基于 HOL4 的形式化方法①

张 杰[1], 饶文博[1], 王少超[1], 李晓娟[2]

[1](北京化工大学 信息科学与技术学院, 北京 100029)
[2](首都师范大学, 北京 100029)

摘 要: 针对计算机系统设计的正确性问题, 研究了一种在测试空间上完备的形式化方法, 探讨了硬件系统在定理证明器 HOL4 中进行形式化验证的一般方法, 其中包括如何采用高阶逻辑形式化描述系统的实现与规范, 以及在 HOL4 中证明目标的一般过程. 同时, 以乘法器为实例, 提出一种功能分解法对需要分析的电路进行形式化建模, 并对模型的性质在 HOL4 中进行推理与验证, 从而证明了乘法器电路设计的模型满足所提取的性质.

关键词: 定理证明; 形式化方法; 高阶逻辑; 乘法器; HOL4; 形式化建模

Formal Method Based on HOL4

ZHANG Jie[1], RAO Wen-Bo[1], WANG Shao-Chao[1], LI Xiao-Juan[2]

[1](College of Information Science & Technology, Beijing University of Chemical Technology, Beijing 100029, China)
[2](Capital Normal University, Beijing 100029, China)

Abstract: In order to solve the correctness issues of computer system design, a formal method which is complete on the test space is studied a general method of formal verification of hardware system in HOL4 is discussed in this paper. The method includes how to use the higher-order logic to describe the implementation and specification of hardware system and the general process of proving goals in HOL4. Meanwhile, taking multiplier as example, a functional decomposition method for modeling the design of circuit logically is proposed. Relevant properties of the circuit are ratiocinated and verified using the theorem prover HOL4, which proves that the model of the circuit design satisfies the properties.

Key words: theorem proving; formal method; high-order logic; multiplier; HOL4; formal modeling

不论是硬件系统还是软件系统, 随着系统复杂度的增加, 系统出现细微错误的可能性也会大大增加, 而有些细微的错误往往会造成巨大的经济损失, 甚至出现灾难性的后果, 因此系统验证问题已经成为人们研究的重点. 传统上的验证方法一般为模拟、仿真和测试等, 这些方法有时能非常有效地检测出系统存在的错误, 但是它们并不能检测目标的所有情况, 也就不能证明错误不存在. 本文研究的形式化方法是建立在严格的数学基础上, 利用数学方法证明系统的正确性, 这能涵盖所验证的性质的所有可能发生的情况, 因此它在测试空间上是完备有效的, 弥补了传统验证方法在这方面的不足.

目前, 形式化验证方法主要有等价性检验、模型检测和定理证明三种[1,2]. 其中, 等价性检验是检验两个模型之间的一致性, 在工业上已得到广泛应用; 模型检测已实现自动化, 但存在状态空间爆炸问题; 而定理证明是依据严格的数学公理和推理规则, 并结合定理证明器来推理和证明系统模型的性质. 本文所用的是定理证明方法, 结合剑桥大学开发的交互式定理证明器 HOL4(Higher-Order Logic 4)进行推理证明[3]. 近年来, HOL4 系统在许多领域得到了应用, 如: Tahar 等人在 HOL4 系统中完成了 Z 变换的形式化[4]; 李黎明等人利用 HOL4 系统形式化验证了一种机械臂避障算法的正确性[5].

① 基金项目:国家自然科学基金(61373034)
收稿时间:2015-05-12;收到修改稿时间:2015-06-03

乘法器是高性能微处理器和数字信号处理器等器件中的核心运算部件. 它不仅用于数学运算, 而且更在图像、加密、语音等数字信号处理领域起着非常重要的作用[6]. 目前乘法器的研究主要集中在如何减少延时, 提高它的运算速度, 从而实现低功率且高速运行的乘法器[7,8]. 但是实现乘法器高速运行的前提是要保证乘法器的设计的正确性, 一个有错误的乘法器是没有任何价值的, 因此对它的验证也是必要的. 乘法器的验证常用的是仿真方法, 但是这种方法在测试空间上是不完备的.

本文以乘法器为例, 探讨了基于 HOL4 系统的形式化验证的一般流程, 首次将其应用于乘法器求和电路的验证. 不仅对求和电路的性质规范进行了形式化描述, 还对其结构设计进行了逻辑抽象建模, 并且基于 HOL4 系统进行了形式化验证.

1 形式化验证的一般流程

形式化验证方法是数学方法论在系统可信性验证上的具体应用, 它是将所要验证的系统的规范和实现用具有形式语义的符号表达出来, 并根据相应的数学理论证明其正确性的一种方法[9,10]. 其中, 具有形式语义的符号可以是一阶逻辑, 也可以是时态逻辑或高阶逻辑[11], 本文基于 HOL4 系统的研究, 并采用高阶逻辑, 相比于其他逻辑语言, 高阶逻辑拥有更强大的表达能力. 对于硬件系统, 形式化验证就是检验系统的实现是否符合它的规范, 具体过程如图 1 所示, 其中, 实现是系统的设计结构, 规范是该系统所满足的性质.

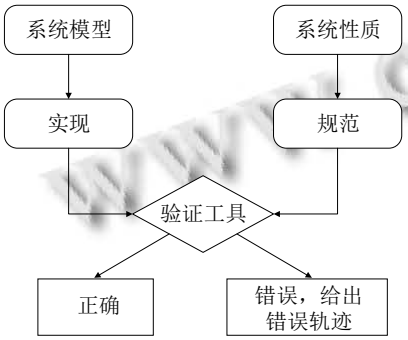

图 1 形式化验证流程

如图 1 所示, 验证者根据系统模型和系统性质, 用高阶逻辑表示出该系统的实现与规范, 这个过程为形式化建模过程; 然后用户可在定理证明器中证明系统实现是否蕴含系统规范, 即形式化验证过程; 最后根据证明结果得出结论. 下面将以一个乘法器为研究对象具体地分析形式化验证方法.

2 乘法器在HOL4中的形式化建模

乘法器的运算原理与手动乘法运算相似, 如图 2 所示, 从最低位到最高位, 用乘数的每一位依次与被乘数相乘, 每次相乘得到的部分积都左移一位, 最后把所有的部分积相加就得到最终的乘积.

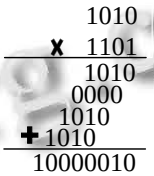

图 2 乘法运算

在乘法器中, 如果乘数的比特位为 0, 则相应的部分积为 0; 如果乘数的比特位为 1, 则相应的部分积为被乘数. 因此, 可以说乘法器的一般运算分两步, 一是计算得出所有部分积, 二是对所有部分积求和[12]. 本文忽略部分积产生电路, 只对乘法器运算的第二个部分部分积求和电路进行形式化建模与验证, 乘法器求和电路的具体形式化验证过程如图 3 所示.

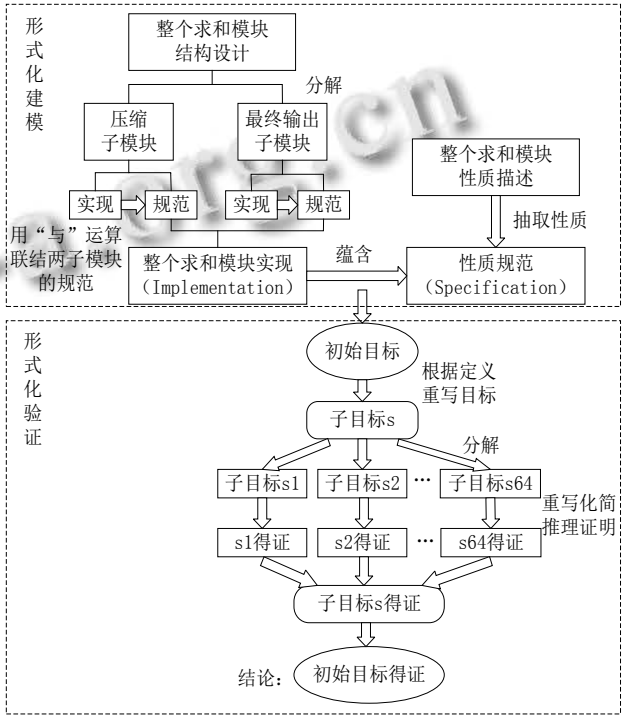

图 3 验证流程

为了采用定理证明对乘法器的部分积求和过程进

行形式化验证，首先要依据高阶逻辑语言的特点对部分积求和过程的规范和实现进行逻辑抽象建模，即形式化建模. 形式化建模是根据部分积求和过程的功能和结构特点，用逻辑谓词进行形式化描述它的规范和实现，其中规范是验证对象所要达到的性质，实现是验证对象的设计结构，最后证明实现蕴含规范.

2.1 性质规范的形式化建模

由于性质规范是表征待验证系统运行过程中所具有的性质或功能. 而乘法器运算的目的是得出两个操作数的乘积，即在乘法器的部分积求和过程中，所有部分积求和之后得到的结果应该为初始的两个操作数之积功能，用 PreSum 表示部分积之和，用 FinalValue 表示部分积经过求和后得到的最终结果，如果整个部分积求和电路的设计是正确的， PreSum 和 FinalValue 相等，用谓词 PPS_SPEC 表示它们的相等关系.

性质 1. 只要电路设计是正确的，无论二进制输入信号为高电平还是低电平，PreSum 和输出 FinalValue 都是等于两个操作数的乘积.

∀a1 a0 b1 b0 c1 c0 s0 s1 s2 s3.

PPS_SPEC(a1 a0 b1 b0 c1 c0 s0 s1 s2 s3) =

(PreSum(a1,a0,b1,b0,c1,c0)=

FinalValue(s0,s1,s2,s3))

其中 a0、a1、b0、b1、c0 和 c1 为电路的输入信号，s0、s1、s2 和 s3 为电路的输出信号，∀a. x(a)表示“对所有 a, x(a)成立”.

由于在实际电路中输入输出信号都是高电平或低电平，用布尔量 T 和 F 表示，而在部分积的和为一个整数值，故首先需要定义一个布尔量到整型量的转换函数，定义如下:

定义 1. bv_def

|- ∀ x. bv(x) = if x then 1 else 0

上述函数 bv 的类型为: bool?int，作用是当输入信号 x 为高电平，即 T 时，函数值为 1；否则为 0.

因为最后比较的乘积是十进制数，故还需要将二进制数转换为十进制数，定义一个函数 value，如下所示，假设为四位的二进制.

定义 2. value_def

|- ∀ a b c d. value(a,b,c,d) = 23 * bv(d) + 22 * bv(c) + 21 * bv(b) + 20 * bv(a)

函数 value 的功能是将四位二进制数转换为相应的十进制数，其中 a、b、c 和 d 是二进制的每一位数值，为 0 或 1. 二位二进制数转换为十进制数的表示形式与之类似，用 value(m,n)表示.

所有部分积在运算之前的和用谓词 PreSum 表示的形式如下所示:

∀ a1 a0 b1 b0 c1 c0.

PreSum(a1,a0,b1,b0,c1,c0) = value(a0,a1) + value(b0,b1) + value(c0,c1)

运算之后得到的结果用谓词 FinalValue 表示的形式如下所示:

∀ s0 s1 s2 s3. FinalValue(s0,s1,s2,s3) = value(s0,s1,s2,s3)

2.2 设计实现的形式化建模

乘法器中的部分积求和过程由两部分构成，分别是部分积压缩过程和最后两个部分积求和过程. 为了更加清晰简便地来对部分积求和过程进行形式化建模，将它分为压缩模块和求和模块，再分别对这两个模块的设计实现进行逻辑抽象建模，最后采用功能分解法对整个模块进行形式化验证，如图 4 所示.

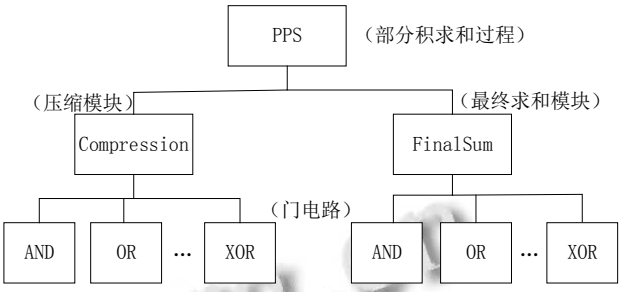

图 4 功能分解法结构框图

图 4 中 PPS 表示乘法器中的整个部分积求和模块，依据功能分解法，将它划分为不同的模块，先分别对每个模块进行形式化建模与验证，再将每一个模块当成一个整体，依据子模块的规范对更高一层的模块进行验证，最后将子模块用逻辑“与”联结起来组成整个电路的设计结构模型. 具体的过程如下:

在 HOL4 系统中，目标的验证是证明实现蕴含规范，即:

|- PPS_IMP ⇒ PPS_SPEC

在图 1 中，PPS 划分为 Compression 模块和 FinalSum 模块:

|-PPS_IMP⇔Compression_IMP∧FinalSum_IMP

则有:

|-Compression_IMP ∧ FinalSum_IMP ⇒PPS_SPEC

所以可以先验证子模块 Compression 和 FinalSum 电路设计的正确性, 从而证明模块 PPS 的正确性:

|- Compression_IMP ⇒ Compression_SPEC

|- FinalSum_IMP ⇒ FinalSum_SPEC

则验证目标为:

|- Compression_SPEC ∧ FinalSum_SPEC ⇒ PPS_SPEC

同理, 模块 Compression 和 FinalSum 的验证可以来自更低一层的逻辑门电路的验证, 从而使所有模块得到证明.

2.2.1 压缩模块

压缩模块是在不产生进位传递时延的情况下, 减少部分积的数量, 直到剩下最后两个部分积, 这一过程的实现通常采用进位保留加法器(Carry-Save Adder, CSA). 由于这种加法器可以使输出个数减少, 故也称为压缩器. 根据输入输出之间的个数比, 有 3:2 压缩器、4:2 压缩器和 5:3 压缩器, 本文研究的 3:2 压缩器是其中最常用的压缩器之一, n 位操作数的进位保留加法器可由 n 个全加器构成, 如图 5 所示.

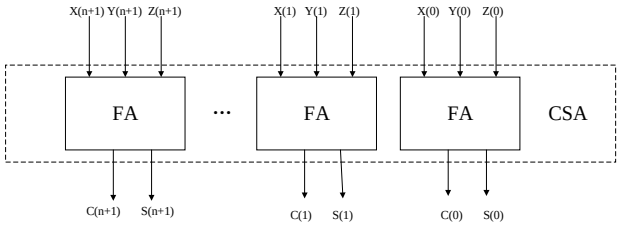

图 5 功能分解法结构框图

图 5 中, 每位产生的进位保留在结果 c_i 中, 并不累加到下一位的本位和 s_{i+1}, 每一位的相加操作相互独立, 可并行执行. 根据全加器的输入输出之间的逻辑关系可知:

$$s_i = x_i \oplus y_i \oplus z_i \quad (1)$$

$$c_i = x_i \cdot y_i + z_i \cdot (x_i + y_i) \quad (2)$$

其中⊕指异或运算.

无论三个输入信号的每一位 x、y 和 z 为高电平还是低电平, 都有进位输出 c 和本位和输出 s, 且 xyz 表示的整数之和等于 c 表示的整数的两倍与 s 表示的整数相加之和, 它们之间的代数关系用谓词 Compression_SPEC 表示:

∀x y z c s.

Compression_SPEC (x,y,z,c,s) = (bv(x) + bv(y) + bv(z) = 2 * bv(c) + bv(s))

根据全加器的逻辑表达式, 将式(1)和(2)用“与”运算符联结成一个表达式, 可得到进位保留加法器每一位的实现, 用谓词 Compression_IMP 表示:

∀ x y z c s.

Compression_IMP (x,y,z,c,s) = (c = (x ∧ y) ∨ (x ∧ z) ∨ (y ∧ z)) ∧(s = (x XOR y) XOR z)

其中, ∧表示与运算, ∨表示或运算, 谓词 XOR 表示异或运算, 它的定义如下所示:

定义 3. XOR_def

|- ∀ x y. XOR x y = (¬x ∧ y) ∨ (x ∧ ¬y)

可以在 HOL4 中用 set_fixity 将 XOR 设置为中缀符.

2.2.2 最终求和模块

最终求和模块是将压缩后剩余的两个部分积进行求和, 为了较少时延, 通常采用超前进位加法器(Carry Look-ahead Adder, CLA). 图 6 为 4 位超前进位加法器的逻辑结构, 其中 P 是两个输入信号的“异或”, G 是两个输入信号的“与”.

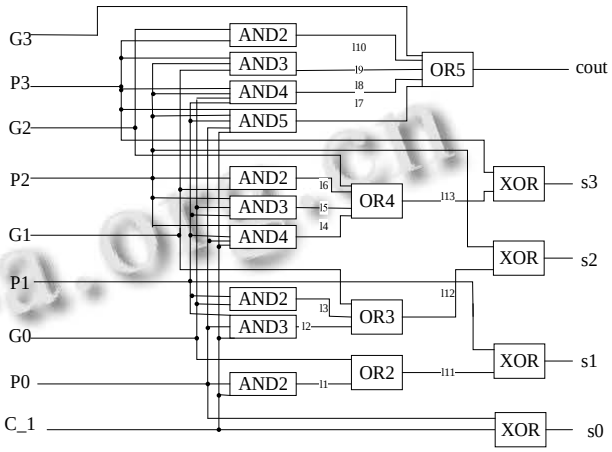

图 6 四位超前进位加法器的逻辑结构图

定义 4. AND2_def

|- ∀inp1 inp2 out. AND2(inp1, inp2, out) = (out = inp1 ∧ inp2)

谓词 AND2 具有与门的功能, 用来形式化两个信号的“与”运算和相应的输出, 另外, 谓词 AND3、AND4 和 AND5 也具有与门功能, 不同的是输入信号的数量.

定义 5. OR2_def

|- ∀inp1 inp2 out. OR2(inp1, inp2, out) = (out = inp1 ∨ inp2)

谓词 OR2 具有或门的功能，用来形式化信号的“或”运算和相应的输出，同样，谓词 OR3、OR4 和 OR5 具有或门功能.

定义 6. XOR2_def

|- ∀inp1 inp2 out. XOR2(inp1, inp2, out) = (out = (inp 1∧ ¬inp2) ∨ (¬inp1 ∧ inp2))

谓词 XOR2 具有异或门的功能，用来形式化信号的“异或”运算和相应的输出.

完成上述谓词的定义后，用 l1, l2, l3, l4, l5, l6, l7, l8, l9, l10, l11, l12, l13 表示逻辑门间的内部信号，并将这些内部信号用存在量词表示，再将这些谓词用逻辑“与”连接起来，即可得到超前进位加法器电路设计实现的形式化模型，用谓词 FinalSum_IMP 表示，如下所示:

∀p0 p1 p2 p3 g0 g1 g2 g3 c_1 cout s0 s1 s2 s3.

FinalSum_IMP(p0,p1,p2,p3,g0,g1,g2,g3,c_1,cout,s0,s1,s2,s3) =

∃ l1 l2 l3 l4 l5 l6 l7 l8 l9 l10 l11 l12 l13.

AND2(g2,p3,l10) ∧ AND3(p3,p2,g1,l9) ∧

AND4(p3,p2,p1,g0,l8) ∧

AND5(p3,p2,p1,p0,c_1,l7)∧

AND2(p2,g1,l6) ∧ AND3(p2,p1,g0,l5) ∧

AND4(p2,p1,p0,c_1,l4) ∧ AND2(p1,g0,l3) ∧

AND3(p1,p0,c_1,l2) ∧ AND2(p0,c_1,l1) ∧

OR5(g3,l10,l9,l8,l7,cout) ∧

OR4(g2,l6,l5,l4,l13)∧

OR3(g1,l3,l2,l12) ∧ OR2(g0,l1,l11) ∧

XOR2(p3,l13,s3) ∧ XOR2(p2,l12,s2) ∧

XOR2(p1,l11,s1) ∧ XOR2(p0,c_1,s0)

无论输入信号是低电平还是高电平，只要电路设计时正确的，超前进位加法器的输出信号必定满足如下关系，用谓词 FinalSum_SPEC 表示:

∀p0 p1 p2 p3 g0 g1 g2 g3 c_1 cout s0 s1 s2 s3.

FinalSum_SPEC(p0,p1,p2,p3,g0,g1,g2,g3,c_1,cout,s0,s1,s2,s3) =

(s0 = p0 xor c_1) ∧

(s1 = p1 xor (g0 ∨ (p0 ∧ c_1))) ∧

(s2 = p2 xor (g1 ∨ (p1 ∧ g0) ∨ (p1 ∧ p0 ∧ c_1)))∧

(s3 = p3 xor (g2 ∨ (p2 ∧ g1) ∨ (p2 ∧ p1 ∧ g0) ∨ (p2 ∧ p1 ∧ p0 ∧ c_1))) ∧

(cout = g3 ∨ (p3 ∧ g2) ∨ (p3 ∧ p2 ∧ g1) ∨ (p3 ∧ p2 ∧ p1 ∧ g0) ∨ (p3 ∧ p2 ∧ p1 ∧ p0 ∧ c_1))

2.2.3 PPS 模块

在完成了两个子模块的形式化验证后，整个求和过程中可以把压缩模块和最终求和模块当成一个整体，不再考虑它们内部的结构，如图 7 所示，a0a1b0b1c0c1 和 cin 是输入信号，s0s1s2s3 是输出信号，m0m1n0n1 是它们的内部信号.

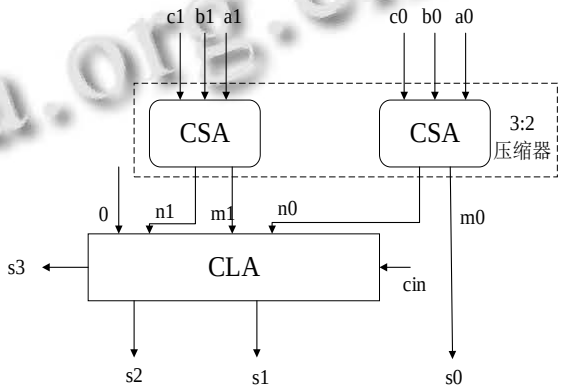

图 7　整体求和过程结构图

整体求和过程的设计实现可以用两个子模块的性质规范表示，将谓词 Compression_SPEC 和谓词 FinalSum_SPEC 中的谓词表达式用“与”运算联结起来，则谓词 PPS_IMP 的表示形式如下:

∀ a0 a1 b0 b1 c0 c1 cin s0 s1 s2 s3.

PPS_IMP(a0,a1,b0,b1,c0,c1,cin,s0,s1,s2,s3) =

∃ m0 m1 n0 n1.

Compression_SPEC(a0,b0,c0,m0,n0) ∧

Compression_SPEC(a1,b1,c1,m1,n1) ∧

FinalSum_SPEC(m0,m1,n0,n1,cin,s0,s1,s2,s3)

在形式化建模过程中，需要合理地从电路结构中提取出描述里蕴含的逻辑关系，并准确、完整地使用形式化语言来建模.

3　基于HOL4的验证

完成整体求和过程的规范和实现的形式化建模后，以实现蕴含规范为初始目标，在定理证明器 HOL4 中建立此目标，即在 HOL4 中完成欲证性质的形式化描述，系统自动建立一个未完成的目标堆栈，并显示初始证明目标，如图 8 所示.

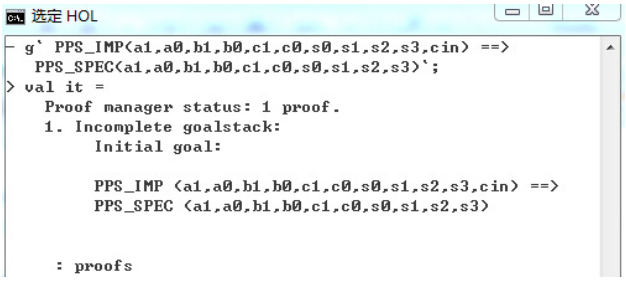

图 8 初始目标

在 HOL4 系统中建立初始目标后，再根据内置的公理、推理规则和已定义的定理来证明初始目标，证明结果如图 9 所示:

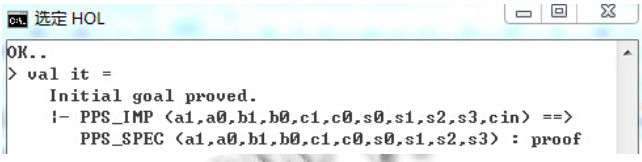

图 9 验证结果

在图 9 中显示的 Initial goal proved 结果表明初始目标已经得到证明，这说明了整个部分积求和模块的电路设计满足性质 1，即由该模块最后得到的结果是所有部分积之和.

在 HOL4 中的验证，首先需要熟悉 HOL4 系统中的对策、化简集和理论，并针对不同阶段证明目标的具体特点，选择恰当的对策及定理完成目标的化简和证明.

4 结语

本文采用形式化方法，利用交互式定理证明器 HOL4 完成了对乘法器中部分积求和模块电路设计的形式化验证. 由于在 HOL4 中进行形式化验证需要人机交互，且很难实现对大规模系统的验证，本文提出一种功能分解法，简化了验证的复杂度，同时减少了验证过程的工作量. 更加重要的是，本文使用的定理证明法是根据严格的数学推理来证明系统设计的正确性，它的测试空间是完备的，正好弥补了传统方法在这方面的不足. 另外，这种形式化方法能够帮助设计人员在设计过程早期发现错误或者证明其设计方案是正确的，因此如将这种验证方法引入硬件电路的设计阶段也是非常有意义的.

但是，基于 HOL4 的定理证明方法也有着它的局限性，由于形式化方法的验证过程是不能实现自动化证明，仍需要人机交互来逐步引导推理证明过程，因此对验证者的要求过高. 未来的研究工作，可以本文研究的基础上，进一步研究更多位或 n 位乘法器的验证，以期完成乘法器设计验证的一般性.

参考文献

1 Gordon M. Twenty-years of theorem proving for HOLs past, present and future. Lecture Notes in Computer Science of the 21st International Conference on Theorem Proving in Higher-Order Logics. Montreal, Canada. Springer Berlin Heidelberg. 2008.

2 姚全珠,王江.基于 UML 的软件形式化需求分析与验证.计算机工程,2010,36(13):30–33.

3 Slind K, Norrish M. A brief overview of HOL4. Lecture Notes in Computer Science of the 21st International Conference on Theorem Proving in Higher Order Logics. Montreal, Canada. Springer Berlin Heidelberg. 2008.

4 Siddique U, Mahmoud MY, Tahar S. On the formalization of z-transform in HOL. Lecture Notes in Computer Science of the 5th International Conference on Interactive Theorem Proving. Vienna, Austria. Springer International Publishing. 2014.

5 Li LM, Shi ZP, Guan Y, et al. Formal verification of a collision-free algorithm of dual-arm robot in HOL4. IEEE International Conference on Robotics & Automation. Hong Kong, China. IEEE. 2014.

6 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[硕士学位论文].西安:西安电子科技大学,2011.

7 Kang JY, Gaudiot JL. A simple high-speed multiplier design. IEEE Trans. on Computers, 2006, 55(10): 1253–1258.

8 Jackuline MD, Anu PK. Design of low-power and high performance Radix-4 multiplier. International Conference on Devices, Circuits and Systems. IEEE Press. 2012. 432–435.

9 张玉鹏,施智平,关永,李黎明,赵春娜,张杰. SpaceWire 译码电路在 HOL4 中的形式化验证.小型微型计算机系统,2013,34(8):1959–1963.

10 韩俊刚,杜慧敏.数字硬件的形式化验证.北京:清华大学出版社,2001.

11 Huth M, Ryan M. Logic in Computer Science. UK: University of Cambridge, 2004.

12 Lin HL, Chang RC, Chan MT. Design of a novel Radix-4 booth multiplier. IEEE Asia Pacific Conference on Circuits and Systems. IEEE. 2004.