Año: 2024. Volumen: 1
Numero:10. ISSN: 2448-623X

Desarrollo de una Estación Remota de Monitoreo Basada en Modbus RTU RS485 para la Estimación en Tiempo Real de la Potencia Generada por un Módulo Fotovoltaico

Vidaña-Aldaba, K.G.[1]; Lara-Cardoso, J.[2]✉; Hernández-Flores, C.[1]; Arjona-López, M. A.[1]

Datos de Adscripción:

[1] Tecnológico Nacional de México / Instituto Tecnológico de La Laguna, División de Estudios de Posgrado e Investigación, Blvd.. Revolución y Av. Instituto Tecnológico de La Laguna, C.P. 27000, Torreón, Coahuila, México.

[2], Tecnológico Nacional de México/ Instituto Tecnológico Superior de Lerdo, Departamento de Posgrado, Av. Tecnológico No. 1555 Sur, Periférico Gómez - Lerdo Km. 14.5, C.P. 35150, Lerdo, Durango, México.

✉ jorge_lara_c@yahoo.com

Resumen **- En este trabajo se presenta el desarrollo y pruebas de una estación remota de monitoreo basada en el protocolo de comunicación MODBUS RTU RS485 para estimar en tiempo real la potencia teórica generada por un módulo fotovoltaico en función de su temperatura de operación y de la irradiancia solar. La estimación de esta potencia es importante en sistemas fotovoltaicos pues permite evaluar el desempeño de los algoritmos MPPT de extracción de potencia máxima requeridos para operar los módulos fotovoltaicos en su valor óptimo de voltaje y corriente. Para lograr esto, se ha configurado un piranómetro Infwin PYR20 y un sensor de temperatura Taidacent SHT30 como esclavos mientras que una tarjeta Arduino UNO es el maestro, la cual recibe los datos de las mediciones de estos sensores desde una distancia de aproximadamente 30 m mediante el estándar industrial de transmisión de datos serie RS485. A su vez, esta información es enviada a través de una interfaz RS232 a un DSP F28335 de Texas Instruments programado en Matlab-Simulink, donde finalmente se calcula de manera continua y en tiempo real la potencia estimada para el módulo fotovoltaico Blue Carbon BCT20M-12 de 20W. Los resultados obtenidos de las pruebas experimentales realizadas bajo diferentes condiciones ambientales y desplegados en tiempo real en el IDE de Code Composer Studio demuestran la correcta comunicación half-duplex del sistema maestro/esclavos y una precisa estimación de la potencia producida por el módulo fotovoltaico.**

Palabras Clave **- Estimación de Potencia, Interfaz RS232, Módulo Fotovoltaico, MODBUS RS485, Piranómetro, Tiempo Real.**

Abstract **- This work presents the development and testing of a remote monitoring station based on the MODBUS RTU RS485 communication protocol to estimate in real time the theoretical power generated by a photovoltaic module as a function of its operating temperature and solar irradiance. This power estimation is important in photovoltaic systems since it allows evaluating the performance of the MPPT algorithms for extracting the maximum power required to operate the photovoltaic modules at its optimum voltage and current value. To achieve this, an Infwin PYR20 pyranometer and a Taidacent SHT30 temperature sensor have been configured as slaves while an Arduino UNO board is the master, which receives the measurement data from these sensors from a distance of approximately 30 m by means of the RS485 serial data transmission industrial standard. In turn, this information is sent through an RS232 interface to a Texas Instruments F28335 DSP programmed in Matlab-Simulink, where the estimated power is finally calculated in a continuous and real-time way for a 20W BCT20M-12 Blue Carbon photovoltaic module. The results obtained from the experimental tests carried out under different environmental conditions and displayed in real time in the Code Composer Studio IDE demonstrate the correct half-duplex communication of the master/slave system and a precise estimation of the power produced by the photovoltaic module.**

Keywords **- MODBUS RS485, Photovoltaic Module, Power Estimation, Pyranometer, Real-Time, RS232 Interface.**

I. INTRODUCCIÓN

Para monitorear correctamente la potencia que un módulo fotovoltaico puede generar se deben considerar las condiciones ambientales de irradiancia solar y temperatura del módulo fotovoltaico, ya que éstas influyen directamente con la energía producida. Llevando a cabo este monitoreo es posible realizar una estimación de la potencia que el módulo fotovoltaico es capaz de proveer (Madeti y Singh, 2017).

Debido a que las instalaciones fotovoltaicas se encuentran en zonas donde el sol irradia con gran intensidad, las cuales están alejadas del punto de consumo, es necesario utilizar protocolos de comunicación con señales de voltaje diferenciales en su salida para monitorear remotamente las variables eléctricas y climáticas involucradas (Triki-Lahiani et al., 2018). El protocolo de comunicación MODBUS RTU (Remote Terminal Unit - Unidad de Terminal Remota) es una opción viable para este propósito, ya que funciona a distancias de conexión de hasta 1200 m. Asimismo, para un cableado de longitud próxima a los 15 m, este protocolo permite una velocidad de transmisión de hasta 10 Mbps (Capocci et al., 2017). Así pues, el emplear el protocolo MODBUS en conjunto con el estándar RS485 permite realizar la comunicación remota con los sensores de irradiancia solar y de temperatura utilizando tan solamente 4 cables (A+, B, VCC y GND) para la comunicación Maestro-Esclavo (Krishna et al., 2019). Las líneas A+ y B- mencionadas anteriormente representan la transmisión de datos diferenciales no invertida e invertida, respectivamente.

La principal ventaja de utilizar la plataforma Arduino para el procesamiento de información es que al ser un software libre su programación está muy expandida a través de librerías en C++ (Herrador, 2009). El presente trabajo de investigación se basa en la librería "SensorModbusMaster.h" (https://github.com/EnviroDIY/SensorModbusMaster), la cual permite el intercambio de datos entre la tarjeta Arduino UNO y los sensores RS485 del módulo fotovoltaico.

El envío de datos a través del protocolo RS232 de la tarjeta Arduino hacia el DSP F28335 de Texas Instruments® es llevado a cabo de manera transparente pues ambas plataformas cuentan con sus respectivos pines de transmisión y recepción serial. En el caso de la Arduino se requiere un transceptor de voltaje MAX3232 donde se utilizan tan solo 3 cables (RX, TX y GND) haciendo posible la comunicación RS232 con el DSP.

Para el procesamiento de los datos recibidos en sistemas embebidos programados con lenguajes de bajo nivel (low-level) es común recurrir a operaciones bit a bit (bitwise) y utilizar máscaras (masks) (Vahid y Givargis, 1999). En este trabajo se utilizó una de estas máscaras basada en la operación binaria 'AND' para así extraer los datos de unidad, decena, centena y miles de los valores de irradiancia y temperatura enviados por los sensores.

De acuerdo a Skoplaki y Palyvos (2009), para la estimación de la potencia producida por un panel fotovoltaico existen una gran variedad de expresiones que son válidas, las cuales incluyen variables como son: la temperatura promedio de la celda solar (Bergene y Løvvik, 1995), los factores de corrección del espectro (Aste et al., 2008) y la eficiencia promedio mensual (Siegel et al., 1981), entre otros. En el presente trabajo, la estimación de la potencia se basó en la expresión lineal de la eficiencia como una función de la temperatura de la celda (Evans y Florschuetz, 1977).

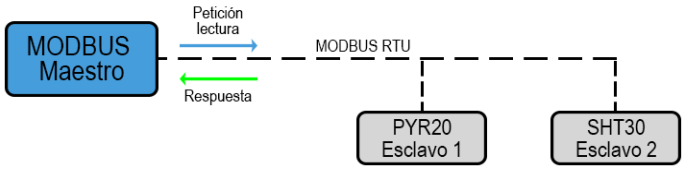
Así, este artículo de investigación presenta el desarrollo, la implementación y las pruebas de una estación remota de monitoreo capaz de estimar la potencia fotovoltaica en función de mediciones de irradiancia solar y de temperatura obtenida de sensores MODBUS RTU RS485. La contribución principal del presente trabajo es que esta estimación de potencia, permite calcular la eficiencia de los algoritmos de seguimiento del punto de máxima potencia (MPPT) y así evaluar su desempeño.

El presente artículo se organiza de la siguiente manera. En la sección II, se describe la parte técnica del artículo, la cual incluye tanto los sensores y dispositivos electrónicos empleados como la programación de la tarjeta Arduino y del DSP para que se comuniquen entre sí. Posteriormente, en la sección III, se muestran los resultados experimentales de las mediciones de irradiancia solar y de temperatura utilizando el IDE de Arduino, el IDE de Code Composer Studio y el osciloscopio digital. Además, también se presenta la estimación en tiempo real de la potencia generada por un panel fotovoltaico. Finalmente, la sección IV presenta las conclusiones.

II. PARTE TÉCNICA DEL ARTÍCULO

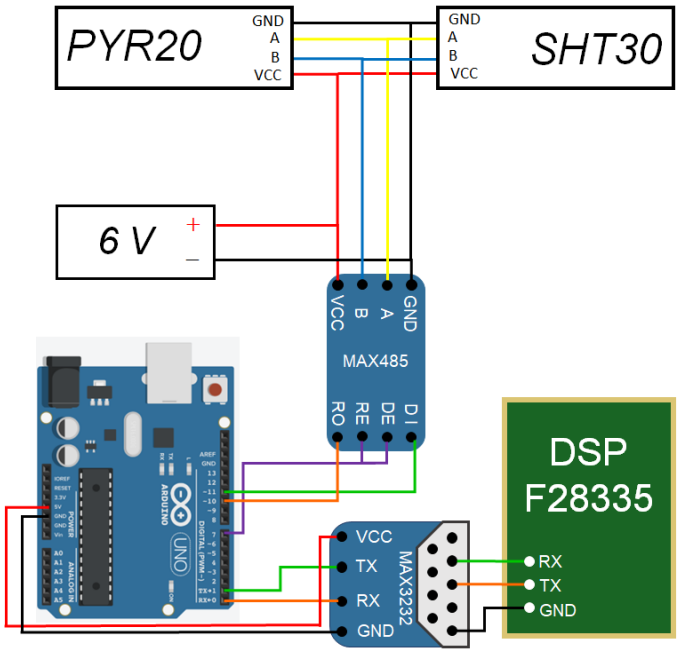
En la Figura 1, se muestra la conexión del maestro (Arduino) y los 2 sensores esclavos (irradiancia solar y temperatura) comunicados mediante el protocolo MODBUS RTU.

Figura 1
Comunicación MODBUS maestro-esclavos.



En la Figura 2, se muestra el diagrama general de conexión para el desarrollo de la estación remota de monitoreo. Este prototipo consiste en un microcontrolador Arduino UNO, un procesador de señales digitales DSP F28335, los sensores de irradiancia solar (PYR20) y de temperatura (SHT30), los transceptores de señal (MAX485 Y MAX3232) y la fuente de alimentación de 6V. El módulo MAX485 realiza la función de adaptar la señal diferencial de voltaje RS485 proveniente de los sensores a una señal tipo TTL compatible con la Arduino. Mientras que, el módulo MAX3232 convierte la señal de voltaje TTL de la Arduino a una RS232 compatible con el DSP.

Figura 2
Esquema de conexión de la estación remota de monitoreo.



2.1 Sensor PYR20 y SHT30

En la Figura 3, se muestran una foto real del sensor de irradiancia solar PYR20 y del sensor de temperatura SHT30, ambos basados en el protocolo MODBUS RTU RS485.

El sensor Infwin PYR20 tiene un rango de medición de 0 a $2000\ W/m^2$ y una resolución de $1\ W/m^2$, lo que lo hace ideal para investigación en aplicaciones fotovoltaicas. Por otro lado, el sensor SHT30 tiene un rango de medición de -40 a $+125\ °C$ y una resolución de $0.015\ °C$. Ambos sensores utilizan el protocolo MODBUS y tienen una interfaz RS485. La configuración predeterminada consiste en una dirección o ID de

Figura 3
(Izquierda) Sensor de irradiancia PYR20. (Derecha) Sensor de temperatura SHT30.

esclavo '1', una velocidad de transmisión de 9600 baudios por segundo (bps), 8 bits de datos, sin paridad y 1 bit de parada. La operación de lectura de la dirección conteniendo los datos de las mediciones de los sensores se realiza utilizando el código 0x03 correspondiente al registro de retención (holding register).

En la Tabla 1, se muestran las direcciones de escritura y lectura utilizadas tanto en el proceso de configuración como durante las mediciones.

Tabla 1
Direcciones de lectura y escritura en los sensores de irradiancia y temperatura.

SENSOR	*DIRECCIÓN (HEX/DEC)*	*TIPO DE OPERACIÓN*	*FUNCIÓN DE REGISTRO*	*RANGO*
PYR20	0x0000 / 0	RO	Irradiancia solar	0-2000 (W/m^2)
SHT30	0x0000 / 0	RO	Temperatura	0-65535
	0x0066 / 102	R/W	ID de esclavo	1-249
PYR20	0x0201 / 513	R/W	Velocidad de transmisión	0-6 0: 1200 bps 1: 2400 bps 2: 4800 bps 3: 9600 bps 4: 19200 bps 5: 38400 bps 6: 115200 bps*
SHT30	0x0067 / 103			

* Los 115200 bps aplican solo al sensor SHT30.
Nota: RO significa solo lectura, mientras que R/W es lectura/escritura.

Para la estación remota desarrollada únicamente se modificó el ID del sensor de temperatura de 1 a 2 y la velocidad de transmisión de datos predeterminada en ambos sensores. Conectando físicamente los pines A y B de los sensores al adaptador RS485-USB mostrado en la Figura 4 y utilizando el software QModBus fue posible escribir los nuevos valores de configuración.
En la Figura 5, se muestra el acceso a la función de código 0x06 correspondiente a la escritura de un registro individual, en la dirección 0x0201 (513) para enviar el dato 5 y así cambiar la velocidad de transmisión del piranómetro PYR20 de 9600 a 38400 bps.
En la Figura 6, se muestra el mismo proceso de escritura, pero ahora en la dirección 0x0067 (103) del sensor de temperatura SHT30.

Figura 4
Adaptador RS485-USB utilizado para escribir los registros de los sensores mediante el software QModBus.

Figura 5
Escritura del registro de velocidad de transmisión del piranómetro PYR20 utilizando el software QModBus.

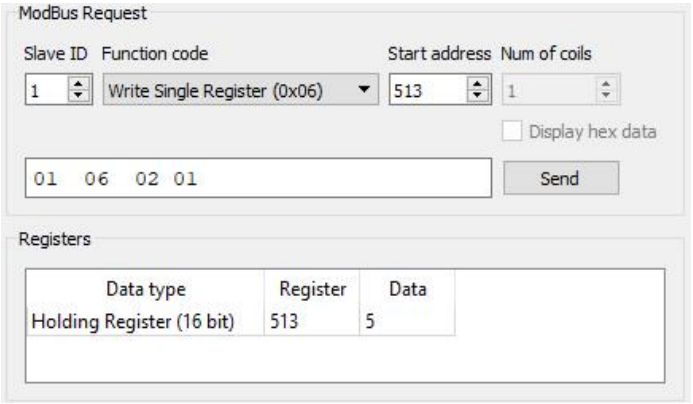
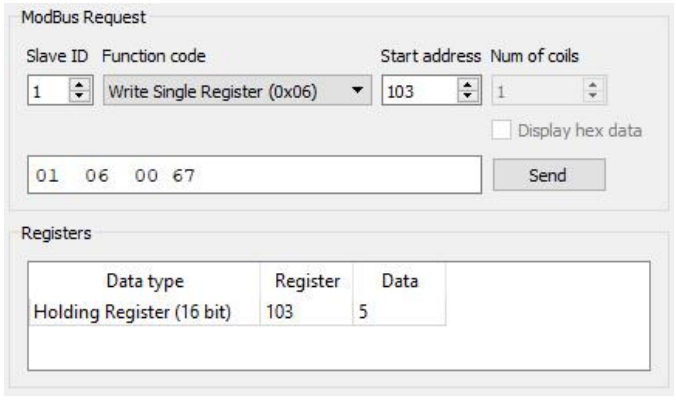


Figura 6
Escritura del registro de velocidad de transmisión del sensor de temperatura SHT30 utilizando el software QModBus.



Después de realizar la escritura a los sensores solo resta apagarlos, y encenderlos nuevamente para que surjan efecto los cambios realizados.

2.2 Módulos Transceptores MAX485 y MAX3232

Anteriormente, en la Figura 2 se mostró la conexión de los módulos MAX485 y MAX3232 en la estación remota de monitoreo. Ambos módulos son transceptores que permiten convertir niveles de tensión para que sean compatibles con otros módulos.

En la Figura 7 se muestra el transceptor MAX485, el cual es muy utilizado en redes de sensores y en comunicaciones a larga distancia. Este módulo se encarga de convertir el nivel de voltaje RS485 (típicamente ±5V) al nivel de voltaje TTL (0 y 5V) para que las señales sean adecuadas a los pines digitales de la tarjeta Arduino.

Figura 7
Módulo transceptor de voltaje MAX485.

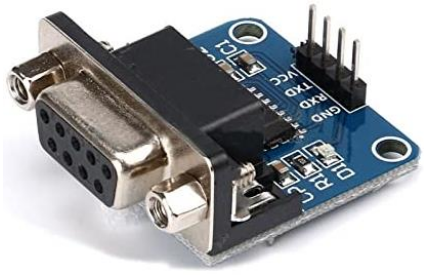
La Tabla 2, muestra la descripción de los 8 pines con los que cuenta el módulo MAX485.

Tabla 2
Pines del módulo MAX485.

PIN	PROPÓSITO
VCC	Voltaje de colector común, fuente de alimentación (5V)
GND	Tierra
A	Línea de transmisión de datos diferenciales no invertid
B	Línea de transmisión de datos diferenciales invertida
RO	Receiver Output - Salida de los datos recibidos
RE	Receiver Enable - Habilita la recepción de datos en 0
DE	Driver Enable - Habilita la transmisión de datos en 1
DI	Data Input - Entrada de los datos a transmitir

Es importante mencionar que, en el prototipo, los pines RE y DE se conectaron a un punto común, esto con el propósito de controlar con una sola señal y un solo pin de la tarjeta Arduino el cambio entre los modos de recepción y transmisión de datos, logrando así una comunicación half-duplex de manera más simplificada.
En la Figura 8, se muestra el transceptor MAX3232, el cual se encarga de convertir el nivel de voltaje TTL (0 y 5V) de la tarjeta Arduino al nivel de voltaje RS232 (típicamente ±12V). Este módulo cuenta con un conector DB9 con nueve pines, de los cuales solo se utilizan 2 (RX y GND) para enviar los datos de la Arduino al DSP de Texas Instruments.

Figura 8
Módulo transceptor de voltaje MAX3232.



La Tabla 3, muestra la descripción de los principales pines con los que cuenta el módulo MAX3232.

Tabla 3
Pines del módulo MAX3232.

PIN	PROPÓSITO
VCC	Voltaje de alimentación (3-5V)
GND	Tierra del lado TTL
RXD	Recepción del lado TTL
TXD	Transmisión del lado TTL
RX	Recepción del puerto DB9
TX	Transmisión del puerto DB9
GND	Tierra del puerto DB9

2.3 Programación de la Tarjeta Arduino

Para una comunicación half-duplex entre los sensores PYR20/SHT30 y la Arduino se utilizaron las librerías “Arduino.h”, “SensorModbusMaster.h” y “SoftwareSerial.h”. También se definió una variable sin signo de 16 bits, la cual almacena los valores de irradiancia y de temperatura. Además, en el protocolo MODBUS RTU, se definió la dirección de esclavo (SLAVE ID) para cada uno de los sensores conectados al bus de comunicación. Las variables declaradas en el programa son las siguientes:

```
uint16_t temperature = 0; // Variable que almacena los valores
byte modbusAddress1 = 0x01; // ID Piranómetro
byte modbusAddress2 = 0x02; // ID Sensor Temperatura
const int DEREPin = 7; // Modo Recepción/Transmisión
long modbusBaudRate = 38400; // Velocidad de comunicación
const int SSRxPin = 10;    // Datos recibidos RO-MAX485
const int SSTxPin = 11;   // Datos enviados DI-MAX485
SoftwareSerial modbusSerial(SSRxPin, SSTxPin);
modbusMaster modbus;
```

En la configuración inicial del programa “setup()”, se utilizó la variable “DEREPin” asignada al pin 7 de la Arduino para conectarse con los pines RE y DE, y así recibir/transmitir los datos a través del transceptor MAX485. A continuación, se muestra esa parte del código desarrollado:

```
if (DEREPin >= 0)
pinMode(DEREPin, OUTPUT);
Serial.begin(38400, SERIAL_8N1);
modbusSerial.begin(modbusBaudRate);
```

Finalmente, en la función que se ejecuta de manera continua “loop()”, se inicializa el protocolo MODBUS, así como la lectura de datos de los sensores, los cuales posteriormente se envían serialmente al DSP de Texas Instruments. El respectivo código se muestra a continuación:

```
modbus.begin(modbusAddress1, modbusSerial, DEREPin);
temperature = modbus.uint16FromRegister(0x03, 0x00, bigEndian);
if (temperature<1000 && temperature>=0)
temperature+=9000;
Serial.print(temperature);
delay(100);
modbus.begin(modbusAddress2, modbusSerial, DEREPin);
temperature = modbus.uint16FromRegister(0x03, 0x00, bigEndian);
Serial.print(temperature);
delay(100);
```

Tanto el valor de irradiancia solar como el de temperatura se almacenan en el registro "`0x00`". En el caso de la lectura de los datos del piranómetro se agregó una condición para convertir los números de 3 cifras a 4 cifras.

2.4 Comunicación de Datos entre la Arduino y el DSP

La Figura 9 muestra una lectura de los datos "crudos" (Raw Data) en el IDE de Code Composer Studio, los cuales son leídos por el DSP. Estos datos de 16 bits sin signo se almacenan en un arreglo con una longitud de dos (uint16[2]). Observe que el dato del sensor de irradiancia corresponde al número 1174 (W/m^2). Sin embargo, este dato decimal primero debe extraerse a partir del valor hexadecimal y después aplicando una máscara y realizando operaciones bit a bit (bitwise).

Figura 9

Ejemplo de una lectura de datos en el IDE de Code Composer Studio.

Name	Value	Type	Radix
Dato_Serial	0x0000C020	unsigned int[2]	hex
[0]	0x3131	unsigned int	hex
[1]	0x3734	unsigned int	hex

En la Figura 10, el recuadro de la izquierda muestra el procesamiento de los datos, en el cual se convierte el valor hexadecimal en decimal y se extrae la información útil. Para esto, primero se aplica un demultiplexor que separa los respectivos datos sin signo contenidos en los índices [0] y [1]. El valor de la unidad y de la centena se ubican en los 4 bits menos significativos de cada índice. Aplicando la máscara 0x000F ('0000000000001111' en binario) con el operador AND se extraen los valores de interés. Los valores de decena y de millar se ubican en la posición 9 a la 12 de la cadena de bits. En este caso, los datos se extraen aplicando la máscara 0x0F00 ('0000111100000000' en binario). Para las decenas y los millares, además es necesario hacer un desplazamiento aritmético de 8 bits para posicionar los valores extraídos en las unidades. Finalmente, para terminar la conversión a decimal del dato, este se multiplica ya sea por 1, 10, 100 o 1000, según corresponda, y después de sumarlos obtener el valor real medido por el sensor.

En Figura 10, el recuadro de la derecha muestra los bloques condicionales necesarios para separar el valor de irradiancia del de temperatura. El primer bloque condicional remueve el valor de 9000 a los datos que sean superiores a éste. Es importante mencionar que la cantidad de 9000 se sumó inicialmente en la tarjeta Arduino solo a los valores de irradiancia menores a 1000 W/m^2 para siempre enviar serialmente un dato de 4 dígitos. Los bloques condicionales de la parte derecha evalúan un límite (threshold) de 2000. Así, aquellos valores superiores al límite corresponden a datos de temperatura, mientras que los valores inferiores corresponden a datos de irradiancia. Por otro lado, cuando una condición no se cumple, simplemente se mantiene el valor anterior. Finalmente, para convertir la temperatura a grados Celsius, el dato obtenido se divide entre 100.

2.5 Potencia Estimada del Panel Fotovoltaico

La potencia P_{PV} producida por un módulo o arreglo fotovoltaico, se puede estimar en función de la irradiancia solar y de la temperatura mediante la siguiente expresión (Evans y Florschuetz, 1977; Skoplaki y Palyvos, 2009):

$$P_{PV} = A \cdot G \cdot n \cdot (1 - \beta_r \cdot (T - T_{STC})) \quad (1)$$

donde

- A es la superficie del panel fotovoltaico $(0.117m^2)^*$
- G es la irradiancia solar
- n es la eficiencia del panel fotovoltaico $(15\%)^*$
- β_r es el coeficiente del decremento de eficiencia por unidad de incremento de temperatura $(0.5\%/°C)^*$
- T es la temperatura promedio del panel fotovoltaico
- T_{STC} es la temperatura de condiciones de prueba estándar

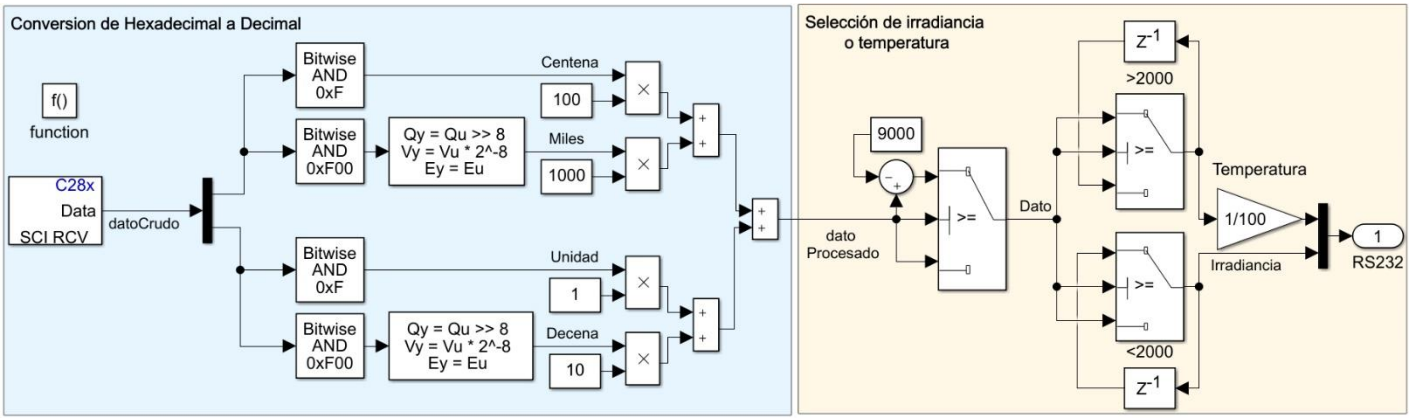
* Nota: Estos datos son aproximados para el modelo utilizado (Blue Carbon BCT20M-12).

2.6 Implementación de la Estación Remota de Monitoreo

En la Figura 11, se muestra la implementación del prototipo de la estación remota de monitoreo. Específicamente, en la Figura 11a, se observa el sensor de irradiancia y el sensor temperatura junto al módulo PV. Como se mencionó previamente, solo se requirieron 4 cables (A+, B-, VCC y GND) tanto para comunicar los sensores como para alimentarlos.

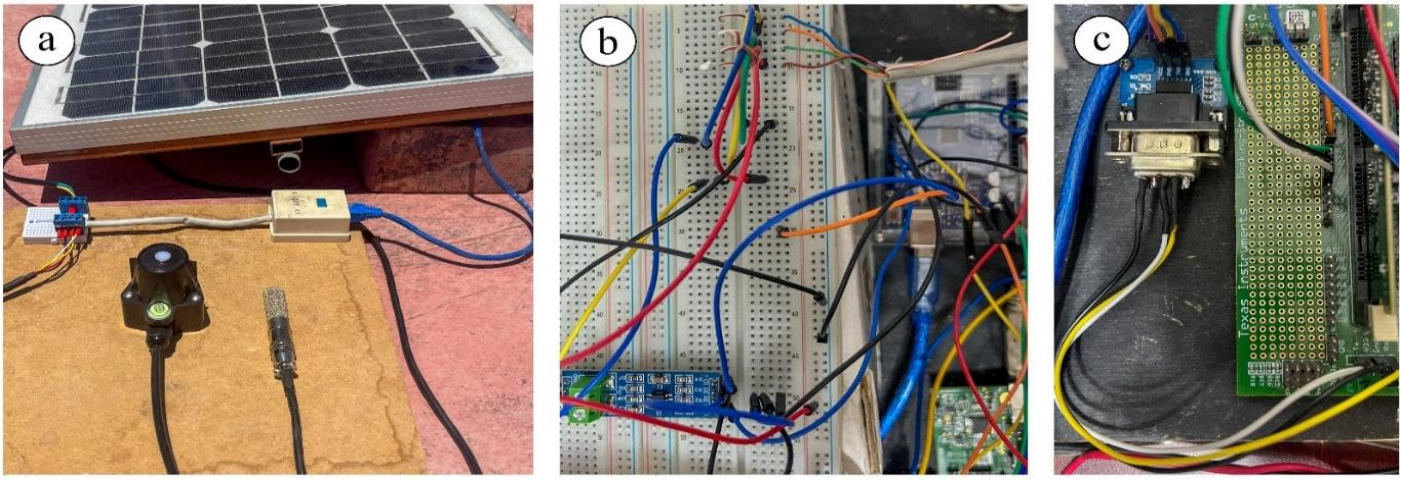
Figura 10

Procesamiento de los datos provenientes de la Arduino a través del puerto serial RS232.



Año: 2024. Volumen: 1
Numero:10. ISSN: 2448-623X

Figura 11
Prototipo de la estación remota de monitoreo: a) Piranómetro y sensor de temperatura, b) MAX485 y Arduino, c) MAX3232 y DSP Texas Instruments.



En la Figura 11b, se muestra la conexión del transceptor MAX485 con la tarjeta Arduino UNO, la cual recibe los datos provenientes de los sensores de irradiancia solar y temperatura.
En la Figura 11c, se muestra la conexión del pin TX de la Arduino con el transceptor MAX3232 y finalmente con el pin RX del puerto serial del DSP F28335.

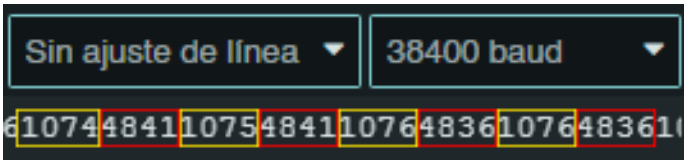
III. RESULTADOS Y DISCUSIÓN

En esta sección se describen los resultados obtenidos de la implementación de la estación remota de monitoreo. Primero, se presentan las mediciones de irradiancia solar y de temperatura en el IDE de Arduino. Después, en el IDE de Code Composer Studio se despliegan los resultados del cálculo de la potencia estimada. Finalmente, se muestran en el osciloscopio digital las formas de onda de irradiancia y de potencia estimada.

3.1 Resultados en el IDE de Arduino

En la Figura 12, se muestran las mediciones en tiempo real de irradiancia solar y temperatura desplegadas en el monitor serie del IDE de Arduino. Observe que la trama de datos seriales se transmiten sin salto de línea.

Figura 12
Desplegado de los valores de irradiancia solar y temperatura en el monitor serie de Arduino.



Los valores enmarcados en color amarillo corresponden a mediciones de irradiancia solar (W/m^2), mientras que, los valores encerrados en color rojo son mediciones de temperatura. Observe que ésta, aún se encuentra en valor entero y en miles, por lo que requiere ser dividido entre 100 en el DSP de Texas Instruments, como previamente se explicó y mostró en la Figura 10. Así, por ejemplo, el dato 4841 en realidad corresponde a 48.41°C.

3.2 Resultados en el IDE de Code Composer Studio

En la Figura 13, se observan las variables de irradiancia solar, temperatura y potencia estimada para el módulo fotovoltaico BCT20M-12, todas ellas desplegadas en tiempo real en la ventana de observación (watch window) de Code Composer Studio. A diferencia de los resultados mostrados en el IDE de Arduino, aquí los valores están en punto fijo iq(32,17) y ya en sus unidades finales de W/m^2, °C y W, respectivamente.

Figura 13
Valores de irradiancia solar, temperatura y potencia estimada del panel fotovoltaico desplegados en la ventana de observación de Code Composer Studio.

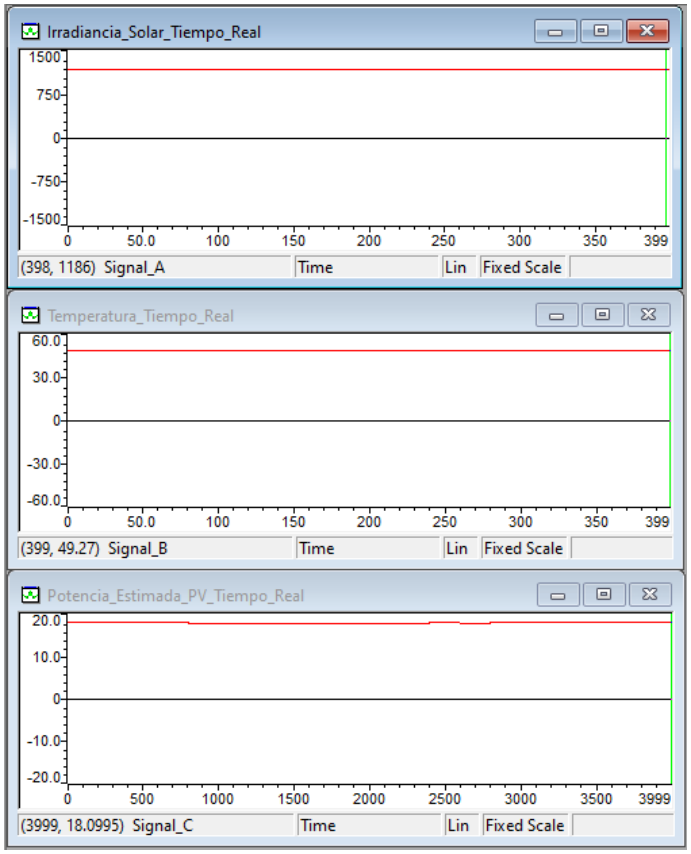
Name	Value	Type	Radix
Irradiancia	1031.0	long	qvalue(17)
Temperatura	47.619995...	long	qvalue(17)
Potencia_pv	16.003021...	long	qvalue(17)

En la Figura 14, se muestran las respectivas formas de onda para la irradiancia solar, la temperatura y la potencia estimada monitoreadas en tiempo real en el IDE de Code Composer Studio. Observe que las curvas de irradiancia y de temperatura se conforman por 400 muestras cada una, mientras que la gráfica para la potencia estimada consiste de 4000 muestras. Es importante mencionar que la frecuencia de muestreo utilizada es de 2kHz, por lo que el periodo de muestreo en las tres curvas es de 500μs.

3.3 Resultados en el Osciloscopio Digital

Una vez que el DSP de Texas Instruments calcula la potencia estimada en base a la irradiancia solar y a la temperatura, es posible enviarla el resultado a través de un pin del procesador para observarla en un osciloscopio digital.
Para esto, primero se normalizan (rango de 0-1) tanto los valores medidos de irradiancia como los calculados de la potencia estimada. Después, se utilizan cada uno de estos valores como una señal modulante en una salida PWM del DSP. Finalmente, a la señal de pulsos cuadrados resultante, se le aplica un filtro pasivo pasabajas consistente de una red RC.

Figura 14
Gráficas de irradiancia solar, temperatura y potencia estimada del panel fotovoltaico desplegadas en tiempo real en Code Composer Studio.



En la Figura 15, se muestran la irradiancia solar, la potencia estimada del panel fotovoltaico y la potencia real extraída aplicando un algoritmo de seguimiento del punto de máxima potencia (MPPT). Estos resultados fueron obtenidos en un día parcialmente nublado. Observe como la irradiancia y las potencias varían muy lentamente en el periodo mostrado de 3 minutos con 20 segundos.

Figura 15
Formas de onda de irradiancia solar, potencia estimada del panel fotovoltaico y potencia real extraída con el algoritmo MPPT.

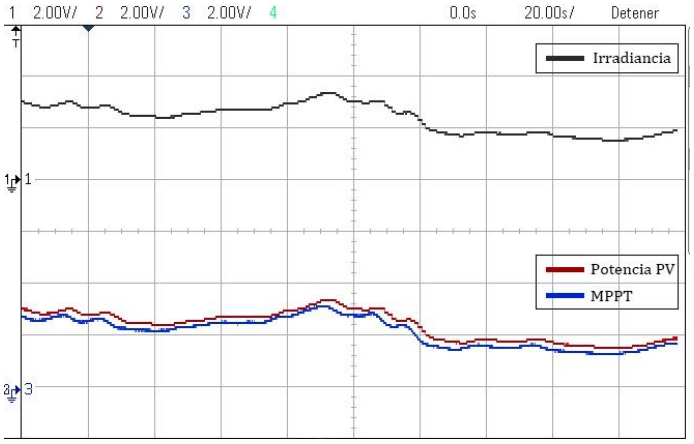
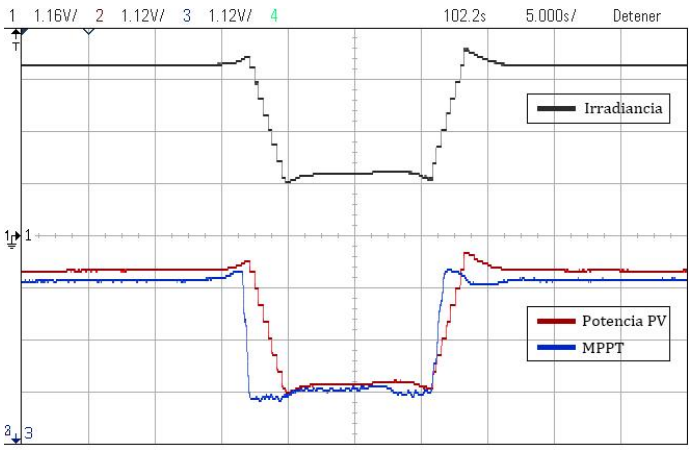


Figura 16
Formas de onda de irradiancia solar, potencia estimada del panel fotovoltaico y potencia real extraída con el algoritmo MPPT ante cambios súbitos de irradiancia.



En la Figura 16, se muestra el comportamiento de la irradiancia, la potencia estimada y la potencia real extraída con un algoritmo MPPT ante un decremento y un incremento súbito en la irrandiancia solar. Estos resultados se obtuvieron en un día despejado y soleado. Los cambios de irradiancia solar se realizaron cubriendo y descubriendo rápidamente tanto el panel fotovoltaico como el piranómetro con una tela delgada semitransparente. Como se puede apreciar en las curvas, la irradiancia y la potencia estimada tienen un perfil trapezoidal, esto es debido al tiempo de respuesta intrínseco del piranómetro. Sin embargo, la potencia extraída con el algoritmo MPPT sí muestra la variación real aplicada tipo escalón. Es importante resaltar como el algoritmo de seguimiento alcanza correctamente y de manera casi instantánea el punto óptimo de operación del módulo fotovoltaico.

La Tabla 4, presenta los valores promedio de las mediciones de irradiancia, temperatura y potencia estimada del panel fotovoltaico antes y después del cambio súbito de irradiancia mostrado en la Figura 16.

Tabla 4
Valores promedio de irradiancia solar, temperatura y potencia estimada antes y después del cambio súbito de irradiancia.

IRRADIANCIA (W/m^2)	***TEMPERATURA*** $(°C)$	***POTENCIA*** (W)
1080	45.17	17.24
380	45.13	5.97

IV. CONCLUSIONES

Este artículo presentó el desarrollo de una estación remota de monitoreo basada en MODBUS RTU RS485 para la estimación de la potencia generada por el módulo fotovoltaico BCT20M-12. Gracias al protocolo MODBUS y al estándar RS485, esta estación remota fue capaz de enviar de manera correcta y confiable los datos de los sensores de irradiancia solar y temperatura a una distancia aproximada de 30 metros con una velocidad de transmisión de 38400 bps. La tarjeta Arduino se

comunicó correctamente con el DSP Texas Instruments al enviarle las mediciones de los sensores por medio del estándar de comunicación RS232, permitiéndole así calcular en tiempo real la potencia estimada del panel fotovoltaico. Los resultados obtenidos prueban que el prototipo desarrollado de la estación remota de monitoreo funciona eficazmente, por lo que es muy útil para evaluar el desempeño de algoritmos MPPT, para identificar si existe algún problema con el módulo fotovoltaico o incluso para saber si es necesario llevar a cabo algún mantenimiento de limpieza.

V. AGRADECIMIENTOS

K. G. Vidaña-Aldaba agradece al Consejo Nacional de Humanidades, Ciencias y Tecnologías (CONAHCYT) por el apoyo económico otorgado en el marco del "Programa de Becas Nacionales para Estudios de Posgrado" con número de solicitud 2022-000002-01NACF-00971.
Los autores también agradecen al TecNM, al Instituto Tecnológico de La Laguna y al Instituto Tecnológico Superior de Lerdo (ITSL) por el apoyo económico recibido para el desarrollo del presente trabajo de investigación.
Igualmente, agradecemos al subdirector de investigación MDGPT. Jesús Alejandro Valdés Nieblas, por el apoyo brindado durante la estancia de investigación realizada en el ITSL donde se obtuvieron los resultados experimentales presentados en este trabajo.

VI. REFERENCIAS

Aste, N., Chiesa, G., y Verri, F. (2008). Design, development and performance monitoring of a photovoltaic-thermal (PVT) air collector. *Renewable Energy*, 33(5), 914–927. https://doi.org/10.1016/j.renene.2007.06.022

Bergene, T., y Løvvik, O. M. (1995). Model calculations on a flat-plate solar heat collector with integrated solar cells. *Solar Energy (Phoenix, Ariz.)*, 55(6), 453–462. https://doi.org/10.1016/0038-092x(95)00072-y

Capocci, R., Dooly, G., Omerdić, E., Coleman, J., Newe, T., y Toal, D. (2017). Inspection-class Remotely Operated Vehicles—A review. *Journal of Marine Science and Engineering*, 5(1), 13. https://doi.org/10.3390/jmse5010013

Evans, D., y Florschuetz, L. (1977). Cost studies on terrestrial photovoltaic power systems with sunlight concentration. *Solar Energy*, 19(3), 255–262. https://doi.org/10.1016/0038-092x(77)90068-8

Herrador, R. E. (2009). *Guía de usuario de Arduino*. Universidad de Córdoba. [Archivo PDF]. https://www.uco.es/aulasoftwarelibre/wp-content/uploads/2010/05/Arduino_user_manual_es.pdf

Krishna, M. S. R., Dinesh, K., y Shanbog, N. S. (2019). Low Cost Remote Monitoring of Solar Plant through RS485 Communication. International *Journal of Innovative Technology and Exploring Engineering*, 8(9), 3034–3037. https://doi.org/10.35940/ijitee.i8166.078919

Madeti, S. R., y Singh, S. N. (2017). Monitoring system for photovoltaic plants: A review. *Renewable and Sustainable Energy Reviews*, 67, 1180–1207. https://doi.org/10.1016/j.rser.2016.09.088

Siegel, M., Klein, S., y Beckman, W. (1981). A simplified method for estimating the monthly-average performance of photovoltaic systems. *Solar Energy*, 26(5), 413–418. https://doi.org/10.1016/0038-092x(81)90220-6

Skoplaki, E., y Palyvos, J. A. (2009). On the temperature dependence of photovoltaic module electrical performance: A review of efficiency/power correlations. *Solar Energy (Phoenix, Ariz.)*, 83(5), 614–624. https://doi.org/10.1016/j.solener.2008.10.008

Triki-Lahiani, A., Bennani-Ben Abdelghani, A., y Slama-Belkhodja, I. (2018). Fault detection and monitoring systems for photovoltaic installations: A review. *Renewable and Sustainable Energy Reviews*, 82, 2680–2692. https://doi.org/10.1016/j.rser.2017.09.101

Vahid, F., y Givargis, T. D. (2003). *Embedded System Design: A Unified Hardware/Software Introduction* (1st ed.). John Wiley & Sons.

VII. AUTORES

Karol Gabriel Vidaña Aldaba
https://orcid.org/0000-0003-2022-1516

Jorge Lara Cardoso
https://orcid.org/0000-0002-2746-7044

Concepción Hernández Flores
https://orcid.org/0000-0002-4757-5309

Marco Antonio Arjona López
https://orcid.org/0000-0003-1826-4066