ), ( f2 = c - d ), ( f3 = e \land g ), and ( f4 = h \lor i ). Here, the independent variables ( a ) and ( b ) are 17-bit decimal numbers, ( c ) and ( d ) are 12-bit decimal numbers; ( e ) and ( g ) are 19-bit ternary data with the range {X, Y, Z}; ( h ) and ( i ) are 20-bit ternary data with the range {M, N, P}. Each function has 2000 pairs of data. Due to space limitations, this paper only lists the first two sets of test cases for the four functions, as shown in Table 1, where the MSD number ‘2’ in Table 1 represents “-1”.

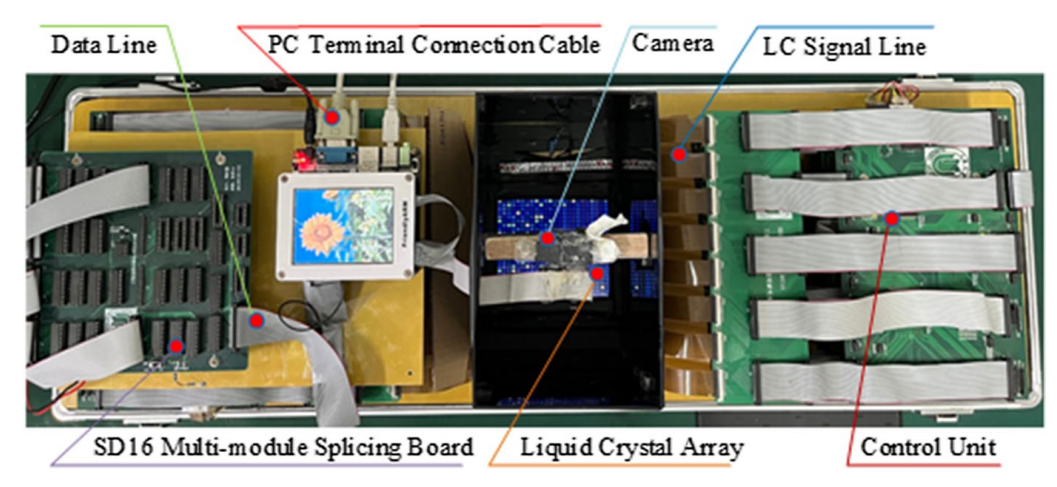


Figure 5:Prototype of SD16

Table 1. Test cases

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Item** | ***f1*** |  | ***f2*** |  | ***f3*** |  | ***f4*** |
| 1 | *a*(21121111122021122)*M* |  | *c*(122210112021)*M* |  | *e*(1001011010111001010)*M* |  | *h*(11111100101011010101)*M* |
| *b*(22122111110220121)*M* |  | *d*(120121201012)*M* |  | *g*(1010101001010100110)*M* |  | *i*(10011111001010100100)*M* |
| 2 | *a*(20022111110220121)*M* |  | *c*(120010112021)*M* |  | *e*(1001011010111001111)*M* |  | *h*(11101101101011010101)*M* |
| *b*(21001111122021122)*M* |  | *d*(120120001012)*M* |  | *g*(1010101001010100111)*M* |  | *i*(10010111001010100101)*M* |

## Experimental Procedure

1. Generating Computation-Data Files

To input operation rules and operands in the interface shown in Figure 4 to generate a computation-data file, follow these steps:

* 1. Press the ‘simple SD’ button in the interface. The computation-data file generation software will then fill in the “8) Data Type” field in Figure 2 with the identifier code “01” for simple structured data types.
  2. Enter the operation rules and data bit size for each component of the simple structured quantity one by one, i.e., for ( f1 ), ( f2 ), ( f3 ), and ( f4 ), press the ‘Enter Comp’ button after each component, and after all components are entered, press the ‘Enter SSD’ button. When ‘Enter Comp’ is clicked for the first time, the software will fill in the “21) Item 1 Operation Rule and 22) Item 1 Data Bit Size” fields in Figure 2 with the information just entered by the user. When ‘Enter Comp’ is clicked for the ( i )-th time, the software will fill in the “Item ( i ) Operation Rule” and “Item ( i ) Data Bit Size” fields with the information just entered by the user.
  3. Continuously enter the structured quantity data, separating each component with a comma and each structured quantity with a semicolon. Each data entered is displayed on the right side of the interface and can be corrected.
  4. After pressing the ‘Enter’ key, the computation-data file generation software will fill the original data entered into the data area in Figure 2, thus creating a ternary optical computer computation-data file in the format X:\j\*.szg, where X is the storage disk specified by the user, j is the folder on disk X, \* is the name of the file, and .szg is the dedicated suffix for ternary optical computer computation data files.

1. Transferring Computation-Data Files to Ternary Optical Processors

After generating the \*.szg file with the operation rules and raw data, it is only necessary to send the computation-data file to the ternary optical computer at the appropriate time in the program. This experiment extends the C programming language with a set of instructions for sending the computation-data file to the ternary optical computer and waiting for the operation result file. This set of extended instructions has the following functions: establish a communication connection with the ternary optical computer; send the \*.szg file to the task management module of the ternary optical computer monitoring system; query whether the operation result file (\*-r.szg) has been returned or suspend the program, wake up when waiting for the operation result file to return; continue executing the program after the \*-r.szg file returns; disconnect the communication connection with the ternary optical computer. The extended statements for the C program in this experiment are as follows:

* 1. void SZG\_Init().Establishes the extended instruction environment for the ternary optical computer. It must be the first SZG class extended instruction in the program and only needs to run once in a program.
  2. int SZG(char \*path).Sends the computation-data file pointed to by the address pointer path to the ternary optical processor.
  3. int SZG\_SearchResult(char \*path).Queries the operation status of the computation-data file pointed to by the address pointer path on the ternary optical processor. A return value of "0" indicates that the operation result file \*-r.szg has been received; a return value of "1" indicates that the operation result file has not been received yet; a return value of "2" indicates that an error occurred on the ternary optical computer side while processing the computation-data file, and no result will be returned.
  4. void SZG\_Suspend().The current program is suspended until the operation result file is returned and the program is awakened.

1. Ternary Optical Processor Bit Allocation

Upon receiving the computation-data file, the ternary optical processor reconstructs the calculator based on the number of operator positions specified in the file. The first 86 processor positions are reconstructed into an adder, 87-147 construct a subtractor, 148-166 form a “logical AND” operator, and 167-186 form a “logical OR” operator. The ternary optical computer employs MSD addition, which is realized through the transformations T, W, T’, W,’ and T2. For a detailed introduction to MSD addition, please refer to reference [19]. The allocation of operator positions for these five transformations is shown in Table 2, and the allocation for the two logical operations is shown in Table 3.

Table 2. Processor bit allocation for adder and subtractor

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | ***T*** | ***W*** | ***T’*** | ***W’*** | ***T2*** |
| *f1* | 0−16 | 17−33 | 34−50 | 51−68 | 69−86 |
| *f2* | 87−98 | 99−110 | 111−122 | 122−134 | 135−147 |

Table 3. Processor bit allocation for logic operators

|  |  |
| --- | --- |
| ***f3* (Logic AND)** | ***f4* (Logic OR)** |
| 148-166 | 167−186 |

## Experimental Results and Analysis

The data processing method of the ternary optical computer constructed in this paper differs significantly from traditional electronic computers in that users no longer need to arrange the sequence of operations and the timing and direction of data transfer for each part of the processor with individual instructions. Instead, users only need to specify computational requirements and processor bit requirements and hand over all raw data to the management program of the ternary optical computer, which controls and allocates processor bit resources. The number of computations under different processors is shown in Figure 6, where Figure 6(a) shows the number of computations processed by an electronic computer. In an electronic computer, a computational instruction includes multiple specific calculation instructions, each controlling its operator once. Thus, a total of 8000 operations are required to complete the computation of 8000 pairs of scalar data. Figure 6(b) shows the number of computations when only one matching processor is reconstructed in an optical computer, requiring only 2000 repetitions of the computation instruction to complete all 2000 pairs of structured data, i.e., all 8000 pairs of scalar data. Figure 6© shows the case of reconstructing two optical processors, where the ternary optical computer reallocates another 186 bits to construct a second composite processor containing four operators, and only 1000 repetitions of the computation instructions are needed to complete the computation of these 8000 pairs of scalar data.

Generally speaking, a ternary optical computer has several thousand bits and can reconstruct dozens of simple structured quantity processors, requiring only dozens of repetitions of the computation instruction to complete the task. For example, in the SD11 experimental system of the ternary optical computer for application research, one basic module of the optical processor has 1024 bits, which can construct five such structured quantity processors. Therefore, SD11 only needs to execute 400 computation instructions to complete the computation task. The fewer the repetitions of the computation instruction, the less time and hardware are required for the computation process.

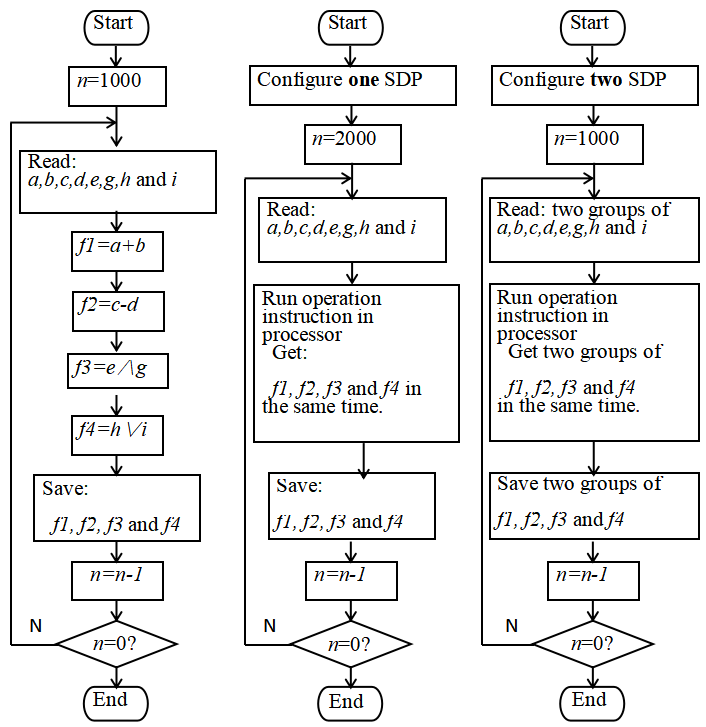


Figure6 The comparison of calculation processes between PC and TOC

Based on the above analysis, the comparison of computational performance between electronic computers and TOC under different processor bit conditions is shown in Table 4:

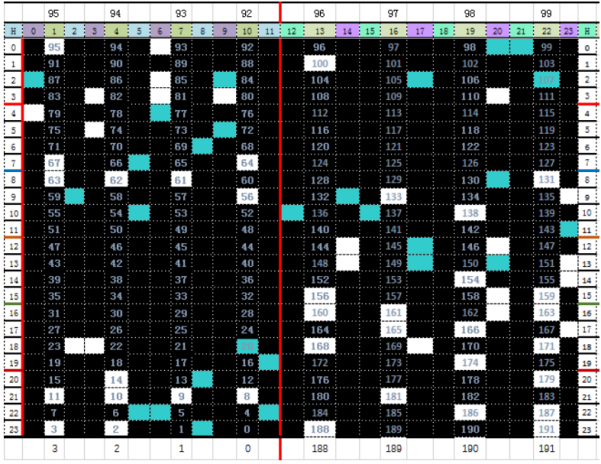
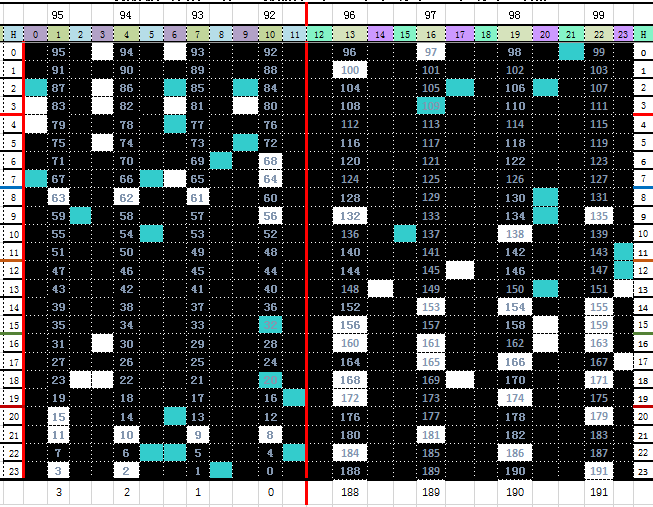
Table 4 Comparison of computing performance between PC and TOC

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Performance index** | **32-bit**  **electronic processor** | **64-bit**  **electronic processor** | **A structured**  **Data processor** | **5 structured**  **Data processors** |
| **Reconstruction time (*T0*)** | **0** | **0** | ***T0* (< 10 *Ty*)** | ***T0* (< 10 *Ty*)** |
| **Data bits (*W*)** | **4 x 32 = 128** | **4 x 64 = 256** | **186** | **1024** |
| **Operation cycles (*Ty*)** | **4 x 2000 = 8000** | **4 x 2000 = 8000** | **2000** | **⌈2000÷5⌉=400** |
| **Amount of resource (*W* x *Ty + T0*)** | **1024000** | **2048000** | **372000 *+ T0*** | **409600 *+ T0*** |

table5. Theoretical Values

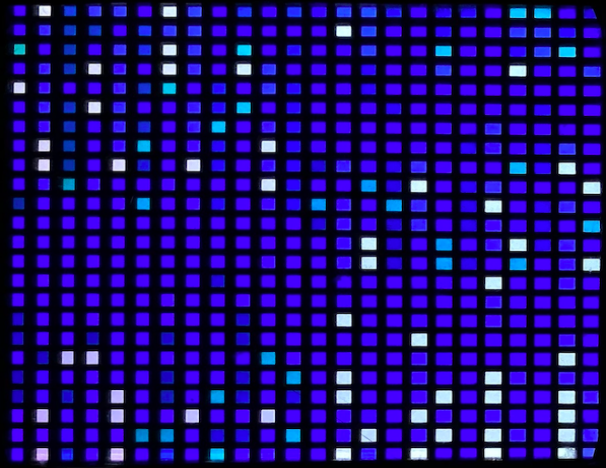
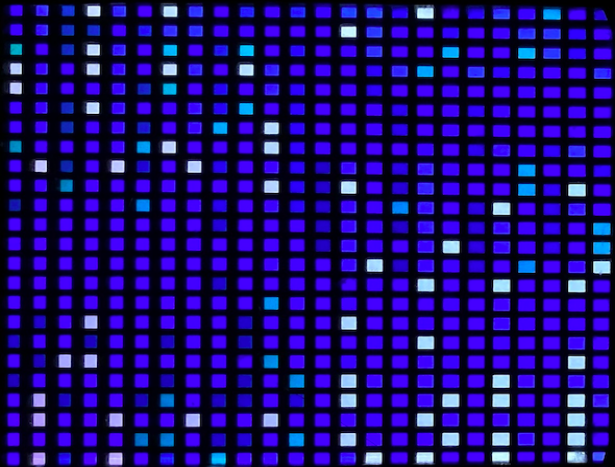
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Item** | ***f1*** |  | ***f2*** |  | ***f3*** | ***f4*** |
| 1 | (0201201111020010200)*M* |  | (01221012120000)*M* |  | (1011111011111101110)*M* | (10011100001010000100)*M* |
| 2 | (0212211111020010200)*M* |  | (01201201020000)*M* |  | (1011111011111101111)*M* | (10000101001010000101)*M* |

The theoretical result values of the experimental cases are shown in Table 5, the theoretical output diagram is shown in Figure 7, and the result diagram output by the ternary optical processor is shown in Figure 8. During the experiment, the diagrams output by the optical processor matched the theoretical values. To illustrate the first set of experimental cases for ( f1 ): Figure 8 shows the output results of the first set of experimental cases for ( f1 ), where the brightest point in the diagram represents vertically polarized light (V light state), the next brightest point represents horizontally polarized light (H light state), and the rest are no light state. The computation result shown in Figure 8(1) is decoded as ( (0201201111020010200)M = a + b = (-107124)D = (-16839)D + (-90285)D ), which is consistent with the theoretical result. The experiment demonstrates that the computation-data file proposed in this paper can correctly generate operation files that the ternary optical processor can calculate, and they are correctly executed.

(1) (2)

Figure7 Theoretical output results

(1) (2)

图8 TOC output results

# Conclusion

This paper presents a data processing method for ternary optical computers that leverages the hardware and computational characteristics of these systems. Using computation-data files, users’ computational rules and raw data are stored in a particular file format tailored to the ternary optical computer’s capabilities. These files serve as the operational objects for extended instructions in high-level languages, effectively shielding users from the complex underlying computational processes of the ternary optical processors and fully harnessing the computational advantages of ternary optical computers. Years of research and experimental results indicate that computation-data files are an effective data processing method in ternary optical computers. This data processing mechanism can be easily ported to other novel computer systems. By utilizing each processor’s respective computation-data files and corresponding extended instructions in programming languages, it is possible to achieve collaborative work among

**Funding**

This work was supported by the National Science Foundation of China (62302307, 62105207, 61866006), the Open Research Fund of Guangxi Key Lab of Human-machine Interaction and Intelligent Decision (GXHIID2209), and University- level general research project of Shanghai Normal University (SK202121).

**Data Availability Statement**

The data used to support the findings of this study are available from the corresponding author upon request.

**Conflict of Interest**

The authors declare that they have no competing interests.

**Acknowledgment**

Thanks to Prof. Jin Yi and all members from TOC research group for their suggestions and comments.

References**:**

1. Jin Y, He C H, Lu Y T. Ternary Optical Computer Architecture, PhysicaScripta. 2005, (T118): 98-101.
2. Jin Yi, He Hua-can, Ai Li-rong. Lane of parallel through carry in ternary optical adder [J]. Science in China (Series F), 2005, 48(1):107-116.
3. Yi J, Wang H J, Ouyang S, et al. Principles, structures, and implementation of reconfigurable ternary optical processor. Science China Information Sciences, 2011, 54(11): 2236-2246.
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. Ouyang S, Peng JJ, Jin Y, Shen YF, Liu XM, Han YX, Li WM. Structure and theory of dual-space storage for ternary optical computer. SCIENTIA SINICA Informationis, 2016, 46(6): 743- 762.
6. Jin Y, Wang X C, Peng J J, et al. Conceptual structure of ternary optical computer and high performance computer merger. HighPerform Comput Tech, 2010, 6: 1-4.
7. Wang Hong-Jian, Jin Yi, Ouyang Shan. Design and Implementation of a 1-bit Reconfigurable Ternary Optical Processor. Chinese Journal of Computers, 2014, 37(7): 1500- 1507
8. Jin Yi, Ouyang Shan, Song Kai, et al. Management of many data bits in ternary optical computers. Scientia Sinica Information Sciences, 2013. 43 (3): 361-373
9. Y. Shen, L. Pan, Y. Jin, J. Peng, and B. Jiang, One-step binary MSD adder for ternary optical computer, SCIENTIA SINICA Informationis, 2012, 42(7):869–881.
10. S. Li, J. Jiang, Z.Wang et al. Basic theory and key technology of programming platform of ternary optical computer[J], International journal for light and electron optics, Vol. 178: 327-336.
11. Zhang SL, Han YX, Shen YF, Jin Y. Principle of Computing Request File of Ternary Optical Computer. 2015 The Third international Conference on High Performance Computing and Applications. 2015.7, 150-157.
12. S. Li, Y. Jin, Simple structured data initial SZG file’s generation software design and implementation. 2016 The 3rd International Conference on Wireless Communication and Sensor Network. 2017, 2: 383-388.
13. H. Gao, Y. Jin, K. Song, Extension of C Language in Ternary Optical Computer, Jour-nal Shanghai University (Nature Science), 2013, 19(3):280–285.
14. Q. Zhang, Y. Jin, K. Song, and H. Gao, “MPI programming based on ternary optical in supercomputer”, Journal Shanghai University (Nature Science), 2014, 20(2):180–189.
15. S. Li, Theory and design of a three-value computer programming platform, Shanghai: Shanghai University, 2019.
16. Theoretical Basis and Implementation Mechanism of the Programming Platform for Ternary Optical Computer.IEEEAccess,2022, Vol.10:5585-5594.
17. JIN Yi, XU Qun, OUYANG Shan et al. Structured data computer—application characteristics of a ternary optical computer [J]. Sci China Ser F-Inf Sci, 2016, 46(3): 311-324.
18. S. Yunfu, W. Zhehe, P. Junjie and O. Shan, "Characteristics of Parallel Carry-Free Three-Step MSD Additions," in IEEE Access, 2021, vol. 9, 49601-49613.
19. J. Peng, R. Shen, Y. Jin, Y. Shen, S. Luo, Design and implementation of modifified signed-digit adder, IEEE Trans. Comput. 63 (5) (2014) 1134–1143.
20. Y. Shen, B. Jiang, J. Peng, Y. Jin, S. Ouyang, and J. Peng, “Principle and design of ternary optical accumulator implementing M-k-B addition”, Optical Engineering, vol. 53, no. 9, pp. 095–108, 2014.
21. X. Wang, J. Peng, M. Li, Z. Shen, and S. Ouyang, “Carry-free vector-matrix multiplication on a dynamically reconfifigurable optical platform”, Applied optics, vol. 49, no.12, pp.2352–2362, 2010.
22. Yan J Y, Jin Y, Zuo K Z. No carry & no borrow n-value operate unit. China Patent, ZL200710041144.1, 2009-10-28.
23. Jin Y, Ouyang S, Peng J J, et al. Reconfigurable tri-value optical processor. China Patent, ZL201010584129.3, 2012-05-02.
24. Jin Y.Management Strategy of Data Bits in Ternary Optical Computer. J. Shanghai Univ. (Nat. Sci.) 2007(05):519-523.
25. J. Peng, R. Shen, Y. Jin, Y. Shen, S. Luo, Design and implementation of modifified signed-digit adder, IEEE Trans. Comput. 63 (5) (2014) 1134–1143.
26. J. Peng, L. Teng, and Y. Jin, Realization of a tri-valued programmable cellular automata with ternary optical computer, International Journal of Numerical Analysis & Modeling, 2012, 2(9): 304–311.
27. Avizienis A. Signed digit number representation for fast parallel arithmetic. IRE Trans Electron Comput, 1961, 10: 389–400.
28. Zhang SL, Han YX, Shen YF, Jin Y. Principle of Computing Request File of Ternary Optical Computer. 2015 The Third international Conference on High Performance Computing and Applications. 2015.7, 150-157.
29. S. Li, Y. Jin, Simple structured data initial SZG file’s generation software design and implementation. 2016 The 3rd International Conference on Wireless Communication and Sensor Network. 2017, 2: 383-388.
30. Zhehe Wang and Yunfu Shen "Design and implementation of bitwise parallel MSD square rooting in ternary optical computer," Optical Engineering,2021, 60(8):085-107.
31. Y. Shen, L. Pan, Y. Jin, J. Peng, and B. Jiang, One-step binary MSD adder for ternary optical computer, SCIENTIA SINICA Informationis, 2012, 42(7):869–881.
32. H. Gao, Y. Jin, K. Song, Extension of C Language in Ternary Optical Computer, Jour-nal Shanghai University (Nature Science), 2013, 19(3):280–285.
33. Q. Zhang, Y. Jin, K. Song, and H. Gao, “MPI programming based on ternary optical in supercomputer”, Journal Shanghai University (Nature Science), 2014, 20(2):180–189.
34. S. Li, Theory and design of a three-value computer programming platform, Shanghai: Shanghai University, 2019.
35. Jiang J B ,Zhang X F, Shen Y F, Ouyang S,et al..Design and Implementation of SJ-MSD Adder in Ternary Optical Computer. ACTA ELECTＲONICA SINICA,2021, 49(02): 275-285.
36. Junjie P , Youyi F , Xiaofeng Z , et al. Implementation of DFT application on ternary optical computer[J]. Optics Communications, 2018, 410:424-430.