XenGT Setup Guide

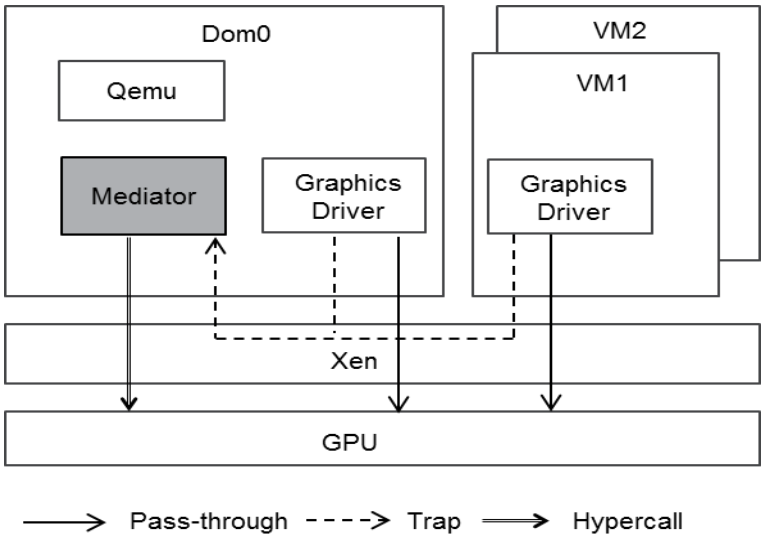
July 15, 2014

Contents

1 Introduction

The Graphics Processing Unit (GPU) has become a fundamental building block in today's computing environment, accelerating tasks ranging from entertaining (gaming, video playback, etc.), GUI acceleration, office applications (such as CAD, photoshop) and high performance computing. Recently, there observes a trend toward adding GPU accelerations to virtual machines provided by popular desktop virtualization. In the meantime, there are also demands for buying GPU computing resources from the cloud. GPU virtualization is becoming a demanding and challenging industry asking.

Intel has brought its answer to this challenge with **XenGT**, which is a mediated pass-through solution based on Intel Gen Graphics hardware using the well-known Xen hypervisor. As illustrated below, XenGT allows running a native graphics driver



in VMs to achieve high performance. Each VM is allowed to access a partial performance critical resource without hypervisor intervention. Privileged operations are trapped and forwarded to the mediator for emulation. The mediator creates a virtual GPU context for each VM and schedules one of them to run on physical Gen graphics hardware. In our implementation, the mediator is a separate driver residing in Dom0 kernel, called ***vgt driver***.

Currently, XenGT supports 4 accelerated VMs (Dom0 + 3 HVM DomU) running

together. We've verified XenGT's functionality using the 64-bit version of Ubuntu 12.04 and 14.04.

2 System Requirements

2.1 Operating System Requirements

The build and install environment has been validated in using x86_64 Ubuntu 12.04 and 14.04 as host.

2.2 Hardware Requirements

4th Generation Intel® Core™ Processor Graphics is required.

2.3 Software Requirements

2.3.1 Proxy Configuration Setup

If you building package behind firewall, you would need to setup following proxies in order to download needed libraries

```
# export http_proxy=<proxy_server>:<proxy_port>
# export https_proxy=<proxy_server>:<proxy_port>
# export ftp_proxy=<proxy_server>:<proxy_port>
```

To configure the Proxy for apt, you need add following lines into /etc/apt/apt.conf

```
Acquire::http::proxy "<proxy_server>:<proxy_port>";
Acquire::https::proxy "<proxy_server>:<proxy_port>";
Acquire::ftp::proxy "<proxy_server>:<proxy_port>";
```

2.3.2 Install Basic Packages in Ubuntu

Some packages only exist on Ubuntu 12.04, e.g., libghc6-bzlib-dev. We may ignore them unless we see any issue.

```
# apt-get update
# apt-get install libarchive-dev libghc-bzlib-dev libghc6-bzlib-dev \
zlib1g-dev mercurial gettext bcc iasl uuid-dev libncurses5-dev kpartx \
```

```
libperl-dev libgtk2.0-dev libc6-dev-i386 libaio-dev libsdl1.2-dev \
nfs-common libyajl-dev libx11-dev autoconf libtool xsltproc bison flex \
xutils-dev xserver-xorg-dev x11proto-gl-dev libx11-xcb-dev vncviewer \
libxcb-glx0 libxcb-glx0-dev libxcb-dri2-0-dev libxcb-xfixes0-dev bridge-utils \
python-dev bin86 git vim libssl-dev pciutils-dev tightvncserver ssh texinfo –y
```

3 Build and Install Instructions

3.1 Source Repositories

Xen: https://github.com/01org/XenGT-Preview-xen

Linux: https://github.com/01org/XenGT-Preview-kernel

Qemu: https://github.com/01org/XenGT-Preview-qemu

3.2 Building Kernel

```
# git clone https://github.com/01org/XenGT-Preview-kernel.git
# cd XenGT-Preview-kernel/
# git clone git://kernel.ubuntu.com/ubuntu/ubuntu-saucy.git linux-vgt
# cd linux-vgt
# git checkout 549fad2377f797d330565a7a7669b478ba474091 -b v3.11.6
# patch –p1 < ../linux-vgt.patch
# cp config-3.11.6-dom0 .config
# make –j8 && make modules_install
# mkinitramfs –o /boot/initrd-vgt-3.11.6-vgt.img 3.11.6-vgt+
# cp arch/x86/boot/bzImage /boot/vmlinuz-vgt-3.11.6-vgt
# cp vgt.rules /etc/udev/rules.d
# chmod a+x vgt_mgr
# cp vgt_mgt /usr/bin
```

3.3 Building Xen and Qemu

```
# git clone https://github.com/01org/XenGT-Preview-xen.git
# cd XenGT-Preview-xen/
# git clone https://github.com/01org/XenGT-Preview-qemu.git
# git clone git://xenbits.xen.org/xen.git xen-vgt
# git clone git://git.qemu-project.org/qemu.git qemu-xen
# cd qemu-xen
# git checkout –b v1.3.0 v1.3.0
```

```
# patch –p1 < ../XenGT-Preview-qemu/qemu-vgt.patch
# cd ../xen-vgt
# git checkout -b RELEASE-4.3.1 RELEASE-4.3.1
# patch –p1 < ../xen-vgt.patch
# cp –r ../qemu-xen tools/
# sed -i
'/QEMU_UPSTREAM_URL/s:http\://xenbits.xen.org/git-http/qemu-upstream-4.3-testing.git:$(XEN
_ROOT)/tools/qemu-xen:' Config.mk
# sed -i
'/QEMU_UPSTREAM_URL/s:git\://xenbits.xen.org/qemu-upstream-4.3-testing.git:$(XEN_ROOT)/t
ools/qemu-xen:' Config.mk
# ./autogen.sh
# ./configure --prefix=/usr    # XEN4.3 changes the default path to /usr/local
# make –j8 xen tools
# cp xen/xen.gz /boot/xen-vgt.gz
# make install-tools PYTHON_PREFIX_ARG=
# rm –f /etc/ld.so.conf.d/lib64.conf # the file may not exist on some Ubuntu distribution
# ldconfig
```

3.4 Grub Setup

You need manually add a new grub entry in /boot/grub/grub.cfg and make the entry as the default one when booting. Below is a reference grub entry for you. UUID (2e01a442-d848-4695-b031-9296ce3105b1) and root partition (hd0, msdos1) below are just reference which should be updated according to the user's environment.

```
menuentry 'Xen-VGT 3.11.6' --class ubuntu --class gnu-linux --class gnu --class os {
    insmod part_msdos
    insmod ext2
    set root='(hd0,msdos1)'
    search --no-floppy --fs-uuid --set=root 2e01a442-d848-4695-b031-9296ce3105b1
    multiboot     /boot/xen-vgt.gz dom0_mem=2048M loglvl=all guest_loglvl=all
conring_size=4M noreboot
```

```
    module    /boot/vmlinuz-vgt-3.11.6-vgt
root=UUID=2e01a442-d848-4695-b031-9296ce3105b1 rw rd_NO_LUKS rd_NO_LVM
LANG=en_US.UTF-8 rd_NO_MD SYSFONT=latarcyrheb-sun16 rhgb crashkernel=auto
KEYBOARDTYPE=pc KEYTABLE=us rd_NO_DM ignore_loglevel console=tty0 console=hvc0
consoleblank=0 log_buf_len=4M xen_vgt.hvm_boot_foreground=1
    module    /boot/initrd-vgt-3.11.6-vgt.img
```

Description of suggested parameters for grub

Configuration option	Description
xen_vgt.hvm_boot_foreground	Make VM immediately visible on the screen, after creation
xen_vgt.vgt	Option to enable/disable vgt for Dom0. 0 to disable vgt, 1 to enable vgt(default value)
xen_vgt. propagate_monitor_to_guest	Option to use direct/indirect display mode(default true, direct display mode)

3.5 Dom0 System Config Setup for XenGT

3.5.1 Starting Xen Services by Default

```
# update-rc.d xencommons defaults
```

3.5.2 Building Graphic Stack – Optional (Not needed for Ubuntu 14.04)

If you want to run 3D workloads in Dom0, the user mode driver update is required to support 4th Generation Intel® Core™ Processor Graphics. It is not a required step if dom0 does not start X.

```
# apt-get install git build-essential libtool autoconf libpthread-stubs0-dev \
libpciaccess-dev xutils-dev xserver-xorg-dev bison x11proto-gl-dev \
x11proto-xext-dev libxdamage-dev xserver-xorg-dev libx11-xcb-dev \
libxcb-glx0-dev libxcb-dri2-0-dev libxext-dev libexpat1-dev libxcb-xfixes0-dev
```

If you want to build and install the libraries to system "/usr/" directory directly, you do not need below environment variables. Just simply add option "--prefix=/usr" for "autogen.sh" command. If you do not want to pollute system, please follow below steps:

```
# export LD_LIBRARY_PATH=/opt/hsw/usr/lib
# export PKG_CONFIG_PATH=/opt/hsw/usr/lib/pkgconfig:/opt/hsw/usr/share/pkgconfig/
# mkdir -p /opt/hsw/usr
# cd /opt/hsw
# git clone git://anongit.freedesktop.org/git/mesa/drm
# cd drm
# git checkout 171666e4b8127c17c68ea0d44cf4e81ec342f2d0
# ./autogen.sh --prefix=/opt/hsw/usr
# make && make install
# cd ..
# git clone
git://anongit.freedesktop.org/git/xorg/driver/xf86-video-intel
# cd xf86-video-intel
# git checkout b6d2bb961517
# ./autogen.sh --prefix=/opt/hsw/usr
# make && make install
# cd ..
```

Notice that in above steps we check out a specific revision of x driver. The reason not to use the latest commit is that the driver has dependence to X. Latest x driver requires you to build latest Xorg as well. Above commit has proved to work fine with default Ubuntu 12.04 Xorg. For Ubuntu 14.04, we recommend you to use system default driver packages.

```
# apt-get build-dep mesa
# git clone git://anongit.freedesktop.org/git/mesa/mesa
# cd mesa
# git checkout a585b8f3a6681d1138ed1a33ed4c3195a53c2a73
# ./autogen.sh    --disable-gallium-egl --disable-gallium-gbm --without-gallium-drivers
--with-dri-drivers=i965 --prefix=/opt/hsw/usr
# make && make install
# cd ..
```

Then use the new driver for your system:

```
# cd /usr/lib/xorg/modules/drivers
backup original intel_drv.so
```

```
# ln -sf /opt/hsw/usr/lib/xorg/modules/drivers/intel_drv.so intel_drv.so
# cp newGL.conf /etc/ld.so.conf.d/
newGL.conf contains only one line of "/opt/hsw/usr/lib"
# ldconfig
```

3.5.3 Configuring Xen Bridge

After Dom0 reboots, run following commands to make a bridge “xenbr0” for guest network. (Assume the IP address of system could be acquired via DHCP and the default network interface with network connection is “eth0”)

```
# brctl addbr xenbr0
# ifconfig eth0 0.0.0.0 down
# brctl addif xenbr0 eth0
# ifconfig eth0 up
# dhclient xenbr0
```

3.6 Guest Setup

Guest Setup describes how user can prepare their own guest OS: Ubuntu or Window that work on XenGT.

3.6.1 General Setup

You need create an empty image with at least 10GB for guest. Here we take Ubuntu 13.04 guest image as example.

```
# dd if=/dev/zero of=system-10G.img bs=1M seek=10000 count=0
```

After you get the installation ISO for Ubuntu 13.04 guest, you could set the xmexample.conf by following section 3.6.2 with 3 changes and start the guest with the config file.

```
disk = [ 'file:/path/to/system-10G.img,hda,w', 'file:/path/to/ubuntu/iso,hdc:cdrom,r' ]
boot="d"
vgt=0
```

Then you could follow the normal procedure to finish the Ubuntu installation.

Note: XenGT now have also supported Windows VM, so here the iso can be substitute with Windows iso.

3.6.2 Guest Config File

You could copy the xmexample.conf from dom0's /etc/xen and modify the parameters as following.

```
Kernel = “hvmloader”
builder = ‘hvm’
memory = 2048
name = “vgtHVMDomain”
vif = [ 'type=ioemu, bridge=xenbr0' ]
disk = [ 'file:/path/to/system-10G.img,hda,w', ',hdc:cdrom,r' ]
#device_model = 'qemu-dm'
device_model_version='qemu-xen'
device_model_override='/usr/lib/xen/bin/qemu-system-i386'
sdl=1
opengl=1
vnc=0
vncpasswd=''
serial='pty'
tsc_mode=0
stdvga = 0
usb=1
usbdevice='tablet'
keymap='en-us'
vgt=1
vgt_low_gm_sz=128
vgt_high_gm_sz=384
vgt_fence_sz=4
vgt_monitor_config_file=’/path/to/monitor.config’ #only valid for indirect display mode
```

The description of the XenGT-specific parameters (don't set them before you upgrade the vgt kernels of Dom0 and DomU, or else, the HVM DomU creation will fail):

Configuration option	Description
Vgt	Enable virtual graphics
vgt_low_gm_sz	The low gm size which is CPU visible.

	For linux guest, it should be at least 64MB For windows guest, it should be at least 128MB
vgt_high_gm_sz	The high gm size which is CPU invisible
vgt_fence_sz	The number of the fence registers, default is 4
vgt_monitor_config_file	Only valid for indirect display mode. Specify the path to monitor configuration file

3.6.3 Linux Guest Setup for XenGT

3.6.3.1 Kernel and Modules Update

Assume that you have an Ubuntu x86_64 13.04 image. You could update the kernel and user mode drivers in guest image with following commands:

```
# kpartx -a -v /path/to/system-10G.img
```

The output will be something like below:

```
add map loop0p1 (253:0): 0 29638656 linear /dev/loop0 2048
add map loop0p2 (253:1): 0 1075202 linear /dev/loop0 29642750
add map loop0p5 : 0 1075200 linear 253:1 2
```

Mount loop0p1 to /mnt:

```
# mount /dev/mapper/loop0p1 /mnt/
```

Follow the steps in Section 3.2 to build Dom0 kernel and modules in guest with "chroot".

```
# chroot /mnt/
# exit
```

Alternatively, you could copy the kernel/initrd and modules from Dom0 to guest directly:

```
# cp /boot/vmlinuz-vgt-3.11.6-vgt /mnt/boot/
# cp /boot/initrd-vgt-3.11.6-vgt.img /mnt/boot/
# cp –r /lib/modules/3.11.6-vgt+ /mnt/lib/modules
```

Then you could run commands as below to fresh the guest image:

```
# umount /mnt
# kpartx -d -v /path/to/system-10G.img
```

Then you should add one new entry in "/boot/grub/grub.cfg" for the new kernel and initrd. Now, the image is ready for XenGT. You need set "vgt=1" in the xmexample.conf and start the guest.

3.6.3.2 Graphics Stack Update for Linux Guest – Optional (Not needed for Ubuntu 13.04)

If you want to run 3D workloads in Linux guest, the user mode driver update is required to support 4th Generation Intel® Core™ Processor Graphics. You could follow the steps in section 3.5.2 to build the Graphics Stack for Linux guest.

3.6.4 Windows Guest Setup for XenGT

According to the 3.6.1, assume you have already setup the Windows Guest. You need to set "vgt=1" in the xmexample.conf to start the Windows Guest. You can get the Windows driver from Intel website:

Windows driver version: 15.33.3621

32-bits:
https://downloadcenter.intel.com/Detail_Desc.aspx?DwnldID=23884&lang=eng&ProdId=3719

64-bits:
https://downloadcenter.intel.com/Detail_Desc.aspx?DwnldID=23885&lang=eng&ProdId=3719

The experimental driver has already enabled the media hardware acceleration.

After installing the driver, a system reboot is required.

4 VM Life Cycle Management

4.1 Guest Creation

To create a Guest, you need a configure file mentioned in section 3.6.2 and use the command:

```
# xl create xmexample.hvm
```

4.2 Listing Information about Domains

To get the information (ID, name, vCPU, Mem, etc) of all domains, you could use “xl list”.

```
# xl list
```

Name	ID	Mem	VCPUs	State	Time(s)
Domain-0	0	730	8	r-----	1139.2
ExampleHVMDomain	7	2048	1	r-----	1.0

4.3 Guest Destroy

Shutting down a Guest should be triggered in Guest and follow the normal procedure for Linux. To destroy a Guest, you could use “xl destroy” with Domain ID or Name, as below:

```
# xl destroy ExampleHVMDomain
```

4.4 Indirect Display Mode

XenGT offers two modes for display, direct display mode which is enabled by default and indirect display mode. In indirect display mode, all display resources are virtualized and virtual port presented to Guest could be different from the physical one.

To enable indirect display mode, you need to add `xen_vgt.propagate_monitor_to_guest=0` as kernel boot parameter and prepare a monitor.config as following.

```
# monitor.config file.

# first bit 1 for text mode, and second bit 2 for number of ports in the config
11

# 04 for PORT_E; and 01 for PORT_B to be override(virtual VGA on physical DP)
0401

# Virtual VGA monitor EDID
00ffffffffffff000469b1232df80000
0f16010381331d782a5ea5a2554da026
115054bfef00d1c095008140818081c0
950fb300714f023a801871382d40582c
4500fd1e1100001e000000ff0043344c
4d54463036333533330a000000fd0032
4b185311000a202020202020000000fc
00415355532050413233380a202000e0
```

Note that we only support virtual DVI and virtual VGA for now.

Add `vgt_monitor_config_file='/path/to/monitor.config'` to the xmexample.conf and then use `xl create xmexample.hvm` to boot the Guest.

5 XenGT Control Interfaces

5.1 Display Switch

When a Guest is created successfully with vgt enabled, the monitor display will be switched to the Guest with xen_vgt.hvm_boot_foreground set. A sys interface is provided to switch display between domains:

```
# xl list
Name                                        ID   Mem VCPUs      State   Time(s)
Domain-0                                     0   730     8      r-----   3263.5
ExampleHVMDomain                            11  2048     1      r-----      4.7
# cat /sys/kernel/vgt/control/foreground_vm
11
```

You could change the display by echo Domain ID into the sys interface:

```
# echo 0 > /sys/kernel/vgt/control/foreground_vm
# cat /sys/kernel/vgt/control/foreground_vm
0
```

5.2 Xen API

Refer to XenGT-API.txt under xen-vgt on how to use XenGT API interfaces.

5.3 Virtual Hotplug(Indirect display mode)

In indirect display mode, we support virtual display hotplug for Guest.

Plug-out:

```
# echo disconnect > /sys/kernel/vgt/vm#/PORT_#/connection    (PORT_# refers to virtual port)
```

Plug-in:

```
# echo –n $edid > /sys/kernel/vgt/vm#/PORT_#/edid_text    ($edid refers to the information of the monitor you want Guest to recognize. PORT_# refers to virtual port)
```

echo “PORT_#” > /sys/kernel/vgt/vm#/PORT_#/port_override (The first is physical port, the second is virtual port)

echo connect > /sys/kernel/vgt/vm#/PORT_#/connection (PORT_# refers to virtual port)

6 Features Supported

6.1 Per-VM Features

Features or Areas	Status
SMP Dom0 and Guest	Supported
2D Blitter	Supported
3D Rendering (Direct3D/OpenGL)	Supported
Single Monitor (HDMI/VGA/eDP/DP)	Supported
Multiple Monitors (HDMI/VGA/eDP/DP)	Supported
PPGTT	Supported
Dom0 S3	Supported
Indirect display	Supported
Windows 7/8.1	Supported
Media Decoding Hardware Acceleration	Experimental Supported

6.2 Virtualization Features

Features or Areas	Status
Up to 3 vGT Guests	3 Guests, each with 128MB low graphics memory.
Render Context Switch	Supported
Display Switch	Supported
VM Life Cycle	Supported
XenGT Interfaces (APIs)	Refer to the API document.
Monitor Hotplug	Supported
Different Monitor Resolutions	Supported
GPU recovery	Preliminarily supported

7 Known Issues

- At least 2GB memory is suggested for VM to run most 3D workloads.
- compiz not working:
  compiz may not work in Dom0 and linux guest. You need select "Unity 2D" during UX login instead. Or you could just edit "/etc/lightdm/lightdm.conf", change "user-session=ubuntu" to be "user-session=ubuntu-2d".
- keymap might be incorrect in guest
  config file may need to explicitly specify "keymap='en-us'". Although it looks like the default value, earlier we saw the problem of wrong keymap code if it is not explicitly set.
- Booting Ubuntu 12.04 VM may see some call traces in system log.
- When using three monitors, doing hotplug between Guest pause/unpause may not be able to lightup all monitors automatically.
- Some specific monitor issues.

8 Appendix

8.1 Licenses

8.1.1 General Statement

Only newly-created files are explicitly specified with a license here. Any changes to existing files of Xen/Linux/Qemu are subject to the licenses of the files.

8.1.2 Licenses of the Newly-created Files

8.1.2.1 Xen Side

All the newly-created files are under GPLv2:

tools/firmware/hvmloader/vgt.h

xen/arch/x86/hvm/vgt.c

xen/arch/x86/vgt.c

xen/include/asm-x86/vgt.h

8.1.2.2 Linux Side

These newly-created files are under GPLv2:

arch/x86/include/asm/xen/x86_emulate.h

arch/x86/xen/vgt_emulate.c

arch/x86/xen/x86_emulate.c

drivers/xen/vgt/debugfs.c

drivers/xen/vgt/dev.c

drivers/xen/vgt/hypercall.c

drivers/xen/vgt/klog.c

drivers/xen/vgt/Makefile

drivers/xen/vgt/sysfs.c

drivers/xen/vgt/trace.h

tools/vgt/klog.c

tools/vgt/Makefile

tools/vgt/README

tools/vgt/vgt_report

vgt.rules

XenGT-API.txt

These newly-created files are under dual GPLv2/MIT:

drivers/gpu/drm/i915/i915_gem_vgtbuffer.c

drivers/xen/vgt/aperture_gm.c

drivers/xen/vgt/cfg_space.c

drivers/xen/vgt/cmd_parser.c

drivers/xen/vgt/cmd_parser.h

drivers/xen/vgt/devtable.h

drivers/xen/vgt/display.c

drivers/xen/vgt/edid.c

drivers/xen/vgt/edid.h

drivers/xen/vgt/fb_decoder.c

drivers/xen/vgt/gtt.c

drivers/xen/vgt/handlers.c

drivers/xen/vgt/instance.c

drivers/xen/vgt/interrupt.c

drivers/xen/vgt/mmio.c

drivers/xen/vgt/reg.h

drivers/xen/vgt/render.c

drivers/xen/vgt/sched.c

drivers/xen/vgt/utility.c

drivers/xen/vgt/vgt.c

drivers/xen/vgt/vgt.h

include/xen/fb_decoder.h

include/xen/vgt.h

include/xen/vgt-if.h

8.1.2.3 Qemu Side

There are 2 newly-created files, which are under dual GPLv2/MIT:

hw/vga-xengt.c

hw/vga-xengt.h