AVANCES EN EL DESARROLLO E IMPLEMENTACIÓN PRÁCTICA DE UN SISTEMA SCADA DIDÁCTICO PARA CONTROL Y SUPERVISIÓN DE NIVEL DE LÍQUIDO[1]

Jonathan D. Krucheski[2]; Franco Olsson[2]; Ricardo Korpys[3]; Guillermo A. Fernández[3]

[1] Experiencia de cátedra, Trabajo Final de Proyecto y Diseño Electrónico, carrera Ingeniería Electrónica, Facultad de Ingeniería, UNaM.
[2] Estudiante de Ingeniería Electrónica, jonathankrucheski@gmail.com; francoolsson1995@gmail.com
[3] Tutor, Ingeniero Electrónico, korpys@fio.unam.edu.ar, fernandez@fio.unam.edu.ar

Resumen

Este artículo se presenta los avances del trabajo final realizado en la asignatura Proyecto y Diseño Electrónico, el cual trata sobre el diseño y la implementación práctica de un sistema didáctico de control y supervisión de nivel de líquido de un tanque, para verificar distintas acciones de control, comprobar la comunicación de datos entre dos sistemas digitales y experimentar con un sistema SCADA (*Supervisory Control and Data Acquisition*). El trabajo final plantea la construcción del tanque, el sistema de desagote y de sensado de nivel de líquido, también de la unidad terminal remota (RTU) que posee el algoritmo de control correspondiente y el desarrollo de la interfaz hombre-máquina a través del software SCADA seleccionado. Con el mismo se puede realizar demostraciones de las diferentes acciones de control implementadas en la RTU, como así también la supervisión del nivel de líquido la variable controlada. Actualmente el proyecto se encuentra en desarrollo y en este articulo contiene un resumen de la revisión bibliográfica, también se tratan cuestiones generales de diseño y se presenta un circuito RTU de pruebas utilizado para llevar a cabo ensayos a medida que se avanza con el diseño del RTU definitivo.

Palabras Clave: SCADA – Nivel de líquido – Control y supervisión.

Introducción

En la actualidad, la gran variedad de procesos que deben controlarse en las industrias, hace que sea fundamental contar con herramientas confiables para su supervisión y control. En este sentido los sistemas SCADA permiten supervisar y controlar varios procesos, a partir de una Unidad Terminal Maestra (MTU), que generalmente puede ser una PC, conectada a varias Unidades Terminales Remotas (RTU). De esta forma, a partir de las órdenes dadas por un operador humano a través de la MTU, el sistema SCADA puede actuar sobre los procesos tomado información de los mismos para efectuar su análisis, o bien actuando sobre estos para ejecutar alguna acción demanda. El desarrollo de numerosos softwares específicos para aplicaciones referentes a supervisión y control y el hecho de que están al alcance de cualquier empresa, hoy día ha aumentado la popularidad de los sistemas SCADA en las industrias (Pérez-López, 2015). En este contexto, el sistema propuesto en el presente trabajo, además de ser utilizado como una plataforma didáctica en asignaturas orientadas al control y la automatización, actualmente dictadas en la carrera de Ingeniería Electrónica, también permitirá que el Laboratorio de Electrónica cuente con una plataforma destinada a divulgar los sistemas SCADA y también promocionar las actividades realizadas por los alumnos durante el cursado de la carrera.

En la Figura 1 se aprecia un diagrama en bloques del sistema propuesto, en el mismo se observan las siguientes partes:

- **Unidad Terminal Maestra (MTU):** La misma está constituida por una computadora personal que posee instalado el software SCADA seleccionado, el cual permite desarrollar la interfaz hombre-máquina (HMI) que comunica al usuario con el proceso que se desea supervisar y controlar modificando los parámetros de control correspondientes. Por ejemplo, mediante la HMI puede establecerse un nivel de líquido determinado como *setpoint* del sistema de control, abrir o cerrar la electroválvula para desagotar el tanque y/o variar los parámetros del controlador embebido en la RTU entre otros.
- **Unidad Terminal Remota (RTU):** Esta parte del sistema corresponde a un circuito que incorpora los recursos de hardware y software necesarios para establecer la comunicación con la MTU de un protocolo de comunicación estándar seleccionado. La RTU también permite recibir y procesar los datos del sensor de nivel y calcular la magnitud de la acción de control necesaria para gobernar la bomba (mediante modulación de ancho de pulso, PWM) y así implementar el control de nivel de líquido del tanque. Además, el circuito de la RTU también permite que el usuario actúe sobre la electroválvula de salida para abrirla o cerrarla y así el usuario pueda manipular el desagote del tanque. Cabe destacar que esta unidad cuenta con todos los circuitos necesarios para la adaptación de las señales de entrada y de salida asociadas a los sensores y actuadores del sistema propuesto.
- **Bomba:** El actuador del sistema lo constituye una bomba controlada a través de la señal PWM proporcionada por la unidad terminal remota. Esta señal actúa sobre su tensión de alimentación para que el controlador pueda modificar el caudal de líquido que ingresa al tanque y así compensar los efectos de la pérdida que ocasiona la electroválvula asociada a la salida de desagote.
- **Tanque:** El tanque utilizado en el sistema posee una capacidad aproximada de 10 litros, para facilitar su manipulación en el laboratorio. Además, posee soportes para una fácil instalación del sensor y la bomba.
- **Sensor de nivel:** Este sensor es el encargado de medir el nivel de líquido en el tanque. Esta medición es utilizada por la RTU para el cálculo de la acción de control necesaria.
- **Electroválvula**: La electroválvula se utiliza para controlar la salida de líquido del tanque y así efectuar una perturbación sobre la variable controlada (nivel de líquido).

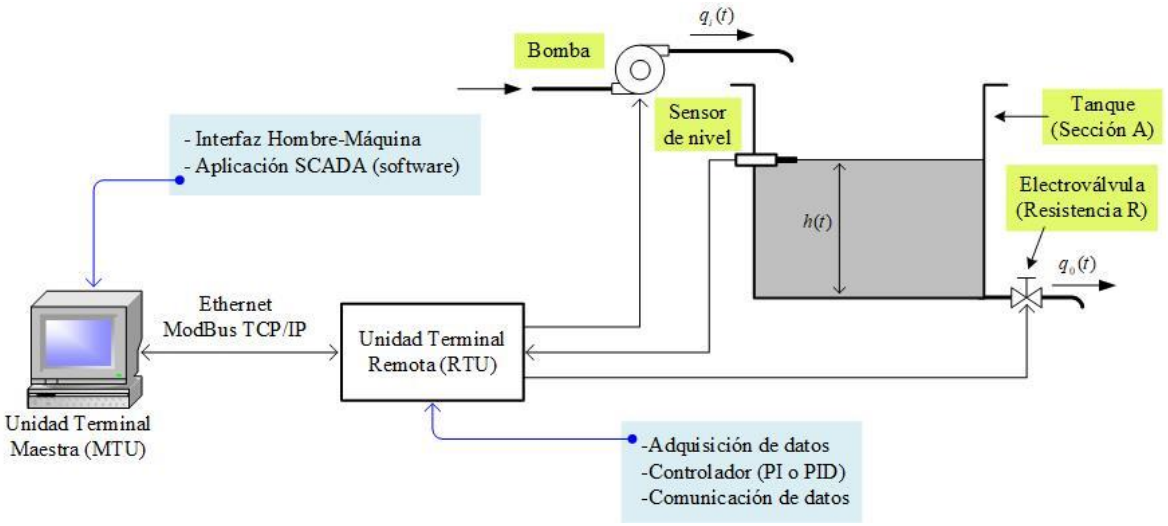

Figura 1. Esquema general del sistema didáctico de control y supervisión de nivel de líquido.

Debido a que el sistema propuesto es del tipo didáctico, tanto el software como el hardware utilizado son de fácil utilización para los estudiantes, así como la RTU utiliza el microcontrolador Atmega328p, el mismo utilizado en la plataforma didáctica Arduino Uno. La selección de este dispositivo se basó en el análisis de su velocidad de procesamiento y también considerando ventajas relacionadas a su disponibilidad en el mercado nacional, facilidad de programación y la gran cantidad de librerías desarrolladas para la utilización de la misma en diferentes aplicaciones.

Por otro lado, la comunicación entre los dispositivos RTU y MTU se lleva cabo mediante el protocolo Modbus TCP/IP bajo el estándar Ethernet, debido a que el mismo es libre y de uso popular en aplicaciones industriales y a la índole didáctica del sistema propuesto, donde es menester que la comunicación sea compatible con las computadoras personales de los estudiantes. Cabe mencionar que en la web puede encontrarse una librería libre y gratuita, completamente funcional del protocolo Modbus TCP/IP para la plataforma Arduino Uno (Atmega328p).

En la siguiente sección se realiza una breve descripción del sistema planteado, haciendo énfasis en la RTU, cuyo diseño se está llevando a cabo en la actualidad.

Metodología

Este trabajo está en proceso de desarrollo, actualmente se está trabajando en dos aspectos del proyecto: hardware, que corresponde al diseño del circuito de la Unidad Terminal Remota (RTU), y software que involucra la selección del software SCADA y el desarrollo de los scripts necesarios para implementar la comunicación del RTU con la MTU.

El circuito de la RTU, consiste en una pequeña computadora que proporciona el soporte necesario para que la Unidad Terminal Maestra (MTU) sea capaz de comunicarse para acceder a la información sensada (en este caso la medición de nivel de líquido en el tanque) y accionar a distancia los elementos finales de control (en este caso la bomba y la electrovávula). La RTU es una unidad de adquisición y control de datos independiente, que tiene la capacidad de ejecutar programas sin la participación de la MTU del sistema SCADA, como así también de recibir información y ejecutar los procesos encomendados por esta última unidad (Chavarría Mesa, 2007). Debido a esto, la RTU es el enlace entre los distintos instrumentos de campo que son los que ejercen la automatización física del sistema, el sensado y el control (Pérez-López, 2015). En la Figura 2 se observa el diagrama en bloques de la RTU diseñada en este trabajo.

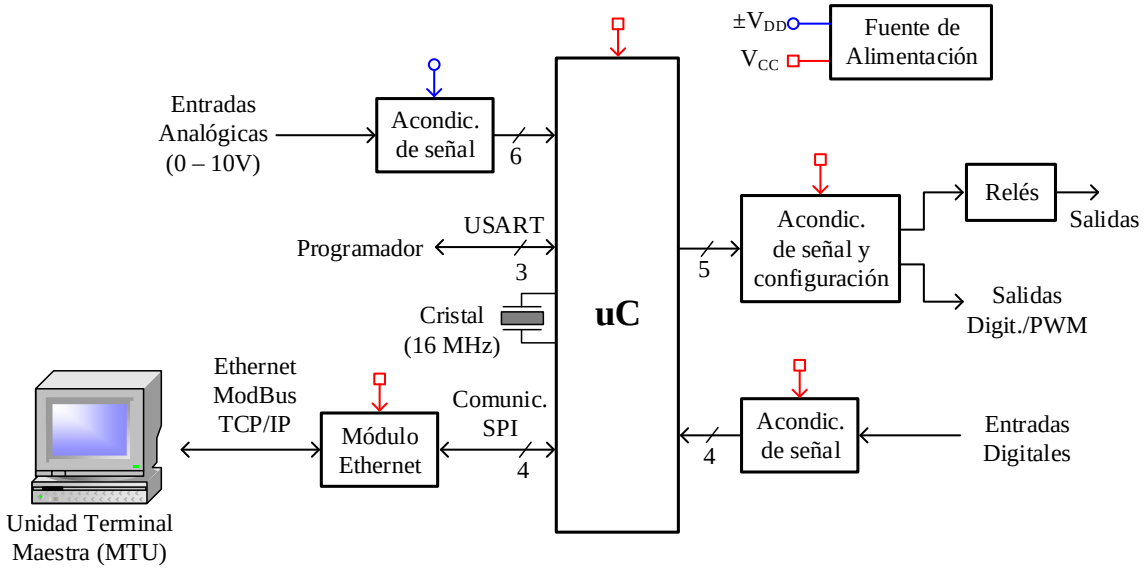

Figura 2. Diagrama en bloques de la Unidad Terminal Remota (RTU).

En la Figura 2 se puede observar que el bloque principal es el microcontrolador, el cual constituye el "cerebro" de la RTU. Su función es proporcionar (en tiempo real) a la aplicación SCADA residente en la MTU, la posibilidad de la exploración de entradas de información, su procesamiento y el almacenamiento de datos; siempre respondiendo a las peticiones de esta última unidad. Para este trabajo, debido a las consideraciones indicadas en la sección anterior, para la construcción de la RTU se ha seleccionado el microcontrolador Atmega328p de la firma ATMEL (ATMEL, 2016).

Como se puede apreciar en la Figura 2, la RTU también cuenta con entradas/salidas digitales y entradas del tipo analógicas, las cuales permiten al sistema SCADA tomar información de la máquina o proceso con el que interactúa, en este caso el nivel de líquido del tanque. Los bloques de acondicionamiento de señal indicados en la Figura 2, tienen la finalidad de modificar las señales externas a la RTU para obtener niveles adecuados de señal a las entradas/salidas del microcontrolador. Por otra parte, estos bloques protegen a las entradas de dicho dispositivo de eventuales sobretensiones, corrientes elevadas e inversión de polaridad. En cuanto a las salidas, además de la adaptación de señales mencionadas para las entradas, los bloques de acondicionamiento de señal permiten escoger el tipo de salida, siendo esta del tipo digital o del tipo relé electromecánico. Para las entradas analógicas, el acondicionamiento de señal no solo adecúa las señales a los niveles de tensión manejados por el microcontrolador, sino que también permiten una adaptación de impedancia (alta impedancia de entrada y baja de salida) para que las entradas analógicas de este dispositivo no interfieran con la medición.

Para la comunicación entre RTU y MTU, el microcontrolador se vale de un módulo Ethernet basado en el circuito integrado W5100, el cual provee a la RTU la capacidad de conectarse a la capa física, implementando mediante hardware la pila de protocolos TCP/IP y liberando al microcontrolador de la carga computacional que esto significa (WIZnet Co., 2009).

Por otra parte, el software SCADA seleccionado se denomina Elipse SCADA, es un software pago pero que posee una versión de evaluación, cuya única limitación es la cantidad de variables (Tags) que puede procesar.

A continuación, se habla sobre un módulo de pruebas desarrollado y se tratan cuestiones puntuales al software utilizado para realizar pruebas de comunicación entre RTU y MTU.

Resultados y Discusión

En la Figura 3 podemos ver el esquema de un circuito RTU de prueba (módulo de ensayos), que se ha desarrollado para ensayar parte del software vinculado a la comunicación de datos mediante el protocolo Modbus TCP/IP, la lectura de entradas analógicas y el accionamiento de salida digitales, como así también la elaboración de la HMI en el software Elipse SCADA. Este módulo de ensayos, cuenta con una conexión Ethernet, cuatro leds conectados a las salidas digitales del microcontrolador y dos potenciómetros conectados a las entradas analógicas del mismo los cuales son monitoreados por la HMI residente en una PC conectada a la misma red Ethernet que el módulo RTU.

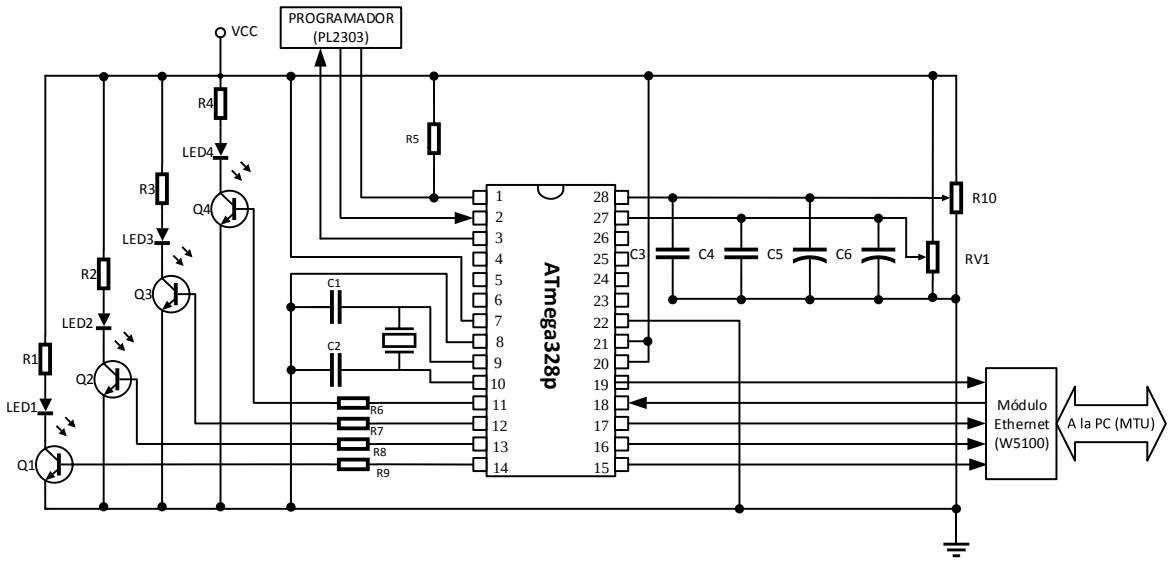

Figura 3. Esquema del circuito correspondiente a la RTU de prueba.

El HMI utilizado, fue desarrollado en el software Elipse SCADA, el mismo posee un gráfico que muestra los valores de las entradas analógicas correspondientes a los potenciómetros del módulo de ensayos y cuatro botones que permiten controlar el encendido o apagado de los leds que posee este circuito, en la Figura 4 podemos ver una captura de pantalla de dicha interfaz.

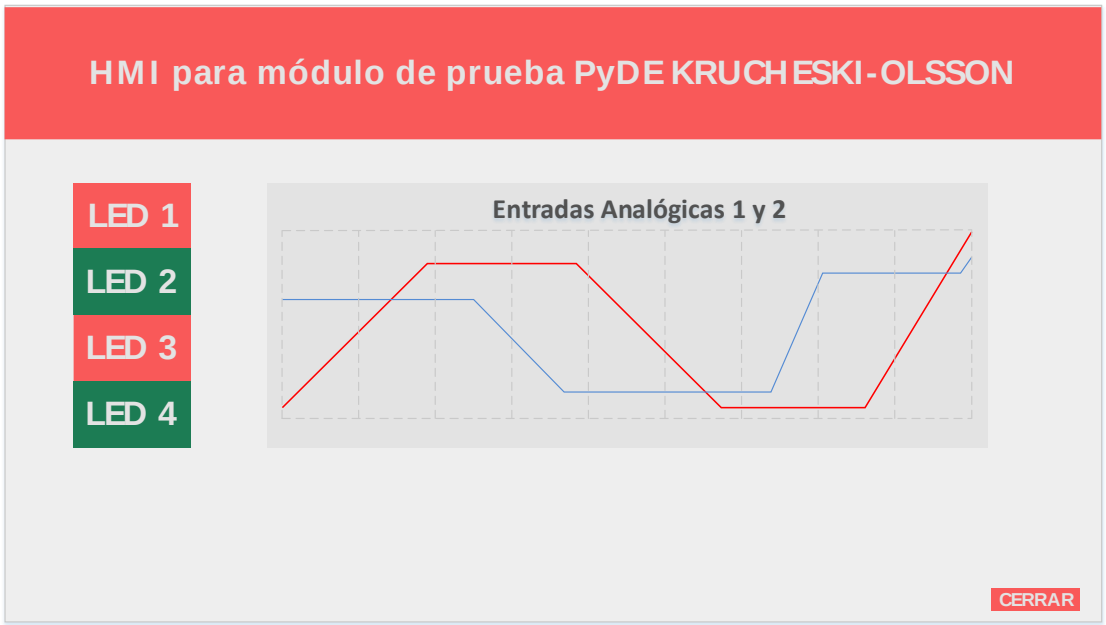

Figura 4. Interfaz desarrollada en el software Eclipse SCADA.

A continuación, se presenta un programa de ejemplo, implementado en el microcontrolador Atmega328p para verificar las cuestiones mencionadas anteriormente a través del módulo de ensayos. En la Figura 5 se puede apreciar la estructura inicial del programa mencionado, el cual, en primer lugar incluye las librerías utilizadas, las cuales cumplen con las siguientes funciones:

- SPI.h: Es utilizada para establecer la comunicación entre el modulo Ethernet y el microcontrolador a través de su módulo interno SPI.
- Ethernet: Se utiliza para acceder a las funciones del W5100 que posee el módulo Ethernet.
- MgsModbus-v0.1.1: Se encarga de armar unidad de datos (ADU) de la trama Modbus a partir de los datos enviados o pedidos por del usuario desde la HMI residente en la PC (MTU) (My Arduino Projects, s.f.).

Luego incluir las librerías mencionadas, se declaran e inicializan las variables utilizadas en el programa. Como podemos observar en la Figura 5, hay dos grupos de variables: las utilizadas por la librería Ethernet para definir la configuración de red del RTU y las utilizadas para indicar los pines de entrada/salida del microcontrolador que será utilizados.

```
// Incluye librerías:
//--------------------------------------------------------------------------
#include <SPI.h>        // Librería SPI para manejo del módulo Ethernet W5100
#include <Ethernet.h>   // Librería Eth. para manejo del módulo Ethernet W5100
#include "MgsModbus.h"  // Librería que genera/interpreta la trama Modbus TCP/IP
// Declaración e inicialización de variables
//--------------------------------------------------------------------------
MgsModbus Mb;
int inByte = 0;
// Ajustes de Ethernet (según MAC y red local):
byte mac[] = {0x90, 0xA2, 0xDA, 0x0E, 0x94, 0xB5 }; // Define MAC del RTU
IPAddress ip(10, 0, 0, 4);                          // Define IP del RTU
IPAddress gateway(10, 0, 0, 2);                     // Define gateway del RTU
IPAddress subnet(255, 255, 255, 0);                 // Define máscara red p/RTU
// Pines de E/S del ATMega328p:
int pinLed1 = 5;      // Pin para LED 1
int pinLed2 = 6;      // Pin para LED 2
int pinLed3 = 7;      // Pin para LED 3
int pinLed4 = 8;      // Pin para LED 4
int analogico5 = A5;  // Pin para entrada analógica 1
int analogico4 = A4;  // Pin para entrada analógica 2
```

Figura 5. Programa de ejemplo, librerías utilizadas y declaración de variables.

Una vez realizada todas las declaraciones necesarias, se ejecuta la función *setup()* indicada en la Figura 6. Esta función (ejecutada una única vez), es utilizada para configurar los pines del microcontrolador como entrada o salida, ajustar la velocidad (*baud rate*) de la comunicación serie, iniciar la interfaz Ethernet e inicializar los espacios de memoria Modbus que serán utilizadas (inicialización del vector *MbData[]* de la librería MgsModbus).

```
// Configuraciones generales de módulos del ATMega328p
//--------------------------------------------------------------------------
void setup()
{
// Configuración de I/O digitales:
  pinMode(pinLed1, OUTPUT);            // Define pinLed1 como salida
  pinMode(pinLed2, OUTPUT);            // Define pinLed2 como salida
  pinMode(pinLed3, OUTPUT);            // Define pinLed3 como salida
  pinMode(pinLed4, OUTPUT);            // Define pinLed4 como salida
```

```
10
11  // Configuración del módulo SPI:
12    Serial.begin(9600);                          // Ajuste del baud rate
13    Serial.println("Interfaz serie iniciada");  // Sólo para debugging
14
15  // Inicialización de la comunicación Ethernet:
16    Ethernet.begin(mac, ip, gateway, subnet);   // start etehrnet interface
17    Serial.println("Interfaz Ethernet iniciada");
18
19    // Imprimir dirección IP Local (sólo para debugging):
20    Serial.print("Dirección IP Local: ");
21    for (byte thisByte = 0; thisByte < 4; thisByte++)
22    {
23      Serial.print(Ethernet.localIP()[thisByte], DEC);
24      Serial.print(".");
25    }
26    Serial.println();
27
28  // Inicializacion de buffers de memoria para datos modbus:
29    Mb.MbData[0] = 0;
30    Mb.MbData[1] = 0;
31    Mb.MbData[2] = 0;
32    Mb.MbData[3] = 0;
33    Mb.MbData[4] = 0;
34    Mb.MbData[5] = 0;
35  }
```

Figura 6. Programa de ejemplo (continuación), función de configuración e inicialización setup().

Seguidamente a la función *setup()*, se encuentra la función *loop()* que contiene la rutina principal del programa indicada en la Figura 7. Inicialmente en esta rutina, se llama a la función *MbsRun()* perteneciente a la librería MgsModbus-v0.1.1, que se encarga de recibir los datos para el RTU desencapsulando las tramas Modbus enviadas por la MTU. Luego se procede a encender o apagar los leds dependiendo de los valores cargados en la matriz *MbData[]*, estos valores son escritos por el usuario mediante la HMI desarrollada en el software Elipse SCADA, y podrán ser 0 (led apagado) o 1 (led encendido). La lectura de las entradas analógicas es almacenada directamente en los lugares del *MbData[]* asignados por el usuario (*MbData[4]* y *MbData[5]* en este caso), y cualquier ajuste necesario para su interpretación será realizado por el software SCADA, en donde se grafican estos valores en tiempo real.

```
1   // Programa principal:
2   //--------------------------------------------------------------------------
3   void loop()
4   {
5     Mb.MbsRun();  // ??
6
7   // Escritura del estado de los leds a partir de los registros Modbus:
8     digitalWrite(pinLed1, Mb.MbData[0]);
9     digitalWrite(pinLed2, Mb.MbData[1]);
10    digitalWrite(pinLed3, Mb.MbData[2]);
11    digitalWrite(pinLed4, Mb.MbData[3]);
12
13  // Lecturas de las entradas analógicas:
14    Mb.MbData[4] = analogRead(analogico4);
15    Mb.MbData[5] = analogRead(analogico5);
16  }
```

Figura 7. Programa de ejemplo (continuación), función principal loop().

Hay que mencionar que cada uno de los elementos del vector *MbData[]* se manipulan como si fueran variables comunes, algunas de ellas conteniendo los valores de las mediciones correspondientes a las entradas analógicas del microcontrolador y otras destinadas a almacenar los estados lógicos para

encender o apagar los leds asociados a sus salida digitales. Esto facilita el desarrollo del programa dedicado a la operación de la RTU, ya que el usuario únicamente debe preocuparse que los elementos asignados a cada dato sean congruentes en el programa realizado y en el software SCADA.

Conclusiones

Como se dijo en un principio, este proyecto se encuentra en proceso de desarrollo. Las actividades realizadas hasta el momento son la revisión bibliográfica de temas relacionados a los sistemas SCADA y protocolos de comunicación, también se efectuó el diseño de un módulo RTU de pruebas a los efectos de realizar ensayos (de software y hardware) a la par de los avances del diseño del circuito del RTU definitivo. Por otra parte, cabe destacar la facilidad de trabajar con el microcontrolador ATmega328P (el mismo que posee la plataforma digital Arduino UNO), ya que cuenta con una gran comunidad de desarrollo y una extensa bibliografía de consulta. Esto se ve reflejado en la reducción del tiempo de desarrollo a la hora de diseñar el software embebido en el microcontrolador, ya que la librería de comunicación utilizada, está completamente desarrollada y bien documentada. También se destaca el software Elipse SCADA, el cual se ha seleccionada para el desarrollo de la HMI. Este software es totalmente compatible con el protocolo de comunicación seleccionado para que la MTU pueda comunicarse con la RTU en el sistema propuesto. Los resultados obtenidos con los ensayos realizados sobre el módulo de pruebas son positivos, ya que se nota el abanico de posibilidades que un sistema como el planteado puede ofrecer, ya sea como maqueta de ensayos de comunicación y control en el laboratorio y/o como plataforma para demostraciones didácticas de temas afines. Por otra parte, en los siguientes meses se prevé la construcción del circuito del RTU definitivo, la selección del sensor a utilizar para la medición del nivel de líquido, la construcción del tanque y las estructuras de soporte para todo el sistema.

Referencias

ATMEL. (2016). Atmega 328/328P Datasheet.

Chavarría Mesa, L. (2007). *SCADA System's & Telemetry*.

My Arduino Projects. (s.f.). Recuperado el 04 de 06 de 2017, de http://myarduinoprojects.com/modbus.html

Pérez-López, E. (2015). Los sistemas SCADA en la automatización industrial. Recinto Grecia, Costa Rica.

WIZnet Co. (2009). W5100 Datasheet.