The Framebuffer Console
=======================

        The framebuffer console (fbcon), as its name implies, is a text
console running on top of the framebuffer device. It has the functionality of
any standard text console driver, such as the VGA console, with the added
features that can be attributed to the graphical nature of the framebuffer.

        In the x86 architecture, the framebuffer console is optional, and
some even treat it as a toy. For other architectures, it is the only available
display device, text or graphical.

        What are the features of fbcon? The framebuffer console supports
high resolutions, varying font types, display rotation, primitive multihead,
etc. Theoretically, multi-colored fonts, blending, aliasing, and any feature
made available by the underlying graphics card are also possible.

A. Configuration

        The framebuffer console can be enabled by using your favorite kernel
configuration tool. It is under Device Drivers->Graphics Support->Support for
framebuffer devices->Framebuffer Console Support. Select 'y' to compile
support statically, or 'm' for module support. The module will be fbcon.

        In order for fbcon to activate, at least one framebuffer driver is
required, so choose from any of the numerous drivers available. For x86
systems, they almost universally have VGA cards, so vga16fb and vesafb will
always be available. However, using a chipset-specific driver will give you
more speed and features, such as the ability to change the video mode
dynamically.

        To display the penguin logo, choose any logo available in Logo
Configuration->Boot up logo.

        Also, you will need to select at least one compiled-in fonts, but if
you don't do anything, the kernel configuration tool will select one for you,
usually an 8x16 font.

GOTCHA: A common bug report is enabling the framebuffer without enabling the
framebuffer console. Depending on the driver, you may get a blanked or
garbled display, but the system still boots to completion. If you are
fortunate to have a driver that does not alter the graphics chip, then you
will still get a VGA console.

B. Loading

Possible scenarios:

1. Driver and fbcon are compiled statically

        Usually, fbcon will automatically take over your console. The notable
        exception is vesafb. It needs to be explicitly activated with the
        vga= boot option parameter.

2. Driver is compiled statically, fbcon is compiled as a module

Depending on the driver, you either get a standard console, or a
garbled display, as mentioned above. To get a framebuffer console,
do a 'modprobe fbcon'.

3. Driver is compiled as a module, fbcon is compiled statically

You get your standard console. Once the driver is loaded with
'modprobe xxxfb', fbcon automatically takes over the console with
the possible exception of using the fbcon=map:n option. See below.

4. Driver and fbcon are compiled as a module.

You can load them in any order. Once both are loaded, fbcon will take
over the console.

C. Boot options

The framebuffer console has several, largely unknown, boot options
that can change its behavior.

1. fbcon=font:<name>

Select the initial font to use. The value 'name' can be any of the
compiled-in fonts: VGA8x16, 7x14, 10x18, VGA8x8, MINI4x6, RomanLarge,
SUN8x16, SUN12x22, ProFont6x11, Acorn8x8, PEARL8x8.

Note, not all drivers can handle font with widths not divisible by 8,
such as vga16fb.

2. fbcon=scrollback:<value>[k]

The scrollback buffer is memory that is used to preserve display
contents that has already scrolled past your view. This is accessed
by using the Shift-PageUp key combination. The value 'value' is any
integer. It defaults to 32KB. The 'k' suffix is optional, and will
multiply the 'value' by 1024.

3. fbcon=map:<0123>

This is an interesting option. It tells which driver gets mapped to
which console. The value '0123' is a sequence that gets repeated until
the total length is 64 which is the number of consoles available. In
the above example, it is expanded to 012301230123... and the mapping
will be:

        tty │ 1 2 3 4 5 6 7 8 9 ...
        fb │ 0 1 2 3 0 1 2 3 0 ...

('cat /proc/fb' should tell you what the fb numbers are)

One side effect that may be useful is using a map value that exceeds
the number of loaded fb drivers. For example, if only one driver is
available, fb0, adding fbcon=map:1 tells fbcon not to take over the
console.

Later on, when you want to map the console the to the framebuffer

device, you can use the con2fbmap utility.

4. fbcon=vc:<n1>-<n2>

This option tells fbcon to take over only a range of consoles as
specified by the values 'n1' and 'n2'. The rest of the consoles
outside the given range will still be controlled by the standard
console driver.

NOTE: For x86 machines, the standard console is the VGA console which
is typically located on the same video card. Thus, the consoles that
are controlled by the VGA console will be garbled.

4. fbcon=rotate:<n>

This option changes the orientation angle of the console display. The
value 'n' accepts the following:

        0 - normal orientation (0 degree)
        1 - clockwise orientation (90 degrees)
        2 - upside down orientation (180 degrees)
        3 - counterclockwise orientation (270 degrees)

The angle can be changed anytime afterwards by 'echoing' the same
numbers to any one of the 2 attributes found in
 /sys/class/graphics/fbcon

        rotate   - rotate the display of the active console
        rotate_all - rotate the display of all consoles

Console rotation will only become available if Console Rotation
Support is compiled in your kernel.

NOTE: This is purely console rotation. Any other applications that
use the framebuffer will remain at their 'normal' orientation.
Actually, the underlying fb driver is totally ignorant of console
rotation.

C. Attaching, Detaching and Unloading

Before going on on how to attach, detach and unload the framebuffer console, an
illustration of the dependencies may help.

The console layer, as with most subsystems, needs a driver that interfaces with
the hardware. Thus, in a VGA console:

console ---> VGA driver ---> hardware.

Assuming the VGA driver can be unloaded, one must first unbind the VGA driver
from the console layer before unloading the driver. The VGA driver cannot be
unloaded if it is still bound to the console layer. (See
Documentation/console/console.txt for more information).

This is more complicated in the case of the framebuffer console (fbcon),
because fbcon is an intermediate layer between the console and the drivers:

```
                              fbcon.txt
console ---> fbcon ---> fbdev drivers ---> hardware
```

The fbdev drivers cannot be unloaded if it's bound to fbcon, and fbcon cannot be unloaded if it's bound to the console layer.

So to unload the fbdev drivers, one must first unbind fbcon from the console, then unbind the fbdev drivers from fbcon. Fortunately, unbinding fbcon from the console layer will automatically unbind framebuffer drivers from fbcon. Thus, there is no need to explicitly unbind the fbdev drivers from fbcon.

So, how do we unbind fbcon from the console? Part of the answer is in Documentation/console/console.txt. To summarize:

Echo a value to the bind file that represents the framebuffer console driver. So assuming vtcon1 represents fbcon, then:

```
echo 1 > sys/class/vtconsole/vtcon1/bind - attach framebuffer console to
                                           console layer
echo 0 > sys/class/vtconsole/vtcon1/bind - detach framebuffer console from
                                           console layer
```

If fbcon is detached from the console layer, your boot console driver (which is usually VGA text mode) will take over. A few drivers (rivafb and i810fb) will restore VGA text mode for you. With the rest, before detaching fbcon, you must take a few additional steps to make sure that your VGA text mode is restored properly. The following is one of the several methods that you can do:

1. Download or install vbetool. This utility is included with most distributions nowadays, and is usually part of the suspend/resume tool.

2. In your kernel configuration, ensure that CONFIG_FRAMEBUFFER_CONSOLE is set to 'y' or 'm'. Enable one or more of your favorite framebuffer drivers.

3. Boot into text mode and as root run:

        vbetool vbestate save > <vga state file>

   The above command saves the register contents of your graphics hardware to <vga state file>. You need to do this step only once as the state file can be reused.

4. If fbcon is compiled as a module, load fbcon by doing:

        modprobe fbcon

5. Now to detach fbcon:

        vbetool vbestate restore < <vga state file> && \
        echo 0 > /sys/class/vtconsole/vtcon1/bind

6. That's it, you're back to VGA mode. And if you compiled fbcon as a module, you can unload it by 'rmmod fbcon'

7. To reattach fbcon:

```
echo 1 > /sys/class/vtconsole/vtcon1/bind
```

8. Once fbcon is unbound, all drivers registered to the system will also
become unbound. This means that fbcon and individual framebuffer drivers
can be unloaded or reloaded at will. Reloading the drivers or fbcon will
automatically bind the console, fbcon and the drivers together. Unloading
all the drivers without unloading fbcon will make it impossible for the
console to bind fbcon.

Notes for vesafb users:
=======================

Unfortunately, if your bootline includes a vga=xxx parameter that sets the
hardware in graphics mode, such as when loading vesafb, vgacon will not load.
Instead, vgacon will replace the default boot console with dummycon, and you
won't get any display after detaching fbcon. Your machine is still alive, so
you can reattach vesafb. However, to reattach vesafb, you need to do one of
the following:

Variation 1:

    a. Before detaching fbcon, do

       vbetool vbemode save > <vesa state file> # do once for each vesafb mode,
                                                 # the file can be reused

    b. Detach fbcon as in step 5.

    c. Attach fbcon

       vbetool vbestate restore < <vesa state file> && \
       echo 1 > /sys/class/vtconsole/vtcon1/bind

Variation 2:

    a. Before detaching fbcon, do:
       echo <ID> > /sys/class/tty/console/bind

       vbetool vbemode get

    b. Take note of the mode number

    b. Detach fbcon as in step 5.

    c. Attach fbcon:

       vbetool vbemode set <mode number> && \
       echo 1 > /sys/class/vtconsole/vtcon1/bind

Samples:
========

Here are 2 sample bash scripts that you can use to bind or unbind the
framebuffer console driver if you are in an X86 box:

```bash
#!/bin/bash
# Unbind fbcon

# Change this to where your actual vgastate file is located
# Or Use VGASTATE=$1 to indicate the state file at runtime
VGASTATE=/tmp/vgastate

# path to vbetool
VBETOOL=/usr/local/bin


for (( i = 0; i < 16; i++))
do
   if test -x /sys/class/vtconsole/vtcon$i; then
      if [ `cat /sys/class/vtconsole/vtcon$i/name | grep -c "frame buffer"` \
           = 1 ]; then
            if test -x $VBETOOL/vbetool; then
               echo Unbinding vtcon$i
               $VBETOOL/vbetool vbestate restore < $VGASTATE
               echo 0 > /sys/class/vtconsole/vtcon$i/bind
            fi
      fi
   fi
done
```

```bash
#!/bin/bash
# Bind fbcon

for (( i = 0; i < 16; i++))
do
   if test -x /sys/class/vtconsole/vtcon$i; then
      if [ `cat /sys/class/vtconsole/vtcon$i/name | grep -c "frame buffer"` \
           = 1 ]; then
          echo Unbinding vtcon$i
          echo 1 > /sys/class/vtconsole/vtcon$i/bind
      fi
   fi
done
```

--
Antonino Daplas <adaplas@pol.net>