Course Name: Embedded Systems II

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

Experiment: Lab 08

Lab Instructor: Prof. Southard

Date Performed: 11/13/2019

Date Submitted: 11/20/2019

Submitted by: Justin Hinds

Electrical and Computer Engineering Department
School of Engineering
Rutgers University, Piscataway, NJ 08854

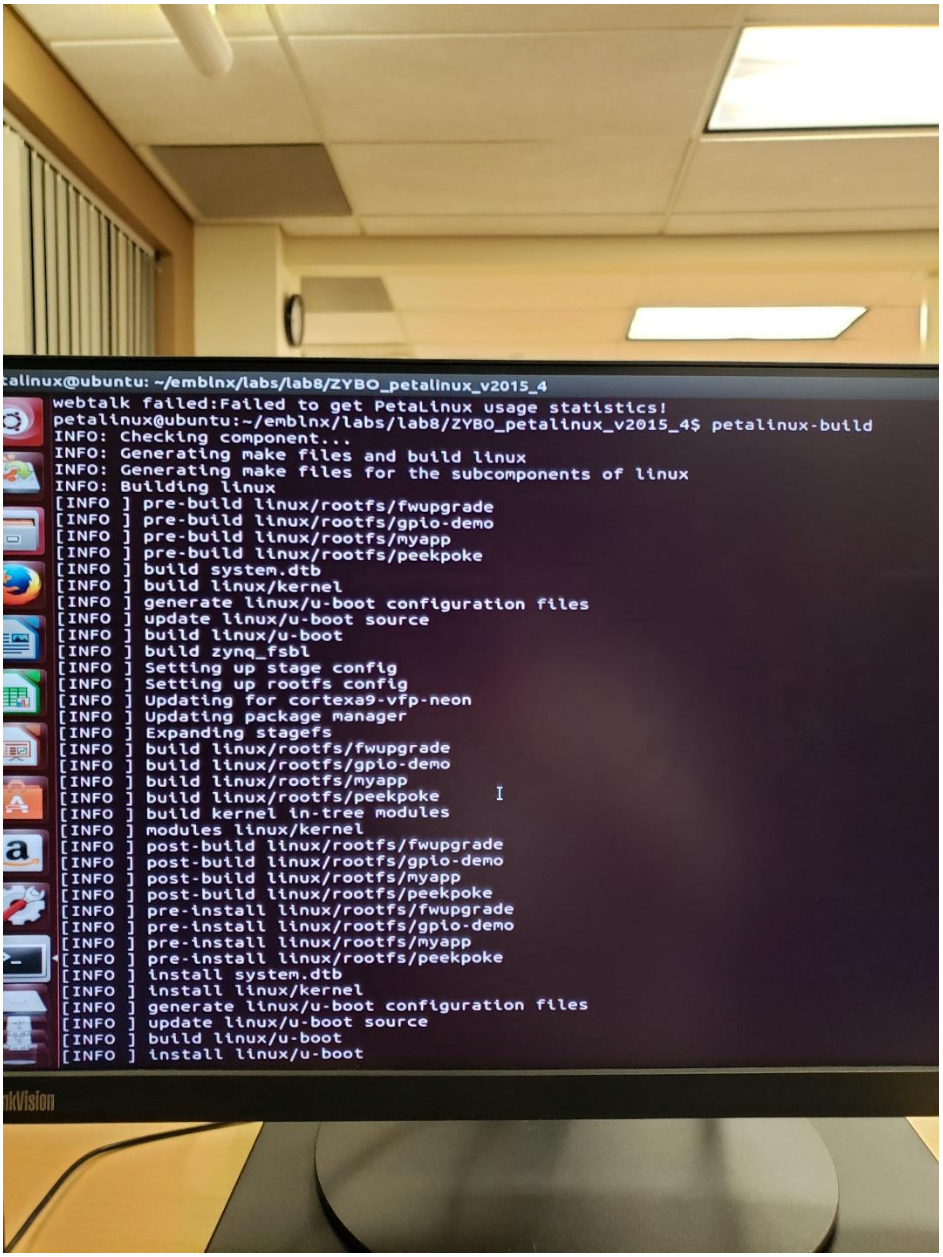
Purpose:

As our previous lab introduced us to building and booting Embedding Linux for an FPGA, we will expand upon the topic by developing our own application within the OS we built. We will compile the Linux kernel and its applications using a Linux PC and use that image on our FPGA. Upon doing we so we develop a simple hello world application to introduce us to the steps required to run your own custom application on an Embedded Linux machine.

Theory of Operation:

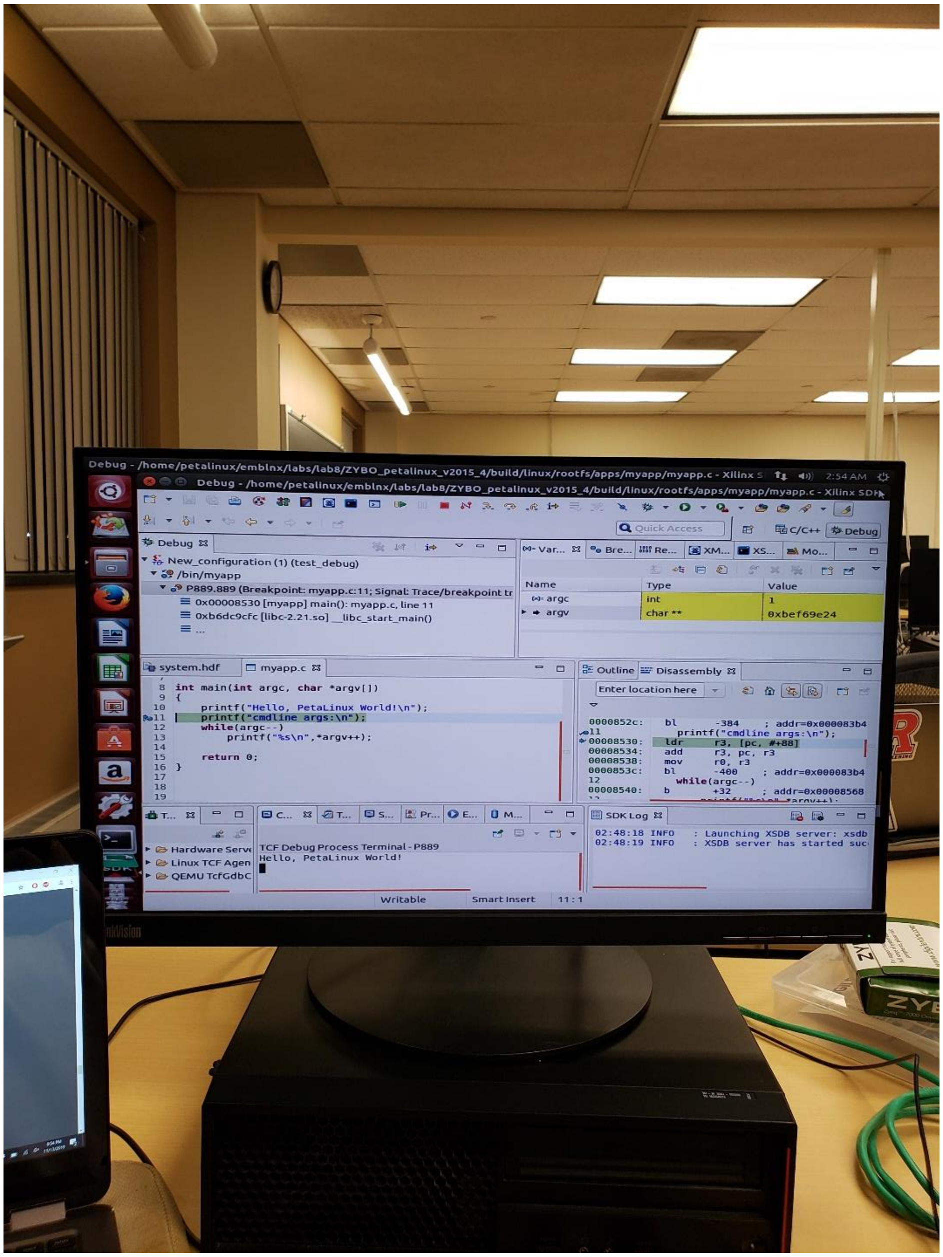
If we follow the steps correctly, we should be able to boot up PetaLinux via the ZYBO. Unlike the previous Lab, this build of PetaLinux will contain our own application. We will begin by creating a new PetaLinux project on the Linux PC and building it. Upon doing so, we can we can use the petalinux tools to develop our own application using the XSDK. After seeing that the application we built is fully functional we can modify the Makefile and the bootup process to launch our application. If we have followed the string will be displayed as we boot up our Petalinux device.

Building the Linux Image:



```
alinux@ubuntu: ~/emblnx/labs/lab8/ZYBO_petalinux_v2015_4
webtalk failed:Failed to get PetaLinux usage statistics!
petalinux@ubuntu:~/emblnx/labs/lab8/ZYBO_petalinux_v2015_4$ petalinux-build
INFO: Checking component...
INFO: Generating make files and build linux
INFO: Generating make files for the subcomponents of linux
INFO: Building linux
[INFO ] pre-build linux/rootfs/fwupgrade
[INFO ] pre-build linux/rootfs/gpio-demo
[INFO ] pre-build linux/rootfs/myapp
[INFO ] pre-build linux/rootfs/peekpoke
[INFO ] build system.dtb
[INFO ] build linux/kernel
[INFO ] generate linux/u-boot configuration files
[INFO ] update linux/u-boot source
[INFO ] build linux/u-boot
[INFO ] build zynq_fsbl
[INFO ] Setting up stage config
[INFO ] Setting up rootfs config
[INFO ] Updating for cortexa9-vfp-neon
[INFO ] Updating package manager
[INFO ] Expanding stagefs
[INFO ] build linux/rootfs/fwupgrade
[INFO ] build linux/rootfs/gpio-demo
[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
```

Developing in the XSDK:



Debug - /home/petalinux/emblnx/labs/lab8/ZYBO_petalinux_v2015_4/build/linux/rootfs/apps/myapp/myapp.c - Xilinx SDK

```
New_configuration (1) (test_debug)
  /bin/myapp
    P889.889 (Breakpoint: myapp.c:11; Signal: Trace/breakpoint tr
      0x00008530 [myapp] main(): myapp.c, line 11
      0xb6dc9cfc [libc-2.21.so] __libc_start_main()
      ...
```

Name	Type	Value
argc	int	1
argv	char **	0xbef69e24

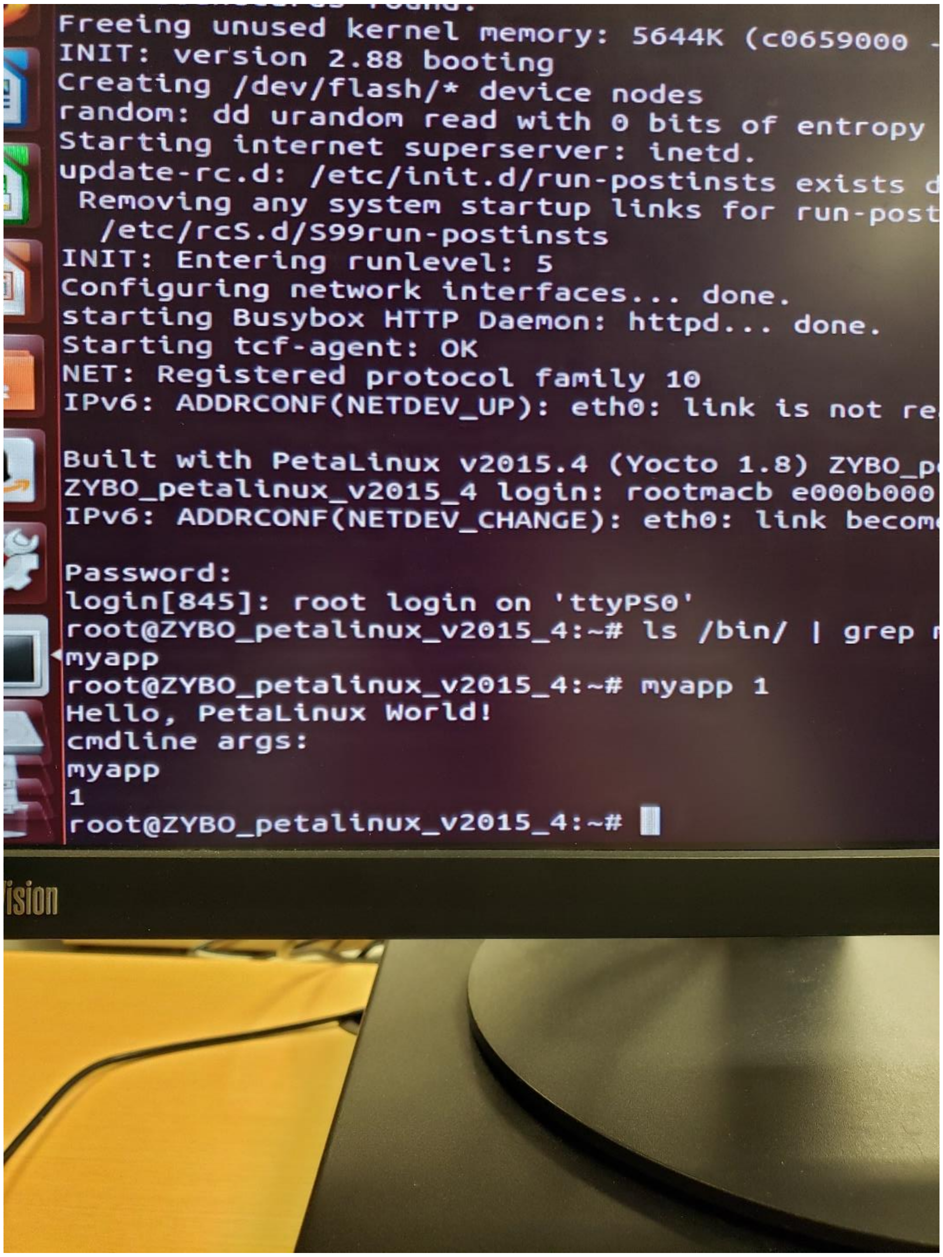
```c
int main(int argc, char *argv[])
{
    printf("Hello, PetaLinux World!\n");
    printf("cmdline args:\n");
    while(argc--)
        printf("%s\n",*argv++);

    return 0;
}
```

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

```
TCF Debug Process Terminal - P889
Hello, PetaLinux World!
```

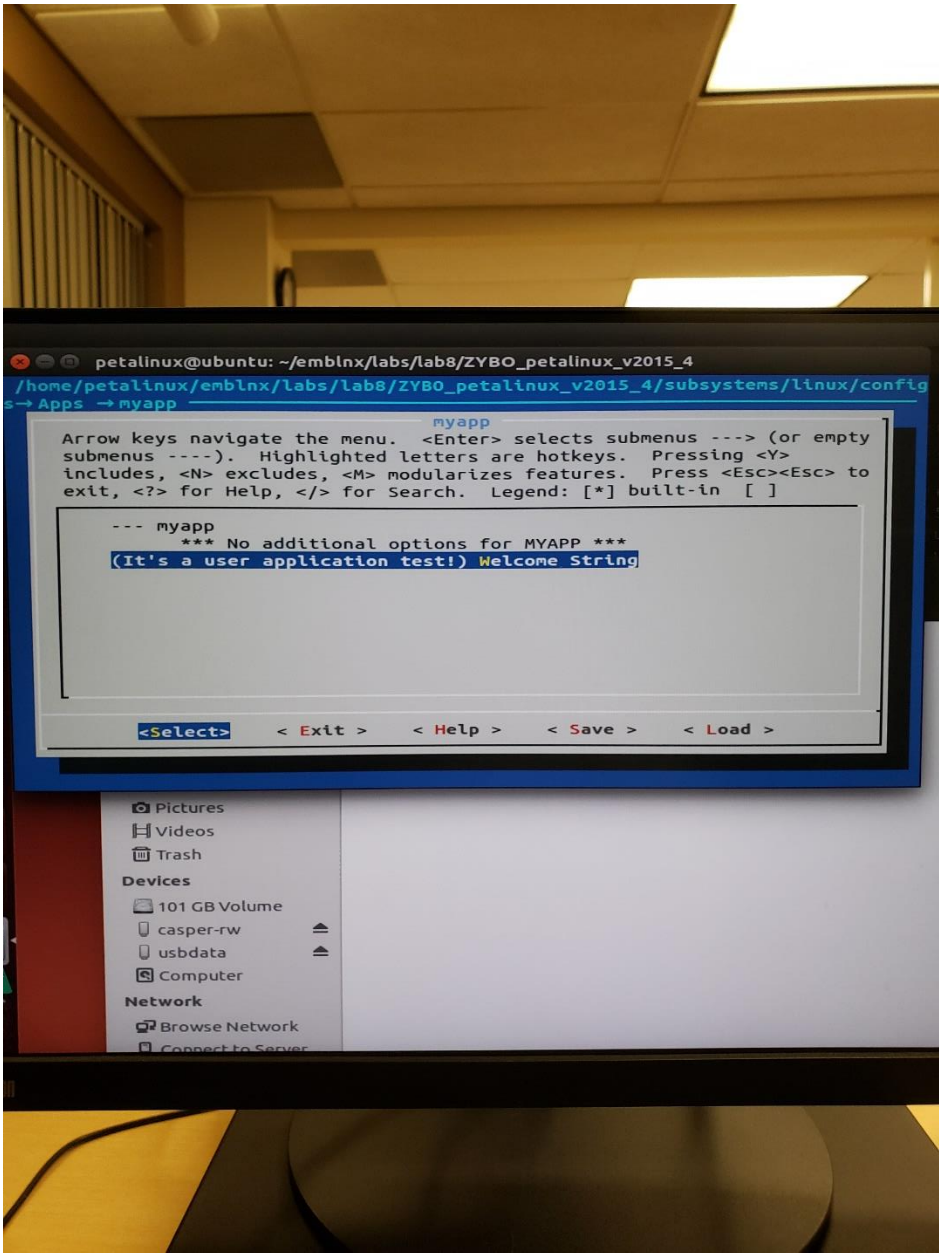
```
02:48:18 INFO   : Launching XSDB server: xsdb
02:48:19 INFO   : XSDB server has started suc
```

Running the application:



```
Freeing unused kernel memory: 5644K (c0659000 -
INIT: version 2.88 booting
Creating /dev/flash/* device nodes
random: dd urandom read with 0 bits of entropy
Starting internet superserver: inetd.
update-rc.d: /etc/init.d/run-postinsts exists d
 Removing any system startup links for run-post
  /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 re

Built with PetaLinux v2015.4 (Yocto 1.8) ZYBO_p
ZYBO_petalinux_v2015_4 login: rootmacb e000b000
IPv6: ADDRCONF(NETDEV_CHANGE): eth0: link becom

Password:
login[845]: root login on 'ttyPS0'
root@ZYBO_petalinux_v2015_4:~# ls /bin/ | grep m
myapp
root@ZYBO_petalinux_v2015_4:~# myapp 1
Hello, PetaLinux World!
cmdline args:
myapp
1
root@ZYBO_petalinux_v2015_4:~#
```

Changing the printed string:



```
petalinux@ubuntu: ~/emblnx/labs/lab8/ZYBO_petalinux_v2015_4
/home/petalinux/emblnx/labs/lab8/ZYBO_petalinux_v2015_4/subsystems/linux/config
s→ 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  [ ]

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

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

Test:

Demoed and Tested on board.

Code:

MyAPP:

```c
/*
 * 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;
}
```

Makefile:

```
ifndef PETALINUX

$(error "Error: PETALINUX environment variable not set.  Change to the root of your PetaLinux install, and source the settings.sh file")

endif



include apps.common.mk
include $(ROOTFS_CONFIG)

ifneq ($(CONFIG_APPS_MYAPP_WELCOME),)
CFLAGS += -DWELCOME=\"$(CONFIG_APPS_MYAPP_WELCOME)\"
endif

APP = myapp

# Add any other object files to this list below
APP_OBJS = myapp.o

all: build install

build: $(APP)

$(APP): $(APP_OBJS)
```

```
	$(CC) $(LDFLAGS) -o $@ $(APP_OBJS) $(LDLIBS)

clean:
	-rm -f $(APP) *.elf *.gdb *.o

.PHONY: install image

install: $(APP)
	$(TARGETINST) -d $(APP) /bin/$(APP)

%.o: %.c
	$(CC) -c $(CFLAGS) -o $@ $<

help:
	@echo ""
	@echo "Quick reference for various supported build targets for $(INSTANCE)."
	@echo "--------------------------------------------------"
	@echo "  clean		clean out build objects"
	@echo "  all		build $(INSTANCE) and install to rootfs host copy"
	@echo "  build		build subsystem"
	@echo "  install	install built objects to rootfs host copy"
```

KConfig:

```
#######################################################################
#########
# Please modify this file if you want to configure your own application.
# You can uncommon and/or change the following Kconfig elements.
#######################################################################
#########


if ROOTFS_COMPONENT_APPS_NAME_MYAPP
	comment "No additional options for MYAPP"
```

```
        config APPS_MYAPP_WELCOME

        string "Welcome String"

        help

        Welcome string for myapp



#       config APPS_MYAPP_OPTION0

#       bool "option0"

#       help

#         Help text

endif
```

Discussion:

- Observations / Discoveries:
  - This lab introduces us to building and running our own application in PetaLinux. I believe I will find the lab extremely useful in developing my final project. I hope the skills I learned in this lab will help me develop a more complex application.
- Questions / Follow-up:
  - I still would like to see a list of required files to submit on Github added to the assignment's module in canvas.
  - This lab answered one of the main questions I had from our previous lab. I thoroughly enjoyed completing this lab, I hope we build upon it.