Linux System Programming

Zhang Haining

Guizhou University

hnzhang1@gzu.edu.cn

April 3, 2018

Overview

System Programming

System programming is the art of writing system software.System software lives at a low level, interfacing directly with the kernel and core system libraries.

There are three cornerstones to system programming in Linux:

1. system calls
2. the C library
3. the C compiler

System Calls

System programming starts with system calls.System calls (often shorted to syscalls) are function invocations made from user space—your text editor, favorite game, and so on—into the kernel (the core internals of the system) in order to request some service or resource from the operating system.

Invoke system calls

It is not possible to directly link user-space applications with kernel space. For reasons of security and reliability, user-space applications must not be allowed to directly execute kernel code or manipulate kernel data. Instead, the kernel must provide a mechanism by which a user-space application can "signal" the kernel that it wishes to invoke a system call. The application can then trap into the kernel through this well-defined mechanism, and execute only code that the kernel allows it to execute. The exact mechanism varies from architecture to architecture.

The C Library

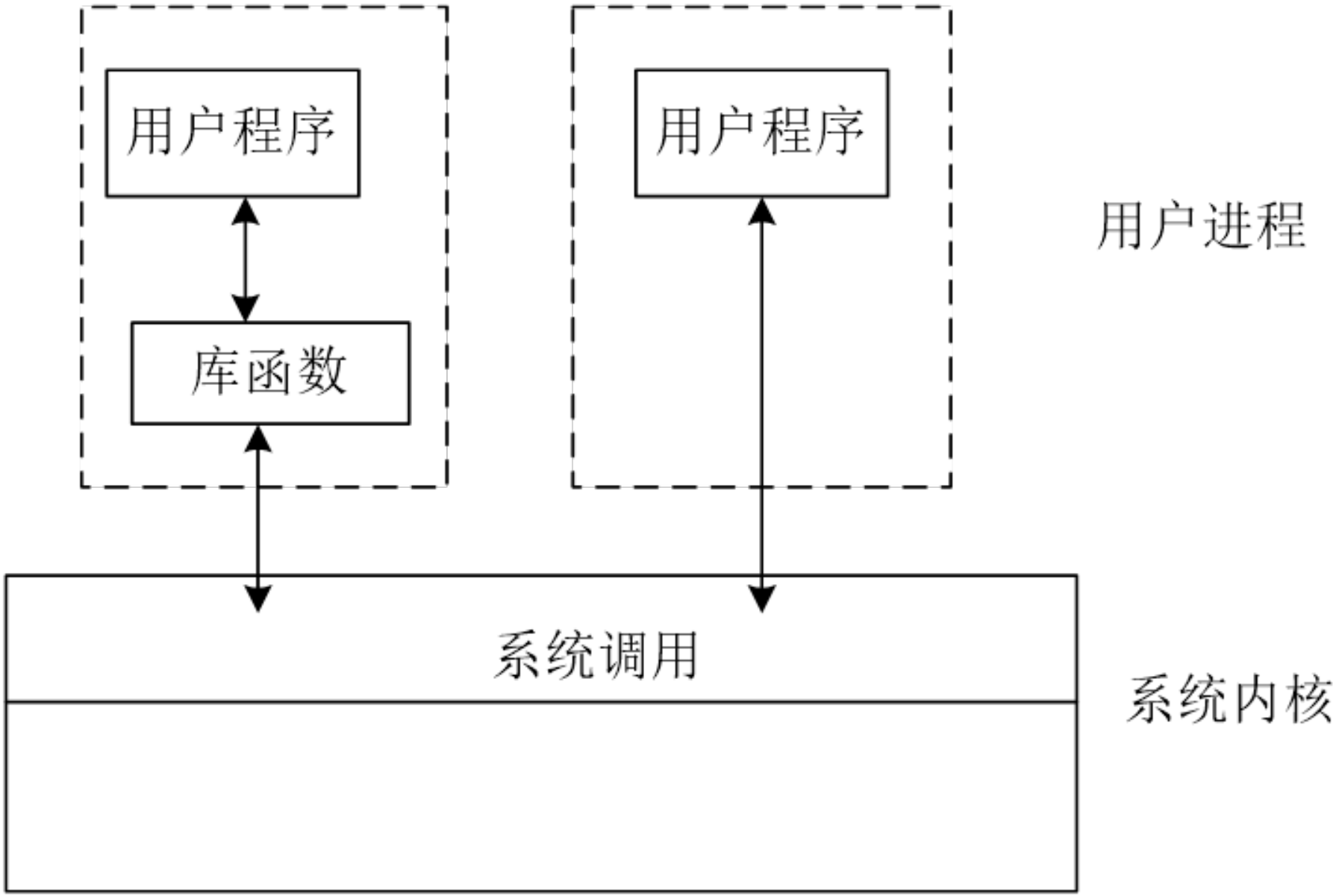
The C library (libc) is at the heart of Unix applications. Even when you're programming in another language, the C library is most likely in play, wrapped by the higher-level libraries, providing core services, and facilitating system call invocation. On modern Linux systems, the C library is provided by GNU libc, abbreviated glibc, and pronounced gee-lib-see or, less commonly, glib-see.

The GNU C library provides more than its name suggests. In addition to implementing the standard C library, glibc provides wrappers for system calls, threading support, and basic application facilities.

The C Compiler

In Linux, the standard C compiler is provided by the GNU Compiler Collection (gcc). Originally, gcc was GNU's version of cc, the C Compiler. Thus, gcc stood for GNU C Compiler. Over time, support was added for more and more languages. Consequently, nowadays gcc is used as the generic name for the family of GNU compilers. However, gcc is also the binary used to invoke the C compiler. In this course, when we talk of gcc, we typically mean the program gcc, unless context suggests otherwise.

用户程序 vs 库函数 vs 系统调用



GCC

The GNU Compiler Collection includes front ends for C, C++, Objective-C, Fortran, Ada, and Go, as well as libraries for these languages (libstdc++,...). GCC was originally written as the compiler for the GNU operating system. The GNU system was developed to be 100% free software, free in the sense that it respects the user's freedom.
GCC, formerly for "GNU C Compiler", has grown over times to support many languages such as C (gcc), C++ (g++), Objective-C, Objective-C++, Java (gcj), Fortran (gfortran), Ada (gnat), Go (gccgo), OpenMP, Cilk Plus, and OpenAcc. It is now referred to as "GNU Compiler Collection". The mother site for GCC is http://gcc.gnu.org/. The current version is GCC 7.3, released on 2018-01-25.

GNU Toolchain

GCC is a key component of "GNU Toolchain", for developing applications, as well as operating systems. The GNU Toolchain includes:

- GNU Compiler Collection (GCC): a compiler suit that supports many languages, such as C/C++ and Objective-C/C++
- GNU Make: an automation tool for compiling and building applications
- GNU Binutils: a suit of binary utility tools, including linker and assembler
- GNU Debugger (GDB)
- GNU Autotools: A build system including Autoconf, Autoheader, Automake and Libtool
- GNU Bison: a parser generator (similar to lex and yacc)

Platform

GCC is portable and run in many operating platforms. GCC (and GNU Toolchain) is currently available on all Unixes. They are also ported to Windows (by MinGW and Cygwin). GCC is also a cross-compiler, for producing executables on different platform.

Install gcc

- Unix/Linux/Mac GCC(GNU Toolchain) is included.
- Windows
  - Cygwin GCC: Cygwin is a Unix-like environment and command-line interface for Microsoft Windows. Cygwin is huge and includes most of the Unix tools and utilities. It also included the commonly-used Bash shell.
  - MinGW: MinGW (Minimalist GNU for Windows) is a port of the GNU Compiler Collection (GCC) and GNU Binutils for use in Windows. It also included MSYS (Minimal System), which is basically a Bourne shell (bash).

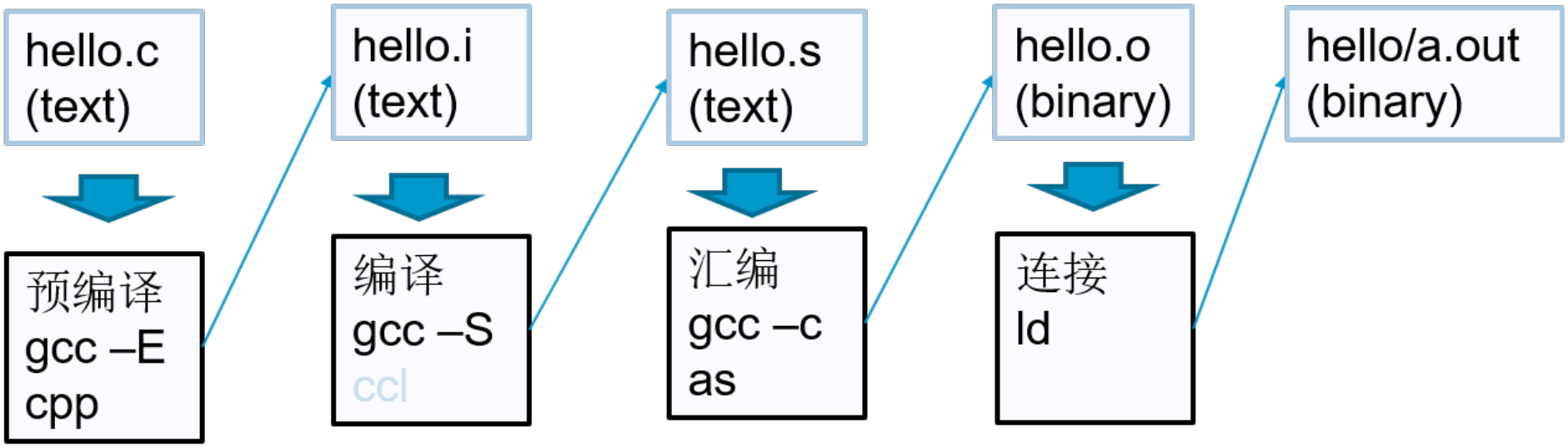
Versions and Help

We can display the version of GCC with –version/-v option, and get help with –help option:

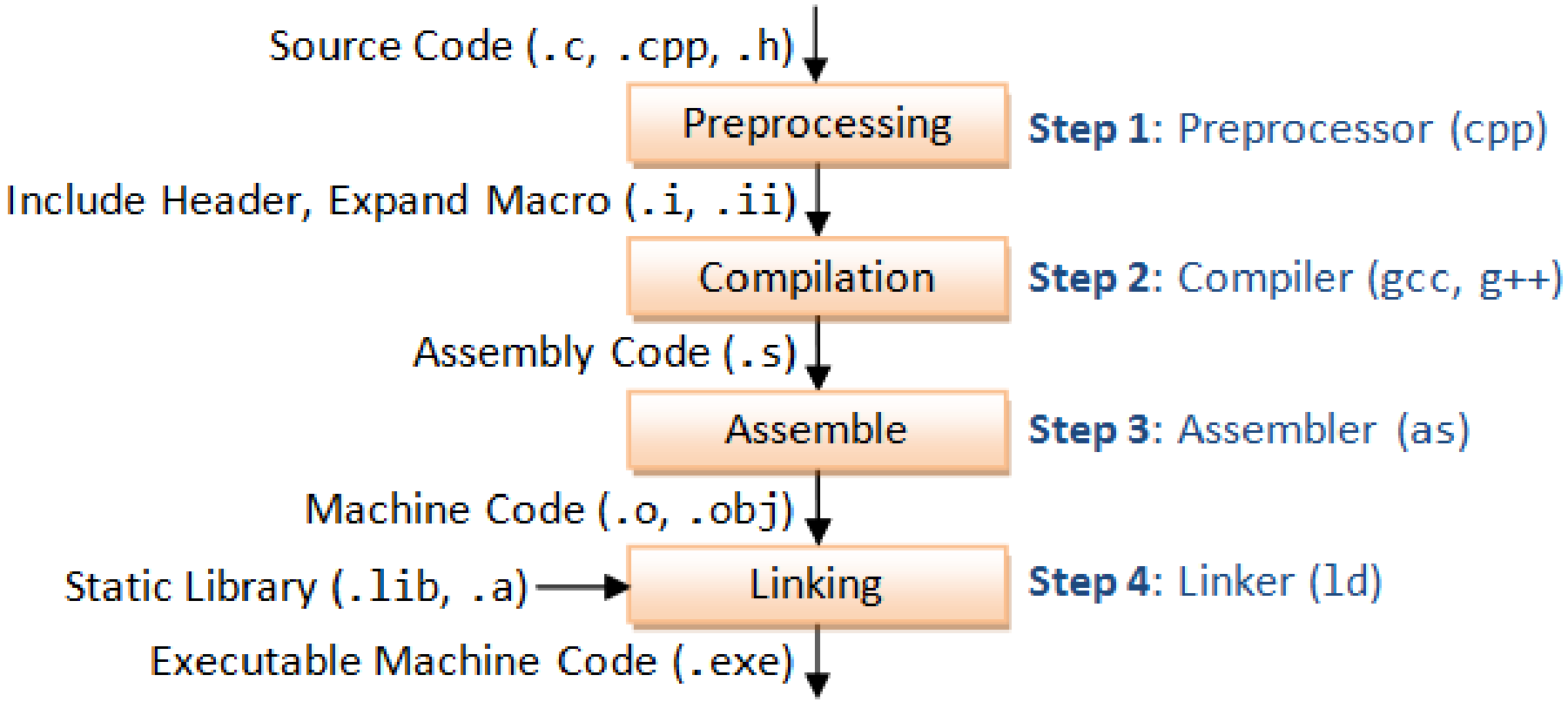
Example (show version and help of GCC)

```
$ gcc -v
$ gcc --version
$ gcc --help
```

编写 c 程序的流程


hello.c
(text)
预编译
gcc –E
cpp
hello.i
(text)
编译
gcc –S
ccl
hello.s
(text)
汇编
gcc –c
as
hello.o
(binary)
连接
ld
hello/a.out
(binary)

编写 c 程序的流程


Source Code (.c, .cpp, .h)
Preprocessing
Step 1: Preprocessor (cpp)
Include Header, Expand Macro (.i, .ii)
Compilation
Step 2: Compiler (gcc, g++)
Assembly Code (.s)
Assemble
Step 3: Assembler (as)
Machine Code (.o, .obj)
Static Library (.lib, .a)
Linking
Step 4: Linker (ld)
Executable Machine Code (.exe)

hello.c

Example

```
// hello.c
#include <stdio.h>

int main(){
        printf("Hello, C.\n");
        return 0;
}
```

Blocks of Highlighted Text

Block 1

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer lectus nisl, ultricies in feugiat rutrum, porttitor sit amet augue. Aliquam ut tortor mauris. Sed volutpat ante purus, quis accumsan dolor.

Block 2

Pellentesque sed tellus purus. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Vestibulum quis magna at risus dictum tempor eu vitae velit.

Block 3

Suspendisse tincidunt sagittis gravida. Curabitur condimentum, enim sed venenatis rutrum, ipsum neque consectetur orci, sed blandit justo nisi ac lacus.

Multiple Columns

Heading

Example (hello.c)

```
// hello.c
#include <stdio.h>

int main(){
        printf("Hello, C.\n");
        return 0;
}
```

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer lectus nisl, ultricies in feugiat rutrum, porttitor sit amet augue. Aliquam ut tortor mauris. Sed volutpat ante purus, quis accumsan dolor.

Table

Treatments	Response 1	Response 2
Treatment 1	0.0003262	0.562
Treatment 2	0.0015681	0.910
Treatment 3	0.0009271	0.296

Table: Table caption

Theorem

Theorem (Mass–energy equivalence)

$E = mc^2$

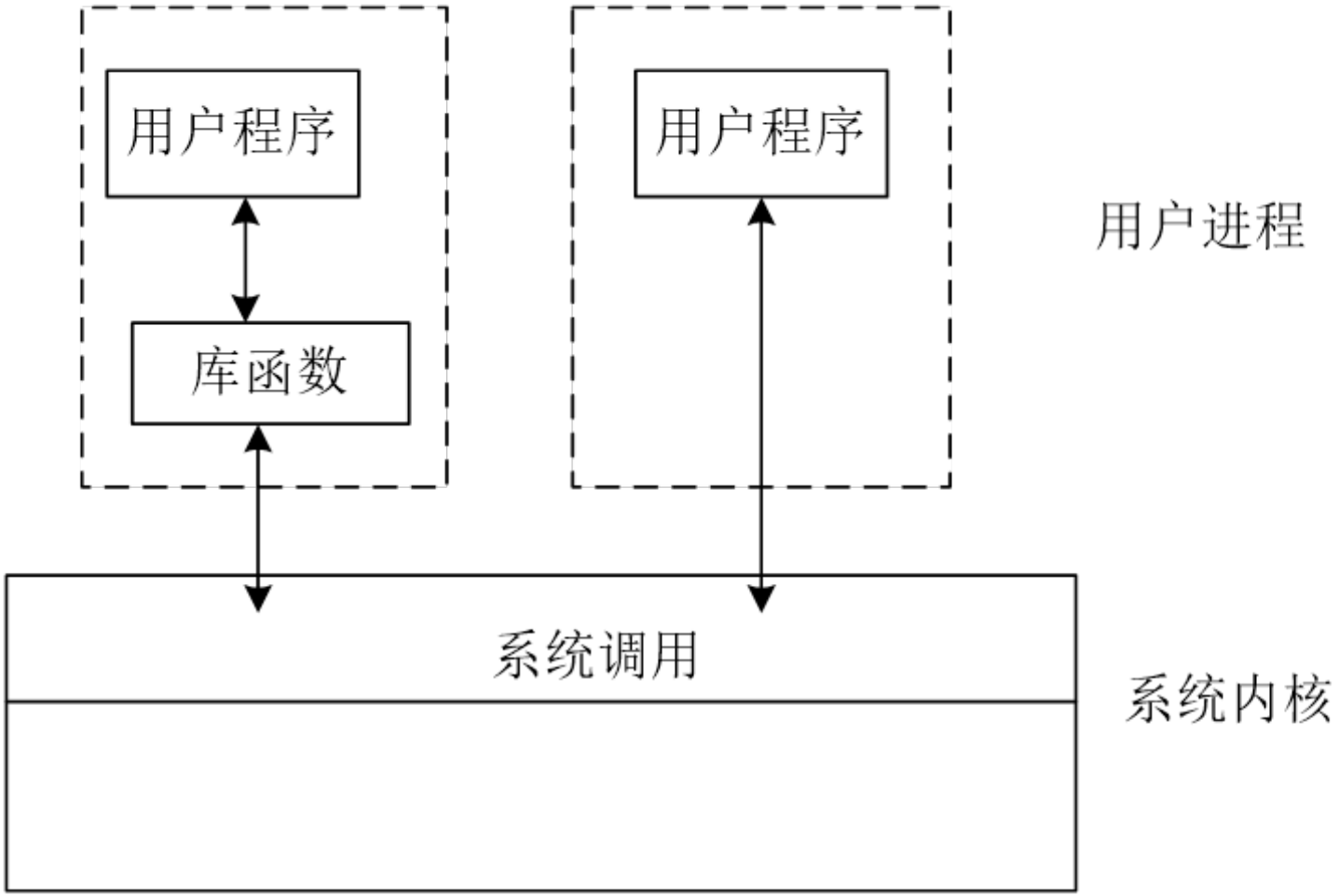
Verbatim

Example (Theorem Slide Code)

```
\begin{frame}
\frametitle{Theorem}
\begin{theorem}[Mass--energy equivalence]
$E = mc^2$
\end{theorem}
\end{frame}
```

Figure

Uncomment the code on this slide to include your own image from the same directory as the template .TeX file.



Citation

An example of the `\cite` command to cite within the presentation:

This statement requires citation [Smith, 2012].

References

John Smith (2012)

Title of the publication

Journal Name 12(3), 45 – 678.

The End