# Práctica No.1. GNU/Linux y Microcontrolador

1st Juan Carlos Garduño Gutiérrez 157302
*Ingeniería en Telecomunicaciones*
*Instituto Tecnológico Autónomo de México*
jgarduo6@itam.mx

2nd Humberto Martínez Barrón y Robles 166056
*Ingeniería en Mecatónica*
*Instituto Tecnológico Autónomo de México*
hmartin3@itam.mx

3rd Sebastián Aranda Bassegoda 157465
*Ingeniería en Mecatónica*
*Instituto Tecnológico Autónomo de México*
sarandab3@itam.mx

***Resumen*—En la primer práctica del laboratorio de principios de mecatrónica, se dio una introducción a los comandos de línea en Bash, uso de control de versión por medio de $Git$ y tres ejercicios de interacción con una tarjeta Arduino Mega: cargar un ejemplo llamado $Blink$, crear un sensor luminoso y mostrar un número hexadecimal en un $Display$ a 7 segmentos. La poca familiaridad de los integrantes del equipo dio pie a pequeños problemas resultantes de una curva de aprendizaje. Se resolvieron estos problemas para llegar a una solución a cada problema que garantiza la funcionalidad especificada.**

***Index Terms*—LED, Arduino, resistencia, fotoresistencia, $Bash$, $Display$ a 7 segmentos.**

## I. Introducción

Los sistemas operativos tipo Unix (y, en particular, Linux) son de los sistemas operativos más utilizados y el tipo de sistema operativo preferido por la mayoría de los desarrolladores de Software y de robótica. Una razón importante de la preferencia generalizada por este tipo de sistemas operativos es la facilidad con la que se pueden utilizar comandos en la herramienta conocida como "terminal". Estos comandos, escritos en un lenguaje conocido como "Bash", son conocidos y utilizados por ingenieros, programadores y desarrolladores de todo el mundo. Por lo tanto, resulta no sólo conveniente, sino importante que un ingeniero que desarrolla sistemas embebidos sepa acerca del uso adecuado de esta herramienta.

Las tarjetas Arduino (y sus microcontroladores AVR) son una herramienta utilizada por una amplia gama de desarrolladores de sistemas embebidos. Al permitir una interacción entre el mundo exterior y el dispositivo, resultan muy útiles para aplicaciones en campos de robótica y otros sistemas que requieran de interacción con el mundo exterior por parte de una computadora. Por lo tanto, su uso correcto es de suma importancia para ingenieros especializados en sistemas embebidos.

Por lo tanto, en este trabajo se tienen dos objetivos: aprender a utilizar algunos comandos esenciales en la terminal de Linux/Unix y aprender a desarrollar algunas aplicaciones básicas para una tarjeta Arduino. En el caso de los comandos, se generó un árbol de archivos y directorios que se especificó en las instrucciones de la práctica. En el caso del uso del microcontrolador, se programaron tres ejemplos como introducción al microcontrolador:

1. Cargar el ejemplo *Blink* y asegurarse de que funcione.
2. Lograr, mediante una fotoresistencia y un LED, que el alumbrado responda a la luminosidad del ambiente, encendiendo el LED cuando la luminosidad sensada es menor que cierto umbral.
3. Programar la tarjeta Arduino para recibir un número (hexadecimal) del teclado y mostrarlo en un *display* a siete segmentos conectado en la *Protoboard*.

Las secciones de este reporte explican: el marco teórico de ambas partes de la práctica (marco teórico), los pasos que se siguieron para llegar al objetivo (desarrollo), los resultados obtenidos (resultados), las conclusiones y el papel que cada integrante del equipo desempeñó.

El código utilizado para programar la tarjeta está disponible en: https://github.com/HumbertoMartinezBarron/Practica1

## II. Marco teórico

Para la primera parte de la practica, se utiliza la tecnología $GitHub$. Esta plataforma de desarrollo colaborativo se usa para alojar proyectos utilizando el sistema de control de versiones $Git$. Se utiliza principalmente para la creación de código fuente de programas de computadora. Las ventajas de usar $Git$ para el sistema de trabajo que adoptaremos para este laboratorio se enumeran en la siguiente lista:

- Trabajo colaborativo.
- Flujo de trabajo adaptable.
- Ramificación.
- Velocidad.
- Seguridad.
- Costo.

Para la segunda parte de la práctica se utiliza una tarjeta de desarrollo $open-source$ llamada $ArduinoMega2560R3$. Esta tarjeta está contruida con un microcontrolador modelo $Atmega2560$ que posee pines de entradas y salidas (E/S), analógicas y digitales. Esta tarjeta es programada en un entorno de desarrollo que implementa el lenguaje Processing/Wiring.

El Arduino Mega tiene 54 pines de entradas/salidas digitales (14 de las cuales pueden ser utilizadas como salidas PWM), 16 entradas análogas, 4 UARTs (puertos serial por hardware),

cristal oscilador de 16MHz, conexión USB, jack de alimentación, conector ICSP y botón de reset. En la siguiente lista se enumeran sus características principales:

- Microcontrolador ATmega2560.
- Voltaje de entrada de – 7-12V.
- 54 pines digitales de Entrada/Salida (14 de ellos son salidas PWM).
- 16 entradas análogas.
- 256k de memoria flash.
- Velocidad del reloj de 16Mhz.

Arduino puede utilizarse en el desarrollo de objetos interactivos autónomos o puede comunicarse a un PC a través del puerto serial (conversión con USB) utilizando lenguajes como Flash, Processing, MaxMSP, etc.

Las restricciones que hasta ahora tenemos son el manejo de los comandos *Git*, es decir, nos hace falta practicarlos más para poder adquirir practica y velocidad al hacer uso de ellos y el uso del $software$ para el desarrollo de Arduino. Aún no somos capaces de visualizar los alcances que tiene esta tarjeta y este *sotware*. Se supone que al final del curso podremos hacer un uso aplicado a la ingeniería de esta tableta.

## III. Desarrollo

### III-A. *Parte 1*

Para cumplir con el objetivo de los comandos de línea simplemente fue necesario analizar la funcionalidad que se requería, comparar con la lista dada de comandos y de su funcionamiento y usar el comando `$ man (comando)` para consultar información adicional. A continuación se detallan los comandos que se utilizaron para resolver la primer parte del laboratorio.

1. Simplemente se accedió a la página.
2. Se dio *click* en la opción de Crear un nuevo repositorio
3. `$ git clone https://github.com/HumbertoMartinezBarron/Practica1.git`
4. `$ ls -al` (sí se encontraba el directorio `.git`)
5. `$ mkdir Practica\ 1`
6. `$ cd Practica\ 1; mkdir -p Dir_1/Dir_12/Dir_121/Dir_1211/Dir_12111; cd Dir_1; mkdir -p Dir_11/Dir_111; cd Dir_11; mkdir Dir_112; cd ..; gedit Dir_11/arch_11` (escribir algo en el archivo, guardarlo, cerrarlo y correr lo siguiente): `$ gedit Dir_11/Dir_111/arch_111` (escribir algo en el archivo, guardarlo, cerrarlo y correr lo siguiente): `$ gedit Dir_11/Dir_112/arch_112` (escribir algo en el archivo, guardarlo, cerrarlo y correr lo siguiente): `$ gedit Dir_12/Dir_121/arch_121` (escribir algo en el archivo, guardarlo, cerrarlo y correr lo siguiente): `$ gedit Dir_12/Dir_121/Dir_1211/Dir_12111/arch_12111` (escribir algo en el archivo, guardarlo y cerrarlo).
7. `$ ls -aR`: imprime el árbol de archivos entero.
8. `$ cp Dir_1 Dir_1_cp`
9. `$ mv Dir_1_cp Copia_dir`
10. `$ cat /opt/arduino-1.8.3/examples/*/*/*.ino >example_file.txt`: copia el contenido de los archivos `.ino` en todos los directorios del directorio `examples` a `example_file.txt`. Las primeras líneas son algún tipo de comentario, las últimas son código. Para averiguar el número de veces que aparece la palabra *arduino*, corremos `$ grep arduino example_file.txt | wc -l` para obtener primero todas las líneas en las que aparece, y después contar el número de líneas en el resultado.
11. `$ rm Dir_12/Dir_121/Dir_1211/Dir_12111/arch_12111`
12. `$ rmdir Dir_12/Dir_121/Dir_1211/Dir_12111`
13. `$ top`: se están ejecutando algunos programas que no se reconocen (ocupando una porción muy pequeña de memoria), Arduino y *FireFox*.

### III-B. *Parte 2*

*III-B1. Cargar el ejemplo Blink:* Este problema fue simple: lo único que fue necesario hacer fue abrir el ejemplo precargado de *Blink* y dar *click* en el botón de *Upload* en la interfaz gráfica de Arduino con el dispositivo conectado. En efecto, todo funcionaba correctamente.

*III-B2. Alumbrado que responde a cambios en la luminosidad del ambiente:* Para este ejemplo se utilizó la información del tutorial de la página *GeekFactory* [1]. En resumen, se conectó un solo LED, se le asignó un puerto de la tarjeta y se utilizó la medición de la fotoresistencia para determinar si el LED se encendería o si se apagaría. Todo el código se puede consultar en la página de *GitHub* proporcionada.

*III-B3. Interacción con Display para mostrar el número hexadecimal:* Para mostrar el número proporcionado en el *Display* a siete segmentos, se utilizó un código un poco más extenso, pues era necesario leer la entrada del teclado (lo cual se logra con la función `Serial.read()` de Arduino). Sin embargo, la función utilizada regresa un valor numérico entero, que representa el código *ASCII* del caracter. Por lo tanto, es necesario utilizar este código para decidir qué LEDs deben de encenderse en la tarjeta.

Una vez leído el caracter, se utilizó el comando `switch` para especificar los segmentos a encender manualmente. Por lo tanto, se definieron todos los valores de cada segmento (`LOW` o `HIGH`) para cada posible valor que nos interese del código *ASCII*. En este caso, sólo se definieron los casos en los que el caracter es una letra minúscula (de la *a* a *f*) o un número. Cualquier otro caracter resultará en la impresión de un mensaje que indica que ese caracter no es válido, apagando todos los segmentos del *Display*.

## IV. Resultados

### IV-A. *Primera parte de la práctica.Linux*

El resultado fue como se esperaba. Se logró generar de manera efectiva el árbol de directorios propuestos por la

práctica. De igual manera se logró almacenar los archivos en estos.

Con este resultado obtenido se ilustra claramente cómo se puede trabajar con archivos de código fácilmente y las ventajas de trabajar con *Git*, también se ilustra una herramienta que será muy útil en el laboratorio de Principios de Mecatrónica.

### *IV-B. Segunda parte de la práctica. Microcontrolador*

En principio, el resultado no fue como se había planteado. Fallamos por razones de tiempo. Nos costó trabajo familiarizarnos con el ambiente de trabajo de Arduino, era la primera vez que abríamos el programa y se conectaba la tarjeta.

Para la simulación del alumbrado que corresponde a cambios en la intensidad luminosa del ambiente no tuvimos mayor problema. Era una simulación relativamente sencilla y funcionó perfectamente, tal y como esta descrito en la práctica.

Para el segundo ejercicio se falló dentro de lo que restaba de las dos horas planeadas, el código aparentemente no presentaría fallas. Sin embargo, a la hora de ejecutarlo, nos preguntaba el caracter a ingresar $n$ veces. El *print* de esa pregunta estaba en un *loop* infinito.

La segunda falla que tuvimos en esta segunda parte de la práctica está relacionada con el alambrado del circuito en la *protoboard*. No consideramos la ~~instalación~~ de resistencia entre los pines del *display* y las conexiones hacia la tarjeta. El código aparentemente funcionaba, pero no se mostraba en el *display*.

~~El martes de 30 de enero de 2019 nos reunimos nuevamente para solucionar ambos problemas.~~ El problema con el código se soluciono cambiando de posición de línea la pregunta para evitar que entrara en un *loop* infinito. El segundo problema se solucionó instalando unas resistencias de 270~~Ω~~ Ω en donde deberían de haber ido desde un principio. Los problemas fueron solucionados satisfactoriamente. ~~A continuación se muestran algunas imá~~genes de la segunda parte de la pr~~áctica corriendo adecuadamente~~.

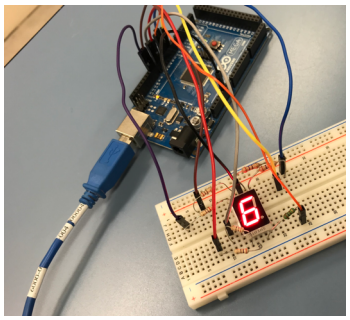

Figura 1. Se muestra en el *display* el número 6 solicitado al usuario por el programa corriendo.

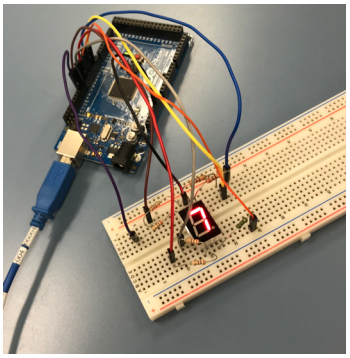

Figura 2. Se muestra en el *display* el número 7 solicitado al usuario por el programa corriendo.

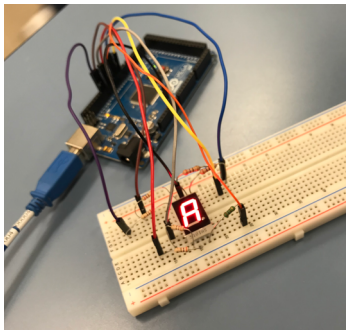

Figura 3. Se muestra en el *display* el caracter *A* solicitado al usuario por el programa corriendo.

## V. Conclusiones

### *V-A. Humberto Martínez*

Finalmente, podemos concluir que la práctica fue un éxito. En las dos sesiones que se le dedicaron, los resultados fueron palpables y útiles para futuros proyectos. Aprendimos a utilizar algunos comandos importantes de la terminal de ~~Unix~~/Linux, un poco de *Git* básico para poder manejar nuestros proyectos desde repositorios colaborativos en línea y logramos programar y correr algunos programas básicos en una tarjeta Arduino.

Como algo a recordar a futuro, se pueden consultar archivos desde algunos comandos fáciles de usar sin tener que explorar cada archivo gracias a la terminal. Además, es posible ver el contenido de los documentos sin tener que abrirlos - además de buscar palabras de interés.

En cuanto a *Git*, una nota mental útil es que puede servir como una gran herramienta de trabajo para equipos de programadores y para mostrar el trabajo que se ha hecho en el pasado a potenciales empleadores, colaboradores y compañeros. La herramienta resulta útil y puede que valga la pena aprender más al respecto.

En cuanto a Arduino, la interacción de la tarjeta con el mundo exterior puede lograr cosas relativamente básicas de forma muy fácil, pero es seguro que, con un mayor dominio de los programas, las bibliotecas y las aplicaciones de la tarjeta, se pueden llegar a construir proyectos de mucho mayor complejidad gracias a la escalabilidad del producto.

### *V-B. Juan Carlos Garduño*

De esta práctica me llevo buenas experiencias para mi vida como ingeniero. En primer lugar, aprendí a usar los principales comandos *Git* y comprendí el porque de ~~que~~ la mayoría de

los estudiantes de ingeniería en computación estén inmanente relacionados con el ambiente *Git*. Es decir, comprendí lo útil que puede ser esta herramienta, no sólo para los ingenieros en computación, sino para las demás ramas de la ingeniería que ocupen almacenar y compartir su código.

En segundo lugar, tuve mi primer acercamiento a la tarjeta Arduino, aunque en el primer ejercicio me sentí desubicado, en el segundo ejercicio entendí que era aplicar una serie de conocimientos previos aprendidos en las diferentes clases ya acreditadas.

También me llevo la lección de que sabiendo buscar, todo se encuentra. Es decir, en múltiples ocasiones el equipo se enfrento con diferentes problemas, entonces nos pusimos investigar en ~~*internet*~~ y todo esta ahí, no estoy diciendo que todo lo que se encuentre ahí este correcto, se tiene que hacer una depuración para lograr encontrar la información correcta.

### *V-C. Sebastián Aranda*

En ésta práctica nos enfrentamos a un nuevo reto muy interesante. Este primer acercamiento en la práctica a los programas de Linux y Arduino, a pesar de que no requería conocimiento muy avanzado de los programas mencionados, fue una tarea difícil porque, al menos para mí, mi conocimiento del funcionamiento del programa era prácticamente nulo. Creo que de esta práctica obtuve herramientas importantes para el resto del curso y de mi carrera. Fue muy útil familiarizarme con la interfaz de programa de Arduino y trabajar con la terminal de Linux para guardar programas en el repositorio de *Github*.

## VI. Rol o Papel

### *VI-A. Humberto Martínez*

En ambas secciones de la práctica, Humberto tomó posesión por más tiempo del teclado. En la primera parte, entre Humberto y Juan Carlos (no tan familiriarizado con el ambiente *Git*) ejecutaron los comandos. En la segunda parte, Humberto iba a escribiendo el código, incluyendo su propia lógica y las de sus compañeros. En cuanto al reporte Humberto elaboró las siguientes secciones:

- Introducción.
- Desarrollo.
- Sus respectivas conclusiones.

### *VI-B. Juan Carlos Garduño*

En la práctica, Juan Carlos se encargó de investigar dudas que iban surgiendo, así como tablas e instructivos. En la primera parte de la practica, se encargó junto con Humberto a ejecutar los comandos *Git*. En la segunda parte, iba dictando algunas secciones del código a Humberto que había investigado y podían ser de utilidad para la compilación del programa. De igual manera, iba revisando que la sintaxis de Humberto en el código, fuera la correcta. En cuanto al reporte Juan Carlos elaboró las siguientes secciones:

- Resultados.
- Colaboró en el marco teórico.
- Rol o papel.
- Sus respectivas conclusiones.

### *VI-C. Sebastián Aranda*

En la práctica, Sebastián se encargo del alambrado, así como de ir colaborando con los demás integrantes en la elaboración del código. En cuanto al reporte el elaboro las secciones siguientes:

- Resumen.
- Colaboró en el marco teórico.
- Sus respectivas conclusiones.

## Referencias


[1] Tutorial Arduino con resistencia LDR, Geek Factory, 3 de noviembre de 2014. Disponible en: https://www.geekfactory.mx/tutoriales/tutoriales-arduino/tutorial-arduino-con-fotoresistencia-ldr/

[2] Esaú A. (2016). Ventajas del uso GIT. enero 20,2019, de Open Webinars Sitio web: https://openwebinars.net/blog/ventajas-del-uso-de-git/

[3] Arduino Mega 2560 R3, Arduino, 30 de enero de 2019. Disponible en: https://arduino.cl/arduino-mega-2560/

[4] ~~Aranda, E. (2014). Herramientas Informá~~ticas de las Matemá~~ticas en Ingeniería.Curso de LATEX. enero 30, 2019, de Departamento de Matemá~~ticas E.T.S. Ingenieros Industriales Universidad de Castilla – La Mancha Sitio web: ~~http://matematicas.uclm.es/earanda/wp-content/uploads/downloads/2013/10/latex.pdf~~