Andrew Gates

Professor Matthew Tolentino

TCES 420

December 14, 2016

Project #4 – Missile Launcher Driver

Machine tested on: Raspberry Pi 3 with 1.2 GHz quad-core ARM Cortex-A53.

This project offered great experience in how to connect what we have learned so far about user-space programming to kernel-space programming to GPIO. Following the link provided on Canvas I completed all 3 walkthroughs in order to learn about how to write a Linux kernel module, a Linux loadable kernel module, and how to connect GPIO to a Linux loadable kernel module.

Taking what I learned from all 3 of these walkthroughs I was able to modify the code used in part 3 in order to interface the cannon and the cannon trigger to my LKM. I used a pullup resistor with my cannon trigger in order to differentiate between HIGH and LOW signals. I also used a transistor to allow voltage through to the cannon when the gpioCannon pin was set HIGH. I setup my LKM so that the 4 different functions that were required were included –

FIRE\_ONE: This function sets the cannon to HIGH and run the cannon until the cannon trigger outputs a LOW signal which means that a missile was fired.

FIRE\_ALL: A similar method was used but I repeated this process until the missileCount variable was 0.

NR\_MISSLES\_REMAINING: This function allows the user to call this function to see the current value of missileCount.

SET\_NR\_MISSLES: This function allows the user to echo in the value that they want to set the missileCount variable to.

My makefile when called will build my AGatesCannon code. Then once this module is installed to the kernel it can be found under /sys/Cannon/gpio26, which will list all 4 of my functions.