Setting up and using kgdb

Martin Brown
Required Hardware
Required Hardware

In order to use kgdb, you need to have two machines:

Development Machine "mDev"

Debugging Machine "mDebug"
Required Hardware cont'd

You'll need a way for the two machines to communicate

- Serial ports
  - "Null-modem serial cable"

- Ethernet (not tested)
Configuring the Machines
Configuring *mDev* - Kernel Config Options

Support for kgdb is included in the kernel since 2.6.26

You want to enable the following kernel CONFIG options on *mDev*

- **Kernel hacking**
  - Magic SysRq key
  - Compile the kernel with debug info
  - KGDB: kernel debugger
    - KGDB: use kgdb over the serial console
Configuring *mDev* - GRUB menu entry

```
# grub menuentry

menuentry 'Linux 3.2.36-kgdb' --class debian --class gnu-linux --class gnu --class os {
    insmod part_msdos
    insmod ext2
    set root=('hd0,msdos1')
    search --no-floppy --fs-uuid --set 651a539d-f4c8-48a5-ac00-a308d08a8103
    echo    'Loading Linux 3.2.36-kgdb ...'
    linux   /boot/vmlinuz-3.2.36-kgdb root=UUID=651a539d-f4c8-48a5-ac00-a308d08a8103 ro quiet
    print-fatal-signals=1 kgdbwait kgdb8250=io,03f8,ttyS0,115200,4 kgdboc=ttyS0,115200 kgdboe=@192.168.26.244/,@192.168.26.245/ kgdbcon
    echo    'Loading initial ramdisk ...'
    initrd  /boot/initrd.img-3.2.36-kgdb
}
```
Configuring *mDev* - GRUB menu entry

```bash
# grub menuentry

menuentry 'Linux 3.2.36-kgdb' --class debian --class gnu-linux --class gnu --class os {
    insmod part_msdos
    insmod ext2
    set root=('hd0,msdos1')
    search --no-floppy --fs-uuid --set 651a539d-f4c8-48a5-ac00-a308d08a8103
    echo 'Loading Linux 3.2.36-kgdb ...'
    linux /boot/vmlinuz-3.2.36-kgdb root=UUID=651a539d-f4c8-48a5-ac00-a308d08a8103 ro
    quiet print-fatal-signals=1 kgdbwait kgdb8250=io,03f8,ttyS0,115200,4 kgdboc=ttys0,115200
    kgdboe=@192.168.26.244/@192.168.26.245/ kgdbcon
    echo 'Loading initial ramdisk ...'
    initrd /boot/initrd.img-3.2.36-kgdb
}
```
Configuring \textit{mDev} - GRUB menu entry - boot arguments

Open /\texttt{etc}/\texttt{grub.d/40\_custom} and create an entry for your new kernel

\texttt{kgdbwait kgdb8250=io,03f8,ttyS0,115200,4 kgdboc=ttyS0,115200 kgdboe=@192.168.1.4/,@192.168.1.3/ kgdbcon}

makes \texttt{kgdb} wait for a debugger connection during booting of the kernel
Configuring *mDev* - GRUB menu entry - boot arguments

Open `/etc/grub.d/40_custom` and create an entry for your new kernel

```
k gdbwait **kgdb8250=io,03f8,ttyS0,115200,4** kgdboc=ttys0,115200 kgdboe=@192.168.1.4/,@192.168.1.3/ kgdbcon
```

*Specifying the serial address*
Configuring *mDev* - GRUB menu entry - boot arguments

Open `/etc/grub.d/40_custom` and create an entry for your new kernel

```plaintext
kgdbwait kgdb8250=io,03f8,ttyS0,115200,4 kgdboc=ttyS0,115200
kgdboe=@192.168.1.4/,@192.168.1.3/ kgdbcon
```

**how to communicate from gdb to kgdb**
Configuring mDev - GRUB menu entry - boot arguments

Open /etc/grub.d/40_custom and create an entry for your new kernel

```
kgdbwait kgdb8250=io,03f8,ttyS0,115200,4 kgdboc=ttyS0,115200
kgdboe=@192.168.1.4/,@192.168.1.3/ kgdbcon
```

*kgdb over ethernet*, not needed if you're using serial ports
Configuring *mDev* - GRUB menu entry - boot arguments

Open `/etc/grub.d/40_custom` and create an entry for your new kernel

```
kgdbwait kgdb8250=io,03f8,ttyS0,115200,4 kgdboc=ttyS0,115200
kgdboe=@192.168.1.4/,@192.168.1.3/ kgdbcon
```

allows you to see `printk()` messages while running gdb on *mDebug*
Configuring *mDebug*

- *mDebug* needs to have a copy of the built *vmlinux* because it contains symbols needed for debugging
  - Copy the built *vmlinux* from *mDev* to *mDebug*

- install gdb on *mDebug*
  - `$ sudo apt-get install gdb`
Debugging
Debugging - Reboot mDev

Reboot *mDev* and select your kgdb-configured kernel at the GRUB menu

You will get a message saying something like:

... kgdb: Waiting for connection from remote gdb...

Entering kdb (current =0x..., pid x) on processor x due to Keyboard Entry
[0]kdb>
Debugging - Connecting Remote gdb

On *mDebug*, cd to the path containing *vmlinux* (which was copied from *mDev*)

```
$ gdb ./vmlinux
(gdb) set remotebaud 115200
(gdb) target remote /dev/ttyS0
(gdb) continue
```

*mDev* will then continue to boot
Debugging - Entering the Kernel Debugger

You can enter the kernel debugger by

1. Waiting for an oops or a fault

2. Manually by using sysrq-g
   a. Alt+SysRq (without letting go of Alt) followed by Alt+g
   or
   b. $ su
      $ echo g > /proc/sysrq-trigger
Debugging - kgdb Breakpoints

...  
#include <linux/kgdb.h>

EXPORT_SYMBOL_GPL("Dual BSD/GPL");

static int hello_init(void) {
    ...
    ...
    kgdb_breakpoint();
    ...
    return 0;
}
Debugging - gdb Commands

(gdb) nexti // next machine instruction,
    // useful after kgdb_breakpoint()

(gdb) backtrace // show the call stack

(gdb) continue // continue normal execution

(gdb) print *(<addr>) // prints a file and line number for the
    // corresponding to a hex address
Debugging - printk()

printk()'s on mDev now show up on mDebug!

You don't have to look through the other messages that show up when you run dmesg.
Debugging - Modules

Page 101 (Chapter 4) of Linux Device Drivers '05 talks about how to add your module's symbols for gdb

A link to the script that they mention is in References

I haven't gotten it to work yet ...
References

- [http://kgdb.linsyssoft.com/quickstart.htm](http://kgdb.linsyssoft.com/quickstart.htm) - How I got started, found out about null-model serial cable
- [http://kernel.org/doc/htmldocs/kgdb.html](http://kernel.org/doc/htmldocs/kgdb.html) - Guide to setting up kgdb
- [http://en.gentoo-wiki.com/wiki/KGDB](http://en.gentoo-wiki.com/wiki/KGDB) - This is a better guide to setting up kgdb IMO, helped me figure out the kernel boot arguments
- [http://lxr.linux.no](http://lxr.linux.no)
  - [http://lxr.linux.no/linux+v3.2.36/kernel/debug/debug_core.c](http://lxr.linux.no/linux+v3.2.36/kernel/debug/debug_core.c) - How I found out about kgdb_breakpoint()
  - [http://lxr.linux.no/linux+v3.2.36/include/linux/kgdb.h#L50](http://lxr.linux.no/linux+v3.2.36/include/linux/kgdb.h#L50) - How I found out what to include for kgdb_breakpoint() symbol
- [http://ww2.cs.fsu.edu/~diesburg/courses/dd/calendar.html](http://ww2.cs.fsu.edu/~diesburg/courses/dd/calendar.html) - Lecture 3 - Reminded me to declare GPL license
- [http://www.stanford.edu/class/cs107/other/gdbrefcard.pdf](http://www.stanford.edu/class/cs107/other/gdbrefcard.pdf) - gdb reference
- Rubini, Alessandro, and Jonathan Corbet. *Linux device drivers*. O'reilly, 2005. - Chapter 4
Demo