BananaPro/Pi:GPU

From BananaPro/Pi

Overview

The Mali series of Graphics Processing Units (GPUs) are semiconductor intellectual property cores produced by ARM Holdings for licensing in various ASIC (Application-specific integrated circuit) designs by ARM partners. The core is mainly developed by ARM Norway, at the former Falanx company site. Like other embedded IP cores for 3D support, the Mali GPU does not feature display controllers driving monitors (such as the combination often found in common video cards). Instead it is a pure 3D engine that renders graphics into memory and hands the rendered image over to another core that handles the display.

Implementatioan

Step 1:Install the package
Install the package below:

```
sudo apt-get update
sudo apt-get install git build-essential make gcc autoconf libtool debhelper \
dh-autoreconf pkg-config automake xutils-dev libx11-dev libxext-dev libdrm-dev \
x11proto-dri2-dev libxfixes-dev xorg-dev libltdl-dev mesa-utils
```

Step 2:Get git repositories
Download the git repositories below:

```
mkdir ~/hw_accleration
cd ~/hw_accleration
git clone https://github.com/robclark/libdri2.git
git clone https://github.com/linux-sunxi/libump.git
git clone https://github.com/linux-sunxi/sunxi-mali.git
git clone https://github.com/ssvb/xf86-video-fbturbo.git
```

You can replace the **hw_acclertion** to your own directory.

Step 3:Check driver modules
Before we build the four packages, we'd better ensure the driver modules "mali", "mali_drm", "ump" has been loaded. We can add the below lines into ***/etc/modules***

```
ump
mali
```

step 4:Install libdri1
Libdri2 is needed by sunxi-mali build process when we use X11 version sunxi-mali (if use framebuffer version, we do not need libdri2).

```
cd ~/hw_accleration
cd libdri2
./autogen.sh
./configure --prefix=/usr
make
sudo make install
sudo ldconfig
```

step 5:Install libump
Build libump,libump is the unified memory provider user-space library.There are two ways to install libump. You can directly install it like below, this will install all the ump libraries into the system :

```
cd ~/hw_accleration
cd libump
autoreconf -i
./configure
make
make install
sudo ldconfig
```

If you do not want to install the libump directly and also want to install it in other machine conveniently, you can use deb package method:

```
cd ~/hw_accleration
cd libump
dpkg-buildpackage -b
sudo dpkg -i ../libump_3.0-0sunxi1_armhf.deb
sudo dpkg -i ../libump-dev_3.0-0sunxi1_armhf.deb
sudo ldconfig
```

step 6:Build sunxi-mali
Because the gle.h and gl2ext.h in sunxi-mali repositories is too old, we need replace them with new ones.The files path is located at:
/sunxi-mali/include/GLES2/ gl2ext.h
/sunxi-mali/include/GLES2/ gl2ext.h
Compile sumxi-mali:

```
cd ~/hw_accleration
cd sunxi-mali
git submodule init
git submodule update
```

For X11 version:

```
make config
sudo make install
```

For framebuffer version:

```
make config ABI=armhf VERSION=r3p0 EGL_TYPE=framebuffer
sudo make install
```

step 7:Install fbturbo

If we want to a GLES capable Xsevers, the we need to install the fbturbo driver.

```
cd ~/hw_accleration
cd xf86-video-fbturbo
autoreconf -v -i
./configure --prefix=/usr
make
sudo make install
```

Make a xorg.conf file to make the Xservers work with fbturbo, make a new file: ***/etc/X11/xorg.conf.d/99-fbturbo.conf***:

```
Section "Screen"
        Identifier        "My Screen"
        Device          "Allwinner A10/A13 FBDEV"
        Monitor         "My Monitor"
EndSection

Section "Device"
        Identifier        "Allwinner A10/A13 FBDEV"
        Driver          "fbturbo"
        Option          "fbdev" "/dev/fb0"
        Option          "SwapbuffersWait" "true"
        Option          "AccelMethod" "G2D"
EndSection

Section "Monitor"
        Identifier        "My Monitor"
        Option          "DPMS" "false"
EndSection
```

Step 8:Edit 50-mali.rules

The default permissions of /dev/ump and /dev/mali make these unusable for normal users. Add a file to ***/etc/udev/rules.d/***, perhaps called ***50-mali.rules***, with the following content:

```
KERNEL=="mali", MODE="0660", GROUP="video"
KERNEL=="ump", MODE="0660", GROUP="video"
KERNEL=="disp", MODE="0660", GROUP="video"
KERNEL=="g2d", MODE="0660", GROUP="video"
KERNEL=="fb*", MODE="0660", GROUP="video"
KERNEL=="cedar_dev", MODE="0660", GROUP="video"
```

Add then add your user account into the video group:

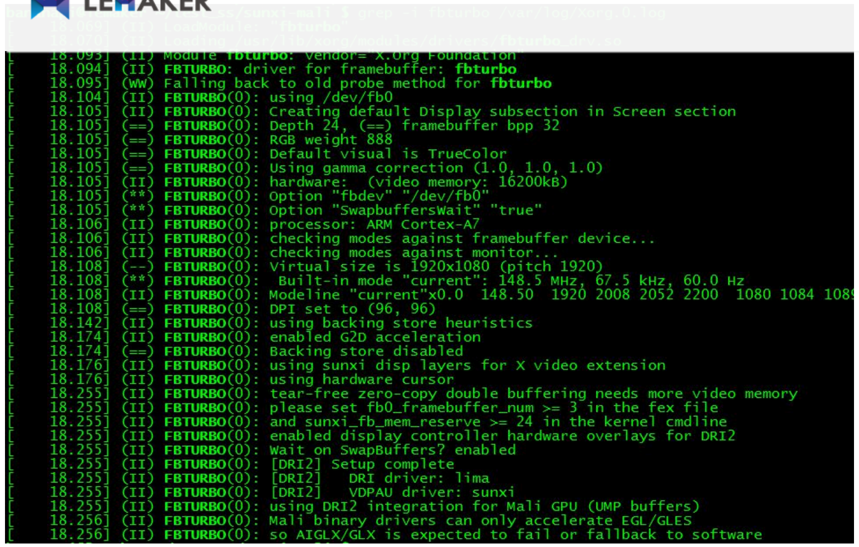
```
sudo usermod -aG video $USER
```

Step 9:Test

After all the steps in last section, you need **reboot the system**.
And then first to check the fbturbo loaded right:

```
grep -i fbturbo /var/log/Xorg.0.log
```

And the output should be like this:



```
banana@lemaker ~/test_ss/sunxi-mali $ grep -i fbturbo /var/log/Xorg.0.log
[    18.069] (II) LoadModule: "fbturbo"
[    18.070] (II) Loading /usr/lib/xorg/modules/drivers/fbturbo_drv.so
[    18.093] (II) Module fbturbo: vendor="X.Org Foundation"
[    18.094] (II) FBTURBO: driver for framebuffer: fbturbo
[    18.095] (WW) Falling back to old probe method for fbturbo
[    18.104] (II) FBTURBO(0): using /dev/fb0
[    18.105] (II) FBTURBO(0): Creating default Display subsection in Screen section
[    18.105] (==) FBTURBO(0): Depth 24, (==) framebuffer bpp 32
[    18.105] (==) FBTURBO(0): RGB weight 888
[    18.105] (==) FBTURBO(0): Default visual is TrueColor
[    18.105] (==) FBTURBO(0): Using gamma correction (1.0, 1.0, 1.0)
[    18.105] (II) FBTURBO(0): hardware:  (video memory: 16200kB)
[    18.105] (**) FBTURBO(0): Option "fbdev" "/dev/fb0"
[    18.105] (**) FBTURBO(0): Option "SwapbuffersWait" "true"
[    18.106] (II) FBTURBO(0): processor: ARM Cortex-A7
[    18.106] (II) FBTURBO(0): checking modes against framebuffer device...
[    18.106] (II) FBTURBO(0): checking modes against monitor...
[    18.108] (--) FBTURBO(0): Virtual size is 1920x1080 (pitch 1920)
[    18.108] (**) FBTURBO(0):  Built-in mode "current": 148.5 MHz, 67.5 kHz, 60.0 Hz
[    18.108] (II) FBTURBO(0): Modeline "current"x0.0  148.50  1920 2008 2052 2200  1080 1084 108
[    18.108] (==) FBTURBO(0): DPI set to (96, 96)
[    18.142] (II) FBTURBO(0): using backing store heuristics
[    18.174] (II) FBTURBO(0): enabled G2D acceleration
[    18.174] (==) FBTURBO(0): Backing store disabled
[    18.176] (II) FBTURBO(0): using sunxi disp layers for X video extension
[    18.176] (II) FBTURBO(0): using hardware cursor
[    18.255] (II) FBTURBO(0): tear-free zero-copy double buffering needs more video memory
[    18.255] (II) FBTURBO(0): please set fb0_framebuffer_num >= 3 in the fex file
[    18.255] (II) FBTURBO(0): and sunxi_fb_mem_reserve >= 24 in the kernel cmdline
[    18.255] (II) FBTURBO(0): enabled display controller hardware overlays for DRI2
[    18.255] (II) FBTURBO(0): Wait on SwapBuffers? enabled
[    18.255] (II) FBTURBO(0): [DRI2] Setup complete
[    18.255] (II) FBTURBO(0): [DRI2]   DRI driver: lima
[    18.255] (II) FBTURBO(0): [DRI2]   VDPAU driver: sunxi
[    18.255] (II) FBTURBO(0): using DRI2 integration for Mali GPU (UMP buffers)
[    18.256] (II) FBTURBO(0): Mali binary drivers can only accelerate EGL/GLES
[    18.256] (II) FBTURBO(0): so AIGLX/GLX is expected to fail or fallback to software
```

Secondly, test the sunxi-mali:

```
cd ~/hw_accleration
cd sunxi-mali
```

For X11:

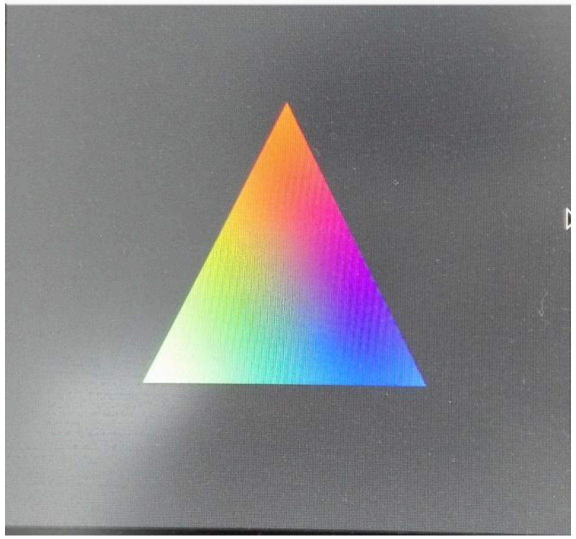
```
cd test
cc -Wall -o test test.c -lEGL -lGLESv2 -lX11
./test
```

For framebuffer:

```
make test
./test/test
```

You will see.And you should be able to see a smoothed triangle, either written out to the top left corner of the framebuffer, or in an X window. The console will tell you which renderer is being used:

```
...
GL Vendor: "ARM"
GL Renderer: "Mali-400 MP"
GL Version: "OpenGL ES 2.0"
...
```



See Also

http://linux-sunxi.org/Mali_binary_driver
http://linux-sunxi.org/Xorg#fbturbo_driver
https://github.com/ssvb/xf86-video-fbturbo