Project 1: Getting Acquainted

By David Baugh and Justin Sherburne
Group 21

Log of commands to build the Kernal:

1. `git clone git://git.yoctoproject.org/linux-yocto-3.19`

   This command clones the yocto client to your current directory

2. `git checkout 'v3.19.2'`

   Checkout the 'v3.19.2' tag and then source the instuctor's files as follows.

3. `source /scratch/files/environment-setup-i586-poky-linux`

4. `cp /scratch/files/core-image-lsb-sdk-qemux86.ext4 core-image-lsb-sdk-qemux86.ext4`

5. `cp /scratch/files/config-3.19.2-yocto-standard .config`

6. `make menuconfig`

   Here we make a couple of changes before building. First we verify that it is in 32-bit mode, then we rename the hostname to whatever we would like. In this case we named it Group 21 HW1.

7. `make -j4 all`

   Make the kernal. This finalizes our build before starting

8. `qemu-system-i386 -gdb tcp::5521 -S -nographic -kernel bzImage-qemux86.bin -drive file=core-image-lsb-sdk-qemux86.ext4, if=virtio -enable-kvm -net none -usb -localtime --no-reboot --append "root=/dev/vda rw console=ttyS0 debug"`

   Now the shell will hang waiting for the CPU to start. In another shell:

   (a) `gdb`
   (b) `target remote localhost:5521`
   (c) `continue`

   Finally, to exit the VM:

9. `poweroff`

Concurrency writeup

Breakdown of qemu command

```
qemu-system-i386 -gdb tcp::5521 -S -nographic -kernel bzImage-qemux86.bin -drive file=core-image-lsb
if=virtio -enable-kvm -net none -usb -localtime --no-reboot --append "root=/dev/vda
rw console=ttyS0 debug"
```

Source: *https://qemu.weilnetz.de/doc/qemu-doc.html*

- `qemu-system-i386`

  This portion of the command tells the system that we are going to be running qemu-i386. Qemu is just a program on the OS server that allows us to emulate our linux core.

- `-gdb tcp::5521 -S`

  This enables GDB support for our core. It also opens a port so you can connect using GDB, in our case 5521. The -S suspends the processor untill a GDB connection is made to allow for debugging during startup. If you remove the -S the kernal will start rather than waiting for GDB to connect.

- `-nographic`

  Disables the graphic UI that qemu usually opens. Instead you only have the CLI.

- `-kernel bzImage-qemux86.bin`

  This command specifies bzImage as the kernal image.

- `-drive file=core-image-lsb-sdk-qemux86.ext4,if=virtio`

  The drive portion of this command devines a new drive for the kernal to use. The `file=core-image-lsb-sdk-qemux` specifies which disk image to use, in this case we copied the disk image from the instructors files. The virtio specifies what type of interface it is using.

- `-enable-kvm`

  Enables full KVM virtualization support

- `-net none`

  Disables the use of any network devices.

- `-usb`

  Enables the USB driver if it is not enabled by default.

- `-localtime`

  This is a shortcut for the -rtc tag that sets up the clock for the kernal. Specifically this enables the time to be set to the local server time.

- `--no-reboot`

  Allows you to exit rather than rebooting.

- `--append "root=/dev/vda rw console=ttyS0 debug"`

  Allows you to direct linux boot without needing a full bootable image.

Questions

1. What do you think the main point of this assignment is?

   I think the main point of the concurrency assignment was to teach us how to manage many threads running at the same time. Creating processes is the easy part of this assignment, getting those to work together successfully is the difficult part.

2. How did you personally approach the problem? Design decisions, algorithm, etc.

   answer 2

3. How did you ensure your solution was correct? Testing details, for instance.

   answer 3

4. What did you learn?

   answer 4

Version Control Log

Work Log