高级综合中控制信息的提取与综合*

叶梅龙 张东晓 恒东辉 金 毅

(北京理工大学ASIC研究所 北京 100081)

摘要 本文扼要地论述了高级综合的过程和其中控制信息的提取与变换,实现了控制流综合与数据流综合结果的衔接,并对FPGA Xilinx设计库单元映射成工艺相关的ASIC,直至生成FPGA器件.

关键词 FSMD,调度,分配,有限状态自动机,时序逻辑综合.

中图法分类号 TP391

1. 高级综合的模型

寄存器传输级综合(又称时序综合)基于有限状态自动机(FSM)模型,而对高级综合而言,需要在FSM基础上增加数据通道模型.

有限状态自动机可以形式化地描述成五元组形式

$$\langle S, I, O, f: S\times I\to S, h: S\times I\to O\rangle$$

其中S为状态集合,I为输入集合,O为输出集合,f为状态转换函数,h为输出函数.

当状态很多时,例如,即使是复杂度较低的元件如I/O接口或总线控制器都至少有几千个状态,这时,FSM模型就很难适应设计需要.为使FSM模型适应更复杂设计的需要,引入了带数据通道的有限自动机(FSMD)的概念.这里数据通道可以是定点或浮点寄存器,这些寄存器可以存储状态变量.

FSMD定义如下:设VAR为存储变量的集合,EXP为表达式的集合

$$EXP=\{f(x,y,z,\ldots)\mid x,y,z,\ldots\in VAR\}$$

A为赋值语句的集合 $$A=\{x\Leftarrow e\mid x\in VAR, e\in EXP\}$$

$STAT$为两组表达式之间的关系(状态信号)

$$STAT=\{Relation(a,b)\mid a,b\in EXP\}$$

则FSMD可以形式化地表示成如下五元组形式

$$\langle S, I\times STAT, O\times A, f: S\times(I\times STAT)\to S, h: S\times(I\times STAT)\to O\times A\rangle$$

可以看出,FSMD的后继状态不仅与当前状态和外部输入信号有关,而且和内部状态信号

---

* 本文研究得到国家自然科学基金资助.作者叶梅龙,女,1938年生,教授,主要研究领域为电子设计自动化EDA,VHDL语言高级综合.张东晓,1972年生,博士生,主要研究领域为VHDL语言,高级综合.恒东辉,1971年生,硕士,主要研究领域为VHDL语言,逻辑综合.金毅,女,1973年生,硕士生,主要研究领域为VHDL语言及建库技术,工艺映时.

本文通讯联系人:叶梅龙,北京100081,北京理工大学ASIC研究所

本文1996-12-06收到修改稿

有关. 另外,FSMD 不仅要更新外部信号的值,而且还要更新内部数据通道中寄存器中变量的值(如图 1 所示).

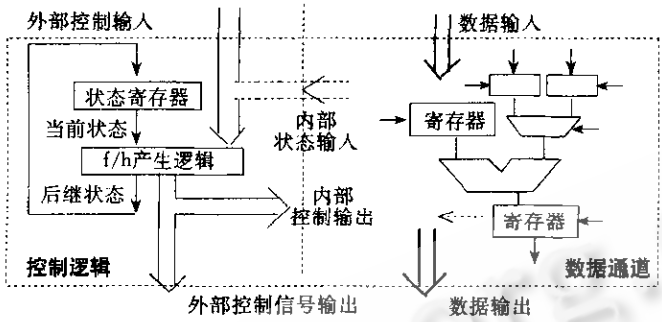


图1 FSMD模型实现示意图

2. 高级综合的过程[1,2]

从图 1 中可以看出,FSMD 模型可分为两个部分:控制逻辑部分和数据通道部分. 控制逻辑部分完成状态及时序控制;数据通道完成信息的处理、存储与传输. 这两部分的实现通常可分别称为控制流综合和数据流综合. 控制流综合涉及的主要方法是传统的寄存器传输级综合和逻辑综合,最终用两级的 PLA 或多级的门网络来实现控制器;而数据流综合则涉及高级综合的理论与方法. 其主要过程如下.

(1) 中间数据结构的生成

综合时首先要对输入的行为描述进行编译,然后将编译结果转换为适于综合的、信息完备的中间数据结构. 常用的中间数据表示有 CDFG,VT(value trace),ADD 等. 北京理工大学高级综合系统 HLS/BIT 以工业标准硬件描述语言 VHDL 作为输入[3],采用控制数据流图(CDFG)作为中间数据结构,在生成 CDFG 之前,HLS/BIT 首先利用编译技术将源描述中的操作序列划分为一系列状态段.

图 2(a)是一个示意性的 VHDL 行为描述,它完成表达式

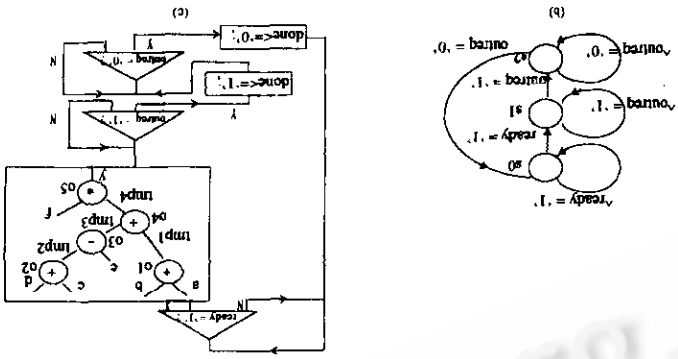
$$y=((a+b)+(e-(c+d)))*f$$

的计算,其相应的状态段及 CDFG 分别如图 2(b)和(c)所示.

```
...
begin
wait until clk'event and clk = '1' and ready = '1';
tmp1:=a+b;
tmp2:=c+d;
tmp3:=e-tmp2;
tmp4:=tmp1+tmp3;
y<=tmp4*f;
wait until clk'event and clk = '1' and outreq = '1';
done<='1';
wait until clk'event and clk = '1' and outreq = '0';
done<='0';
end process;
...
```

(a) VHDL 描述

图 2



(b) 状态段转换　　(c) CDFG

图 2

(2) 调度(Scheduling)

CDFG 中的 CFG(控制流图)用于生成电路的控制信息，DFG(数据流图)用于生成电路数据通道. 调度ッ任务是把 DFG 中的操作按时间顺序分配到各个控制步(时钟周期)，从而确定操作执行的时序，并使 DFG 中的各个操作的执行既符合数据相关，又能在满足设计约束的前提下实现最大限度的并行.

图 3(a)和(b)是对图 2(c)中 DFG 的两种调度结果，图 3(b)调度结果的实现造价较低.

(3) 分配(Allocation)

分配的任务是根据每个控制步的寄存器传输方程推导出数据通道，它又分为两个子任务：单元选择和单元装配. 单元选择确定设计中所使用的 RTL 基元的种类和数目，单元装配是将 DFG 中的操作、变量及数据传输映射到选定的硬件基元，图 4 是对应于图 3(b)调度的一种分配结果. 图中 tmp1，tmp2，tmp4 共享寄存器 reg1，reg2 存放 tmp3.

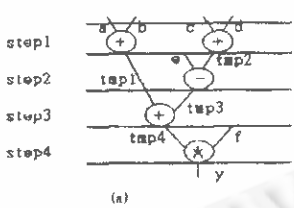


图 3　调度结果

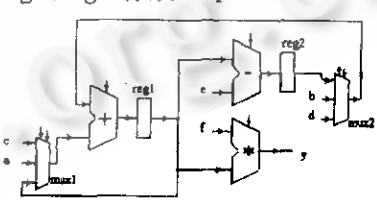


图 4　分配结果

分配结束后，产生了设计的数据通道部分，另外，根据控制流图及调度过程中产生的状态信息，可以通过控制流综合综合出控制器部分. 这样，数据通道与控制器就组成了一个完整的结构化设计.

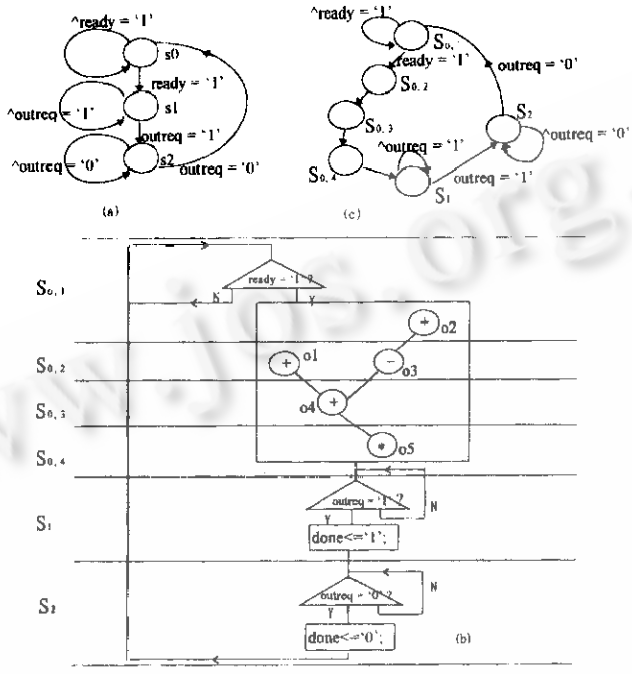
1　控制信息的提取

控帜信息提取的任务就是找出状态转换函数 fs 和控制输出函数 fo. 控制信息最终通过控制流综合生成相应的控制部件，用来控制数据通道的执行.

1.1　状态转换函数的提取

状态转换函数是输入变量集合 I 与状态集合 S 到状态集合 S 上的一个映射. HLS/BIT

的综合过程中首先将操作流程转化为相应的状态段，并确立各状态段间的转换关系. 状态段可以视为一系列状态的集合，每个状态对应于调度的一个控制步，每个控制步都无条件地转换到下一个控制步. 图 5 表示了图 2(a)中 VHDL 描述对应的状态段，调度后的 CDFG 及最终形成的状态转换图.



(a) 状态段转换 (b) 调度后的 CDFG (c) 最终的状态转换图

图 5 状态段转换、调度 CDFG 及最终状态转换图的对应关系

从图 5 中的对应关系可以看出，状态转换函数由两方面的因素决定：状态段间的转换关系和状态段内调度的控制步.

若状态段 s_i 中控制步 j 对应的状态记作 $s_{i,j}$，在条件 c_i 下，状态段 s_i 转移到 s_{i+1} 则有

$$s_{i,j} \rightarrow s_{i,j+1} \qquad s_{i,\max step(i)} \xrightarrow{c_i} s_{I+1,1}$$

其中→表示状态转换关系，$\max step(i)$ 表示 s_i 的最大控制步数目.

1.2 控制输出函数的提取

控制输出函数为输入变量 I 与状态集合 S 到输出变量集合 O 的映射. 输出变量用于控制数据通道中数据的处理、存储与传输. 因此，控制输出函数确定操作的执行条件，变量的存储条件和数据传输的传输条件，进而确定相应功能单元、存储单元和互连单元的执行或选通条件. 实际上，VHDL 的各种控制语句包括 if，case，loop，wait 语句等，在综合时都是用于产生状态转换(如图 5 所示)，结合上述状态段的概念，可以导出操作的执行条件：

$$c_{oi}=\begin{cases} state=s_{i,j}(j>1)，若操作\ o_i\ 在状态段\ s_i\ 中调度到控制步\ j \\ state=sp_i \wedge c_i=true，若操作\ o_i\ 在状态段\ s_i\ 中调度到控制步\ 1\ 且\ sp_i \xrightarrow{c_i} s_i \end{cases}$$

同理可以得到变量的存储条件 C_{mi} 和数据的传输条件 C_{ti}.

在综合完成分配工作后,某些操作、变量或数据传输可能共享同一功能单元、存储单元或互连单元,则这些元件的控制条件为共享该单元的所有操作、变量或数据传输条件的"或".对功能单元,设功能单元 FU_k 可执行的操作集合为$\{o_{k1},\ldots,o_{kn}\}$,操作 o_{ki} 的执行条件为 C_{ki},则 FU_k 的执行条件 C_{FU_k} 为各操作执行条件的"或"

$$C_{Fu_k}=\bigvee_{i=1}^{n}C_{ki}$$

同理可得出存储单元和互连单元的控制条件.当两个或多个以上信息引到同一元件的输入端时,则需加入多路选择器,设多路选择器的输入为$(t_1,\ldots,t_n)$,则多路选择器的端口选择条件可以表示为

$$C_{mux}=i \text{ if } C_{ti}=true$$

2　控制信息变换为有限状态自动机

控制流综合的输入信息是从数据流综合的结果文件 *cfile* 中提取出来的,其中的状态转换信息是从数据流综合系统的调度过程提取的,而控制函数则是由数据流综合系统的分配过程提取的.信息提取后,首先进行转换,并存入预先定义的各种表中.

提取与转换包括两部分内容:首先,从状态转换表和控制函数中提取信息填写到事先定义好的各种表中;然后,再把状态转换条件和控制信号条件转变成多维体形式表示.

存储表格有标识符表、输入表、表达式表、状态转换表、控制信号表、输出函数表、位串表等.通过 *YACC* 把相关信息写入各种表中.表达式的处理最为复杂,它要访问输入表和标识符表,而且表达式的内容随后还要转变成多维体表示.

控制流信息转变成多维体表示算法如下:

第 1 步:查找输入表中的所有变量 Var_i,并计算它们的宽度$|Var_i|$,假设输入表中有 n 个变量,则有 $inpt\_cube\_size\leftarrow\sum_{i=1}^{n}|Var_i|$.

第 2 步:从状态转换表中取出一行 $STT(CS,NS,CO)$,把状态转换条件(表达式)转变成多维体表示:$trans\_to\_cube(CO)$.

第 3 步:当状态转换表不为空时,转第 2 步,否则算法结束.

表达式的形式为(op,left,right),op 为操作符,left,right 为操作数,设转变后的多维体为 *cube*,则 $trans\_to\_cube(expr)$ 的算法如下:

第 1 步:$cube\leftarrow U_n=XX\ldots X$;

第 2 步:若 $expr=\Phi$,则返回;否则,顺序执行;

第 3 步:查找输入表中和 OP 相同的输入,并把 *cube* 中相应的位元素由"*X*"转变成"1",这是 OP 为标识符的情况,否则

第 4 步:当 OP 不为标识符时,

```
CASE OP OF
AND:
    cube ← cube_set_and(trans_to_cube(op→left),trans_to_cube(op→right))
OR:
    cube ← cube_set_or(trans_to_cube(op→left),trans_to_cube(op→right))
NOT:
    cube ← cube_set_not(trans_to_cube(op→left))
```

返回.

这样,将高层次语义的有限状态自动机描述转换成内部的多张表格并建立相互联系,使离散的关系联系起来,将面向人的数据表示形式转化为面向机器的表示形式.在扫描 cfile

文件时还进行相应的语法、逻辑关系检查.

转换结果表明,控制信号表是将当前状态和外部输入信号结合起来,单独表示成控制信号.这些控制信号的标识符被用于在控制点信息表中的Condition域中,表示控制外部输出函数产生的当前状态和外部输入信号.而状态转换表则不包含输出信息,仅指明:在当前状态下,在外部输入信号的激励下将要转向的下一状态.

2.1 控制流描述抽象成时序机

根据控制流形式化描述,可以抽象成Mealy型时序机.

(1) 状态集S 状态转换表中的所有状态可以看作一个时序机的状态集.

(2) 输入I 所有状态转换条件和控制函数中出现的基本器件变量可作为时序机的输入.设基本输入变量有n个,则输入个数为n个.

(3) 输出O 控制函数即为时序机的输出.

(4) 状态转换函数N 由状态转换表非显式地给出.

(5) 输出函数Z 条件语句中的控制信号,可作为时序机的输出函数.

2.2 状态转换条件和控制函数的形式化表示

采用多维体集合表示状态转换条件和控制输出逻辑之后,状态转换表可用三元式表示:

$$STT(CS,NS,CON)$$

其中CS为当前状态,NS为下一状态,CON为状态转换条件(多维体集合).

控制逻辑也可以用三元式表示

$$CF(F,S,CO)$$

其中F表示控制信名;S为当前状态,在该状态下,F才有可能为逻辑"1";CO为控制条件(多维体集合).

3 正确性检查

对有限状态自动机描述文件(cfile文件)的正确性检查是在提取过程中进行的,检查包括两部分内容:(1)词法、语法正确性检查:借助LEX,YACC实现;(2)逻辑关系正确性检查:在内部表已经建立起来以后进行,主要功能有:外部输入信号宽度一致性检查;控制条件存在检查;状态逻辑条件完全性和一致性检查.

4 与周边系统的接口

控制流综合子系统与数据流综合子系统的接口原先只有一个cfile文件,该文件由数据流综合提取的控制信息组成,由一个或多个部分组成.到目前为止,最多有5部分,每部分包括3张表:控制信号表、状态转换表、控制点信息表.经与数据流综合协调,由数据流综合再提供一个接口文件:cfile.in.该文件提供了外部输入信号的名字、位数、元件号、端口号及时钟信息.控制流综合子系统根据cfile.in修改iosign.pla文件,将修改后的iosign.pla文件传递给工艺映射子系统,供其进一步使用.

5 结 论

本文阐明高级综合中调度与分配的过程.讨论了从数据流综合的调度过程提取出状态转换信息;从资源分配过程提取出控制函数,并利用 UNIX 系统的 LEX 和 YACC 将其转换后存入预先定义的表格中.再经处理后即可转换成 Mealy 型时序机,乃至多维体列阵完成高级综合与有限状态自动机的综合的衔接.

本文工作的成功是完成时序逻辑综合并进行工艺映射[4]之后,又与数据流综合结果准确地联成整机,不但能自动生成全机逻辑图,而且可将该结果转换成 VHDL 结构描述,并进行模拟验证.特别是已有若干设计实例生成了 FPGA 器件并给出正确运行结果.

参考文献

1 刘明业.数字系统自动设计.北京:高等教育出版社,1996.

2 刘明业,张东晓,许庆平.VHDL 高级综合系统设计中某些关键问题的技术决策.全国首届 VHDL 语言及其应用学术会议,1996.

3 张东晓,刘明业.VHDL 语言高级综合子集的确立及其实现方法.全国首届 VHDL 语言及其应用学术会议,1996.

4 马聪等.VHDL 高级综合与底层物理设计的衔接.全国首届 VHDL 语言及其应用学术会议,1996.

EXTRACTION AND SYNTHESIS OF CONTROL INFORMATION IN HIGH LEVEL SYNTHESIS

YE Meilong　ZHANG Dongxiao　HENG Donghui　JIN Yi

(*ASIC Research Center　Beijing Institute of Technology　Beijing*　100081)

Abstract　The authors discuss concisely, the process of high level synthesis and the extraction and transformation of control information generated in the said process. The authors also discuss the methodology of linking the result of data flow synthesis and that of control flow synthesis as well as the mapping to technology-dependent ASIC based on Xilinx FPGAs.

Key words　FSMD, scheduling, allocation, finite state machine, sequential logic synthesis.

Class number　TP391