Nvidia Optimus

Некоторые современные ноутбуки оснащены технологией «nVidia Optimus». Это гибридный графический процессор (GPU) nVidia и Intel; это НЕ два отдельных GPU в одном ноутбуке. Графика nVidia обеспечивает производительность, графика Intel — энергосбережение при обычном использовании. К сожалению, средства настройки или переключения не включены в проприетарные закрытые драйверы, в отличие от поставляемых Advanced Micro Devices Inc. (AMD).

Вопреки этой неприятности существует открытый проект Шмель (Bumblebee), нацеленный на решение указанной проблемы. Тем не менее Шмель не одинок. Разработчики nVidia помогают проекту исправлением ошибок закрытого драйвера, к которому у разработчиков Шмеля нет доступа.

Во избежание путаницы здесь и далее будем использовать nVidia для ссылки на компанию и `nvidia` для ссылки на закрытый проприетарный драйвер.

В командах терминала этой страницы начинающиеся с `$` строки означают команды, выполняемые обычным пользователем, а начинающиеся с `#` должны выполняться суперпользователем root.

Также обратите внимание на разницу между Bumblebee и bumblebee. bumblebee является демоном первого.

X11 и прочие программы без Шмеля

Шмель НЕ нужен для использования X11, KDE, Compiz-Fusion, Blender, Adobe Flash Player и даже некоторых игр вроде Guild Wars или Warcraft III. Графика Intel обеспечивает достаточное ускорение 3D (а иногда и производительнее) для плавной работы подобных программ без участия GPU от nVidia. Дополнительным плюсом является экономия энергии.

Если всё ещё хотите использовать Шмеля для переключения между GPU nVidia и Intel, читайте дальше. Мы просто хотели пояснить, что Intel в ускорении 3D продвинулась довольно далеко, и не каждой программе действительно требуется для работы GPU от nVidia.

Установка Шмеля

Автоматизированный способ

Благодаря слакеру Ryan McQuen есть новый автоматизированный установки Шмеля.

ИСПОЛЬЗОВАТЬ ТОЛЬКО НА СВОЙ РИСК! Если вы правили слакбилды, скорее

всего этот сценарий вам не подойдет.

Можно выполнить от root:

```
curl https://raw.githubusercontent.com/ryanpcmcquen/linuxTweaks/master/slackware/crazybee.sh | sh
```

Либо загрузите файл сценария и выполните его от root.

Сценарий обнаруживает мультилиб, создает необходимую группу, добавляет пользователей и вносит изменения в /etc/rc.d/rc.local, а также поддерживает параметр STABLE:

```
curl https://raw.githubusercontent.com/ryanpcmcquen/linuxTweaks/master/slackware/crazybee.sh | STABLE=yes sh
```

Он также использует `upgradepkg --reinstall --install-new` после обновлений ядра.

Ручной способ

Получение слакбилдов

Уважаемый слакер jgeboski изначально предоставил слакбилды для Шмеля в соответствии с требованиями SlackBuilds.org. Теперь другой слакер продолжил начатое в своём репозитории github. Не забудьте сперва проверить текущий каталог, поскольку именно в него будет производиться загрузка. Теперь, когда готовы, слакбилды могут быть получены командой `git clone` как обычно. Затем мы перейдём в новый каталог и приступим к сборке.

```
$ git clone https://github.com/whitewolf1776/Bumblebee-SlackBuilds.git
$ cd Bumblebee-SlackBuilds
```

Вместо этого можно загрузить архив со слакбилдами:

```
$ wget --content-disposition https://github.com/whitewolf1776/Bumblebee-SlackBuilds/tarball/master
$ tar xf whitewolf1776-Bumblebee-SlackBuilds-<шестнадцатеричная строка>  # Полезно автодополнение по Tab.
$ cd whitewolf1776-Bumblebee-SlackBuilds
```

Файл README прилагается, но поскольку вряд ли захотите читать ещё и его в дополнение к этой инструкции, приведём здесь основные моменты.

Получение исходных файлов

Исходные файлы по обыкновению для слакбилдов нужно вручную загрузить и разместить в соответствующих каталогах. В составе репозиториев есть файл SLACKBUILDS.TXT, содержащий ссылки на исходные файлы и их контрольные суммы MD5 для проверки целостности. Для их

загрузки и проверки можно воспользоваться предоставленным сценарием:

```
./download.sh
```

Подготовка окружения

Приступим, но сперва нужно кое-что подготовить. Прежде всего, потребуется группа для пользователей, которым разрешен доступ к технологии Шмель. Для простоты назовём группу «bumblebee». Помимо самой группы нужно добавить в неё пользователей, которым разрешено использование Шмеля:

```
# groupadd bumblebee
# usermod -aG bumblebee <ПОЛЬЗОВАТЕЛИ>
```

где <ПОЛЬЗОВАТЕЛИ> – имя или список имён пользователей, добавляемых в группу.

Сборка и установка Шмеля

1. Сборка и установка: bbswitch (необязательно, но настоятельно рекомендуется).

```
# cd bbswitch # ''cd'' не обязательно от root, но кому захочется потом каждый раз ''su''?
# ./bbswitch.SlackBuild
# upgradepkg --install-new bbswitch-*.t?z
```

2. Сборка и установка: libbsd (обязательно).

```
# cd ../libbsd
# ./libbsd.SlackBuild
# upgradepkg --install-new libbsd-*.t?z
```

3. Сборка и установка: bubmlebee (очевидно обязательно).

```
# cd ../bumblebee
# ./bumblebee.SlackBuild
# upgradepkg --install-new bumblebee-*.t?z
```

Сборка и установка прочих зависимостей

Некоторые слакбилды поддерживают параметр COMPAT32 для мультибиблиотечных систем. Ниже показана сборка с и без 32-битной совместимости. Используйте один требуемый вариант и не забывайте об этом при сборке указанных пакетов.

Шмель может использовать primusrun (требует пересборку mesa в Slackware 14.0 и ранее) или собственный optirun. Выберите, чем будете пользоваться (можно

поставить оба). У каждого есть свои плюсы и минусы.

Primus

Primus, как и optirun (инструкции ниже) можно использовать для запуска выбранной программы через Шмеля. Primus также поддерживает параметр COMPAT32.

Если используете Slackware 14.0 или старше, потребуется пересобрать mesa. Новый репозиторий github не не предоставляет слакбилды mesa для Slackware 14.0 и старше. Поэтому необходимо использовать Slackware 14.1 или новее, либо пересобирать mesa вручную другим сценарием. Возможно выбрать старый репозиторий и соответственно поправить слакбилд, либо добавить `--enable-shared-glapi` в mesa.SlackBuild как в новой версии Slackware.

```
# # Для старого репозитория выполните:
# cd ../mesa
# ./mesa.SlackBuild
```

У mesa нет параметра COMPAT32, поскольку это официальный пакет Slackware, но создать пакет compat32 после оригинального несложно:

```
# ./mesa-compat32.SlackBuild # только для старого репозитория
```

1. Сборка и установка: primus

```
# cd ../primus
# ./primus.SlackBuild
# upgradepkg --install-new primus-*.t?z
```

```
# COMPAT32=yes ./primus.SlackBuild
```

VirtualGL

Хотя VirtualGL (и его зависимость libjpeg-turbo) ещё поддерживаются проектом Шмель и сопровождается его разработчиками, новый репозиторий слакбилдов для этих пакетов не содержит. Этот раздел для старого репозитория github от jgeboski. Если вам нужен VirtualGL (скажем, у вас тот редкий случай, когда VirtualGL производительнее, чем primus), то можете использовать эти (устаревшие) или собственные слакбилды.

1. Сборка и установка: libjpeg-turbo (доступна поддержка 32 бит)

```
# cd ../libjpeg-turbo
# ./libjpeg-turbo.SlackBuild                              # Без поддержки 32 бит.
# upgradepkg --install-new libjpeg-turbo-*.t?z
```

Если нужна сборка с поддержкой 32-битной совместимости, добавьте COMPAT32=yes перед вызовом сценария:

```
# COMPAT32=yes ./libjpeg-turbo.SlackBuild  # Только для систем x86_64, поддержка 32-битных библиотек и файлов.
```

libjpeg-turbo устанавливается в /opt, поскольку в противном случае может перезаписать файлы оригинального libjpeg.

2. Сборка и установка: VirtualGL (доступна поддержка 32 бит)

```
# cd ../VirtualGL
# ./VirtualGL.SlackBuild
# upgradepkg --install-new VirtualGL-*.t?z
```

Думаю, уже уловили идею...

```
# COMPAT32=yes ./VirtualGL.SlackBuild
```

Проприетарный драйвер nvidia

Если хотите использовать проприетарные драйверы nVidia, необходим отказ от nouveau, поскольку они конфликтуют друг с другом. Возможные варианты: удалить nouveau, установить xf86-video-nouveau-blacklist из /extra, вручную занести nouveau в чёрный список.

Эта часть целиком необязательна. Slackware 13.37 и последующие поставляются с xf86-video-nouveau, открытым драйвером для видеокарт nVidia. Если используете его, закрытые проприетарные драйверы nVidia не нужны. Однако, если хотите использовать закрытый драйвер, установите перечисленные в этом разделе пакеты, прежде чем продолжить.

Некоторые версии драйвера nvidia несовместимы с отдельными версиями ядра и наоборот по различным причинам. Как правило, лучше использовать пакет драйвера nvidia, совместимый с пакетами ядра Slackware. Можно действовать по инструкции SlackBuilds.org для самостоятельного ручного обновления версий отдельных пакетов.

Если используете Slackware 14.1 или старше, нужно установить libvdpau. Более новые выпуски Slackware уже содержат libvdpau.

```
# cd ../libvdpau
# ./libvdpau.Slackbuild
```

```
# upgradepkg --install-new libvdpau-*.t?z
```

1. Сборка и установка: nvidia-bumblebee

```
# cd ../nvidia-bumblebee
# ./nvidia-bumblebee.SlackBuild
# upgradepkg --install-new nvidia-bumblebee-*.t?z
```

```
# COMPAT32=yes ./nvidia-bumblebee.SlackBuild # Только для систем x86_64, поддержка 32-битных библиотек и файлов.
```

2. Сборка и установка: nvidia-kernel

```
# cd ../nvidia-kernel
# ./nvidia-kernel.SlackBuild
# upgradepkg --install-new nvidia-kernel-*.t?z
```

После установки

Превосходно. Пора сделать послеустановочные настройки. Пакет bumblebee предоставляет сценарий rc.bumblebee в /etc/rc.d, где расположены и остальные стартовые сценарии. Не забудьте сделать его исполняемым и, при желании, запустить!

```
# chmod +x /etc/rc.d/rc.bumblebee
# /etc/rc.d/rc.bumblebee start
```

Если хотите автоматически запускать демона Шмеля при старте системы, можете добавить в /etc/rc.d/rc.local следующее:

```
[ -x /etc/rc.d/rc.bumblebee ] && /etc/rc.d/rc.bumblebee start
```

Можете пойти ещё дальше, добавив остановку демона Шмеля в /etc/rc.d/rc.local_shutdown. Если последний отсутствует, можете создать.

```
[ -x /etc/rc.d/rc.bumblebee ] && /etc/rc.d/rc.bumblebee stop
```

Готово! Для запуска программ на видеокарте nVidia воспользуемся «optirun» или «primusrun»:

```
$ optirun glxspheres
$ primusrun glxspheres
```

Правда легко? Полагаю, не для всех. Если выбрали использование VirtualGL (optirun), некоторые программы прямо вот так не заработают. Некоторые потребуют предварительно запустить bash через optirun:

```
$ optirun bash
```

Теперь можно запустить программу не добавляя в начале «optirun»:

```
$ wine ~/.wine/drive_c/Program_Files/Starcraft\ 2/Starcraft\ 2.exe
```

Правка файлов настроек

Несколько общих правок etc/bumblee/bumblebee.conf (скорее всего, подойдут всем):

```
# Если вначале добавили пользователя в отличную от «bumblebee» группу, измените соответственно строку «ServerGroup=»
# Здесь мы полагаем «ImAllergicToBees» (у меня аллергия на пчёл):
ServerGroup=ImAllergicToBees

# Возможно, не нравится использование для X11 дисплея :8. Это тоже можно поменять, например, на 3:
VirtualDisplay=:3

# Хочу указать используемый bumblebee по умолчанию драйвер. Показан «nouveau», но поменять на «nvidia» не составит труда.
# Далее до конца этого небольшого раздела будем для простоты использовать «nouveau».
Driver=nouveau

# Не забудьте также добавить соответствующие ему строчки.Don't forget to add their
KernelDriver=nouveau
Module=nouveau
```

ЧаВо, на которые может не ответить Google

---

1. В: Я обновил ядро, теперь получаю «Fatal: module bbswitch not found» и шмель не работает! Что мне делать?

О: bbswitch и nvidia-kernel являются модулями ядра, файлы модулей расположены в /lib/modules/<версия_этого_ядра>, поэтому после смены ядра просто пересоберите и переустановите пакеты nvidia-kernel и bbswitch, чтобы их новые файлы попали в новый каталог /lib/modules/<версия_нового_ядра>.

2. В: При обновлении nvidia-kernel нужно ли обновлять nvidia-bumblebee (и наоборот)?

О: Нет. Несмотря на использование общих исходных файлов, устанавливают они весьма разные вещи. nvidia-kernel является обычным модулем ядра, а nvidia-bumblebee представляет остальную часть приложения.

3. В: (Продолжение №2) А при установке новой версии проприетарного драйвера nvidia?

О: В этом случае обновить нужно и nvidia-bumblebee, и nvidia-kernel.

4. В: Что такое мультилиб/compat32 и нужно ли мне это?

О: Это не особенность шмеля и зависит от вашего решения. Прочтите про мультибиблиотечность и решите, нужно это вам или нет.

Отдельные модели ноутбуков и GPU nVidia

Шмель показывает разные результаты в зависимости от видеокарты nVidia и драйвера. Где-то лучше, где-то хуже. Надеюсь, если у вас есть предложения по настройке, опишите их в разделе обсуждения, чтобы помочь владельцам аналогичных видеокарт с настройкой nVidia Optimus (затем настройки будут добавлены к статье). Одни настройки могут работать на всех ноутбуках, другие нет. На всякий случай, будьте осторожны. Поскольку неизвестно, сколь малые различия возможны между видеокартами, не забывайте указать модель ноутбука и используемый драйвером xorg.conf.

Следующее изменение подходит для мультибиблиотечных систем.

- /etc/bumblebee/bumblebee.conf

```
# Для мультибиблиотечной системы в LibraryPath используются и /usr/lib64, и /usr/lib:
LibraryPath=/usr/lib64/nvidia-bumblebee:/usr/lib/nvidia-bumblebee:/usr/lib64:/usr/lib
XorgModulePath=/usr/lib64/nvidia-bumblebee/xorg,/usr/lib64/xorg/modules
```

Источники

- Проект Шмель: http://bumblebee-project.org/
- Вики проекта Шмель: https://github.com/Bumblebee-Project/Bumblebee/wiki
- Создание страницы стало возможным благодаря: jgeboski
- Исходные слакбилды: https://github.com/jgeboski/Bumblebee-SlackBuilds
- Новые слакбилды: https:*github.com/whitewolf1776/Bumblebee-SlackBuilds * Автор: TommyC * Перевод: Serg Bormant*
  howtos, software, hardware, nvidia, author tommyc, translator bormant

From:
https://docs.slackware.com/ - **SlackDocs**

Permanent link:
https://docs.slackware.com/ru:howtos:hardware:nvidia_optimus

Last update: **2016/06/17 20:16 (UTC)**