Course Name: Embedded Systems II

Course Number and Section: **14:332:493:10**

Experiment: Lab 8

Instructor: Professor Phillip Southard

Date Submitted: 11/19/19

Submitted by: Christopher Cordi 182002852

Purpose

Lab 8 part 1-The first part of this lab is to learn how to create, develop, and build our own application. This application will be built on the host computer using the petalinux that we will install from the usb provided. The user application will be a simple created app utilizing the PetaLinux tools. We will also build the new user application by cross-compiling into the embedded linux.

Lab 8 part 2- The second part of this lab is to run and debug the application that we built in the prior step. The first step of this part is to run the application to see the output, understanding what it starts out with. Next we will debug the application using the system debugger. To finish this part out we will also be customizing the application following the steps from the lab manual.

Theory of Operation

This lab will be most similar to our last weeks lab where we loaded a form of Linux onto the desktop computers and creating a port to the zybo. To start making the application we will first have to decide whether we want to use C or C++. Once the app is made we will need to enable it and have the build debug-able, we will also be enabling the TCF agent. After everything is enabled wwe will be building the image and running the application on QEMU. Next is to debug using the XSDK, and making sure we have the proper BOOT.BIN file on the SD card. We will run the DHCPserver on the host, power on the board, and boot the new linux image on the board. Next, is to create a new debug configuration and configure the setup of the target. Once we have taken a look at the process of debugging we will be changing the application and rerunning it to see the changes.

When the program is correct and fully loaded we will be able to see that the proper function is achieved when application output appears on our serial monitor (pictured at the end). We will be seeing the step-through process of debugging and the ability of changing the application.

Test

Similar to last lab, we need to first insert the thumb drive into the desktop and restart the computer in order to change the boot op sequence. When doing so we will be led into a series of prompts to finish opening up the ubuntu linux, the screen will appear to resemble that of a raspberry pi. In the simple screen we will need to open up its form of terminal, once booted up we can go to the next phase. Hooking up the zybo is the next step. We have to change the pin combination on the JP5 from JTAG to the MicroSD, then plug the zybo up to the computer. We will then need to login and power the board, and then following given commands we will be able to set up the read/write access, and select the serial port terminal application from the desktop. Once done we will have to reset the board. At this point we want to stop the boot up sequence. We will need to follow the prompts on the terminal from the desktop in order to do the next parts.

Next is to generate our own application by first fixing the path to the proper project directory. We will have to then use the petalinux-create command to help kick start creating the new application. Once we have created a new application, the next step for us is to compile and build it.We will also need to enable the build debuggable applications and also build the image. Once built we will run the application that was made in QEMU in order to see the output of our creation.

To finish the lab we need to take the image file we created and run it, once we see the output we will then use the XSDK to debug our application. Following the steps and linking the ports to and from the ZYBO we will be able to implement all that is needed to ensure everything works aka the debugger. Once we have played around with that we move on to changing the application to learn a new tool. Using steps similar to creating the application we will be opening and editing the application, adding new code for a changed output. Once finished our application will be run using the QEMU once again to observe the output.

Principal IP Integrator Block Diagram- Vivado was not used for this experiment

Design code- No design code for this week's lab, everything provided in micro sd and USB thumb drive.

Implementation - Vivado was not used for this experiment

Implementation- no implementation needed

Vivado Elaboration Schematic - Screenshot- not needed for this lab

Vivado Synthesis Schematic- not needed for this lab

Vivado Project Summary Images- not needed for this lab

Discussion -No Lab questions were found in any of the exercises included in this lab.

Observations / Discoveries-

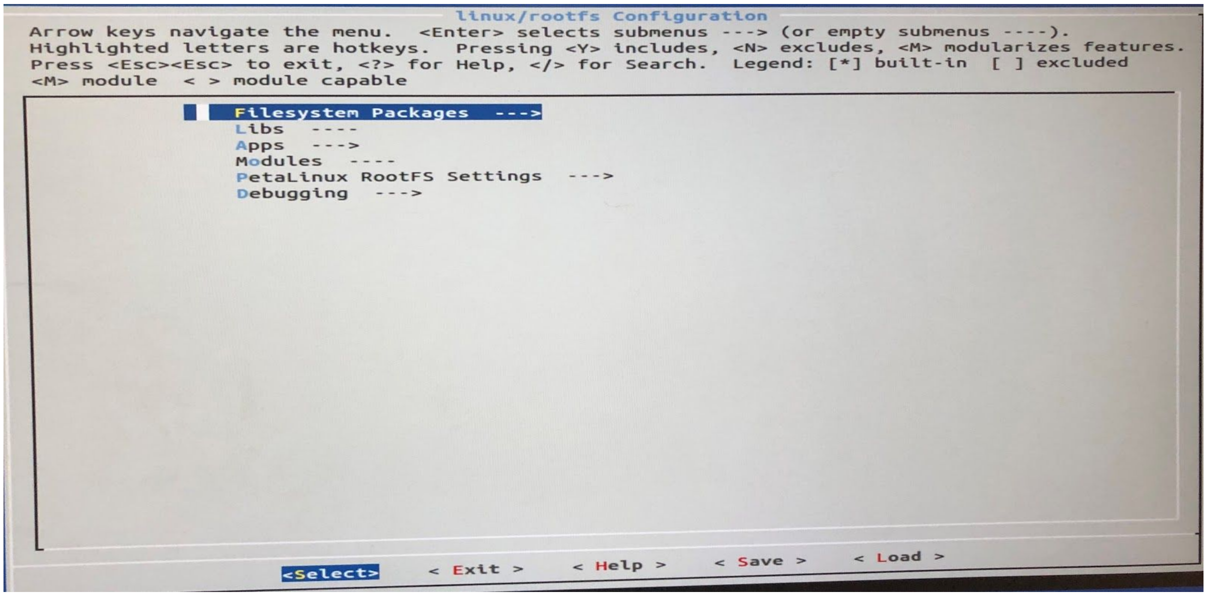
This lab was to further dive into work we have done in the past lab when we ran a form of embedded linux on the lab rooms desktop and pinging data between the desktop and the zybos. This experiment utilized the petalinux we were using to generate a form of linux to be used on the ZYBOs. We were able to extract needed information from the USB and generating the proper files, while also creating an application that would be loaded onto the board. Once everything was created and loaded up we can see that the application that we created is displayed on the terminal, which is what would be the resulting execution from our embedded program. We are also able to use the debugger to check functionality and check for any errors more easily than just by reading through the code. The debugger is a very useful tool to implement when hitting an obstacle creating applications.

Questions / Follow Up-

This lab was definitely more interesting than the last one, dealing with an application that we built and generated on our desktop version of linux that could be loaded onto the zybo board. Building on from what we did with the last lab we built an application that we were able to run and observe the output. After, we can simply debug it to find errors within our code. Lastly, we can change and edit that application to how we want. These tools we learned are going to be quite helpful later on in class.

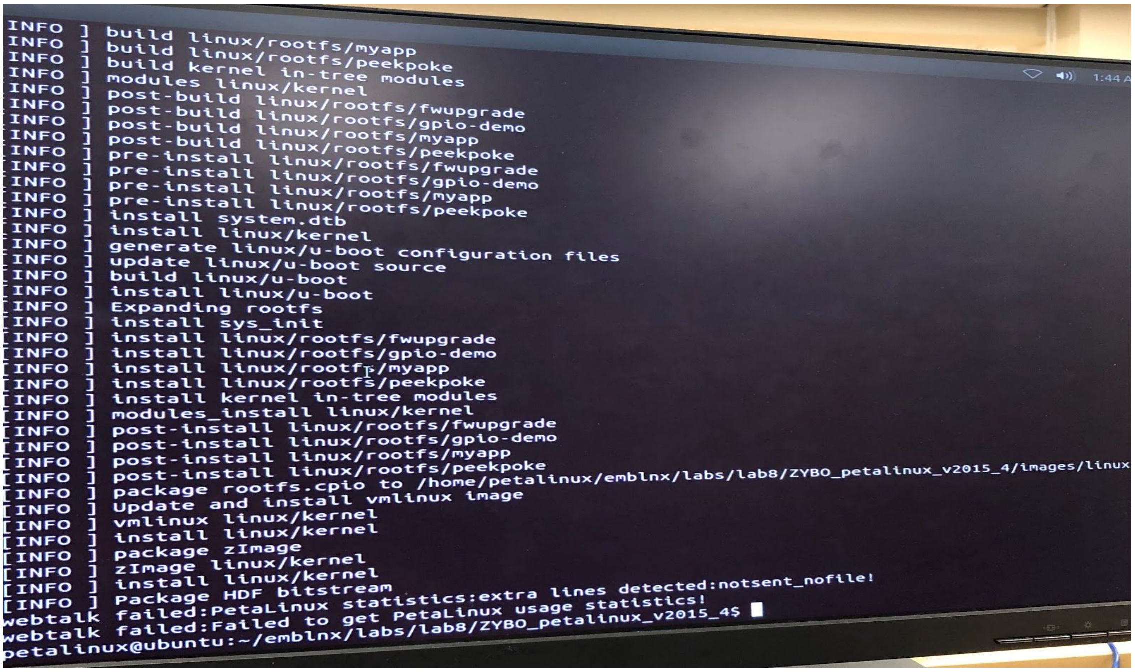
There was not much trouble resulting from this lab in fact it was one of the most interesting, the only thing that could have made this lab more fulfilling would be figuring out how to implement more actions within the application. Maybe including a file that we can run as a pre built function that was more complex than a regular hello world. Over all very interesting.

Going into filesystem to enable debugger



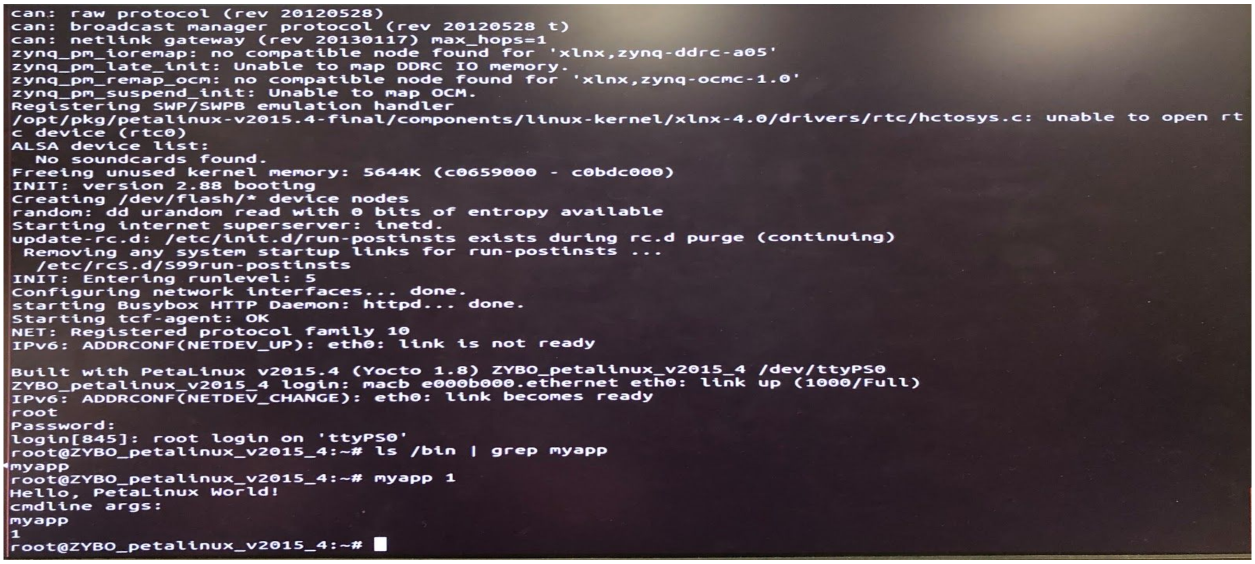
```
                         linux/rootfs Configuration
 Arrow keys navigate the menu.  <Enter> selects submenus ---> (or empty submenus ----).
 Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes, <M> modularizes features.
 Press <Esc><Esc> to exit, <?> for Help, </> for Search.  Legend: [*] built-in  [ ] excluded
 <M> module  < > module capable

                 Filesystem Packages  --->
                 Libs  ----
                 Apps  --->
                 Modules  ----
                 PetaLinux RootFS Settings  --->
                 Debugging  --->

            <Select>    < Exit >    < Help >    < Save >    < Load >
```

Creating the Application

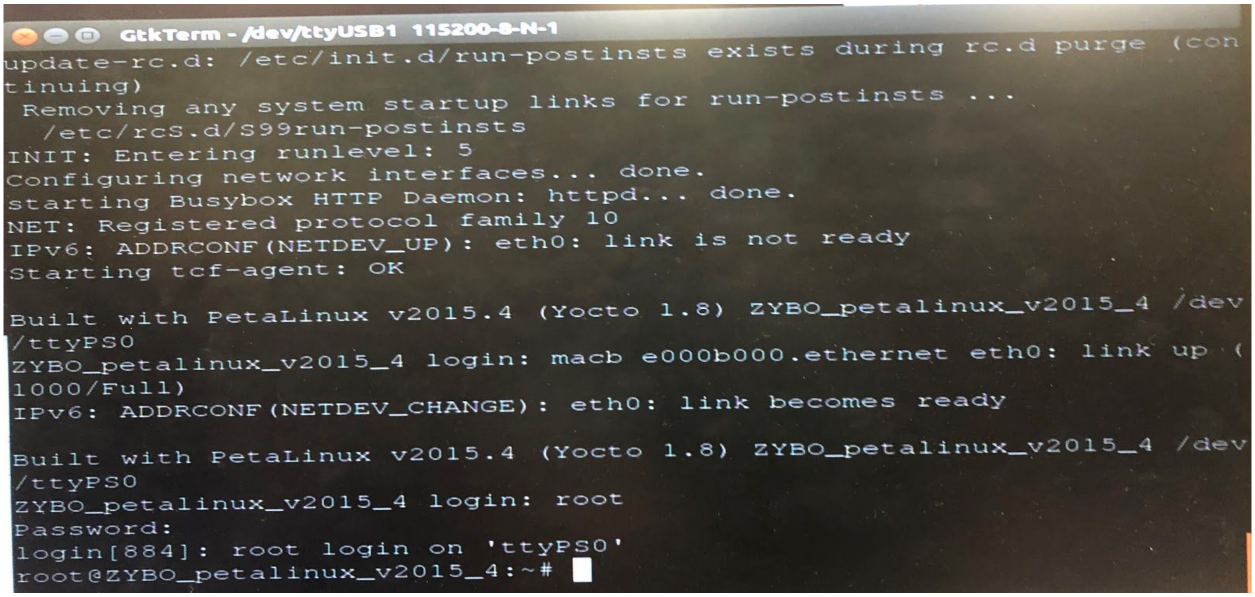


```
INFO ] build linux/rootfs/myapp
INFO ] build linux/rootfs/peekpoke
INFO ] build kernel in-tree modules
INFO ] modules linux/kernel
INFO ] post-build linux/rootfs/fwupgrade
INFO ] post-build linux/rootfs/gpio-demo
INFO ] post-build linux/rootfs/myapp
INFO ] post-build linux/rootfs/peekpoke
INFO ] pre-install linux/rootfs/fwupgrade
INFO ] pre-install linux/rootfs/gpio-demo
INFO ] pre-install linux/rootfs/myapp
INFO ] pre-install linux/rootfs/peekpoke
INFO ] install system.dtb
INFO ] install linux/kernel
INFO ] generate linux/u-boot configuration files
INFO ] update linux/u-boot source
INFO ] build linux/u-boot
INFO ] install linux/u-boot
INFO ] Expanding rootfs
INFO ] install sys_init
INFO ] install linux/rootfs/fwupgrade
[INFO ] install linux/rootfs/gpio-demo
[INFO ] install linux/rootfs/myapp
[INFO ] install linux/rootfs/peekpoke
[INFO ] install kernel in-tree modules
[INFO ] modules_install linux/kernel
[INFO ] post-install linux/rootfs/fwupgrade
[INFO ] post-install linux/rootfs/gpio-demo
[INFO ] post-install linux/rootfs/myapp
[INFO ] post-install linux/rootfs/peekpoke
[INFO ] package rootfs.cpio to /home/petalinux/emblnx/labs/lab8/ZYBO_petalinux_v2015_4/images/linux
[INFO ] Update and install vmlinux image
[INFO ] vmlinux linux/kernel
[INFO ] install linux/kernel
[INFO ] package zImage
[INFO ] zImage linux/kernel
[INFO ] install linux/kernel
[INFO ] Package HDF bitstream
webtalk failed:PetaLinux statistics:extra lines detected:notsent_nofile!
webtalk failed:Failed to get PetaLinux usage statistics!
petalinux@ubuntu:~/emblnx/labs/lab8/ZYBO_petalinux_v2015_4$
```

Running the application

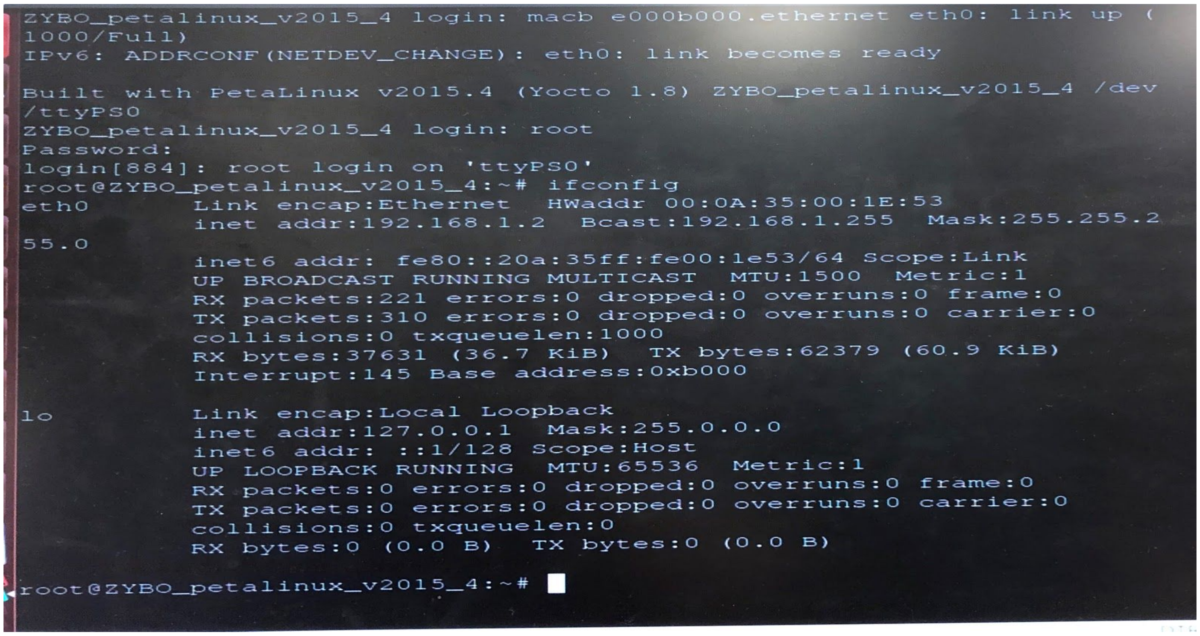


```
can: raw protocol (rev 20120528)
can: broadcast manager protocol (rev 20120528 t)
can: netlink gateway (rev 20130117) max_hops=1
zynq_pm_ioremap: no compatible node found for 'xlnx,zynq-ddrc-a05'
zynq_pm_late_init: Unable to map DDRC IO memory.
zynq_pm_remap_ocm: no compatible node found for 'xlnx,zynq-ocmc-1.0'
zynq_pm_suspend_init: Unable to map OCM.
Registering SWP/SWPB emulation handler
/opt/pkg/petalinux-v2015.4-final/components/linux-kernel/xlnx-4.0/drivers/rtc/hctosys.c: unable to open rt
c device (rtc0)
ALSA device list:
  No soundcards found.
Freeing unused kernel memory: 5644K (c0659000 - c0bdc000)
INIT: version 2.88 booting
Creating /dev/flash/* device nodes
random: dd urandom read with 0 bits of entropy available
Starting internet superserver: inetd.
update-rc.d: /etc/init.d/run-postinsts exists during rc.d purge (continuing)
 Removing any system startup links for run-postinsts ...
  /etc/rcS.d/S99run-postinsts
INIT: Entering runlevel: 5
Configuring network interfaces... done.
starting Busybox HTTP Daemon: httpd... done.
Starting tcf-agent: OK
NET: Registered protocol family 10
IPv6: ADDRCONF(NETDEV_UP): eth0: link is not ready

Built with PetaLinux v2015.4 (Yocto 1.8) ZYBO_petalinux_v2015_4 /dev/ttyPS0
ZYBO_petalinux_v2015_4 login: macb e000b000.ethernet eth0: link up (1000/Full)
IPv6: ADDRCONF(NETDEV_CHANGE): eth0: link becomes ready
root
Password:
login[845]: root login on 'ttyPS0'
root@ZYBO_petalinux_v2015_4:~# ls /bin | grep myapp
myapp
root@ZYBO_petalinux_v2015_4:~# myapp 1
Hello, PetaLinux World!
cmdline args:
myapp
1
root@ZYBO_petalinux_v2015_4:~# █
```

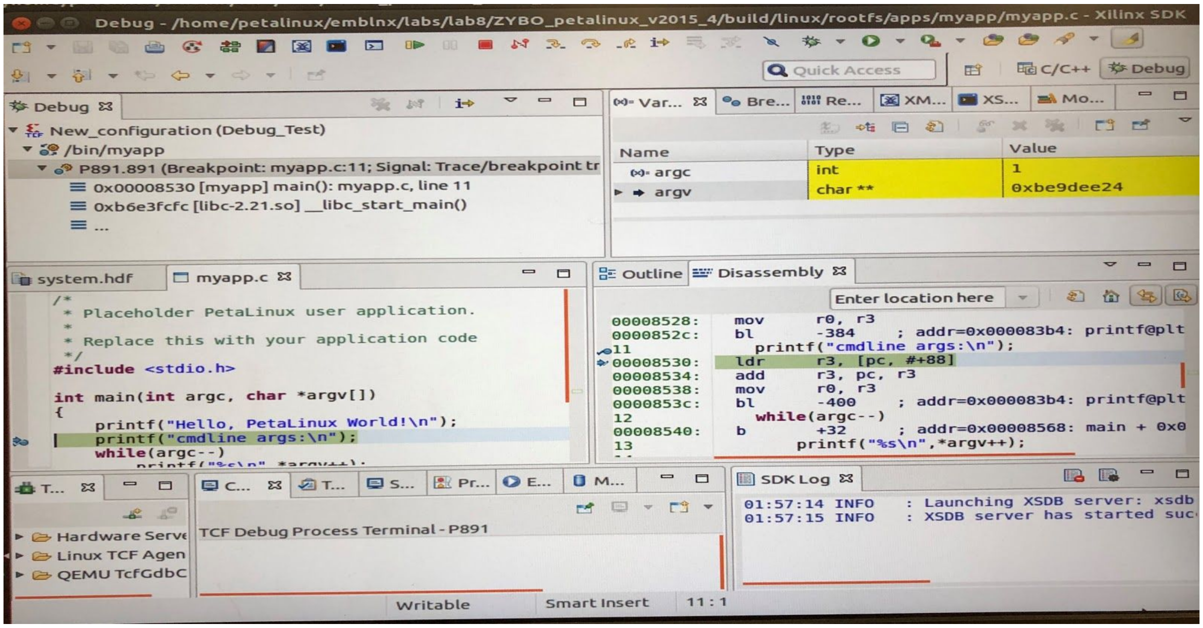


```
update-rc.d: /etc/init.d/run-postinsts exists during rc.d purge (con
tinuing)
 Removing any system startup links for run-postinsts ...
  /etc/rcS.d/S99run-postinsts
INIT: Entering runlevel: 5
Configuring network interfaces... done.
starting Busybox HTTP Daemon: httpd... done.
NET: Registered protocol family 10
IPv6: ADDRCONF(NETDEV_UP): eth0: link is not ready
Starting tcf-agent: OK

Built with PetaLinux v2015.4 (Yocto 1.8) ZYBO_petalinux_v2015_4 /dev
/ttyPS0
ZYBO_petalinux_v2015_4 login: macb e000b000.ethernet eth0: link up (
1000/Full)
IPv6: ADDRCONF(NETDEV_CHANGE): eth0: link becomes ready

Built with PetaLinux v2015.4 (Yocto 1.8) ZYBO_petalinux_v2015_4 /dev
/ttyPS0
ZYBO_petalinux_v2015_4 login: root
Password:
login[884]: root login on 'ttyPS0'
root@ZYBO_petalinux_v2015_4:~# █
```

Running Debugger



```
ZYBO_petalinux_v2015_4 login: macb e000b000.ethernet eth0: link up (
1000/Full)
IPv6: ADDRCONF(NETDEV_CHANGE): eth0: link becomes ready

Built with PetaLinux v2015.4 (Yocto 1.8) ZYBO_petalinux_v2015_4 /dev
/ttyPS0
ZYBO_petalinux_v2015_4 login: root
Password:
login[884]: root login on 'ttyPS0'
root@ZYBO_petalinux_v2015_4:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0A:35:00:1E:53
          inet addr:192.168.1.2  Bcast:192.168.1.255  Mask:255.255.2
55.0
          inet6 addr: fe80::20a:35ff:fe00:1e53/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:221 errors:0 dropped:0 overruns:0 frame:0
          TX packets:310 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:37631 (36.7 KiB)  TX bytes:62379 (60.9 KiB)
          Interrupt:145 Base address:0xb000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

root@ZYBO_petalinux_v2015_4:~# 
```



```
New_configuration (Debug_Test)
  /bin/myapp
    P891.891 (Breakpoint: myapp.c:11; Signal: Trace/breakpoint tr
      0x00008530 [myapp] main(): myapp.c, line 11
      0xb6e3fcfc [libc-2.21.so] __libc_start_main()
      ...
```

Name	Type	Value
argc	int	1
argv	char **	0xbe9dee24

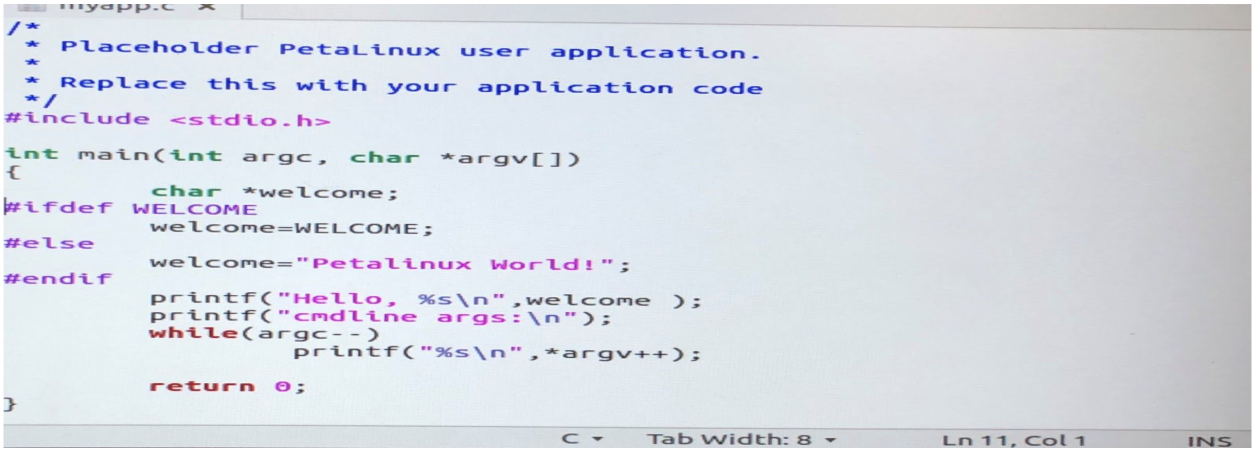
```
/*
 * Placeholder PetaLinux user application.
 *
 * Replace this with your application code
 */
#include <stdio.h>

int main(int argc, char *argv[])
{
    printf("Hello, PetaLinux World!\n");
    printf("cmdline args:\n");
    while(argc--)
        printf("%s\n",*argv++);
```

```
00008528:   mov     r0, r3
0000852c:   bl      -384      ; addr=0x000083b4: printf@plt
                printf("cmdline args:\n");
11
00008530:   ldr     r3, [pc, #+88]
00008534:   add     r3, pc, r3
00008538:   mov     r0, r3
0000853c:   bl      -400      ; addr=0x000083b4: printf@plt
                while(argc--)
12
00008540:   b       +32       ; addr=0x00008568: main + 0x0
                    printf("%s\n",*argv++);
13
```

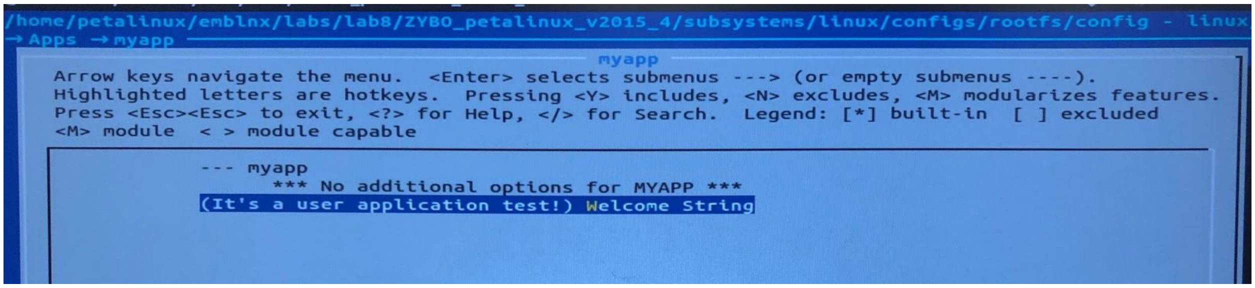
TCF Debug Process Terminal - P891

```
01:57:14 INFO   : Launching XSDB server: xsdb
01:57:15 INFO   : XSDB server has started suc
```

Fixed application

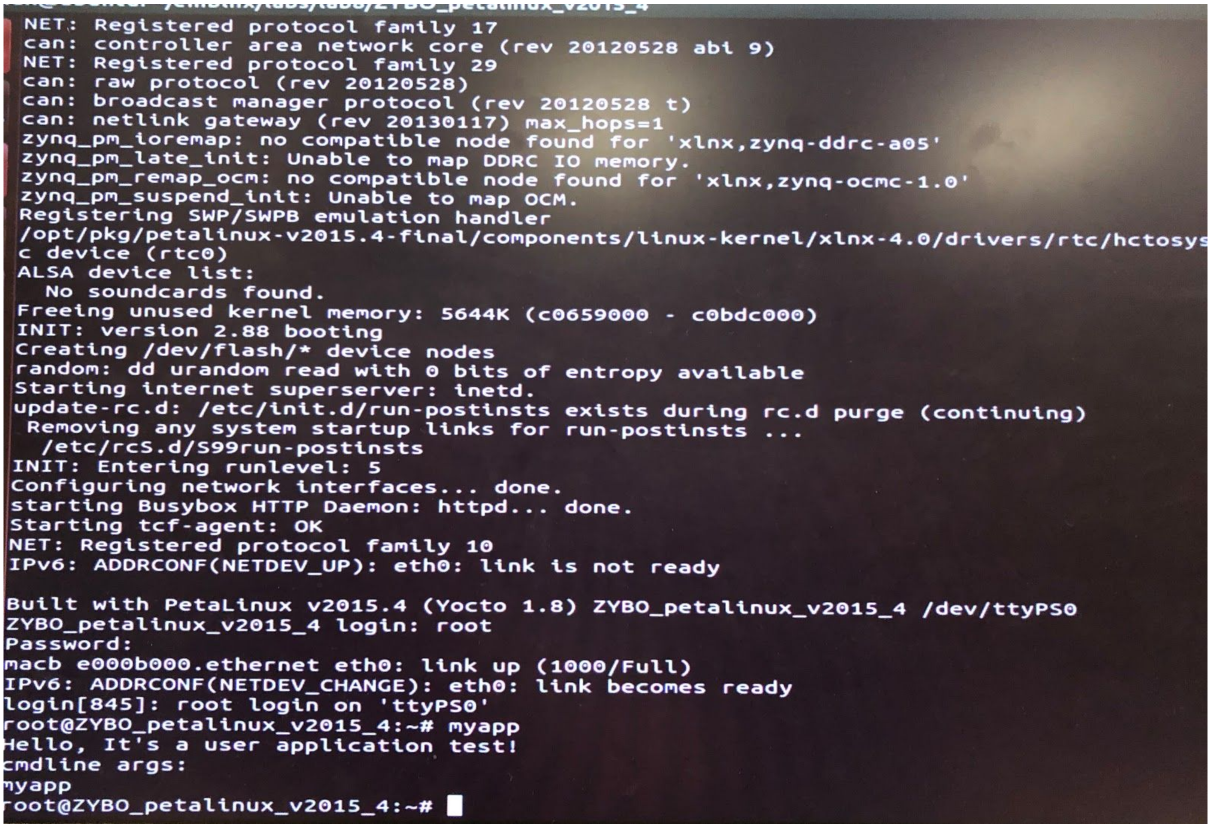


```
/*
 * Placeholder PetaLinux user application.
 *
 * Replace this with your application code
 */
#include <stdio.h>

int main(int argc, char *argv[])
{
        char *welcome;
#ifdef WELCOME
        welcome=WELCOME;
#else
        welcome="Petalinux World!";
#endif
        printf("Hello, %s\n",welcome );
        printf("cmdline args:\n");
        while(argc--)
                printf("%s\n",*argv++);

        return 0;
}
```



```
/home/petalinux/emblnx/labs/lab8/ZYBO_petalinux_v2015_4/subsystems/linux/configs/rootfs/config - linux
→ Apps → myapp
                                   myapp
  Arrow keys navigate the menu.  <Enter> selects submenus ---> (or empty submenus ----).
  Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes, <M> modularizes features.
  Press <Esc><Esc> to exit, <?> for Help, </> for Search.  Legend: [*] built-in  [ ] excluded
  <M> module  < > module capable

                --- myapp
                      *** No additional options for MYAPP ***
                (It's a user application test!) Welcome String
```

Running new application



```
NET: Registered protocol family 17
can: controller area network core (rev 20120528 abi 9)
NET: Registered protocol family 29
can: raw protocol (rev 20120528)
can: broadcast manager protocol (rev 20120528 t)
can: netlink gateway (rev 20130117) max_hops=1
zynq_pm_ioremap: no compatible node found for 'xlnx,zynq-ddrc-a05'
zynq_pm_late_init: Unable to map DDRC IO memory.
zynq_pm_remap_ocm: no compatible node found for 'xlnx,zynq-ocmc-1.0'
zynq_pm_suspend_init: Unable to map OCM.
Registering SWP/SWPB emulation handler
/opt/pkg/petalinux-v2015.4-final/components/linux-kernel/xlnx-4.0/drivers/rtc/hctosys
c device (rtc0)
ALSA device list:
  No soundcards found.
Freeing unused kernel memory: 5644K (c0659000 - c0bdc000)
INIT: version 2.88 booting
Creating /dev/flash/* device nodes
random: dd urandom read with 0 bits of entropy available
Starting internet superserver: inetd.
update-rc.d: /etc/init.d/run-postinsts exists during rc.d purge (continuing)
 Removing any system startup links for run-postinsts ...
  /etc/rcS.d/S99run-postinsts
INIT: Entering runlevel: 5
Configuring network interfaces... done.
Starting Busybox HTTP Daemon: httpd... done.
Starting tcf-agent: OK
NET: Registered protocol family 10
IPv6: ADDRCONF(NETDEV_UP): eth0: link is not ready

Built with PetaLinux v2015.4 (Yocto 1.8) ZYBO_petalinux_v2015_4 /dev/ttyPS0
ZYBO_petalinux_v2015_4 login: root
Password:
macb e000b000.ethernet eth0: link up (1000/Full)
IPv6: ADDRCONF(NETDEV_CHANGE): eth0: link becomes ready
login[845]: root login on 'ttyPS0'
root@ZYBO_petalinux_v2015_4:~# myapp
Hello, It's a user application test!
cmdline args:
myapp
root@ZYBO_petalinux_v2015_4:~# █
```