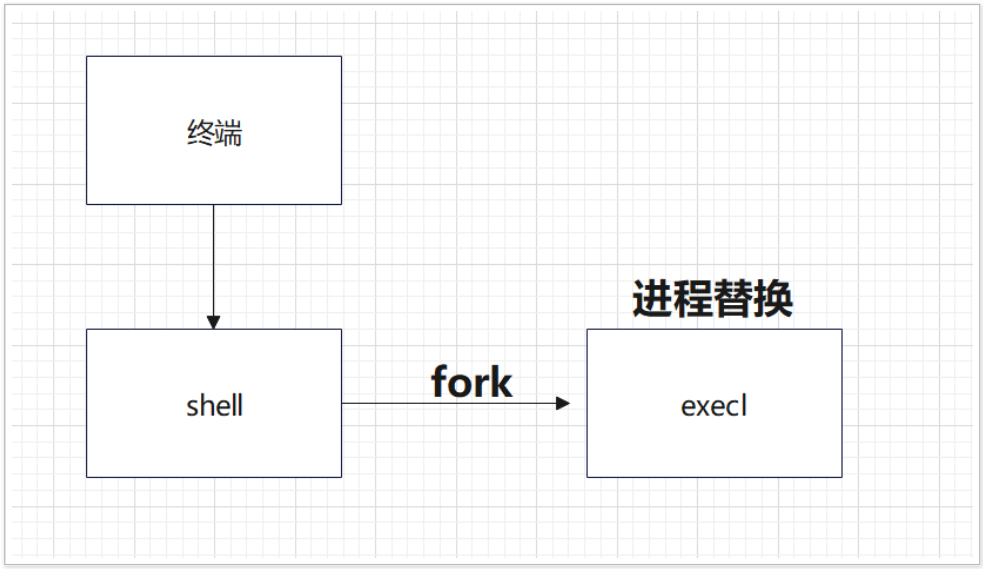
3.5 进程的替换_物联网 / 嵌入式工程师 - 慕课网

> 慕课网慕课教程 3.5 进程的替换涵盖海量编程基础技术教程，以图文图表的形式，把晦涩难懂的编程专业用语，以通俗易懂的方式呈现给用户。

- 创建一个进程后，pid 以及在内核中的信息保持 保持不变, 但进程所执行的代码进行替换
- 作用：通过一个进程启动另外一个进程
- 应用场景:
    - Linux 终端应用程序，执行命令时，通过创建一个进程后，在替换成命令的可执行程序，在执行
    -



- 在 Linux 系统中提供了一组用于进程替换的替换，共有 6 个函数

函数原型

int execl(const char *pathname, const char arg, ... / (char *) NULL */);

int execlp(const char *file, const char arg, ... / (char *) NULL */);

int execle(const char *pathname, const char arg, ... /, (char *) NULL, char *const envp[] */);

int execv(const char *pathname, char *const argv[]);

int execvp(const char *file, char *const argv[]);

int execvpe(const char *file, char *const argv[], char *const envp[]);

- 函数参数
-
    - path：可执行文件的路径名
    - file：可执行文件名，可以通过 path 环境变量指定的路径
    - arg：参数列表，以 NULL 结尾
    - argv[]：参数数组
    - envp[]：环境变量数组

- 函数返回值:
    - 成功：0
    - 失败：-1
    -
- 示例：通过 execl 函数族执行 ls -l 命令

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>


int main(void)
{
    int ret;

    ret = execl("/bin/ls","ls","-l",NULL);
    if (ret == -1){
        perror("[ERROR] execl(): ");
        exit(EXIT_FAILURE);
    }

    return 0;
}
```

练习：

实现 minishell，输入不同的名，执行不同的命令（ls -l / pwd ... 等）

- 提示:
    - 字符串分割函数 strtok
    - 需要创建子进程，execl 替换子进程
    - 父进程负责从标准输入获取用户的输入
    - 函数原型
    - `char *strtok(char *str, const char *delim);`
    - 参数:
        - str：字符串
        - delim：分割符字符串
    - 返回值
        - 成功返回子字符串的首地址，分割字符串到达'\0'位置返回 NULL
    - 注意:
        - 第一次调用需要传递字符串的首地址，后面调用传递 NULL(告诉它使用上一次的位置继续分割)
- 示例程序:

```
#include <stdio.h>
#include <string.h>

int main(void)
{

    char string[] = "ABC 123 DEF";
    char *first = NULL,*other = NULL;

    first = strtok(string," ");

    printf(" %s ",first);

    while((other = strtok(NULL,"")) != NULL)
    {
        printf(" %s \n",other);
    }
```

```
    return 0;
}
```

全文完