

## 1 KERNEL AND VM SETUP STEPS TAKEN:

- 1) Open up two shells, call them Shell A and Shell B.
- 2) Git clone git://git.yoctoproject.org/linux-yocto-3.19
- 3) Change extension to 3.19.2
- 4) Be inside the linux-yocto-3.19.2 directory
- 5) Mv ../../files/bzImage-qemu86.bin .
- 6) Mv ../../files/config-3.19.2-yocto-standard .
- 7) Mv ../../files/environment-setup-i586-poky-linux .
- 8) Enter bash for both Shell A and B
- 9) Shell A: source environment-setup-i586-poky-linux
- 10) Shell A: qemu-system-i386 -gdb tcp::5514 -S -nographic -kernel bzImage-qemu86.bin -drive file=core-image-lsb-sdk-qemu86.ext4,if=virtio -enable-kvm -net none -usb -localtime -no-reboot -append "root=/dev/vda rw console=ttyS0 debug".
- 11) Shell B: source environment-setup-i586-poky-linux
- 12) Shell B: \$GDB
- 13) Shell B: Target remote : 5514, then enter continue and hit enter
- 14) Shell A: enter root and hit enter. Enter reboot and hit enter when ready to quit
- 15) Hey it worked! Now to make our own
- 16) Mv config-3.19.2-yocto-standard .config
- 17) Make -j4 all
- 18) Enter in make menuconfig
- 19) Go to general and change the kernel info to group 14 homework 1
- 20) Find bzImage in the folder arch/x86/boot/bzImage
- 21) Cp arch/x86/boot/bzImage /scratch/fall2017/14/linux-yocto-3.19.2
- 22) Shell A: source environment-setup-i586-poky-linux
- 23) Shell A: qemu-system-i386 -gdb tcp::5514 -S -nographic -kernel bzImage -drive file=core-image-lsb-sdk-qemu86.ext4,if=virtio -enable-kvm -net none -usb -localtime -no-reboot -append "root=/dev/vda rw console=ttyS0 debug".
- 24) Shell B: source environment-setup-i586-poky-linux
- 25) Shell B: \$GDB
- 26) Shell B: Target remote :5514
- 27) Shell B: Continue
- 28) Shell A: Root
- 29) Logged into VM and ready to go
- 30) Uname -a prints out our group number 14, homework 1

## 2 FLAGS

### Iqemu flags

- 1) -gdb tcp::5514  
(A) This opens a gdbserver on TCP port 5514.
- 2) -S  
(A) This means Do not start CPU at startup.
- 3) -nographic  
A Disables graphical output so that Qemu is a simple command line application.
- 4) -kernel bzImage  
A This command says to use the bzImage as the kernel image, which can either be a linux kernel or in multiboot format.
- 5) -drive  
A file=core-image-lsb-sdk-qemu86.ext

- (a) This option defines which disk image to use with this drive. For this situation, the disk image would be core-image-lsb-sdk-qemux86.ext
- B if=virtio
  - (a) This option defines on which type of interface is connected.
- 6) -enable-kvm
  - (A) Enable KVM full virtualization support. This option is only available if KVM support is enabled when compiling
- 7) -net-none
  - (A) Indicate that no network devices should be configured. It is used to override the default configuration which is activated if no -net options are provided
- 8) -usb
  - (A) Simply enables the USB driver
- 9) -localtime
  - (A) This is used to let the RTC start at the current local time
- 10) -no-reboot
  - (A) Exit instead of rebooting
- 11) -append root=/dev/vda rw console=ttyS0 debug
  - (A) This uses root=/dev/vda rw console=ttyS0 debug as the kernel command line

### 3 WORK LOG

#### Work.log for Chase Coltman

- 1) Began working on installing Kernel and attempting to get it run. Worked through instructions until I got to
  - (A) Steps Taken:
    - (a) Created the repo and gave permissions for Alec to be able to enter and make changes
    - (b) Attempted to move files to the correct location
  - (B) Date - 9/30/2017
  - (C) Time - About 2 hours
  - (D) End-point - Frustrated and not sure what I am doing wrong
  - (E) Finished - No, the Kernel had a bunch of error so I waited until next class to ask a peer
- 2) Attempted to work some more on getting Kernel to work
  - (A) Steps Taken:
    - (a) Moved all the files into the correct directory
    - (b) Seems to be working but not sure as it freezes when I run the code
  - (B) Date - 10/3/2017
  - (C) Time - About 2 hours
  - (D) End-point - Compiles and runs but freezes at the end
  - (E) Finished - No, Kernel is compiling but not sure on next step
- 3) Completed all the requirements for the Kernel to run
  - (A) Steps Taken:
    - (a) Finally got it working. Everything was actually working on the 3rd when I attempted to do it before, just had to open another shell client.
    - (b) Finished the implementation and did the write-up with all the steps that I took to figure it out
  - (B) Date - 10/5/2017
  - (C) Time - About 3-4 hours
  - (D) End-point - Runs and I am able to enter the kernel and VM with both the bzImage file and the one we make when we run make -j4 all
  - (E) Finished - Yes, It worked and I followed all the steps to make sure

4) Finished writing up what all the flags do

(A) Steps Taken:

(a) Searched for qemu i386 flags to give me a detailed readme

(b) Found this link: <http://www.tin.org/bin/man.cgi?section=1&topic=qemu-system-i386> \item Wrote in what each flag did

(c) Wrote in what each flag did and learned what exactly we do when we run that long command

(B) Date - 10/7/2017

(C) Time - About 1 hour

(D) End-point - Finished

(E) Finished - Yes, Reason for it being approximately a time is because I was at work and actually have no idea how long it took

5) Learning how to LaTeX

(A) Steps Taken:

(a) Watched the online tutorial

(b) Copy-pasted write up from Google Doc and formatted for LaTeX

(B) Date - 10/9/2017

(C) Time - About 2 hours

(D) End-point - Finished on my end, sending to Alec for his log and final product

(E) Finished - No