

gitlog.tex)

Project 1 and Conquerency 1

by Sinan Topkaya

CS 444 - Spring 2016

Abstract: This paper includes information on how to build and work on kernel plus information on choices I made making a multi threaded consumer producer program.

WRITE-UP

Project 1

Log of commands for Project1:

- git init
- git clone git://gityoctoproject.org/linux-yocto-3.14
- git checkout -b v3.14.26
- git tag -l
- cp .config cp sb-sdk-qemux.ex3
- source /scratch/opt/environment-setup-i586-linux
- make -j4 all
- qemu-system-i386 -gdb tcp::5630 -S -nographic -kernel linux-yocto-3.14/arch/x86/boot/bzImage -drive file=core-image-lsb-sdk-qemux86.ext3,if=virtio -enable-kvm -net none -usb -localtime -no-reboot -append "root=/dev/vda rw console=ttyS0 debug"
- opened another putty session
- gdb
- target remote localhost:5630
- c
- uname -r

Qemu Command and its flags:

- The code: qemu-system-i386 -gdb tcp::5630 -S -nographic -kernel linux-yocto-3.14/arch/x86/boot/bzImage -drive file=core-image-lsb-sdk-qemux86.ext3,if=virtio -enable-kvm -net none -usb -localtime -no-reboot -append "root=/dev/vda rw console=ttyS0 debug"
- -gdb tcp::5630: This flag allows us to debug the program using gdb, we have to connect to localhost 5630 to see what is exactly going on with the running program
- -S: This option makes qemu launch wait till gdb connection.
- -nographic: Disables graphical output so that QEMU is a simple command line application.
- -kernel: This option is needed to provide the Linux kernel image.
- -drive: defines the drive.
- -if=virtio: specifies the controller's PCI.

- -enable-kvm: Enables KVM full virtualization support.
- -net none: Indicates that no network devices should be configured.
- -usb: Enables USB drivers.
- -localtime: Set RTC start at local time.
- -no-reboot: Exit instead of rebooting.
- -append: is used to give the kernel command line argument.

Concurrency 1

Write-up of Concurrency solution: My implementation of the concurrency problem was done in C language. I have used Mersenne Twister to generate my random numbers. The problem has some constraints such as, while an item is being added or removed from the buffer, the buffer must be in an inconsistent state, I have achieved this through locking my the buffer for the specific thread that has to use it. Also if a consumer thread arrives while buffer is empty, it has to clock until producer adds a new item, I have achieved this by creating a checkempty funtion, this function check if the buffer is empty, and initializes it to 1 or in other words true if so. Another constraint was that a producer thread should be able to put more items if the buffer is full, I have achieved this by using the same function but in !checkempty format. I have used getrandinit32 funtion for creating my random numbers.

Reflection:

1. I think the main point was to learn how to use program multi-threaded programs, this assignment refreashed my memory working with them.
2. Since I did not remember too much, I read and watched few videos on producer-consumer problems. Also I approached the assignment piece by piece I created threads then worked on producer first, once I was sure the thread was doing what it was supposed to do, I continued working on the consumer.
3. Approaching the problem piece by piece made it much easier. As mentioned above I made sure the producer thread worked right before I started working on the consumer, I have also ran the program multiple times and waited if I am generating correct random numbers or not.
4. Well I could say I have learned alot about threads. It was actually a really fun assignment to learn more about threads and how to handle them, also thinking about error it might have was really interesting like blocking signals.

git log

work log

When	What
4/4/2016	Started with Project 1, connected to the os-class and ran the already given bzImage, got familiar with git and build my own version control system.
4/5/2016	Built kernel using git because last time I could not. Started with latex, but end up removing it.
4/6/2016	Everything finally worked well with qemu and Project 1 is completely finished.
4/7/2016	Forgot to commit changes. I started with implementing threads and tested if they were succesfully created
4/8/2016	Created the producer thread and function, it works fine and started working on the cpnsumer
4/9/2016	I am having troubles deleting and I will start working on the consumer
4/10/2016	Worked on Weekly Summaries
4/11/2016	Working one th consumer thread, the program runs well but needs few fixes.