第 33 卷第 5 期　　河 北 科 技 大 学 学 报　　Vol. 33, No. 5
2012 年 10 月　　Journal of Hebei University of Science and Technology　　Oct. 2012

文章编号：1008-1542(2012)05-0443-05

# 在 8 位微程序控制的模型计算机中 Booth 算法的实现

王晓东，富　坤，耿恒山，秘海晓，孙晓丽

（河北工业大学计算机科学与软件学院，天津　300401）

**摘　要**：描述了在 8 位微程序控制的模型计算机中，通过编程实现了 Booth 算法的运算过程。对 Booth 算法进行了分析，绘出了实现 Booth 算法的流程图，编写了汇编语言程序，在 8 位微程序控制的模型计算机中实现了 Booth 算法，达到了预期的结果。

**关键词**：乘法；Booth 算法；模型计算机；补码

**中图分类号**：TP301　　**文献标志码**：A

# Implementation of Booth algorithm in the 8-bit microprogram controlled model computer

WANG Xiao-dong, FU Kun, GENG Heng-shan, MI Hai-xiao, SUN Xiao-li

(School of Computer Science and Engineering, Hebei University of Technology, Tianjin 300401, China)

**Abstract**: This paper describes the implementation of Booth algorithm in a 8-bit microprogram controlled model computer. Through analyzing the Booth algorithm by using the flow chart and program assembly language, it is found that the Booth algorithm can be applied to 8-bit microprogram controlled model computer.

**Key words**: multiplication; Booth algorithm; model computer; complement

计算机组成原理是计算机专业的核心课程之一。该课程中的部分运算方法详细阐述了数据的表示和运算，解决了计算机中最基本的问题，能使学习者认识到计算机在自动解题过程中数据信息的加工处理流程，从而进一步加深对计算机硬件组成及整体工作原理的理解。其中乘法又是很重要的一部分，由于补码相对于原码的优势，机器大都采用补码做加减运算，进而在乘法运算中使补码乘法得到普遍应用。Booth 算法是补码算法中最常用的一种算法，因此有必要深入学习该算法的原理和实现流程。本实验是在 8 位微程序控制的模型计算机上实现补码的 1 位乘法，这有助于巩固和加深课程中所学知识，熟练掌握对实验板的操作，以及掌握复杂指令在计算机中的工作流程[1-2]。

## 1　Booth 算法简介

Booth 算法也就是补码 1 位乘的比较法。被乘数为$[X]_{补}$，乘数为$[Y]_{补}$，$[P]_{补}$为乘积，按执行顺序得出每一步的部分积。

收稿日期：2012-06-01；修回日期：2012-09-06；责任编辑：陈书欣
基金项目：国家自然科学基金青年基金资助项目(31100711)
作者简介：王晓东(1987-)，男，河北大名人，硕士研究生，主要从事计算机控制技术方面的研究。

© 1994-2013 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

$[P_0]_{补}=0$，

$[P_1]_{补}=\{[P_0]_{补}+(Y_{n+1}-Y_n)[X]_{补}\}2^{-1}$，

⋮

$[P_{n+1}]_{补}=\{[P_n]_{补}+(Y_1-Y_0)[X]_{补}\}=[X\cdot Y]_{补}$。

Booth 算法的运算规则如下。

1)乘数的最低位为 $Y_n$，在其后再添加一位 $Y_{n+1}$，其值为 0。

2)$Y_{i+1}$与 $Y_i$ 为相邻 2 位，$(Y_{i+1}-Y_i)$有“0”，“1”和“−1”3 种情况。

①$Y_{i+1}-Y_i=0$($Y_{i+1}Y_i=00$ 或 11)，部分积直接右移 1 位。

②$Y_{i+1}-Y_i=1$($Y_{i+1}Y_i=10$)，部分积加$[X]_{补}$，右移 1 位。

③$Y_{i+1}-Y_i=-1$($Y_{i+1}Y_i=01$)，部分积加 $[-X]_{补}$，右移 1 位。

3)按以上执行 $n+1$ 步，最后一步($i=n+1$)不移动[3-4]。

## 2 8 位微程序控制的模型计算机

8 位微程序控制的模型计算机作为组成实验设备，该实验平台由微程序控制模块、时序电路模块、存储器模块和运算器模块等部分构成。通过输入微指令，配合时序电路，实现对数据的运算、存储等功能。8 位微程序控制的模型计算机是较早的一种 8 位组成原理实验设备，且各模块界限又是统一整体[5-6]，通过实验环节，进一步融合贯通所学内容，理解计算机内部各个模块的工作过程及其原理，以及复杂指令在计算机中的工作流程。

## 3 在 8 位微程序控制的模型计算机中实现 Booth 算法

### 3.1 实现方法

8 位微程序控制的模型计算机中寄存器由 2 片 74LS670 芯片组成，构成 4 个 8 位寄存器 A，B，C，D，笔者使用到其中的 A，B 和 C 3 个寄存器，各种运算操作由寄存器 A 和寄存器 B 配合完成，寄存器 C 作为计数器。静态存储器由 HM6116 芯片构成，被乘数、乘数和部分积以及其他一些数据都存放在存储器中，需要时调入到寄存器中。每一步操作开始时进行乘数最后 2 位的判断，基于该实验平台所支持的指令系统，并考虑到指令长度和冗余程度，本实验采取对后 2 位进行逐位判断的方式进行判断，进而得出 $[X]_{补}$或$[-X]_{补}$还是部分积直接移位。由于该实验平台上的存储单元为 8 位，部分积分为高位部分和低位部分，分别存储在 2 个存储单元中，部分积移位时，配合相应指令将部分积高位部分的最低位取出，该位数据替换到右移 1 位后的低位部分的最高位，接着将部分积的高位部分右移 1 位，完成整体部分积的右移操作。

### 3.2 Booth 算法控制流程

乘法运算前寄存器 A 被清零，作为初始部分积高位部分。寄存器 B 也相应清零。被乘数补码存储在寄存器 A 中，乘数的补码末尾补 0(作为附加位)后存放在寄存器 B 中，寄存器(计数器)C 存放乘数的位数 $n$。乘法开始后，根据寄存器 B 末 2 位 $B_n$，$B_{n+1}$的状态决定部分积与被乘数相加还是相减，或是自身值不变直接根据补码规则进行算术移位，这样重复 $n$ 次，最后根据寄存器 B 的末 2 位状态决定部分积是与被乘数相加还是相减，或者部分积的值不变，所得结果不必移位，即最终结果[7]。补码乘法乘积的符号位在运算中自然形成，见图 1。

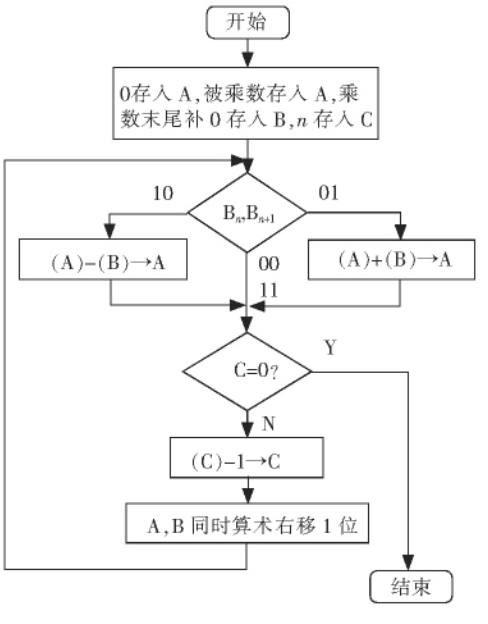


图 1 Booth 算法控制流程图

Fig. 1 Control flow graph of Booth algorithm

© 1994-2013 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

### 3.3　Booth 算法所需的硬件配置

Booth 算法运算基本硬件配置图见图 2。

图 2 中部分积高 8 位部分、低位以及乘数部分和被乘数都是存储在静态存储器 HM6116 芯片中，当需要进行运算时调入到寄存器，寄存器 A，B 分别为 74LS670 芯片中寄存单元，寄存器 A 中为部分积高位，调入到寄存器 B 中的数据由部分积低位，由部分的末 2 位判断得到，然后 2 个寄存器中数据送入 ALU 即 74LS181 进行运算，结果存储到原地址中，借助 ALU 和移位门完成右移操作。

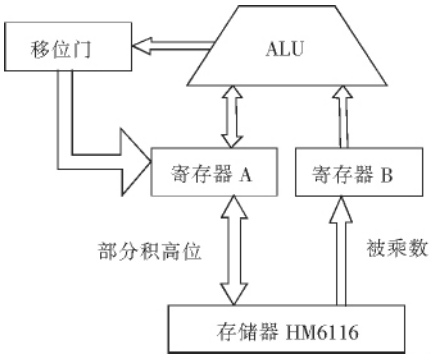


图 2　Booth 算法运算基本硬件配置

Fig. 2　Basic hardware configuration of Booth algorithm operation

### 3.4　操作步骤

本实验以 $X=-0.110101$，$Y=0.101101$ 为例，则 $[X]_{补}=11.001011$，$[Y]_{补}=0.101101$，$[-X]_{补}=0.110101$。此例题采用双符号位，按照 Booth 算法计算，计算过程见表 1。

表 1　Booth 算法计算过程

Tab. 1　Calculation process of Booth algorithm

| 部分积高位 | 低位及乘数 | 说明 |
|---|---|---|
| 00.000000 | 0.10110 10 | 初始值乘数最低位之后补 0 |
| + 00.110101 | | $Y_7Y_6=01$，$+[-X]_{补}$ |
| 00.110101 | | |
| →00.011010 | 1.01011 01 | 右移 1 位 |
| +11.001011 | | $Y_6Y_5=10$，$+[X]_{补}$ |
| 11.100101 | | |
| →11.110010 | 1.10101 10 | 右移 1 位 |
| + 00.110101 | | $Y_5Y_4=01$，$+[-X]_{补}$ |
| 00.100111 | | |
| →00.010011 | 1.11010 11 | 右移 1 位 |
| →00.001001 | 1.11101 01 | $Y_4Y_3=11$ 直接右移 1 位 |
| + 11.001011 | | $Y_3Y_2=10$，$+[X]_{补}$ |
| 11.010100 | | |
| →11.101010 | 0.11110 10 | 右移 1 位 |
| + 00.110101 | | $Y_2Y_1=01$，$+[-X]_{补}$ |
| 00.011111 | | |
| →00.001111 | 1.01111 01 | 右移 1 位 |
| + 11.001011 | | $Y_1Y_0=10$，$+[X]_{补}$ |
| 11.011010 | 1.0111101 | 除去最后 2 位为最终结果 |

$[X\cdot Y]_{补}=1.011010101111$，$X\cdot Y=-0.100101010001$。

在该实验平台上，按照机器执行顺序的具体操作步骤如下。

1）预置地址

根据该实验操作流程，将开关 $K_8$ 置于输入位置，合上开关 $K_6$ 进行单指令运行，按 $A_3$ 键进行总清；将初始地址 00000001 置于数据开关上，按 $A_2$ 键输入首地址[8]。

© 1994-2013 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

2）输入数据

将部分积、被乘数、乘数以及辅助运算数据存入静态存储器 HM6116 中。如 1）预置存储数据的首地址，将要输入的数据置于数据开关上，按 $A_1$ 键，这时输入的内容和单元的地址同时由数据灯和地址灯显示；如果下一个字节数据紧接上一个单元存放，则回到 2），2）中将输入的数据置于数据开关处继续，直至程序全部输入完毕，见表 2。表 2 中数据 10000000，00000010 等用于配合“与”指令提取对位数据。

表 2 内存中数据存储结构

Tab. 2 Data storage structure in memory

| 内存地址 | 数据 |
|---|---|
| 11000000 | 部分积高位，初始为 0 |
| 11000001 | 低位及乘数（最后补 0） |
| 11000010 | $[-X]_{补}$ |
| 11000011 | 0 |
| 11000100 | 10000000 |
| 11000101 | 00000010 |
| 11000110 | 00000001 |
| 11000111 | 11111100 |
| 11001000 | 00000110，C 中次数 |
| 11001001 | 01000000 |
| 11001010 | $[X]_{补}$ |

3）输入程序

本实验平台的指令系统有 30 条指令，结合这些指令实现该算法。根据上文控制流程描述的过程，按照 2）中输入数据的方式顺序输入程序。

本实验程序所实现的过程如下。

1）部分积进行加法运算。利用访问内存的 LDA 指令将低位及乘数提取到寄存器 B 中，再取出 00000001 放入寄存器 A 中，运用 ANA 指令运算得出只包含低位及乘数的最低位的 8 位二进制数，最低位分为 0 和 1 两种，结合 JZ 指令，在这 2 种情况下进一步判断低位及乘数的次低位，从而得出 01，10 或是 00(11)，按照所得结果从内存中取出相应数据，和部分积高位做加法运算，并存储到原部分积高位部分的地址中。

2）判断是否循环结束。寄存器 C 中存储的数据是循环次数，做完 1 次加法运算后，用 JZ 指令对寄存器 C 中数据进行 1 次判断，如果是 0 则表明循环操作结束，跳出循环至判断溢出部分，否则数据减 1 并存回寄存器 C 中，继续执行移位操作。

3）实现部分积右移。由于部分积存在于 2 个 8 位内存单元，因此对部分积进行移位操作分 2 部分进行。

①部分积低位右移。得到部分积高 8 位的最低位，将其存入寄存器 B 中的最高位位置，再从内存将部分积低位及乘数提取到寄存器 A 中，对寄存器 A 中数据进行算术右移并和寄存器 B 中数据进行“与”运算，完成部分积低位部分右移 1 位操作，将数据存回原内存单元；

②部分积高位右移。部分积的高位部分右移需要考虑到符号位，最高位符号位在移位时保持自身不变，取出部分积高位部分的最高位，将其放入寄存器 B 中最高位位置，这时寄存器 A 中的部分积高位部分右移 1 位并和寄存器 B 中数据进行“与”操作完成右移 1 位，存回原来内存单元中，从而完成部分积整体右移。再跳转到开头循环执行。表 3 为本实验所用到的指令[9]。

4）判断溢出

执行完以上操作后，对最后结果判断是否溢出，通过判断部分积的最高 2 位是否为相同数值即可得出结果，不同证明结果溢出跳转到停机，相同则输出结果。

5）检查内存

用 1）中方法，将要检查的单元地址 00000001 预置好，将 $K_8$ 置于检查处，合上开关 $K_6$，按 $A_1$ 键，这时被检查单元地址和内容分别由地址灯和数据灯显示，再按 $A_1$ 键顺序检查下一个单元内容，或者重新预置要检查的地址，重复上述操作直到检查完毕。

6）输出结果

首先按 1）所述步骤，预置程序入口地址，将 $K_8$ 置于执行程序位置，合上单微指开关 $K_7$ 和单指令开关 $K_6$，模型机为连续运行状态，按 $A_1$ 键顺序执行，最终在数据灯上显示乘积结果。

© 1994-2013 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

表 3　指令表

Tab. 3　Instruction list

| 指令 | 指令格式 | 说明 |
|---|---|---|
| STA address | 1000 00 00<br>address | 将输入的数据存储到内存中的相应地址 |
| LDA address | 0111 00 00<br>address | 将数据从内存中相应地址读出 |
| ADD r | 1001 00 r | 寄存器 A 和寄存器 r 数据相加 |
| ANA r | 1011 00 r | 寄存器 A 和寄存器 r 数据相与 |
| MOV $r_1$,$r_2$ | 0010 $r_1$ $r_2$ | 寄存器 $r_2$ 数据存储到寄存器 $r_1$ 中 |
| SHR | 1010 00 00 | 右移 1 位 |
| JMP address | 1101 00 00<br>address | 无条件跳转 |
| DCR M | 1111 11 00 | 寄存器 C 中循环计数减 1 |
| JZ address | 0101 11 00<br>address | 如果寄存器 A 中数据为 0,跳转到相应地址 |
| JCZ address | 1101 11 11<br>address | 如果寄存器 C 中数据为 0,跳转到相应地址 |

## 4　结　语

阐述了在 8 位微程序控制的模型计算机中 Booth 算法的实现过程,通过分析 Booth 算法运算规则,结合该实验平台自身特点和指令系统,设计并给出了详细的操作步骤。本实验作为组成原理课程的加深和延续,起到巩固所学知识的作用,加深了对计算机工作原理的理解,锻炼培养了实践能力和创新能力。

## 参考文献:


[1]　耿恒山. 计算机组成原理[M]. 北京:机械工业出版社,2009.

[2]　唐朔飞. 计算机组成原理[M]. 北京:高等教育出版社,2008.

[3]　陈苏豫. 用 BOOTH 算法改进的计算机定点乘法运算[J]. 晋中学院学报(Journal of Jinzhong University),2008,25(3):91-93.

[4]　孙启良. 二进制补码一位乘法规律的推导[J]. 电脑知识与技术(Computer Knowledge and Technology),2009,5(25):7 278-7 280.

[5]　谭小兰,陈　多,陈华光. 8 位模型机的设计与实现[J]. 湖南工程学院学报(自然科学版)(Journal of Hunan Institute of Engineering(Natural Science Edition)),2011,21(1):71-73.

[6]　李　英,汤作华,赵　勇. 对 NJS-Ⅱ型计算机组成实验仪的改进[J]. 华东地质学院学报(Journal of East China Geological Institute),2002,25(2):171-173.

[7]　王晓兰,吴秀敏,方运潭. JZYL-Ⅱ型计算机组成原理实验平台研制与开发[J]. 实验室研究与探索(Research and Exploration in Laboratory),2009,28(9):80-82.

[8]　李俊红,解建军. 定点补码乘法的一种实现方案[J]. 河北师范大学学报(自然科学版)(Journal of Hebei Normal University(Natural Science Edition)),2000,24(3):312-313.

[9]　刘乃文. 冯·诺依曼机原理的教学研究与应用[J]. 计算机工程与设计(Computer Engineering and Design),2006,27(10):1 831-1 834.

[10]　柏　磊,龙　涛. 基于单片机的数制转换算法设计[J]. 河北工业科技(Hebei Journal of Industrial Science and Technology),2010,27(4):229-231.


© 1994-2013 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net