



下载APP



01 | 程序的运行过程：从代码到机器运行

2021-05-10 LMOS

[进入课程 >](#)**讲述：陈晨**

时长 08:14 大小 7.56M



你好，我是 LMOS。

欢迎来到操作系统第一课。在真正打造操作系统前，有一条必经之路：你知道程序是如何运行的吗？

一个熟练的编程老手只需肉眼看着代码，就能对其运行的过程了如指掌。但对于初学者来说，这常常是很困难的事，这需要好几年的程序开发经验，和在长期的程序开发过程中对编程基本功的积累。



我记得自己最初学习操作系统的时候，面对逻辑稍微复杂的一些程序，在编写、调试代码时，就会陷入代码的迷宫，找不到东南西北。

不知道你现在处在什么阶段，是否曾有同样的感受？我常常说，**扎实的基本功就像手里的指南针，你可以一步步强大到不依赖它，但是不能没有。**

因此今天，我将带领你从“Hello World”起，扎实基本功，探索程序如何运行的所有细节和原理。

一切要从牛人做的牛逼事说起

第一位牛人，是世界级计算机大佬的传奇——Unix 之父 Ken Thompson。


在上世纪 60 年代的一个夏天，Ken Thompson 的妻子要回娘家一个月。呆在贝尔实验室的他，竟然利用这极为孤独的一个月，开发出了 UNiplexed Information and Computing System (UNICS) ——即 UNIX 的雏形，一个全新的操作系统。

要知道，在当时 C 语言并没有诞生，从严格意义上说，他是用 B 语言和汇编语言在 PDP-7 的机器上完成的。



牛人的朋友也是牛人，他的朋友 Dennis Ritchie 也随之加入其中，共同创造了大名鼎鼎的 C 语言，并用 C 语言写出了 UNIX 和后来的类 UNIX 体系的几十种操作系统，也写出了对

后世影响深远的第一版 “Hello World”：

 复制代码

```
1 #include "stdio.h"
2 int main(int argc, char const *argv[])
3 {
4     printf("Hello World!\n");
5     return 0;
6 }
```

计算机硬件是无法直接运行这个 C 语言文本程序代码的，需要 C 语言编译器，把这个代码编译成具体硬件平台的二进制代码。再由具体操作系统建立进程，把这个二进制文件装进其进程的内存空间中，才能运行。

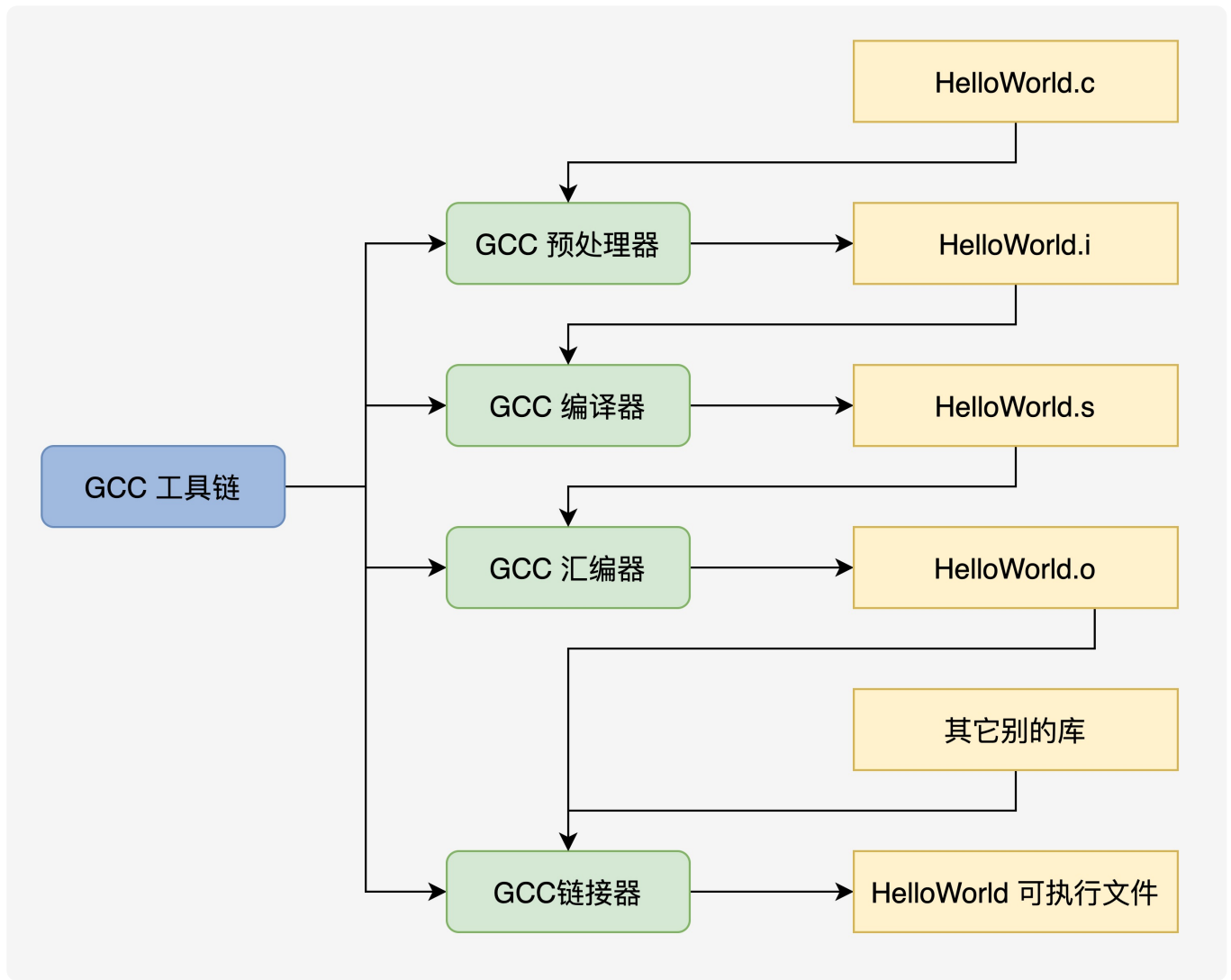
听起来很复杂？别急，接着往下看。

程序编译过程

我们暂且不急着摸清操作系统所做的工作，先来研究一下编译过程和硬件执行程序的过程，约定使用 GCC 相关的工具链。

那么使用命令：gcc HelloWorld.c -o HelloWorld 或者 gcc ./HelloWorld.c -o ./HelloWorld，就可以编译这段代码。其实，GCC 只是完成编译工作的驱动程序，它会根据编译流程分别调用预处理程序、编译程序、汇编程序、链接程序来完成具体工作。

下图就是编译这段代码的过程：



HelloWorld编译流程

其实，我们也可以手动控制以上这个编译流程，从而留下中间文件方便研究：

`gcc HelloWorld.c -E -o HelloWorld.i` 预处理：加入头文件，替换宏。

`gcc HelloWorld.c -s -c HelloWorld.s` 编译：包含预处理，将 C 程序转换成汇编程序。

`gcc HelloWorld.c -c HelloWorld.o` 汇编：包含预处理和编译，将汇编程序转换成可链接的二进制程序。

`gcc HelloWorld.c -o HelloWorld` 链接：包含以上所有操作，将可链接的二进制程序和其它别的库链接在一起，形成可执行的程序文件。

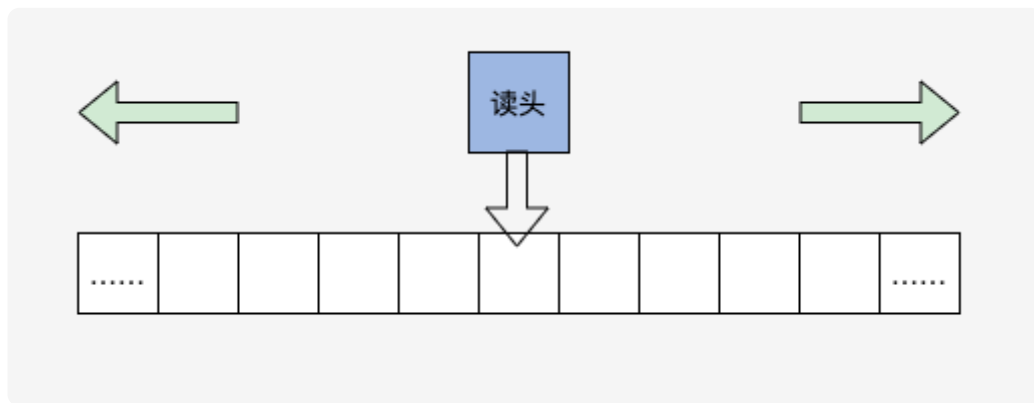
程序装载执行

对运行内容有了了解后，我们开始程序的装载执行。

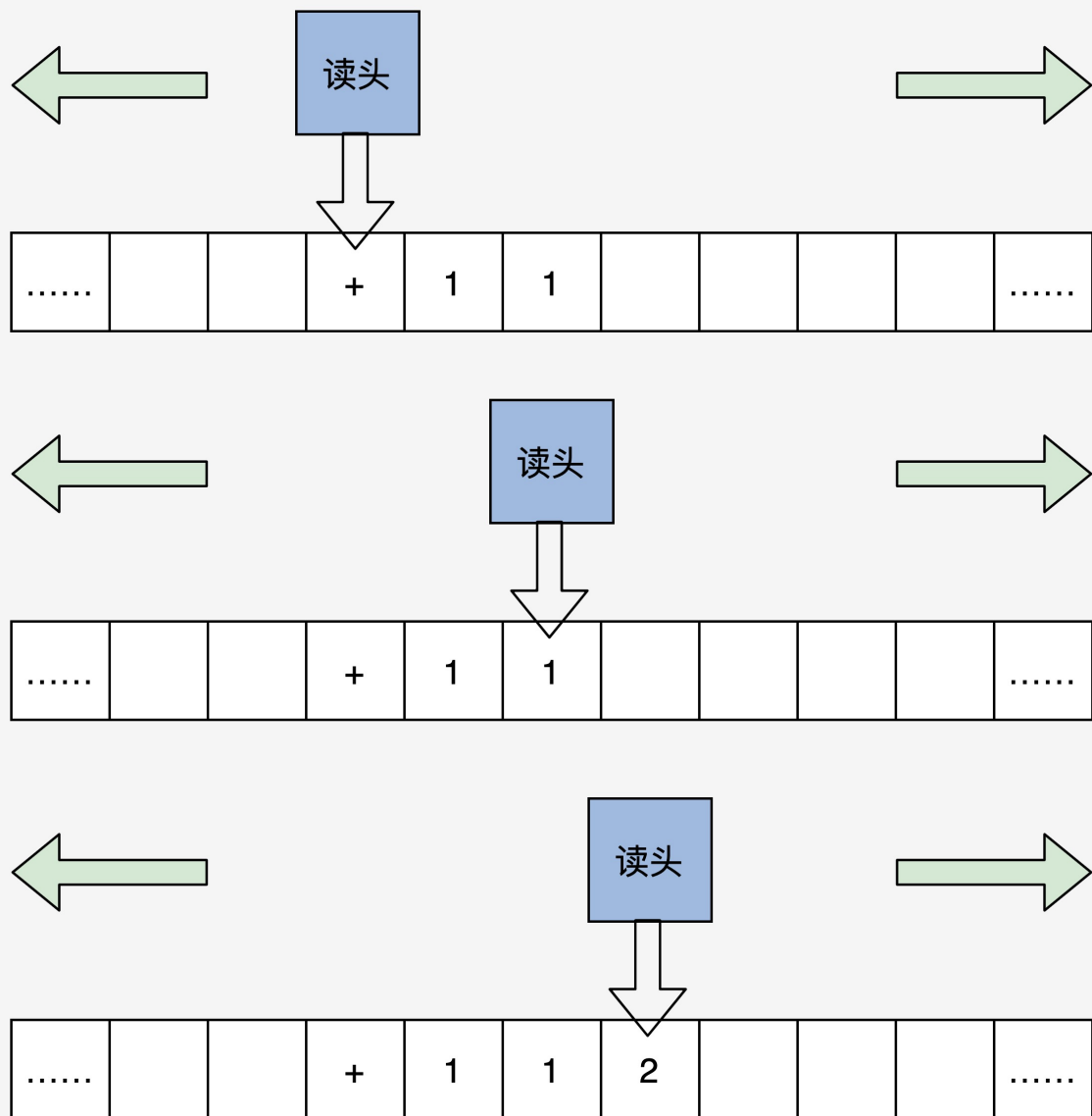
我们将请出**第三位牛人——大名鼎鼎的阿兰·图灵**。在他的众多贡献中，很重要的一个就是**提出了一种理想中的机器：图灵机**。

图灵机是一个抽象的模型，它是这样的：有一条无限长的纸带，纸带上有无限个小格子，小格子中写有相关的信息，纸带上有一个读头，读头能根据纸带小格子里的信息做相关的操作并能来回移动。

文字叙述还不够形象，我们来画一幅插图：



不理解？下面我再带你用图灵机执行一下“ $1+1=2$ ”的计算，你就明白了。我们定义读头读到“+”之后，就依次移动读头两次并读取格子中的数据，最后读头计算把结果写入第二个数据的下一个格子里，整个过程如下图：



图灵机计算过程演示

这个理想的模型是好，但是理想终归是理想，想要成为现实，我们得想其它办法。

于是，第四位牛人来了，他提出了电子计算机使用二进制数制系统和储存程序，并按照程序顺序执行，他叫冯诺依曼，他的电子计算机理论叫冯诺依曼体系结构。

根据冯诺依曼体系结构构成的计算机，必须具有如下功能：

把程序和数据装入到计算机中；

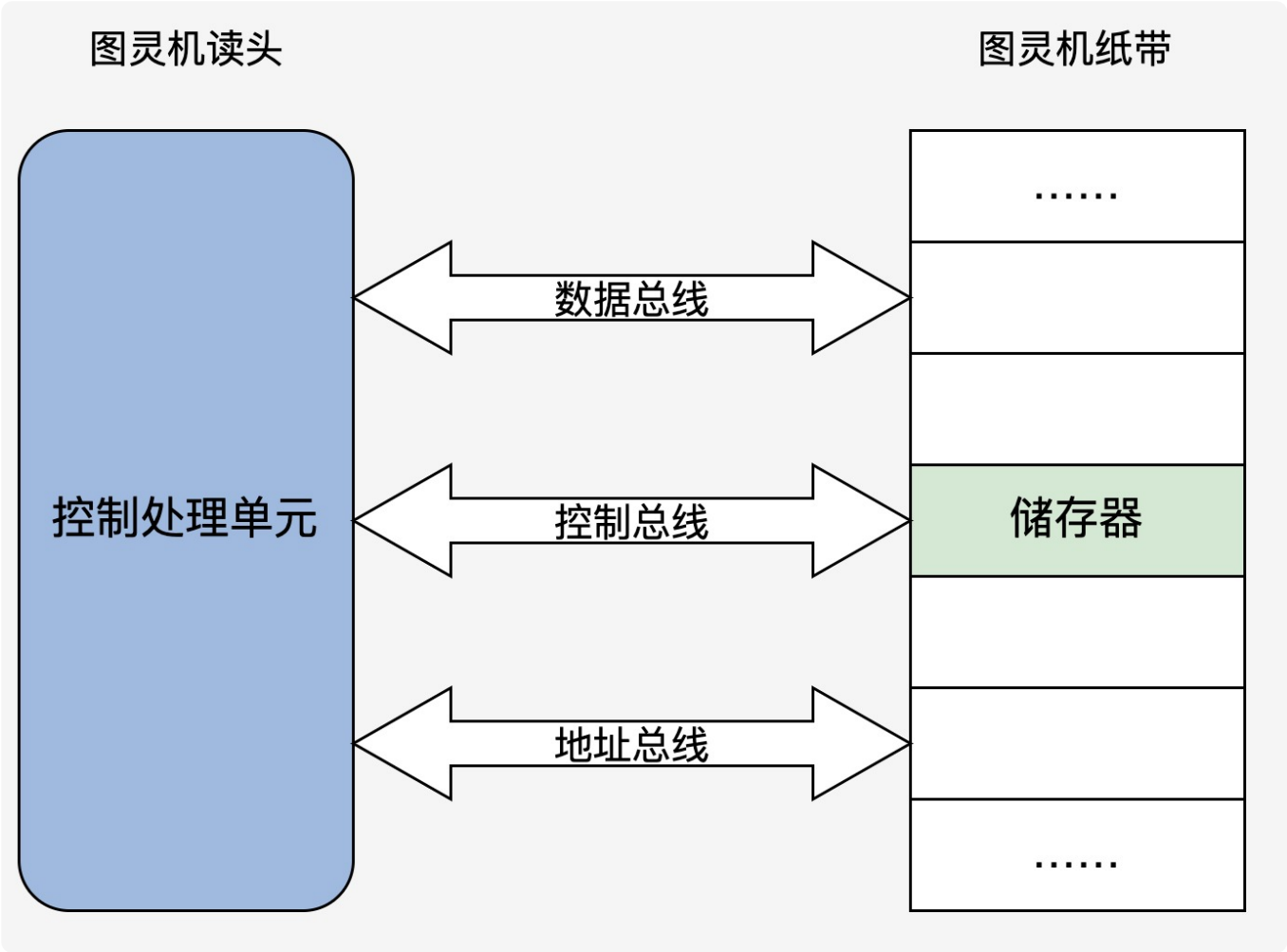
必须具有长期记住程序、数据的中间结果及最终运算结果；

- 完成各种算术、逻辑运算和数据传送等数据加工处理；
- 根据需要控制程序走向，并能根据指令控制机器的各部件协调操作；
- 能够按照要求将处理的数据结果显示给用户。

为了完成上述的功能，计算机必须具备五大基本组成部件：

- 装载数据和程序的输入设备；
- 记住程序和数据的存储器；
- 完成数据加工处理的运算器；
- 控制程序执行的控制器；
- 显示处理结果的输出设备。

根据冯诺依曼的理论，我们只要把图灵机的几个部件换成电子设备，就可以变成一个最小核心的电子计算机，如下图：



是不是非常简单？这次我们发现读头不再来回移动了，而是靠地址总线寻找对应的“纸带格子”。读取写入数据由数据总线完成，而动作的控制就是控制总线的职责了。

更形象地将 HelloWorld 程序装入原型计算机

下面，我们尝试将 HelloWorld 程序装入这个原型计算机，在装入之前，我们先要搞清楚 HelloWorld 程序中有什么。

我们可以通过 `gcc -c -S HelloWorld` 得到（只能得到其汇编代码，而不能得到二进制数据）。我们用 `objdump -d HelloWorld` 程序，得到 `/lesson01/HelloWorld.dump`，其中有很多库代码（只需关注 main 函数相关的代码），如下图：

```
0000000000000063a <main>:
63a: 55                push    %rbp
63b: 48 89 e5          mov     %rsp,%rbp
63e: 48 83 ec 10       sub     $0x10,%rsp
642: 89 7d fc          mov     %edi,-0x4(%rbp)
645: 48 89 75 f0       mov     %rsi,-0x10(%rbp)
649: 48 8d 3d 94 00 00 00 lea     0x94(%rip),%rdi    # 6e4 <_IO_stdin_used+0x4>
650: e8 bb fe ff ff   callq   510 <puts@plt>
655: b8 00 00 00 00   mov     $0x0,%eax
65a: c9               leaveq  %eax
65b: c3               retq
65c: 0f 1f 40 00      nopl    0x0(%rax)
```

以上图中，分成四列：第一列为地址；第二列为十六进制，表示真正装入机器中的代码数据；第三列是对应的汇编代码；第四列是相关代码的注释。这是 x86_64 体系的代码，由此可以看出 x86 CPU 是变长指令集。

接下来，我们把这段代码数据装入最小电子计算机，状态如下图：

图灵机读头

图灵机纸带

伪代码

PS：上图内存条中，一个小格子中只要一个字节，但是图中放的字节数目不等，这是为了方便阅读，不然图要画得很大。

重点回顾

以上，对应图中的伪代码你应该明白了：现代电子计算机正是通过内存中的信息（指令和数据）做出相应的操作，并通过内存地址的变化，达到程序读取数据，控制程序流程（顺序、跳转对应该图灵机的读头来回移动）的功能。

这和图灵机的核心思想相比，没有根本性的变化。只要配合一些 I/O 设备，让用户输入并显示计算结果给用户，就是一台现代意义的电子计算机。

到这里，我们理清了程序运行的所有细节和原理。还有一点，你可能有点疑惑，即 `printf` 对应的 `puts` 函数，到底做了什么？而这正是我们后面的课程要探索的！

这节课的[配套代码](#)，你可以从[这里](#)下载。

思考题

为了实现 C 语言中函数的调用和返回功能，CPU 实现了函数调用和返回指令，即上图汇编代码中的 “`call`”，“`ret`” 指令，请你思考一下：`call` 和 `ret` 指令在逻辑上执行的操作是怎样的呢？

期待你在留言区跟我交流互动。如果这节课对你有所启发，也欢迎转发给你的朋友、同事，跟他们一起进步。

57 人觉得很赞 | 提建议

© 版权归极客邦科技所有，未经许可不得传播售卖。页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

上一篇 开篇词 | 为什么要学写一个操作系统？

下一篇 02 | 几行汇编几行C：实现一个最简单的内核

精选留言

写留言

由作者筛选后的优质留言将会公开显示，欢迎踊跃留言。