中图分类号：TP302.1　　单位代号：10280
密　　级：　　学　　号：15721297

上海大学

硕士学位论文

SHANGHAI UNIVERSITY
MASTER'S DISSERTATION

题目	三值光学计算机 MSD 乘法第三方平台设计与实现

作　　者　孔 帅
学科专业　计算机系统结构
导　　师　彭俊杰
完成日期　2018 年 1 月

姓　　名：孔帅　　学号：15721297

论文题目：三值光学计算机 MSD 乘法第三方平台设计与实现

上海大学

本论文经答辩委员会全体委员审查,确认符合上海大学硕士学位论文质量要求。

答辩委员会签名：

主任：

委员：

导　　师：

答辩日期：

原 创 性 声 明

本人声明：所呈交的论文是本人在导师指导下进行的研究工作。除了文中特别加以标注和致谢的地方外，论文中不包含其他人已发表或撰写过的研究成果。参与同一工作的其他同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示了谢意。

签　名：_____日 期：_____

本论文使用授权说明

本人完全了解上海大学有关保留、使用学位论文的规定，即：学校有权保留论文及送交论文复印件，允许论文被查阅和借阅；学校可以公布论文的全部或部分内容。

（**保密的论文在解密后应遵守此规定**）

签 名：_____导师签名：_____日期：_____

上海大学工学硕士学位论文

三值光学计算机 MSD 乘法第三方平台设计与实现

姓　　名：孔帅

导　　师：彭俊杰

学科专业：计算机系统结构

上海大学计算机工程与科学学院

2018 年 1 月

A Dissertation Submitted to Shanghai University for the

Degree of Master in Engineering

Design and Implementation of Third Platform for MSD Multiplication on Ternary Optical Computer

MA Candidate：Kong Shuai

Supervisor：Peng Junjie

Major：Computer System Architecture

School of Computer Engineering and Science

Shanghai University

January, 2018

摘　要

光自身的特点与优势，使得光学计算机相对于电子计算机具有更多的优势，如：高并行性、有更多的物理状态、处理器位数众多等。三值光学计算机是一种新型的、光电混合的计算机系统，它使用光的三种状态来表达三值信息，更确切的说，它采用偏振方向上相互正交的两个偏振光状态和无光态来表达三值信息。随着三值光学计算机发展，提出许多重要的理论，构建了许多原型系统。近些年，针对三值光学计算机的应用进行了大量研究。但是支撑原型系统的软件系统正在设计与发展，这要求用户参与三值光学计算机上的实验，必须要了解三值光学计算机工作原理和实现细节，给用户应用三值光学计算机增加了难度，限制了三值光学计算机的发展。

考虑到面向三值光学计算机应用的研究有很多，乘法运算是其他运算如快速傅里叶变换，矩阵向量乘法等的基础运算之一，所以研究 modified signed-digit（MSD）乘法在三值光学计算机上的实现。同时，为了降低三值光学计算机应用的难度，使得用户能够更好地关注应用本身，而不必了解三值光学计算机的工作原理和实现细节，本文设计并实现了三值光学计算机 MSD 乘法应用第三方平台。应用该平台，用户只需输入运算请求包括运算类型、操作数和运算精度等，该平台根据运算请求，自动生成完成乘法运算必要的信息，并将生成的信息格式化，传送给三值光学计算机的下位机，等待运算结束，将结果返回给用户。

本文主要从 MSD 乘法的并行实现及相应第三方平台的实现过程开展了大量研究，主要贡献如下。

（1）针对乘法运算过程中部分积生成和部分积求和进行了讨论，并结合三值光学计算机的特点给出乘法不同实现方案以及相应处理器位的分配策略。

（2）分析了三值光学计算机 MSD 乘法并行实现过程中所需要的信息，给出了 MSD 乘法第三方平台实现机制、上位机与下位机的工作原理及实现细节。

（3）结合三值光学计算机 MSD 乘法第三方平台实现原理，针对三值光学

计算机 MSD 乘法第三方平台进行了设计并实现。

（4）为了保证 MSD 乘法并行运算连续的完成，给出了数据回馈策略。

（5）基于设计的 MSD 乘法第三方平台进行了大量实验，实验结果表明 MSD 乘法并行实现和 MSD 乘法第三方平台实现机制正确可行，使用该平台，不需要用户参与，就可以完成乘法运算，从而为光学计算机的推广与普及奠定了基础。

关键词：三值光学计算机，modified signed-digit，MSD 乘法，MSD 乘法第三方平台

ABSTRACT

Characteristics and advantages of light make optical computer have many advantages over its electronic counterpart traditional computer, which includes high parallelism, high information capability, many processor bits and so on. As a new typical computer system, ternary optical computer (TOC) is a photoelectric hybrid computer system. It uses three states of light to express information, which includes two orthogonal polarizations and dark state of light. With the development of TOC, many important principles have been established and prototypes have been built. In recent year, more and more research has focused on the study of the applications on TOC. However, software system to support the prototype is still under design and development. This causes that the experiments on TOC need participation of users. And users have to be familiar with the working principle and implementation detail of TOC before utilization of the system, which brings great difficulty for users to utilize TOC and much limits the development of TOC.

Considering there might be many different kinds of applications that can be implemented on TOC, and multiplication is one of the most basic operations in other numerical operations, such as fast Fourier transform, matrix-vector multiplication and so on, modified signed-digit (MSD) multiplication on TOC is studied. At the same time, in order to decrease the difficulty of applying ternary optical computer, and make users focus on the application instead of knowing working principle and implementation detail of TOC, third platform for MSD multiplication on TOC is designed and implemented. Using the platform, users only need to input operation parameters that include operation type, operands and operation precision etc. And the platform automatically generates the necessary information that is needed in multiplication on TOC according to the operation request. And it sends the information formatted to the slave computer of TOC. Meanwhile, it answers for

sending back the results to the users when the operation on TOC is accomplished.

Generally, the thesis is mainly focus on the parallel implementation of MSD multiplication and third platform for MSD multiplication on TOC. The main contributions of the study are as follows.

(1) Generation of partial product and the sum of partial products in the process of multiplication are discussed. And different implementation schemes of multiplication based on the characteristics of TOC are proposed, and corresponding allocation strategies of processor bits are presented.

(2) The information needed in the parallel implementation of MSD multiplication on TOC is analyzed. And the implementation mechanism of third platform for MSD multiplication on TOC is proposed. Meanwhile, the working principle and implement detail of master computer and slave computer in TOC are presented.

(3) Third platform for MSD multiplication on TOC is designed and implemented according to the principle of third platform for MSD multiplication.

(4) In order to ensure that MSD multiplication is implemented automatically and continuously, data feedback strategy in multiplication is presented.

(5) Extensive experiments have been carried out based on the designed third platform for MSD multiplication. Experimental results show that parallel implementation of multiplication and the implementation mechanism of third platform for MSD multiplication on TOC are feasible and correct. With the platform, MSD multiplication can be implemented without needing participation of users. This lays the foundation for promotion and popularization of TOC.

Keywords: ternary optical computer, modified signed-digit, MSD multiplication, third platform for MSD multiplication

目　　录

第一章　绪论

1.1 课题来源

本课题来源：（1）编号为 No.61572305 的国家自然科学基金项目“三值光学计算机乘法器及计算机例程平台关键技术研究”；（2）编号为 No.61103054 的国家自然科学基金项目“基于冗余数的光学并行加法器研究”；（3）中国航天科工集团二院的创新基金。

1.2 课题研究的目的和意义

随着数值计算应用的计算复杂度增加，对电子计算机的计算效率提出了更高要求，虽然目前电子计算机可以满足应用计算效率的需求，但是电子计算机存在的问题日益突出：硬件结构复杂、高能耗等。而且在特定的数值计算领域如密码学、地质勘探等，数据处理等方面的需求对计算机提出了更高的要求，而另一方面，半导体材料的物理极限、散热及高能耗等多方面的影响使得电子计算的发展在很大程度上受到了制约。这迫使人们寻求新的解决方案，因此针对新型计算机进行了研究，比如量子计算机[1,2]、生物计算机[3]以及光学计算机[4-10]等。

光自身的特点与优势使得光学计算机相对于电子计算机具有更多的优势：高并行性，有更多的物理状态，处理器位数众多[11-13]，低能耗等。因此许多研究学者从不同角度研究光学计算机，并试图构建全光计算机系统[9,10]。然而，由于目前光学计算机相关技术、部件尚不成熟，如光学存储器[14]、光学芯片[15]等，因此真正的全光学计算机还未问世[16]。

基于该现状，金翊等提出了新型的光电混合计算机系统结构-三值光学计算机，并给出了三值光学计算机的原理和结构[6-8]。它将光的偏振方向和光强结合来表达信息，更确切的说，它采用偏振方向上相互正交的两个偏振光状态和无光态来表达三值信息。它利用液晶和偏振片构成运算器完成光学运算，使用电子计算机进行运算控制、数据解码和数据存储。因此光处理、电控制是三值光

学计算机的突出特点。自三值光学计算机原理和结构提出以来，在理论研究和原型系统方面发展顺利。在三值光学计算机理论研究方面代表性的成果有：降值设计理论[17]、可重构理论[18-21]、数据位管理理论和技术[22]、三值光文件即 SZG 文件和双空间存储器[23]等。同时在原型系统方面取得重大进展，比较有代表性的是 2011 年研制的面向应用研究的实验系统，简称上大 11 即 SD11。

课题组基于已有的理论和 SD11 原型系统对于数值运算进行了大量研究。由于加法运算是数值运算的基础，而且传统的加法运算存在进位延时的问题，因此前期的研究主要集中在加法运算方面，解决加法运算过程中进位延时的问题。其实在光学计算领域对加法进位延时问题早就有研究，代表性的有符号数算法[24]，修改的符号数算法[25,26]和符号替换算法[27]。课题组对于加法进位延时问题也进行大量研究。2003 年，金翊等提出进位直达理论[28]，但是由于物理设备的限制并没有实现；2009 年，金翊等提出 MSD 加法原理和结构[29]；此后针对 MSD 加法进行了大量研究，集中体现在两个方面：MSD 加法效率[30-34]和 MSD 加法光路结构[35,36]。现在 MSD 加法以及加法器的研究趋于成熟，因此基于 MSD 加法针对其他数值运算也进行了大量的研究，比如乘法例程[37,38]，除法例程[39]，矩阵向量乘[40]，快速傅里叶变换运算[41]等，这些数值运算充分发挥光学计算的优势。但是这些研究都没有完整的底层软件支撑系统，这使得大量的实验过程中需要人工参与，如 MSD 加法过程中数据回馈过程。这意味着用户在应用三值光学计算机时，必须要了解三值光学计算机工作原理和实现细节，这给用户应用三值光学计算机增加了难度，限制了三值光学计算机的发展。

考虑到乘法运算是数值运算中最为常见的应用之一，又是其他运算如快速傅里叶变换、矩阵向量乘法等最为广泛应用的基础之一。本文以用户为中心，以三值光学计算机为研究背景，针对 MSD 乘法，设计了连接用户和三值光学计算机的桥梁即第三方平台。应用该平台，用户只需输入运算请求包括操作数和运算精度等，该平台根据运算请求，自动生成完成乘法运算必要的信息，并将生成的信息格式化，传送给三值光学计算机的下位机解析，等待运算结束后，将结果返回给用户。同时该平台使得乘法运算可以连续完成，提高了运算效率。此外，本文研究的 MSD 乘法第三方平台为其他应用第三方平台的设计与实现

奠定了基础。

1.3 国内外研究概况

1.3.1 电子乘法器

在电子计算机发展的过程中，乘法器的理论研究和实现已经比较成熟。而且它已经成为电子计算机中非常重要的一部分。对于电子乘法器的研究主要体现在提高乘法计算效率、降低能耗、减小占用面积、结构优化等方面[42-51]。

提高乘法计算效率的研究主要集中在减少部分积的个数和提高部分积求和速度。在减少部分积个数方面，代表性成果是 1951 年，Booth A D 提出的 Booth 编码算法[45]，主要思想是通过对乘法操作数进行重新编码，达到减少部分积个数的目的，最终提高部分积求和的速度。此后的研究主要是针对 Booth 编码算法改进，衍生出许多 Booth 变种算法，其核心思想和 Booth 编码算法一样[46, 47]。在提高乘法器的部分积求和速度方面，主要是通过提高部分积求和的并行度来提高乘法的运算效率，其中代表性的有 Wallace 树型结构[48]、ZM 树阵列[49]和 OS 树阵列[50]等。这些部分积求和阵列旨在提高部分积求和的并行度，进而减少部分积求和的时间。

还有一些学者将减少部分积个数和提高部分积求和速度结合起来，提高乘法器的计算效率，如孙振玮通过结合 Booth 编码算法和 Wallace 树型结构，设计了一款 18 位乘法器，提升了乘法器运算效率[51]。

1.3.2 光学计算机

由于光自身的特性与优势使得光学计算相对于电子计算有很多优势，在电子计算机发展阶段，很多研究学者就发现这点，并且在此领域开展了大量研究，取得了许多研究成果。经过长期的研究与探讨，目前国际上多数研究学者一致认为，构造“全光计算机”的设想是不现实。因此，这个领域的研究主要集中在光电混合型计算机。

目前在此领域的主要分支有三个[52]。第一个分支是：充分发挥光学码元的

高速率特性，欲用高速的光学多稳态器件（包括双稳态）取代电子双稳态器件，沿着电子计算机的理论、结构和发展轨道发明光学计算机，最好是全光计算机。其工作重点在于寻找或发明某种高速节能的光学新材料、用新材料构造高速光学多稳态新器件或逻辑功能器件。代表性成果主要有：1989 年由 AT&T 贝尔实验室提出的采用自电光效应器件的逻辑门阵列[53]；2004 年华中科技大学提出一种基于级联半导体光放大器中的交叉增益调制效应实现的全光逻辑与门的新方案[54]等。该研究分支的难点在于研制开发比较理想的新材料或发明令人满意的新器件比较困难。第二个分支是：试图充分发挥光的空间并行性，利用光学图像处理的二维信息并行变换方式来完成数值和逻辑的并行运算。其工作重点在于发明新型高效的空间调制器和空间滤波器。代表性的成果主要有：光学阵列逻辑的应用[55]，基于“符号替换”的光学计算机结构[56]等。该研究分支目前面临的主要难点在于精确数字计算的方法及实现技术。第三个分支是：以计算机的基本原理来考察光的物理特征，以现有电子计算机技术为基础，研发能充分发挥光学优势的计算机系统，从利用光学计算的个别优势逐步发展到全面发挥更多的光学优势，其工作重点在于探索能发挥光学特点的计算机理论和结构。该分支以三值光学计算机[6-8]研究为代表，主要成果有：（1）理论研究方面：降值设计理论[17]、可重构理论[18-21]、数据位管理理论和技术[22]、三值光文件即 SZG 文件和双空间存储器[23]等。（2）原型系统方面：2007 年研制的 360 位三值光学计算机原理实验系统、2009 年研制的千位三值光学计算机结构实验系统（配有指令和初级管理软件，用户可以联机使用）以及 2011 年研制的面向应用研究的上大 11 实验系统等。

1.3.3 光学乘法器

许多学者对光学乘法器的设计实现进行了大量研究。最早是 1970 年，S. H. Lee 等人提出使用光学计算的方法实现向量-矩阵乘法（optical matrix-vector multiplication，OVMM），并从数学上证明了使用光学方法进行向量-矩阵乘法的可行性。后来人们在研究 OVMM 时提出了一些并行光学体系结构[57,58]，但它们都是光学模拟系统。Carter III 和 Pape 等人研究设计了光学向量矩阵协处理

器，可以用于处理 8 维向量与 8×8 矩阵的乘法，吞吐率为 1MHz[59]；Davis 等人采用液晶作为空间调制器对光学计算进行了研究，通过控制光的相位、幅度或相位和幅度的组合来实现光学乘法运算[60]。国内在此领域的研究也很多，如：张锐和李修建等人将数学中的编码方法与光学原理相结合，利用光折变晶体的四波混频效应，提出了一种可实现矩阵-矩阵的混合负二进制编码的光学系统[62]；中国科学院光电技术研究所的杨林等人提出了一种由激光调制器阵列、多路复用器、分路器、微环调制器和光电探测器阵列构成的芯片上的光学矩阵向量乘法器[63]；中科院半导体所卢洋洋等采用维度为 16×16 实现了一套基于空间向量-矩阵乘法运算的光电混合数字信号处理系统，能够实现乘法累加运算[64]；国防科大的聂永名利用光的相干特性，结合光学设计原理，设计建构了基于双数字微镜的光学矢量矩阵乘法器[65]。所有这些研究基本上都是集中在向量矩阵乘法器的实现上，并且多集中在光强、光的相干特性等模拟方法上。

在三值光学计算机的乘法器方面，胡晓俊从例程开发角度探讨了三值光学计算机 40 位整数乘法例程[66]，蒋本朋等人从限制输入方面探讨了限制输入三值光学连加法器的设计与实现[67]，沈戎以改进加法效率为主要突破口探讨了三值光学计算机乘法器相关的关键技术[68]。

1.3.4 第三方平台

电子计算机自产生以来，人机交互越来越友好。虽然面向电子计算机的应用有很多，但是用户只需输入运算请求，不需要关注电子计算机内部结构和实现细节，电子计算机各个部件协同工作完成运算，将运算结果返回给用户，因此电子计算机的用户群不断变大。随着光学计算机不断发展，其理论体系不断健全，原型系统逐渐成熟，开展了大量面向光学计算机的应用研究，但是这些属于模块化研究，需要了解光学计算机的工作原理和实现细节，给用户应用光学计算机增加了难度。第三方平台是连接用户和光学计算机的桥梁，使用户无需关心光学计算机工作原理和实现细节，同时使运算能够自动完成。平先顺等对第三方平台进行了初步探索[69]，为第三方平台的研究奠定了基础。

1.4 论文的主要研究内容

本论文共分为六章，具体内容如下。

第一章 绪论，本课题的来源、本课题的研究目的和意义，以及国内外研究现状。

第二章 MSD 乘法第三方平台的理论基础，介绍了三值光学处理器，MSD 数字系统及其特点，并基于该数字系统介绍了 MSD 加法的并行运算规则和 MSD 乘法运算原理。

第三章 MSD 乘法第三方平台实现机制，首先讨论了 MSD 乘法运算部分积生成方案、部分积求和方案，给出 MSD 乘法运算方式和处理器位的分配策略。接着给出了三值光学计算机第三方平台运算实现的原理，设计了三值光学计算机第三方平台。考虑到光学处理器的特点，选择了 MSD 乘法并行实现方案，结合三值光学计算机系统，分析了其实现过程所需的信息，给出了 MSD 乘法第三方平台的实现机制。

第四章 MSD 乘法第三方平台的设计与实现，给出了下位机的 M 变换器结构、针对三值光学计算机目前的处理器位数，对控制信息文件进行具体设计、给出 MSD 乘法第三方平台实验用例的上位机与下位机实现方法。

第五章 实验验证，给出了三值光学计算机 MSD 乘法第三方平台实现的实验环境，对实验结果进行了讨论与分析，并将三值光学计算机 MSD 乘法实现的计算延时与不同乘法器的计算延时进行了对比。

第六章 结论与展望，对三值光学计算机 MSD 乘法第三平台研究内容进行总结，并针对三值光学计算机第三平台的研究工作进行展望。

第二章　MSD 乘法第三方平台的理论基础

三值光学计算机遵循着电子计算机的设计思想产生，它将光强和光的偏振方向结合来表达三值信息。由于光学计算机的一些关键技术处于研究和发展中，比如光学存储器、光学芯片等，所以三值光学计算机目前不是全光学计算机。它借用电子计算机部件实现三值光学计算机部分功能。因此，电控制、光处理是它的突出特点之一。自三值光学计算机概念提出以来，经历了快速的发展，时至今日，各种基本理论及关键模块的建立特别是几代三值光学计算机原型系统的研制，为三值光学计算机 MSD 乘法第三方平台的研究与实现奠定了硬件基础。此外，三值光学计算机采用 MSD 数字系统，该数字系统和光的三种物理状态：水平偏振光，垂直偏振光和零光强相对应。并且基于该数字系统对加法运算进行研究，消除加法运算过程中的串行进位问题，实现加法并行运算，这就是 MSD 并行加法。这些研究为三值光学计算机 MSD 乘法第三方平台研究与实现奠定了理论基础。本节主要对三值光学计算机处理器，MSD 数字系统，MSD 并行加法和 MSD 乘法原理进行了详细介绍。

2.1 三值光学处理器

三值光学计算机是一种新型、光电混合的计算机体系结构[6, 7, 8]，其处理器架构如图 2-1 所示，主要包括：编码器，TOC 处理器，解码器，重构器，任务管理软件和用户程序。图 2-1 中的黑色细线表示自然光线，黑色虚线表示三态光：水平偏振光、垂直偏振光和无光，黑色粗线表示数据传输。编码器的作用将二值的电信号转换成三种光信号，即通过电子计算机的二值信号控制液晶的旋光性配合偏振片调制出三种光的状态：水平偏振光，垂直偏振光和无光。重构器的作用是根据重构命令在 TOC 处理器中重构相应的逻辑运算器，同时将主光路编码数据写入主光路编码寄存器。TOC 处理器的作用是完成光学运算，TOC 处理器的运算基元是典型的三明治结构，由两个偏振片和一层液晶组成，根据运算基元主光路两个偏振片的偏振方向，运算区被划分成 4 个部分：HH 区，

HV 区，VV 区和 VH 区，每个运算区的液晶编号是相同的，液晶被分为左右两个部分，左半部分液晶编号是从上到下，从左到右的顺序，右半部分的液晶编号是从下到上，从右到左，具体分布如图 2-2 所示。TOC 解码器的作用是将 TOC 运算的结果即三种光信号转换成电子信号，并将该转换值传送给 TOC 任务管理软件，由 TOC 任务管理软件返回给用户。TOC 任务管理软件负责用户请求处理，数据组织和数据传输。

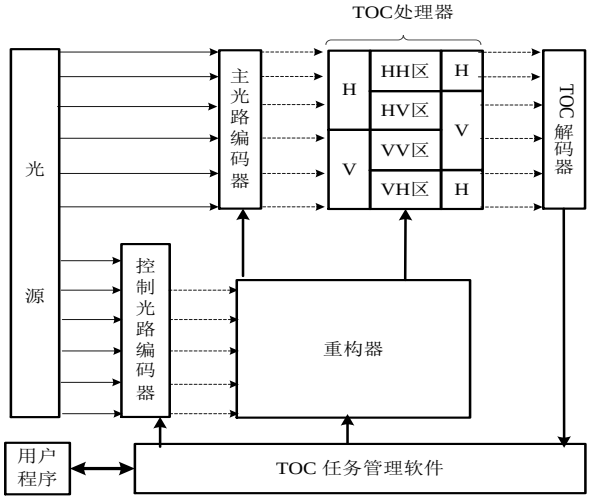

图 2-1 三值光学处理器架构

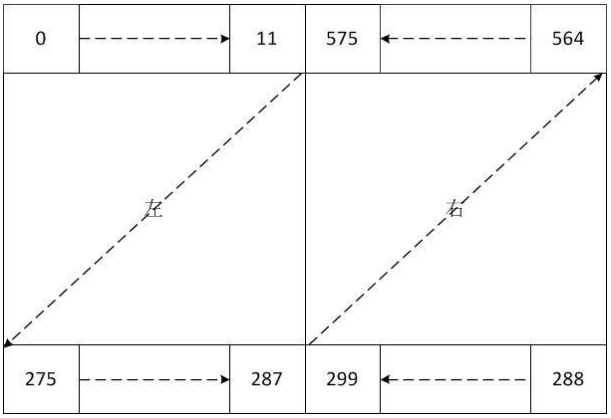

图 2-2 三值光学处理器运算区液晶编号

2.2 MSD 数字系统

传统电子计算机采用只有 0 和 1 的二进制进行运算，而且任意的十进制数都有唯一的表达形式。而三值光学计算机采用 MSD 数字系统，该数字系统表示一种带符号位的三值二进制数，使得任意的十进制数具有多种表达形式，这

为无进位加法的实现提供理论基础。1961 年，A. Avizienis 等人提出 MSD 数字系统[70]，并在 1986 年被 Drake 等人引入了光学计算领域[71]。它的符号集为{u，0，1}，其中 u 表示数值-1，任意的一个十进制数 D 能够用 MSD 数字系统表达，表达式如下：

$$D = \sum_i d_i \times 2^i \qquad (2\text{-}1)$$

公式（2-1）中 d_i 的取值符号集为{u，0，1}，其中 i 表示整数。2^i 表明这个数字系统仍然是二进制数字系统。该数字系统相对于传统的二值二进制具有如下特点。

（1）将 MSD 数的每一位都进行取反即得其相反数。取反规则是 1 变成 u，u 变成 1，0 保持不变。比如十进制 3 用 MSD 数字系统可以表示为：$(10u)_{MSD}$，对其按位取反，后得到：$(u01)_{MSD}$，将其转换成十进制是-3。

（2）最高位为正数是正数，反之是负数；比如 $(10u)_{MSD}$ 表示 3，$(u01)_{MSD}$ 表示-3。

（3）具有冗余性，即一个十进制数具有多种表达形式；$(3)_D=(10u)_{MSD}=(1u0u)_{MSD}=(1uu0u)_{MSD}$。

（4）正数和负数用同一表示形式。

（5）数值 0 的表示形式唯一。

由于三值光学计算机的三种物理状态和 MSD 数字系统的三个符号一一对应，因此课题组将 MSD 数字系统引入三值光学计算机，同时该数字系统的冗余性，解决了加法运算过程中串行进位问题。

2.3 MSD 并行加法

三值光学计算机的 MSD 加法是指基于 MSD 数字系统，使用四种三值逻辑运算表（T（T2），W，T'，W'）来完成 2 个 MSD 数相加的运算，真值表如表 2.1 所示，其中 T2 变换和 T 变换相同。表 2.1 中 a，b 分别表示 1 位的 MSD 数。由于 MSD 数字系统的冗余特性，不论加数和被加数的位数是多少，完成一次 MSD 加法运算只需 3 个时钟周期，即 MSD 加法运算的时钟周期独立于加数和被加数的位数。在运算时，如果加数和被加数位数不相等，需要在位数少的操作数

高位补 0，使两个操作数的位数相等，然后进行运算。假设加数和被加数的位数相等，被加数 A 有 n 位，记 $A=a_{n-1}\ldots a_i\ldots a_1a_0$，其 MSD 表达式 A=$\sum a_i\times 2^i$，$i=0,1,...,n-1$；加数 B 有 n 位，记 $B=b_{n-1}\ldots b_j\ldots b_1b_0$，其 MSD 表达式 B=$\sum b_j\times 2^j$，$j=0,1,...,n-1$，$a_i$，$b_j$ 符号集均为{u，0，1}，其中 n 和 m 是正整数，i 和 j 是自然数，$0\le i\le n-1$，$0\le j\le n-1$。MSD 加法并行运算过程如下。

（1）对输入的 A 和 B 按位进行 T，W 运算：T 运算的结果低位补 0，W 运算的结果高位补 0；

（2）对第一步的结果按位进行 T'，W'运算，T'运算的结果低位补 0，W'运算的结果高位补 0；

（3）对第二步的结果按位进行 T2 运算，得到最终结果，即 A 和 B 的和。

表 2.1 MSD 加法真值表

a	b	T(T2)	W	T'	W'
u	u	u	0	u	0
u	0	u	1	0	u
u	1	0	0	0	0
0	u	u	1	0	u
0	0	0	0	0	0
0	1	1	u	0	1
1	u	0	0	0	0
1	0	1	u	0	1
1	1	1	0	1	0

MSD 加法的并行性主要体现在两个方面：

（1）由于 MSD 数加法运算过程中的结果各数据位之间相互独立，所以单个逻辑变换可以并行完成。比如 T 变换：$T(a_{n-1},b_{n-1})$，$T(a_{n-2},b_{n-2})$，……，$T(a_i,b_i)$，……，$T(a_0,b_0)$可以并行完成。

（2）由 MSD 加法运算原理可知，不同变换可以并行执行：比如 T 和 W 变换，T'和 W'变换。

2.4 MSD 乘法原理

三值光学计算机 MSD 乘法运算是基于 MSD 数字系统使用 M 变换和 MSD 加法来完成的。其中 M 变换是指两个 1 位 MSD 数的逻辑变换，可以完成 1 位的

MSD 数乘法运算，其真值表如表 2.2 所示。表 2.2 中，a，b 分别表示 1 位的 MSD 数。设被乘数 A 有 n 位，记 $A=a_{n-1}\ldots a_i\ldots a_1a_0$，其 MSD 表达式 A=$\sum a_i\times 2^i$，$i=0,1,...,n-1$；乘数 B 有 m 位，记 $B=b_{m-1}\ldots b_j\ldots b_1b_0$，其 MSD 表达式 B=$\sum b_j\times 2^j$，$j=0,1,...,m-1$，$a_i$，$b_i$ 符号集均为 {u，0，1}，其中 i 和 j 表示自然数，$0\le i\le n-1$，$0\le j\le m-1$，n 和 m 表示正整数。因此 A×B 可表示为：

$$C=A\times B=A\times\left(\sum b_j\times 2^j\right)=\sum A\times b_j\times 2^{j-1}=\sum S_j\times 2^j=\sum P_j \qquad (2\text{-}2)$$

从公式（2-2）可知，部分积 $S_j=A\times b_j$，和数项 $P_j=S_j\times 2^j$。运算过程可分为以下 4 个步骤。

第 1 步：由乘数 B 的每一位 b_j 复制成长度为 n 的辅助数据 B_j，记为 $B_j=b_j\ldots b_j\ldots b_jb_j$（共 n 位），总共得到 m 个新数据 $B_0,B_1,\ldots,B_j,\ldots,B_{m-1}$。

第 2 步：将 A 与每个 B_j 按位进行 M 变换，变换的结果为 m 个部分积 $S_0,S_1,\ldots,S_j,\ldots,S_{m-1}$。

第 3 步：将每个部分积 S_j 左移 j 位，即在每个 S_j 后面补 j 个 0，得到 m 个和数项 $P_0,P_1,\ldots,P_j,\ldots,P_{m-1}$。

第 4 步：对 m 个和数项 $P_0,P_1,\ldots,P_{m-1}$ 用 MSD 加法进行求和，累加的结果就是乘积 C。

表 2.2 M 变换真值表

b	a		
	u	0	1
u	1	0	u
0	0	0	0
1	u	0	1

2.5 本章小结

本章针对三值光学计算机 MSD 乘法第三方平台的理论基础进行介绍，主要包括三值光学计算机处理器，MSD 数字系统及其特点，并基于 MSD 数字系统，介绍了 MSD 加法的并行运算规则以及 MSD 乘法原理。

第三章　MSD 乘法第三方平台的实现机制

由于光的自身特点与优势，使得光学计算机相对于电子计算机具有一定的优势，但是用户应用三值光学计算机，必须要了解其工作原理和实现细节，这给用户应用三值光学计算机增加了难度，限制了三值光学计算机的发展。针对该现状，本文设计了第三方平台来连接用户与三值光学计算机，使用户应用光学计算机更加方便。由于乘法运算是众多数值运算的关键步骤之一，比如矩阵向量乘，傅立叶变换，神经网络等，所以本章以 MSD 乘法为突破口，以三值光学计算机为研究背景，给出了 MSD 乘法第三方平台的实现机制。

3.1 MSD 乘法

MSD 乘法实现包括部分积生成和部分积求和两个部分，因此本节针对部分积的实现方案，部分积求和的实现方案及 MSD 乘法的实现方案进行了讨论。设被乘数 A 的位数有 n 位，小数位数有 k 位，记 $A=a_{n-1}\ldots a_i\ldots a_1a_0$，其 MSD 表达式 A=$\sum a_i\times 2^{i\text{-}k}$，$i=0,1,...,n-1$；乘数 B 有 m 位，小数部分有 s 位，记 $B=b_{m-1}\ldots b_j\ldots b_1b_0$，其 MSD 表达式 B=$\sum b_j\times 2^{j\text{-}s}$，$j=0,1,...,m-1$，$a_i$，$b_j$ 的符号集均为{u，0，1}，其中 n 和 m 表示正整数，$1<m\le n$，k，s，i 和 j 均表示自然数，$0\le i\le n-1$，$0\le j\le m-1$，$0\le k\le n$，$0\le s\le m$。

3.1.1 部分积的实现方案

依据 MSD 乘法原理，部分积是进行 M 变换的结果，部分积 S_j 的表达式如公式（3-1）所示。

$$S_j=U_{j(n-1)}U_{j(n-2)}U_{j(n-3)}...U_{j2}U_{j1}U_{j0},\quad 0\le j\le m-1 \tag{3-1}$$

由公式（3-1）可知，部分积 S_j 是一个 n 位的 MSD 数，它的位数和被乘数

的位数相同。部分积S_j中每位的取值是$U_{ji}=M(a_i,b_j)$，$0\le i\le n-1$，$M(a_i,b_j)$表示 a_i 和 b_j 经过 M 变换的值，a_i 表示被乘数中的第 i+1 位，b_j 表示乘数中的第 j+1 位，$M(a_i,b_j)$取值如公式（3-2）所示。

$$M(a_i,b_j)=\begin{cases}1, & a_i=1,\ b_j=1\ or\ a_i=u,\ b_j=u\\ 0, & a_i=0\ \text{or}\ b_j=0\\ u, & a_i=1,\ b_j=u\ or\ a_i=u,\ b_j=1\end{cases} \tag{3-2}$$

部分积相邻数据位之间，高位无需等待低位的进位，所以不同数据位之间的 M 变换可以并行完成。对于部分积生成，当只有 1 个 M 变换器时，需要 m 个时钟周期，当有 m 个 M 变换器时，需要 1 个时钟周期，因此 M 变换完成的时钟周期 T 的取值范围：$1\le T\le m$。

3.1.2 部分积求和的实现方案

得到部分积后，需要对部分积进行移位操作，得到和数项，然后对和数项求和，得到乘积，移位规则如表 3.1 所示。

表 3.1 部分积的移位规则

部分积	移位位数	和数项
S_0	0	P_0
S_1	1	P_1
S_2	2	P_2
……	……	……
S_j	j	P_j
……	……	……
S_{m-1}	m-1	P_{m-1}

表 3.1 中，S_j 表示第 j+1 个部分积，j 表示自然数。得到部分积后，需要对部分积求和，求和公式（3-3）所示。

$$C=P_0+P_1+P_2+...+P_j+...+P_{m-1} \tag{3-3}$$

公式 3-3 中 P_j 表示和数项，对于部分积的求和，有如下方案：

（1）依据 MSD 乘法原理对于部分积求和，可以采用顺序求和的方案。

顺序求和，首先计算 P_0+P_1，然后使用 P_0+P_1 的结果和 P_2 相加，累加至 P_{m-1}，得到乘积。这种方式下，需要计算的时钟周期 T1 如公式（3-4）所示。

$$T1=3\times(m-1) \tag{3-4}$$

部分积求和的第 2 种方案和第 3 种方案都是基于二叉迭代求和理论，使用该理论实现部分积求和，能够减少部分积求和的等待时间，提高乘法运算效率，其原理如图 3-1 所示，二叉迭代求和需要进行$\lceil \log_2 m \rceil$层 MSD 加法运算。图 3-1 中，Add_{11} 表示第 1 层第 1 个 MSD 加法器，依次类推。tj 表示第 j 层 MSD 加法器的个数：tj=$\lceil m/2^j \rceil$，$1\le j\le\lceil \log_2 m \rceil$。第一层 MSD 加法器的输入是 M 变换结果经过处理后的数据，如果 M 变换的结果个数为奇数，需要添加一个值为 0 的 MSD 数使操作数的个数成为偶数，第 j 层 MSD 加法器的输入是第 j-1 层 MSD 加法结果经过处理过的数据，$2\le j\le\lceil \log_2 m \rceil$，如果第 j-1 层的 MSD 加法结果的个数是奇数，需要添加一个值为 0 的 MSD 数使操作数的个数成为偶数。

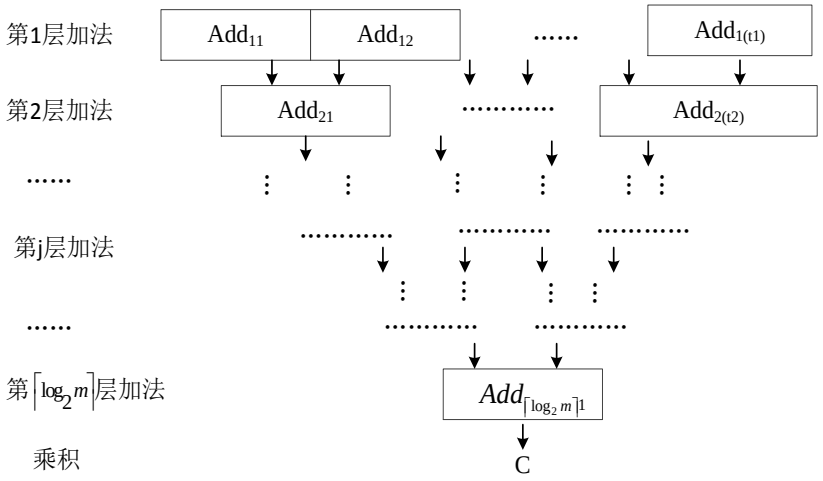

图 3-1 二叉迭代求和的原理

（2）依据 MSD 加法原理结合二叉迭代求和的思想，部分积求和可以采用流水线实现方案，所谓流水线实现方案是指每一层的 MSD 加法使用流水线的方式实现，其原理如图 3-2 所示。从图 3-2 可以看出 3 个时钟周期出运算结果，此后每个周期得到一个运算结果，因此使用该方法计算部分积求和时，第 1 层 MSD 加法所需的时钟周期数 T2 如公式（3-5）所示。

$$T2=2+\lceil m/2 \rceil \tag{3-5}$$

二叉迭代求和需要$\lceil \log_2 m \rceil$层 MSD 加法，部分积求和所需时钟周期总数 T3 如公式（3-6）所示。

$$T3=\sum_{i=1}^{\lceil \log_2 m \rceil}(2+\lceil m/2^i \rceil) \tag{3-6}$$

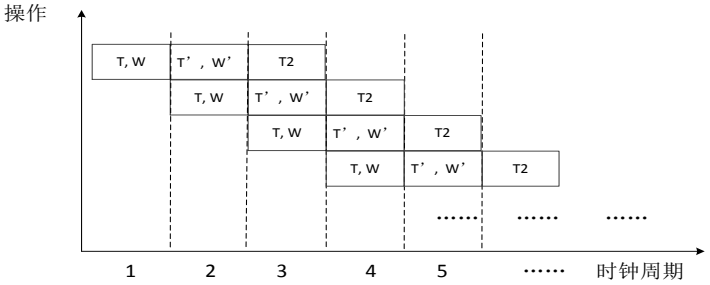

图 3-2 部分积求和流水线实现方案原理

（3）依据 MSD 加法原理结合二叉迭代求和的思想，部分积求和可以采用并行方案，所谓并行实现方案是指每一层 MSD 加法使用并行方式实现，其原理如图 3-3 所示。从图 3-3 可以看出第一层 MSD 加法所需的时钟周期是 3，二叉迭代求和需要$\lceil \log_2 m \rceil$层 MSD 加法，其时钟周期数 T4 如公式（3-7）所示。

$$T4=3\times\lceil \log_2 m \rceil \tag{3-7}$$

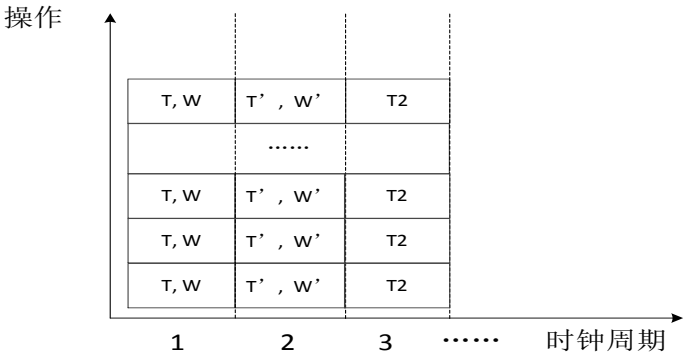

图 3-3 部分积求和并行实现方案原理

表 3.2 部分积求和方案时钟周期

和数项个数	实现方式		
	方案 1	方案 2	方案 3
2	3	3	3
4	9	7	6
8	21	13	9
16	45	23	12
32	93	41	15
64	189	75	18
……	……	……	……

根据以上讨论的部分积求和方案，给出相应方案所需的时钟周期，如表 3.2 所示。表 3.2 中的数据表明，三种部分积求和方案所需的时钟周期和和数项个数呈正相关，即时钟周期数随着和数项个数的增加而变大，且方案（2）和（3）的时钟周期比方案（1）的时钟周期具有一定的优势，这种优势会随着和数项个数的增加而更加明显。

3.1.3 MSD 乘法实现方案

根据 3.1.1 和 3.1.2 的讨论与分析，该部分提出 MSD 乘法的实现方案：流水线实现方案和并行实现方案。针对具体的方案，分析了处理器位的分配情况。

MSD 乘法的这两种实现方案中，部分积求和都是基于二叉迭代求和理论，因此构造 MSD 加法器的两种典型方案。

（1）根据每层 MSD 加法的操作数位数，分配各逻辑变换器的处理器位，然后重构 MSD 加法器，此时需要重构$\lceil \log_2 m \rceil$次 MSD 加法器。

（2）根据最后一层 MSD 加法的操作数位数分配处理器位，此时只需要重构 1 次 MSD 加法器。

由于每次重构运算器需要花费一定的时间，所以本文选择第二种方案来重构 MSD 加法器。

根据第二种方案，分析单个 MSD 加法器的处理器位使用情况。首先根据 MSD 乘法运算原理得到，m 个部分积，每个部分积的长度是 n，根据表 3.1 的移位规则，得到和数项，和数项中最大的长度是 n+m-1，将所有和数项的长度按 n+m-1 对齐，由 MSD 加法原理可知，MSD 加法结果的位数比操作数的位数多 2 位，而二叉迭代求和需要$\lceil \log_2 m \rceil$层 MSD 加法，所以最后一层 MSD 加法操作数的长度是$L = n + m - 1 + 2 \times (\lceil \log_2 m \rceil - 1) = n + m + 2 \times \lceil \log_2 m \rceil - 3$，因此构造单个 MSD 加法器，T 和 W 变器换所需的处理器位是 L，T'和 W'变换器所需的处理器位是 L+1，T2 变换器所需的处理器位是 L+2，所以单个 MSD 加法器需要处理器位总数是 L+3。

（1）流水线实现方案。乘法的流水线实现主要是通过构造两个 M 变换器

和 MSD 加法器实现，其原理如图 3-4 所示，总共需要处理器位数是 2×n+5×L+4。假设处理器位的起始地址是 p，处理器位的分配情况如表 3.3 所示。

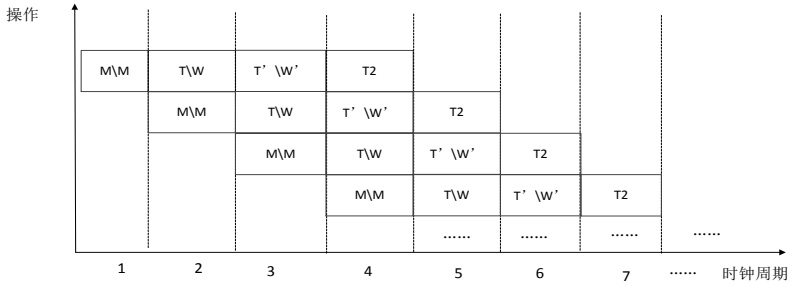

图 3-4 MSD 乘法流水实现方案原理

表 3.3 流水线实现方案处理器位的分配情况

运算器		处理器位区间
第一个 M 变换器		p~(p+n-1)
第二个 M 变换器		(p+n)~(p+2×n-1)
MSD 加法器	T 变换器	(p+2×n)~(p+2×n+L-1))
	W 变换器	(p+2×n+L)~(p+2×n+2×L-1)
	T'变换器	(p+2×n+2×L)~(p+2×n+3×L)
	W'变换器	(p+2×n+3×L+1)~(p+2×n+4×L+1)
	T2 变换器	(p+2×n+4×L+2)~(p+2×n+5×L+3)

（2）并行实现方案。乘法并行方案是通过构造 m 个变换器和$\lceil m/2 \rceil$个 MSD 加法器实现，原理如图 3-5 所示，因此总共需要处理器位数是$\lceil m/2 \rceil \times (5 \times L + 4) + n \times m$，处理器位的分配情况如表 3.4 所示。

MSD 乘法并行性主要体现在以下几个方面。

（2.1）在 MSD 乘法实现过程，M 变换可以并行执行的，所谓并行执行是不同数据位之间的并行执行。

（2.2）在 MSD 乘法实现过程，部分积的生成是可以并行执行的。

（2.3）在 MSD 乘法实现过程，部分积求和采用二叉迭代求和，每层 MSD 加法可以并行执行。

（2.4）在 MSD 加法过程中，每个逻辑变换是可以并行执行，所谓并行执行是不同数据位之间的并行执行。

（2.5）由 MSD 加法运算原理可知：不同变换可以并行执行：比如 T 和 W 变换，T'和 W'变换。

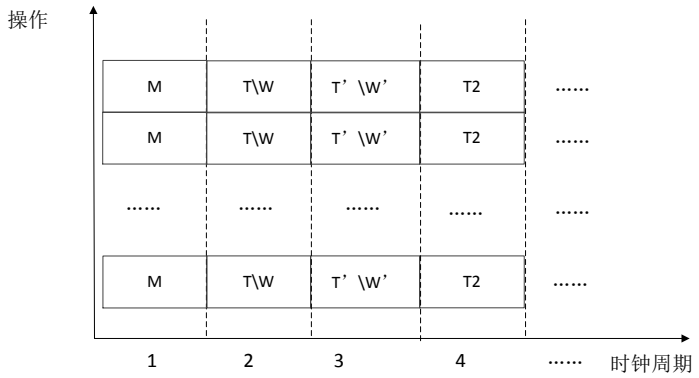

图 3-5 MSD 乘法并行实现方案原理

表 3.4 并行实现方案处理器位的分配情况

运算器	处理器位区间
第一个 M 变换器	p~(p+n-1)
第二个 M 变换器	(p+n)~(p+2×n-1)
……	……
第 k 个 M 变换器	(p+(k-1)×n)~(p+k×n-1)
……	……
第 m 个 M 变换器	(p+(m-1)×n)~(p+n×m-1)
第一个 MSD 加法器	(p+n×m)~(p+n×m+5×L+3)
第二个 MSD 加法器	(p+n×m+5×L+4)~(p+n×m+10×L+7)
……	……
第 j 个 MSD 加法器	(p+n×m+(j-1)×(5×L+4))~(p+n×m+j×(5×L+4)-1)
……	……
第⌈m/2⌉个加法器	(p+n×m+(⌈m/2⌉-1)×(5×L+4))~(p+n×m+⌈m/2⌉×(5×L+4)-1)

3.2 三值光学计算机第三方平台

随着三值光学计算机理论研究不断完善，原型系统不断成熟，针对一些应用进行了大量研究。但是这些属于模块化验证研究，需要人工参与。同时，用户应用三值光学计算机，必须要了解光学计算机工作原理和实现细节，这给用户应用三值光学计算机增加难度，限制了三值光学计算机的发展。针对该现状，本文设计了连接用户与三值光学计算机的第三方平台，用户只需给出运算信息，然后等待三值光学计算机回送结果即可，如图 3-6 所示。

根据该指导思想，分析与研究了三值光学计算机第三方平台，结合 MSD 乘法，设计并实现了三值光学计算机 MSD 乘法第三方平台。

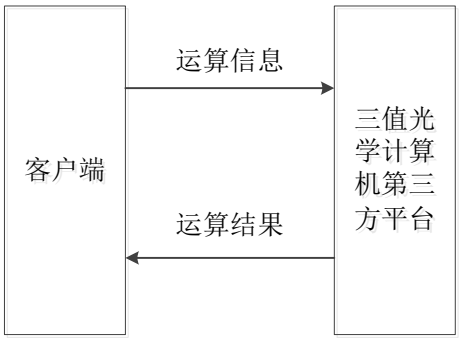

图 3-6 三值光学计算机第三方平台

3.2.1 三值光学计算机第三方平台原理

三值光学计算机数值运算的研究是从运算本身的角度，比如：乘法、除法、矩阵向量乘和快速傅里叶变换等，结合光学计算机的特点，研究其可行性、光路结构和运算效率，但总体来说没有从系统和用户的角度考虑，相反需要用户了解三值光学计算的工作原理和实现细节，而本文正是从三值光学计算机系统和用户的角度，考虑数值运算实现的整个过程，从而提高光学计算机易用性。在此指导思想下，设计了三值光学计算机第三方平台，其工作原理如图 3-7 所示，图 3-7 中，虚线框表示三值光学计算机第三方平台。工作原理具体表述如下。

（1）用户提出运算请求，给出待处理运算的运算类型、操作数和运算精度，由系统根据用户运算请求的信息自动生成完成运算所需的信息，主要包括：运算类型、操作数长度、操作数的小数位数、逻辑变换器的处理器位起始地址、逻辑变换器所需处理器位数和主光路数据与控制光路数据等信息，此前这些信息是需要用户自己参与设定的，而在本文设计的 MSD 乘法第三方平台中将自动实现。

（2）上位机对生成的运算信息进行格式化，并将格式化的数据封装成文件，由上位机的通信模块传输给三值光学计算机的下位机。

（3）下位机接收到文件后，按照事先预定的规则对文件进行解析，同时根据给定的规则，引导光学处理器完成相应的运算。

（4）当运算完成后，下位机将运算结果返回给上位机的通讯模块，上位机

对结果进行处理后返回给用户。

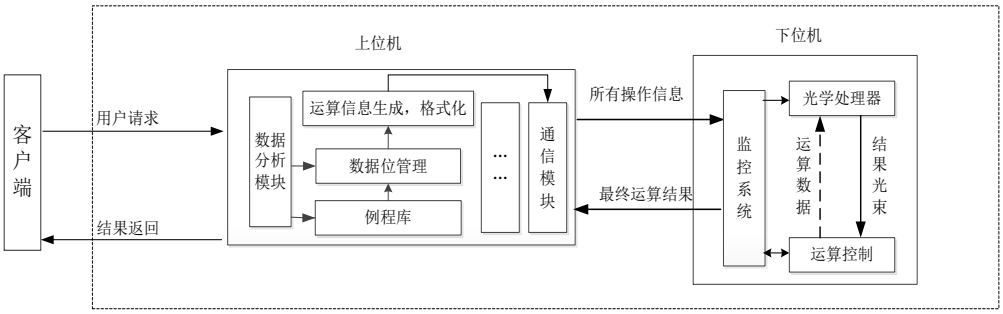

图 3-7 三值光学计算机第三方平台工作原理

3.2.2 三值光学计算机第三方平台运算信息

表 3.5 三值光学计算机第三方平台运算信息

控制信息区	
运算类型	
操作数信息	操作数 1 的位数
	操作数 1 的小数位数
	操作数 2 的位数
	操作数 2 的小数位数
	操作数 3 的位数
	操作数 3 的小数位数
	……
	操作数 n 的位数
	操作数 n 的小数位数
变换器的个数	
运算器的处理器位起始地址以及所需处理器位情况	变换器 1 的处理器位的起始地址
	变换器 1 所需处理器位数
	变换器 2 的处理器位的起始地址
	变换器 2 所需处理器位数
	……
	变换器 n 的处理器位的起始地址
	变换器 n 所需处理器位数
……	
数据区	
控制光路的原始数据，根据三值光学计算的编码器对其编码	
主光路原始数据，根据三值光学计算的编码器对其编码	

由图 3-7 可知，三值光学计算机第三方平台的核心部分是：生成运算信息、运算信息的封装和运算信息文件的解析。针对不同的运算进行分析，提取了这些运算的公共信息，将其分为控制信息区和数据区两部分，控制信息区包括运

算类型、操作数、运算精度、运算器的个数、运算器的处理器位起始地址和所需处理器位数。数据区包括操作数编码的数据。关于控制信息区和数据区的内容见表 3.5。

将以上所有信息封装好，通过网络传输的方式发送给三值光学计算机的下位机，由下位机的解释程序进行解析，根据解析出来的有效数据，引导光学处理器完成运算，等待运算结束后，将运算结果返回给三值光学计算机的上位机，最终返回给用户。

3.3 MSD 乘法第三方平台的实现机制

由 3.1 的分析可知，MSD 乘法运算的实现包括部分积生成和部分积求和两个部分，其运算的时钟周期数和和数项的个数呈正相关，而和数项的个数由乘数的位数决定，所以乘数的位数越多，乘法运算的时钟周期越大。因此，在乘法实现的过程中，选择位数较少的操作数作为乘数，另一个作为被乘数。由于三值光学处理器的处理器位众多、易扩展、按位可分配和按位可重构的特点，所以本文选择 MSD 乘法并行实现作为第三方平台研究目标，给出了 MSD 乘法第三方平台的实现机制。该平台可以作为 MSD 乘法并行运算的专用平台。

3.3.1 MSD 乘法并行实现分析

根据三值光学计算机系统结合 MSD 乘法并行实现方案，分析了 MSD 乘法并行实现过程所需要的信息。

（1）文件类型：三值光学计算机下位机接收的文件不止一种类型，为了区分控制信息文件和其他文件，将控制信息文件的类型定为三值光文件，即 SZG 文件。

（2）运算类型“×”：在实际应用过程中，三值光学计算机处理的运算可能会有很多，因此，对于用户的乘法运算而言，必须标明运算类型，下位机才能控制光学处理器完成相应的运算。

（3）乘数：由 MSD 乘法的运算原理可知，根据用户提供的操作数，选择数据位数少的操作数作为乘数，这样能够减少部分积个数，进而减少部分积求

和时间，提高乘法运算效率。

（4）被乘数：两个操作数中选择数据位数少的操作数作为乘数后，剩下的一个作为被乘数。

（5）乘数的位数：乘数的位数决定了 M 变换和部分积求和的次数，下位机需要根据乘数的位数得到这些信息，控制 MSD 乘法运算，假定乘数的位数是 m，m 是大于 1 的正整数。

（6）乘数中小数的位数：根据用户需求设定，假定乘数中小数的位数是 s，$0 \le s \le m$。

（7）被乘数的位数：根据被乘数的位数进行处理器位分配，据此在光学处理器的相应位置构造出乘法运算所需的逻辑变换器，假定被乘数的长度为 n，n 是大于 1 的正整数。

（8）被乘数中小数的位数：根据用户需求设定，假定被乘数中小数的位数是 k，$0 \le k \le n$。

（9）乘积中的小数位数：由乘数中的小数位数和被乘数中的小数位数确定，假设 p 表示乘积中的小数位数，必须满足 $p = k \times s$。

（10）控制光路数据与编码：由于 MSD 乘法运算所需的逻辑变换真值表是对称的，因此对于控制光路的数据没有特别的要求。所以在部分积生成过程中，M 变换选择被乘数作为控制光路数据。在部分积求和过程中，第 1 层 MSD 加法，T 变换和 W 变换选择奇数项 M 变换的结果作为控制光路的数据，T'和 W'变换选择 W 变换结果作为控制光路数据，T2 变换选择 W'变换的结果作为控制光路数据。当 1 层迭代求和得不到乘积时，从第 2 层迭代求和开始，T 和 W 变换选择上一层奇数项 MSD 加法运算结果作为控制光路数据，其余不变。

表 3.6 逻辑符号与内码的对应关系

光的状态	符号表示	液晶控制信息号 1	液晶控制信号 2
垂直偏振光	u	0	0
水平偏振光	1	0	1
无光	0	1	0

控制光路数据需要转换成三值光学计算机能够处理的编码，从而调制出相应光的状态，参加运算，其中 MSD 数字符号与编码的对应关系如表 3.6 所示[72]。

由表 3.6 可知，经过编码后的数据，其数据位数变为原来数据位数的 2 倍。

（11）主光路数据与编码：由于 MSD 乘法运算所需的逻辑变换真值表是对称的，因此对于主光路的数据同样没有特别的要求。所以在部分积生成过程中，M 变换选择由乘数生成的辅助数据作为主光路数据。在部分积求和过程中，第一层 MSD 加法，T 变换和 W 变换选择偶数项 M 变换的结果作为主光路数据，T'和 W'变换选择 T 变换的结果作为主光路数据，T2 变换选择 T'变换的结果作为主光路数据。当 1 层迭代求和得不到乘积时，从第 2 层迭代求和开始，T 和 W 变换选择上一层偶数项 MSD 加法运算结果作为主光路数据，其余不变。

主光路的数据编码规则和控制光路的数据编码规则相同。

（12）乘数的复制：由 MSD 乘法的运算原理可知，需要将乘数从低位向高位的逐位复制生成辅助数据，辅助数据的位数和被乘数的位数相同，然后分别和被乘数按位做 M 变换得到部分积。如果将该操作放在上位机会增加控制信息文件的开销，主要体现在：增大控制信息文件的生成时间、增加控制信息文件的容量和增加上位机与下位机的通信时间。由于该操作是乘法特有的操作，因此将其作为下位机解释程序的固定部分，当解析到乘数即主光路数据时进行复制操作，以此减少控制信息文件的开销。

表 3.7 乘法运算过程中变换器的基元数量

变换器	所需要的基元数量			
	HH	HV	VH	VV
M	1	1	1	1
T	3	0	0	3
W	0	2	2	0
T'	1	0	0	1
W'	2	0	0	2
T2	3	0	0	3

（13）乘法运算的三值逻辑运算表：MSD 乘法运算的真值表是构造乘法运算器的依据。在乘法运算过程有 6 个逻辑运算，但实际只需要 5 个逻辑变换真值表，其中 MSD 加法过程中 T 变换和 T2 变换相同。根据 MSD 乘法运算所需的 5 个逻辑变换真值表，依据降值设计理论的分解定理得到运算基元表，根据基元表得到各逻辑变换器的重构命令，最终构造乘法器。乘法运算过程中各逻辑运算器所需基元数量如表 3.7 所示。由于乘法运算需要的重构命令和操作数

无关，如果进行多次乘法运算，每次生成重构命令写入控制信息文件，这会增加上位机和下位机的通信开销，因此提前将上位机生成乘法运算所需逻辑变换器的重构命令存储在下位机，当下位机解析到乘法运算符时调用重构命令即可。

（14）M 变换结果的移位：由 MSD 乘法的运算原理可知，M 变换结束后，得到部分积，需要对部分积进行移位，得到和数项。

（15）和数项的位数：和数项的位数是由乘数的位数、被乘数的位数和和数项求和方式决定的，也是 MSD 加法器的处理器位的分配依据。

（16）乘法运算方案：考虑到光学处理器的处理器位众多、易扩展、按位可分配和按位可重构，所以选择了 MSD 乘法并行的运算方案。

表 3.8 MSD 乘法并行实现处理器位的分配情况

运算器	处理器位区间
第 1 个 M 变换器	0~(n-1)
第 2 个 M 变换器	n~(2×n-1)
……	……
第 k 个 M 变换器	((k-1)×n)~(k×n-1)
……	……
第 m 个 M 变换器	((m-1)×n)~(n×m-1)
第 1 个 MSD 加法器	(n×m)~(n×m+5×L+3)
第 2 个 MSD 加法器	(n×m+5×L+4)~(n×m+10×L+7)
……	……
第 j 个 MSD 加法器	(n×m+(j-1)×(5×L+4))~(n×m+j×(5×L+4)-1)
……	……
第⌈m/2⌉个加法器	(n×m+(⌈m/2⌉-1)×(5×L+4))~(n×m+⌈m/2⌉×(5×L+4)-1)

（17）逻辑变换器所需处理器位数：由于逻辑变换器每次重构需要的时间较长，因此对于逻辑变换器的重构，采取整体重构策略，即一次构造出完成乘法运算所需的所有逻辑变换器。变换器的重构，需要计算其所需处理器位数，以便重构命令与相应的处理器位对应起来。根据 MSD 乘法并行实现方案处理器位的分配情况，需要根据被乘数和乘数的位数计算出构造单个逻辑变换器所需的处理器位数，即计算出构造单个 M，T，W，T'，W'和 T2 变换器所需的处理器位数，其中 T2 变换器和 T 变换器相同。前期的研究表明，M 变换器所占运算器的大小和被乘数的位数相关，MSD 加法过程中每个逻辑运算得到的数据的位数均与 M 变换结果的位数相关，而且每次 MSD 加法运算的结果位数比操作数的位数多 2 位。由（5）和（7）知乘数的位数为 m，被乘数的位数为 n，

由于该光学处理器作为 MSD 乘法专用处理器，所以默认第一个逻辑运算器的起始地址是 0，MSD 乘法并行实现的处理器位的分配情况如表 3.8 所示。

（18）乘法器的处理器位起始地址：乘法运算过程中，下位机需知道乘法器的处理器位起始地址即起始液晶编号，用变量 productLCM 表示乘法器中 M 变换的处理器位起始液晶编号。每个变换器所需的处理器位数与被乘数和乘数的位数有关，给定了乘法器的 M 变换的起始液晶编号 productLCM，便可推算出各个变换器的起始液晶编号，相应的也就可推算出在 FPGA 中逻辑运算区的每个变换器的起始地址。

（19）M 变换器的个数：为了取 M 变换器的处理器位的起始地址、所占处理器位的数和 M 变换的结果。

（20）MSD 加法器的个数：为了取 MSD 加法过程中各个变换器的起始地址、所占处理器位数和相应逻辑变换的结果。

（21）专用数据存储区：由于 MSD 乘法运算过程是无进位的逻辑运算，前面逻辑运算的结果需要经解码存储在专用数据存储区，经过数据位调整后送回至光学处理器的输入端，作为后续逻辑运算的输入，这个过程称为数据回馈。数据回馈过程主要包括：

（21.1）M 变换结果进行数据位调整后作为第 1 层 MSD 加法器的输入。

（21.2）在二叉迭代求和的过程中，相邻两层的 MSD 加法，对上一层 MSD 加法运算结果进行数据位调整，作为下一层 MSD 加法的输入。

（21.3）在每层 MSD 加法运算过程中，T，W 变换结果的数据位调整之后作为 T'，W'变换的输入，T'和 W'变换结果的数据位调整之后作为 T2 变换的输入。

（22）数据位调整：数据位调整包括补 0 和舍弃 0，补 0 和舍弃 0 中 0 是指编码后 0，0 对应的编码是 10，具体体现在以下几个方面。

（22.1）M 变换的结果按照表 3.1 的移位规则补 0。

（22.2）在二叉迭代求和过程，和数项的位数按照二叉迭代求和操作数的位数要求补 0。

（22.3）在二叉迭代求和过程中，相邻两层的 MSD 加法，上一层 MSD 加

法运算的结果舍弃高位的两个 0。

（22.4）在 MSD 加法过程中，T 和 T'变换结果低位补 0。

（22.5）在 MSD 加法过程中，W 和 W'变换结果高位补 0。

（23）奇偶校验：由于和数项求和采取二叉迭代求和，所以需要对每层迭代求和的操作数个数进行奇偶校验，当迭代求和操作数的个数是奇数时，需要添加一个值为 0 的 MSD 数使操作数个数为偶数，然后进行运算。

（24）与解码器通信次数：通信次数决定运算次数。MSD 乘法并行实现需要与解码器通信次数 count 如公式（3-8）所示。

$$count = 1 + 3 \times \lceil \log_2 m \rceil \tag{3-8}$$

当与解码器进行通信次数达到 count 时，可判断出本次解码结果为最后的运算结果即乘积。

3.3.2 MSD 乘法并行实现上位机实现方法

经过 3.3.1 的分析，得到 MSD 乘法并行实现所需要的信息，但是并非所有 MSD 乘法并行实现需要的信息都需要存放在上位机，只需要将一些必要的信息封装好，然后传输给下位机，下位机根据上位机传输的信息生成完成乘法运算所需的其他信息，然后控制光学处理器完成乘法运算。因此需要甄选出上位机需要生成的运算信息。根据 3.3.1 的分析，给出上位机需要生成的信息。

（1）运算类型：下位机根据运算类型区分不同的数值运算。

（2）乘数的位数：下位机根据乘数的位数生成乘法运算所需的控制信息。

（3）被乘数的位数：由下位机根据被乘数的位数生成乘法运算所需的控制信息。

（4）乘数中小数位数：用于确定乘积的小数位数。

（5）被乘数中小数位数：用于确定乘积的小数位数。

（6）变换器的处理器位起始地址以及所需处理器位数：用于重构乘法运算器，即 M 变换器和 MSD 加法器。

（7）M 变换器的个数：为了取 M 变换器的处理器位的起始地址、所占处理器位的数和 M 变换的结果。

（8）MSD 加法器的个数：为了取 MSD 加法过程中各个逻辑变换器的起始地址、所占处理器位数和相应逻辑变换的结果

（9）乘数和被乘数：用于乘法运算。

假设被乘数 A 有 n 位，小数部分有 k 位，记 $A=a_n\ldots a_i\ldots a_1a_0$，其 MSD 表达式 A=$\sum a_i\times 2^{i\text{-}k}$，$i=0,1,...,n-1$；乘数 B 有 m 位，小数部分有 s 位，记 $B=b_{m-1}\ldots b_j\ldots b_1b_0$，其 MSD 表达式 B=$\sum b_j\times 2^{j\text{-}s}$，$j=0,1,...,m-1$，$a_i$，$b_j$ 符号集均为{u，0，1}，三值光学处理器的处理器位是 N。n，m 和 N 是正整数，$1<m\le n$，i，j，k，s 是自然数，$0\le i\le n-1$，$0\le j\le m-1$，$0\le k\le n$，$0\le s\le m$。根据上位机需要生成信息，给出其实现方法。

（1）运算类型：基于三值光学处理器的运算正在研究发展中，假设用 a 个字节表示。

（2）乘数的位数：由 3.1.3 分析可知光学处理器并行完成 MSD 乘法需要的处理器位数是$\lceil m/2\rceil\times(5\times n+5\times m+10\times\lceil\log_2 m\rceil-11)+n\times m$，其中 n 和 m 分别表示被乘数的位数和乘数的位数，同时 m 还表示 M 变换器的个数，⌈m/2⌉表示 MSD 加法器的个数，$5\times n+5\times m+10\times\lceil\log_2 m\rceil-11$表示单个 MSD 加法器的处理器位数。其必须满足$\lceil m/2\rceil\times(5\times n+5\times m+10\times\lceil\log_2 m\rceil-11)+n\times m\le N$，因此乘数的位数由处理器位决定。假设用 b 个字节表示。

（3）乘数中小数的位数：由乘数的位数决定，因此使用 b 个字节表示。

（4）被乘数的位数：由（2）可知，被乘数乘数的长度由处理器位决定。假设用 c 个字节表示。

（5）被乘数中小数的位数：由被乘数的位数决定，因此用 c 个字节表示。

（6）M 变换器的个数：MSD 加法器的个数是由乘数的位数决定，因此用 b 个字节表示。

（7）M 变换器的处理器位起始地址：是由处理器位数 N 决定。假设用 d 个字节。

（8）M 变换器的所需处理器位数：由被乘数的位数决定，因此用 c 个字节

表示。

表 3.9 三值光学计算机第三方平台的乘法运算信息

控制信息区		
项		值
运算类型		a 个字节
乘数的位数		b 个字节
乘数中小数的位数		b 个字节
被乘数的位数		c 个字节
被乘数中小数的位数		c 个字节
M 变换器的个数		b 个字节
M 变换器	M 变换器 1 的处理器起始地址	d 个字节
	M 变换器 1 的处理器位数	c 个字节
	……	……
MSD 加法器的个数		b 个字节
MSD 加法器 1	T 变换器 1 的处理器位起始地址	d 个字节
	T 变换器 1 的处理器位数	e 个字节
	W 变换器 1 的处理器位起始地址	d 个字节
	W 变换器 1 的处理器位数	e 个字节
	T'变换器 1 的处理器位起始地址	d 个字节
	T'变换器 1 的处理器位数	e 个字节
	W'变换器 1 的处理器位起始地址	d 个字节
	W'变换器 1 的处理器位数	e 个字节
	T2 变换器 1 的处理器位起始地址	d 个字节
	T2 变换器 1 的处理器位数	e 个字节
……		……
数据区		
被乘数		f 个字节
乘数		g 个字节

（9）MSD 加法器的个数：由乘数的位数决定，因此用 b 个字节表示。

（10）MSD 加法各变换器的处理器位起始地址：由处理器位 N 决定。因此用 d 个字节表示。

（11）MSD 加法各变换器的所需处理器位数：是由乘数长度、被乘数长度以及和数项求和方式决定。其中 T2 变换所需处理器位最多。假设用 e 个字节表示，那么 T 变换、W 变换、T'变换和 W'变换需要处理器大小均可用 e 个字节表示。

（12）被乘数：需要的字节数由光学处理器的处理器位数和编码规则决定，假设用 f 个字节表示。

（13）乘数：需要的字节数由光学处理器的处理器位数和编码规则决定，假设用 g 个字节表示。

针对以上信息，将其分为控制信息和数据信息，控制信息从第（1）项到第（11）项，数据信息从第（12）项到第（13）项，具体信息如表 3.9 所示，其中 a~g 表示正整数。

3.3.3 MSD 乘法并行实现下位机实现方法

对于 3.3.2 设计的信息封装以后，通过网络传输的方式，发送给三值光学计算机的下位机，由下位机的解释程序进行解析，并根据解析出来的有效数据，引导光学处理器完成运算。下位机主要涉及两个问题：专用数据存储区和光学处理器数据组织。

（1）专用数据存储区

由于 MSD 乘法运算过程是无进位的逻辑运算，前面逻辑运算的结果需要经解码存储在专用数据存储区，每次存储一整屏数据。然后根据处理器位的分配情况，提取运算结果，对运算结果进行数据位调整，进行后续运算直到运算结束。在三值光学计算机中，解码器使用电子计算机中 2 位来表示 MSD 数字系统中的 1 位，因此缓冲区的大小 Buffersize=K×2×[N/8]，K 是正整数，N 是正整数，表示处理器的处理位数量。

对于数据存储可以采用如下的存储策略。

（1.1）使用覆盖式的存储策略：每次运算结果从该缓冲区取出处理后，清空该缓冲区，为存储下次运算结果做准备，直到运算结束。存储的数据包括主光路数据和控制光路数据，存储结构如图 3-8 所示。主光路数据的存储空间大小是 2×[N/8]字节，N 表示正整数，表示处理器位数，其存储结构如图 3-9 所示，其中 MSD 加法运算结果的存储结构如图 3-10 所示。控制光路数据的存储空间和主光路数据空间相同，因此总共需要 4×[N/8]个字节。但是这种存储方式存在缺点，如果运算结果出现错误，很难发现是哪一次中间结果出现问题。

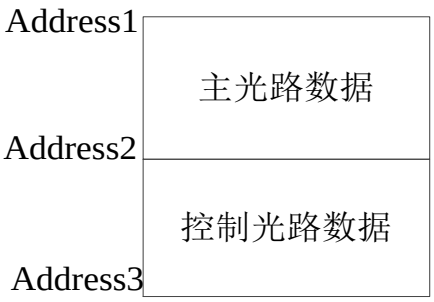

图 3-8 存储结构

M变换结果1	M变换结果2	……	MSD加法结果1	MSD加法结果2	……

图 3-9 主光路数据存储结构

T变换结果	W变换结果	T’变换结果	W’变换结果	T2变换结果

图 3-10 MSD 加法结果的存储结构

（1.2）使用非覆盖式的存储策略：保存每次运算的结果，运算的时候从缓冲区取操作数，对操作数的数据位调整后应用于后续运算，直至运算结束。存储的数据包括主光路数据和控制光路数据，存储结构和第一种方式相同，不同的是存储主光路数据空间变为$(1+3\times\lceil\log_2 m\rceil)\times 2\times\lceil N/8\rceil$字节，总存储空间变为$(1+3\times\lceil\log_2 m\rceil)\times 4\times\lceil N/8\rceil$字节。这种存储方式可以清楚的知道每次运算的中间结果，便于数据校验，但是需要存储空间比第一种存储策略的存储空间要大。

在实际开发过程中，根据下位机运算需求，选择相应的存储策略。

（2）光学处理器的数据组织

将运算所需的变换器重构完成后，将操作数送入光学处理器的相应处理器位区间，完成运算。但是并非所有的变换器都在进行有效的运算，比如 MSD 加法运算。在 T，W 变换器运算时，T'，W'以及 T2 变换器并没有在计算有效数据，但是必须给这些变换器送入数据，称为无效数据。在读取运算结果时，只需要根据变换器的处理器位的分配情况，就能够取得相应的运算结果，因此光学处理器的数据组织显得尤为重要。

MSD 乘法并行计算是一次重构出所有的变换器，其逻辑结构如图 3-11 所示。部分积生成时，向 M 变换器送入有效数据，向所有的 MSD 加法器送入无

效数据，MSD 加法器是由 T 变换器，W 变换器，T'变换器，W'变换器和 T2 变换器组成。得到部分积后，对部分积的数据位进行调整，得到和数项，然后对和数项求和。在迭代求和过程中，数据组织包括以下三个方面：向 M 变换器送入无效数据，向 MSD 加法器的 T，W 变换器送入有效数据，向 T'，W'以及 T2 变换器送入无效数据；向 M 变换器送入无效数据，向 MSD 加法器中的 T'，W'变换器送入有效数据，向 T，W 以及 T2 变换器送入无效数据；向 M 变换器送入无效数据，向 MSD 加法器中的 T2 变换器送入有效数据，向 T，W，T'和 W'变换器送入无效数据。

图 3-11 三值光学计算机光学处理器乘法器的逻辑结构

3.4 本章小结

本章针对 MSD 乘法的实现进行了分析，考虑到三值光学处理器的处理器位众多，处理位易扩展、按位可分配和按位可重构，本文选择了 MSD 乘法的并行实现方案作为第三方平台的研究目标。接着给出连接用户与三值光学计算机的第三方平台设计，并给出了 MSD 乘法第三方平台的实现机制。

第四章　MSD 乘法第三方平台的设计与实现

4.1 三值光学计算机 MSD 乘法器的结构

MSD 乘法器由 M 变换器和 MSD 加法器构成，文献[73]已经给出了 MSD 加法器的结构，因此本文只需要给出 M 变换器的运算结构。下面给出了 M 变换器的构造过程。

（1）首先写出 M 变换真值表，具体参见表 2.1。

（2）根据 M 变换器的真值表写出对应的状态迁移表，如表 4.1 所示。表 4.1 中，a，b 表示输入光的状态，c 表示输出光的状态，W 表示无光态，V 表示垂直偏振光，H 表示水平偏振光。

表 4.1 M 变换真值表对应的状态迁移表

a	b	c
W	W	W
W	V	W
W	H	W
V	W	W
V	V	H
V	H	V
H	W	W
H	V	V
H	H	H

表 4.2 分解定理得到基元表

a	b	c		C_1		C_2		C_3		C_4
W	W	W		W		W		W		W
W	V	W		W		W		W		W
W	H	W		W		W		W		W
V	W	W		W		W		W		W
V	V	H	=	H	+	W	+	W	+	W
V	H	V		W		V		W		W
H	W	W		W		W		W		W
H	V	V		W		W		V		W
H	H	H		W		W		W		H

（3）对表 4.1 应用降值设计理论的分解定理，得到基元表，如表 4.2 所示。

表 4.2 中，a，b 表示输入光的状态，c 表示输出光的状态，C_1，C_2，C_3 和 C_4 表示基元光路输出光的状态，对 C_1，C_2，C_3 和 C_4 输出光的状态进行迭合，其结果就是最终输出光的状态 c。

表 4.3 M 变换器的重构命令

不同区的基元	重构命令
HH 区	00 010 1 00
HV 区	01 100 0 00
VH 区	11 010 1 00
VV 区	10 100 0 00

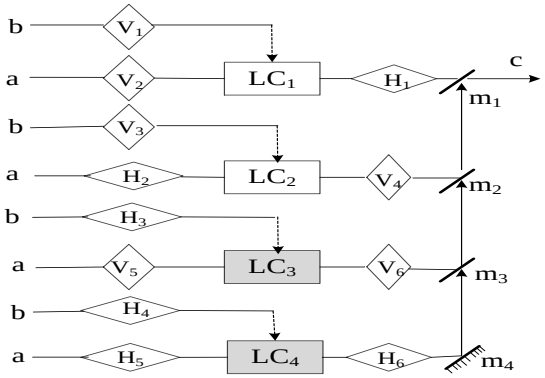

图 4-1 M 变换器光路结构

（4）根据文献[73]给出基元表对应的基元光路，使用半反半透镜和反光镜，实现各个基元输出状态的迭合，底层控制软件根据每种基元对应的重构命令，构造出 M 变换器的硬件光路。M 变换器的重构命令如表 4.3 所示，硬件光路如图 4-1 所示。图 4-1 中，V_1-V_6 表示垂直偏振片，只能透过垂直偏振光；H_1-H_6 表示水平偏振片，只能透过水平偏振光；LC_1-LC_2 表示常不旋光液晶，当液晶有控制信号时，透过光的方向旋转 90 度；LC_3-LC_4 表示常旋光液晶，当液晶无控制信号时，透过光的方向旋转 90 度；a，b 表示输入光的状态，c 表示输出光的状态，m_1-m_3 表示半反半透镜，m_4 表示反光镜。

为了验证 M 变换器光路的正确性，从表 4.2 随机选择一组数据进行分析。a 表示垂直偏振光，作为主光路，b 表示垂直偏振光，作为控制光路，此时输出光的状态 c 应该是水平偏振光。将 a 和 b 作为输入光对 M 变换器的光路进行分析，主光路上，垂直偏振光透过 V_2 和 V_5，被 H_2 和 H_5 吸收，因此垂直偏振光到达 LC_1 和 LC_3，无光到达 LC_2 和 LC_4，V_4 和 H_6 无光输出，控制光路上，垂直

偏振光透过 V_1 和 V_3，被 H_3 和 H_4 吸收，透过 LC_1-LC_4 的光均旋转 90 度，垂直偏振光透过 LC_1 变成水平偏振光，透过 H_1，输出水平偏振光，垂直偏振光透过 LC_3 变成水平偏振光，被 V_6 吸收，无光输出，经过半反半透镜和反光镜迭合，得到最终输出光的状态 c 是水平偏振光，因此光路正确可行。

M 变换器的重构过程表明，重构命令是逻辑变换器重构的依据，文献[74]给出 MSD 加法器中各个逻辑变换器的重构命令，其中 T2 变换器的重构命令和 T 变换器的重构命令相同。

4.2 控制信息文件具体设计

根据 3.3.2 的分析，针对三值光学计算机目前单个模块 576 个处理器位，对 MSD 乘法运算的信息进行了具体设计。

（1）乘法运算符：由于目前三值光学计算机数值计算应用的研究处于发展中，所以运算类型使用一个字节足够表示目前数值计算应用的类型，目前三值光学计算机加法使用（00000011）$_B$ 表示，下标 B 表示二进制，00000011 是 1 个字节，因此乘法运算符也使用 1 个字节表示。

（2）乘数的位数：三值光学计算机总共有 576 个处理器位，其中 64 个处理器位作为冗余区，因此 MSD 乘法并行实现所需要的处理器位数需要小于等于 512，即 $\lceil m/2 \rceil \times (5\times n+5\times m+10\times \lceil \log_2 m \rceil -11)+n\times m \le 512$ ，n 和 m 都是正整数，m 表示乘数的位数，n 表示被乘数的位数，$1<m\le n$。所以三值光学处理器在不采用数据截断技术并且运算器只重构一次的情况下，$1<m\le 8$，因此使用 1 个字节表示乘数的位数。

（3）乘数中小数位数：由乘数的位数决定，所以使用 1 个字节表示乘数中小数位数。

（4）被乘数的位数：三值光学计算机总共有 576 个处理器位，其中 64 个处理器位作为冗余区，因此 MSD 乘法全并行实现所需要的处理器位需要小于等于 512，即 $\lceil m/2 \rceil \times (5\times n+5\times m+10\times \lceil \log_2 m \rceil -11)+n\times m \le 512$，n 和 m 都是正整数，m 表示乘数的位数，n 表示被乘数的位数，$1<m\le n$。所以三值光学处

理器在不采用数据截断技术并且运算器只重构一次的情况下，$1<n\le 71$，使用1个字节表示被乘数的位数。

（5）被乘数中小数位数：由被乘数位数决定，使用1个字节表示被乘数中小数位数。

（6）乘数：目前光学计算机使用并行的方式完成乘法运算，在不采用数据截断技术且运算器只重构一次的情况下，能够完成的乘法运算中，乘数的最大位数是8，由于传输的数据是使用2位电子计算机编码表示1位MSD符号，因此使用2个字节足够表示乘数。

（7）被乘数：目前光学计算机使用并行的方式完成乘法运算，在不采用数据截断技术且运算器只重构一次的情况下，在能够完成的乘法运算中，被乘数的最大位数是71，由于传输的数据是使用2位电子计算机编码表示1位MSD符号，因此使用18个字节足够表示被乘数。

（8）变换器的处理器位起始地址：单个模块光学处理器的处理器位数是576，因此变换器的处理器起始地址使用2个字节表示即可。

（9）M 变换器的处理器位数：三值光学处理器在不采用数据截断技术且运算器只重构一次的情况下，能够完成的乘法运算中，被乘数的最大位数是71。因此只需要使用1个字节就可以表示M变换器的处理器位数。

（10）MSD加法器中各变换器的处理器位数：MSD加法器中T2变换器的偏移量是：$n+m+2\times\lceil\log_2 m\rceil-1$，n和m是正整数，n表示被乘数的位数，m表示乘数的位数，$1<m\le n$。三值光学处理器在不采用数据截断技术且运算器只重构一次的情况下，在能够完成的乘法运算中，T2变换器的处理器位数最大值是74，因此使用1个字节就可以表示MSD加法器中各变换器的处理器位数。

（11）M变换器的个数：由（2）分析可知，目前的光学处理器在不采用数据截断技术且运算器只重构一次的情况下，最多可以重构8个M变换器，因此使用1个字节就可以表示M变换器的个数。

（12）MSD 加法器的个数：由（2）分析可知，目前的光学处理器在不采用数据截断技术且运算器只重构一次的情况下，最多可以重构 4 个 MSD 变换器，因此使用1个字节就可以表示MSD加法器的个数。

表 4.4 乘法运算生成的运算信息文件

<table>
<tr><td colspan="3">控制信息区</td></tr>
<tr><td colspan="2">运算类型</td><td>1 个字节</td></tr>
<tr><td colspan="2">乘数的位数</td><td>1 个字节</td></tr>
<tr><td colspan="2">乘数中小数位数</td><td>1 个字节</td></tr>
<tr><td colspan="2">被乘数的位数</td><td>1 个字节</td></tr>
<tr><td colspan="2">被乘数的中小数位数</td><td>1 个字节</td></tr>
<tr><td colspan="2">M 变换器的个数</td><td>1 个字节</td></tr>
<tr><td rowspan="3">M 变换器</td><td>M 变换器 1 的处理器位起始地址</td><td>2 个字节</td></tr>
<tr><td>M 变换器 1 的处理器位数</td><td>1 个字节</td></tr>
<tr><td>……</td><td>……</td></tr>
<tr><td colspan="2">MSD 加法器的个数</td><td>1 个字节</td></tr>
<tr><td rowspan="10">MSD 加法器 1</td><td>T 变换器的处理器位起始地址</td><td>2 个字节</td></tr>
<tr><td>T 变换器的处理器位数</td><td>1 个字节</td></tr>
<tr><td>W 变换器的处理器位起始地址</td><td>2 个字节</td></tr>
<tr><td>W 变换器的处理器位数</td><td>1 个字节</td></tr>
<tr><td>T’变换器的处理器位起始地址</td><td>2 个字节</td></tr>
<tr><td>T’变换器的处理器位数</td><td>1 个字节</td></tr>
<tr><td>W’变换器的处理器位起始地址</td><td>2 个字节</td></tr>
<tr><td>W’变换器的处理器位数</td><td>1 个字节</td></tr>
<tr><td>T2 变换器的处理器位起始地址</td><td>2 个字节</td></tr>
<tr><td>T2 变换器的处理器位数</td><td>1 个字节</td></tr>
<tr><td>……</td><td>……</td><td>……</td></tr>
<tr><td colspan="3">运算数据区</td></tr>
<tr><td colspan="2">被乘数</td><td>18 个字节</td></tr>
<tr><td colspan="2">乘数</td><td>2 个字节</td></tr>
</table>

根据以上分析，针对目前三值光学计算机处理位，设计的控制信息文件如表 4.4 所示。

4.3 MSD 乘法第三方平台实验用例

4.3.1 控制信息文件

假设输入两个 MSD 数 A=（u1011）$_{MSD}$，B=（10u1）$_{MSD}$ 和乘法运算符，其中操作数 A 的小数位数是 2，操作数 B 的小数位数是 2 位，因为操作数 A 的位数大于操作数 B 的位数，所以选取操作数 A 作为被乘数，操作数 B 作为乘数。据此给出上位机生成控制信息文件的内容。

（1）乘法运算符：乘法使用（00000100）$_B$表示，下标 B 表示二进制。

（2）乘数的位数：乘数的位数是 4。

（3）乘数的小数位数默认是 0，本次设定是 2。

（4）被乘数的长度：被乘数的长度是 5。

（5）被乘数的小数位数默认是 0，本次设定是 2。

（6）乘数：乘数按照编码规则得到（01100001）$_B$，下标 B 表示二进制。

（7）被乘数：被乘数按照编码规则得到（0001100101）$_B$，下标 B 表示二进制。

（8）处理器位的分配情况：由于该三值光学处理器作为 MSD 乘法专用机，所以处理器位默认分配的起始地址是 0，根据 3.3.1 分析可知，处理器位的分配情况如表 4.5 所示。

表 4.5 处理器位的分配情况

运算器		处理器位区间
M 变换器 1		0~4
M 变换器 2		5~9
M 变换器 3		10~14
M 变换器 4		15~19
MSD 加法器 1	T 变换器	20~29
	W 变换器	30~39
	T'变换器	40~50
	W'变换器	51~61
	T2 变换器	62~73
MSD 加法器 2	T 变换器	74~83
	W 变换器	84~93
	T'变换器	94~104
	W'变换器	105~115
	T2 变换器	116~127

（9）M 变换器的个数：M 变换器的个数是 4。

（10）MSD 加法器的个数：MSD 加法器的个数是 2。

根据以上的具体分析，生成控制信息文件的内容，并对其进行格式化，得到最终的控制信息文件，如表 4.6 所示，其中被乘数只给出了前 2 个字节。

表 4.6 乘法并行实现的运算信息

控制信息区		
运算类型		00000100
乘数的位数		00000100
乘数小数位数		00000010
被乘数的位数		00000101
被乘数的小数位数		00000010
M 变换器的个数		00000100
M 变换器	M 变换器 1 的处理器位起始地址	0000000000000000
	M 变换器 1 的处理器位数	00000101
	M 变换器 2 的处理器位起始地址	0000000000000101
	M 变换器 2 的处理器位数	00000101
	M 变换器 3 的处理器位起始地址	0000000000001010
	M 变换器 3 的处理器位数	00000101
	M 变换器 4 的处理器位起始地址	0000000000001111
	M 变换器 4 的处理器位数	00000101
MSD 加法器的个数		00000010
MSD 加法器 1	T 变换器的处理器位起始地址	0000000000010100
	T 变换器的处理器位数	00001010
	W 变换器的处理器位起始地址	0000000000011110
	W 变换器的处理器位数	00001010
	T'变换器的处理器位起始地址	0000000000101000
	T'变换器的处理器位数	00001011
	W'变换器的处理器位起始地址	0000000000110011
	W'变换器的处理器位数	00001011
	T2 变换器的处理器位起始地址	0000000000111110
	T2 变换器的处理器位数	00001100
MSD 加法器 2	T 变换器的处理器位起始地址	0000000001001010
	T 变换器的处理器位数	00001010
	W 变换器的处理器位起始地址	0000000001010100
	W 变换器的处理器位数	00001010
	T'变换器的处理器位起始地址	0000000001011110
	T'变换器的处理器位数	00001011
	W'变换器的处理器位起始地址	0000000001101001
	W'变换器的处理器位数	00001011
	T2 变换器的处理器位起始地址	0000000001110100
	T2 变换器的处理器位数	00001100
数据区		
被乘数		1010100001100101
乘数		1010101001100001

4.3.2 变换器的地址映射

上位机给出了各个变换器的起始地址后，下位机只需将各变换器的重构命令写入 FPGA 板的寄存器，就可以完成运算器的重构。表 4.7 是根据欧阳山设计的全并行液晶控制 FPGA 开发板中重构指令寄存器及其地址的映射关系得出，具体对应关系参见文献[74]。

表 4.7 FPGA 开发板中运算器重构指令寄存器的地址映射

逻辑运算区		运算区		冗余区	
		开始地址	结束地址	开始地址	结束地址
M变换器		10000000000	10000000100	11000000000	11000000011
		10000000101	10000001001	11000000100	11000000111
		10000001010	10000001110	11000001000	11000001011
		10000001111	10000010011	11000001100	11000001111
MSD加法器1	T	10000010100	10000011101	11000010000	11000010011
	W	10000011110	10000100111	11000010100	11000010111
	T'	10000101000	10000110010	11000011000	11000011011
	W'	10000110011	10000111101	11000011100	11000011111
	T2	10000111110	10001001001	11100000000	11100000011
MSD加法器2	T	10001001010	10001010011	11100000100	11100000111
	W	10001010100	10001011101	11100001000	11100001011
	T'	10001011110	10001101000	11100001100	11100001111
	W'	10001101001	10001110011	11100010000	11100010011
	T2	10001110100	10001111111	11100010100	11100010111

4.3.3 专用数据存储区

对于下位机的控制程序，开发板采用的是 Friendly ARM Mini2440，该开发板使用了两片 32MB 的 SDRAM 芯片，总共 64MB。其中从 0x30000000 到 0x31000000 的 16MB 作为程序下载区的存储区间，从 0x31000000 作为专用存储区的起始地址。

在下位机控制程序的设计过程中，数据存储方式为覆盖式的存储，即下一次存入的数据覆盖之前的数据，因此该回馈存储区的大小设定应满足每次写入数据的大小即可，写入的数据为每次与解码器通信获得的解码结果。

按照当前的实验系统提供的光学处理器位数为 576 位，且每次解码器对运算结果进行解码，使用电子计算机中 2 位编码表示 MSD 数字系统中的 1 位，

因此主光路数据存储空间应满足[576/8]×2=144 字节，主光路数据存储空间扩展时，应当以 144 字节为单位进行扩展，控制光路数据存储空间和主光路数据存储空间相同，专用数据区的存储结构如图 4-2 所示。

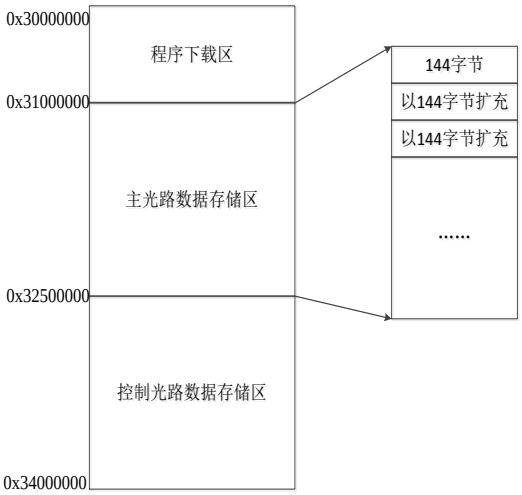

图 4-2 专用数据区的存储结构

4.4 本章小结

本章根据已有的理论，给出下位机 M 变换器的结构，并针对目前三值光学计算机处理器位的情况分析了，对完成 MSD 乘法的控制信息文件进行具体设计。设计 MSD 乘法第三方平台的实验用例，给出上位机生成完成 MSD 乘法运算的具体信息，同时给出下位机重构命令和 FPGA 的对应关系表以及专用数据存储区的设计结构。这些为 MSD 乘法第三方平台的实验奠定了基础。

第五章　实验验证

基于前文对三值光学计算机 MSD 乘法、MSD 乘法第三方平台以及 MSD 乘法第三方平台实现机制的分析与探讨，本章将对它们的正确性和可靠性进行实验验证，并对实验结果进行分析。

5.1 实验环境

物理实验环境使用的是三值光学计算机（SD11）实验系统。该实验系统由上位机和下位机两部分组成。SD11 的上位机的硬件结构为：CPU：4 核，Intel（R）Core（TM）i7-4790 CPU @ 3.60GHz，DDR3 4GB RAM，32 位 Windows7。上位机主要用于接收用户的输入、对用户输入数据进行调整、编码和处理器位的分配等，然后生成完成 MSD 乘法运算的控制信息文件。下位机是光学处理器部分，主要包含编码器、光学处理器、解码器等部件。编码器由两块液晶阵列加一块垂直偏振片组成，解码器由分光镜、一块垂直偏振片、一块水平偏振片和一个水平方向的光电转换器和垂直方向的光电转换器构成。光学处理器由两块偏振片和一块液晶阵列构成，其中光学处理器采用的是 576 位的液晶阵列。处理器分区结构如图 5-1 所示，根据光学处理器中偏振片的方向不同，主光路分为 HH、HV、VV 和 VH 区，控制光路分为 H 和 V 两个分区。主光路的每个分区均由编码器、光学处理器、解码器构成。控制光路的每个分区由编码器和解码器构成。物理实验系统如图 5-2 所示，由于拍照限制此处显示拍摄出的两个区的实物图。

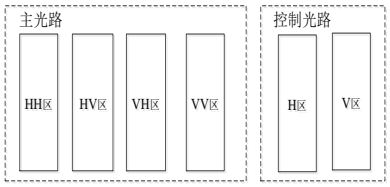

图 5-1 处理器分区硬件结构

图 5-2 物理实验系统

5.2 实验结果

该部分给出仿真实验结果和物理实验结果。

5.2.1 仿真实验

为了验证 MSD 乘法实验结果的正确性，进行大量的仿真实验，从中随机选取了 5 组实验，实验结果如表 5.1 所示。

表 5.1 仿真实验结果

组别	操作数		MSD 结果	十进制结果	对比结果
第 1 组	被乘数	101	001u00u10	30	正确
	乘数	110			
第 2 组	被乘数	u0uu	001u000010	66	正确
	乘数	uu0			
第 3 组	被乘数	u1011	000000u00u01	-35	正确
	乘数	10u1			
第 4 组	被乘数	10uu	00000uu010u	-45	正确
	乘数	u0u1			
第 5 组	被乘数	u1011	0000000u10u11u0	-70	正确
	乘数	10u10			

5.2.2 物理实验

物理实验一方面证明光学计算机可以完成 MSD 乘法运算，另一方面说明 MSD 乘法第三方平台实现机制正确可行。

图 5-3 运算请求

用户输入运算请求，点击确定按钮后，将运算请求发送给三值光学计算机

算，然后等待运算结果。图 5-3 是用户输入的运算请求，图 5-4 是用户获取的运算结果。

图 5-4 运算结果

针对用户输入运算请求后，光学计算机内部具体运算过程如下。

首先，将乘数的每一位复制成和被乘数长度相同的辅助数据，结果是：（11111）$_{MSD}$，（uuuuu）$_{MSD}$，（00000）$_{MSD}$ 和（11111）$_{MSD}$。对被乘数编码，结果是：（0001100101）$_{B}$，下标 B 表示二进制。对辅助数据的得到：（0101010101）$_{B}$，（000000000）$_{B}$，（1010101010）$_{B}$ 和（0101010101）$_{B}$。将被乘数的编码结果作为控制光路数据，辅助数据的编码结果作为主光路数据，将其分别写入主光路编码寄存器和控制光路编码寄存器，然后送入运算指令，完成 M 变换，其结果是：（u1011）$_{MSD}$，（1u0uu）$_{MSD}$，（00000）$_{MSD}$ 和（u1011）$_{MSD}$，如图 5-5 所示。

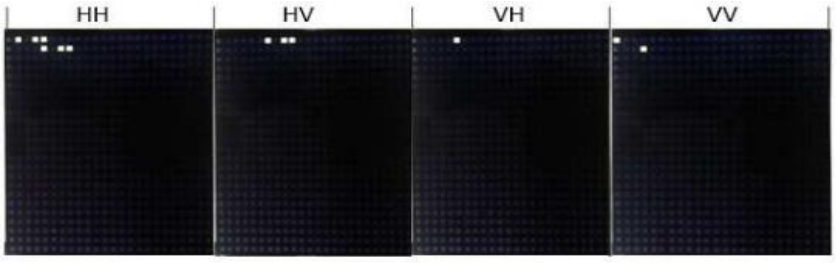

图 5-5 M 变换结果

然后对部分积求和，如果操作数的个数是奇数，添加一个值为 0 的 MSD 数，使操作数的个数为偶数，然后继续运算，否则直接运算。由 M 运算的结果可知，部分积的个数是偶数，因此直接开始运算。首先将 M 变换的结果进行移位，得到和数项：（u1011）$_{MSD}$，（1u0uu0）$_{MSD}$，（0000000）$_{MSD}$ 和（u1011000）$_{MSD}$，和数项高位补 0，使其长度是 5+4+2×2-3=10，得到二叉迭代求和的操作数：

（00000u1011）$_{MSD}$，（00001u0uu0）$_{MSD}$，（0000000000）$_{MSD}$和（00u1011000）$_{MSD}$，对操作数编码得到：（1010101010000110 0101）$_{B}$，（1010101001001000 0010）$_{B}$，（1010101010101010 1010）$_{B}$和（1010000110010110 1010）$_{B}$，这些操作数作为 MSD 加法器中 T 变换器和 W 变换器的输入。将（1010101010000110 0101）$_{B}$和（1010101001001000 0010）$_{B}$作为第 1 个 MSD 加法器的输入，将（1010101010101010 1010）$_{B}$和（1010000110010110 1010）$_{B}$作为第 2 个 MSD 加法器的输入。操作数序号从 0 开始，指定偶数项的作为 T 和 W 变换的主光路数据，即（1010101010000110 0101）$_{B}$和（1010101010101010 1010）$_{B}$作为主光路数据，奇数项作为 T 和 W 变换的控制光路数据，即（1010101001001000 0010）$_{B}$和（1010000110010110 1010）$_{B}$作为控制光路数据，分别将操作数送入相应的寄存器，发出运算指令，完成运算，运算结果如图 5-6 所示。第 1 对操作数的 T 变换和 W 变换的结果是：（00001u1u01）$_{MSD}$和（0000u0u10u）$_{MSD}$，第 2 对操作数的 T 变换和 W 变换的结果是：（00u1011000）$_{MSD}$和（0001u0uu00）$_{MSD}$。

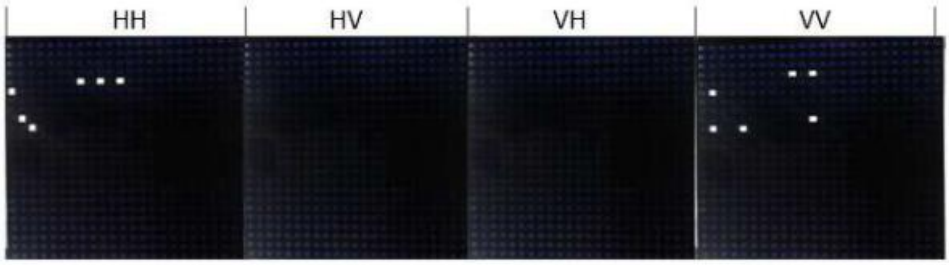

图 5-6 第一层 MSD 加法 T，W 变换的结果

对 T 和 W 变换的运算结果进行解码，在 T 变换结果的解码数据低位补 0，并将其作为 T'和 W'变换的主光路数据，在 W 变换结果的解码数据高位补 0，将其作为 T'和 W'变换的控制光路数据，然后将主光路数据和控制光路数据写入相应的寄存器，发出运算指令，完成运算。第 1 对操作数的 T'和 W'的运算结果是：（00000u0u000）$_{MSD}$和（0000101011u）$_{MSD}$，第 2 对操作数的 T'和 W'的运算结果是：（00010000000）$_{MSD}$和（00u0u10u000）$_{MSD}$，如图 5-7 所示。

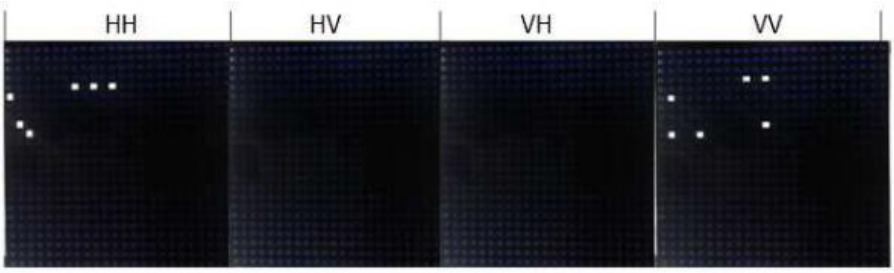

图 5-7 第一层 MSD 加法 T'，W'变换的结果

对 T'和 W'变换运算结果进行解码，在 T'变换结果的解码数据的低位补 0，并将其作为 T2 变换的主光路数据，在 W'变换结果的解码数据高位补 0，将其作为 T2 变换的控制光路数据，然后将主光路数据和控制光路数据写入相应的寄存器，发出运算指令，完成运算。第一对操作数的和是（00000000011u）$_{MSD}$，第二对操作数的和是（00000u10u000）$_{MSD}$，如图 5-8 所示。

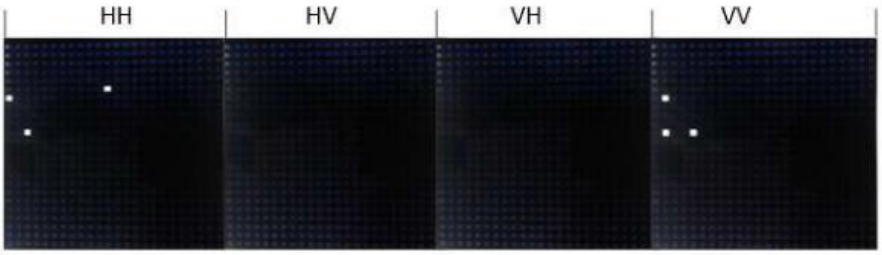

图 5-8 第一层 MSD 加法 T2 变换的结果

二叉迭代求和需要$\lceil \log_2 4 \rceil = 2$层 MSD 加法才能得到乘积。其中最后一层 MSD 加法的 T 和 W 变换的结果是：（000u10u11u）$_{MSD}$ 和（0001u01uu1）$_{MSD}$，如图 5-9（a）所示。在 T 变换结果的解码数据的低位补 0，并将其作为 T'和 W'变换的主光路数据，在 W 变换结果的解码数据高位补 0，将其作为 T'和 W'变换的控制光路数据，然后将主光路数据和控制光路数据分别写入相应的寄存器，发送运算指令，完成运算。T'和 W'变换的结果是：（000010010u0）$_{MSD}$ 和（000u0uu0001）$_{MSD}$，如图 5-9（b）所示。在 T'变换结果的解码数据的低位补 0，并将其作为 T2 变换的主光路数据，在 W'变换结果的解码数据高位补 0，将其作为 T2 变换的控制光路数据，然后将主光路数据和控制光路数据分别写入相应的寄存器，发出运算指令，完成运算。T2 运算的结果即乘积是：（000000u00u01）$_{MSD}$，如图 5-9（c）所示。

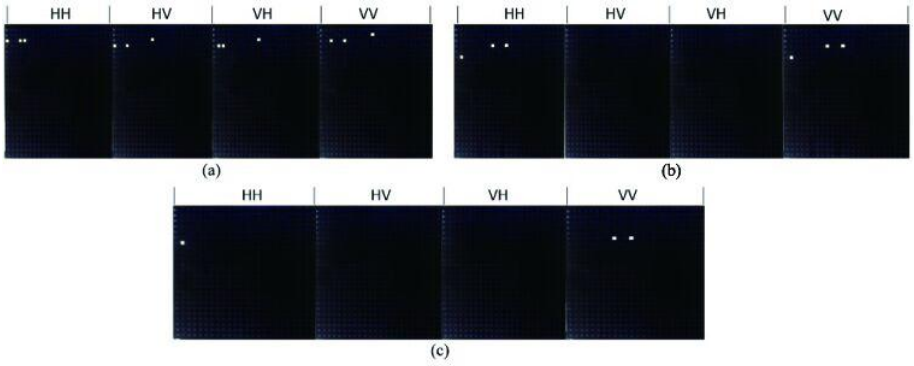

图 5-9 （a）表示最后一层 MSD 加法 T，W 变换的结果，（b）表示最后一层 MSD 加法 T'，W'变换结果，（c）表示最后一层 T2 变换的结果即最后的乘积。

5.3 实验结果分析

三值光学计算机处理器的运算区分为 HH，HV，VH，VV 四个区，因此每次运算之后，四个运算区的结果迭合后，得到最终的运算结果。不同运算区的液晶相同位置的像素有且至多有一个是亮的，如果亮的像素出现在 HH 或者 VH 运算区，它代表的值就是 1，如果亮的像素出现在 HV 或者 VV 运算区，它代表的值是 u，如果四个运算区的液晶同一位置都不亮，它代表的值是 0。因此根据图 5-5，可以对 M 变换的结果进行分析，如表 5.2 所示。

表 5.2 M 运算结果

操作数	第一组	第二组	第三组	第四组
操作	M 变换	M 变换	M 变换	M 变换
HH 运算区	01011	00000	00000	01011
HV 运算区	00000	01011	00000	00000
VH 运算区	00000	10000	00000	00000
VV 运算区	10000	00000	00000	10000
M 运算结果	u1011	1u0uu	00000	u1011

表 5.3 乘法运算的乘积

项	值
HH	000000000001
HV	000000000000
VH	000000000000
VV	000000100100
乘积	000000u00u01

对图 5-9 中（c）T2 变换的结果即乘积分析，如表 5.3 所示，得到乘积是

（000000u00u01）$_{MSD}$，由于乘数和被乘数的小数位数都是 2 位，所以乘积会有 4 位小数，因此将乘积转换成十进制是-2.1875。将被乘数和乘数分别转换成十进制是-1.25 和 1.75 得到的乘积时-2.1875，所以运算结果正确。该实验不仅证明三值光学计算机可以完成 MSD 乘法运算，而且证明本文设计 MSD 乘法第三方平台实现机制正确可行。

5.4 与电子计算机的对比

（1）同时文献[38]给出三值光学计算机并行乘法器计算延时与传统电子计算机计算延时的对比，如表 5.4。表 5.4 中的数据表明，三值光学计算机乘法运算的并行实现速度比传统乘法器的速度快，尤其是在乘法操作数的数据位数较多情况下，优势更加明显，主要原因是三值光学处理器的处理器位众多、处理器位易扩展、按位可分配和按位可重构，使乘法运算可以并行实现，同时部分积求和过程不存在进位延时。

表 5.4 不同乘法器的计算延时的对比

乘法器类型	延时（单位：纳秒）		
	16×16	32×32	64×64
Vedic 乘法器	24.541	31.835	33.882
Dadda 乘法器	28.037	35.372	37.57
Braun 乘法器	28.155	35.444	37.642
Wallace 乘法器	33.354	40.643	42.842
TOC 并行乘法器	13	16	19

（2）乘法运算能耗主体现在部分积产生、部分积压缩以及部分积求和过程等。在电子计算机乘法运算过程中，部分积求和过程随着数据位数增加，能耗呈现线性增长甚至是指数增长[75]。三值光学计算机下位机采用液晶和嵌入式系统，而液晶的能耗比较低，嵌入式系统只在数据传输、数据存储和发送液晶的控制信号时才工作，因此三值光学计算机能耗主要集中在数据传送、数据存储和发送液晶控制信号。根据文献[18，76]指出，带有 16 个模块的三值光学计算机的耗能不会超过 800 瓦，虽然还没有具体量化三值光学计算机的能耗，但是它的优势明显高于电子计算机。

5.5 本章小结

本章首先介绍了实验环境，给出 MSD 乘法第三方平台实验用例的实验结果，并给出了 MSD 乘法的运算过程。最后对三值光学计算机 MSD 乘法并行运算的延时与不同乘法器的运算延时进行了对比。

第六章　结论与展望

随着三值光学计算机理论研究不断完善，原型系统不断成熟，针对一些应用进行了大量研究，但是实验过程需要人工参与。用户应用三值光学计算机必须要了解光学计算机的工作原理和实现细节，这给用户应用光学计算机增加了难度，限制了三值光学计算机的发展。针对该现状，本文设计了针对 MSD 乘法的第三方平台。该平台一方面使运算能够连续完成，另一方面让用户像使用电子计算机一样应用三值光学计算机，为三值光学计算机推广奠定基础。

本文设计并实现了三值光学计算 MSD 乘法第三方平台，考虑到三值光学计算机处理器的处理器位众多、易扩展、按位可分配和按位可重构，所以本文选择了 MSD 乘法的并行实现方案，给出了 MSD 乘法第三方平台的实现机制。本文针对 MSD 乘法第三方平台进行了大量实验，实验结果表明三值光学计算机可以完成 MSD 乘法运算，无需用户了解光学计算机的工作原理和实现细节，MSD 乘法第三方台的实现机制是正确可行。

6.1 结论

本文以用户为中心，以三值光学计算机为研究背景，设计并实现了 MSD 乘法第三方平台，并给出 MSD 乘法第三方平台实现机制，主要成果如下。

（1）根据 MSD 乘法运算原理，分析了 MSD 乘法两种实现方式：MSD 乘法流水线实现方案和 MSD 乘法并行实现方案，并给出两种实现方案所需时钟周期和处理器位的分配情况。

（2）三值光学计算机处理器的处理器位众多、易扩展、按位可分配和按位可重构，因此本文选择 MSD 乘法的并行实现方案，并结合三值光学计算机系统的特点，设计与实现 MSD 乘法。MSD 乘法的实现为后续大数据位乘法并行实现的研究和可以并行实现的应用比如矩阵向量乘法、快速傅里叶变换等提供依据。

（3）为了方便用户应用三值光学计算机，使三值光学计算机的乘法运算能够实现，本文设计并实现了三值光学计算机 MSD 乘法第三方平台，并给出 MSD

乘法第三方平台的实现机制。

（4）针对 MSD 乘法第三方平台进行了大量的仿真实验和物理实验，实验结果表明，三值光学计算机可以完成 MSD 乘法运算，无需用户了解光学计算机的工作原理和实现细节，MSD 乘法第三方台的实现机制正确可行。

6.2 展望

三值光学计算机 MSD 乘法第三方平台能够让更多的用户更方便的应用三值光学计算机，同时可以让三值光学计算机的运算连续的完成，提高运算效率。作者认为今后的研究工作主要包括：

（1）本文探讨 MSD 并行实现是具有小数据位的特点，如果三值光学处理器处理大数据位的乘法运算需要进一步讨论。

（2）MSD 乘法在三值光学计算机上的实现，为并行实现的应用比如矩阵向量乘法、快速傅里叶变换等在三值光学计算机上的研究与实现提供依据，这些应用的研究与实现亟待展开。

（3）本文只是以 MSD 乘法为突破口，设计并实现了三值光学计算机 MSD 乘法第三平台，而三值光学计算机其他应用第三方平台的研究亟待展开，比如矩阵向量乘法，快速傅里叶变换等。

（4）在乘法运算过程中，可以借鉴 Booth 编码的思想，寻找一种适合 MSD 数的编码方法，减少产生部分积的个数，提高 MSD 乘法运算效率。

参考文献

【1】. 吴楠，宋方敏. 量子计算与量子计算机[J]. 计算机科学与探索, 2007.1(1): 1-16.

【2】. Ladd T D, Jelezko F, Laflamme R, et al. Quantum computers[J]. Nature, 2010, 464(7285): 45-53.

【3】. Lewin D I. DNA Computing[J]. Computing in Science & Engineering, 2002. 4(3): 5-8.

【4】. Ichioka Y. Optical Computer[J]. Oyobuturi, 2010, 54.

【5】. Caulfield H J, Vikram C S, Zavalin A. Optical logic redux[J]. Optik - International Journal for Light and Electron Optics, 2006, 117(5):199-209.

【6】. Jin Y, He H C, Lu Y T. Ternary optical computer principle[J], Science China Information Sciences , 2003, 46(2):145–150.

【7】. Jin Y, He H C, Lu Y T. Ternary Optical Computer Architecture[J]. Physica Scripta, 2006, 118(T118):98.

【8】. 金翊. 三值光计算机原理和结构[D]. 西北工业大学博士学位论文，西安, 2003.

【9】. Abdeldayem H, Frazier D O. Optical computing: need and challenge[M]. ACM, 2007.

【10】. Ambs P. A short history of optical computing: rise, decline, and evolution[J]. Proceedings of SPIE - The International Society for Optical Engineering, 2009, 7388:73880H-73880H-14.

【11】. Fushimi A, Tanabe T. All-optical logic gate operating with single wavelength.[J]. Optics Express, 2014, 22(4):4466-4479.

【12】. Chanalia P, Gupta A. Realization of High Speed All-Optical Logic Gates based on the Nonlinear Characteristics of a SOA [J]. Indian Journal of Science & Technology, 2016, 9(36).

【13】. Majumdar A, Dodson C M, Fryett T K. Cavity enhanced nonlinear optics for few photon optical bistability [J]. Optics Express, 2015, 23(12):16246-16255.

【14】. Gonzalez J, Orosa L, Azevedo R. Architecting a computer with a full optical RAM[C]. Electronics, Circuits and Systems (ICECS), 2016 IEEE International Conference on. IEEE, 2016: 716-719.

【15】. HPE's New Chip Marks a Milestone in Optical Computing. http://spectrum.ieee. org

【16】. Beeler C, Partridge C. All-Optical Computing and All-Optical Networks are Dead[J].

Queue, 2009, 7(3):10-11.

【17】. Yan J Y, Zuo K Z. Decrease-radix design principle for carrying/ borrowing free multi-valued and application in ternary optical computer[J], Science in China, 2008, 51 (10):1415-1426.

【18】. Jin Y, Wang H J. Principles, structures, and implementation of reconfigurable ternary optical processors[J]. Science China, 2011, 54(11):2236-2246.

【19】. 王宏健, 金翊, 欧阳山. 一位可重构三值光学处理器的设计和实现[J]. 计算机学报, 2014, 37(7):1500-1507.

【20】. Song K, Liu Y P. Reconfigurable ternary optical processor based on row operation unit[J], Optics Communications , 2015, 350: 6-12.

【21】. Shen Z Y, Wu L L, Yan J Y, The reconfigurable module of ternary optical computer[J], Optik - International Journal for Light and Electron Optics 2013, 124(13): 1415-1419.

【22】. 金翊, 欧阳山, 宋凯, 等. 三值光学处理器的数据位管理理论和技术[J]. 中国科学: 信息科学, 2013, 43(3): 361-373.

【23】. 展豪君, 金翊, 欧阳山,等. 内存空间在双空间存储器上的推移技术实验[J]. 上海大学学报（自然科学版）, 2017, 23(2):201-215.

【24】. Li G Q, Liu L R, Qian J J, Yin Y Z. Optical parallel negabinary arithmetic based on logic operation and signed-digit representation[J]. Chinese Journal of Lasers, 1997, 7:659-664.

【25】. Cherri A K. Efficient optical negabinary modified signed-digit arithmetic: one-step addition and subtraction algorithms[J]. Optical Engineering, 2003, 43(2):420-425.

【26】. Al-Zayed A S, Cherri A K. Improved all-optical modified signed-digit adders using semiconductor optical amplifier and Mach–Zehnder interferometer[J]. Optics & Laser Technology, 2010, 42(5):810-818.

【27】. Bocker R P, Drake B L, Lasher M E, et al. Modified signed-digit addition and subtraction using optical symbolic substitution[J]. Applied Optics, 1986, 25(15):2456.

【28】. Jin Y, He H C, Ai L R. Lane of parallel through carry in ternary optical adder[J]. Science in China, 2005, 48(1):107-116.

【29】. Jin Y, Shen Y F, Peng J J, et al. Principles and construction of MSD adder in ternary optical computer[J]. Science China, 2010, 53(11):2159-2168.

【30】. Shen Y F, Pan L. Principle of a one-step MSD adder for a ternary optical computer[J]. Science China Information Sciences, 2014, 57(1):1-10.

【31】. Shen Y, Pan L, Jin Y, et al. One-step binary MSD adder for ternary optical computer[J]. Scientia Sinica, 2012, 42(7):869-878.

【32】. Kong S, Peng J J, Fu Y Y, and Wei X Y. Carry-free full-symbol one-step modified signed-digit addition[J]. Appl. Opt., 2017,56: 9620-9628 .

【33】. Song K, Liu Y P. Design and implementation of the one-step MSD adder of optical computer[J]. Applied Optics. 2012, 51(7):917-926.

【34】. Peng J J, Shen R, Jin Y, et al. Design and Implementation of Modified Signed-Digit Adder[J]. IEEE Transactions on Computers, 2014, 63(5):1134-1143.

【35】. He H, Peng J J, Liu Y P, Wang X C, Song K. Research and Design of a MSD Adder of Ternary Optical Computer[J]. 2010 The 3rd International Conference on Computational Intelligence and Industrial Application, Wuhan, China. V:265-268.

【36】. Liu Y P, Peng J J, Chen Y Y, He H, Su H T. A New Carry-Free Adder Model for Ternary Optical Computer[C], Distributed Computing and Applications to Business, Engineering and Science (DCABES), 2011 Tenth International Symposium on, 2011, 64-68.

【37】. Hu X J, Jin Y, Ouyang S. A 40-Bit Multiplication Routine of Ternary Optical Computer[J]. Journal of Shanghai University, 2014, 20(5): 645-657.

【38】. Xu Q, Wang X C, Xu C. Design and implementation of the modified signed digit multiplication routine on a ternary optical computer[J]. Appl Opt, 2017, 56(16):4661-4669.

【39】. Xu Q, Jin Y, Shen Y F, et al. MSD iterative division algorithm and implementation technique for a ternary optical computer[J]. Scientia Sinica, 2016, 46(4): 539-550.

【40】. Wang X C, Peng J J, Li M, et al. Carry-free vector-matrix multiplication on a dynamically reconfigurable optical platform.[J]. Applied Optics, 2010,

49(12):2352-2362.

【41】. 彭俊杰，魏鑫燏，张晓峰，沈云付,付友谊.基于三值光学计算机的并行快速傅里叶算法实现[J]. 中国科学信息科学, 2017, 07:846-862.

【42】. Caro D D, Petra N, Strollo A G M, et al. Fixed-Width Multipliers and Multipliers-Accumulators With Min-Max Approximation Error[J]. IEEE Transactions on Circuits & Systems I Regular Papers, 2013, 60(9):2375-2388.

【43】. Chuang C H, Lin C L. A Novel Synthesizing Genetic Logic Circuit: Frequency Multiplier[J]. IEEE/ACM Transactions on Computational Biology & Bioinformatics, 2014, 11(4):702-713.

【44】. 郑伟, 姚庆栋, 张明,等. 一种高性能、低功耗乘法器的设计[J]. 浙江大学学报（工学版）, 2004, 38(5):534-538.

【45】. Booth A D. A signed binary multiplication technique[J]. The Quarterly Journal of Mechanics and Applied Mathematics, 1951, 4(2): 236-240.

【46】. Bewick G W. Fast multiplication: algorithms and implementation[D]. Stanford University, 1994.

【47】. Dawoud D S. Modified Booth algorithm for higher radix fixed-point multiplication[C]//Communications and Signal Processing, 1997. COMSIG '97, Proceedings of the 1997 South African Symposium on. IEEE, 1997: 95-100.

【48】. Wallace C S. A Suggestion for a Fast Multiplier[J]. IEEE Transactions on Electronic Computers, 1964, ec-13(6):14 - 17.

【49】. Al-Twaijry H, Flynn M. Multipliers and datapaths[R]. Technical Report CSL-TR-94-654, Stanford University, 1994.

【50】. Mou Z J, Jutand F. "Overturned-Stairs" Adder Trees and Multiplier Design[J]. IEEE Computer Society, 1992, 41(8):940-948.

【51】. 孙振玮. 基于优化 Booth 算法实现的可配置 18 位乘法器硬核设计与验证[D]. 电子科技大学硕士论文, 西安, 2011.

【52】. 金翊. 走近光学计算机[J]. 上海大学学报（自然科学版）, 2011, 17(4):401-411.

【53】. Lentine A L, Hinton S, Miller D A B, et al. Symmetric self-electro-optic effect device:

Optical set-reset latch[M]// Applied Physics Letters. 1998.

【54】. 王颖, 张新亮, 黄德修. 基于级联半导体光放大器中交叉增益调制效应的新型全光逻辑与门[J]. 中国激光, 2004, 31（12）:1433-1436.

【55】. Mal P, Cantin J F, Jr B F. Development of a multitechnology field-programmable gate array suitable for photonic information processing.[J]. Applied Optics, 2005, 44(22):4753-4760.

【56】. 申铉京, 千庆姬. 基于符号替换算法的光电混合型计算机的设计[J]. 光学技术, 2000, 26(1):62-65.

【57】. Dickinson A G, Huang A. Programmable optical computer architecture[C]// Int'l Optics in Complex Sys. Garmisch, Frg. International Society for Optics and Photonics, 1990, 1319: 177-178.

【58】. Cao L, Ma Q, Jin G. Parallel data processing based on high density volume holographic correlator[J]. Proceedings of SPIE - The International Society for Optical Engineering, 2009, 7420:74200O-74200O-8.

【59】. Carter J A, Pape D R. High-performance optical vector-matrix coprocessor[J]. Proceedings of SPIE - The International Society for Optical Engineering, 1994, 2297:225-236.

【60】. Davis J A, Yzuel M J, Campos J, et al. Operation of liquid-crystal displays for optical computing[J]. Proceedings of SPIE - The International Society for Optical Engineering, 2005, 5907:590701-590701-14.

【61】. Ehsanpour M, Moallem P, Vafaei A. Design of a novel reversible multiplier circuit using modified full adder[C]// International Conference on Computer Design and Applications. IEEE, 2010:V3-230-V3-234.

【62】. 张锐, 李修建, 杨建坤,等. 一种混合负二进制编码的光学矩阵乘法系统[J]. 光学与光电技术, 2007, 5(3):66-68.

【63】. Yang L, Zhang L, Ji R Q. On-chip optical matrix-vector multiplier for parallel computation[J]. Proceedings of SPIE - The International Society for Optical Engineering, 2013, 8855(43).

【64】. 卢洋洋, 周平, 朱巍巍,等. 基于光学向量矩阵乘法器的光学信息处理系统研究[J]. 光电子 激光, 2013(9):1656-1661.

【65】. 聂永名. 相干光双 DMD 矢量矩阵乘法器[D]. 国防科学技术大学硕士学位论文, 湖南, 2008.

【66】. 胡晓俊. 三值光学计算机整数乘法例程[D]. 上海大学硕士学位论文, 上海, 2013.

【67】. Shen Y F, Jiang B P, Jin Y, et al. Principle and design of ternary optical accumulator implementing M- k -B addition[J]. Optical Engineering, 2014, 53(9):744-752.

【68】. 沈戎. 三值光学计算机乘法器关键技术研究[D]. 上海大学硕士学位论文, 上海, 2014.

【69】. Ping X S, Peng J J, Ouyang S, et al. A Platform for Routine Development of Ternary Optical Computers[C]// International Conference on High Performance Computing and Applications. Springer International Publishing, 2015:143-149.

【70】. Avizienis A. Signed-Digit Numbe Representations for Fast Parallel Arithmetic[J]. Electronic Computers Ire Transactions on, 1961, EC-10(3):389-400.

【71】. Bocker R P, Drake B L, Lasher M E, et al. Modified signed-digit addition and subtraction using optical symbolic substitution[J]. Applied Optics, 1986, 25(15):2456-2457.

【72】. 孙浩, 金翊, 严军勇. 三值光计算机编码器与解码器原理的实验研究[J]. 计算机工程与应用, 2004, 40(16):82-83.

【73】. 贺辉. 三值光学计算机 MSD 加法器关键技术研究[D]. 上海大学硕士学位论文, 上海, 2010.

【74】. 刘艳萍. 三值光学计算机 MSD 加法器研究与实验[D]. 上海大学硕士学位论文, 上海, 2011.

【75】. 林钰凯. 高性能并行乘法器关键技术研究[D]. 西安电子科技大学硕士学位论文, 西安, 2010.

【76】. 欧阳山. 三值光学处理器控制电路设计和实现[D]. 上海大学博士学位论文, 上海, 2012.

作者在攻读硕士学位期间公开发表的论文

【1】 Kong Shuai, Peng Junjie, Fu Youyi, and Wei Xinyu. Carry-free full-symbol one-step modified signed-digit addition. Applied Optics, Volume 56, 2017, Pages 9620-9628. SCI, JCR 3 区.

【2】 Peng Junjie, Fu Youyi, Zhang Xiaofeng, Kong Shuai, Wei Xinyu.Implementation of DFT application on ternary optical computer. Optics Communications, Volume 410, 1 March 2018, Pages 424-430. SCI, JCR 3 区.

作者在攻读硕士学位期间所作的项目

【1】. 国家自然科学基金项目（No.61572305）“三值光学计算机乘法器及计算机例程平台关键技术研究”。

【2】. 国家自然科学基金项目（No.61103054）“基于冗余数的光学并行加法器研究”。

【3】. 中国航天科工集团二院的创新基金。

致　　谢

回顾了研究生两年的生活，从学习生活的不适应到应对自如，取得优异的成绩，从科研小白到独立进行科学研究，撰写学术论文，从面对学生工作无从下手到有条不紊的完成工作，得到同学们的认可，从有生活陋习到养成良好的生活习惯，从缺少生活常识到非常了解生活细节，一路不断收获，不断成长，在此需要感谢太多的人。

研究生的学习生活，从大学的半年一学期到现在十周一学期，学习节奏明显加快，我没有很好的适应学校的学习节奏，当时萌生退学的想法，但是通过和身边同学交流，他们也面临同样的问题，所以大家相互鼓励，相信自己一定可以完成学习任务，渐渐地开始习惯了学校的学习生活节奏，并取得优异的学习成绩，因此感谢相遇的同学。

在科研生活中，非常感谢我的导师彭俊杰教授。彭老师知识渊博，对于课题把握具有高度的前瞻性，从科研定题到科研指导再到论文撰写，彭老师对我的指导尽心尽力。彭老师在注重学生科研能力培养的同时，还注重学生科研素养的培养，比如作报告时的语态、眼神、用语等。同时还要感谢金翊教授，平时科研过程中遇到疑惑，金老师都能耐心解答，为自己的科研提供巨大帮助，此外还要感谢老师沈云付、欧阳山和周时强，博士生江家宝、王哲河、张红红和张素兰以及硕士生付友谊等在科研过程中对于自己的指导和帮助，让自己科研之路更加平坦。

在工作中，由于学习压力大和学生工作多，所以对于学生工作感觉力不从心，感谢学院老师对我的信任，给了我很大的动力，同时也感谢庞毅杰、谢俊、刘懿霆及毛顺亿等同学的帮助，使我处理学生工作更加得心应手。

在生活中，主要感谢我的舍友赵涛涛、梁文飞和刘尧，他们帮我改掉生活中的一些陋习，使我养成了良好的生活习惯，在学习中遇到问题能够相互交流，不断鼓励，共同进步。还要感谢我的朋友们，他们让我学会了很多生活常识。

最后，感谢我的父母和姐姐们，感谢你们对我的鼓励，感谢你们对我生活

上的支持，我坚信自己会变得更好，不会令你们失望。