第01章 概 述

编译器是将一种程序（源程序）翻译成另一种程序（目标程序）的计算机程序。本书介绍一个非常简单的编译器（ TinyC 编译器）的实现。

TinyC 编译器可将 TinyC 源程序编译成 Linux 下的可执行程序，其编译及运行基本流程如下图，首先利用 TinyC 前端将 TinyC 源程序编译成中间代码 Pcode ，再利用 Nasm 的宏程序将 Pcode 翻译成 x86（32位） 汇编指令，然后利用 Nasm 汇编成二进制目标程序，最后链接成 Linux 下的 32 位可执行程序，可直接在 Linux 下运行。另外，中间代码 Pcode 也可以用 Pcode 模拟器直接运行。

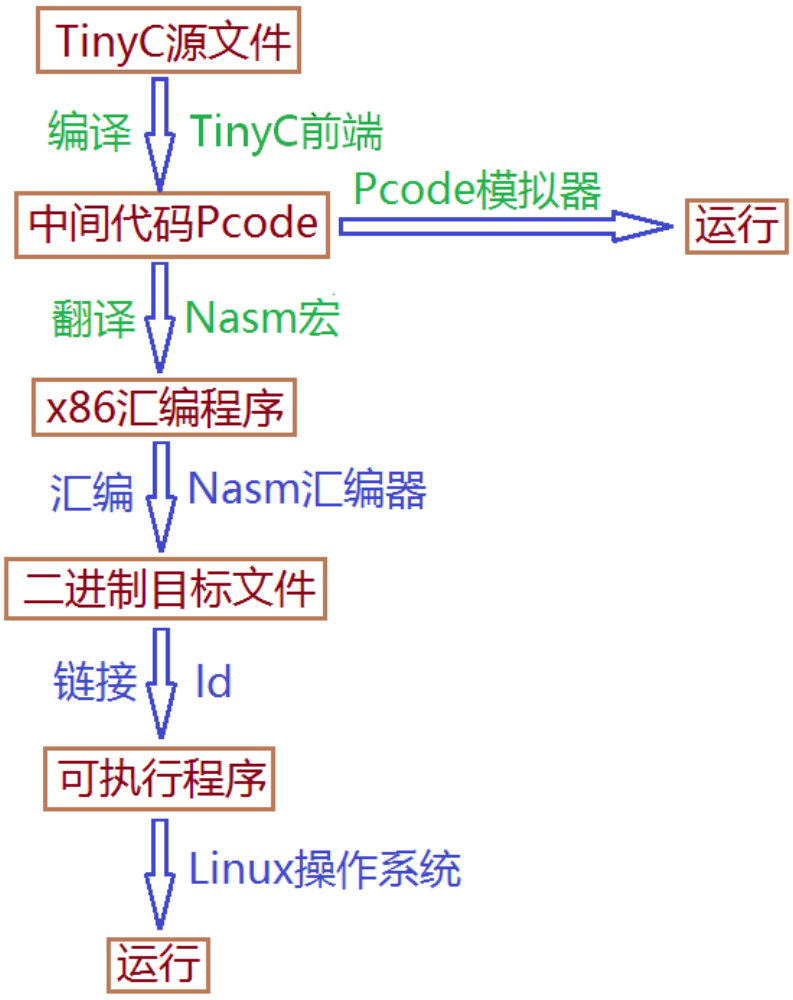

图**1.1 TinyC** 编译及运行基本流程

上图中绿色部分的 TinyC 前端、 Nasm 宏以及 Pcode 模拟器为由本人编程实现的部分，其他则是利用了 Nasm 和 ld 来生成可执行的二进制机器码。本书将 Nasm 宏、 Nasm 汇编器、 ld 链接器一起称为 TinyC 后端，将 TinyC 前端和 TinyC 后端一起称为 TinyC 编译器。

本书的组织结构如下：

第 01 章概述 TinyC 编译器的基本流程以及本书的组织结构；

第 02 章介绍源程序 TinyC 的语法；

第 03 ～ 04 章介绍中间代码 Pcode 的语法、 Pcode 虚拟机的结构以及如何使用 Pcode 模拟器来运行 Pcode ；

第 05 章介绍手工将 TinyC 源程序翻译成中间代码 Pcode 的方法；

第 06 章系统介绍编译器的基本流程；

第 07 ～ 08 章介绍词法分析的基本方法，以及如何使用 flex 进行词法分析；

第 09 章介绍上下文无关语法以及语法分析的基本思路；

第 10 ～ 13 章介绍两种典型的语法分析方法——自顶向下分析法和自底向上分析法，详细描述这两种分析方法的构造流程及解析流程，并介绍如何使用 bison 进行语法分析；

第 14 章介绍 TinyC 前端的实现，从一个简陋的语法分析器开始，一步一步的扩充为完整的 TinyC 前端；

第 15 章介绍 TinyC 后端的实现，针对所有 Pcode 命令编写相应的同名 Nasm 宏，将 Pcode 中间代码翻译成 x86 汇编，然后汇编、链接成 Linux 操作系统下的可执行程序。

第 16 章将 TinyC 前端和后端整合，形成完整的 TinyC 编译器。

本书的主要目的是用简单的实例来说明编译原理中的一般过程及原理，让读者在自己动手的实践过程中对编译原理有更深的理解，读者需要具备以下基础知识：

（1） 熟悉计算机系统架构，熟悉 x86 汇编；

（2） 熟练使用 C 语言，熟悉常见的算法，熟悉指针、内存布局等概念；

（3） 会简单的 Linux 终端操作命令，会简单的 makefile 语法。

TinyC 编译器在 Linux/debian 系统下开发完成，建议读者也使用 debian 或其他桌面版 Linux ，若在 windows 环境下，可以使用 VirtualBox 装一个虚拟 Linux 系统。本书所有源程序均放在 https://github.com/pandolia/tinyc 。

第 **1** 章完