Guía paso a paso

Construcción "Cerradura digital con código" en Arduino

Scafati, Diego Ariel <dscafati@gmail.com>

Ingeniería de sistemas
Facultad de Ciencias Exactas - UNICEN Tandil
2016

Indice

Introducción

En este informe se describirán los pasos necesarios para la construcción de una “**cerradura digital con códigos**” completamente configurable.

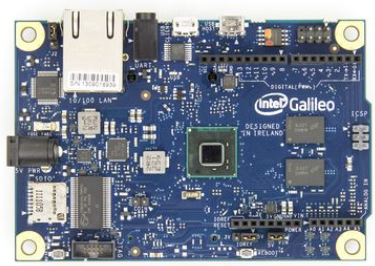
“Un cerrojo electrónico o teclado de código es un teclado electrónico usado para teclear un código de seguridad, que permite abrir una puerta de un edificio.
Se utiliza en lugar de la llave haciendo accionar un electroimán de apertura de los cerrojos de las puertas en diferentes lugares como por ejemplo: las puertas de acceso a áreas restringidas, las puertas de las habitaciones de los hoteles, las puertas de los edificios para permitir el acceso a las personas autorizadas,etc..”

Fuente: Wikipedia

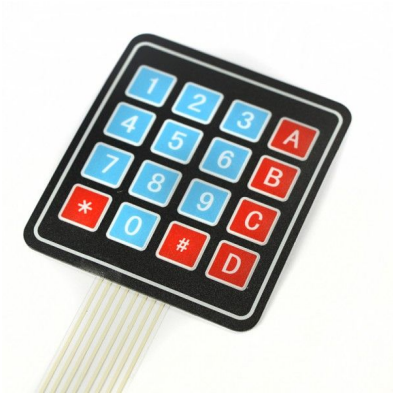
Elementos necesarios

Para realizar este proyecto se utilizaron los siguientes elementos:

Placa Intel Galileo primera generación



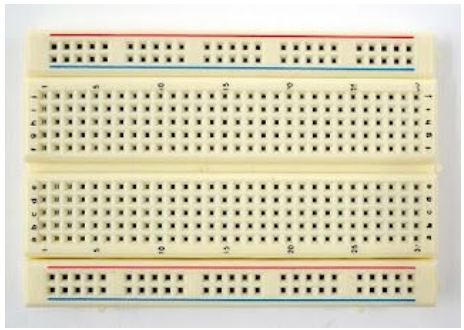
Teclado matricial de membrana 4x4 (3x4 es suficiente)



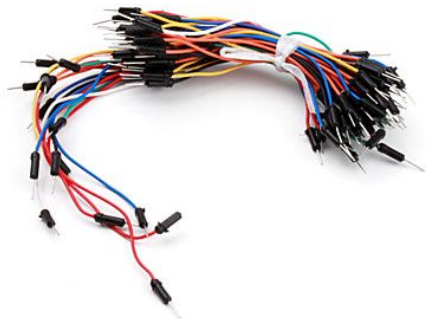
Cable ethernet (Cruzado)

Cable mini usb

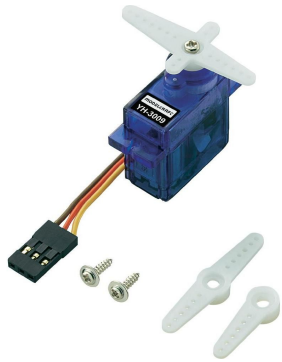
Protoboard (Placa de pruebas)



Cable Macho a Macho para protoboard



Servo



Tarjeta micro SD 2GB

Placa WiFi mini-pcie Atheros ar5b225 * (Opcional)

Los elementos a utilizar no necesariamente tienen que ser igual a los mencionados.

Por ejemplo, con un **teclado matricial de 3x4** es suficiente. La tarjeta micro SD puede ser de capacidades mayores a 2gb (**hasta 32gb**).

* En este proyecto, se utilizó una placa WiFi recuperada de una notebook que no estaba en funcionamiento. De todas formas, los modelos **Intel® Centrino® Wireless-N 135** e **Intel® Centrino® Advanced-N 6205** serían mejores opciones dado que funcionan con cualquier imagen de Linux provista por Intel (Este tema se verá más adelante)

Funcionamiento general

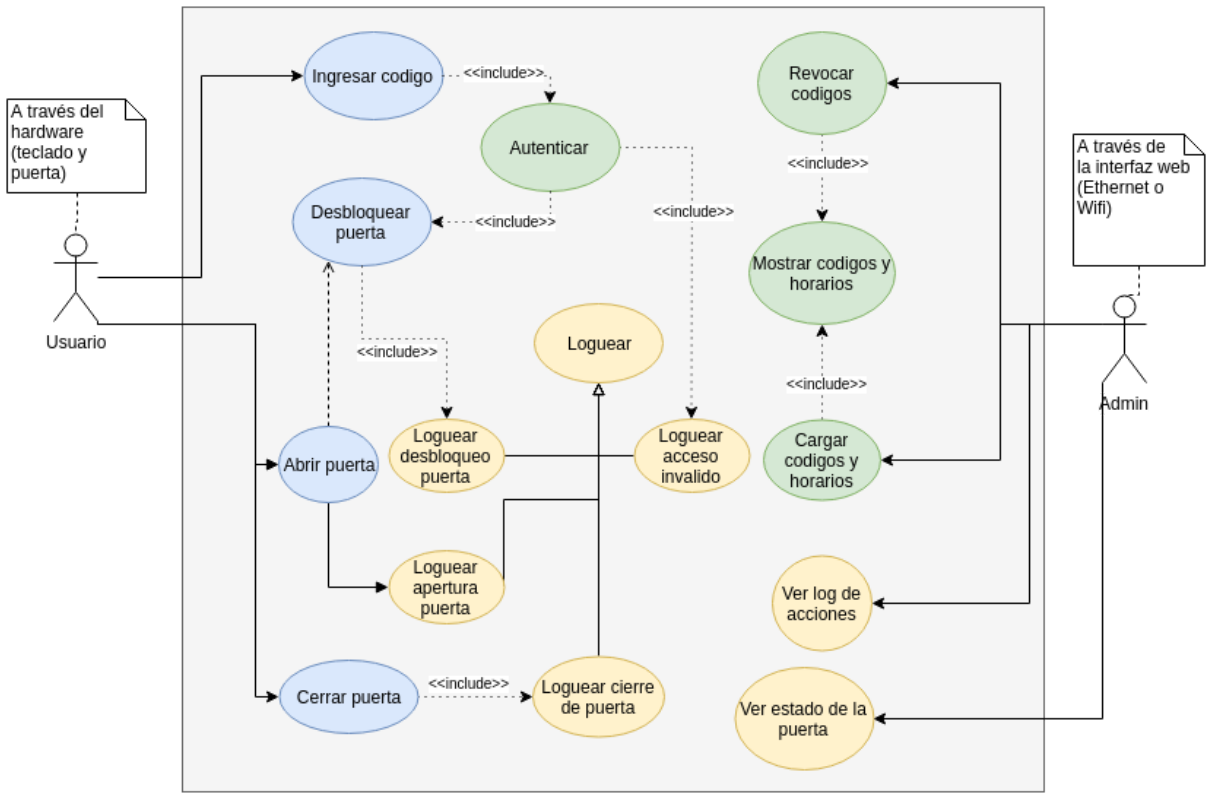
El funcionamiento de la cerradura definido para este proyecto es el siguiente:

Se cuenta con dos tipos de actores (usuarios): Por un lado las personas que utilizarán la cerradura con el objetivo de abrir la puerta y acceder al sitio restringido en cuestión, a partir de ahora llamadas "**usuarios**"; Por otro lado las personas que utilizarán la cerradura con fines de configurarla y realizar auditorías de seguridad (ver cuando se abrió la puerta, quien la desbloqueó, etc.), que serán llamados "**administradores**"

Por el lado de los usuarios, se desea que puedan ingresar un código el cual desbloquea la puerta. Seguido a esto, puedan abrir la puerta y finalmente cerrarla luego de acceder.

Por el lado de los administradores, se desea que puedan agregar códigos y asociarlos a un rango horario, que puedan revocar códigos, que puedan ver el registro de acciones (En que momento se desbloqueó la puerta, cuando fue abierta y cuando cerrada, y en qué momento se intentó acceder con un código inválido).

Esta funcionalidad se detalla mejor en el siguiente diagrama de casos de uso UML.



Ademas se agregó otra funcionalidad accesoria, como mostrar el estado actual de la puerta, y la posibilidad de sincronizar el reloj interno a través de la interfaz web.

Demostración:

https://youtu.be/Qq4_uDILZd0

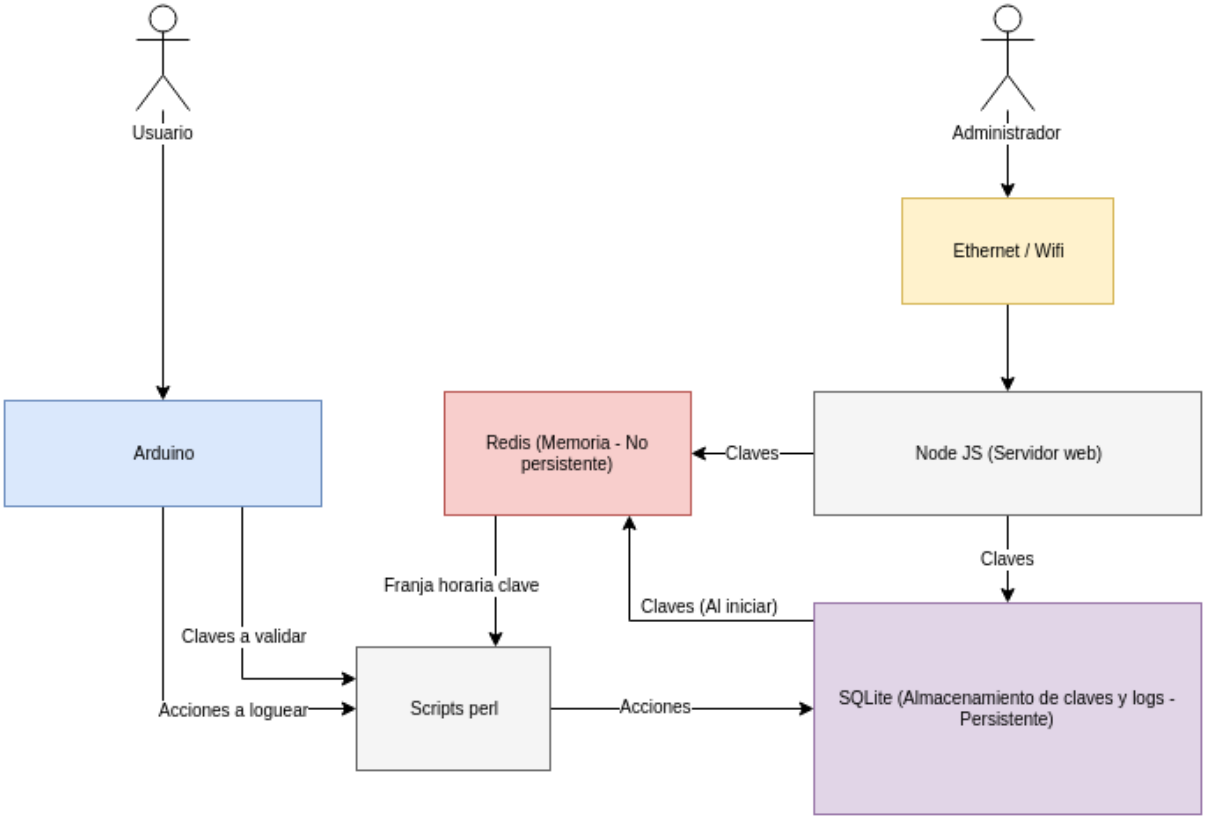
Arquitectura del proyecto

En esta sección se detallarán cuestiones relacionadas con la arquitectura del sistema. Esto permite entender cómo está funcionando el proyecto internamente para poder extenderlo o modificarlo con mayor claridad. Sin embargo no es necesario comprender esto para hacer funcionar el proyecto tal cual esta implementado hoy dia. Simplemente se deben seguir los pasos mencionados en la sección siguiente.

Notar que una de las características más importantes de las placas Galileo es que cuentan con un sistema operativo Linux ejecutándose en todo momento y al cual se puede acceder desde el código Arduino tradicional (C++), a través de llamadas al sistema (método system()). Para este proyecto se aprovecharon muchas de las funcionalidades provistas por esta imagen Linux

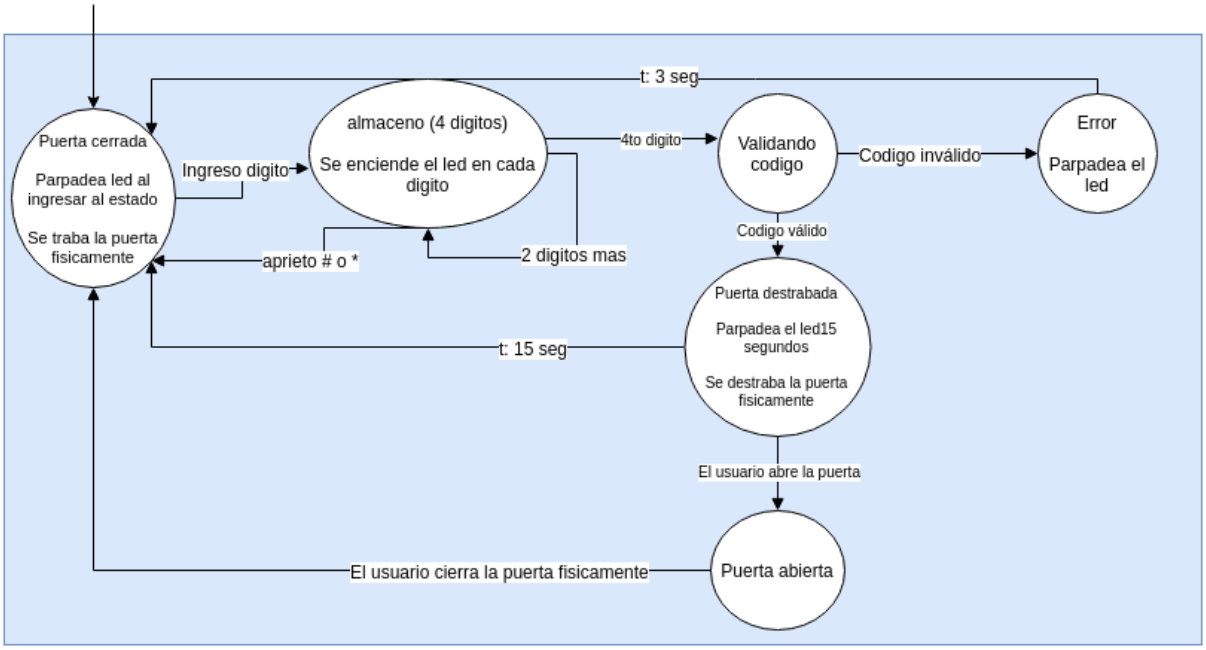
Arquitectura general del sistema

En el siguiente esquema se muestra a grandes rasgos la arquitectura del sistema:



Se observan dos formas de almacenamiento: Redis (Para mantener las claves en memoria y permitir validar los accesos de forma instantánea) y SQLite (Para recuperar las claves luego de reiniciar el dispositivo y para almacenar los logs que no requieren tiempos de respuesta extremadamente cortos)
Ademas se observa que se cuenta con codigo Arduino (c++) en ejecucion, asi como tambien un servidor web en NodeJs (javascript). Para acceder a las opciones de persistencia desde Arduino (redis/sqlite) se utilizan scripts (perl) intermediarios.

Máquina de estados corriendo en Arduino



La funcionalidad programada en la placa arduino se representa en forma de máquina de estados como se muestra en la imagen anterior.

Diseño base de datos

La base de datos contiene un diseño trivial, sin relaciones. Se cuenta con dos tablas, una para almacenar los códigos y otra para llevar control de las acciones realizadas:

claves	
PK	**id**
	codigo
	user
	h_desde
	h_hasta

log	
PK	**id**
	user
N	accion
	date_time

Guía Paso a Paso para realizar el proyecto

En esta sección se detallan los pasos a realizar para configurar la placa arduino, conectarla, compilar y subir el código necesario para el funcionamiento del mismo.
En las primeras secciones se proveerán links de descarga que no son fáciles de encontrar en internet dado que (casi) todos los tutoriales y guías existentes contienen enlaces caidos

1. Preparando la placa Arduino

La preparación de la placa Arduino consta de dos pasos. En primer lugar debemos actualizar el firmware de nuestra placa (si no lo hemos hecho anteriormente). Luego y para este proyecto en particular, debemos descargar y hacer funcionar una imagen de Linux especial para poder contar con todas las herramientas que necesitamos, especialmente los drivers WiFi.

Instalacion Drivers (solo Windows)

Para conectar la placa a la PC necesitamos en primer lugar instalar los drivers. Estos vienen incluidos dentro de la herramienta de actualización de firmware, que puede conseguirse desde el siguiente enlace:
https://downloadcenter.intel.com/download/24748/

Una vez descargado y extraido el archivo .zip, procedemos a instalar los drivers, que se encuentran dentro de la carpeta “Galileo Driver”.

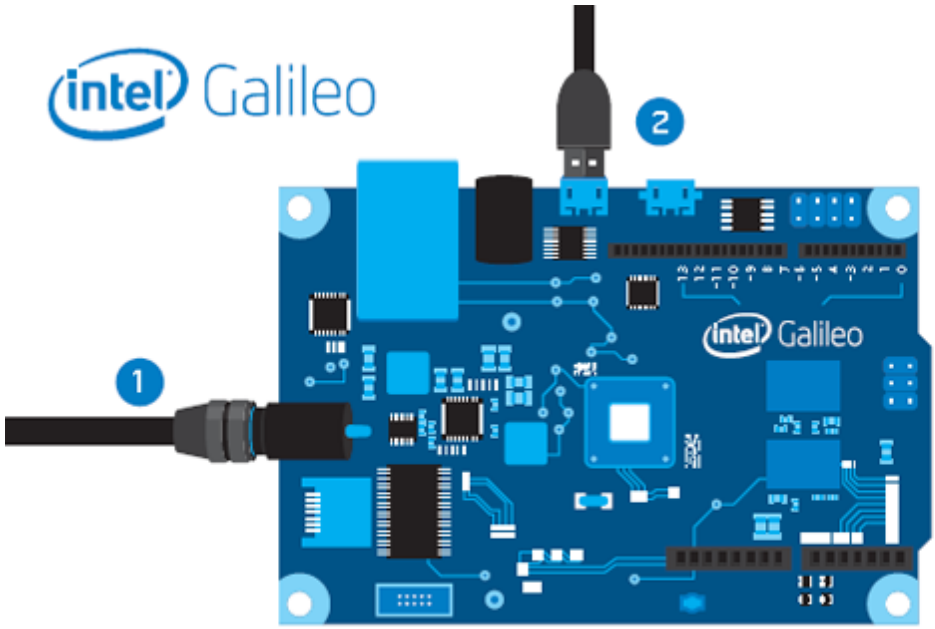
Para esto debemos reiniciar el equipo, con la opción “*Deshabilitar el uso obligatorio de controladores firmados*” habilitada.

En windows 8.1:

1. Mover el mouse a la derecha de la pantalla
2. Click en Opciones -> Cambiar configuración de PC -> Actualizar y recuperar -> Recuperación -> Inicio Avanzado -> Reiniciar ahora.
3. Luego debemos ir a “Solucionador de problemas” -> “Opciones avanzadas” -> “Configuración de inicio”
4. Finalizamos el reinicio del sistema

Estos pasos se encuentran explicados en el siguiente enlace:
http://josmangarsal.es/instalar-drivers-no-firmados-en-windows-8-y-8-1/

Una vez reiniciado el sistema, debemos conectar la placa Arduino a la pc de la siguiente manera:



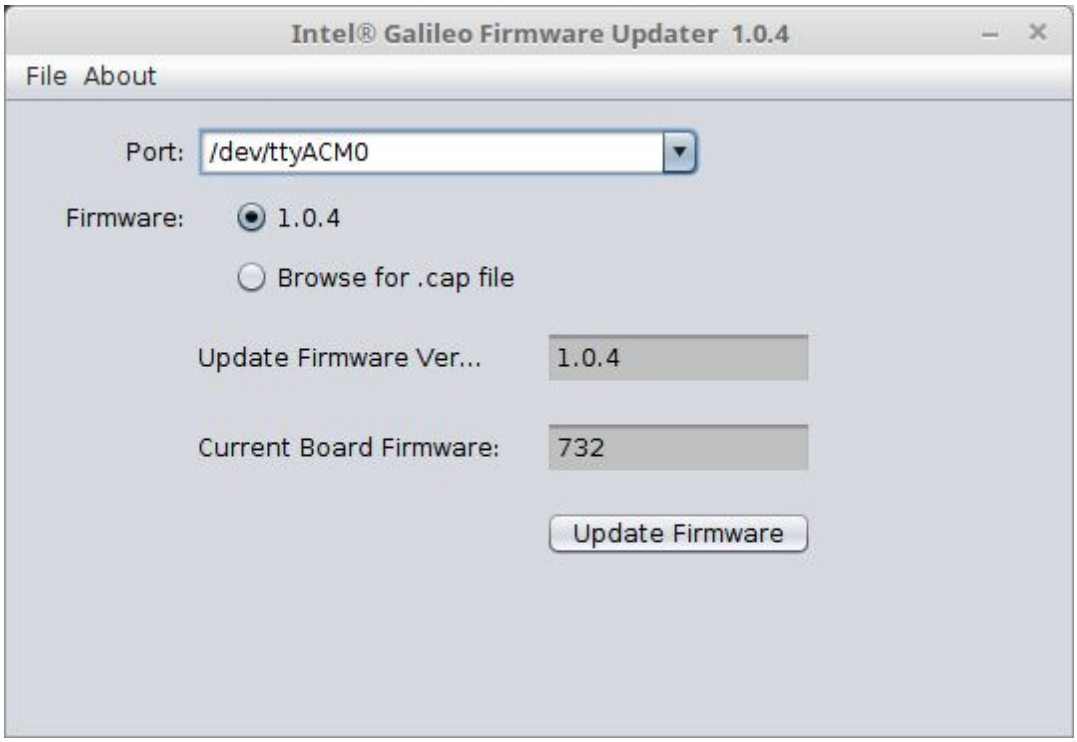
1 Connect power cable to Galileo and to a power outlet
2 Connect USB cable to the Galileo USB device port and to a PC

Notar que estamos usando el cable mini usb, conectado en la placa al puerto más cercano al puerto ethernet.

Al encender la placa y ser detectada por el sistema operativo, procedemos a instalar los drivers.
Para esto debemos ir a la sección del Panel de Control “Sistema”, y luego a *“Administrador de dispositivos”*
En el listado de dispositivos debemos buscar aquel que se corresponde con la placa arduino, que reconoceremos por el icono  .
Le hacemos doble click -> Actualizar controlador -> Buscar software de controlador en el equipo -> Elegir en una lista de controladores de dispositivo en el equipo -> Mostrar todos los dispositivos -> “Usar disco…”
Luego indicamos la ruta al archivo “linux-cdc-acm” dentro de la carpeta “Galileo Driver” descargada anteriormente, y finalizamos la instalación dando click en aceptar y siguiente.

Actualización Firmware

Para actualizar el firmware simplemente debemos ejecutar la herramienta descargada anteriormente, conectando la placa a la pc de la misma manera. El firmware viene incluido dentro de la herramienta por lo que no es necesario descargar archivos adicionales.



Si bien esta herramienta es multiplataforma, no se la pudo hacer funcionar correctamente en Linux por lo que se recomienda utilizar únicamente windows para este paso.

Booteando con una imagen de Linux más completa

Luego de actualizar el firmware, procedemos a cargar la imagen de linux apropiada para el desarrollo de este proyecto.
Si bien uno puede compilar sus propias imágenes de linux embebidas (proyecto yocto), Intel nos provee de dos imágenes adicionales listas para usar en la placa Galileo. La que utilizaremos se consigue en el siguiente enlace:
https://software.intel.com/sites/landingpage/iotdk/board-boot-image.html

Esta imagen también provee herramientas de integración con el IDE Intel xdk, el cual nos permite conectarnos y desarrollar fácilmente con la placa. Esto no es requerido para este proyecto.
También, en otros proyectos se pueden usar imagenes mas chicas, como la siguiente:
https://downloadmirror.intel.com/24355/eng/sdcard.1.0.8.tar.bz2

Entonces, una vez descargada la imagen, extraemos el archivo .bz2 y nos encontramos con un archivo llamad *"iot-devkit-prof-dev-image-galileo-********.direct"*
Este archivo contiene la imagen que debemos extraer sobre la memoria micro SD.

Si estamos en linux debemos ejecutar el comando:

*sudo dd if=iot-devkit-prof-dev-image-galileo-20160606.direct of=/dev/**sdc** bs=3M conv=fsync*

Reemplazando sdc por el identificador correspondiente a la tarjeta micro sd, y asegurandonos que la imagen se encuentra en el archivo "iot-devkit-prof-dev-image-galileo-20160606.direct"
Atencion! Al ejecutar esta acción, todo el contenido de la tarjeta de memoria se perderá, por lo que debemos estar seguros de no ejecutar el comando sobre otra unidad de disco. Para averiguar el identificador de la memoria micro SD podemos ejecutar alguna de las herramientas de gestión de discos que vienen incluidas por defecto en la mayoría de las distribuciones linux.

Este paso no se probó en windows, pero debería poder realizarse de forma análoga usando herramientas como la siguiente:
http://www.chrysocome.net/dd

Una vez finalizado este proceso, la tarjeta micro SD está lista para ser conectada a la placa en el slot correspondiente. Si la placa estaba encendida, debemos reiniciarla para que bootee desde la imagen recién cargada.

Para verificar esto debemos observar 2 cosas:

1) El led correspondiente a la tarjeta micro SD parpadea al encender
2) La versión del sistema operativo cargada se corresponde con la imagen nueva y no con la original.

Para el segundo caso debemos acceder al sistema operativo de la placa. Esto se logra a través de una conexión serie (necesitamos un cable especial) o a través de ssh, como se explica a continuación. Si pudimos acceder por ssh, entonces ya comprobamos que la imagen de linux fue cargada correctamente.

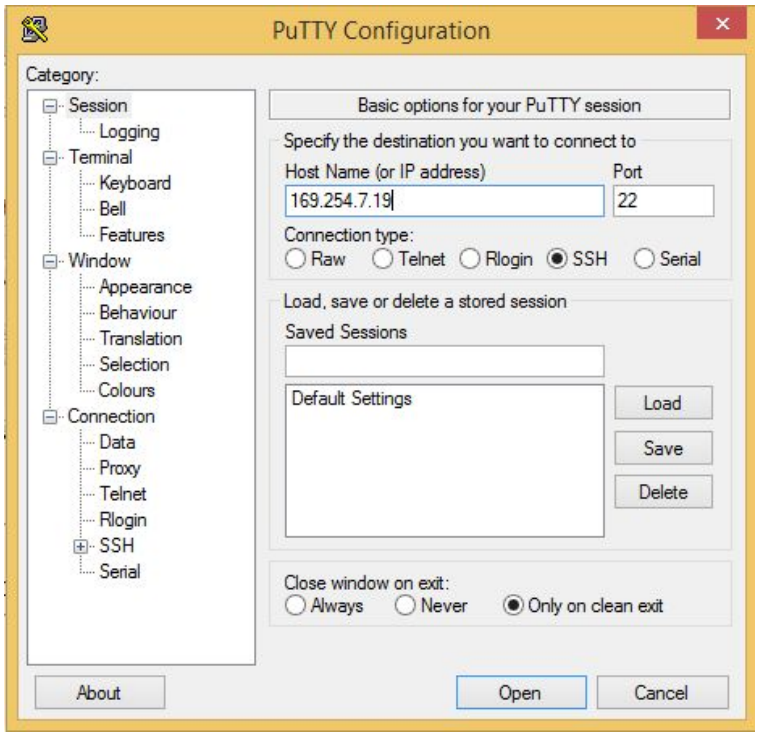
2. Accediendo a la placa por SSH

Para acceder por ssh debemos conectar la placa a la pc a través de los puertos ethernet, utilizando el cable Ethernet ya mencionado. (Luego podrá hacerse por wifi siguiendo los mismos pasos)
Esta conexión depende del sistema operativo utilizado y de la configuración particular de cada pc, por lo que no se explicará en este informe. Generalmente, si no especificamos direcciones ip fijas esta conexión deberia realizarse de manera automática.

Una vez realizada la conexion, podemos utilizar la herramienta ssh de linux (comando ssh), o la herramienta putty en windows: http://www.putty.org/
Si estamos en linux simplemente debemos ejecutar el comando

```
ssh root@169.254.7.19
```

La dirección ip puede variar en cada conexión.
Desde windows nos conectamos de la misma manera, pero indicando el usuario (root) en el paso siguiente:



Lo explicado aquí es análogo para una conexión ssh a través de wifi.

3. Configurando la interfaz WiFi (Opcional)

Conectando la tarjeta WiFi

La conexion de la placa wifi resulta casi trivial. Debemos conectar la placa al slot mini pci-e que se encuentra del lado inferior de la placa Galileo, y luego conectar la antena a la misma. Debemos asegurarnos de desconectar la placa Galileo antes de realizar esta conexión, o la tarjeta wifi podría dañarse.

La conexión resultante es como se muestra en la imagen anterior, aunque también existe un chasis que permite asegurar la placa a su posición y que no se mueva:

Si no disponemos de este chasis, debemos buscar la forma de que la placa wifi no se desconecte mientras está encendida. Una solución es utilizar cinta adhesiva

Verificando reconocimiento de drivers

Luego de conectar la tarjeta wifi, debemos verificar que esta fue reconocida por el sistema operativo Linux.
Para esto accedemos por ssh a través de la conexión ethernet (como ya se explicó), y ejecutamos el comando iwconfig. Entre las interfaces listadas deberíamos poder ver “wlan0”, “wlp1s0” u opciones similares.

```
wlan0     IEEE 802.11bgn  ESSID:"Siamese"  Nickname:"<WIFI@REALTEK>"
          Mode:Managed  Frequency:2.462 GHz  Access Point: E8:3E:FC:C3:0D:70
          Bit Rate:54 Mb/s   Sensitivity:0/0
          Retry:off   RTS thr:off   Fragment thr:off
          Power Management:off
          Link Quality=100/100  Signal level=100/100  Noise level=0/100
          Rx invalid nwid:0  Rx invalid crypt:0  Rx invalid frag:0
          Tx excessive retries:0  Invalid misc:0   Missed beacon:0

lo        no wireless extensions.

eth0      no wireless extensions.
```

Si no aparece listada la placa wifi, debemos utilizar otra placa que sea compatible. (Existen alternativas pero implican recompilar la imagen de linux, y otros procesos que varían según cada fabricante)

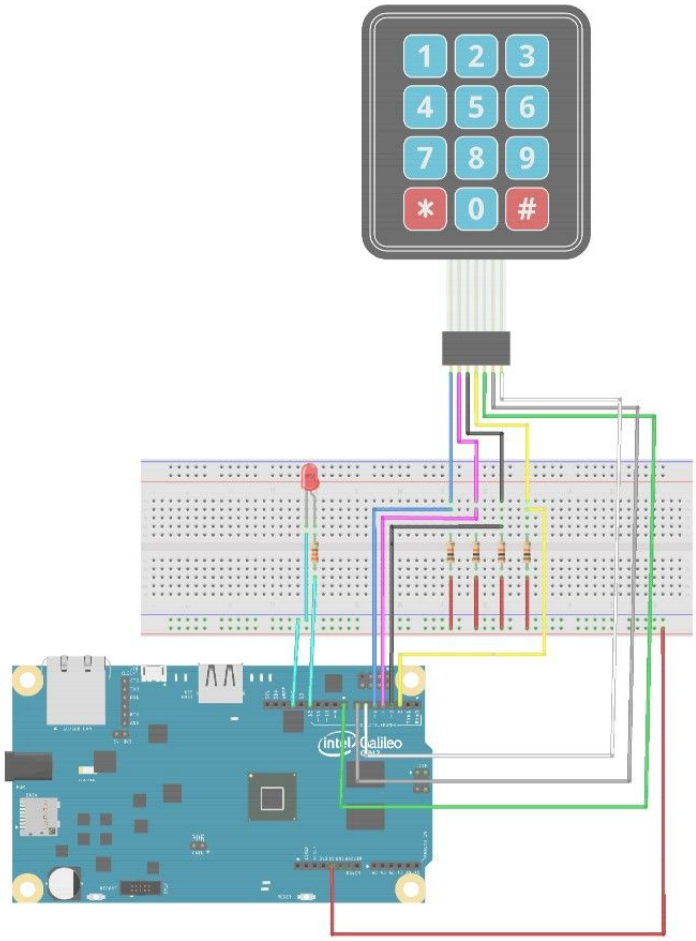
Conectando a una red WiFi

Si la placa aparece listada, procederemos a conectarnos a una red wifi.
Para ello utilizamos el comando “connmanctl”, ejecutandolo desde la misma terminal por SSH.
Esto se encuentra bien explicado en el siguiente enlace:
https://software.intel.com/en-us/node/519955

4. Conectando los elementos de hardware

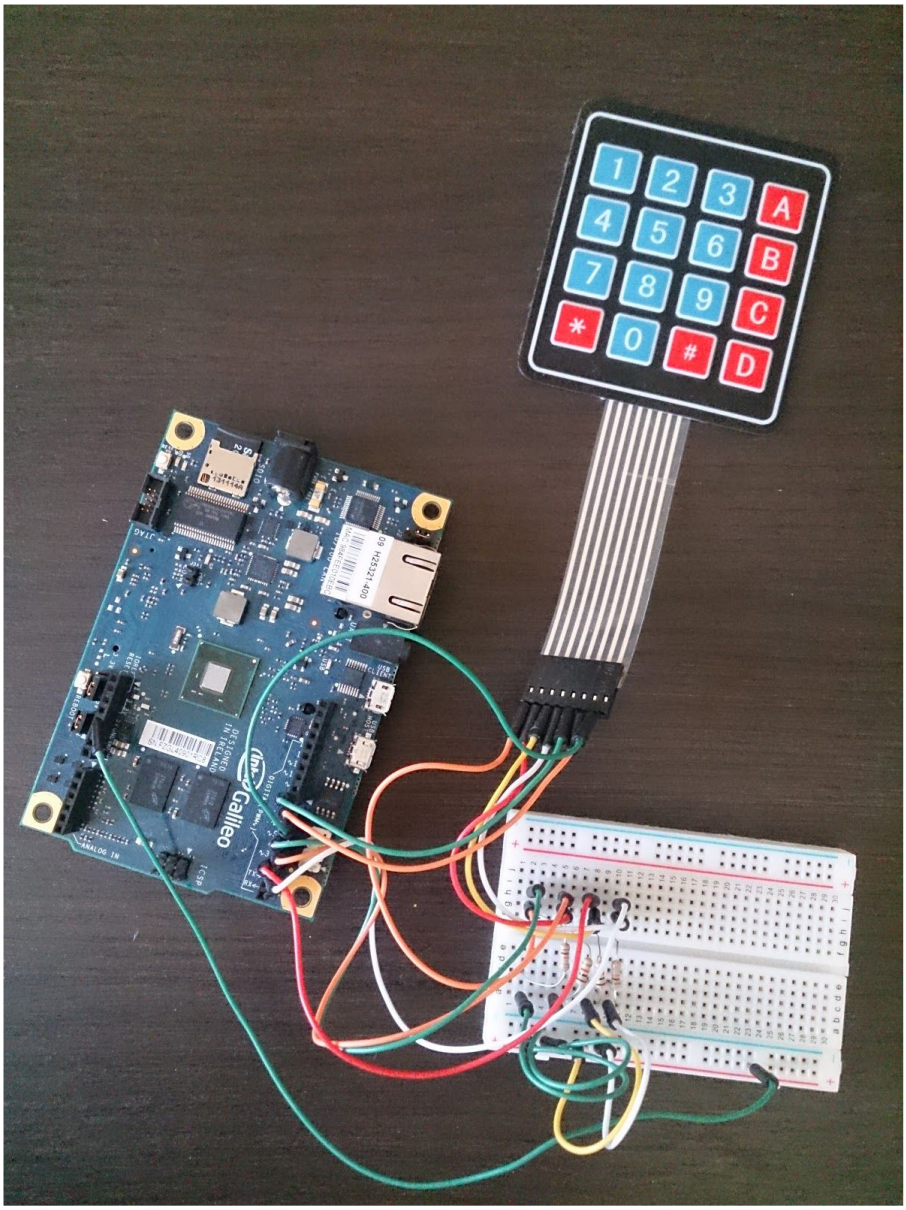
Teclado

La conexión del teclado se muestra en el siguiente esquema:



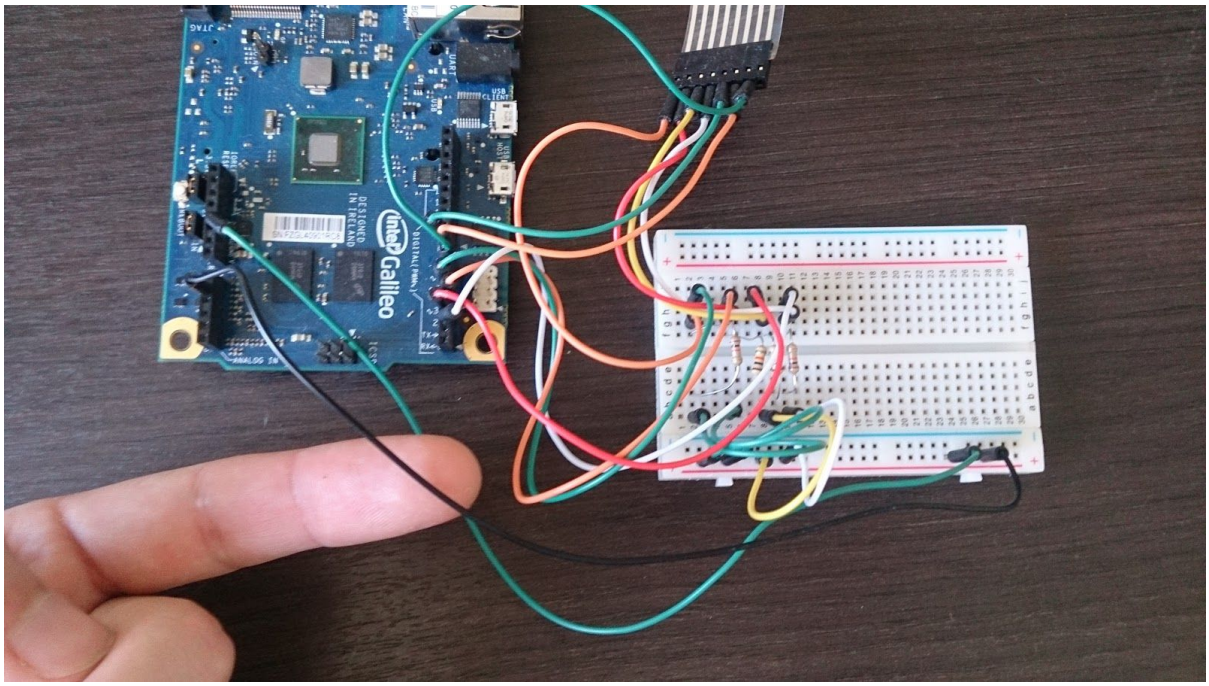
(La conexión del led mostrado en la imagen no es necesaria)

En este proyecto se utilizó un teclado de 4x4, pero dado que solo se requieren las columnas que contienen numeros, se debe dejar desconectado el ultimo pin.
Las resistencias que conectan cada pin con Vcc (5v) son las llamadas resistencias pull-up, las cuales son necesarias debido que se encontraron fallos en las resistencias pull-ups incluidas en la placa Galileo. En otro caso la conexión podría ser directa, sin necesidad de utilizar un protoboard.
En nuestro caso se asignaron los mismos pines digitales mostrados en el esquema anterior, resultando en la siguiente conexión:



Interruptor

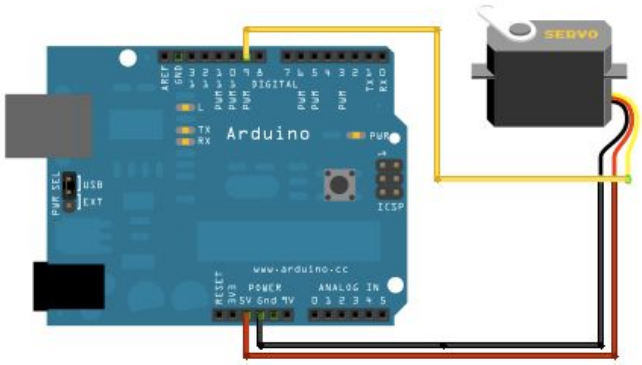
El interruptor representa un sensor que indica si la puerta se encuentra abierta o cerrada. Simplemente conectamos un pin analogico a 5v y lo desconectamos al momento de “abrir la puerta”. En este proyecto se utilizo el **pin analogico 1**.



En la imagen se muestra el cable correspondiente al “interruptor”. Conectado al pin A1

Servo

La conexión del servo resulta bastante sencilla. Debemos identificar cada uno de los pines del mismo y conectarlos a 5v, gnd y al pin digital 11 respectivamente
(Notar que en la imagen se utiliza el pin 9, pero en este proyecto se utilizó el **pin 11**)



El propósito del servo es representar el bloqueo o desbloqueo de la puerta, como si fuese una llave que gira al momento de desbloquear la puerta por software. Este puede cambiarse por cualquier otro accesorio de hardware más sofisticado.

5. Programando la placa (Arduino)

Librerías requeridas

Para facilitar la programación de la placa arduino se utilizaron las siguientes librerías, las cuales deben ser descargadas y extraídas dentro de la carpeta “libraries” donde se encuentra el IDE Arduino:

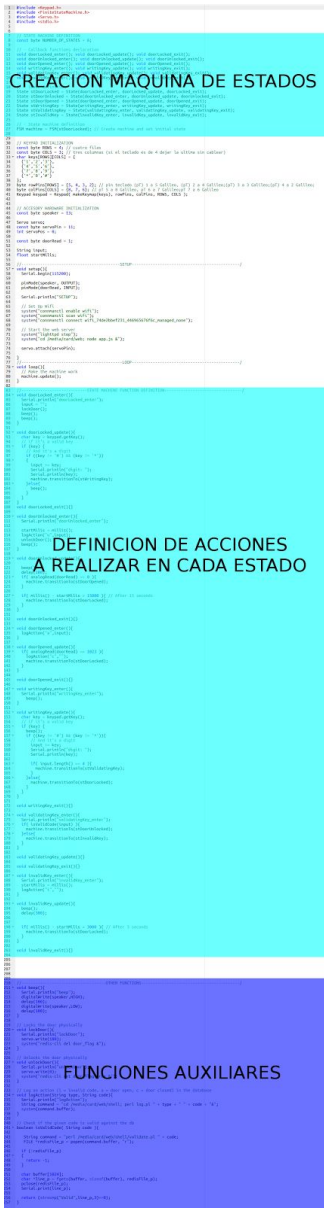
http://playground.arduino.cc/Code/Keypad: Permite utlilizar de manera sencilla el teclado
http://playground.arduino.cc/Code/FiniteStateMachine#Download: Permite trabajar con máquinas de estado de manera sencilla.
Las demás librerías utilizadas se encuentran integradas por defecto en el IDE

Descargando el código

El código puede conseguirse en el siguiente enlace:
https://github.com/dscafati/arduino-electronic-door-lock/tree/master/ino

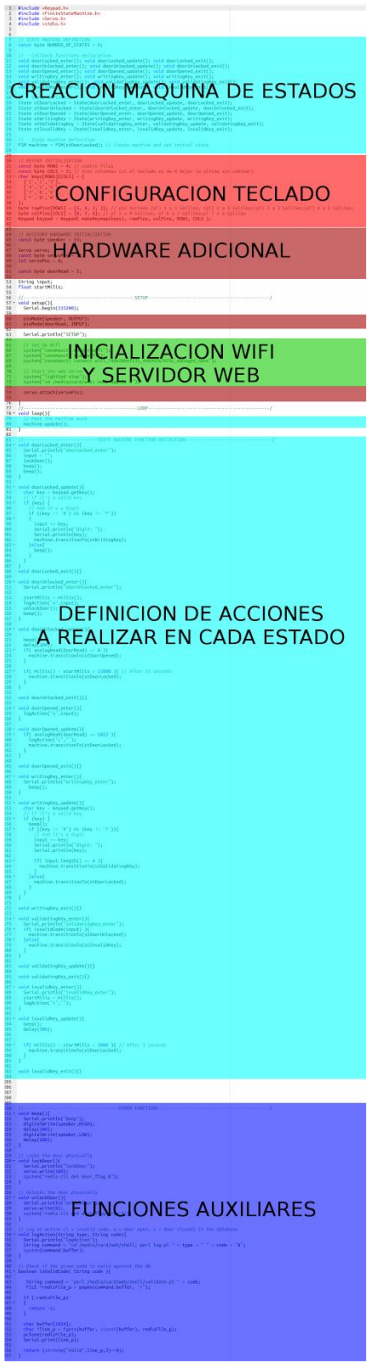
Explicación breve del código

Como se mostró anteriormente, la funcionalidad de la cerradura se modeló como una máquina de estados. Cada estado ejecuta funciones en 3 momentos distintos: Al ingresar al estado; Al permanecer en el estado (se ejecutan continuamente); Al salir del estado.
El código se encuentra estructurado en este aspecto de la siguiente manera:



El detalle de la implementación de cada función se puede observar claramente en el código. La misma corresponde a lo mostrado en la máquina de estados anteriormente.
Las funciones llamadas “auxiliares” son aquellas compartidas por varios estados, o que realizan funcionalidad más avanzada, como llamados al sistema operativo para interactuar con los datos del sistema web.

También se observa en la parte superior del archivo la inicialización y configuración del teclado matricial, y el hardware adicional (servo, interruptor).
El código restante corresponde a la importación de librerías, variables auxiliares, configuración de la transmisión serial, y la conexión wifi, ejecutando los mismos comandos mostrados anteriormente en la terminal SSH.



Programando la placa

Los pasos para programar la placa son iguales a cualquier otro proyecto Arduino. No se necesitan IDEs especiales para utilizar la placa galileo.
Simplemente seleccionamos la placa y el puerto correspondiente, y subimos el código

6. Configurando el servidor web (Node Js)

Descargando el código fuente

El código fuente se puede conseguir desde el siguiente enlace:
https://github.com/dscafati/arduino-electronic-door-lock
El proyecto fue desarrollado utilizando las siguientes herramientas, las cuales deben ser instaladas para trabajar y descargar las dependencias que el proyecto necesita:

Git (Solo para desarrollar)

Linux
sudo apt-get install git
Windows
https://git-scm.com/downloads

NodeJS y Npm

Windows y Linux
https://nodejs.org/en/
Linux
Algunas distribuciones permiten instalarlo mediante el gestor de paquetes.

Grunt (Solo para desarrollar)

Se instala utilizando la herramienta npm
sudo npm install -g grunt

Redis (Solo para desarrollar)

Linux:
sudo apt install redis-server
Windows:
Ver https://github.com/MSOpenTech/redis dado que redis no da soporte oficial para windows

Luego de instalar las herramientas anteriores, el siguiente paso consiste en clonar el proyecto, o descargarlo como un archivo zip.

Windows

Se recomienda descargar como zip y extraer en una carpeta llamada “web”

Linux

git clone git@github.com:dscafati/arduino-electronic-door-lock.git web

Luego dentro de la carpeta web debemos ejecutar los siguientes comandos:

Opción 1: Si se desea editar el código:

npm install
bower install

Esto descargará las dependencias que necesita el proyecto para desarrollar sobre el mismo.

Luego al momento de trabajar ejecutamos:

grunt workon

Esto ejecutará la aplicación web y estará a la escucha de modificaciones en los archivos del proyecto para recargar la misma automáticamente, entre otras tareas.

Opción 2: Si solo se desea subir el código a la placa:

npm install --production
bower install

Esto instalará solamente aquellas dependencias necesarias para la ejecución de la aplicación, no para el desarrollo de la misma. Lo que nos permitirá ahorrar espacio en la memoria micro SD. Se debe trabajar en otra copia del proyecto donde no se haya ejecutado nunca “npm install”

Subiendo los archivos a la placa

Para subir los archivos debemos conectar la tarjeta micro sd nuevamente a la pc. Una forma fácil de realizar esto es conectando la placa arduino mediante el cable usb a la pc. Esto nos permitirá acceder a la memoria sin desconectarla de la placa.

Si observamos las particiones dentro de la memoria micro Sd, notaremos una partición pequeña (53MB) que contiene archivos relacionados al booteo. Si bien podríamos utilizar cualquier carpeta de cualquier partición de la memoria, se prefirió utilizar esta dado que se monta automáticamente tanto en linux como en windows, incluso en la imagen embebida (/media/card)

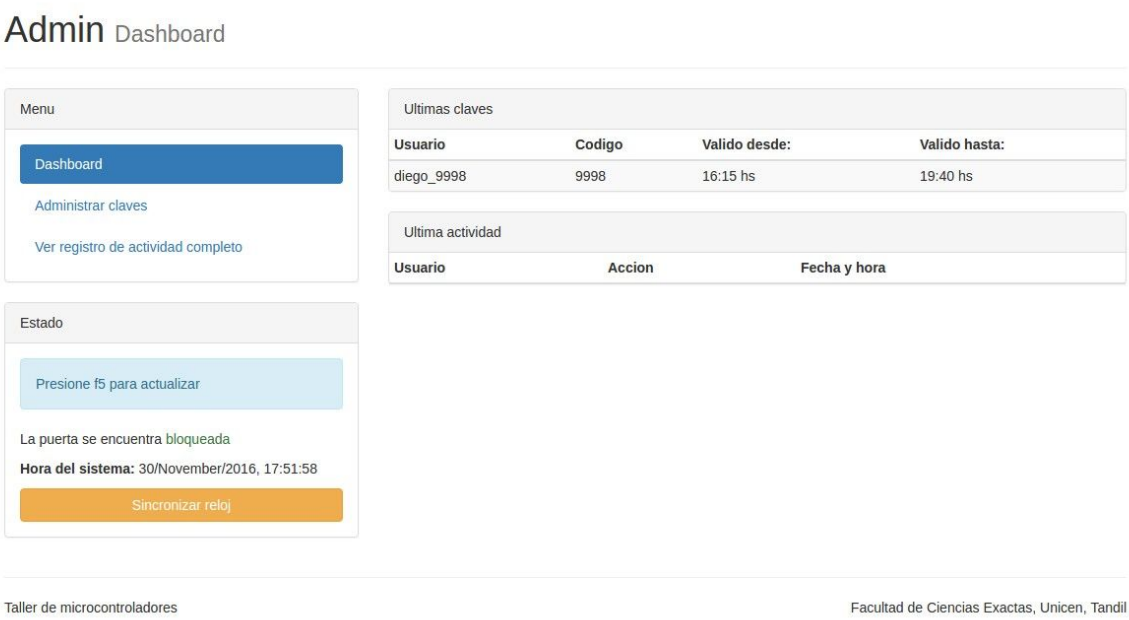
En esta partición entonces creamos una carpeta llamada “web” con el código del proyecto dentro de la misma.

Por ejemplo deberíamos ver el archivo “MEMORIA_SD/web/app.js” dentro de la tarjeta, sin subcarpetas intermedias.

Accediendo al servidor web

Para acceder al servidor web debemos ingresar a http://IP.DE.LA.PLACA:3000
Notar que el mismo es levantado al iniciarse la placa, solo si se cargó el archivo .ino correspondiente.

El servidor web se debe ver como en la siguiente imagen:



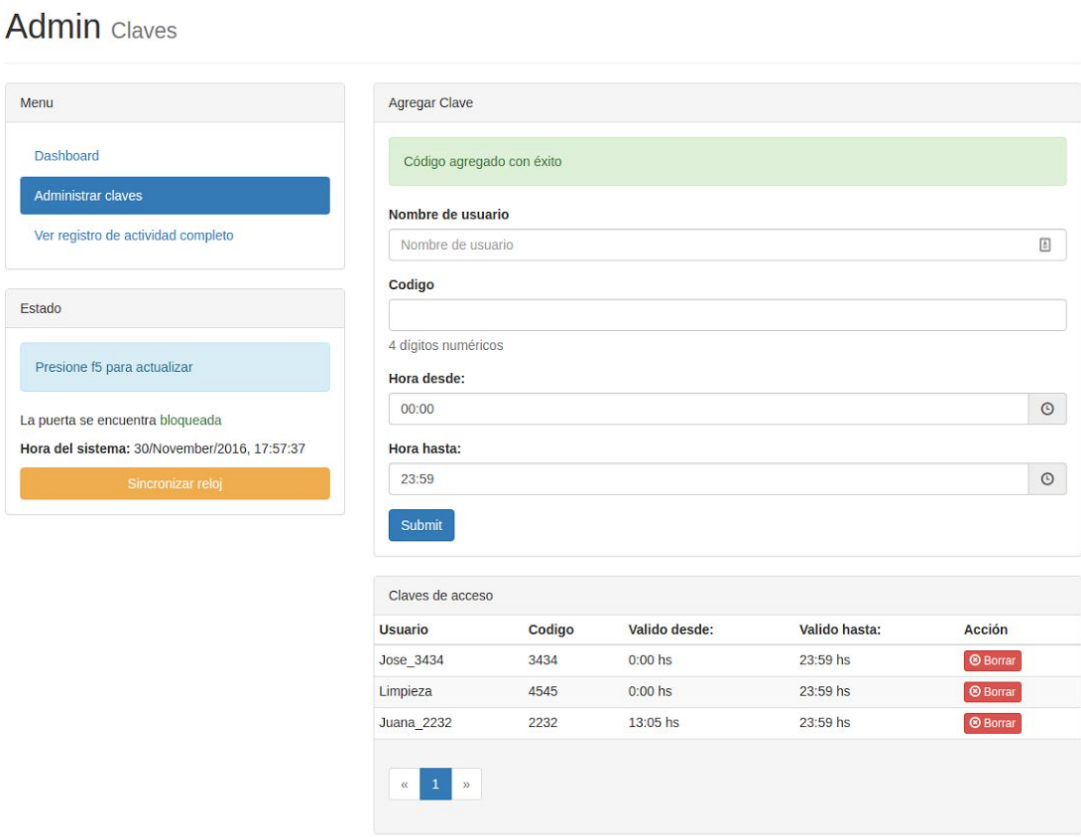
Al encender la placa, debemos presionar el botón “Sincronizar reloj”, sino los rangos horarios no seran consistentes.
El panel de administración cuenta con 3 secciones:

1. *Dashboard*

Es la sección que se muestra al ingresar al servidor web. Contiene la barra lateral con el menú e información, y muestra las 3 últimas claves creadas junto a las 3 últimas actividades registradas

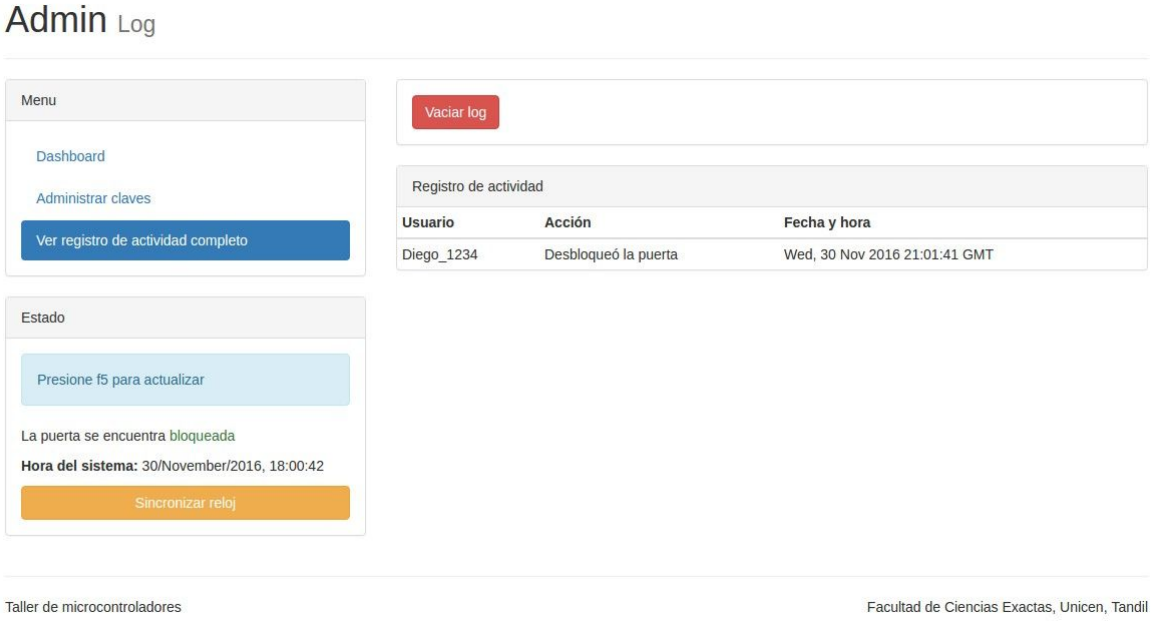
2. *Claves*

Esta sección permite crear y borrar códigos



3. *Log*

Permite ver el registro de actividad completo



Solución de problemas frecuentes

Arduino IDE no detecta la placa

Luego de instalar los drivers y reiniciar en modo normal (windows) es posible que estos se desasocien del hardware correspondiente. Para solucionar esto debemos instalar nuevamente los drivers pero sin la necesidad de reiniciar la pc en modo “Deshabilitar el uso obligatorio de controladores firmados”.

La herramienta de actualización de firmware se “congela” antes de terminar

La herramienta de actualización presenta fallos en distribuciones de Linux. Se deberá cambiar a windows.

No puedo acceder por ssh

Para acceder por ssh debemos estar seguros de conocer la dirección IP que tomo la placa. En algunas ocasiones suele ocurrir que las conexiones wifi y ethernet de la pc presentan conflictos dado que las rutas de red no se encuentran bien definidas. Si no se sabe como configurar esto, se recomienda deshabilitar las interfaces de red que no se estén utilizando para conectarse a la placa Galileo.
Si aún no funciona, debemos probar hacer ping a la placa. Si esta no responde, el problema sigue estando en la configuración de red o en la conexión misma.
Si la placa responde, es probable que la imagen de Linux cargada en la memoria micro SD no haya booteado, por lo tanto el servicio ssh no estara disponible.

No hay suficiente espacio en la memoria microSd para subir el proyecto

En este caso debemos eliminar los archivos que no son necesarios.
Dentro de la carpeta web podemos descartar la carpeta “ino” y “docs”.
Dentro de las carpetas web/assets/bower_components/* podemos eliminar todos aquellos archivos que no pertenezcan a las subcarpetas “dist”.
Recordar que si se ha ejecutado el comando “npm install” sobre la carpeta web, este habrá descargado dependencias que no son necesarias para la ejecución de la aplicación, solo para el desarrollo. Por esto debemos tener dos copias del proyecto, una para desarrollo y otra para subir a la tarjeta micro SD

Se decidió usar otra carpeta / partición para subir el código y ahora no funciona

Esto sucede porque existen referencias a la carpeta /media/card dentro del código. Se deberán corregir las mismas en el archivo .ino, app.js y storage.js

No funciona el servidor web

Como primer paso debemos verificar que no se trate de alguno de los problemas mencionados anteriormente (Falta de conectividad, codigo subido a una partición diferente).
Luego podemos acceder a la placa por ssh, y ver en la lista de procesos si se encuentra en ejecución “node app.js”
Para ver los procesos ejecutamos el comando “ps”
Si no existe el proceso, entonces el archivo .ino no fue cargado correctamente o no apunta a la carpeta correcta donde se encuentra el servidor web. Verificar que exista el archivo /media/card/web/app.js
Si existe el proceso, debemos detenerlo (kill <PID>) donde PID es el id del proceso tal cual figura en el listado devuelto por el comando ps, y luego ejecutarlo manualmente para ver que errores arroja.
Para esto último nos paramos en la carpeta “web” y ejecutamos “node app.js”
Luego de unos segundos deberíamos ver un mensaje de error.

El servidor web funciona pero solo muestra un mensaje “It works”

Debemos agregar el puerto :3000 a la dirección url.