QEMU with hardware graphics acceleration

Introduction

QEMU can be compiled with additional support for Virgil, which is a solution that allows it to access the host machine's graphics adapter OpenGL capabilities. This way, the machine emulated by QEMU essentially uses the actual hardware graphics card for accelerating graphics rendering.

Sadly, QEMU packages shipped inside distributions don't include support for this : it is thus necessary to compile a custom build which will support this functionality. This page shows how to create such a build and run the GDP image while using a custom QEMU.

This guide was written and tested on Ubuntu 17.04 and qemu 2.9.1. Other distributions might also provide the necessary packages, but their names and/or versions may differ : your mileage may vary.

UPDATE: Tested with qemu 2.10.1 with good results. That test was done on Fedora 25, GNOME 3.22.2

Preparing

You need the basic development tools for your platform, as well as kernel headers and headers for the libraries that QEMU depends on. This command should install everything that's needed :

```
sudo apt install build-essential libepoxy-dev libdrm-dev libgbm-dev libx11-dev libvirglrenderer-dev libpulse-dev libsdl2-dev
```

On Fedora, this translates to something like :

```
sudo dnf install  libepoxy-devel libdrm-devel mesa-libgbm-devel virglrenderer-devel pulseaudio-libs-devel SDL2-devel
```

Compiling

Start out by downloading and extracting the QEMU source tarball.

```
wget http://download.qemu-project.org/qemu-2.12.0.tar.xz
tar xf qemu-2.12.0.tar.xz
cd qemu-2.12.0
```

Now issue the configure command in order to generate build instructions for make.

```
./configure --enable-sdl --with-sdlabi=2.0 --enable-opengl --enable-virglrenderer --enable-system --enable-modules --audio-drv-list=pa --target-list=x86_64-softmmu --enable-kvm
```

If you installed all the development packages for dependent libraries properly in the previous step, this should end successfully with a long report telling you which features of QEMU will be enabled in the build. Verify that the following lines appear :

```
SDL support       yes (2.x.y) # x.y depends on the actual version found
virgl support     yes
KVM support       yes
OpenGL support    yes
OpenGL dmabufs    yes
```

If any of these lines have "no" instead of "yes", then some packages might still be missing.

The configure script usually tells you which need to be installed on your system - for example since you requested SDL (Simple Direct Medialayer, not SmartDeviceLink), it must be installed etc.

If everything's okay up to this point, issue this command to compile QEMU. All of your processor's cores will be used in order to speed up the compilation.

```
make -j$(grep -c ^processor /proc/cpuinfo)
```

If the file `x86_64-softmmu/qemu-system-x86_64` exists after this command finishes, it means that the compilation was successful. That file is the main QEMU executable that we're going to use.

Running

It is assumed that `$QEMU_VIRGL_PATH` points to the location of the QEMU executable that was just compiled. Replace this string with the path where the executable sits on your computer, or simply set the variable to have the shell do your work for you.

For example:

```
QEMU_VIRGL_PATH=$PWD/x86_64-softmmu/qemu-system-x86_64
```

(You need also to adjust the paths to the bzImage file, and the genivi-dev-platform ext4 image.)

Running doesn't really differ a lot from the normal way of running QEMU - there's just a couple of extra flags.

```
"$QEMU_VIRGL_PATH" \
-kernel bzImage \
-drive "file=genivi-dev-platform-qemux86-64.ext4,format=raw" \
-enable-kvm \
-device virtio-vga,virgl=on \
-device nec-usb-xhci \
-device usb-tablet \
-net nic \
-net user,hostfwd=tcp::5555-:22 \
-cpu core2duo \
-no-reboot \
-soundhw ac97 \
-m 1024 \
-display sdl,gl=on \
--append "vga=0 uvesafb.mode_option=640x480-32 root=/dev/hda rw mem=512M oprofile.timer=1"
```

Comments:

- Using -drive + format instead of "-hda" avoids a warning about unspecified format
- To avoid issues with mouse input - the argument "-usbdevice tablet" had to be added. The new version of this is **-device usb-tablet**. The old was **-usbdevice tablet.**
  When using **-device usb-tablet** you must also preceed it with the USB controller: **-device nec-usb-xhci** for USB3.
  From version 2.10, there is a warning saying to prefer "-device usb-tablet".
- When running later image features like Chromium browser you should increase memory, for example: **-m 2048**

Once the system loads, you can verify that the hardware acceleration is working by ssh'ing to the machine and checking the kernel messages :

```
# dmesg | grep gl
[drm] virgl 3d acceleration enabled
```