原创: 百问网论坛wuweidong

整理: 百问网(也称”百问科技”)运营团队

移植uboot的目的是启动内核，启动内核的目的是运行应用程序，从内核的启动流程中可以知道内核启动的第一个应用程序是busybox里的/sbin/init进程。

但是我们的最终目的不是启动init进程，而是运行客户的程序！
那么init进程是如何选择性的运行客户的程序呢？我们猜测init进程肯定需要：
（1） 读取一个配置文件
（2） 解析该配置文件
（3） 根据配置文件执行客户的程序

下面我们来阅读busybox中init程序的源码，在init.c中的init\_main（）中：
**1.首先是设置信号**

    signal(SIGHUP, exec\_signal);
    signal(SIGQUIT, exec\_signal);
    signal(SIGUSR1, shutdown\_signal);
    signal(SIGUSR2, shutdown\_signal);
    signal(SIGINT, ctrlaltdel\_signal);
    signal(SIGTERM, shutdown\_signal);
    signal(SIGCONT, cont\_handler);
    signal(SIGSTOP, stop\_handler);
    signal(SIGTSTP, stop\_handler);

**2.初始化串口/dev/console**

console\_init();

**3.解析配置文件**

if (argc 1
     (!strcmp(argv[1], single) || !strcmp(argv[1], s) || LONE\_CHAR(argv[1], '1'))
    )

{
    } else

{
        parse\_inittab();
    }

内核启动/sbin/init没有传任何参数，所以进入parse\_inittab（）函数，
我们进入到该函数：

file = fopen(INITTAB, r);
#define INITTAB      /etc/inittab

由此可以知道init进程读取的配置文件就是/etc/inittab, busybox中的inittab文件中规定了/etc/inittab内容的格式如下：
<id>:<runlevel>:<action>:<process>
Id:id会加上一个/dev前缀作为一个控制终端（stdin，stdout，stderr）
Runlevel：忽略
Action：执行的时机, 包括sysinit，wait，once， respawn，askfirst等

表17.6 /etc/inittab文件中<action>字段的意义

|  |  |  |
| --- | --- | --- |
| action名称 | 执行条件 | 说明 |
| sysinit | 系统启动后最先执行 | 只执行一次，init进程等待它结束才继续执行其他动作 |
| wait | 系统执行完sysinit进程后 | 只执行一次，init进程等待它结束才继续执行其他动作 |
| once | 系统执行完wait进程后 | 只执行一次，init进程不等待它结束 |
| respawn | 启动完once进程后 | init进程监测发现子进程退出时，重新启动它 |
| askfirst | 启动完respawn进程后 | 与respawn类似，不过init进程先输出“Please press Enter to activate this console.”，等用户输入回车键之后才启动子进程。 |
| shutdown | 当系统关机时 | 即重启、关闭系统命令时 |
| restart | Busybox中配置了CONFIG\_FEATURE\_USE\_INITTAB，并且init进程接收到SIGHUP信号时 | 先重新读取、解析/etc/inittab文件，再执行restart程序 |
| ctrlaltdel | 按下Ctrl+Alt+Delete组合键时 |  |

Process：要执行的应用程序或者脚本
继续分析parse\_inittab（）：

    if (file == NULL)
{
    new\_init\_action(CTRLALTDEL, reboot);

 new\_init\_action(SHUTDOWN, umount -a -r);
    new\_init\_action(RESTART, init);
    new\_init\_action(ASKFIRST,bb\_default\_login\_shell);

new\_init\_action(ASKFIRST,bb\_default\_login\_shell, VC\_2);

new\_init\_action(ASKFIRST,bb\_default\_login\_shell, VC\_3);

new\_init\_action(ASKFIRST,bb\_default\_login\_shell, VC\_4);
    new\_init\_action(SYSINIT, INIT\_SCRIPT);
    return;
#if ENABLE\_FEATURE\_USE\_INITTAB
}

如果配置文件/etc/inittab不存在则执行if语句，即如果没有/etc/inittab的话init进程会直接调用new\_init\_action来构造默认配置项，根据if语句里的内容，我们可以反推出等效的/etc/inittab的内容如下：
：：CTRLALTDEL：reboot
：：SHUTDOWN：umount -a –r
：：RESTART：init
：：ASKFIRST：-/bin/ah
tty2:: ASKFIRST:-/bin/sh
tty3:: ASKFIRST:-/bin/sh
tty4:: ASKFIRST:-/bin/sh
::SYSINIT:/etc/init.d/rcS

继续分析parse\_inittab（）：
具体代码就不贴了，后面就是对配置文件/etc/inittab里的内容进行解析，如#则视作注释等等，最后调用new\_init\_action(a->action, command, id);将/etc/inittab里的每一条配置项做成一个init\_action结构体并添加到具有相同执行时机的init\_action\_list中去。说到底，解析/etc/inittab配置文件就是为了把各配置项添加到对应的init\_action\_list中去。
至此parse\_inittab（）就结束了，我们继续看函数init\_main（）

**4.开始运行parse\_inittab()帮我们添加到init\_action\_list中的程序或脚本：**

    run\_actions(SYSINIT);
    run\_actions(WAIT);
    run\_actions(ONCE);
while (1)
{
run\_actions(RESPAWN);
run\_actions(ASKFIRST);
}

run\_actions为运行一类程序或脚本，这里的一类就是按照执行时机来分类的。由上面这段代码我们就可以看出执行时机的优先级了：SYSINIT> WAIT> ONCE> RESPAWN> ASKFIRST，具体的可以继续分析源码,

在/etc/inittab文件文件的控制下，init进程的行为总结如下：

① 在系统启动前期，init进程首先启动<action>为sysinit、wait、once的3类子进程。

② 在系统正常运行期间，init进程首先启动<action>为respawn、askfirst的两类子进程，并监视它们，发现某个子进程退出时重新启动它。

③ 在系统退出时，执行<action>为shutdown、restart、ctrlaltdel的3类子进程(之一或全部)。

如果根文件系统中没有/etc/inittab文件，Busybox init程序将使用如下默认的inittab条目：

::sysinit:/etc/init.d/rcS

::askfirst:/bin/sh

tty2::askfirst:/bin/sh

tty3::askfirst:/bin/sh

tty4::askfirst:/bin/sh

::ctrlaltdel:/sbin/reboot

::shutdown:/sbin/swapoff -a

::shutdown:/bin/umount -a -r

::restart:/sbin/init

（2）/etc/inittab实例。

仿照Busybox的examples/inittab文件，创建一个inittab文件，内容如下：

# /etc/inittab

# 这是init进程启动的第一个子进程，它是一个脚本，可以在里面指定用户想执行的操作

# 比如挂接其他文件系统、配置网络等

::sysinit:/etc/init.d/rcS

# 启动shell，以/dev/ttySAC0作为控制台

ttySAC0::askfirst:-/bin/sh

# 按下Ctrl+Alt+Delete之后执行的程序，不过在串口控制台中无法输入Ctrl+Alt+Delete组合键

::ctrlaltdel:/sbin/reboot

# 重启、关机前执行的程序

::shutdown:/bin/umount -a -r

参考教程:韦东山1期视频