Comunicación SCADA entre Arduino y Siemens WinCC a través de Modbus TCP/IP para simular el Control de Presión de una Válvula

Gonzalo H. Domínguez gonzalohernandominguez@gmail.com

Universidad Tecnológica Nacional, Facultad Regional La Plata
La Plata (1900), Argentina

Contexto

En el marco del Laboratorio CODAPLI (*Proyecto de Investigación en Codiseño de Hardware y Software para Aplicaciones en Tiempo Real*), perteneciente al Departamento de Ingeniería en Sistemas de Información de la Universidad Tecnológica Nacional - Facultad Regional La Plata; dentro del contexto del Proyecto de Investigación denominado "*Codiseño Aplicado con redes de sensores inalámbricos en tiempo real*". Se realiza el presente trabajo con el objetivo de poner en relación los conceptos aprendidos durante su desarrollo.

Resumen

La necesidad de las redes industriales en la automatización de un proceso o planta productiva, radica en la toma de decisiones que se debe realizar cuando se eligen los diferentes tipos redes, desde la base operativa hasta la gerencial, utilizando lineamientos de calidad, planeamiento de la producción, demanda de insumos, seguridad del personal, gestión del mantenimiento, diseño de producto, rentabilidad y competitividad. Es fundamental considerar las condiciones que hacen idónea su implementación, en el contexto de las situaciones que generalmente no son analizadas con la profundidad necesaria. El diseño de los diferentes tipos de redes industriales en la implementación de un sistema automatizado, sostendrá el nivel de automatización, utilizado por un determinado proceso industrial, el que lo hace operable de forma óptima. Esto se debe a que el proceso con esta base podrá contar con diferentes aplicaciones, no sólo en la planta para el personal de producción, sino también en la fase de mantenimiento, la gestión de calidad, la logística, el planeamiento de producción y la seguridad con la que se trabaja. Un factor importante a tener en cuenta son las normas de calidad existentes para los diferentes giros que se producen en la industria en la cual es fundamental el manejo y el registro de la información.

El objetivo de este trabajo es conocer cuáles son los protocolos que se utilizan en el ámbito industrial, y utilizar específicamente uno de ellos (en este caso MODBUS TCP/IP), para simular el control de presión de una válvula. Se utilizará una comunicación SCADA entre un Arduino y el software Siemens WinCC, para así poder visualizar el flujo de presión que se produce en la máquina usada en el proceso industrial.

Palabras Clave: *SCADA, Arduino, Modbus, Protocolos Industriales, Procesos Industriales, Automatización*

1. Introducción

La *automatización industrial* es el uso de sistemas o elementos computarizados y electromecánicos para fines industriales. Al ser una disciplina más amplia de la ingeniería que un sistema de control, abarca la instrumentación industrial, que incluye los sensores, los transmisores de campo, los sistemas de control y supervisión, los sistemas de transmisión y recolección de datos y las aplicaciones de software en tiempo real para así poder supervisar y controlar las operaciones de plantas o los procesos industriales [1].
La comunicación en las plantas industriales se ha hecho imprescindible en la industria moderna. Muchos sistemas están conformados por equipos de diferentes fabricantes y funcionan en diferentes niveles de automatización. Pese a que pueden estar distanciados entre sí, a menudo se desea que trabajen en forma coordinada para un resultado satisfactorio del proceso. El objetivo principal es la comunicación totalmente integrada en el sistema.

Esto reporta la máxima flexibilidad y permite integrar sin problemas productos de otros fabricantes a través de las interfaces software estandarizadas. Esta integración total se conoce como CIM (**Computer Integrated Manufacturing**), que se puede observar en la *Figura 1* [2]. El concepto CIM, en sí mismo, no indica la aplicación de ninguna tecnología o procedimiento en particular, sino la aplicación de ciertas técnicas con una visión integradora. Podría definirse como: "*una metodología de trabajo y una filosofía de diseño de los sistemas de automatización, producción y gestión orientados a la mejora de los niveles de calidad y a la optimización en los procesos de fabricación*" [3].

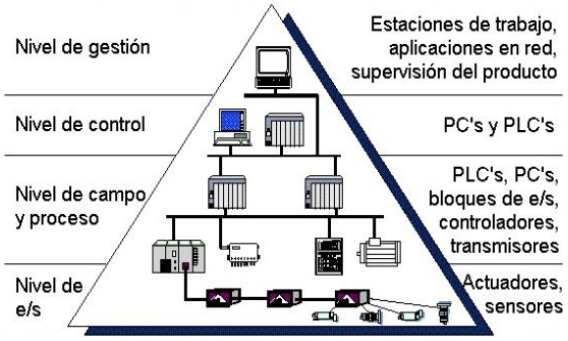

Figura 1: Pirámide CIM [2]

- **Nivel de E/S**: Es el nivel más próximo al proceso [2]. Está formado por los elementos de medida (sensores) y mando (actuadores) distribuidos en una línea de producción [4].
- **Nivel de campo y proceso**: En este nivel se sitúan los elementos capaces de gestionar los actuadores y sensores del nivel anterior, tales como autómatas programables o equipos de aplicación específica, basados en microprocesadores como robots, máquinas herramienta o controladores de motor [4].
- **Nivel de control**: Este nivel es el encargado de enlazar y dirigir las distintas zonas de trabajo [2]. Todos los dispositivos de control existentes en planta es posible monitorizarlos mientra exista un sistema de comunicación adecuado. Este debe ser capaz de comunicar estos elementos con otros tipos de dispositivos dedicados a la gestión y supervisión, los que habitualmente están constituidos por computadoras o sistemas de visualización tales como pantallas industriales. En este nivel es posible visualizar cómo se están llevando a cabo los procesos de planta y, a través de entornos **SCADA** (*Supervisión, Control y Adquisición de Datos*), poseer una "*imagen virtual de la planta*" de manera que ésta se pueda recorrer de manera detallada; o bien mediante pantallas de resumen ser capaces de disponer de un "*panel virtual*" en el cual se muestren las posibles alarmas, fallos o alteraciones en cualquiera de los procesos que se llevan a cabo. Mediante este tipo de acciones es posible disponer de acceso inmediato a cada uno de los sectores de la planta. Para ello, resulta imprescindible la conexión con el nivel de control mediante *buses de campo* [4].
- **Nivel de gestión**: Es el nivel más elevado y se encarga de integrar todos los niveles de la pirámide en una estructura de fábrica, e incluso de múltiples factorías [2]. El nivel de gestión estará principalmente constituido por computadoras, ya que se encuentra más alejado de los procesos productivos [4]. Estas estaciones de trabajo hacen de puente entre el proceso productivo y el área de gestión, en el cual se supervisan las ventas, stocks, entre otros [2]. De hecho, en este nivel no es relevante el estado y la supervisión de los procesos de planta, en cambio, sí adquiere importancia toda la información relativa a la producción y su gestión asociada. Se deduce que, a través del nivel de control es posible obtener información global de todos los niveles inferiores de una o varias plantas [4].

En el desarrollo de este trabajo nos centramos directamente en la parte inferior de la pirámide de automatización, donde se encuentran los llamados *dispositivos de campo* que actúan directamente sobre el proceso productivo [5].

1.1 Buses de Campo

Un bus de campo, también denominado red de campo, es, en líneas generales, "*un sistema de dispositivos de campo (sensores y actuadores) y dispositivos de control, que comparten un bus*

digital , el cual se yergue como una serie bidireccional que transmite informaciones entre los dispositivos, sustituyendo a la convencional transmisión analógica punto a punto". Permiten sustituir el cableado entre sensores-actuadores y los correspondientes elementos de control [6]. En este trabajo utilizamos el protocolo denominado **Modbus** debido a que es de público acceso y por su gratuidad. Además resulta sencillo de implementar debido a su escaso desarrollo [7].

1.2 Modbus

Modbus es un protocolo de transmisión para *sistemas de control y supervisión de procesos* (SCADA) con control centralizado, que puede comunicarse con una o varias *estaciones remotas* (RTU) con la finalidad de obtener datos de campo para la supervisión y control de un proceso. La Interfaces de Capa Física puede estar configurada en: *RS-232, RS-422, RS-485* [8]. En Modbus los datos pueden intercambiarse en dos modos de transmisión [5]:

- **Modo RTU**: (*Remote Terminal Unit*), donde se envían 4 caracteres hexadecimales (4 bits cada uno) para cada mensaje. Esta opción es más empleada en transmisiones inalámbricas.
- **Modo ASCII**: enviando dos caracteres (2 bytes) para cada mensaje, pudiendo haber hasta 1 segundo de tiempo de diferencia entre ellos.

Actualmente se está impulsando el empleo de MODBUS sobre **TCP/IP** para aprovechar las infraestructuras que se están implantando en Internet, además de usar protocolos industriales mientras se emplean las mismas líneas y se empaquetan mensajes MODBUS dentro de los paquetes TCP/IP. Para esto son necesarios unos módulos de encapsulado y desencapsulado que conecten con módulos tradicionales MODBUS [5].

1.3 Arduino

Arduino es una compañía de fuente y hardware abiertos, así como un proyecto y comunidad internacional que diseña y manufactura placas de desarrollo de hardware para construir dispositivos digitales e interactivos que puedan detectar y controlar objetos del mundo real [9]. En este trabajo se utilizó la placa de Arduino denominada *Mega* (*Figura 2*) como transmisor de las tramas MODBUS, que luego, a través del protocolo mencionado, llegan a la computadora (receptor), siendo el software WinCC aquel que se encarga de monitorear y controlar el proceso.

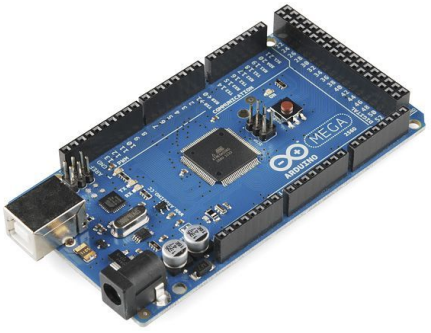

Figura 2: Placa Arduino Mega

1.4 Siemens WinCC

SIMATIC WinCC es un sistema de supervisión y adquisición de datos (SCADA) e interfaz hombre-máquina (HMI) de *Siemens*. Los sistemas SCADA se utilizan para monitorear y controlar los procesos físicos involucrados en la industria y la infraestructura a gran escala y en largas distancias. SIMATIC WinCC se puede utilizar en combinación con los controladores Siemens. WinCC está desarrollado para el sistema operativo Microsoft *Windows* [10].

2. Elementos del Trabajo y metodología

Para el desarrollo del presente proyecto, y como metodología de base, utilizamos los siguientes elementos de trabajo con el objetivo de poder simular el control de presión de una válvula [11]:

- Tarjeta Arduino Mega.
- Arduino Ethernet Shield W5100.
- Cable UTP.
- Cable USB para Arduino.
- Tarjeta Protoboard.
- Software IDE de Arduino.
- Diodo Led de cualquier color (Común).
- Resistencia de 220 / 330 Ohms.
- Potenciómetro para conectar en un Protoboard, de 10 Kilos.

En una primera instancia se precisaron las librerías de Modbus para Arduino, utilizando el hardware como esclavo, y la computadora como maestro [12]. Luego se fue modificó el código para así adaptarlo

a las necesidades de este proyecto. Para explicar lo anterior, se cita un ejemplo: todo aquello que estaba relacionado con las comunicaciones en serie, terminaban sido eliminadas, y, como resultante sólo se hubo tenido en cuenta lo referente a las comunicaciones analógicas.

A partir de aquí, explicaremos cómo se realizó la implementación de la trama Modbus sobre TCP/IP [13].

Para comenzar el programa se incluirán todas las librerías necesarias, a saber:

```
#include <SPI.h>
#include <Ethernet.h>
#include "MgsModbus.h"

MgsModbus Mb;
```

Luego configuramos la red Ethernet, dependiendo de la configuración de la MAC y de la red local:

```
byte mac[] = {0x14, 0x18, 0x77, 0xB2, 0xB8, 0x4B };
```

Las siguientes instrucciones detallan la dirección IP de nuestro Arduino, la puerta de enlace y la máscara utilizada:

```
IPAddress ip (169, 254, 97, 239);
IPAddress gateway (192, 168, 0, 1);
IPAddress subnet (255, 255, 0, 0);
```

En el *Setup* de nuestra programa, podemos observar que inicializamos un pin del Arduino como salida. El mencionado pin es el que vamos a utilizar para obtener los datos del movimiento de nuestro potenciómetro.

```
void setup ()
{

  // serial setup
  pinMode (2,OUTPUT);
```

Luego inicializamos la interface de Ethernet:

```
  Ethernet.begin (mac, ip, gateway, subnet);
```

Acto seguido se nos presentan aquellas direcciones en donde se almacenan los datos. En esta oportunidad, se observan dos tipos de datos: por un lado la dirección 0 , y por el otro la dirección 1, las cuales corresponden al registro 40001 y al registro 40002

```
  Mb.MbData[0] = 0; // Holding Reg 40001
  Mb.MbData[1] = 0; // Holding Reg 40002
```

El problema actual planteaba la imposibilidad de poder leer el dato que mencionaba el potenciómetro en WinCC. Para lograr resolver la problemática, se equiparó el registro 40001 a la salida analógica asociada al pin 2 anteriormente mencionado..En consecuencia el dato resultante y legible, será el que se guarde en la variable MbData:

```
void loop()
{
   Mb.MbData[0] = analogRead (2);
   digitalWrite(2, bitRead (Mb.MbData[1],0));

//  Mb.MbmRun();
  Mb.MbsRun();
}
```

Ahora bien, si quisiéramos recepcionar ese dato que se ha guardado en el registro, utilizaremos la función *digitalWrite* para así poder escribir el valor leído desde el pin antes mencionado. Para realizar el guardado en el registro 40002 , mientras se utiliza la función *bitRead*, se colegirá de la lectura que el bit será de valor 0. Se acaba de describir el código para la comunicación Modbus entre el hardware Arduino y WinCC.

Acto seguido nos plantearemos dirigirnos a WinCC, como se observa en la *Figura 3*, para así configurar un "*Nuevo Proyecto*" :

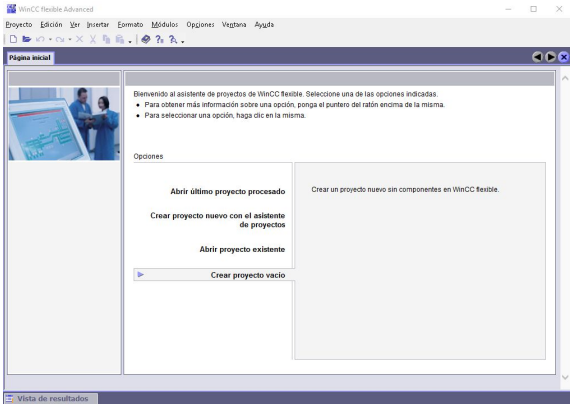

Figura 3: Interfaz de Siemens WinCC Flexible

Cuando nos decidimos a emprender el presente proyecto, lo planteamos visualizado de la siguiente manera (*Figura 4*), como un árbol con ramas que se desprenden:

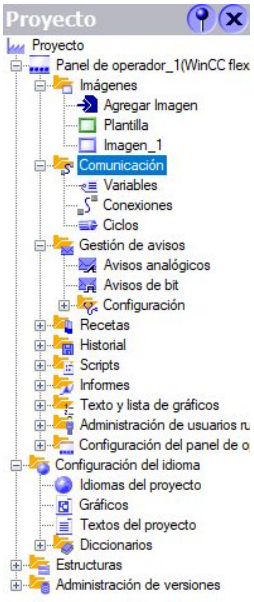

Figura 4: Barra de Navegación en WinCC

Luego nos dirigimos al ítem "*Comunicación*" y al subítem "*Conexiones*" (Figura 5). En este caso se configuraron dos conexiones de tipo "*Modicon MODBUS TCP/IP*", una por cada registro:

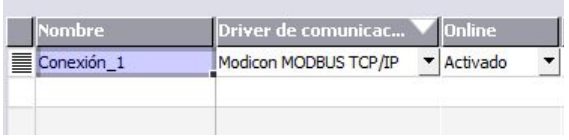

Figura 5: Conexiones en WinCC

En la misma sección, como vemos en la *Figura 6*, podemos configurar el Tipo de CPU del Arduino, la IP del Arduino (Servidor), el Puerto y la Dirección del Esclavo Remoto.

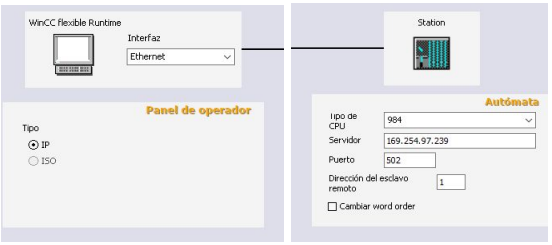

Figura 6: Configuración del Maestro/Esclavo en WinCC

Si las conexiones hubieran utilizado el protocolo MODBUS RTU, se hubiera configurado de forma similar, con la diferencia que en la pestaña "Drive de comunicación" tendríamos que haber elegido la opción "Modicon MODBUS" (*Figura 7*), para luego seleccionar el tipo de interfaz física (RS 232, RS 422 ó RS 485), junto con la Velocidad de transferencia, la Paridad, los Bits de datos y los Bits de parada, como se puede observar en la *Figura 8* y *9*, respectivamente:

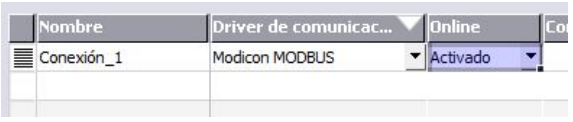

Figura 7: Modbus RTU en WinCC

Figura 8: Configuración del "Panel de operador" para Modbus RTU en WinCC

Figura 9: Configuración del protocolo y del autómata en WinCC

En el apartado "*Variables*" (*Figura 10*) se configuraron las dos variables (HR_4001 y HR_4002) que fueron usadas, las cuales fueron leídas desde la **Conexión_1** definida en "*Conexiones*". Cada una de ellas posee un enlace que la conduce a su dirección correspondiente, 40001 y 40002 respectivamente.

Figura 10: Variables en WinCC

A partir de allí nos enfocamos en la solapa "*Imagen*" e ideamos una nueva imagen. En este punto creamos una gráfica para poder representar el valor obtenido (*Figura 11*). Luego nos dirigimos a "*Curva*", y para que se nos muestre en pantalla lo descripto, debimos modificar la configuración en la variable definida anteriormente a HR_4001.

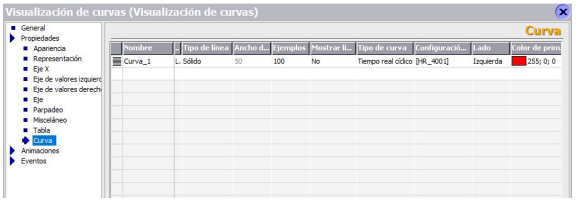

Figura 11: Visualización de Curvas

Para abordar la configuración de los botones, los cuales simulan el abrir y cerrar la llave de la válvula, nos conducimos hacia los siguientes comandos como "Botón, "Eventos" y "Pulsar". Es allí donde nos aparece la opción "ActivarBitEnVariable" como podemos observar en la *Figura 12*, y seteamos a la variable HR_4002 como entrada.

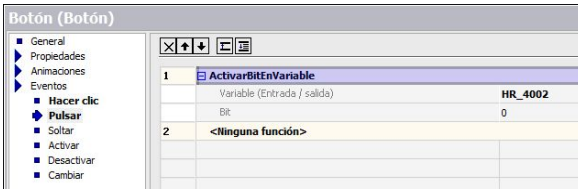

Figura 12: Configuración de Botones

De esta manera se realiza la conexión entre Arduino y WinCC. Una vez cargado y configurado todo el proceso, procedemos a la conexión del Arduino a través de un cable Ethernet con nuestro equipo (Figura 13).

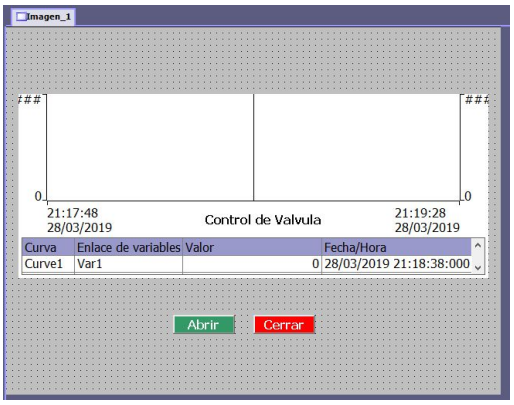

Figura 13: SCADA del Proyecto

3. Resultados

En la siguiente figura podemos observar cómo se representa el valor del potenciómetro en SCADA y cómo varía dicho valor en la gráfica observada:

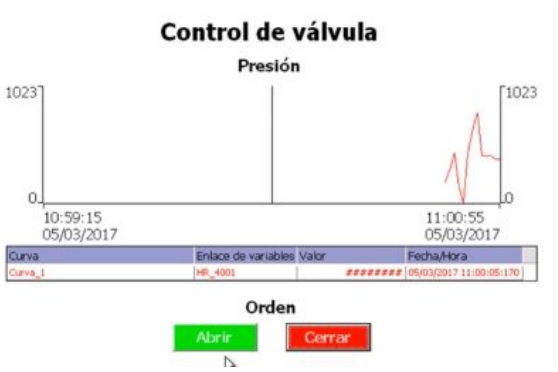

Figura 14: Resultados en Tiempo Real

Podemos observar que el valor obtenido se representa al girar el potenciómetro. Al proceder a apretar los procesos de "Abrir" y "Cerrar", logran la lógica que se hubo planteado al pensar el proyecto.

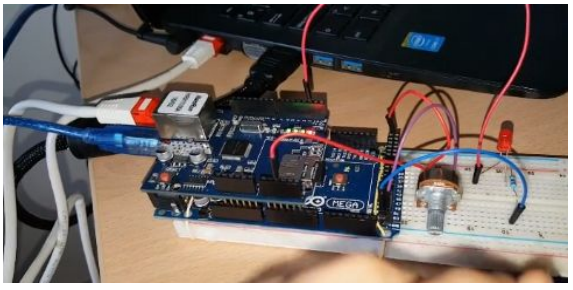

Figura 15: Conexión de los elementos

Es verdaderamente satisfactorio el resultado, y también poder hacer una descripción acabada del mismo. La relación entre el hardware Arduino, la formulación de objetivos y un desarrollo de

pensamiento, logra como resultante la comunicación Modbus entre el Arduino y el software utilizado.

4. Conclusión y trabajo futuro

Es de suponer que este trabajo, sea sólo el puntapié inicial en un sinfin de proyectos de investigación en el área, que presupone adentrarse en el mundo de las redes de comunicación industriales.
Se espera poder ahondar en el conocimiento de otros protocolos industriales que permitan una comunicación segura y confiable de aparatos electrónicos. El abordaje de las ideas anteriormente expuestas redundará en una mayor sapiencia, mayor reflexión y mayor compromiso con el trabajo investigativo.

Referencias

[1] Automatización Industrial. Wikipedia [online]. Disponible: https://es.wikipedia.org/wiki/Automatización_industrial, Consulta: 12 de Abril del 2019.

[2] J. A. Sirgo Blanco, "*Comunicaciones Industriales*", Asignatura Ingeniería Electrónica y Automática, Área de Ingeniería de Sistemas y Automática, Departamento de Ingeniería Eléctrica, Universidad de Oviedo. Año 2014.

[3] F. Pascual Morales, "*Sistemas Industriales Distribuidos: C.I.M. (Computer Integrated Manufacturing). Una filosofía de automatización*", Departamento de Ingeniería Electrónica, Universidad de Valencia.

[4] F. J. Menchón Ruiz, "*Configuración y puesta en marcha de una red de autómatas programables basada en PROFIBUS, MPI y GSM para el control y monitorización de módulos de fabricación flexible*", Proyecto Fin de Carrera, Departamento de Ingeniería de Sistemas y Automática, Escuela Técnica Superior de Ingeniería Industrial, Universidad Politécnica de Cartagena. Año 2006.

[5] A. Rosado, "*Redes de Comunicaciones Industriales*", Sistemas Industriales Distribuidos, Universidad de Valencia.

[6] J. M. Hurtado Torres, "*Introducción a las Redes de Comunicación Industrial*", Módulo de Comunicaciones Industriales, Departamento de Electricidad-Electrónica I.E.S Himilce-Linares, Ciclo Superior de Automatización y Robótica Industrial.

[7] Modbus. Wikipedia [online]. Disponible: https://es.wikipedia.org/wiki/Modbus, Consulta: 12 de Abril del 2019.

[8] A.I.E (Asociación de la Industria Eléctrica-Electrónica), "*Protocolos de Comunicaciones Industriales*", Artículo. Año 2006. Disponible: http://www.aie.cl/files/file/comites/ca/articulos/agosto-06.pdf

[9] Arduino. Wikipedia [online]. Disponible: https://es.wikipedia.org/wiki/Arduino, Consulta: 12 de Abril del 2019.

[10] WinCC. Wikipedia [online]. Disponible: https://en.wikipedia.org/wiki/WinCC, Consulta: 12 de Abril del 2019.

[11] Manual de Prácticas de Arduino [online]. "*Controlar el brillo de un LED con un potenciómetro*". Disponible: https://manualarduinos52.wordpress.com/2013/12/06/practica-3/

[12] MyArduinoProjects.com [online]. "*Modbus library*". Disponible: http://myarduinoprojects.com/modbus.html

[13] A. Romero, "*Comunicación Modbus Arduino y Siemens WinCC*". Disponible: https://www.youtube.com/channel/UCTX_hUB56dbf6NHPuKYUUAw/videos

Agradecimientos

Al Laboratorio CODAPLI de la Universidad Tecnológica Nacional, Facultad Regional La Plata, por su guía y apoyo incondicional para llevar a cabo este trabajo: Ing. Omar Rodriguez, Ing. José Rapallini e Ing. Marcelo Zabaljauregui por su aporte.

Datos de Contacto

Gonzalo Hernán Domínguez. Universidad Tecnológica Nacional - Facultad Regional La Plata. Código Postal: 1900. La Plata. Email: gonzalohernandominguez@gmail.com