Console Drivers

The linux kernel has 2 general types of console drivers. The first type is assigned by the kernel to all the virtual consoles during the boot process. This type will be called 'system driver', and only one system driver is allowed to exist. The system driver is persistent and it can never be unloaded, though it may become inactive.

The second type has to be explicitly loaded and unloaded. This will be called 'modular driver' by this document. Multiple modular drivers can coexist at any time with each driver sharing the console with other drivers including the system driver. However, modular drivers cannot take over the console that is currently occupied by another modular driver. (Exception: Drivers that call take_over_console() will succeed in the takeover regardless of the type of driver occupying the consoles.) They can only take over the console that is occupied by the system driver. In the same token, if the modular driver is released by the console, the system driver will take over.

Modular drivers, from the programmer's point of view, has to call:

```
take_over_console() - load and bind driver to console layer
give_up_console() - unbind and unload driver
```

In newer kernels, the following are also available:

```
register_con_driver()
unregister_con_driver()
```

If sysfs is enabled, the contents of /sys/class/vtconsole can be examined. This shows the console backends currently registered by the system which are named vtcon<n> where <n> is an integer from 0 to 15. Thus:

```
ls /sys/class/vtconsole
.  ..  vtcon0  vtcon1
```

Each directory in /sys/class/vtconsole has 3 files:

```
ls /sys/class/vtconsole/vtcon0
.  ..  bind  name  uevent
```

What do these files signify?

1. bind - this is a read/write file. It shows the status of the driver if read, or acts to bind or unbind the driver to the virtual consoles when written to. The possible values are:

   0 - means the driver is not bound and if echo'ed, commands the driver to unbind

   1 - means the driver is bound and if echo'ed, commands the driver to bind

2. name - read-only file. Shows the name of the driver in this format:

   ```
   cat /sys/class/vtconsole/vtcon0/name
   ```

(S) VGA+

'(S)' stands for a (S)ystem driver, ie, it cannot be directly commanded to bind or unbind

'VGA+' is the name of the driver

cat /sys/class/vtconsole/vtcon1/name
(M) frame buffer device

In this case, '(M)' stands for a (M)odular driver, one that can be directly commanded to bind or unbind.

3. uevent - ignore this file

When unbinding, the modular driver is detached first, and then the system driver takes over the consoles vacated by the driver. Binding, on the other hand, will bind the driver to the consoles that are currently occupied by a system driver.

NOTE1: Binding and unbinding must be selected in Kconfig. It's under:

Device Drivers -> Character devices -> Support for binding and unbinding console drivers

NOTE2: If any of the virtual consoles are in KD_GRAPHICS mode, then binding or unbinding will not succeed. An example of an application that sets the console to KD_GRAPHICS is X.

How useful is this feature? This is very useful for console driver developers. By unbinding the driver from the console layer, one can unload the driver, make changes, recompile, reload and rebind the driver without any need for rebooting the kernel. For regular users who may want to switch from framebuffer console to VGA console and vice versa, this feature also makes this possible. (NOTE NOTE NOTE: Please read fbcon.txt under Documentation/fb for more details).

Notes for developers:
=====================

take_over_console() is now broken up into:

register_con_driver()
bind_con_driver() - private function

give_up_console() is a wrapper to unregister_con_driver(), and a driver must be fully unbound for this call to succeed. con_is_bound() will check if the driver is bound or not.

Guidelines for console driver writers:
======================================

In order for binding to and unbinding from the console to properly work, console drivers must follow these guidelines:

1. All drivers, except system drivers, must call either register_con_driver()

or take_over_console(). register_con_driver() will just add the driver to the console's internal list. It won't take over the console. take_over_console(), as it name implies, will also take over (or bind to) the console.

2. All resources allocated during con->con_init() must be released in con->con_deinit().

3. All resources allocated in con->con_startup() must be released when the driver, which was previously bound, becomes unbound. The console layer does not have a complementary call to con->con_startup() so it's up to the driver to check when it's legal to release these resources. Calling con_is_bound() in con->con_deinit() will help. If the call returned false(), then it's safe to release the resources. This balance has to be ensured because con->con_startup() can be called again when a request to rebind the driver to the console arrives.

4. Upon exit of the driver, ensure that the driver is totally unbound. If the condition is satisfied, then the driver must call unregister_con_driver() or give_up_console().

5. unregister_con_driver() can also be called on conditions which make it impossible for the driver to service console requests. This can happen with the framebuffer console that suddenly lost all of its drivers.

The current crop of console drivers should still work correctly, but binding and unbinding them may cause problems. With minimal fixes, these drivers can be made to work correctly.

==========

Antonino Daplas <adaplas@pol.net>