JIDeTEV- Año 2022 -ISSN 2591-4219

Sistema para supervisión remota de variables de proceso

Lucas S. Kirschner [a *], Cintia L. Mayer [a], Fernando Botterón [b], Guillermo A. Fernández [b]

[a] *Facultad de Ingeniería, Universidad Nacional de Misiones (UNaM), Oberá, Misiones, Argentina.*
[b] *IMAM-GIDE, FI-UNaM, Oberá, Misiones, Argentina.*
Juan Manuel de Rosas 325, Oberá, Misiones, Argentina
kirschnerlucas1@gmail.com, cintialisetmayer1998@gmail.com, botteron@fio.unam.edu.ar, guillermo.fernandez@fio.unam.edu.ar

Resumen

El presente trabajo tiene como objetivo el diseño y puesta en marcha de un sistema orientado al monitoreo y control de variables de proceso en forma remota, desarrollado en base a un sistema SCADA. Para ello, se propone establecer la comunicación entre dos estaciones de control distantes, utilizando un protocolo de comunicación industrial. El sistema de supervisión propuesto consiste en el desarrollo de un sistema embebido, que opera en base a un microcontrolador, y la programación de una aplicación SCADA. Los ensayos realizados con el sistema obtenido, presentan resultados satisfactorios en cuanto al cumplimiento de las especificaciones requeridas para este, permitiendo concluir que se ha cumplido con el objetivo propuesto. A esto puede agregarse que la realización del trabajo posibilitó la integración de varias temáticas desarrolladas en diferentes asignaturas de la carrera, como así también el aprendizaje acerca de aplicaciones destinadas a la implementación de sistemas SCADA y redes de comunicación de datos, temáticas abordadas en asignaturas de cursos superiores. Se destaca la enorme experiencia adquirida en este ámbito y su utilidad para ser aplicada en futuros proyectos de igual índole. Por otra parte, resulta importante mencionar que el sistema desarrollado pasará a formar parte de la instrumentación del Laboratorio de Electrónica de la Facultad de Ingeniería.

Palabras Clave *– SCADA, supervisión, control, comunicación, proceso, protocolo, UTR, UTM.*

1. Introducción

El presente trabajo surge como Actividad Integradora Final de la asignatura Técnicas Digitales 2 de la carrera de Ingeniería Electrónica y también como parte de las actividades de adscripción al proyecto de investigación “Bombeo de agua con energías renovables, almacenamiento de energía y conexión a la red para pequeñas huertas rurales comunitarias: Estudio, diseño y puesta en funcionamiento (16/I1083-PDTS)”. En el mismo se integran los saberes necesarios para el diseño e implementación de circuitos electrónicos que emplean dispositivos programables, como así también otros saberes adquiridos en la carrera.

El objetivo del trabajo es desarrollar un sistema para el monitoreo y control de variables de proceso en forma remota utilizando un sistema SCADA. Para esto se propone la realización de un sistema

*Autor en correspondencia.

embebido que, mediante un protocolo de comunicación de datos, interactúa con una aplicación SCADA residente en una computadora remota. Seguidamente se describe lo mencionado.

Un sistema SCADA (*Supervisory, Control and Data Acquisition*) comprende hardware de uso específico y un conjunto de aplicaciones de software, diseñados para la adquisición de datos, monitoreo y control de procesos de forma remota [1]. Permiten la comunicación con dispositivos remotos de campo, para monitorear y controlar el proceso en forma automática desde la interfaz de usuario visualizada en la pantalla de un ordenador (generalmente instalado en una oficina), la cual puede ser configurada y modificada con facilidad por el usuario. El sistema SCADA mencionado, provee a diversos usuarios el acceso a la información adquirida desde el proceso productivo. De esta manera, pueden existir usuarios con diferentes niveles de jerarquía para la toma de decisiones sobre los procesos que son controlados mediante los dispositivos de campo (tales como controladores autónomos, autómatas programables, sistemas de dosificación, etc.) [2]. La posibilidad de monitorear y controlar de manera remota los procesos productivos es la principal ventaja que ofrecen los sistemas SCADA frente a otros sistemas que requieren que el "panel de operación" se encuentre situado muy cerca de la máquina o dispositivo que se pretende controlar.

Este trabajo plantea el desarrollo de un sistema electrónico compuesto por una Unidad Terminal Remota (UTR) y una Unidad Terminal Maestra (UTM), vinculadas mediante un protocolo de comunicación de datos de uso industrial. La Fig. 1 presenta el diagrama de bloques correspondiente al sistema propuesto. La UTM mostrada en la figura está basada en una Notebook, donde se desarrolla la aplicación que posee la interfaz gráfica del sistema SCADA, a través de la cual los usuarios pueden monitorear y tener control sobre las variables medidas por los dispositivos de campo vinculados a la UTR. Esta última constituye un sistema embebido construido en base al microcontrolador ATmega328, presente en la placa de desarrollo Arduino NANO. El sistema embebido desarrollado posee acceso a tres entradas digitales, dos entradas analógicas, tres salidas digitales con opción PWM y dos salidas digitales tipo relés. Para el intercambio de datos entre las unidades UTM y UTR, se propone el uso del protocolo de comunicación industrial Modbus TCP/IP, implementado mediante el estándar Ethernet.

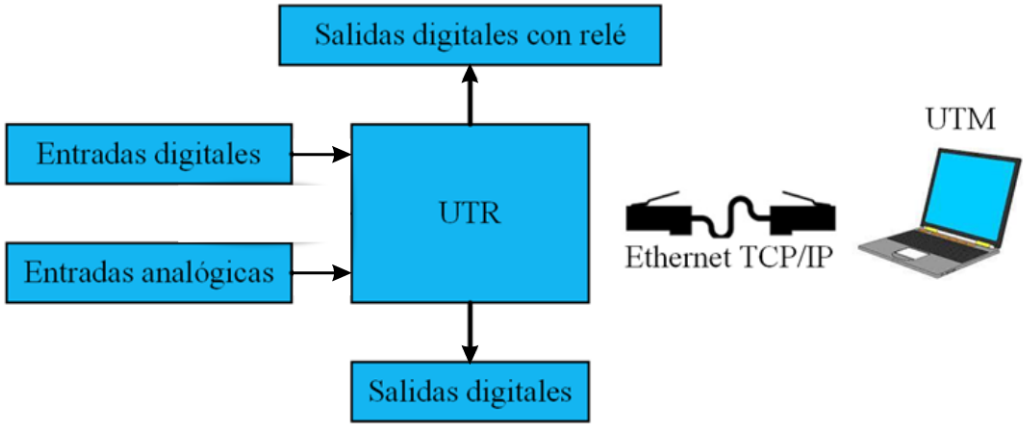

Fig. 1. Diagrama de bloques del sistema para supervisión remota de variables de proceso.

La estructura del presente artículo es dividida en cuatro secciones principales. En primer lugar, se presenta el sistema propuesto, indicando todas las partes que lo conforman y la justificación sobre la selección del mismo ante otras posibilidades existentes para la supervisión de variables de proceso. Seguidamente, el artículo presenta el diseño del sistema, expresando todos los cálculos y consideraciones tenidos en cuenta para la elección de los componentes electrónicos. A continuación, se evalúan los resultados obtenidos, verificando el cumplimiento de las especificaciones requeridas. Finalmente, están las conclusiones obtenidas tras el desarrollo del trabajo, haciendo mención de los principales problemas afrontados y las soluciones halladas.

2. Descripción del Sistema de Supervisión

El sistema electrónico de supervisión desarrollado consta de dos partes principales; una denominada Unidad Terminal Remota (UTR) y otra Unidad Terminal Maestra (UTM). Las unidades están vinculadas mediante un protocolo de comunicación industrial y conforman lo que se define como sistema SCADA.

2.1. Unidad Terminal Remota (UTR)

Esta unidad es un dispositivo basado en microprocesadores o microcontroladores, que permite obtener señales independientes de los procesos que se están controlando y enviar la información correspondiente a un sitio remoto donde la misma es procesada. Generalmente este sitio remoto es definido como Unidad Terminal Maestra [3].

La UTR contiene todos los recursos de software y hardware necesarios para establecer la comunicación entre las variables de campo de interés y la UTM, como así también, brinda la posibilidad de que el usuario, a través de la interfaz gráfica de la UTM, tenga total control sobre las variables de proceso que están siendo monitoreadas.

La UTR realiza una exploración periódica de las variables del proceso para mantener así los datos actualizados en tiempo real. Además, tiene la capacidad de ejecutar algoritmos de control programados por el usuario, sin la intervención de la UTM.

La implementación del sistema embebido que constituye la UTR es realizado con un Arduino NANO, cuyo esquema se indica en la Fig. 2. Este dispositivo es una placa de desarrollo de tamaño compacto, compatible con *protoboards* y basada en el microcontrolador ATmega328 de la firma ATMEL. La placa mencionada posee 14 pines de entrada/salida digital (de los cuales 6 pueden ser usados con PWM), 6 entradas analógicas, posee un cristal de 16 MHz, opera con una tensión de alimentación de 5 V y cuenta con conexión Mini-USB para depurar y grabar el programa en la memoria del microcontrolador [4].

La comunicación entre la UTM y la UTR se lleva a cabo mediante el protocolo de comunicación industrial Modbus TCP/IP [5]. Este es un protocolo de aplicación ubicado en la capa 7 del modelo OSI. Su topología es cliente-servidor, actuando como cliente la UTM y como servidor la UTR. Esta última, no posee una base de datos a la cual puede acceder el cliente (UTM), sino que en su operación realiza la lectura secuencial de las entradas correspondientes, para posteriormente transmitir los datos recopilado a la unidad maestra cuando esta lo solicite. Utiliza el estándar de redes Ethernet como medio físico para la transmisión de datos.

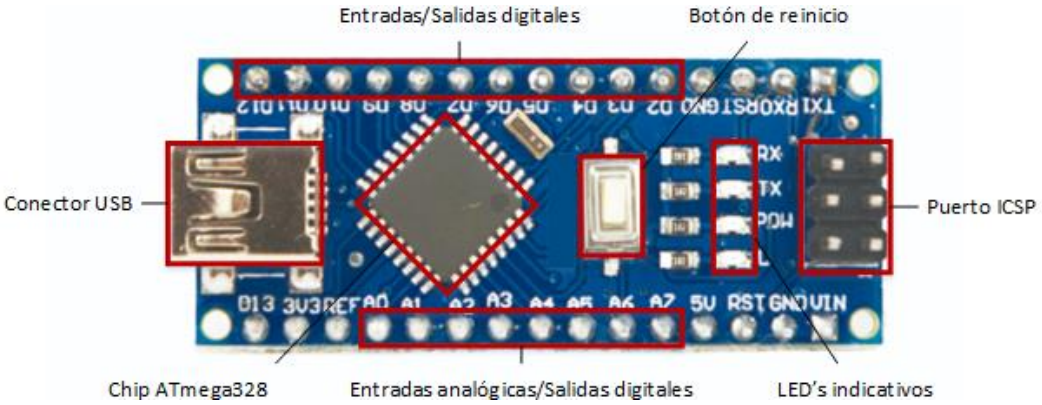

Fig. 2. Placa de desarrollo Arduino NANO [4].

Para la comunicación entre la UTR y la UTM, el microcontrolador la placa Arduino NANO se vale de un módulo Ethernet que provee a la UTR la capacidad de conectarse a la capa física. El módulo utilizado es el Arduino Ethernet Shield 1 [6] presentado en la Fig. 3. Este hace posible la conexión directa desde el Arduino NANO de la UTR a una red mediante cable, bajo el estándar Ethernet.

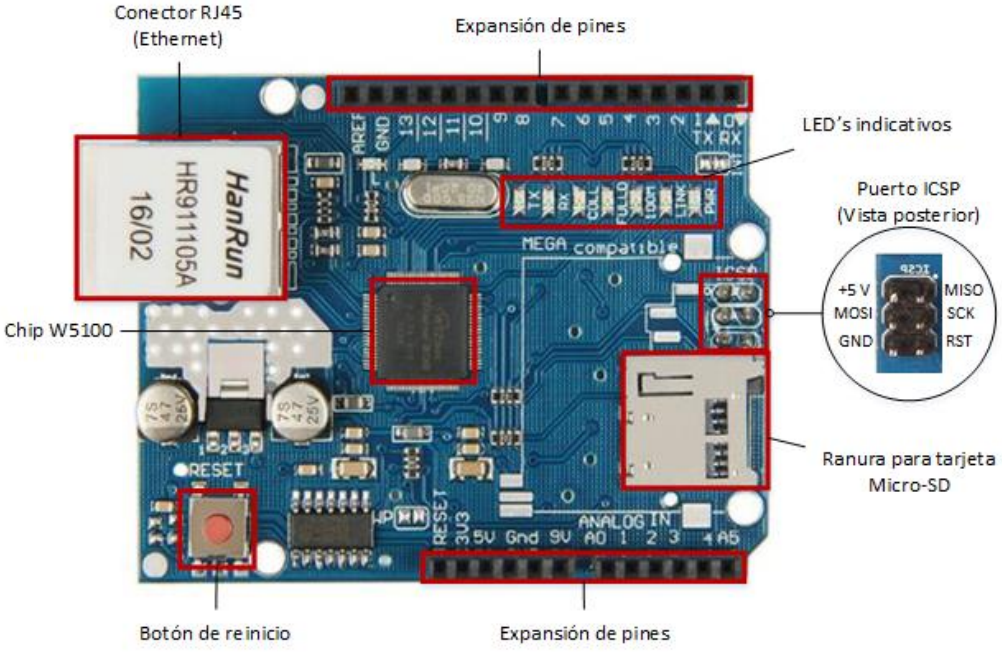

Fig. 3. Placa Arduino Ethernet Shield 1 [6].

El módulo Arduino Ethernet Shield 1 de la Fig. 3 incorpora una pila de TCP/IP por hardware y buffer interno de 16 KB para transmisión serie, lo que permite liberar de estas tareas al microcontrolador, siendo una de sus principales ventajas frente a otros controladores de Ethernet como el ENC28J60. Este módulo se basa en el chip Ethernet W5100 [7] fabricado por la empresa

Wiznet y está especialmente diseñado para aplicaciones embebidas, ya que cuenta con una interfaz SPI con la cual puede conectarse a un microcontrolador.

Si bien el módulo de la Fig. 3 está orientado a la conexión directa con los modelos Arduino UNO y MEGA, el uso del mismo con el Arduino NANO utilizado en la UTR, no representa mayores inconvenientes en la conexión, ya que la comunicación es efectuada mediante la interfaz SPI. Esta conexión es implementada usando las líneas: MISO (*Master In Slave Out*), para comunicar datos desde esclavos (Arduino Ethernet Shield 1) al maestro (Arduino NANO); MOSI (*Master Out Slave In*), para comunicar datos desde el maestro a los esclavos; SCK (*Clock*), para los pulsos de reloj de maestro a los esclavos; /SS (*Salve Select*), línea mediante la cual el maestro selecciona al esclavo con el cual quiere comunicarse. En este trabajo fueron utilizadas las líneas mencionadas, disponibles en el puerto ICSP mostrado en la Fig. 2.

La conexión de la UTR con la UTM, es efectuada mediante el conector RJ45 de la Fig. 3, implementando una red de área local bajo el estándar Ethernet. En este proyecto, el cable que une a ambas unidades es un cable Ethernet del tipo cruzado, como el utilizado para conectar dos dispositivos del mismo tipo (como ser la interconexión entre dos PCs o una PC con un switch, [8]).

2.2. Unidad Terminal Maestra (UTM)

Esta unidad es el elemento central del sistema de supervisión. Adquiere todos los datos procedentes de la UTR y los presenta en una interfaz gráfica al usuario del sistema SCADA. Así también, posibilita el envío de comandos a la estación remota (UTR) con el fin de controlar las variables de proceso correspondientes. Cabe destacar que la UTM es la encargada del almacenamiento histórico de datos, ya que la UTR continuamente toma los datos, pudiendo efectuar un registro mínimo temporal de los mismos [9]. Generalmente la UTM inicia la comunicación con la unidad terminal remota. Los mensajes de interrogación son transmitidos a cada UTR (en caso de varias) de forma secuencial, volviendo a iniciar la secuencia al finalizar un ciclo de interrogación.

La unidad terminal maestra puede implementarse con una computadora de escritorio o bien una portátil. En ambos casos debe contar con la aplicación SCADA, la cual posee la interfaz gráfica para que los usuarios interactúen con las variables de campo supervisadas, y también el protocolo de comunicación que le permite conectarse a la UTR.

En este trabajo, para el desarrollo de la aplicación SCADA se ha utilizado el software especializado Wonderware Indusoft Web Studio 8.0, el cual es una poderosa colección de herramientas de automatización que proporcionan todos los componentes básicos de la automatización para desarrollar interfaces hombre-máquina (HMI), aplicaciones SCADA y soluciones de instrumentación integradas. Es completamente configurable por el usuario y permite el monitoreo en tiempo real de variables a través de gráficos y otros objetos relacionados con las variables físicas que interesan medir. También permite realizar accionamientos, enviar y recibir información desde equipos remotos y automatizar diversas tareas de acuerdo a las necesidades del usuario. Seguidamente se explicará la aplicación SCADA desarrollada para la UTM del sistema de supervisión propuesto.

3. Diseño del Sistema de Supervisión

3.1. Diseño del Programa para la Unidad Terminal Remota

El diseño del programa correspondiente al sistema embebido que constituye la UTR, se llevó a cabo utilizando el Entorno de Desarrollo Integrado (IDE) de Arduino [10]. Este entorno brinda un conjunto de herramientas que facilitan la programación, tales como el uso de librerías, la detección de errores, selección de puertos y el modelo de la placa Arduino a programar. Las librerías proporcionan funcionalidades extra a la hora de trabajar con hardware externo a la placa Arduino NANO y facilita el manejo de la información. El IDE trae instaladas una serie de librerías genéricas que suelen ser las más utilizadas en el desarrollo de los programas, como así también, es posible el uso de librerías desarrolladas por terceros. Para este trabajo fue necesario el uso de las siguientes librerías: SPI.h, para la comunicación entre el módulo Ethernet y el microcontrolador mediante SPI; Ethernet.h, para el acceso a las funciones del chip W5100 que posee el módulo Ethernet utilizado; TimerOne.h, para configurar el Timer 1 del microcontrolador ATmega328; MgsModbus.h: Incluye varias funciones del protocolo Modbus, que facilitan la comunicación de la información entre las estaciones remota y maestra.

La UTR desarrollada tiene una serie de entradas y salidas a través de las cuales puede interactuar con el proceso que se desea supervisar. Estas entradas y salidas interactúan con las variables de campo a supervisar con el sistema desarrollado. Las entradas y salidas del Arduino NANO utilizadas para esto son: A3 y A4 configuradas como entradas analógicas (de 0 a 5V); A2, D7 y D8 configuradas como entradas digitales (0 = 0 V y 1 = 5 V); D2 y D3 configuradas como salidas digitales pero asociadas a relés y D5, D6 y D9 configuradas como salidas digitales con opción de PWM.

Como lo muestra la Fig. 4, en el programa desarrollado para la UTR, luego de la inclusión de librerías y la definición e inicialización de variables, se procede a efectuar el ajuste de los parámetros de red necesarios para la comunicación Ethernet. Inicialmente se especifica una dirección MAC (*Media Access Control*), que se trata de un código de 6 bytes propio de todo dispositivo de red. Usualmente los dispositivos traen una dirección MAC fija, pero si este no es el caso, es posible especificar una mediante software. Luego, en base a los datos de la red Ethernet local es definida la dirección IP con un código de 4 bytes se identifica a un dispositivo dentro de una red (en este caso será la dirección IP de la UTR en la red de área local). Cabe mencionar que, si hubiera varias UTRs, cada una debería contar código IP distinto a modo que no existan dos o más iguales en la misma red. Seguidamente a la dirección IP es establecida la puerta de enlace, que no es otra cosa que la dirección IP del router (aunque esta puede tener cualquier dirección lo usual es que su dirección IP sea la primera de la red). Como parte final de la configuración de red, en el programa de la UTR se define también la máscara de subred, que al igual que la dirección IP, está compuesta por 4 bytes, los cuales permiten conocer cómo se deben asignar las direcciones IP en la red.

Como lo muestra la Fig. 4, luego de configurar los parámetros de red, se crea un objeto de la clase MgsModbus.h, necesario para tener control sobre un espacio de datos que ofrece esta clase, definido por el vector *MbData[]*, en el cual es posible leer y escribir los datos transmitidos entre las unidades remota y maestra, pudiendo ambas trabajar sobre los mismos registros.

```
//---------------- Inclusión de Librerías ---------------------//
#include <SPI.h>
#include <Ethernet.h>
#include "MgsModbus.h"
#include <TimerOne.h>

//------------- Entradas/salidas del ATmega 328 ---------------//
int   out_d1 = 2,   out_d2 = 3;
int   out_pwm_1 = 6, out_pwm_2 = 5, out_pwm_3 = 9;
int   in_d1  = A2, in_d2  = 7, in_d3 = 8;
int   in_a1  = A3, in_a2 = A4;
int   led_prueba = A5;
int   cont = 0;

//--- Ajustes de Modbus y de Ethernet según MAC y RED local ---//
byte mac[] = {0x90, 0xA2, 0xDA, 0x0D, 0xA0, 0x88 };
IPAddress ip(169,254,250,190);
IPAddress gateway(169,254,250,1);
IPAddress subnet(255,255,0,0);
MgsModbus Mb;
```

Fig. 4. Inclusión de librerías, definición de variables y configuración de red.

En la primera parte de la función *setup()* plasmada en la Fig. 5, se efectúa la configuración de los pines según corresponda para las entradas y salidas de la UTR que estará ligadas a las variables de proceso a supervisar. Las salidas digitales son inicializadas en estado bajo para prevenir los riesgos que un estado en alto podría ocasionar. Seguidamente, es inicializado el Timer 1 con una temporización de 0,1 s (100000 µs), habilitándose la interrupción por desbordamiento del contador de este Timer. A continuación, se establece el *baud rate* de la comunicación serie, quedando de esta manera inicializada la comunicación SPI que es establecida con el módulo Ethernet.

Para iniciar la comunicación Ethernet es utilizada la función *begin*. La misma inicializa la librería ethernet y configura los parámetros de red necesarios para el módulo Ethernet. Los parámetros enviados a la función son las direcciones MAC e IP del dispositivo, la puerta de enlace y la máscara de subred.

Luego de las configuraciones anteriores, en la función *setup()* de la Fig. 5, es inicializado el espacio de memoria para los datos Modbus. Los espacios *MbData[0]* y *MbData[1]* son utilizados para almacenar el estado (cero o uno) de las salidas digitales asociadas a los relés. Los espacios *MbData[2]*, *MbData[3]* y *MbData[4]* son asignados para las tres salidas digitales con opción PWM. Los espacios *MbData[5]*, *MbData[6]* y *MbData[7]* están relacionados a las entradas digitales definidas en la configuración de entradas y salida. Los espacios *MbData[8]* y *MbData[9]* se emplean para guardar el valor digital correspondiente a las entradas analógicas.

```
void setup()
{
//-----------------Config de E/S digitales y analógicas ----------------------------//
  pinMode(out_d1,OUTPUT);     pinMode(out_d2,OUTPUT);
  pinMode(out_pwm_1,OUTPUT);  pinMode(out_pwm_2,OUTPUT);  pinMode(out_pwm_3,OUTPUT);
  pinMode(in_d1,INPUT);       pinMode(in_d2,INPUT);       pinMode(in_d3,INPUT);
  pinMode(in_a1,INPUT);       pinMode(in_a2,INPUT);
  pinMode(led_prueba,OUTPUT);
//--------------------- Inicializa salidas digitales -------------------------------//
  digitalWrite(out_d1,LOW); digitalWrite(out_d2,LOW);
  digitalWrite(out_pwm_1,LOW); digitalWrite(out_pwm_2,LOW);digitalWrite(out_pwm_3,LOW);
//----------------------- Configuración del Timer 1 ----------------------------------//
  Timer1.initialize(100000);
  Timer1.attachInterrupt(ISR_Blink);
//------------------- Configuración del módulo SPI ---------------------------------//
  Serial.begin(9600);
  Serial.println("Interfaz serie iniciada");
//-------------- Inicialización de la comunicación Ethernet ------------------------//
  Ethernet.begin(mac, ip, gateway, subnet);
  Serial.println("Interfaz Ethernet iniciada");
//---------- Inicializacion de buffers de memoria para datos modbus -----------------//
  Mb.MbData[0] = 0; Mb.MbData[1] = 0;
  Mb.MbData[2] = 0; Mb.MbData[3] = 0; Mb.MbData[4] = 0;
  Mb.MbData[5] = 0; Mb.MbData[6] = 0; Mb.MbData[7] = 0;
  Mb.MbData[8] = 0; Mb.MbData[9] = 0;
}
```

Fig. 5. Función *setup()*.

```
void loop()
{
//------------------- Habilitación de la comunicación -----------------------------//
  Mb.MbsRun();
//----------------- Control de salidas digitales con relés -----------------------//
  digitalWrite(out_d1,Mb.MbData[0]);
  digitalWrite(out_d2,Mb.MbData[1]);
//------------------ Control de salidas digitales con opción PWM -------------------//
  digitalWrite(out_pwm_1,Mb.MbData[2]);
  digitalWrite(out_pwm_2,Mb.MbData[3]);
  digitalWrite(out_pwm_3,Mb.MbData[4]);
//--------------------- Control de entradas digitales ----------------------------//
  Mb.MbData[5] = digitalRead(in_d1);
  Mb.MbData[6] = digitalRead(in_d2);
  Mb.MbData[7] = digitalRead(in_d3);
//-------------------- Control de entradas analógicas ----------------------------//
  Mb.MbData[8] = analogRead(in_a1);
  Mb.MbData[9] = analogRead(in_a2);
}
```

Fig. 6. Función *loop()*.

En la función *loop()* que presenta la Fig. 6, en primer lugar está la habilitación de la comunicación Modbus. La función *MbsRun()* es la encargada de recibir los datos que llegan a la UTR y desencapsular las tramas Modbus enviadas por la UTM. Seguidamente a la función *MbsRun()*, es realizada la lectura de las entradas digitales y analógicas asignado sus valores a los registros

correspondientes del vector *MbData[]*. Así también, se establece el estado de las salidas digitales en base a los valores establecidos a través del protocolo Modbus por la UTM.

```
50 //----------------- RSI del desbordamiento del Timer 1 ----------------------------//
51 void ISR_Blink()
52 {
53   if(cont < 9)
54   {
55     digitalWrite(led_prueba,LOW);
56     cont = cont + 1;
57   }
58   else
59   {
60     digitalWrite(led_prueba,HIGH);
61     cont = 0;
62   }
63 }
```

Fig. 7. Rutina de Servicio a la Interrupción por desbordamiento del Timer 1.

El temporizado efectuado por el Timer 1, usado en el programa desarrollado para la UTR, es utilizado para originar el parpadeo de un LED indicativo cada 1 s. El mismo actúa como testigo, para indicar que el programa está ejecutándose en el microcontrolador ATmega328 de la placa Arduino NANO. La Fig. 7 expone sección del código correspondiente a la Rutina de Servicio a la Interrupción (RSI) por desbordamiento del contador del Timer 1, encargada de producir el parpadeo del LED testigo.

3.2. Diseño del Hardware de la Unidad Terminal Remota

Esta subsección presenta los cálculos y consideraciones correspondientes a la selección de los distintos componentes que conforman el hardware de la UTR.

La placa Arduino NANO cuenta con una entrada para alimentación externa VIN, a través de la cual el microcontrolador ATmega328 puede energizarse mediante un regulador de tensión interno. Para la entrada VIN es recomendable una tensión entre 7 y 12 V, siendo el límite absoluto de 6 a 20 V. Considerando esto, la UTR desarrollada utiliza una fuente externa de 12 V, la cual además es la encargada de alimentar a los relés, tal como se detalla más adelante.

En el circuito de la UTR, a la entrada de alimentación VIN se agrega un capacitor electrolítico C_1 = 100 µF encargado de estabilizar la tensión. En paralelo al anterior el circuito también posee un capacitor cerámico C_2 = 100 nF para eliminar los ruidos en las altas frecuencias.

Las tres entradas digitales de la UTR utilizan jumpers con resistencias *pull-up*, a modo de simular el estado lógico 0 y 1 en las variables que serán supervisadas a través de dichas entradas. Las resistencias dimensionadas a través de las ecuaciones (1) y (2), donde $V_{OH\min}$ es la mínima tensión de salida proporcionada por el Arduino NANO, $V_{IH\min}$ es la mínima tensión considerada como un estado alto en las entradas y $I_{IH\max}$ es la máxima corriente de entrada en los pines digitales actuando como salida.

$$R_{PU} \leq \frac{V_{OH\min} - V_{IH\min}}{I_{IH\max}} \quad (1)$$

$$P_{RPU} \geq 1{,}5 \cdot \frac{(V_{OH\max})^2}{R_{PU}} \Rightarrow P_{RPU} \geq \frac{1}{4} W \quad (2)$$

Generalmente para las resistencias *pull-up* es una práctica adoptar resistencias de 10 kΩ de 1/4 W. El fundamento teórico para esto establece que el valor debe ser suficientemente alto como para no consumir una corriente excesiva, pero no tanto como para influir en el estado lógico "1" al que se desea fijar la entrada digital. La potencia necesaria para la misma es calculada considerando que el jumper de la entrada correspondiente está conectando la entrada digital al potencial de 0 V (masa), ya que esta condición corresponde al caso de mayor disipación de energía en las *pull-up*.

Las tres salidas digitales con opción PWM posee en paralelo LEDs de alta luminosidad de 3 mm (color azul) que indican el estado activo de las mismas. Estos presentan una caída de tensión de 3,7 V. En base a experiencias prácticas realizadas se decidió utilizar resistencias limitadoras de 330 Ω y 1/4 W, ya que con las mismas se consiguió una adecuada visualización del estado encendido.

Para el LED usado como indicador de programa ejecución en el microcontrolador de la UTR, se selecciona un LED amarillo de 3 mm de alta luminosidad, el cual presenta una caída directa de 2 V. Considerando esto, en serie con el mismo se colocó una resistencia limitadora de 330 Ω y 1/4 W.

Para las salidas digitales asociadas con los relés en la UTR, es seleccionado el relé SRD-S-112D [11] de la firma Sanyou. Estos dispositivos presentan una tensión y corriente de bobina de 12 V y 30 mA, respectivamente. En el circuito desarrollado para la UTR, cada relé posee en paralelo a su bobina un LED´s de color rojo de alta luminosidad para indicar su estado. Estos LED´s presentan una caída en directa de 2 V. En serie a los mismos se agregan resistencias limitadoras de corriente de 1 kΩ y 1/4 W. Resultando una corriente directa de 10 mA a través de los LED´s.

Sumando la corriente de LED y bobina de relé, el valor resultante es excesivo para que pueda ser manejado por una salida del microcontrolador de la UTR. Por tal motivo, esta carga es controlada mediante llaves electrónicas implementadas con transistores BJT del tipo NPN del modelo BC337 [12]. Los mismos presentan encapsulado TO-92, corriente de colector máxima I_C = 800 mA, caída de tensión colector-emisor de saturación V_{CEsat} = 0,7 V, caída de tensión base-emisor V_{BEon} = 1,2 V y ganancia de corriente mínima β = 100. Con estas especificaciones, y la ecuación (3) se calcula el valor de las resistencias de base para dichos transistores.

$$R_B = \frac{V_{OH\min} - V_{BEon}}{5 \cdot I_C / \beta} \quad (3)$$

El factor de seguridad de 5 veces la corriente de base presente en la ecuación (3) es considerado asegurar que el transistor opere en saturación. A partir de esto se seleccionan resistencias comerciales de 1,5 kΩ y 1/4 W, con lo cual la corriente de base resultante es de 2 mA, muy inferior al límite

máximo de corriente de 20 mA (máx. 40 mA) que los pines digitales del microcontrolador pueden proporcionar.

En cuanto a la máxima potencia con la que pueden operar estos transistores, la misma está da por lo que resulta de la ecuación (4).

$$P_{D\max} = \frac{T_{J\max} - T_{amb}}{\theta_{JA}} \tag{4}$$

Siendo la temperatura máxima de juntura $T_{J\max}$ = 150 °C, la resistencia térmica de juntura-carcasa θ_{JA} = 200 °C/W, para una temperatura ambiente T_{amb} = 50 °C, la potencia de operación máxima no debe superar $P_{D\max}$ = 500 mW. Para verificar esto, la potencia de operación de cada transistor que maneja los relés de la UTR se obtiene mediante la ecuación (5), cuyo resultado verifica que $P_D << P_{D\max}$.

$$P_D = I_C \cdot V_{CEsat} + I_B \cdot V_{BEon} = 30,4\ \text{mW} \tag{5}$$

Para finalizar con el diseño del hardware de la UTR, es constatado que no se excede el límite de corriente que el regulador interno del Arduino NANO es capaz de proporcionar. Este regulador es el AMS1117-5.0, con tensión de salida regulada de 5 V [13]. Considerando que no cuenta con un disipador térmico y que la temperatura ambiente es T_{amb} = 50 °C, la máxima potencia con la que puede operar dicho regulador es de aproximadamente 700 mW. Si a esto se suma que la tensión de entrada es de 12 V, se obtiene como resultado que la corriente que es posible extraer del regulador sin superar la máxima disipación de potencia es de 100 mA. Haciendo la suma de todas las corrientes en las salidas digitales, se obtiene aproximada 45 mA. Agregando a esto el consumo de 20 mA que posee el propio ATmega328 y los 20 mA consumidos por el chip W5100 del Arduino Ethernet Shield 1, quedan disponibles 15 mA para las entradas analógicas que se conecten y las salidas digitales con opción PWM. Por lo tanto, a la hora utilizarlos debe tenerse en cuenta que el consumo en conjunto no exceda este límite y colocar alguna etapa de amplificación de corriente en caso de ser necesario.

3.3. Diseño de la interfaz gráfica de la Unidad Terminal Maestra

El diseño de la interfaz gráfica para la UTM fue desarrollado mediante el software ya mencionado en la sección 2, el cual hace posible el proceso de automatización y que proporciona todos los componentes básicos para la creación del sistema SCADA.

En la aplicación programa para la UTM se crean variables vinculadas al vector de datos *MbData[]* utilizado en la función *setup*() de la Fig. 5. El vínculo se establece a través del protocolo Modbus, mediante la red Ethernet de la PC que actúa como unidad maestra.

La interfaz gráfica desarrollada presenta al usuario el estado de las entradas digitales. Así también, es posible visualizar los valores de tensión leídos a través de las entradas analógicas. La interfaz indica en todo momento el valor instantáneo de la medición, como así también una gráfica con la variación de la señal durante el último minuto transcurrido. La tensión en la entrada analógica puede tratarse de una señal obtenida de un potenciómetro o preset, de un sensor de temperatura o cualquier otro dispositivo sensor que proporcione una señal de tensión continua en el rango de 0 a 5 V.

Cabe destacar que la UTM recibe el valor directo obtenido del conversor analógico-digital que posee el microcontrolador de la UTR y lleva a cabo la transformación al valor de tensión correspondiente. Esto se hace así para reducir los errores cometidos por truncamiento y redondeo.

Así también, el usuario tiene total control sobre las salidas de la UTR. Estas son las dos salidas digitales asociadas a los relés y las tres salidas digitales con opción de señal PWM.

Para establecer la comunicación con la unidad remota, solo es necesario tener conectado el cable Ethernet entre el puerto 502 de la PC y el puerto RJ45 del Módulo Ethernet. En la aplicación SCADA se configuró la dirección IP especificada previamente en la UTR, estando ya definidas la máscara de subred y la puerta de enlace.

4. Resultados

La Tabla 1 presenta una lista de los materiales y componentes utilizados para la construcción del circuito correspondiente a la unidad remota y la comunicación de esta con la unidad maestra.

Tabla 1. Materiales y componentes utilizados en la UTR.

Material/Componentes	Cantidad	Descripción
Arduino NANO	1	Placa de desarrollo
Arduino Ethernet Shield 1	1	Módulo Ethernet
Cable UTP	1	Cruzado
SRD-S-112D	2	Relé de 12 V
BC337	2	Transistor BJT
Jumper	3	Comunes
Capacitor de 100 μF − 16 V	1	Electrolítico
Capacitor de 0,1 μF − 50 V	1	Cerámico multicapa
Resistencia de 1 kΩ − 1/4 W	2	Film de carbón
Resistencia de 1,5 kΩ − 1/4 W	2	Film de carbón
Resistencia de 10 kΩ − 1/4 W	3	Cermet
Resistencia de 330 Ω − 1/4 W	4	Cermet
LED de 3 mm	1	Amarillo
LED de 3 mm	2	Rojo
LED de 3 mm	3	Azul
Borneras dobles	4	Para conexión externa
Borneras triples	4	Para conexión externa
Placa virgen para PCB	1	Simple faz de 10 cm x10 cm.

La Fig. 8 muestra el esquema de conexión entre la placa Arduino NANO y el módulo Ethernet, como así también con los distintos componentes que conforman el sistema embebido desarrollado como UTR para el sistema de supervisión propuesto. La selección de los pines de entradas/salidas usadas se basó en gran medida en la facilidad existente a la hora de efectuar la conexión de los mismos en el circuito impreso.

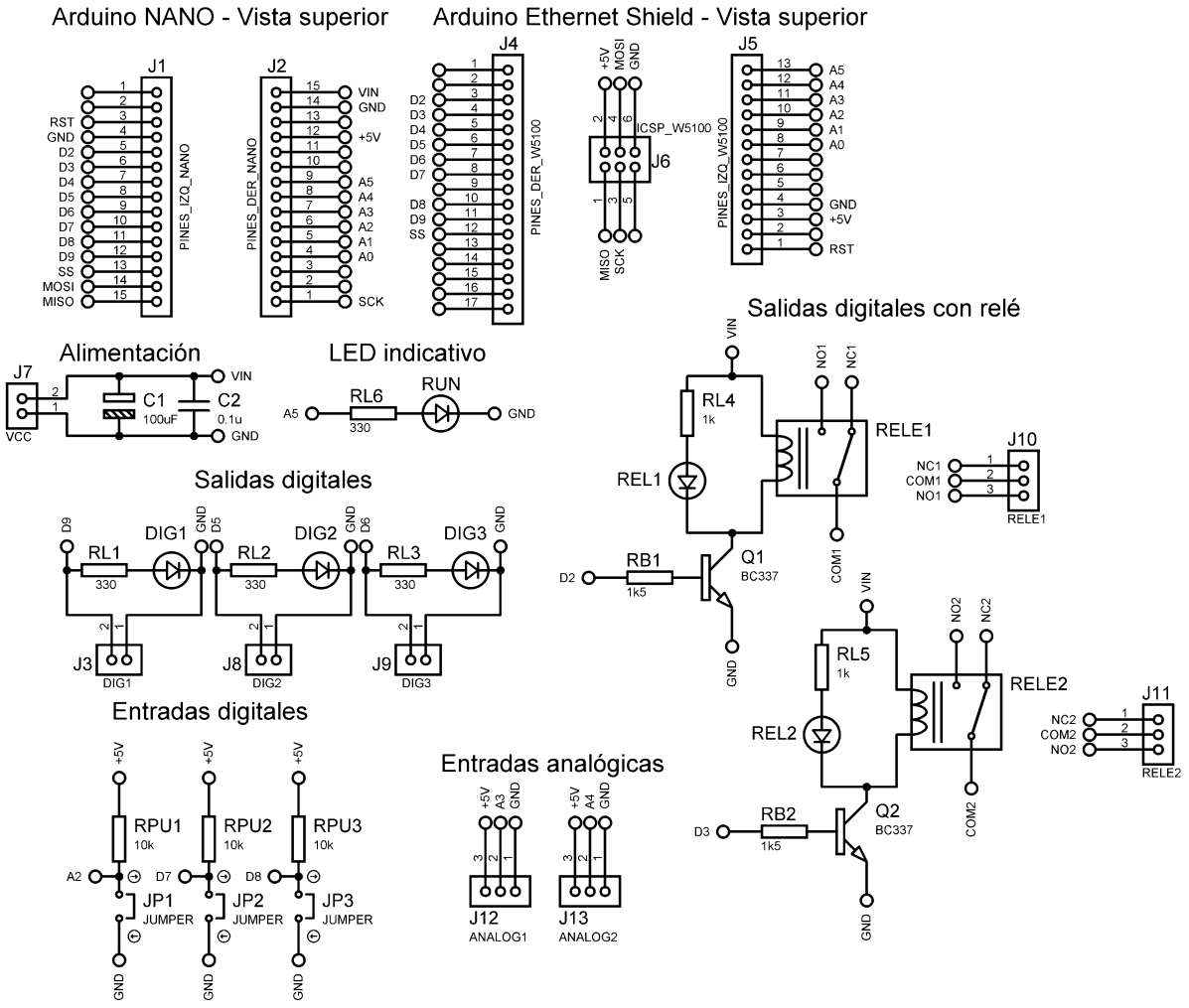

Fig. 8. Circuito de la Unidad Terminal Remota (UTR).

La placa construida para la UTR es presentada en la Fig. 9. En la misma están indicadas las borneras de entradas y salidas, las cuales deben conectarse a las variables a supervisar. También se aprecian, montadas a través de tiras de pines, las placas del Arduino NANO y el Arduino Ethernet Shield 1. Debajo de este último están ubicados los jumpers vinculados a las entradas digitales. En el extremo inferior de la placa se disponen las borneras relacionadas a las salidas digitales, junto a los respectivos LED's indicativos los cuales permiten simular el accionamiento de entradas digitales al sistema de supervisión. A la izquierda de los jumpers, se aprecian los dos relés con sus respectivos LED´s. Los relés de la UTR tienen disponibles los contactos NC, NA y C en las dos borneras triples que están en la parte inferior derecha de la placa. A estas borneras le sigue una bornera doble mediante la cual puede efectuarse la alimentación de todo el circuito, con una tensión de 7 a 12 V. Las dos borneras triples restantes de la parte superior de la placa, permiten conectar señales analógicas a la UTR, las cuales puede presentar variaciones entre 0 y 5 V.

La Fig. 10 muestra la interfaz gráfica de la aplicación SCADA desarrollada para la UTM y a través de cual el usuario puede interactuar con el sistema de supervisión desarrollado. El estado de las

entradas digitales de la UTR es visualizado mediante las llaves de la parte superior, definidas como Entrada digital 1, 2 y 3. A la derecha de estas llaves, están los interruptores que posibilitan el control de las salidas digitales con opción PWM, disponibles en la UTR.

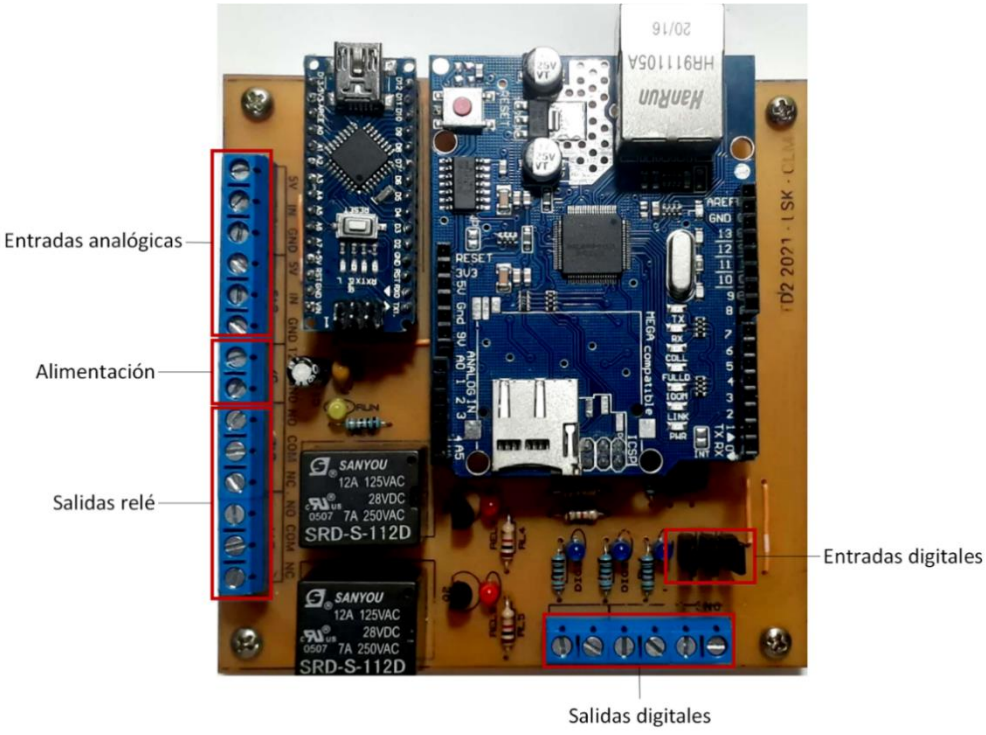

Fig. 9. Vista frontal de la placa.

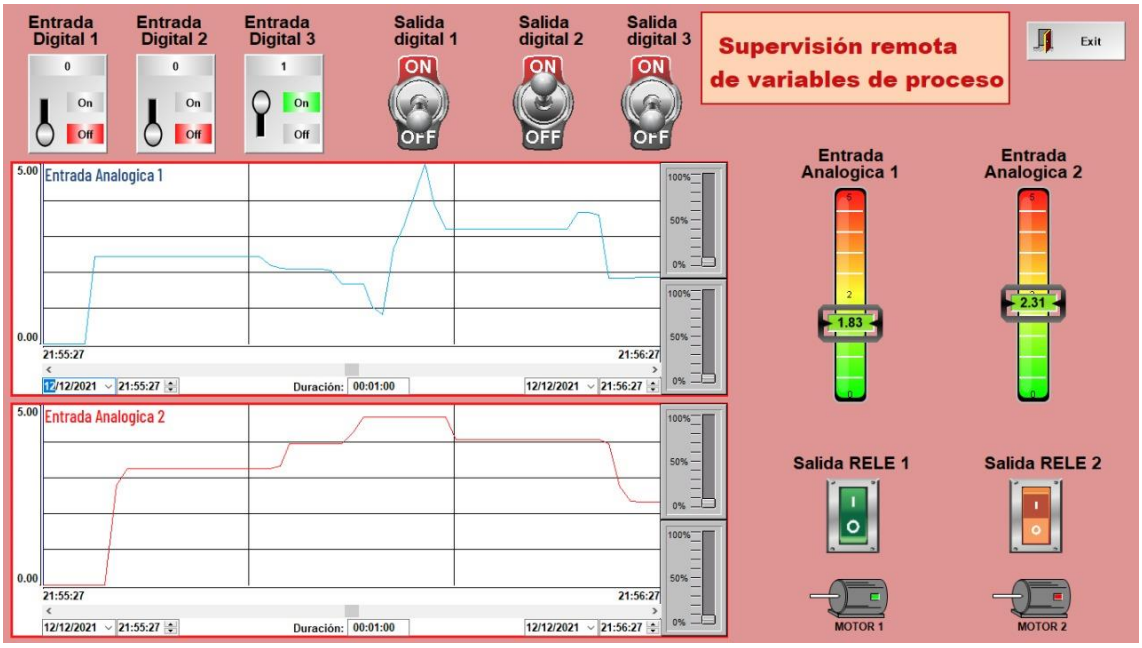

Fig. 10. Interfaz gráfica de la aplicación SCADA correspondiente a la UTM.

En la interfaz de la Fig. 10, el valor instantáneo correspondiente a cada una de las señales aplicadas a las entradas analógicas de la UTR, puede observarse con los dos indicadores tipo barras de la parte derecha. Adicionalmente a esto, los gráficos de la interfaz presentan la variación temporal de las señales analógicas, en el transcurso del último minuto. Por último, con los interruptores de la parte inferior derecha de la interfaz indicada en la Fig. 10, es posible comandar el estado de los relés de la UTR, los cuales pueden ser usados para controlar el estado de motores, por ejemplo.

Para verificar el correcto funcionamiento del sistema se hicieron reiteradas pruebas de encendido y apagado de las salidas digitales y las salidas tipo relé mediante la interfaz gráfica de la aplicación SCADA (Fig. 10), visualizando el estado correspondiente de los LEDs en los indicadores de dicha interfaz y en la placa, los cuales concordaron en todos los casos. Así también, variando la posición de la perilla de dos potenciómetros conectados en las entradas analógicas, pudo apreciarse como se modificaban los datos registrados en la interfaz gráfica, cuya variación temporal relevada es presentada en las curvas de tensión (V) – tiempo (s) mostradas en la Fig. 10. Por último, también fue constatado el monitoreo de las entradas digitales, conectando y desconectando los Jumpers situados en la placa de la UTR, con esto pudo visualizarse la correcta indicación en la interfaz de la UTM, en todos los casos. A través de las pruebas mencionadas, queda de manifiesto el correcto funcionamiento del sistema de supervisión desarrollado y el cumplimiento de los requerimientos especificados para el mismo en la actividad integradora final.

5. Conclusiones

Con este trabajo se han cumplido las especificaciones establecidas para la Actividad Integradora Final de la asignatura Técnicas Digitales 2, correspondiente al 4to. año de la carrera Ingeniería Electrónica. Fue posible resolver la problemática planteada mediante el diseño de un sistema capaz de comunicar la unidad remota (UTR) con la unidad maestra (UTM), para que de esta manera el usuario del sistema de supervisión desarrollado sea capaz de monitorear y controlar las variables de proceso de manera remota.

El desarrollo del trabajo, permitió aplicar los saberes adquiridos durante el cursado de asignatura mencionada, coma sí también implicó el uso de los conocimientos adquiridos en diversas asignaturas de la carrera de Ingeniería Electrónica. En el diseño del Software de la UTR fueron muy útiles los conceptos de programación desarrollados en las asignaturas Informática y Computación. Para el cálculo de la etapa de diseño del hardware de la UTR se aplicó los saberes adquiridos en las asignaturas Dispositivos Electrónicos y Electrónica Analógica, conceptos que además se reafirmaron en la asignatura Técnicas Digitales 1. Por otro lado, fue necesario investigar acerca de protocolos de comunicación industrial, redes de comunicación de datos y sistemas SCADA, temas que se desarrollan en asignaturas del 5to año de la carrera. Los principales saberes adquiridos en este sentido, corresponden al manejo de aplicaciones utilizadas para el diseño de sistemas SCADA y en el establecimiento de la comunicación entre dispositivos que están distantes uno de otro. Esto último incluye la selección del adecuado protocolo de comunicación, de acuerdo a las características del proceso que se esté llevando a cabo. Con esto se pone en evidencia el gran número de conceptos que intervinieron en el desarrollo de la actividad, exigiendo de esta manera reafirmar conocimientos ya adquiridos, como así también, investigar en áreas que aún resultaban por conocer.

En cuanto a las actividades relacionadas con el proyecto de investigación, puede decirse que los resultados obtenidos con este trabajo constituyen un avance importante para el plan de trabajo correspondiente a la adscripción. El hecho de poder implementar la comunicación entre la UTR y la UTM y desarrollar la aplicación SCADA, facilitarán el desarrollo del sistema de supervisión para la microrred empleada como fuente de energía en el sistema de bombeo de agua de las huertas rurales.

Por otra parte, también puede destacarse que los saberes adquiridos con este trabajo, resultan de suma importancia para la extrapolación de los mismos a otros proyectos de investigación en el que actualmente están trabajando otros miembros del grupo de investigación.

También puede mencionarse que el sistema desarrollado queda a disposición del Laboratorio de Electrónica de la Facultad de Ingeniería, para efectuar eventuales ensayos relativo a sistemas de esta índole.

6. Referencias

[1] Aquilino Rodríguez Penin, "Sistemas SCADA," *in Sistemas SCADA*, 3th ed. Alfaomega Grupo Editor, S.A. de C.V., México.

[2] Esteban Pérez, "Descripción general de un SCADA," en *Los sistemas SCADA in la automatización industrial*, 3th ed., vol. 28, J.

[3] *Unidad Terminal Remota*. [Online]. Available: https://es.wikipedia.org/wiki/Unidad_Terminal_Remota

[4] *Arduino NANO*. [Online]. Available: https://store.arduino.cc/products/arduino-nano

[5] *Protocolo Modbus*. [Online]. Available: https://modbus.org/docs/Modbus_Application_Protocol_V1_1b3.pdf

[6] *Arduino Ethernet Shield 1*. [Online]. Available: https://programarfacil.com/blog/arduino-blog/ethernet-shield-arduino/

[7] *W5100 Datasheet*. [Online]. Available: https://www.sparkfun.com/datasheets/DevTools/Arduino/W5100_Datasheet_v1_1_6.pdf

[8] *Cable cruzado y directo*. [Online]. Available: https://community.fs.com/es/blog/patch-cable-vs-crossover-cable-what-is-the-difference.html

[9] *Unidad Terminal Maestra*. [Online]. Available: https://acortar.link/D21QAo

[10] *Arduino IDE*. [Online]. Available: https://www.arduino.cc/en/software

[11] *SRD-S-112D Datasheet*. [Online]. Available: http://www.datasheet.es/PDF/897968/SRD-S-112D-pdf.html

[12] *BC337 Datasheet*. [Online]. Available: https://pdf1.alldatasheet.es/datasheet-pdf/view/156196/ONSEMI/BC337.html

[13] *AMS1117 Datasheet*. [Online]. Available: https://pdf1.alldatasheet.es/datasheet-pdf/view/520186/FCI/LM1117-5.0-220.html