

**DISEÑO E IMPLEMENTACIÓN DE UN
MEDIDOR MONOFÁSICO CON
COMUNICACIÓN INALÁMBRICA Y UN
SISTEMA DE GENERACIÓN DE REPORTE
DE CONSUMO**

Proyecto de Ingeniería

Ciclo I 2013

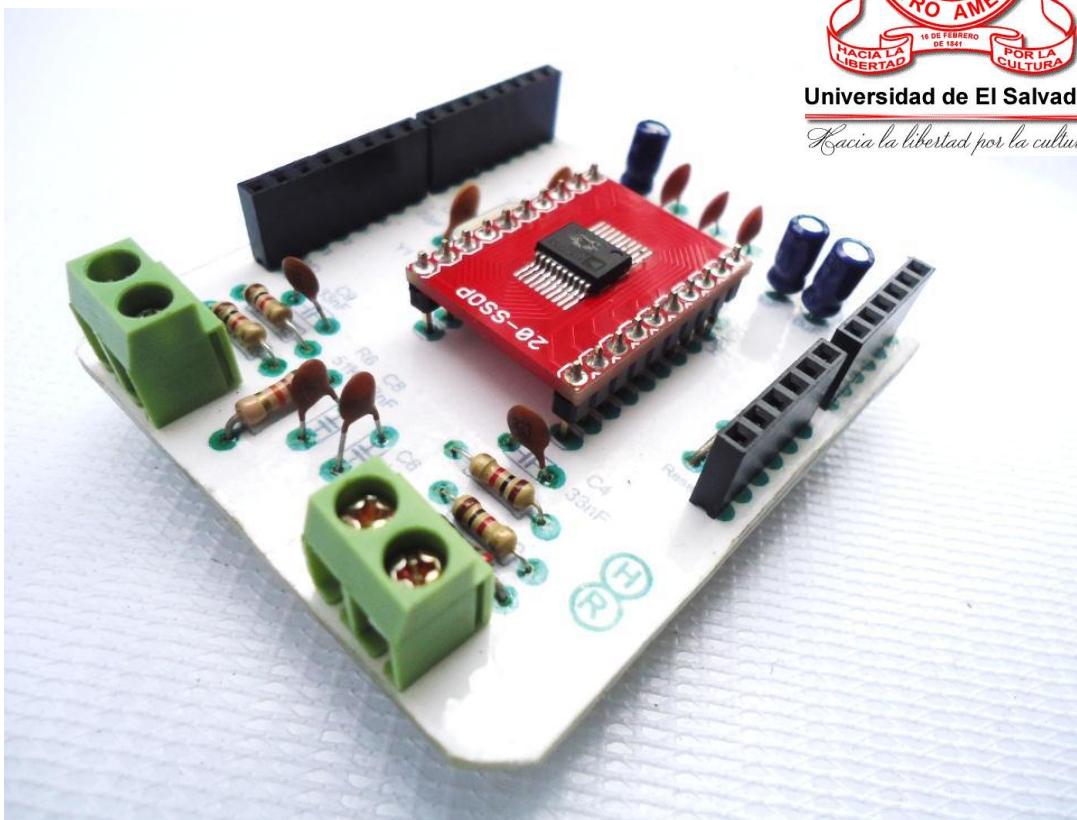
Héctor Franco y Ricardo Pacheco

**UNIVERSIDAD DE EL SALVADOR
FACULTAD DE INGENIERÍA Y ARQUITECTURA
ESCUELA DE INGENIERÍA ELÉCTRICA
PROYECTO DE INGENIERIA**



Universidad de El Salvador

Hacia la libertad por la cultura



Presentan:

Héctor Alcides Franco Paredes.

Ricardo José Pacheco Méndez.

Coordinador:

Ing. José Wilber Calderón Urrutia.

Ciudad universitaria, ciclo I 2013.

TABLA DE CONTENIDO

INTRODUCCION	6
OBJETIVOS	7
CAPÍTULO 1: ADE7753.....	8
DESCRIPCIÓN GENERAL.....	8
CARACTERÍSTICAS.....	8
DESCRIPCIÓN DE LA CONFIGURACIÓN Y FUNCIONES DE LOS PINES	9
DIAGRAMA DE BLOQUES.....	12
CANAL 1.....	12
CÁLCULO DE CORRIENTE RMS.....	13
CANAL 2.....	14
CÁLCULO DE VOLTAJE RMS	14
CÁLCULO DE POTENCIA ACTIVA	15
CÁLCULO DE ENERGÍA.....	17
CALIBRACIÓN DEL OFFSET PARA LA POTENCIA.....	19
CONVERSIÓN DE ENERGÍA A FRECUENCIA.....	19
MODO DE ACUMULACIÓN POSITIVA SOLAMENTE.....	20
UMBRAL DE NO-CARGA	20
CÁLCULO DE LA POTENCIA REACTIVA.....	20
SIGNO DEL CÁLCULO DE LA POTENCIA REACTIVA	21
FRECUENCIA CLKIN.....	22
ADE7753 INTERFACE SERIAL	22
ADE7753 OPERACIÓN ESCRITURA SERIAL.....	24
ADE7753 OPERACIÓN DE LECTURA SERIAL.....	25
REGISTROS DEL ADE7753	25
REGISTRO DE MODO	28
CAPITULO 2: XBEE 802.15.4 [SERIE 1]	30
MÓDULO XBEE	30
CARACTERÍSTICAS PRINCIPALES	30
ESPECIFICACIONES	31
PINES DE SEÑAL.....	32

MODOS DE OPERACIÓN	33
MODO DE RECEPCIÓN/TRANSMISIÓN	33
MODO “SLEEP” DE BAJO CONSUMO.....	34
MODO DE COMANDO	36
MODO TRANSPARENTE.....	36
MODO API	37
IDLE.....	38
SOFTWARE X-CTU.....	38
CONFIGURACIÓN DE LOS MÓDULOS	39
DIRECCIONAMIENTO	39
MODO DE CONEXIÓN TRANSPARENTE	40
MODO DE CONEXIÓN API.....	46
COMANDOS IMPORTANTES	47
CAPÍTULO 3: PROCEDIMIENTOS PRELIMINARES DE INTEGRACIÓN ADE7753-ARDUINO-LABVIEW.	51
INTEGRACIÓN ENTRE ARDUINO Y ADE7753 - DATALOGGER.....	51
LIBRERIAS E INICIALIZACIONES PRINCIPALES PARA COMUNICACIÓN ADE7753-ARDUINO.....	51
LIBRERIAS, SENTENCIAS IMPORTANTES PARA INTEGRACIÓN ADE7753-DATALOGGER-ARDUINO.....	53
LABVIEW.....	55
LABVIEW INTERFACE FOR ARDUINO (LVIFA O LIFA)	56
ARDUINO TOOLKIT	56
OBTENCIÓN E INSTALACIÓN DEL ARDUINO TOOLKIT	57
HERRAMIENTAS DEL LVIFA.....	60
NI-VISA	61
Ni VISA contra LVIFA.....	62
LVIFA VENTAJAS	62
LVIFA DESVENTAJAS	63
NI-VISA VENTAJAS	63
NI-VISA DESVENTAJAS	63
MÉTODO A UTILIZAR	63
CAPÍTULO 4: HARDWARE PARA IMPLEMENTACIÓN DEL MEDIDOR.....	64

DIAGRAMA DE CONEXIÓN DEL CIRCUITO.....	64
DISEÑO DEL PCB.....	64
IMPRESIÓN DEL PCB.....	67
SHIELD ARDUINO MEDIDOR ADE7753	70
CAPÍTULO 5 SOFTWARE LABVIEW.....	73
INTERFAZ GRÁFICA LABVIEW	73
INICIO	75
DETECCIÓN DE DISPOSITIVO INALÁMBRICO.....	75
BLOQUE BÁSICO DE COMUNICACIÓN SERIAL	76
MEDICIÓN EN TIEMPO REAL	79
SINCRONIZACIÓN DE LA COMUNICACIÓN	81
ACONDICIONAMIENTO DE DATOS	82
OPTIMIZANDO ESPACIO	83
DESCARA DE MEDICIONES	84
ACONDICIONAMIENTO DE DATOS	87
SALIR.....	88
CAPITULO 6: CÓDIGO ARDUINO, CONTROL DEL CIRCUITO INTEGRADO.....	89
DETALLE DEL CÓDIGO: RUTINAS DE LECTURA, FUNCIONES MÁS IMPORTANTES, MODOS DE OPERACIÓN	89
CAPÍTULO 7: RESULTADOS	95
CARGA DE PRUEBA.....	95
MEDICIONES.....	95
GRÁFICAS.....	96
CONCLUSIONES	99
ANEXOS	100
ANEXO 1	100
EJEMPLO1: CONTROL BASICO DEL ADE7753	100
ADE7753 INTERFACE SERIAL	101
ANEXO 2	106
EJEMPLO 2: LVIFA.....	106
ANEXO 3	111

HOJA DE DATOS DE LA PINZA DE CORRIENTE 111

INTRODUCCION

Este proyecto está cimentado en las capacidades de medición de energía del IC ADE7753, este IC cuenta con los convertidores analógicos-digitales y procesadores digitales de señal que dan una enorme fiabilidad en los resultados de las mediciones, puesto que el trabajo arduo que requiere el cálculo de potencia activa, aparente y de valores RMS de voltaje y corriente y energía consumida tienen lugar dentro del IC; a los datos de estos cálculos es posible acceder a través de una interface serial usando un micro controlador, para este proyecto se ha utilizado la plataforma de hardware libre Arduino que sirve como medio de control y comunicación con el IC; el primer capítulo de este documento contiene una descripción general del ADE7753 y su forma de trabajo.

El medidor cuenta con un soporte de almacenamiento de datos en tiempo real mientras realiza las diferentes mediciones y cálculos, para esto se utiliza una shield para Arduino que permite guardar datos con la fecha y hora que fueron medidos en una memoria SD. Además el medidor puede ser controlado en forma remota y está diseñado para operar de manera autosuficiente, sin necesidad de estar conectado a una PC, se puede acceder al historial de mediciones desde una distancia de alrededor siete metros y hacer un monitoreo en tiempo real de voltaje, corriente, potencia y energía consumida; una descripción de la forma en que la comunicación inalámbrica tiene lugar se detalla en el capítulo dos.

Todo esto es posible a través de una interfaz de usuario desarrollada en LabVIEW, se utilizan las herramientas que este programa tiene para manipular el puerto serie donde se conecta la XBee que es el dispositivo que permite la comunicación inalámbrica, desde la interfaz creada en LabVIEW es posible no solamente recolectar datos sino también acceder a diferentes modos de operación del medidor, inclusive ejecutar comandos básicos sobre el IC.

El capítulo tres muestra la manera en que se ha llevado acabo la integración de los diferentes componentes utilizados, esta descripción en general tiene como objetivo dar una idea clara todo lo que conforma el medidor. Los restantes capítulos son una descripción del hardware, la programación con software LabVIEW y Arduino.

OBJETIVOS

Objetivo General:

- ✚ Construir un medidor de energía utilizando las características del ADE7753, controlándolo a través de la plataforma Arduino, que permita el almacenamiento de los datos censados y el acceso de forma inalámbrica desde una interfaz de usuario.

Objetivos específicos:

- ✚ Obtener monitoreo de los valores rms de las señales de voltaje y corriente.
- ✚ Crear una interfaz para el control de medidor en LabVIEW.
- ✚ Permitir un acceso inalámbrico al medidor utilizando unas antenas XBee.
- ✚ Censar y almacenar continuamente los datos de voltaje, corriente, potencia en una memoria que permita un respaldo y acceso futuro a los datos.
- ✚ Generar un reporte de consumo que contenga las mediciones documentadas y el consumo total durante ese periodo.
- ✚ Obtener el valor de energía consumida.

CAPÍTULO 1: ADE7753

DESCRIPCIÓN GENERAL

El ADE7753 característicamente posee ADC's y DSP¹ para una alta precisión sobre grandes variaciones en las condiciones ambientales y de tiempo. El ADE7753 incorpora dos ADC's de segundo orden de 16-bit, un integrador digital (en CH1), circuitos de referencia, sensor de temperatura, y todo el procesamiento de señales requerido para realizar las mediciones de energía activa, reactiva y aparente, el periodo del voltaje de línea, y el cálculo de rms en el voltaje y la corriente.

El integrador digital disponible en el chip proporciona una interfaz directa a sensores de corriente di/dt, tales como bobinas de Rogowski², eliminando la necesidad de un integrador analógico externo y que resulta en una excelente estabilidad a largo plazo y la coincidencia precisa de la fase entre los canales de corriente y tensión.

El ADE7753 proporciona una interfaz serie para leer los datos, y una frecuencia de salida de pulsos (CF), que es proporcional a la potencia activa. Diversas funciones de calibración del sistema, es decir, la corrección de compensación del canal, calibración de fase, y calibración de potencia, garantizan una alta precisión. El dispositivo también detecta las variaciones de tensión bajas o altas de corta duración.

El modo de sólo acumulación positiva da la opción de acumular energía solamente cuando la potencia positiva se detecta. Un umbral interno sin carga asegura que el dispositivo no presente ninguna deformación cuando no hay carga. La salida de cruce por cero (ZX) produce un pulso que está sincronizado con el punto de cruce por cero de la tensión de línea. Esta señal se utiliza internamente en los modos de ciclo de línea de acumulación de energía activa y aparente, lo que permite una rápida calibración.

El registro de estado de interrupción indica la naturaleza de la interrupción y el registro de habilitación de interrupción controla que evento se produce, una salida en el pin IRQ, un drenaje abierto, salida activa lógico bajo.

CARACTERÍSTICAS

- ✚ Alta precisión, admite IEC 60687/61036/61268 y IEC 62053-21/62053-22/62053-23³.
- ✚ El integrador digital interno permite una interfaz directa a sensores de corriente con salidas di/dt.

¹ Convertidor Analógico Digital (ADC) y Procesamiento digital de señales (DSP).

² La bobina de Rogowski son transductores para medir corriente alterna, su creador fue Walter Rogowski.

³ Estándares de la Comisión Electrotécnica Internacional.

- ✚ Un PGA⁴ en el canal de corriente permite una interfaz directa a derivaciones y transformadores de corriente.
- ✚ Energía activa, reactiva y aparente; muestrear formas de onda; tensión rms y corriente rms.
- ✚ Menos del 0.1% de error en la medición de energía activa a través de un rango dinámico de 1000-1 a 25 ° C.
- ✚ Modo de acumulación de energía (solo positiva) disponible.
- ✚ Umbral programable por el usuario para sobretensión de línea y supervisión PSU⁵ y SAG⁶.
- ✚ Calibración digital para potencia, fase y desplazamiento de entrada.
- ✚ Sensor de temperatura interno (± 3 ° C típico).
- ✚ Interfaz serial compatible SPI.
- ✚ Salida de impulsos con una frecuencia programable.
- ✚ Pin de solicitud de interrupción (IRQ) y registro de estado.
- ✚ Referencia de 2.4 V con capacidad de sobre marcha externo.
- ✚ Alimentación única de 5V, de baja potencia (25 mW típico).

DESCRIPCIÓN DE LA CONFIGURACIÓN Y FUNCIONES DE LOS PINES

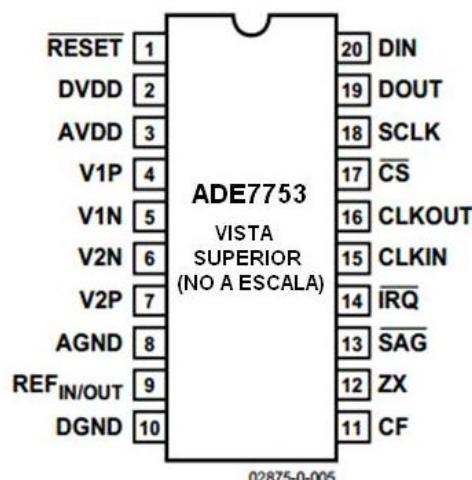


Figura 1.1 Configuración de pines (Empaquetado SSOP).

1. RESET : Pin de reposición del ADE7753. Un valor lógico bajo en este pin mantiene a los ADC's y la circuitería digital (incluyendo la interfaz en serie) en una condición de reposición.

⁴ Amplificador Programable de Ganancia, se necesita para mejorar el rango de conversión para detectar pequeñas y grandes corrientes.

⁵ Unidad de Suministro de Potencia.

⁶ Se refiere a las caídas o sobre tensiones debidos a transitorios.

2. DVDD: Fuente de Alimentación Digital. Este pin proporciona la tensión de alimentación para los circuitos digitales en el ADE7753. La tensión de alimentación debe mantenerse a $5\text{ V} \pm 5\%$ para la operación especificada. Este pin debe estar desacoplado con DGND con un condensador de $10\text{ }\mu\text{F}$ en paralelo con un condensador cerámico 100 nF .
3. AVDD: Fuente de alimentación analógica. Este pin proporciona la tensión de alimentación para los circuitos analógicos en el ADE7753. La alimentación debe ser mantenida a $5\text{ V} \pm 5\%$ para la operación especificada. Debe hacerse un desacople adecuado para minimizar el rizado en la fuente de alimentación y el ruido en este pin. Este pin debe estar desacoplado a AGND con un condensador de $10\text{ }\mu\text{F}$ en paralelo con un condensador cerámico 100 nF .
4. V1P y 5 V1N: Entradas analógicas del canal 1. Este canal está diseñado para su uso con un transductor de di/dt de corriente tal como una bobina de Rogowski u otro sensor de corriente tal como una derivación o transformador de corriente (CT). Estas entradas son entradas de tensión completamente diferencial con niveles máximos de la señal diferencial de entrada de $\pm 0.5\text{ V}$, $\pm 0.25\text{ V}$ y $\pm 0.125\text{ V}$, dependiendo de la escala seleccionada. El canal 1 también tiene un PGA con selecciones de ganancia de 1, 2, 4, 8, o 16. El nivel de señal máximo en estos pines con respecto a AGND es de $\pm 0.5\text{ V}$. Ambas entradas tienen circuitos internos de protección ESD, y, además, una sobretensión de $\pm 6\text{ V}$ puede ser sostenida en estas entradas sin riesgo de daño permanente.
6. V2N y 7 V2P: Entradas analógicas del Canal 2. Este canal está diseñado para su uso con el transductor de tensión. Estas entradas son entradas de tensión completamente diferenciales con una diferencia de nivel de señal máximo de $\pm 0.5\text{ V}$. Este canal también tiene un PGA con selecciones de ganancia de 1, 2, 4, 8, o 16. El nivel máximo de la señal en los pines con respecto a AGND es de $\pm 0.5\text{ V}$. Las dos entradas tienen circuitos internos de protección ESD, y una sobretensión de $\pm 6\text{ V}$ puede ser sostenido en estas entradas sin riesgo de daño permanente.
8. AGND: Tierra de referencia analógica. Este pin proporciona la referencia de tierra para la circuitería analógica del ADE7753, es decir, ADC's y de referencia. Este pin debe estar vinculado a la placa de tierra analógica o la referencia de tierra con menos ruido en el sistema. Esta referencia de tierra tranquila debe ser utilizada para todos los circuitos analógicos, por ejemplo, filtros anti-aliasing⁷, transductores de corriente y de tensión, etc., para mantener el ruido de tierra alrededor del ADE7753 a un mínimo, el plano de tierra tranquila debe estar conectado al plano de tierra digital en sólo un punto. Es aceptable para colocar todo el dispositivo en el plano de tierra analógica.
9. REF_{IN/OUT}: El acceso a la tensión de referencia en el Chip. La referencia en el chip tiene un valor nominal de $2.4\text{ V} \pm 8\%$ y un coeficiente de temperatura típico de $30\text{ ppm} / {}^\circ\text{C}$. Una fuente de referencia externa también se puede conectar a este pin. En cualquier caso, este pin debe estar desacoplado a AGND con un condensador de $1\text{ }\mu\text{F}$ cerámico.

⁷ Es un filtro paso bajo que esta antes de la conversión digital analógica y tiene como objetivo eliminar la presencia de frecuencias superiores $\frac{f_s}{2}$ donde f_s es la frecuencia de muestreo.

10. DGND: Tierra de referencia Digital. Este pin proporciona la referencia de tierra para la circuitería digital en la ADE7753, es decir, multiplicador, filtros, y de digital a un convertidor de frecuencia. Debido a que las corrientes de retorno digitales en el ADE7753 son pequeñas, es aceptable para conectar este pin al plano de tierra analógica del sistema. Sin embargo, el bus de alta capacidad en el pin DOUT podría resultar en ruido digital actual, lo que podría afectar su rendimiento.
11. CF: Salida lógica de calibración de frecuencia. La salida lógica CF da información de la potencia activa. Esta salida está destinada a ser utilizada a efectos operativos y de calibración. La frecuencia de salida a escala completa se puede ajustar por escrito a los registros CFDEN y CFNUM.
12. ZX: Salida de cruce por cero de forma de onda de voltaje (Canal 2). Esta salida se activa y desactiva con un alto lógico y un bajo lógico en el cruce por cero de la señal diferencial en el Canal 2.
13. SAG: Esta salida de lógica de drenaje abierto, se activa en bajo cuando no se detectan cruces por cero o un umbral de bajo voltaje (Canal 2) es atravesado por una duración determinada.
14. IRQ: Salida de solicitud de interrupción. Se trata de una salida activa baja de lógica de drenaje abierto. Las interrupciones enmascarables incluyen registro de energía activa de vuelco, el registro de energía activa a nivel medio, y la llegada de muestras de una nueva forma de onda.
15. CLKIN: Reloj maestro de ADC's y Procesamiento Digital de Señales. Un reloj externo puede ser proporcionado en esta entrada lógica. Alternativamente, un cristal resonante en paralelo AT se puede conectar a través de CLKIN y CLKOUT para proporcionar una fuente de reloj para el ADE7753. La frecuencia de reloj para la operación especificada es 3.579545 MHz. Condensadores cerámicos de carga de entre 22 pF y 33 pF deben ser usados con el circuito oscilador.
16. CLKOUT: Un cristal se puede conectar a través de este pin y CLKIN como se ha descrito al Pin 15 para proporcionar una fuente de reloj para el ADE7753. El pin CLKOUT puede soportar una carga CMOS cuando sea un reloj externo se suministra a CLKIN o un cristal que se esté usando.
17. CS: Selección de chip. Es parte de la interfaz de 4-hilos serie SPI⁸. Esta entrada de lógica activa baja permite el ADE7753 a compartir el bus serial con otros dispositivos.
18. SCLK: Entrada de reloj de serie para la interfaz serie síncrona. Todas las transferencias de datos en serie se sincronizan con este reloj. El SCLK tiene una entrada de Schmitt-trigger⁹ para su uso con una fuente de reloj que tiene un tiempo de flanco de transición lenta, por ejemplo, opto-aislador de salida.

⁸ Bus SPI, (del inglés Serial Peripheral Interface) es un estándar de comunicaciones, para transferencia de información.

⁹ Es un tipo especial de circuito comparador que tiene como objetivo prevenir el ruido.

19. DOUT: Salida de datos de la interfaz serie. Los datos se desplazan a este pin en el flanco ascendente de SCLK. Esta salida lógica se encuentra normalmente en un estado de alta impedancia a menos que este conduciendo datos al bus serie de datos.
20. DIN: Entrada de datos para la interfaz serie. Los datos son obtenidos en este pin en el flanco de bajada de SCLK.

DIAGRAMA DE BLOQUES

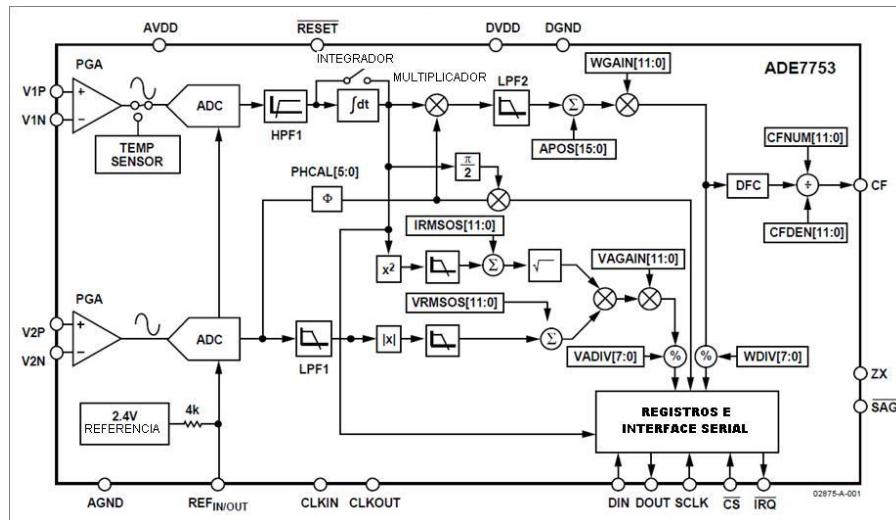


Figura 1.2 Diagrama de bloques.

CANAL 1

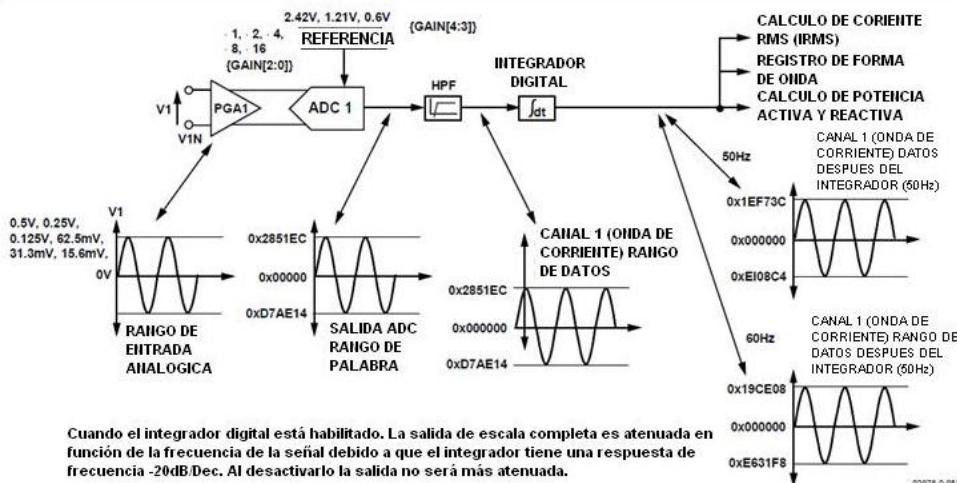


Figura 1.3 ADC y Procesamiento de señal de canal 1.

La figura 1.3 muestra el ADC y la cadena de procesamiento de señal para el canal 1. En el modo de muestreo de forma de onda, el ADC genera un complemento a dos con signo de palabras de datos de 24-bits en un máximo de 27.9 kSPS (CLKIN/128). Con la escala completa especificada de la señal de entrada analógica de 0.5 V (o 0.25 V o 0.125 V) el ADC produce un código de salida que es aproximadamente entre 0x2851EC (2.642.412 d) y 0xD7AE14 (-2.642.412 d) – véase la Figura 1.3.

CÁLCULO DE CORRIENTE RMS

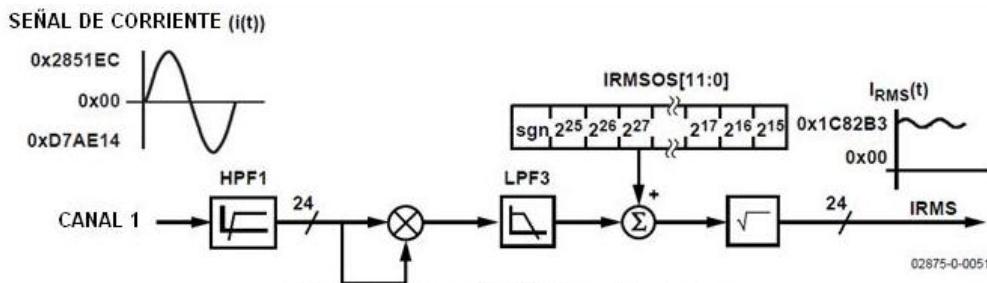


Figura 1.4 Procesamiento de señal rms del canal 1.

La raíz cuadrada media (rms) de una señal continua $V(t)$ se define como:

$$VRMS = V_{rms} = \sqrt{\frac{1}{T} \int_0^T V^2(t) dt} \quad (1.1)$$

Para señales muestreadas en el tiempo, el cálculo rms implica elevar al cuadrado la señal, tomando la media y obteniendo la raíz cuadrada:

$$VRMS = V_{rms} = \sqrt{\frac{1}{N} \sum_{i=1}^N V^2(i)} \quad (1.2)$$

El ADE7753 simultáneamente calcula los valores rms para el canal 1 y el canal 2 en diferentes registros. La figura 1.4 muestra el detalle de la cadena de procesamiento de señales para el cálculo rms en el canal 1. El valor rms del canal 1 se procesa a partir de las muestras obtenidas en el Canal 1 en el modo de muestreo de forma de onda. El valor rms del canal 1 se almacena en un registro de 24-bits sin signo (IRMS). Un LSB del registro rms del canal 1 es equivalente a un LSB de una muestra de la forma de onda del canal 1. La tasa de actualización de la medición del canal 1 rms es CLKIN / 4.

Con la plena escala especificada de la señal de entrada analógica de 0.5 V, el ADC produce un código de salida que es de aproximadamente $\pm 2,642,412$ d. El equivalente en valor eficaz de una señal de corriente alterna a gran escala son 1,868,467d (0x1C82B3). La medición de la corriente rms en el ADE7753 tiene una precisión de 0.5% para la señal de entrada. La Tabla 1 muestra el tiempo de establecimiento para la medición IRMS, que es el tiempo que toma el registro rms para reflejar el valor de la entrada para el canal actual.

Tabla 1.1 Tiempo de establecimiento para la medición IRMS

	95%	100%
Integrador apagado	219 ms	895 ms
Integrador encendido	78.5 ms	1340 ms

CANAL 2

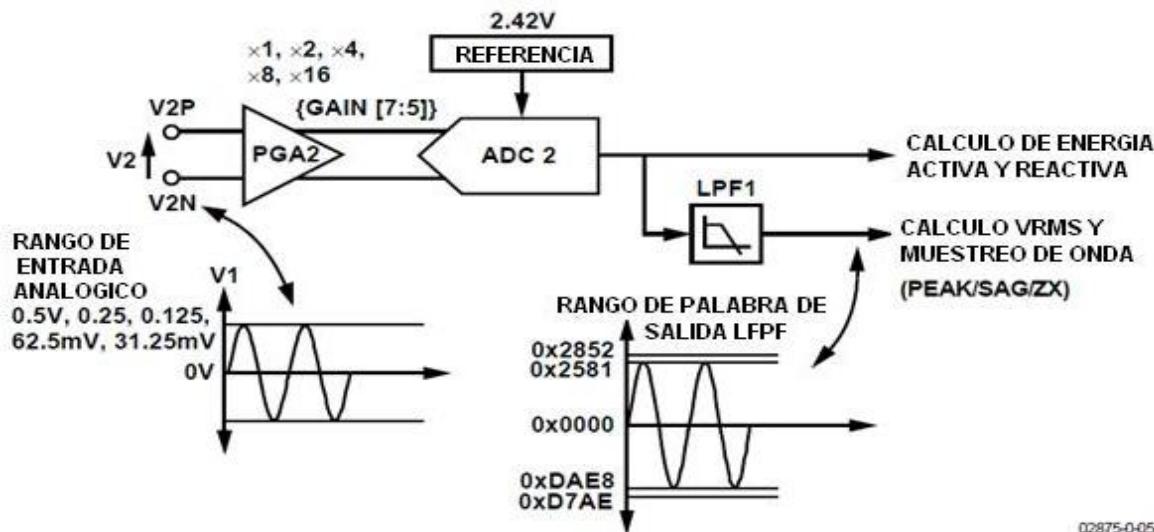


Figura 1.5 ADC y procesamiento de señal del canal 2.

CÁLCULO DE VOLTAJE RMS

La figura 1.6 muestra los detalles de la cadena de procesamiento de señales para la estimación de rms en el canal 2. Esta estimación rms del canal 2 se realiza en el ADE7753 utilizando el cálculo del valor absoluto de la media, como se muestra en la Figura 1.6. El valor rms del canal 2 se procesa a partir de las muestras utilizadas en el canal 2 en modo de muestreo de forma de onda. El valor rms es ligeramente atenuado por LPF1. El valor rms del canal 2 se almacena en un registro de 24 bits sin signo VRMS. La frecuencia de actualización del canal 2 de medida rms es CLKIN / 4.

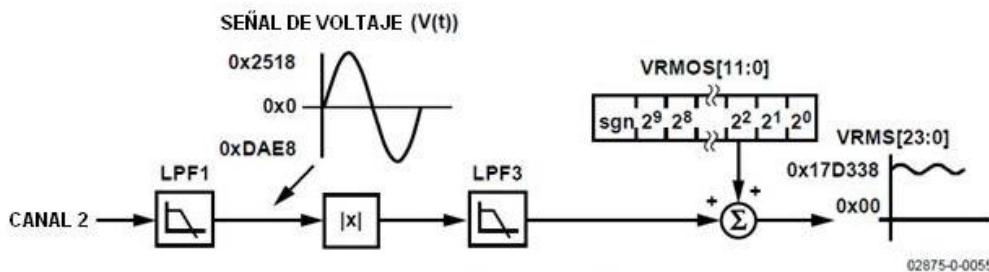


Figura 1.6 Procesamiento de señal rms del canal 2.

Con la escala completa especificada de la señal analógica de entrada de CA de 0.5 V, la salida de los LPF1 oscila entre 0x2518 y 0xDAE8 a 60 Hz. El valor eficaz equivalente de esta señal de escala completa de CA es de aproximadamente 1,561,400 (0x17D338) en el registro VRMS. La tensión de medición rms prevista en el ADE7753 tiene una precisión de $\pm 0.5\%$ para la señal de entrada entre escala completa y la vigésima parte de la escala completa. La Tabla 2 muestra el tiempo de establecimiento para la medición VRMS, que es el tiempo que toma para que el registro RMS refleje el valor de la entrada para el canal de tensión. La conversión del valor de registro a voltios debe hacer externamente en el microprocesador usando una relación voltios / LSB constante. Dado que el filtrado de paso bajo utilizado para calcular el valor rms es imperfecta, hay algo de ruido de onda del término 2ω presente en la medición rms. Para minimizar el efecto del ruido en la lectura, hay que sincronizar la lectura rms con los cruces por cero de la tensión de entrada.

Tabla 1.2 Tiempo de establecimiento para la medición VRMS.

95%	100%
220 ms	670 ms

CÁLCULO DE POTENCIA ACTIVA

La potencia se define como la tasa de flujo de energía desde la fuente a la carga. Se define como el producto de las formas de onda de voltaje y corriente. La forma de onda resultante se denomina la señal de potencia instantánea y es igual a la tasa de flujo de energía en cada instante de tiempo. La unidad de potencia es el watt o Joules/seg. La ecuación 5 da una expresión para la señal de potencia instantánea en un sistema de CA.

$$V(t) = \bar{V} \sin(\omega t) \quad (1.3)$$

$$i(t) = \bar{I} \sin(\omega t) \quad (1.4)$$

Donde:

\bar{V} es voltaje rms.

\bar{I} es corriente rms.

$$p(t) = v(t) \cdot i(t) \quad (1.5)$$

$$p(t) = VI - VI \cos 2\omega t \quad (1.6)$$

La potencia media en un número entero de ciclos de la línea (n) viene dada por la expresión en la Ecuación 6.

$$P = \frac{1}{nT} \int_0^{nT} p(t) dt = VI \quad (1.7)$$

Donde:

T es el periodo de un ciclo de línea.

P se refiere tanto a la potencia activa como real.

Tenga en cuenta que la potencia activa es igual a la componente continua de la señal de potencia instantánea $p(t)$ en la ecuación 4, es decir, VI . Esta es la relación utilizada para calcular la potencia activa en el ADE7753. La potencia de la señal instantánea $p(t)$ se genera al multiplicar las señales de corriente y voltaje. La componente continua de la señal de potencia instantánea es entonces extraído LPF2 (filtro de paso bajo) para obtener la información de la potencia activa. Este proceso se ilustra en la Figura 1.7.

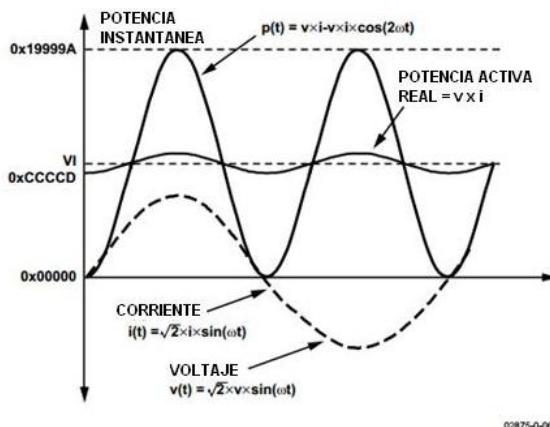


Figura 1.7 Cálculo de la potencia activa.

La señal de potencia activa tiene cierto rizado debido a la señal de potencia instantánea. Esta onda es sinusoidal y tiene una frecuencia igual al doble de la frecuencia de línea. Debido a que la onda es sinusoidal, es retirado cuando la señal de potencia activa es integrada para calcular la energía.

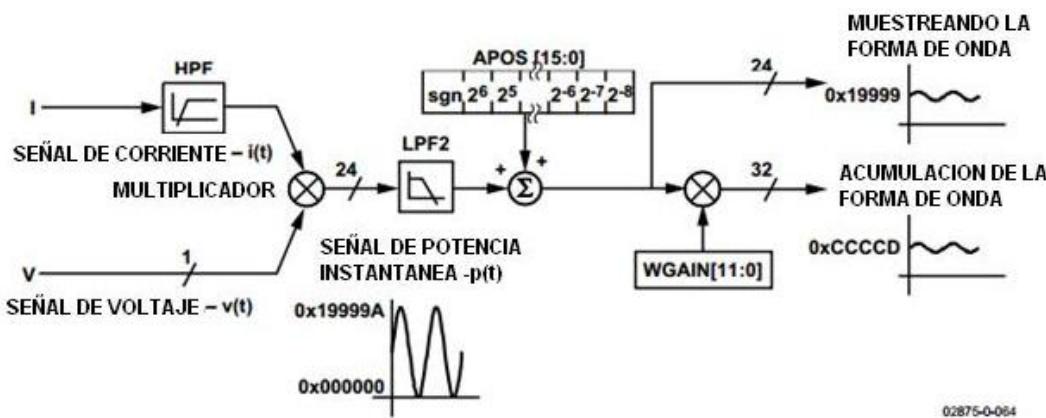


Figura 1.8 Procesamiento de la señal de la potencia activa.

La Figura 1.8 muestra la cadena de procesamiento de señales para el cálculo de la potencia activa en el ADE7753. Como se ha explicado, la potencia activa se calcula por un filtrado paso bajo la señal de potencia instantánea. Tenga en cuenta que cuando la lectura de las muestras de forma de onda de la salida de LPF2, la ganancia de la energía activa se puede ajustar usando el multiplicador y el registro de ganancia de watts (WGAIN [11:0]). La ganancia se ajusta escribiendo una palabra de 12 bits en complemento a dos al registro de ganancia de watts. La ecuación 7 muestra cómo el ajuste de la ganancia está relacionado con el contenido del registro de ganancia de watts:

$$\text{Salida WGAIN} = \text{Potencia Activa} \cdot 1 + \frac{WGAIN}{2^{12}} \quad (1.8)$$

CÁLCULO DE ENERGÍA

Como se dijo anteriormente, la potencia se define como la tasa de flujo de energía. Esta relación se puede expresar matemáticamente en la Ecuación 8.

$$P = \frac{dE}{dt} \quad (1.9)$$

Donde:

E es Energía.

P es Potencia.

A la inversa, la energía se da como la integral de la potencia.

$$E = P dt \quad (1.10)$$

El ADE7753 logra la integración de la señal de potencia activa acumulando de forma continua la señal de potencia activa en un registro de energía interno de 49 bits. El registro de energía activa (AENERGY [23:00]) representa los 24 bits superiores de este registro interno. Esta acumulación de tiempo discreto o suma es equivalente a la integración en tiempo continuo. La ecuación 10 expresa la relación.

$$E = \int p(t) dt = \lim_{t \rightarrow 0} \sum_{n=1}^{\infty} p(nT) T \quad (1.11)$$

Donde:

N es el número de muestras de tiempo discreto.

T es el periodo de muestreo.

El período de muestreo de tiempo discreto (T) para la acumulación registrado en el ADE7753 es $1.1\mu s$ (4/CLKIN). Así como el cálculo de la energía, esta integración elimina los componentes sinusoidales que pueden estar en la señal de potencia activa. La figura 1.9 muestra esta integración o acumulación de tiempo discreto. La señal de potencia activa en el registro de forma de onda se añade continuamente al registro de energía activa interna. Esta adición es una adición con signo, por lo tanto la energía negativa se resta el contenido de energía activa. La excepción a esto es cuando POAM se selecciona en el registro MODE[15:0]. En este caso, sólo la energía positiva contribuye a la acumulación de energía activa.

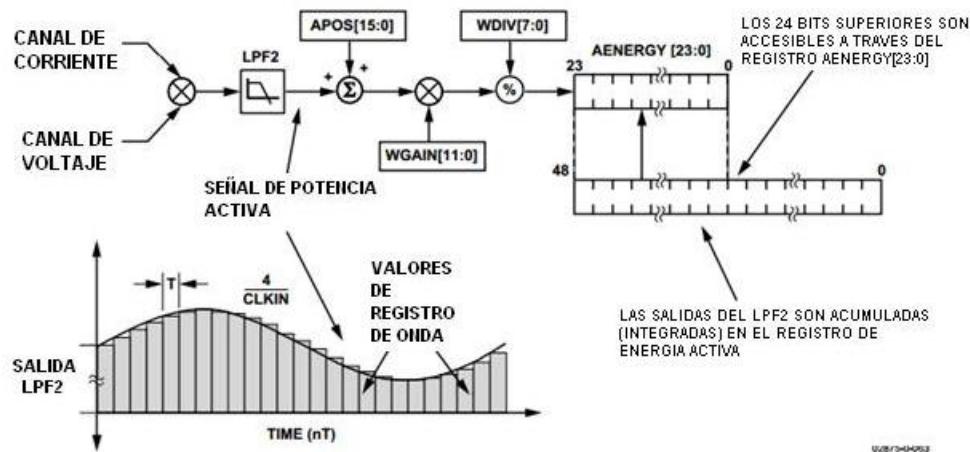


Figura 1.9 Cálculo de la energía activa del ADE7753.

La salida del multiplicador se divide por WDIV. Si el valor en el registro WDIV es igual a 0, entonces el registro interno de energía activa se divide por 1. WDIV es un registro de 8 bits sin signo. Despues de dividir por WDIV, la energía activa se acumula en un registro interno de 49 bits de acumulación de energía. La parte superior 24 bits de este registro son accesibles a través de una

lectura al registro de energía activa (AENERGY [23:0]). Una lectura en el registro RAENERGY devuelve el contenido del registro AENERGY y la parte superior de 24 bits del registro interno se han borrado. Como se muestra en la Figura 1.9, la señal de potencia activa se acumula en un registro interno 49 bits con signo. La señal de potencia activa se puede leer del registro de la forma de onda estableciendo MODE [14:13] = 0,0 y establecer el bit WSMP (bit 3) en el registro de habilitación de interrupción a 1.

CALIBRACIÓN DEL OFFSET PARA LA POTENCIA.

El ADE7753 también incorpora un registro para el offset de la potencia activa (APOS[15:0]). Este es un registro 16-bit signado complemento a dos que puede ser usado para remover el offset en los cálculos de la potencia activa. Un offset podría existir en el cálculo de la potencia activa debido a una interferencia entre los canales en el PCB o en el IC mismo.

La calibración del offset permite que el contenido del registro de la potencia activa sea mantenido a "0" cuando ninguna potencia es consumida.

CONVERSIÓN DE ENERGÍA A FRECUENCIA.

El ADE7753 también provee un convertidor de energía a frecuencia que sirve para propósitos de auto calibración. Después de la inicial calibración de fabricación, el fabricante o el consumidor final usualmente verifican la calibración de la medición de energía. Una manera conveniente de verificar esto es que el fabricante provea una salida de frecuencia, que es proporcional a la energía o a la potencia activa bajo carga estable. Esta salida de frecuencia puede proveer una simple, único alambre, ópticamente aislado para un equipo externo de calibración.

Un convertidor digital a frecuencia (DFC) es utilizado para general el pulso en la salida CF. El DFC genera un pulso cada vez que 1 LSB en el registro de energía activa es acumulado. Un pulso de salida es generado cuando $(CFDEN+1)/(CFNUM+1)$ números de pulsos son generados en la salida DFC. Bajo condiciones de carga constante la frecuencia de salida es proporcional a la energía activa.

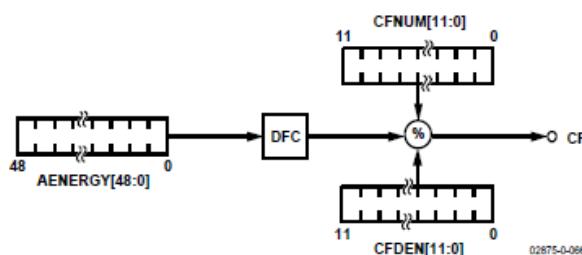


Figura 1.10 Conversión de energía a frecuencia.

MODO DE ACUMULACIÓN POSITIVA SOLAMENTE.

En este modo la acumulación de energía está hecha solo para potencia positiva, ignorando cualquier ocurrencia de potencia negativa sobre o debajo del umbral de no carga, como se muestra en la figura 1.11. El pulso CF también refleja este método de acumulación. El ADE7753 es puesto en POAM seleccionando el MSB del registro de modo MODE [15]. El ajuste por defecto es apagado. Las transiciones en las que la potencia fluye, yendo de negativo a positivo o de positivo a negativo, PPOS y PNG muestran cual transición ha ocurrido.

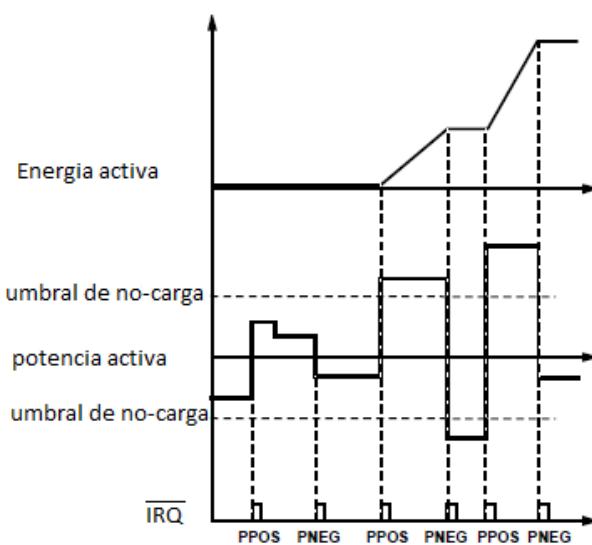


Figura 1.11 Acumulación de energía en POAM. (Estado de las interrupciones)

UMBRAL DE NO-CARGA.

El ADE7753 incluye una utilidad de umbral de no carga en la energía activa que elimina cualquier debilidad en la medición, esto se logra no acumulando energía si la salida del multiplicador está por debajo del umbral de no carga. Este umbral es el 0.001% de la salida a escala completa en frecuencia del multiplicador.

CALCULO DE LA POTENCIA REACTIVA.

La potencia reactiva está definida como el producto de la corriente por el voltaje cuando una de estas señales está desfasada 90° . La resultante forma de onda es llamada potencia reactiva instantánea.

El promedio de la potencia reactiva sobre un número entero de ciclos de línea está dado por la ecuación siguiente

$$RP = \frac{1}{nT} \int_0^{nT} Rp(t) dt = V I \sin(\theta) \quad (1.12)$$

Donde:

θ es la diferencia de fase entre el voltaje y la corriente.

T es el periodo del ciclo de la línea.

Notar que la potencia reactiva es igual a la componente de DC de la potencia reactiva instantánea, esta última se genera multiplicando el canal 1 y el canal 2. En este caso la fase del canal 1 está desfasada 90° . La componente de DC es entonces extraída por un filtro paso bajo con el objetivo de obtener la información de la potencia reactiva. La figura 1.12 muestra el proceso de cálculo de la potencia reactiva.

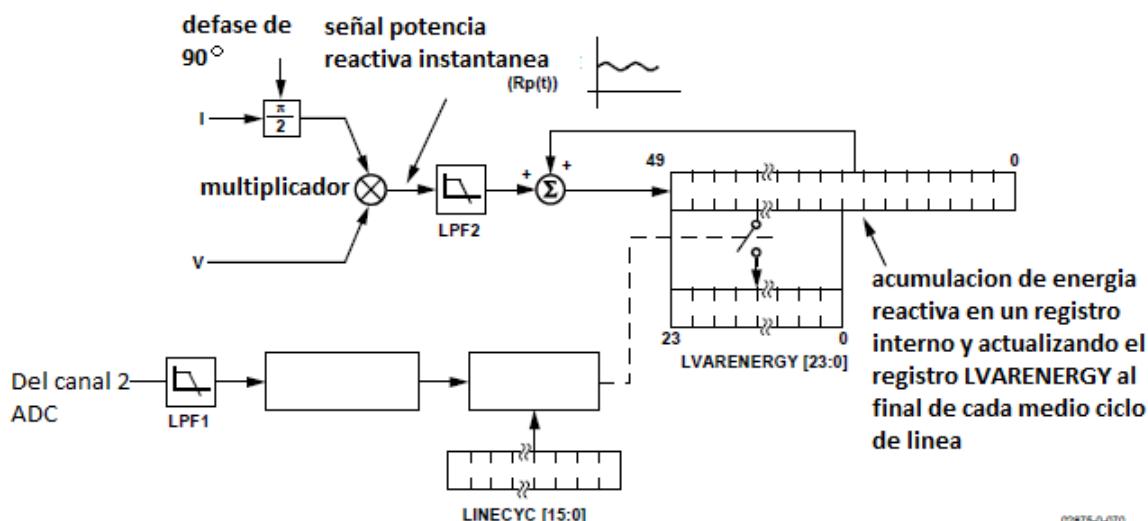


Figura 1.12 Procesado de la señal de potencia reactiva.

SIGNO DEL CÁLCULO DE LA POTENCIA REACTIVA.

Notar que el promedio de la potencia reactiva es un cálculo signado. El filtro de cambio de fase tiene -90° de cambio en la fase cuando el integrador está habilitado, y $+90^\circ$ cuando el integrador está deshabilitado, la tabla 3 resume las relaciones entre las diferencias de fase entre el voltaje y la corriente y el signo del resultante cálculo para la VAR.

Tabla 1.3. Signo del cálculo de la potencia reactiva.

Angulo	Integrador	Signo
Entre 0° a 90°	Apagado	Positivo
Entre -90° a 0°	Apagado	Negativo

Angulo	Integrador	Signo
Entre 0° a 90°	Encendido	Positivo
Entre -90° a 0°	Encendido	Negativo

FRECUENCIA CLKIN.

En la hoja de datos las características del ADE7753 son mostradas cuando el CLKIN es igual a 3.5795 MHZ. Sin embargo el IC está diseñado para tener la misma precisión entre un rango especificado de frecuencias. Si el CLKIN no es 3.5795 MHZ varios temporizadores y filtros son necesarios para redefinir con la nueva frecuencia. Por ejemplo las frecuencias de corte de todos los filtros LPF1, LPF2, HPF1, cambian en proporción al cambio en CLKIN de acuerdo a la siguiente ecuación.

$$\text{nueva frecuencia} = \text{frecuencia original} \frac{\text{frecuencia CLKIN}}{3.5795\text{MHz}} \quad (1.13)$$

El cambio en CLKIN no afecta las características de tiempo de la interface serial porque la trasferencia de datos esta sincronizada con la señal de reloj serial SCLK. La tabla 4 muestra los cambios que hay que tener en cuenta debido al cambio en CLKIN.

Tabla 1.4. Parámetros de dependencia de CLKIN.

Parámetros	Dependencia de CLKIN
Frecuencia de Nyquist canal 1 y 2	CLKIN/8
PHCAL resolución (segundos por LSB)	4/CLKIN
Actualización de registros energía activa Hz	CLKIN/4
Máximo periodo ZXTOUT	524,288/CLKIN
WAVSEL 1,0= 00	CLKIN/128
01	CLKIN/256
10	CLKIN/512
11	CLKIN/1024

ADE7753 INTERFACE SERIAL.

Todas las funcionalidades del ADE7753 son accesibles a través de muchos registros que ya están dentro del chip, ver figura 1.13. Los contenidos de estos registros pueden ser actualizados o leídos usando la interface serial del chip. Después de encender, mandar a RESET un pulso bajo o en el borde de bajada de *cs*. El ADE7753 es puesto en modo de comunicación; en este modo el IC espera que se escriba en su registro de comunicación. El dato escrito en el registro de comunicación determina si la siguiente operación es una lectura o una escritura y además indica la dirección del registro a acceder. Entonces todas las transferencias de datos no importando si es lectura o escritura deben iniciar con una escritura en el registro de comunicaciones.

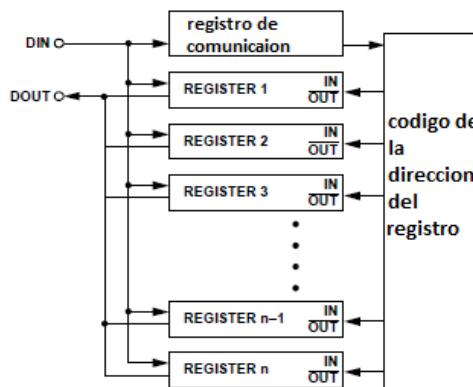


Figura 1.13 Direccionando los registros del ADE7753 vía registro de comunicación.

El registro de comunicación es de un byte de ancho. El MSB determina si la siguiente transferencia es una lectura o escritura. Los seis LSB contienen la dirección del registro al que se accederá. La figura 1.14 y 1.15 muestran las secuencias de transferencias de datos para una lectura y escritura respectivamente. Una transferencia de datos está completa cuando el LSB de un registro del ADE7753 se ha escrito o recibido desde o para el ADE7753.

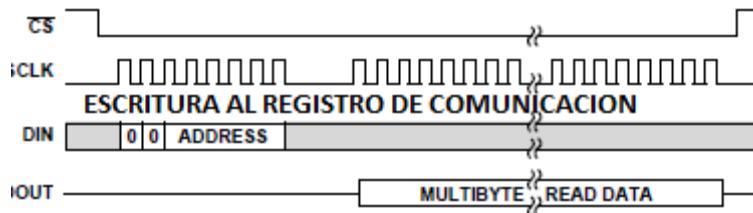


Figura 1.14 Leyendo datos desde el ADE7753 a través de interface serial.



Figura 1.15 Escribiendo datos desde el ADE7753 a través de interface serial.

La interface serial del ADE7753 está hecha de cuatro señales SCLK, DIN, DOOUT, y *cs*. El reloj serial para una transferencia de datos es aplicado en la entrada SCLK. Esta entrada lógica tiene un Schmitt-trigger como estructura de entrada que permite una suave elevación (y caída) del flanco del reloj. Todas las operaciones de transferencia de datos son sincronizadas con el reloj serial. Los datos entran al ADE7753 a través de DIN en el flanco de bajada del SCLK. Los datos salen del ADE7753 en el flanco de subida del SCLK. La entrada lógica *cs* es la selectora de chip. Esta entrada es usada cuando múltiples dispositivos utilizan el bus serial. Un flanco de bajada de *cs* resetea la interface serial y pone al ADE7753 en modo de comunicación.

ADE7753 OPERACIÓN ESCRITURA SERIAL.

La secuencia de escritura serial toma lugar como sigue. Con el ADE7753 en modo de comunicación una escritura en el registro de comunicación primero es necesaria. El MSB de este byte a transferir es un 1, indicando que la operación de transferencia es una escritura. El LSB de este byte contiene la dirección del registro al que se le escribirá. El ADE7753 empieza el movimiento al registro de datos en el siguiente flanco de bajada del SCLK. Todos los bits restantes del registro de datos son enviados en el flanco de bajada de los subsecuentes pulsos del SCLK, ver figura 1.16. Durante una operación de escritura de datos hacia el ADE7753 los datos son transferidos hacia los registros un byte a la vez. Después que un byte es transferido al puerto serial hay un tiempo finito antes de que se transfiera a los registros del ADE7753. Sin embargo otro byte puede ser enviado al puerto serial mientras el byte anterior es enviado a los registros, esta segunda transferencia de byte no debería de finalizar hasta $4\mu s$ después que termine la trasferencia del byte anterior. Esta especificación está ilustrada en la figura 1.16 mediante el tiempo t_6 . Si una escritura es abortada durante una transferencia de byte cs en alto, entonces ese byte no puede ser escrito al registro destino.

Los registros de destino pueden ser de hasta tres bytes de ancho. Por lo tanto el primer byte enviado al puerto serial se convertirá en el MSB de registro destino, si por ejemplo el registro destino tiene 12 bits de ancho, 2 transferencias de bytes tienen lugar. Los 4 MSB del primer byte enviado serán descartados y los 4 bits LSB del primer byte enviado serán los 4 MSB de la palabra de 12 bits del registro destino.

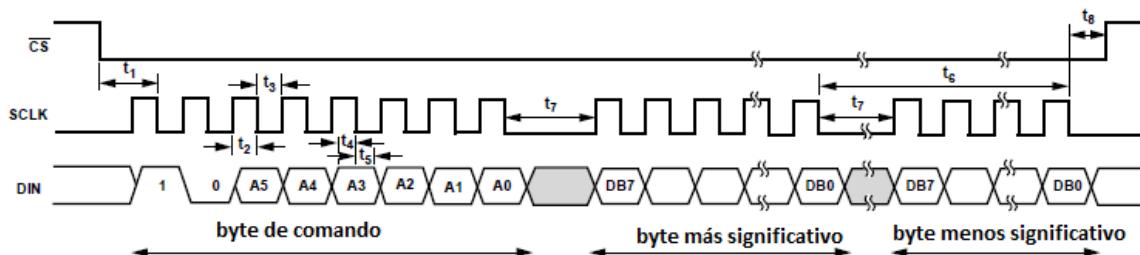


Figura 1.16 Escritura con interface serial.

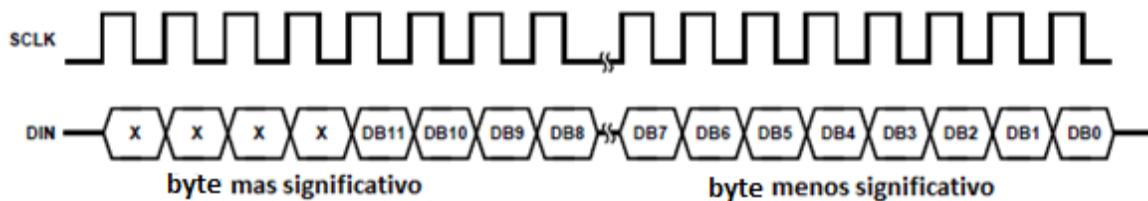


Figura 1.17 Escritura serial en un registro de 12 bits.

ADE7753 OPERACIÓN DE LECTURA SERIAL

Durante la operación de lectura de datos del ADE7753, los datos son extraídos a través del pin DOUT que es una salida lógica, esto sucede en los flancos de elevación del SCLK. Como es el caso con la escritura, una lectura de datos debe ser precedida por una escritura en el registro de comunicaciones.

Con el ADE7753 en modo de comunicación una escritura de 8 bits es lo primero que ocurre, el MSB de este byte debe ser un cero para indicarle al IC que se efectuara una lectura, el LSB contiene la dirección del registro a leer. El ADE7753 empieza a extraer los datos en el siguiente flanco de subida de la señal SCLK véase la figura 1.18. En este punto DOUT, la salida lógica deja el estado de alta impedancia, y empieza a manejar el bus de datos, todos los bits restantes son sacados uno por uno por cada flanco de subida del SCLK.

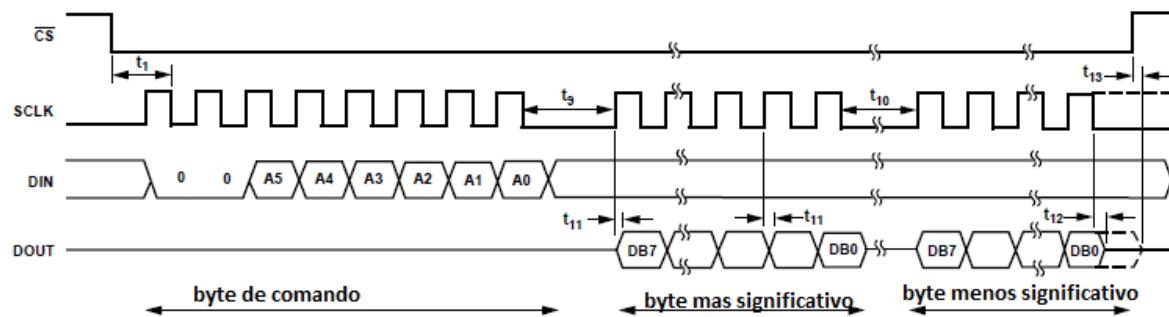


Figura 1.18 Tiempos de lectura de la interface serial.

REGISTROS DEL ADE7753

Tabla 1.5. Resumen de los registros por dirección.

Dirección	Nombre	E/L	No Bits	Por defecto	Tipo	Descripción
0X01	WAVEFOR M	L	24	0X0	S	Este es un registro de solo lectura contiene la forma de onda del canal 1,2, o de la potencia activa. La fuente de los datos es seleccionada con el bit 14,13 del registro modo.
0X02	AENERGY	L	24	0X0	S	La potencia activa es acumulada (integrada)
0X03	RENERGY	L	24	0X0	S	Lo mismo que el anterior excepto que el registro se vuelve 0 una vez efectuada la lectura.
0X04	LAENERGY	L	24	0X0	S	Acumulación de energía activa de la línea. La potencia activa instantánea
0X05	VAENERGY	L	24	0X0	U	La potencia aparente es acumulada

Dirección	Nombre	E/L	No Bits	Por defecto	Tipo	Descripción
						sobre el tiempo.
0X06	RVAENERGY	L	24	0X0	U	Igual que el anterior solo que el registro se pone en cero cada vez que hay una lectura.
0X07	LVAENERGY	L	24	0X0	U	La potencia real instantánea es acumulada en este registro de solo lectura
0X08	LVARENERGY	L	24	0X0	S	La potencia instantánea reactiva es acumulada en este registro.
0X09	MODE	E/L	16	0X000C	U	A través de este registro muchas funcionalidades del IC son revisadas, ritmo de muestreo de las señales, habilitación de filtro etc.
0X0A	IRQEN	E/L	16	0X40	U	Las interrupciones pueden ser deshabilitadas poniendo el correspondiente bit en lógico 0.
0X0B	STATUS	L	16	0X0	U	Registro del estatus de las interrupciones contiene información referente a la fuente de las interrupciones en el ADE7753.
0X0C	RSTSTATUS	L	16	0X0	U	Igual que el anterior excepto que después de la lectura todas las banderas se ponen en lógico 0.
0X0D	CH1OS	E/L	8	0X00	S'	Ajuste del offset del canal 1.
0X0E	CH2OS	E/L	8	0X00	S'	Ajuste del offset del canal 2.
0x0F	GAIN	E/L	8	0X0	U	Ganancia PGA, es utilizado para ajustar la ganancia PGA en los canales 1 y 2.
0X10	PHCAL	E/L	6	0X0D	S	Registro de calibración de fase.
0X11	APOS	E/L	16	0X0	S	Corrección de offset para la potencia activa.
0X12	WGAIN	E/L	12	0X0	S	Ajuste de ganancia de potencia. El cálculo de la potencia activa puede ser calibrado escribiendo en este registro.
0X13	WDIV	E/L	8	0X0	U	El registro interno de energía activa es dividido por el valor de este registro antes de ser guardado en el registro AENERGY.
0X14	CFNUM	E/L	12	0X3F	U	Registro numerador divisor de frecuencia.
0X15	CFDEN	E/L	12	0X3F	U	Registro denominador divisor de frecuencia.

Dirección	Nombre	E/L	No Bits	Por defecto	Tipo	Descripción
0X16	IRMS	L	24	0X0	U	Valor rms del canal 1.
0X17	VRMS	L	24	0X0	U	Valor rms del canal 2.
0X18	IRMSOS	E/L	12	0X0	S	Registro para la corrección del offset del canal 1.
0X19	VRMSOS	E/L	12	0X0	S	Registro para la corrección del offset del canal 2.
0X1A	VAGAIN	E/L	12	0X0	S	La potencia aparente puede ser calibrada escribiendo en este registro.
0X1B	VADIV	E/L	8	0X0	U	El registro interno de energía aparente es dividido por el valor en este registro antes de almacenarse en VAENERGY.
0X1C	LINECYC	E/L	16	0xFFFF	U	Este registro es utilizado durante la acumulación de energía de un ciclo de línea.
0X1D	ZXTOUT	E/L	12	0xFFFF	U	Si ningún cruce por cero es detectado en un periodo determinado por el valor en este registro se activa una interrupción.
0x1E	SAGCYC	E/L	8	0xFF	U	Este registro de 8 bits determina el número consecutivo de ciclos de linea en el canal 2 por debajo de SAGLVL antes que la salida SAG sea activada.
0X1F	SAGLVL	E/L	8	0X0	U	Este registro determina a que nivel de la señal pico en el canal 2 el pin SAG se vuelve activo.
0x20	IPKLV	E/L	8	0xFF	U	Este registro pone el nivel de umbral de detección de la corriente pico.
0X21	VPKLVL	E/L	8	0xFF	U	Este registro pone el nivel de umbral de detección del voltaje pico.
0X22	IPEAK	L	24	0X0	U	Guarda el máximo valor de entrada del canal de corriente desde el último valor guardado en este registro.
0X23	RSTIPEAK	L	24	0X0	U	Igual que el canal 1 excepto que el contenido del registro es puesto a cero después de la lectura.
0X24	VPEAK	L	24	0X0	U	Guarda el máximo valor de entrada del canal de voltaje desde el último valor guardado.
0X25	RSTVPEAK	L	24	0X0	U	Igual que el anterior pero pone a cero el registro después de la

Dirección	Nombre	E/L	No Bits	Por defecto	Tipo	Descripción
						lectura.
0X26	TEMP	L	8	0X0	S	Registro de temperatura.
0X27	PERIOD	L	16	0X0	U	Periodo del canal 2.
0X28- 0X3C	Reservada.					
0X3D	TMODE	E/L	8	-	U	Registro de modo prueba.
0X3E	CHKSUM	L	6	0X0	U	Revisa la suma.
0X3F	DIEREV	L	8	-	U	Registro de revisión.

S=signado por complemento a dos, U=no signado, S'=signado por el método de magnitud.

REGISTRO DE MODO.

Las funcionalidades del ADE7753 se configuran al escribir en el registro de modo.

La tabla 6 describe las funcionalidades de cada bit en el registro.

Tabla 1. 6. Funcionalidades de cada bit en el registro.

Locación De bit	Nombre de Bit	Valor por Defecto	Descripción
0	DISHPF	0	HPF filtro paso alto en canal 1 es deshabilitado cuando este bit es igual a 1.
1	DISLPF2	0	LPF filtro paso bajo después de multiplicador esta deshabilitado cuando este bit está en 1.
2	DISCF	1	La salida de frecuencia CF esta deshabilitada cuando este bit está en 1.
3	DISSAG	1	Detección del SAG en el voltaje de línea.
4	ASUSPEND	0	Apaga los convertidores digital analógico.
5	TEMPSEL	0	La conversión de temperatura inicia cuando este bit está en 1.
6	SWRST	0	RESET del chip con software.
7	CYCMODE	0	Poniendo el bit en 1, coloca al chip en modo de acumulación de energía en ciclo de línea.
8	DISCH1	0	Las entradas del ADC 1 están cortocircuitadas interiamente
9	DISCH2	0	Las entradas del ADC 2 están cortocircuitadas interiamente
10	SWAP	0	Poniendo este bit en 1, las entradas analógicas V2P y V2N están conectadas al ADC 1, y las entradas analógicas V1P y V1N están conectadas al ADC 2.
12,11		00	Estos bits son usados para seleccionar el tiempo de actualización del registro WAVEFORM. DTRT1 DTRT2 tasa de actualización 0 0 27.9ksps (CLKIN/128) 0 1 14 kSPS (CLKIN/256)

Locación De bit	Nombre de Bit	Valor por Defecto	Descripción		
			1	0	7 kSPS (CLKIN/512)
			1	1	3.5 kSPS (CLKIN/1024)
			Estos bits son usados para seleccionar la fuente de los datos muestreados que se guardan en el WAVEFORM.		
			WAVSEL1,0	Longitud fuente	
14, 13	WAVSEL1,0	00	0	0	señal de potencia activa
			0	1	reservado
			1	0	canal 1
			1	1	canal 2
15	POAM	0	Escribiendo un 1 lógico permite la acumulación de potencia activa solo positiva.		

CAPITULO 2: XBEE 802.15.4 [SERIE 1]

MÓDULO XBEE

El módulo XBee fue diseñado para cumplir con los estándares IEEE 802.15.4 y cubrir las necesidades de bajo costo y bajo consumo de energía de redes de sensores inalámbricos. El módulo requiere un mínimo de energía y proporciona la entrega fiable de datos entre dispositivos. El módulo funciona en la banda ISM¹⁰ de 2,4 GHz de frecuencia.



Figura 2.1. Módulo XBee Serie 1

CARACTERÍSTICAS PRINCIPALES

INTEGRIDAD DE DATOS EN UN LARGO ALCANCE:

- Interior / urbano: hasta 100 '(30 m)
- Exterior con línea vista: hasta 300 '(100 m)
- Potencia de transmisión: 1 mW (0 dBm)
- Sensibilidad del receptor: -92 dBm

REDES DE TRABAJO AVANZADAS Y SEGURAS:

Los dispositivos pueden ser utilizados como Fuente o Destino con direccionamiento Unicast¹¹ y Broadcast¹² en comunicación punto a punto, punto a multipunto y par a par, también son soportadas topologías con funciones de Coordinador o Fin para las operaciones de cada dispositivo.

Disponen de Reintentos y Reconocimientos de secuencias directas de espectro disperso DSSS¹³. Cada canal de secuencia directa tiene más de 65.000 direcciones de red única disponible esto

¹⁰ Banda ISM (Industrial, Scientific & Medical) es una banda civil utilizada también por los protocolos bluetooth y WiFi.

¹¹ Unicast la comunicación es de un punto a otro

¹² Broadcast la comunicación es entre un nodo y todos

¹³ DSSS (Direct Sequence Spread Spectrum) Secuencia directa de espectro disperso.

implica que se pueden usar más dispositivos de los que podría necesitar en alguna aplicación extensa.

BAJA POTENCIA:

- ✚ Corriente TX: 45 mA (@ 3.3 V)
- ✚ Corriente RX: 50 mA (@ 3.3 V)
- ✚ Corriente de apagado: <10 µA

FACIL DE USAR:

- ✚ No necesita configuración previa para comunicarse, al sacarse de la caja.
- ✚ Programa gratis de prueba y configuración X-CTU.
- ✚ Modos de comando AT y API, para configurar los parámetros de seguridad y comunicación del módulo.
- ✚ Larga lista de comandos.
- ✚ Factor de forma pequeño.
- ✚ Soporte experto de RF ilimitado y gratuito.

ESPECIFICACIONES

Tabla 2.1 Especificaciones técnicas de módulo XBee.

ESPECIFICACIÓN	MÓDULO XBEE
RENDIMIENTO	
Alcance interior/urbano	Arriba de 100 ft. (30 m)
Alcance en exterior con línea vista	Arriba de 300 ft. (100 m)
Potencia de Transmisión (Selectable)	1mW (0 dBm)
Velocidad de datos en comunicación de RF	250,000 bps
Velocidad de datos en comunicación serial(Selectable)	1200 - 115200 bps(también soporta velocidad fuera de standard)
Sensibilidad del receptor	-92 dBm (1% error de paquetes)
REQUERIMIENTOS DE POTENCIA	
Voltaje de alimentación	2.8 – 3.4 V
Corriente de transmission	45mA (@ 3.3 V)
Corriente Idle/Recepción	50mA (@ 3.3 V)
Corriente de apagado	< 10 µA
GENERAL	
Frecuencia de operación	ISM 2.4 GHz
Dimensiones	0.960" x 1.087" (2.438cm x 2.761cm)
Temperatura de operación	-40 a 85º C (industrial)
Opciones de antenas	Whip integrada, Chip o conector UFL
REDES Y SEGURIDAD	
Topologías soportadas	Punto a punto, punto a multipunto y par a

	par
Número de canales(Selectable)	16 canales de secuencia directa
Opciones de direccionamiento	PAN ID, direcciones y canales
APROVACIONES DE AGENCIA	
United States (FCC Part 15.247)	OUR-XBEE
Industry Canada (IC)	4214A XBEE
Europe (CE)	ETSI
Japan	n/a

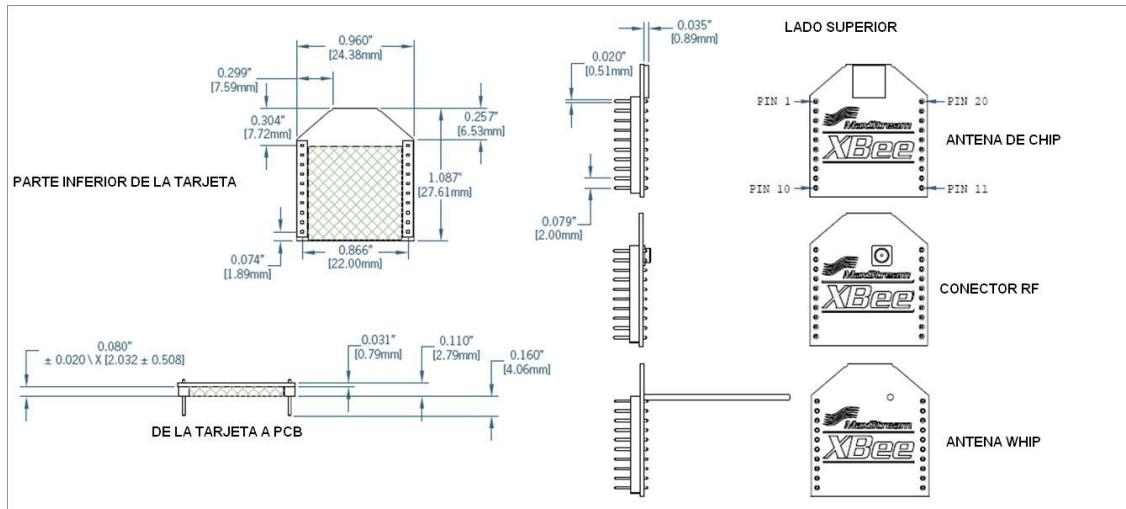


Figura 2.2 Especificaciones mecánicas del módulo XBee.

PINES DE SEÑAL

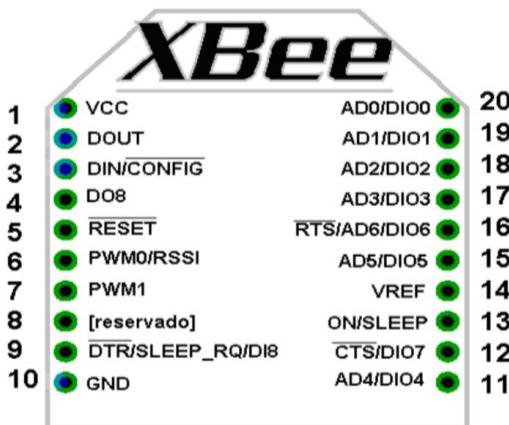


Figura 2.3 Vista superiores y distribución de pines del módulo XBee.

Tabla 2.2 Descripción de los pines del módulo XBee.

PIN #	NOMBRE	TIPO	DESCRIPCIÓN
1	VCC	-	Suministro de potencia
2	DOUT	Salida	Salida de datos UART

PIN #	NOMBRE	TIPO	DESCRIPCIÓN
3	DIN / <i>CONFIG</i>	Entrada	Entrada de datos UART
4	DO8	Salida	Salida digital 8
5	<i>RESET</i>	Entrada	Restablece el módulo(el pulso debe durar al menos 200 ns)
6	PWM0 / RSSI	Salida	Salida.PWM0 / Indicador de la señal RX
7	PWM1	Salida	salida PWM1
8	[RESERVADO]	-	Sin conexión
9	<i>DTR</i> /SLEEP_RQ/DI8	Entrada	Control del modo Sleep o entrada digital 8
10	GND	-	Tierra
11	AD4 / DIO4	Cualquiera I/O	Entrada analógica 4 o I/O digital 4
12	<i>CTS</i> / DIO7	Cualquiera I/O	Control de flujo de envío o I/O digital 7
13	ON / <i>SLEEP</i>	Salida	Indicador del estado del módulo
14	VERF	Entrada	Voltaje de referencia para entradas A/D
15	Asociar / AD5 / DIO6	Cualquiera I/O	Indicador asociado, Entrada analógica 6 o I/O digital 6
16	<i>RTS</i> / AD6 / DIO6	Cualquiera I/O	Control de flujo de petición de envío, entrada analógica 6 o I/O digital 6
17	AD3 / DIO3	Cualquiera I/O	Entrada analógica 3 o I/O digital 3
18	AD2 / DIO2	Cualquiera I/O	Entrada analógica 2 o I/O digital 2
19	AD1 / DIO1	Cualquiera I/O	Entrada analógica 1 o I/O digital 1
20	AD0 / DIO0	Cualquiera I/O	Entrada analógica 0 o I/O digital 0

MODOS DE OPERACIÓN

MODO DE RECEPCIÓN/TRANSMISIÓN

Se trabaja en este modo cuando se envía información serial al buffer del pin 3 (UART¹⁴ Data in) para ser transmitida (modo Transmit) o cuando al módulo llega algún paquete de RF a través de la antena (modo receive).

La información que se transmite puede ser directa o indirecta. En el modo directo la información se transmite inmediatamente a la dirección de destino, mientras que en el modo indirecto la información es retenida durante un periodo de tiempo y es enviada hasta que la dirección de destino la solicita.

¹⁴ UART “Universal Asynchronous Receiver-Transmitter” (*Transmisor-Receptor Asíncrono Universal*). Éste controla los puertos y dispositivos serie. Se encuentra integrado en la placa base o en la tarjeta adaptadora del dispositivo. Existe un dispositivo electrónico encargado de generar la UART en cada puerto serie.

Es posible enviar información por dos modos: Broadcast y Unicast.

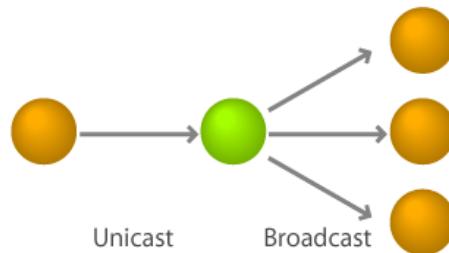


Figura 2.4 Representación gráfica de comunicación unicast y broadcast.

En la comunicación unicast la comunicación es de un punto a otro, y es el único modo que permite respuesta de quien recibe el paquete RF, es decir, quien recibe debe enviar un ACK¹⁵ (paquete llamado así, y que indica que recibió el paquete, el usuario no puede verlo, es interno de los módulos) a la dirección de origen. Quien envió el paquete, espera recibir un ACK, en caso de que no le llegue, reenviará el paquete hasta 3 veces o hasta que reciba el ACK. En el modo Broadcast la comunicación es entre un nodo y todos los demás nodos de la red. En este modo, no hay confirmación por ACK.

MODO “SLEEP” DE BAJO CONSUMO

El “sleep mode” hace posible que el módulo RF entre en un modo de bajo consumo de energía cuando no se encuentra en uso.

Para poder entrar en modo de sueño, se debe cumplir una de las siguientes condiciones:

- Sleep_RQ(pin 9) está en alto y el módulo está en pin sleep mode (SM= 1,2 o 5)
- El módulo está en reposo (no hay transmisión ni recepción de datos) por la cantidad de tiempo definido por ST (Time before Sleep). [ST sólo está activado cuando SM=4,5]

La configuración de los ciclos de sueño se realiza principalmente con el comando SM. Por defecto, los modos de sueños están deshabilitados (SM=0), permaneciendo el módulo en estado de reposo/recepción. En este estado el módulo está siempre preparado para responder a un comando, ya sea, por el puerto serial o la interfaz RF.

MODO DE SUEÑO CONTROLADO POR PIN

Pin de hibernación: Este modo minimiza el consumo de energía cuando el módulo se encuentra en reposo. Este modo se habilita cuando Sleep_RQ(pin 9) está en alto, el módulo terminará cualquier transmisión, recepción o procedimientos de asociación y entrará en modo de reposo y

¹⁵ ACK, viene de Acknowledge que en español significa “reconocimiento”.

luego en modo de sueño. En este estado el módulo no responderá a comandos entrantes, ya sea, desde la interfaz serial como RF.

Cuando se baja el estado lógico de Sleep_RQ (pin 9) el módulo saldrá del modo de sueño y estará listo para recibir o enviar datos.

Pin doze: Este modo funciona de la misma forma que el modo Pin de Hibernación, sin embargo, Pin Doze presenta un tiempo de activación menor y mayor consumo de energía.

Para despertar un módulo operando en modo Pin Doze, se debe bajar Sleep_RQ (pin9) y éste comenzará a transmitir o recibir información cuando la línea CTS este en nivel lógico bajo.

MODO DE SUEÑO CÍCLICO

Sueño cíclico remoto [SM=4]: El modo de sueño cíclico remoto permite que el módulo revise la data por la interfaz RF periódicamente. Cuando el parámetro SM es configurado a 4, el módulo se configura para efectuar ciclos de sueño, luego, despierta una vez por ciclo para revisar si existen datos en el coordinador de sueño de la red (SM = 0, CE = 1). El módulo remoto, envía esta solicitud al coordinador a intervalos de tiempo determinados por el parámetro ST (Periodo de dormido). El coordinador transmitirá los datos que se puedan encontrar en su buffer de salida al módulo remoto una vez recibida la solicitud de datos.

En el caso que el coordinador no tenga datos para ser enviados al módulo que realiza la solicitud, el coordinador no transmitirá y el módulo remoto retornará a su estado de sueño. Si existen datos para ser enviados al módulo solicitante, el coordinador se quedará despierto y transmitiendo hasta el Timer ST (Tiempo antes de dormir) se complete.

Sueño cíclico remoto y pin para despertar [SM=5]: Este modo se utiliza para despertar un módulo remoto, ya sea por la interfaz RF o por poner en estado bajo el pin Sleep_RQ utilizado para comunicación orientada a eventos. El sueño cíclico funciona de la misma forma que el modo de sueño cíclico remoto con la funcionalidad extra de poder despertar el módulo utilizando un pin.

Cualquier actividad limpiará la cuenta de ST(Tiempo antes de dormir), de esta manera el módulo sólo volverá a dormir luego que no exista alguna actividad durante el periodo de ST. Si el módulo despierta por cambios en el pin Sleep_RQ, los posteriores cambios serán ignorados durante su funcionamiento.

Coordinador de sueño: Este modo configura al módulo para funcionar como coordinador de sueño. El coordinador acepta mensajes a un módulo específico con direcciones de 16 o 64 bit y los mantiene en su buffer interno hasta que los módulos remotos despiertan y solicitan datos al coordinador. El parámetro SP del coordinador debe ser seteado con el mismo valor que los módulos remotos, para que se pueda producir la comunicación entre los ciclos de sueño.

MODO DE COMANDO

Este modo permite ingresar comandos AT al módulo XBee, para configurar, ajustar o modificar parámetros. Permite ajustar parámetros como la dirección propia o la de destino, así como su modo de operación entre otras cosas. **Para poder ingresar los comandos AT es necesario utilizar el programa X-CTU, un terminal serial o algún micro controlador que maneje UART y tenga los comandos guardados en memoria o los adquiera de alguna otra forma.**

Para ingresar a este modo se debe esperar un tiempo dado por el **comando GT** (Guard Time, por defecto ATGT=0x3E84 que equivalen a 1000ms) luego ingresar +++ y luego esperar otro tiempo GT. Como respuesta el módulo entregará un OK. El módulo XBee viene por defecto con una velocidad de 9600bps. En caso de no poder ingresar al modo de comandos, es posible que sea debido a la diferencia de velocidades entre el módulo y la interfaz que se comunica vía serial.

Para salir del modo de Comandos se ingresa ATCN y se presiona ENTER. En caso de que no se ingrese ningún comando AT válido durante el tiempo determinado por CT (Command Mode Timeout), el módulo se saldrá automáticamente. Para que los cambios realizados tengan efecto se debe ingresar ATCN (sale del modo de comandos) o ATAC (aplica los cambios inmediatamente). Con el comando ATWR, se guardan los cambios en la memoria no volátil del módulo, pero sólo tendrán efecto una vez ingresado el comando AC o CN.

MODO TRANSPARENTE

En este modo todo lo que ingresa por el pin 3 (Data in), es guardado en el buffer de entrada y luego transmitido y todo lo que ingresa como paquete RF, es guardado en el buffer de salida y luego enviado por el pin 2 (Data out). El modo Transparente viene por defecto en los módulos XBee.

Este modo está destinado principalmente a la comunicación punto a punto, donde no es necesario ningún tipo de control. También se usa para reemplazar alguna conexión serial por cable, ya que es la configuración más sencilla posible y no requiere una mayor configuración.

En este modo, la información es recibida por el pin 3 del módulo XBee, y guardada en el buffer de entrada. Dependiendo de cómo se configure el comando RO, se puede transmitir la información apenas llegue un carácter (RO=0) o después de un tiempo dado sin recibir ningún carácter serial por el pin 3. En ese momento, se toma lo que se tenga en el buffer de entrada, se empaqueta, es decir, se integra a un paquete RF, y se transmite. Otra condición que puede cumplirse para la transmisión es cuando el buffer de entrada se llena, esto es, más de 100 bytes de información.

MODO API

Este modo es más complejo, pero permite el uso de frames¹⁶ con cabeceras que aseguran la entrega de los datos, al estilo TCP¹⁷. Extiende el nivel en el cual la aplicación del cliente, puede interactuar con las capacidades de red del módulo.

Cuando el módulo XBEE se encuentra en este modo, toda la información que entra y sale, es empaquetada en frames, que definen operaciones y eventos dentro del módulo. Así, un frame de Transmisión de Información (información recibida por el pin 3 o DIN) incluye:

- Frame de información RF transmitida.
- Frame de comandos (equivalente a comandos AT).

Mientras que un frame de Recepción de Información incluye:

- Frame de información RF recibida.
- Comando de respuesta.
- Notificaciones de eventos como Reset, Associate, Disassociate, etc.

Esta API, provee alternativas para la configuración del módulo y ruteo de la información en la capa de aplicación del cliente. Un cliente puede enviar información al módulo XBee. Estos datos serán contenidos en un frame cuya cabecera tendrá información útil referente el módulo.

Esta información además se podrá configurar, esto es, en vez de estar usando el modo de comandos para modificar las direcciones, la API lo realiza automáticamente. El módulo así enviará paquetes de datos contenidos en frames a otros módulos de destino, con información a sus respectivas aplicaciones, conteniendo paquetes de estado, así como el origen, RSSI (potencia de la señal de recepción) e información de la carga útil de los paquetes recibidos.

Entre las opciones que permite la API, se tienen:

- Transmitir información a múltiples destinatarios, sin entrar al modo de Comandos.
- Recibir estado de éxito/falla de cada paquete RF transmitido.
- Identificar la dirección de origen de cada paquete recibido.

¹⁶ Frame (trama de red o marco de datos). En redes de computadora, un frame (o marco o trama) es un paquete de datos de longitud fija o variable, que ha sido codificado por un protocolo de comunicaciones en la capa de enlace de datos, para la transmisión digital sobre un enlace nodo-a-nodo.

¹⁷ Transmission Control Protocol(*Protocolo de Control de Transmisión*) o TCP, es uno de los protocolos fundamentales en Internet. Muchos programas dentro de una red de datos compuesta por computadoras, pueden usar TCP para crear conexiones entre ellos a través de las cuales puede enviarse un flujo de datos.

IDLE

Cuando el módulo no se está en ninguno de los otros modos, se encuentra en éste. Es decir, si no está ni transmitiendo ni recibiendo, ni ahorrando energía ni en el modo de comandos, entonces se dice que se encuentra en un estado al que se le llama IDLE¹⁸.

SOFTWARE X-CTU

X-CTU es una aplicación basada en Windows proporcionada por Digi. Este programa fue diseñado para interactuar con los archivos de firmware que se encuentran en los productos de Digi RF y proporcionar a los usuarios una interfaz gráfica fácil de usar para ellos.

X-CTU está diseñado para funcionar con todos los equipos basados en Windows .CTU puede ser descargado desde el sitio web de Digi o un CD de instalación. Cuando se instala correctamente se puede iniciar haciendo clic en el ícono en el escritorio del PC.

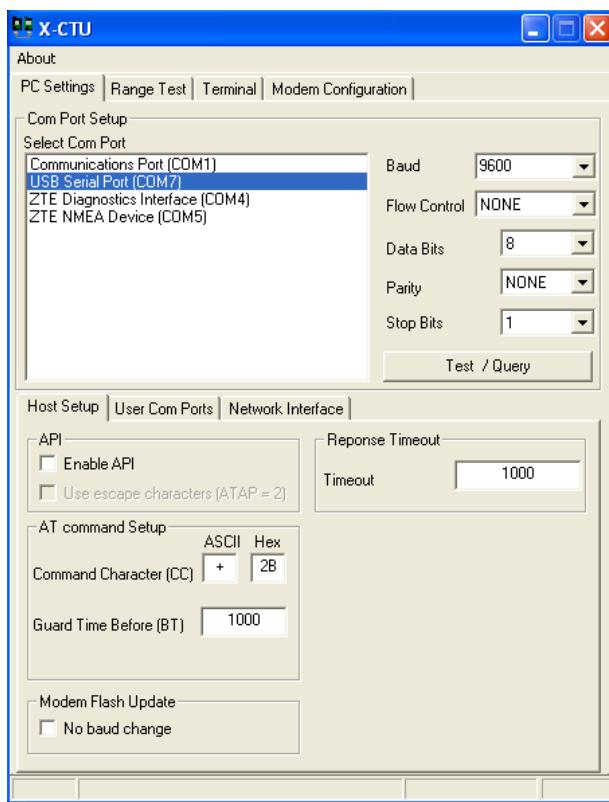


Figura 2.5. Interfaz gráfica del software X-CTU.

Cuando se inicia, verá cuatro pestañas en la parte superior del programa. Cada una de estas pestañas tiene una función diferente. Las cuatro pestañas son:

¹⁸ El **idle** o **Idle Time** es la inactividad de un usuario en IRC, que no quiere decir que no esté; sólo toma la inactividad dentro de IRC, no enteramente en la computadora.

Configuración de PC: Permite al usuario seleccionar el puerto COM deseado y configurar este puerto para adaptarse a la configuración de radios.

Prueba de Rangos: Permite al usuario enviar una cadena de datos de cualquier tipo para probar el rango de alcance de la señal. Esto genera automáticamente datos y los envía por el módulo, de tal forma que permite verificar cuales datos llegan buenos y cuáles no y a partir de esa estadística determinar el rango o alcance de la señal.

Terminal: Permite el acceso al puerto COM de la PC con un programa de emulación de terminal. Esta pestaña también permite la posibilidad de acceder a las radios 'firmware utilizando los comandos AT.

Configuración del Módem: Permite la capacidad de programar la configuración de las radios 'firmware a través de una interfaz gráfica de usuario. Esta pestaña también permite a los clientes la capacidad de cambiar las versiones de firmware.

CONFIGURACIÓN DE LOS MÓDULOS

En apartado contiene los comandos básicos que se encargan de configurar a los módulos en los diferentes modos de funcionamiento descritos anteriormente, dichos comandos pueden ser modificados ya sea desde la pestaña “Terminal” o desde la pestaña “Modem Configuration” de X-CTU o bien desde una terminal serial, (Hyperterminal en Windows o GTKTerm en Linux).

DIRECCIONAMIENTO

Los módulos permiten 2 tipos de direccionamiento. La de 16 bit y la de 64 bits. La principal diferencia es que en la de 64 bit, es posible obtener una mayor cantidad de direcciones y por lo tanto, una mayor cantidad de nodos o equipos funcionando en la misma red. Son a través de estas direcciones que los módulos se comunican entre sí.

DIRECCIONAMIENTO DE 16 BITS

El **comando MY**, define un número de 16 bit como dirección del módulo dentro de la red. El rango se encuentra entre 0x0 y 0xFFFF (la dirección 0xFFFF y 0xFFFF son para habilitar la dirección de 64-bit, por lo que si se desea utilizar direccionamiento de 16 bits, estos valores no deben ser usados). Para definirla se ingresa ATMY y el número en formato hexadecimal, pero sin el „0x“. Por ejemplo si a un módulo se le quiere asignar la dirección 0x3BF1 (15345 en decimal), entonces se debe ingresar el comando ATMY3BF1.

El **comando DL**, permite definir un número de 16 bit como dirección del módulo de destino dentro de la red al cual se va a realizar la comunicación. El rango debe estar entre 0x0 y 0XFFFF (las direcciones 0xFFFF y 0xFFFF se utilizan para direccionamiento de 64 bits).

Así para habilitar el direccionamiento de 16 bit, se debe utilizar una dirección menor a 0xFFFF con el comando MY, de igual modo para DL y se debe dejar en cero el comando **DH=0** (ATDH0). No se permite usar la dirección 0xFFFF ni 0xFFFF para el direccionamiento de 16.

DIRECCIONAMIENTO DE 64 BITS

El número 0xFFFF y 0xFFFF del comando MY, se usa cuando se desea desactivar el direccionamiento de 16 bit, y se habilita el uso de la dirección de 64 bit. Con este direccionamiento ya no es posible definir la dirección de origen del módulo, ya que ésta se asigna automáticamente. En este caso, la dirección del módulo corresponde a su número serial, que viene de fábrica y el cual es imposible de cambiar. Este número se encuentra guardado en dos variables de 32 bit cada una (**SL** y **SH**) y es único. SL lee los 32 bit menos significativos del número serial y SH los 32 más significativos.

Cuando se utiliza direccionamiento de 64 bit, para asignar una dirección de destino, se utilizan los **comandos DL y DH**. Éstos son de 32 bit cada uno (para el direccionamiento de 16 bit, DL se maneja como uno de 16, mientras que DH se mantiene en cero) y juntos (DL+DH) forman el número de 64 bit que debe corresponder con el número serial de otro módulo formado por SL+SH. Así para algún dato, DL debe ser igual a SL y DH debe ser igual a SH, donde SL+SH corresponden al número serial de un módulo destino configurado para direccionamiento de 64 bits.

Para el direccionamiento de 64 bit, se debe dejar MY como 0xFFFF (ATMYFFFF) o 0xFFFFE (ATMYFFFE) y elegir una dirección de destino usando DL+DH, que debe corresponder a una dirección de 64 bit de otro módulo, indicando su número serial dado por SL+SH. Para consultar este número se debe ingresar ATSL (32 bit menos significativos) y luego ATSH (32 bit más significativos), entregando como respuesta los números seriales en formato hexadecimal.

MODO DE CONEXIÓN TRANSPARENTE

Esta es la conexión que viene por defecto y es la más sencilla forma de configurar el módem. Básicamente todo lo que pasa por el puerto UART (DIN, pin 3), es enviado al módulo deseado, y lo que llega al módulo, es enviado devuelta por el mismo puerto UART (DOUT, pin2).

Existen básicamente 4 tipos de conexión transparente. La diferencia principal radica en el número de nodos o puntos de acceso, y la forma en que éstos interactúan entre sí.

PUNTO A PUNTO

Es la conexión ideal para reemplazar comunicación serial por un cable. Sólo se debe configurar la dirección. Para ello se utilizan los comandos MY y el DL. La idea, es que se define arbitrariamente una dirección para un módulo, usando el comando MY, el cual se va a comunicar con otro que tiene la dirección DL, también definida arbitrariamente. Con esto cada módulo define su dirección con MY, y escribe la dirección del módulo al cual se desea conectar usando DL.

En este modo, el módulo receptor del mensaje envía un paquete al módulo de origen llamado ACK¹⁹ que indica que el mensaje se recibió correctamente. Una vez configurado, el módem se encuentra listo para funcionar. Así todo lo que se transmite por el pin DIN de un módulo, es recibido por el pin DOUT del otro. Para que el modo Punto a Punto funcione, los módulos deben pertenecer a la misma PAN ID y al mismo canal.

PUNTO A MULTIPUNTO.

Esta conexión, permite prestaciones extras. Se diferencia del Broadcast, en que permite transmitir información, desde la entrada serial de un módulo (DIN, pin 3) a uno o varios módulos conectados a la misma red de manera más controlada, ya que se necesitan las direcciones de los otros módulos, por lo que existe mayor seguridad. Para esto se necesitan dos comandos más aparte de MY y DL. Se utilizará el direccionamiento de 16 bits.

El primer es el **comando ID** de la PAN (Personal Area Network- Red de Área Personal). Todos los módulos que tengan idéntico PAN ID, pertenecerán a la misma red. El comando para configurar este valor es ID, es decir, ATID, y su rango va entre 0x0 y 0xFFFF. Por ejemplo si queremos ajustar el PAN ID como 0x3332, se debe ingresar ATID3332. Este parámetro también es arbitrario, al igual que MY y DL.

El otro comando corresponde al canal por el cual se va a comunicar. Se disponen de 16 canales según el protocolo IEEE 802.15.4. Esta norma indica que entre cada canal, deben existir 5 MHz de diferencia, partiendo de la frecuencia base 2.405 GHz, se llegan hasta los 2.480 GHz.

Hay 16 canales disponibles, sin embargo, los valores se asignan desde el 11 hasta el 26. Para calcular la frecuencia central se utiliza la siguiente fórmula:

$$\text{Canal} = 2.405 + 0.005 \text{ } CH - 11 \text{ [GHz]}$$

Donde CH equivale al número del canal entre 11 y 26. Así para cambiar de canal se utiliza el **comando CH** con el número de canal en formato hexadecimal. Es decir, si se desea ocupar el canal 15 (0x10), se ingresa ATCH10. La siguiente tabla muestra la frecuencia central de cada canal, así como su límite inferior y superior:

Tabla 2.3 Límites de frecuencia y frecuencia central de cada canal RF.

Canal	Hexadecimal	Inferior [GHz]	Central [GHz]	Superior [GHz]	Comando AT
11	0x0B	2,4025	2,4050	2,4075	ATCH0B
12	0x0C	2,4075	2,4100	2,4125	ATCH0C
13	0x0D	2,4125	2,4150	2,4175	ATCH0D
14	0x0E	2,4175	2,4200	2,4225	ATCH0E
15	0x0F	2,4225	2,4250	2,4275	ATCH0F
16	0x10	2,4275	2,4300	2,4325	ATCH10

¹⁹ ACK, viene de Acknowledge que en español significa “reconocimiento”.

Canal	Hexadecimal	Inferior [GHz]	Central [GHz]	Superior [GHz]	Comando AT
17	0x11	2,4325	2,4350	2,4375	ATCH11
18	0x12	2,4375	2,4400	2,4425	ATCH12
19	0x13	2,4425	2,4450	2,4475	ATCH13
20	0x14	2,4475	2,4500	2,4525	ATCH14
21	0x15	2,4525	2,4550	2,4575	ATCH15
22	0x16	2,4575	2,4600	2,4625	ATCH16
23	0x17	2,4625	2,4650	2,4675	ATCH17
24	0x18	2,4675	2,4700	2,4725	ATCH18
24	0x19	2,4725	2,4750	2,4775	ATCH19
26	0x1A	2,4775	2,4800	2,4825	ATCH1A
Frecuencia base = 2,405 GHz					

La elección del canal debe ser cuidadosa, ya que otras tecnologías como Wi-Fi o Bluetooth utilizan el mismo espectro de frecuencias, por lo que se podría producir interferencia.

Con todo lo anterior, es posible configurar una PAN y hacer una conexión punto a multipunto. Así en cada nodo se configura una dirección MY distinta, pero utilizando el mismo canal y el mismo PAN ID, para que cada módulo reciba la información, debe ser estrictamente necesario que tengan tanto el mismo canal, como el mismo PAN ID. Incluso si se trabaja en Broadcast o punto a punto los módulos deben coincidir en ello. Los módulos vienen con el canal 0x0C y el PAN ID 0x3332 por defecto.

Esta configuración, permite enviar información más controlada, ya que es necesario pertenecer tanto al mismo canal, como a la misma red. Además para enviar información se debe ingresar la dirección del módulo de destino, por lo que es necesario el conocimiento completo de la red. En los módulos más avanzados, como XBEE PRO, el reconocimiento de la red se realiza automáticamente

BROADCAST

Esta configuración permite el envío de información desde un nodo a varios nodos en una misma red. La información recibida es la misma para todos los nodos. Para configurar los módulos, es necesario ajustarlos con la dirección de Broadcast. Cualquier módulo que reciba un paquete con una dirección de destino de Broadcast será aceptado.

La dirección de Broadcast es:

DL=0x0000FFFF

DH=0x00000000

Esta dirección debe ser configurada en todos los nodos de la red, ya sea que estén en direccionamiento de 16 o 64 bits. Así se debe ingresar ATDH0 y ATDL0000FFFF en todos los módulos para que el modo broadcast esté habilitado.

Si se envía algún dato por un módulo, la información enviada será recibida por igual en el resto de los módulos. Del mismo modo si se envía algún dato por otro módulo, este dato le llegará al resto. Cabe mencionar que este tipo de red o de envío de datos, no entrega respuesta de recibo o ACK, por lo que no es posible saber si el paquete fue entregado correctamente o si es que llegó.

Si se ajusta la dirección PAN ID del módulo como ID=0xFFFF, se produce Broadcast a todas las redes PAN. Esto es, los datos son transmitidos a las distintas redes PAN, pero no se confirma la entrega de éstos (no se recibe ACK). Si se ingresa ID=0xFFFF y además DL=0xFFFF se realiza doble broadcast, es decir, además de transmitirse los datos a todas las redes PAN, el mensaje es transmitido a todos los módulos de cada una de ellas. Si se ingresa ID=0xFFFF y DL=0xAAAA (dirección arbitraria), los datos son transmitidos a todos los módulos que posean la dirección AAAA, pero que no necesariamente se encuentren en la misma red PAN.

CABLE VIRTUAL I/O

Esta opción de configuración permite crear los llamados Cables Virtuales. Se utilizan para crear un canal de comunicación de manera transparente entre los pines de un módulo y otro.

Cada pin de entrada tiene su propio pin de salida ya definido entre nodos, esto permite una forma totalmente simple de enviar información, controlar o medir de manera sencilla y rápida, sin necesidad de complicadas configuraciones (ver distribución de pines).

CONEXIÓN NONBEACON. PEER-TO-PEER.

Una red peer-to-peer permite que todos los módulos, se conecten con todos, es decir, se crea una conexión de par en par con cada uno de los módulos de la red. El modo de conexión NonBeacon²⁰ es la configuración por defecto y permite establecer una red peer-to-peer donde cada módulo puede hacer las funciones de maestro o esclavo.

La configuración de red Non-Beacon, se refiere a que cada nodo se mantiene despierto siempre. Por lo que los demás dispositivos que se conectan a él, pueden entrar en modo SLEEP (ahorro de energía), y sólo despertarse cuando sea necesario para enviar datos. En una red Beacon, los dispositivos enrutadores están siempre en modo SLEEP, y envían señales de su existencia (llamadas Beacon) cada cierto intervalo al resto de la red. Así para poder comunicarse, deben estar totalmente organizados todos los dispositivos, ya que de no ser así, existe la posibilidad de perderse la señal Beacon y no poder enviar hasta la próxima entrega. La ventaja de las redes

²⁰ Beacon (faro), representa la manera en como los dispositivos entran en estado de sueño y despiertan periódicamente enviando una “señal beacon” o “señal faro” para pedir correspondencia pendiente al dispositivo coordinador.

Beacon, es el ahorro de energía. Por este motivo las redes Non-Beacon están pensadas para dispositivos que posean una alimentación segura, mientras las Beacon, para alimentación autónoma, como baterías. Los módulos XBEE Series 1, sólo soportan redes NonBeacon.

Para esto, cada módulo se debe configurar como Dispositivo Terminal (End Device) y todos deben tener el mismo canal (ATCH) y la misma PAN (ATID). Para configurar los módulos como dispositivos terminales, se debe ingresar el **comando CE=0 (ATCEO)**.

CONEXIÓN NONBEACON C/COORDINADOR.

Es básicamente lo mismo que una red punto-multipunto, con la diferencia de que existe un módulo central que posee ciertas propiedades y características que le permiten administrar mejor la red. En esta red, el módulo central es llamado Coordinador, mientras que el resto de módulos son llamados Dispositivos Terminales (End Device). Un mismo módulo XBee puede ser configurado para funcionar como Coordinador o como Dispositivo Terminal.

Para configurar esta red, todos los módulos deben tener el mismo canal (ATCH) y la misma PAN (ATID). El módulo Coordinador se configura como ATCE=1 (ATCE1), mientras que todos los demás, los cuales serán llamados Dispositivos Terminales, se configuran como ATCE=0 (ATCEO).

En este tipo de red, los Coordinadores pueden ser usados para usar transmisiones directas o indirectas. En las directas, la información es enviada de inmediato, mientras que en la indirecta, la información es guardada un tiempo dado por el parámetro SP (Cyclic Sleep Period). Si SP=0, la transmisión es directa. Si SP está en un rango entre 1 y 0x68B0 (x10 milisegundos), es el tiempo que espera antes de enviar.

Para este tipo de configuraciones NonBeacon con Coordinador, se requiere crear una relación llamada Asociación. Ésta mantiene un control del Coordinador sobre los Dispositivos Terminales. Este tipo de configuración con un Coordinador se utiliza cuando se requiere una unidad central para enviar mensajes a varios módulos, o juntar información proveniente de varios Dispositivos Terminales, asignar canales o ID de redes PAN.

Una red de datos RF consistente de un Coordinador y uno o varios Dispositivos Terminales, forman lo que se llama una PAN (Personal Area Network). Cada dispositivo en una PAN tiene un identificador llamado ID (ATID), el cual debe ser el mismo para todos los módulos de la misma PAN.

El ID del Coordinador se debe configurar utilizando dos comandos. El primero el ya visto ID (ATID), mientras que el segundo corresponde al A2 (ATA2 – Coordinator Association). Un dispositivo Terminal se puede asociar a un Coordinador, sin saber la dirección, el PAN ID o el canal al cual está conectado. El parámetro A1 (ATA1 – End Device Association), asigna dinámicamente la dirección, canal y PAN ID para asignarse a un coordinador. Además este parámetro determina la flexibilidad de un dispositivo Terminal para realizar la asociación.

Cabe mencionar, que tanto los módulos en modo Coordinador, como los en modo Dispositivos Terminales deben tener la misma versión del Firmware. Una red PAN, puede ser configurada usando varios módulos como dispositivos terminales y uno maestro como coordinador. Los dispositivos terminales se configuran con el comando A1, mientras que los Coordinadores con el A2.

El **comando A1** puede tener un rango entre 0 y 0xF (16 en decimal y 1111 en binario). Se observa que escrito en binario, éste valor posee 4 bit. Luego cada uno de esos bits configura ciertas características del Dispositivo Terminal cuando CE=0. La siguiente tabla indica el nombre de cada bit y su configuración dado cierto valor.

Tabla 2.4 Descripción de configuración de cada bit del comando A1.

Nº	Número de bit	valor	Configuración Dispositivo Terminal usando A1
0	ReassingPanID	0	Se asociará con un coordinador que opere en una PAN ID que coincida con la del nodo identificador.
		1	Se puede asociar con un coordinador que opere en cualquier PAN ID.
1	ReassingChannel	0	Se asociará con el Coordinador que opere en el mismo canal que el valor de CH (Canal) del nodo.
		1	Se puede asociar a un Coordinador que opere en cualquier canal.
2	AutoAssociate	0	El dispositivo no intentará asociarse.
		1	El dispositivo intentará asociarse hasta que tenga éxito.
3	PollCoordOnPinWake	0	El PIN WAKE, no le preguntará al Coordinador por data pendiente.
		1	El PIN WAKE, enviará Solicitudes de Encuesta al Coordinador para extraer cualquier data pendiente.

Por ejemplo, la configuración por defecto es 0x06 (0110 en binario), donde el bit N°0, corresponde al último dígito en la secuencia.

El **comando A2** puede tener un rango entre 0 y 0x07 (111 en binario). Cada uno de estos bits, configura el modo de operación del coordinador, en caso de que el dispositivo se encuentre configurado como Coordinador (CE=1).

Tabla 2.5 Descripción de configuración de cada bit del comando 2.

Nº	Número de bit	Valor	Configuración Coordinador usando A2
0	ReassingPanID	0	Coordinador no realizará Active Scan para localizar PAN ID disponibles. Operará sobre su PAN dada por el parámetro ID.
		1	Coordinador realizará Active Scan para determinar una PAN ID disponible. Si una PAN ID tiene conflicto, el parámetro ID

Nº	Número de bit	Valor	Configuración Coordinador usando A2
			cambiará.
1	ReassingChannel	0	Coordinador no realizará Energy Scan para determinar canales libres. Operará en el canal determinado por el parámetro CH.
		1	Coordinador realizará Energy Scan para encontrar canales libres. Luego operará sobre ese canal.
2	AllowAssociate	0	Coordinador no permitirá a ningún dispositivo asociarse a él.
		1	Coordinador permitirá que dispositivos se asocien.

MODO DE CONEXIÓN API

Esta conexión, agrega información extra a los paquetes de datos RF. Ya no son enviados de forma transparente, sino que cada paquete de datos, son almacenados dentro de un frame, con una estructura definida que permite una forma más robusta para enviar datos. Esto permite entre otras cosas determinar el origen de algún paquete recibido dentro de la red.

Cuando la configuración API está activada, cada paquete RF que se envía o que se recibe se encapsula en un frame de datos UART. Para esto se utiliza el **comando AP**.

Existen tres posibilidades de configuración. Con AP=0, se deshabilita el frame API y el módulo trabaja en modo transparente. Con AP=1, el módulo trabaja en el modo API. Y con AP=2, el módulo trabaja en modo API, pero con Carácter de Escape. Este modo es necesario sólo cuando se envían bytes que interfieren con la estructura del frame. Éstos son:

- ⊕ 0x7E – Delimitador de Frame.
- ⊕ 0x7D – Escape
- ⊕ 0x11 – XON
- ⊕ 0x13 – XOFF

Este modo, ingresa un carácter de escape, además de otra operación sobre el bytes de interferencia. Esto hace que el frame sea más grande, al agregar bytes, pero evita que la cabecera del frame se confunda con los datos enviados. Otra ventaja es el Checksum, que permite verificar que los datos entregados no se hayan corrompido.

Entre las posibilidades que permite la API, es la posibilidad de cambiar parámetros a través de comandos AT, enviándolos al módulo de destino. Así, desde un módulo, es posible configurar otro utilizando el modo API. También es posible consultar sobre el estado de algún parámetro en otro módulo. Además se puede consultar sobre el estado del módem, como saber si está asociado a un coordinador, o si el módulo es o no un coordinador.

COMANDOS IMPORTANTES

A continuación se muestra una tabla resumen con los comandos AT para el módulo XBee más importantes. El contenido se muestra con el rango permitido por el comando, una descripción, y las configuraciones para cada valor del parámetro. Para utilizar el programa se debe ingresar AT y luego, sin espacios, el comando a configurar y el valor del parámetro en caso de que se quiera ajustar, o sin nada en caso de que se quiera consultar el valor de ese parámetro.

Tabla 2.6 Resumen y descripción de comandos importantes.

Comando AT	Rango	Descripción
A1	0 - 0x0F	Describe el modo de Asociación de un módulo. Utilizado como Dispositivo Terminal (CE=0). Defecto=0.
A2	0 - 0x0F	Describe el modo de Asociación de un módulo utilizado como Coordinador (CE=1). Defecto=0.
AC	-	Aplica los cambios realizados explícitamente en la configuración.
AP	0-0x02	Habilita el modo de operación API. Defecto=0. 0 Modo API Deshabilitado. 1 Modo API habilitado. 2 Modo API habilitado con carácter de escape.
BD	0 – 0x7	Ajusta la tasa de transmisión entre el módulo y su cliente conectado a través de la interfaz serial. Para valores no-estándar revisar el manual. Defecto=3. 0 1200 1 2400 2 4800 3 9600 4 19200 5 38400 6 57600 7 115200
CC	0 – 0xFF	Establece el carácter de secuencia a ser usado entre tiempos de esperas para entrar al modo de comandos. Defecto=0x2B (carácter ASCII +)
CH	0x0B – 0x1A	Establece el canal por el cual se realiza la conexión RF entre módulos. Verificar Tabla 5-1 Frecuencia de Canales para configurar este parámetro. Defecto=0x0C.
CE	0 – 1	Indica el comportamiento del módulo. Defecto=0. 0 Dispositivo Terminal. 1 Coordinador.
CN	-	Sale del modo de Comando.
D0 – D4	0 – 5	Ajusta la configuración de los pines I/O. 0 Deshabilitado. 1 (n/a)

Comando AT	Rango	Descripción
		2 ADC. 3 Entrada Digital. 4 Salida Digital LOW.
D5	0 – 5	Mismas funciones que D0 - D4, exceptuando lo siguiente: 1 Indicador de asociación.
D6	0 – 5	Mismas funciones que D0 - D4, exceptuando lo siguiente: 1 Control de Flujo RTS. 2 No tiene conversor ADC.
D7	0 – 5	Mismas funciones que D0 - D4, exceptuando lo siguiente: 1 Control de Flujo CTS. 2 No tiene conversor ADC.
D8	Solo 0 y 3	Ajusta la configuración del pin DI-8 (pin 9). 0 Deshabilitado. 3 Entrada Digital.
DB	0x17 – 0x5C (x – 1dB)	Lee la potencia de la señal del módulo del cual provino el último paquete RF recibido.
DL	0 - 0xFFFFFFFF	Ajusta los 32 bits menos significativos para direccionamiento. Defecto = 0.
DH	0 - 0xFFFFFFFF	Ajusta los 32 bits más significativos para direccionamiento. Defecto = 0.
GT	2 - 0x0CE4 (x 1 ms)	Tiempo de espera antes y despues de ingresar el carácter de secuencia para entrar al modo de comandos. Defecto = 0x3E8.
IA	0 - 0xFFFFFFFFFFFFFF	Utilizado para crear el Cable Virtual. Indica la dirección del módulo de origen de los datos. Defecto= 0xFFFFFFFFFFFFFF (no permite el recibo de ningún paquete para cambiar las salidas.)
ID	0 - 0xFFFF	Ajusta la dirección PAN del módulo. Defecto = 0x3332
IR	0 - 0xFFFF (x1 ms)	Ajusta la tasa de muestreo de los pines I/O. Defecto = 0.
IS	1 - 0xFF	Fuerza al módulo a leer todos sus pines I/O. Si AP=0, el resultado se retorna del siguiente modo: - Número de Muestras. - Máscara de Canal. - Datos DIO. - Datos conversores ADC (se repite por cada conversor habilitado).
IT	1 - 0xFF	Número de muestras DIO y ADC que se deben esperar, antes de transmitir. Defecto = 1.
IO	8 bits	Ajusta los niveles de las salidas digitales. Cada bits representa el nivel de los pines I/O configurados como salida.
MO – M1	0 - 0x03FF	Ajusta el ciclo de trabajo de la salida PWM0 y PWM1. Si Mn=0 (0% PWM), Mn=0x01FF (50% PWM) y si

Comando AT	Rango	Descripción
		Mn=0x03FF (100% PWM). Defecto=0
MY	0 - 0xFFFF	Configura la dirección de 16 bits para el módulo. Si My=0xFFFF o 0xFFFE, se habilita el modo de direccionamiento de 64 bit. Defecto = 0.
NB	0 – 4	Ajusta la Paridad para la comunicación serial UART del módulo. Defecto =0. 0 8 bit sin paridad o 7 bit con cualquier paridad. 1 8 bit even. 2 8 bit odd. 3 8 bit mark. 4 8 bit space.
ND	-	Reporta todos los dispositivos que se encuentren en el mismo canal y en la misma PAN que el módulo. El formato de respuesta es el siguiente cuando se encuentra en el modo Transparente. - MY (dirección de 16 bit) - SH (Serial Number High) - SL (Serial Nmuber Low). - DB (Fuerza de la señal proveniente de este módulo)
NI	String de 20 caracteres ASCII.	- NI (Identificador del Nodo) Define con un String el nodo o módulo
P0 – P1	0 – 2	Configura el pin PWM0 y PWM1. Defecto P0 =1, Defecto P1=0. 0 Deshabilitado. 1 RSSI. 2 PWM habilitado.
RE	-	Restaura los valores de los parámetros a los valores por defecto que vienen de fábrica.
SM	0 – 6	Configura el modo de operación SLEEP. Defecto = 0. 0 Deshabilitado. 1 Pin de Hibernado. 2 Pin Doze. 3 (reservado) 4 Remoto Cyclic SLEEP. 5 Remoto Cyclic SLEEP (con pin Wake-up). 6 SLEEP Coordinador
SL	0 - 0xFFFFFFFF	Entrega los 32 bit menos significativos del Número Serial del módulo
SH	0 - 0xFFFFFFFF	Entrega los 32 bit más significativos del Número Serial del módulo.
SP	1 - 0x68B0 (x10 ms)	Ajusta el tiempo de duración en que un módulo duerme o se mantiene en el modo SLEEP. Una vez terminado el período, busca por data entrante, si no hay nada vuelve a

Comando AT	Rango	Descripción
		dormir y espera por un nuevo ciclo.
ST	1 - 0xFFFF (x1 ms)	Ajusta el tiempo de inactividad (datos ni recibidos ni enviados ya sea por RF o serial) antes de que el módulo ingrese al modo SLEEP. Defecto = 0x1388.
T0 – T7	0 - 0xFF (x100 ms)	Tiempo de espera de apagado para los Cables Virtuales. Si luego de este tiempo no hay cambios en alguna salida I/O, éstas cambian a su valor por defecto. Defecto = 0xFF.
IU	0 – 1	Habilita o no la salida I/O UART 0 Deshabilitado. Paquetes RF recibidos no serán enviados por UART. 1 Habilitado. Paquetes RF recibidos serán enviados por UART.
VL	-	Entrega la versión del Firmware de forma Verbal. La respuesta entrega fecha de compilación de la aplicación, MAC, PHY y versión del bootloader y sus fechas de compilación.
VR	0 - 0Xffff	Indica cual versión de firmware se encuentra actualmente en el módulo.
WR	-	Guarda en la memoria no-volátil del módulo, todos los valores de los parámetros.

CAPÍTULO 3: PROCEDIMIENTOS PRELIMINARES DE INTEGRACIÓN ADE7753-ARDUINO-LABVIEW.

Se pretende utilizar Arduino para que funcione como dispositivo controlador o dispositivo Master del IC ADE7753, este se encarga de realizar los muestreos de la señal y de realizar los cálculos correspondientes para obtener parámetros de voltaje, corriente y potencia, luego los guarda en sus registros de memoria que posee y los actualiza continuamente; el trabajo de Arduino consiste entonces en poder realizar lecturas y escrituras de configuración del IC, todo esto se lleva a cabo bajo el estándar de comunicación SPI que soportan ambos dispositivos.

Además se requiere tener la capacidad de almacenamiento de datos para que estos puedan ser descargados en cualquier momento, para esto se hace uso de un Datalogger que básicamente se conforma de dos componentes básicos que son: un puerto para memoria SD (o Micro SD) y un RTC (Real Time Clock) que servirá para poder guardar los datos con el tiempo en que fueron adquiridos.

Una vez adquiridos y documentados, estos datos deben ser descargados para su análisis, esto se pretende implementar por medio de un VI de LabVIEW, el VI de LabVIEW deberá proporcionar al usuario un control sobre los datos almacenados en el dispositivo Arduino, para lograr esta interacción LabVIEW posee una conjunto de herramientas de comunicación Arduino-LabVIEW la cual permite generar una interfaz gráfica para controlar al Arduino de manera intuitiva para el usuario.

INTEGRACIÓN ENTRE ARDUINO Y ADE7753 - DATALOGGER

LIBRERIAS E INICIALIZACIONES PRINCIPALES PARA COMUNICACIÓN ADE7753-ARDUINO.

El IDE Arduino ya incorpora la librería <SPI.h> que por supuesto es de código abierto, la configuración necesaria para poder comunicarse con el IC se basa en 4 sentencias principales que dependen particularmente de los dispositivos que van a funcionar en modo esclavo.

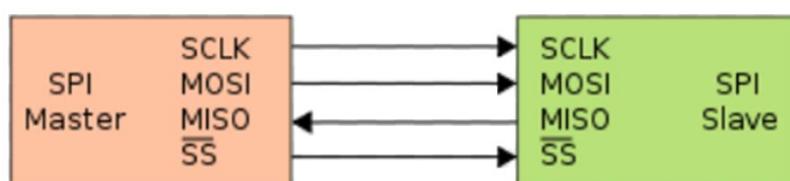


Figura 3.1 Esquema básico Maestro-Esclavo de comunicación SPI.

En el micro controlador Arduino los pines son como sigue:

-MOSI Master Out Slave In. Es el pin D11.

-MISO Master In Slave Out. Es el pin D12.

-SCLK Serial Clock. Es el pin D13.

-SSSlave Selector. Puede ser cualquier pin digital conveniente.

Para conocer los parámetros es necesario leer cuidadosamente las especificaciones de los fabricantes del IC; para el ADE7753 tenemos que la primera sentencia importante es:

```
SPI.setDataMode(SPI_MODE2);
```

En general, hay 4 modos de transmisión. Estos modos controlan si los datos salen o entran en un flanco de subida o bajada de la señal de reloj, esto se llama ‘fase del reloj’. Y cuando el reloj está libre es decir su estado inicial se llama polaridad del reloj.

Tabla 3.1 Los cuatro modos diferentes de trabajo del reloj.

MODO	Polaridad del Reloj CPOL	Fase del Reloj CPHA
SPI_MODE0	0	0
SPI_MODE1	0	1
SPI_MODE2	1	0
SPI_MODE3	1	1

El ADE7753 utiliza el modo 2 por tanto si **CPHA = 0**, el reloj de cambio es la OR de SCLK con la terminal Slave Selector. Tan pronto como el terminal Slave Selector se coloca en un nivel lógico 0, los nuevos datos se ponen en la línea y el primer filo del reloj se leen los datos. Si **CPOL** se activa a un nivel lógico ‘1’, el primer borde de reloj baja y los bits de datos subsecuentes se leen en cada filo de bajada sobre la línea de reloj. Cada nuevo bit se pone en la línea cuando el reloj tiene un flanco ascendente de Reloj.

La segunda sentencia importante es la configuración de la frecuencia del SCLK o los pulsos de reloj para la comunicación SPI.

```
SPI.setClockDivider(SPI_CLOCK_DIV32);
```

Esta sentencia configura el divisor del reloj SPI relativo al reloj del sistema, para Arduino la frecuencia del sistema es de 16MHz, por lo tanto la frecuencia correspondiente que se le ha configurado al SCLK es de 500KHz.

Otra sentencia que es clave para poder realizar la comunicación con el ADE7753 es el orden en que salen y entran los bits.

```
SPI.setBitOrder(MSBFIRST);
```

Esto es importante puesto que configura si los bits más significativos van primero o si lo menos significativos se envían primero, por supuesto tanto en la escritura como en la lectura se tiene la misma configuración.

La cuarta sentencia inicializa el bus SPI, declarando como salidas SCLK, MOSI, SS; poniendo SCLK y MOSI en bajo y en alto el SS. Pero para que la configuración del Slave Selector tenga lugar debe hacerse como se muestra en la segunda sentencia a continuación.

```
SPI.begin();
```

```
SPI.begin(slaveSelectPin)
```

Para este caso específico se usa la primera sentencia pues tendremos más dispositivos esclavos.

LIBRERIAS, SENTENCIAS IMPORTANTES PARA INTEGRACIÓN ADE7753-DATALOGGER-ARDUINO.

Hay dos problemas al intentar hacer esta integración, en primer lugar está el hecho que los pines para el almacenamiento en la memoria SD son los mismos que los utilizados por el ADE7753, es decir ambos dispositivos utilizan el protocolo SPI de comunicación; el procedimiento para resolver este problema es el siguiente:

Se tienen que utilizar los “selectores de chip” o “slave selector” de ambos dispositivos de manera que cada vez que queramos comunicarnos uno en específico los pines del otro dispositivo tienen que estar en alta impedancia o “eléctricamente desconectados” de esa manera se asegura que la comunicación se está realizando con un solo dispositivo a la vez.

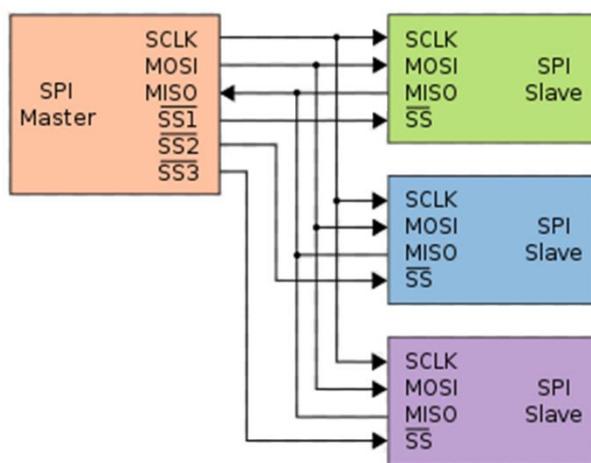


Figura 3.2 esquema general para trabajar con varios dispositivos esclavos con protocolo SPI.

Escogeremos el pin D9 del Arduino como *SS* del IC porque la shield de la Datalogger viene configurada para utilizar el pin D10. Por lo tanto el esquema ilustrativo físico de conexión es el que

se muestra en la figura 3.3, cada vez que se necesite comunicarse con uno de los dispositivos se tiene que acompañar de los siguientes comandos:

```
digitalWrite(9,LOW);
```

"aquí instrucciones de comunicación con dispositivo."

```
digitalWrite(9,HIGH);
```

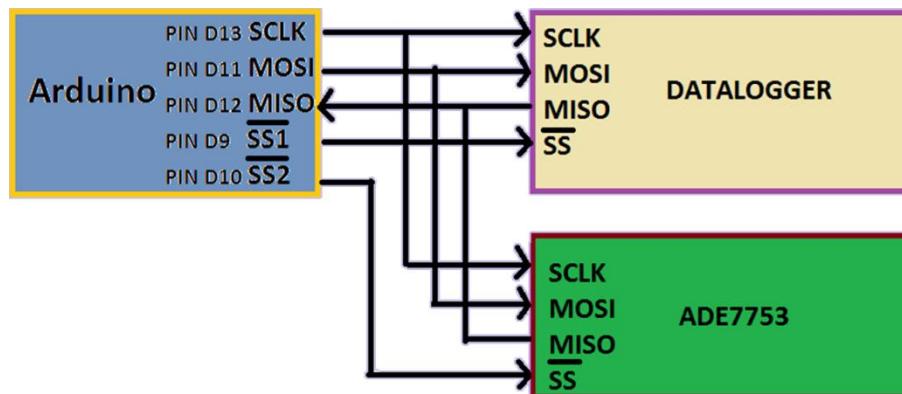


Figura 3.3 esquema de conexión entre Arduino-Datalogger-ADE7753.

El segundo problema que aparece a la hora de integrar estos dispositivos tiene que ver con las librerías que utiliza la Datalogger, estas son <SD.h><Wire.h><RTCLib.h> las dos ultimas no generan problema alguno, pero en el código de la librería <SD.h> aparentemente no se define la polaridad del reloj de trabajo que tendrá la comunicación SPI por lo tanto significa que por defecto trabaja con "SPI_MODE0", es decir el almacenamiento de datos en una SD se realiza con una polaridad distinta a la que utiliza el circuito integrado. Según la documentación encontrada sobre este tema este no suele ser un problema común debido a que todas las shield que se hacen para Arduino siempre trabajan con la misma polaridad de reloj, por tanto la peculiaridad del IC de trabajar con "SPI_MODE2" impide que la comunicación se lleve a cabo, siendo insuficiente los selectores de chip.

La solución para esto es la siguiente:

- 1- La inicialización de la comunicación SPI con el integrado debe preceder a la inicialización de la Datalogger.

Lo que esto significa es que el comando "while (!Serial)" y el comando "if (!SD.begin(10))" que sirven para inicializar la comunicación con la SD deben de estar después de haber hecho las declaraciones de polaridad del reloj para el integrado y del comando "SPI.begin()"

2- Cada vez que se guarden datos en la SD debe ir precedido del correspondiente comando de cambio de polaridad y al finalizar la comunicación se deberá restablecer la polaridad para que la comunicación con el IC sea posible.

```
SPI.setDataMode(SPI_MODE0);
```

"Aquí instrucciones para guardar o leer datos."

```
SPI.setDataMode(SPI_MODE2);
```

LABVIEW

Como es bien conocido, la plataforma Arduino es un micro controlador de software y hardware libre que se utiliza en muchas aplicaciones en el área de informática aplicada.

Mientras que **LabVIEW** es el acrónimo de Laboratorio de Instrumentación Virtual para Plataformas de Trabajo de Ingeniería, (Laboratory Virtual Instrumentation Engineering Workbench) es un entorno de desarrollo altamente productivo que los ingenieros y científicos utilizan para la programación gráfica y la integración de hardware sin precedentes, para diseñar y desplegar rápidamente sistemas de medidas y control. En esta plataforma flexible, los ingenieros escalan del diseño a las pruebas y de sistemas pequeños a grandes, al reutilizar IP y perfeccionar sus procesos para alcanzar el rendimiento máximo.

Este programa fue creado por National Instruments (1976) para funcionar sobre máquinas MAC, salió al mercado por primera vez en 1986. Ahora está disponible para las plataformas Windows, UNIX, MAC y GNU/Linux.

Los programas desarrollados con LabVIEW se llaman Instrumentos Virtuales, o VIs, y su origen provenía del control de instrumentos, aunque hoy en día se ha expandido ampliamente no sólo al control de todo tipo de electrónica sino también a su programación implícita, comunicaciones, matemáticas, etc. LabVIEW consigue combinarse con todo tipo de software y hardware, tanto del propio fabricante -tarjetas de adquisición de datos, PAC, Visión, instrumentos y otro Hardware como de otros fabricantes.

Características principales de LabVIEW:

 **Lenguaje desarrollado para medición, control y automatización**

A diferencia de los lenguajes de propósito general, LabVIEW provee funcionalidad específica para que pueda acelerar el desarrollo de aplicaciones de medición, control y automatización.

 **Ambiente de desarrollo intuitivo para incrementar la productividad**

LabVIEW entrega herramientas poderosas para crear aplicaciones sin líneas de texto de código. Con LabVIEW se jala y coloca objetos ya construidos para rápidamente crear interfaces de usuario. Después se especifica la funcionalidad del sistema armando diagramas de bloques.

 **Fácil Integración con miles de instrumentos y dispositivos de medición**

LabVIEW se puede conectar de manera transparente con virtualmente todo tipo de hardware incluyendo instrumentos de escritorio, tarjetas insertables, controladores de movimiento y controladores lógicos programables (PLCs).

 **Ambiente abierto para usar con otras aplicaciones**

Con LabVIEW se puede conectar con otras aplicaciones y compartir datos a través de ActiveX, la Web, DLLs, librerías compartidas, SQL, TCP/IP, XML, OPC y otros.

 **Compilado para optimizar el desempeño del sistema**

En muchas aplicaciones, la velocidad de ejecución es vital. Con un compilador incluido que genera código optimizado, sus aplicaciones en LabVIEW entregan velocidades de ejecución comparables con programas C compilados. Con LabVIEW puede desarrollar sistemas que cumplan con sus requerimientos de desempeño a través de las plataformas incluyendo Windows, Macintosh, UNIX o sistemas de tiempo real.

LABVIEW INTERFACE FOR ARDUINO (LVIFA O LIFA)

ARDUINO TOOLKIT

El Toolkit NI LabVIEW Interface para Arduino ayuda a establecer una interfaz fácilmente con el microcontrolador Arduino usando LabVIEW.

Con este juego de herramientas y LabVIEW, se puede controlar y adquirir datos desde el microcontrolador Arduino. Una vez que la información está en LabVIEW, se puede analizar usando los cientos de bibliotecas integradas de LabVIEW, es posible desarrollar algoritmos para controlar el hardware Arduino y presentar conclusiones en un VI.

Un sketch para el microcontrolador Arduino actúa como un motor de E/S que se conecta con los VIs de LabVIEW a través de una conexión serial. Esto le ayuda a mover información rápidamente desde pines Arduino a LabVIEW sin ajustar la comunicación, la sincronización o incluso una sola línea de código C. Al usar Open, Read/Write, Close en LabVIEW, se puede tener acceso a las señales digitales, analógicas, moduladas por ancho de pulso, I²C y SPI del microcontrolador Arduino.

El microcontrolador Arduino debe estar conectado a la PC con LabVIEW a través de un enlace USB, serial, Bluetooth o XBee. Este juego de herramientas no hace posible una operación autónoma.

En resumen: para utilizar el Arduino con el Kit de herramientas de LabVIEW, es necesario instalar el sketch, que convierte el Arduino en un dispositivo que responde a los comandos para leer las

entradas de Arduino, y el establecimiento de las salidas desde el ordenador a través de la conexión USB.

OBTENCIÓN E INSTALACIÓN DEL ARDUINO TOOLKIT

REQUISITOS:

- ✚ Versión instalada de LabVIEW 2009 o mayor.
- ✚ Las versiones más recientes de NI-VISA Drivers.
- ✚ Instalar el IDE Arduino y controladores para Windows.
- ✚ Instalar el LabVIEW 2011 Run-Time Engine (si no se cuenta con la versión LabVIEW 2011 o superior).
- ✚ Instalar VIPM(VI Package Manager).

Uno de los principales requisitos es tener LabVIEW 2009 o posterior instalado. Los VI que se incluyen en el LVIFA se guardan en LV 2009, por lo que esta es la versión de LV que se debe tener para ser capaz de utilizar el LVIFA. Si no se tiene esta versión de LabVIEW, se puede descargar una evaluación de 30 días desde ni.com / tryLabVIEW para probar el toolkit.

Además se debe instalar NI-VISA Drivers. Para LabVIEW, el Arduino aparece como un dispositivo serial. Para comunicarse con los instrumentos seriales en LabVIEW, se necesita tener la última versión del controlador NI-VISA. Esto también se puede resolver instalando el “LabVIEW 2011 Run-Time Engine”.

OBTENCIÓN DEL LVIFA:

El LVIFA está disponible como un paquete VI a través de la Red de Herramientas de LabVIEW. Dicho paquete se puede descargar gratuitamente de la página web de National Instruments: www.ni.com habrá que crear una cuenta si es primera vez que se accede a la página como se muestra en la figura 3.4:

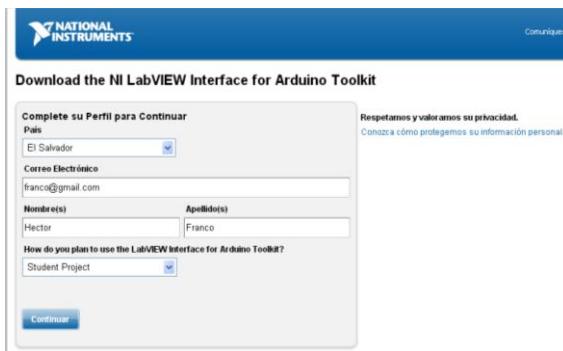


Figura 3.4 Crear cuenta en www.ni.com.

A continuación se mostrará una página con el link para realizar la descarga del LVIFA. Además esta página (figura 3.5) hace mención de los requisitos previos para poder utilizar el Toolkit así que si no se poseen dichos elementos se pueden descargar e instalar cómodamente desde esta misma página.

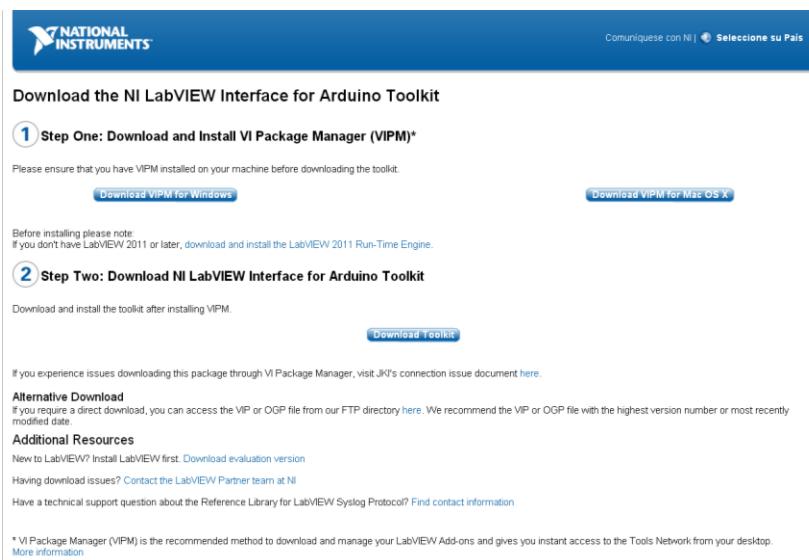


Figura 3.5 Pagina de descarga del LIVFA.

Primero se debe instalar VIPM (VI Package Manager), este es el gestor de paquetes de LabVIEW lo más recomendable es utilizar siempre el VIPM para instalar o desinstalar herramientas de LabVIEW. Una vez VIPM está instalado, se muestra una interface como se muestra en las figuras 3.6, 3.7 y 3.8:



Figura 3.6 Ventana de inicio del VIPM.

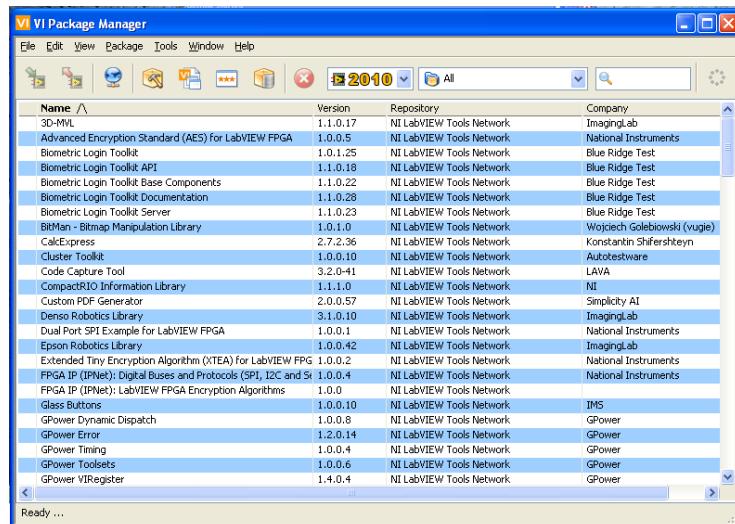


Figura 3.7 Interface del VIPM.

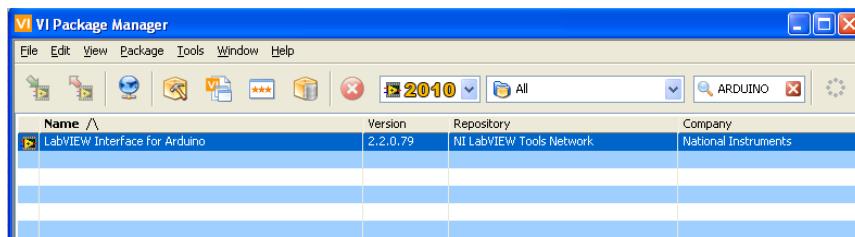


Figura 3.8 VIPM para buscar el LVIFA.

Se puede desde el VIPM hacer una búsqueda del LVIFA como se muestra en la figura 3.8 pero la mejor manera de obtener el LVIFA es haciendo clik sobre el botón “Download toolkit” (ver figura 3.5) esto genera una activación automática para el VIPM para descargar directamente el toolkit, como se muestra en la figura 3.9, si se cumplen los requisitos para instalar el toolkit bastará con dar clik en “Install” para completar el proceso.

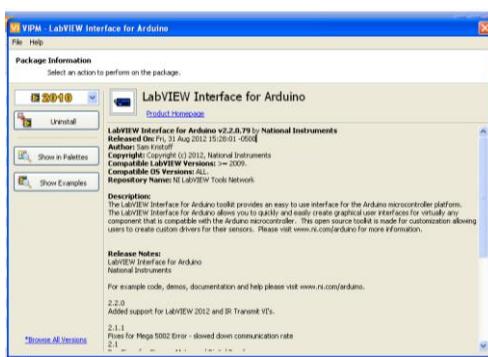


Figura 3.9 ventana de descarga de LVIFA.

Una vez terminada la instalación se debe montar el sketch 'LIFA_Base.pde' al Arduino. La LIFA viene con un sketch que debe ser cargado en el Arduino para poder utilizar los VIs de comunicación con él. Se debe utilizar el software de Arduino IDE instalado para hacer esto. El sketch se encuentra en la dirección:

C:\ Archivos de programa \ National Instruments \ LabVIEW 2010 \ vi.lib \ LabVIEW Interfaz para Arduino \ Firmware \ LVIFA_Base \ LVIFA_Base.pde

HERRAMIENTAS DEL LVIFA

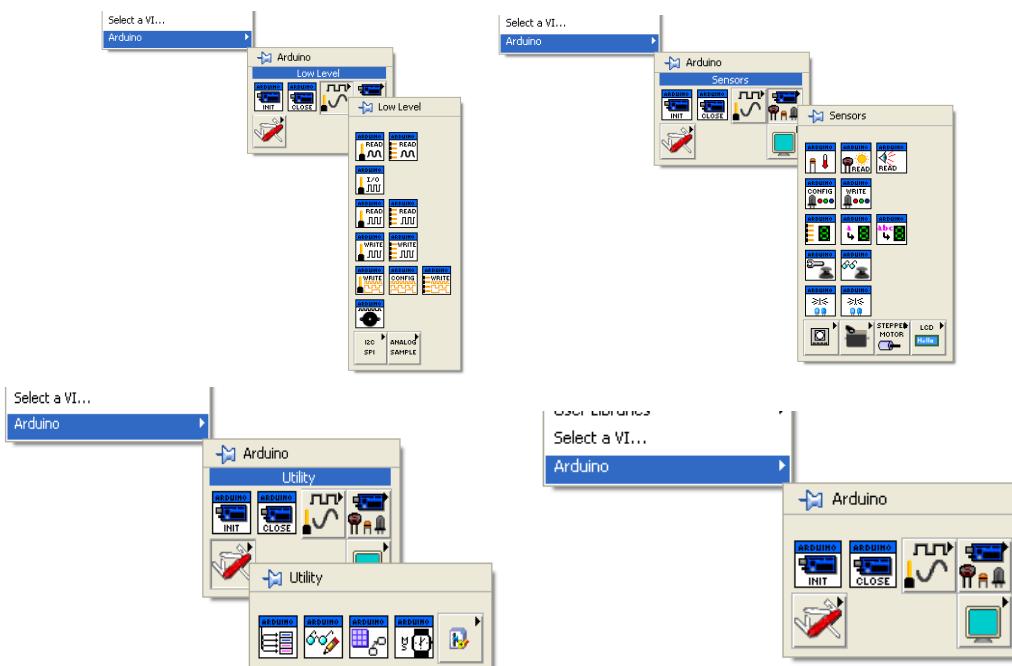


Figura 3.10 Paleta de herramientas Arduino.

Principales funciones de la paleta de herramientas Arduino:

- Init (iniciar comunicación)
- Close (finalizar comunicación)
- Low level
 - ✓ Analog read pin (Leer de un pin analógico)
 - ✓ Analog read port (Leer el puerto analógico)
 - ✓ Set digital pin mode (Cambiar el modo de un pin: entrada/salida)
 - ✓ Digital read pin (leer de un pin digital)
 - ✓ Digital read port (Leer el puerto digital)

- ✓ Digital write pin (Escribir en un pin digital)
- ✓ Digital write port (Escribir en el puerto digital)
- ✓ PWM write pin (Escribir en un pin PWM)
- ✓ PWM write port (Escribir en el puerto PWM)
- ✓ PWM configure port (Configurar el puerto PWM)
- ✓ Tone (Tono)
- ⊕ Sensors (Sensores)
 - ✓ Thermistor read (Leer termistor)
 - ✓ Photocell read (Leer foto celda)
 - ✓ IR sensor read (Leer sensor IR)
 - ✓ RGB led configure (Configurar un led RGB)
 - ✓ RGB led write (Escribir en un led RGB)
 - ✓ Seven segment configure (Configurar un siete segmentos)
 - ✓ Seven segment write char (Mapa de escritura para siete segmentos)
 - ✓ Seven segment write string (Escribir cadena para siete segmentos)
 - ✓ Thumbstick configure (Configurar una palanca para pulgar)
 - ✓ Thumbstick read (Leer palanca para pulgar)
 - ✓ Servo (Servo motor)
 - ✓ Stepper Motor (Motores paso)
 - ✓ LCD
- ⊕ Utility (Herramientas)
 - ✓ Packetize (Empaquetar datos)
 - ✓ Send receive (Enviar recibir)
 - ✓ Calculate update rates (Calcula frecuencias y tiempo actuales)
 - ✓ Get timing data (Obtiene datos de tiempo)
 - ✓ Helpers (Ayuda)
- ⊕ Examples (Ejemplos)

NI-VISA

La Arquitectura de Software de Instrumentos Virtuales (VISA, Virtual Instrument Software Architecture) es un estándar para la configuración, programación y solución de problemas para sistemas de instrumentación que comprenden interfaces GPIB, VXI, PXI, Serial, Ethernet, y/o USB. VISA proporciona la interfaz de programación entre los entornos de hardware y desarrollo, tales como LabVIEW, LabWindows/CVI y Measurement Studio para Microsoft Visual Studio.

NI-VISA es la implementación de National Instruments de la norma VISA estándar E/S. NI-VISA incluye librerías de software, servicios interactivos, como la NI trazo de E/S y el Control Interactivo VISA y programas de configuración a través del Measurement & Automation Explorer. NI-VISA es un estándar en toda la línea de productos de National Instruments.

Las herramientas del LVIFA están basadas en NI-VISA para lograr la comunicación con la tarjeta Arduino a través del puerto USB, pero LabVIEW además posee toda una paleta de funciones para utilizar herramientas de comunicación VISA²¹ (ver figura 3.11).

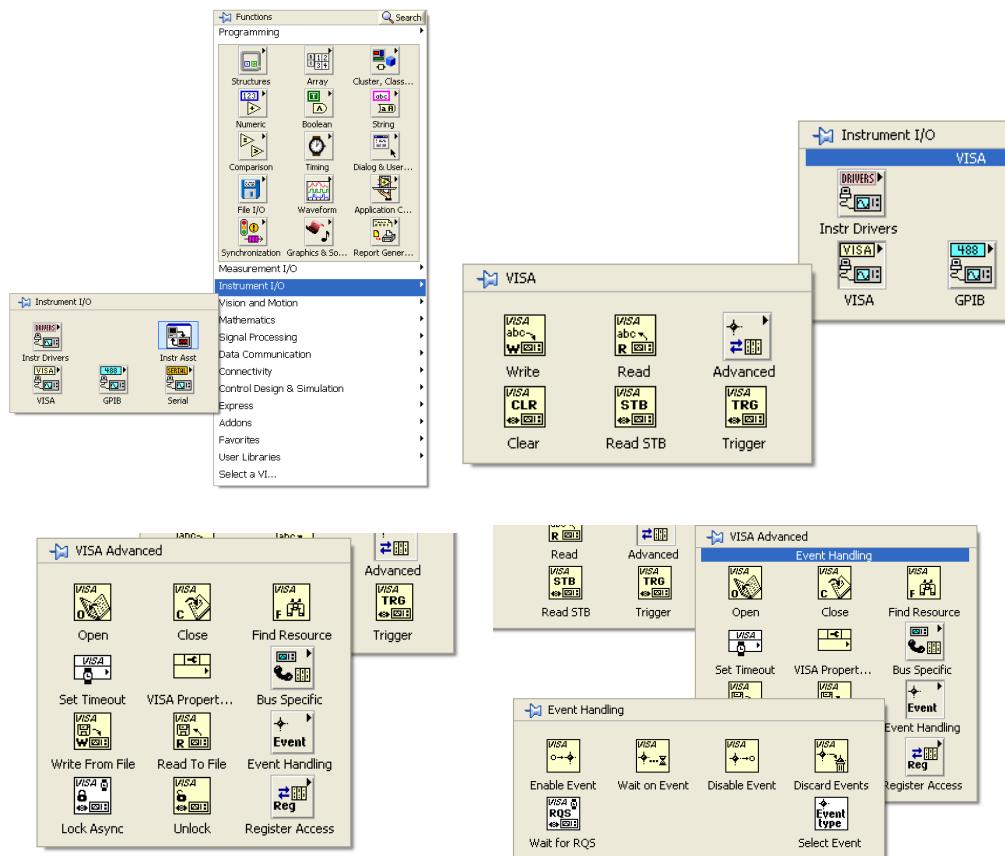


Figura 3.11 Paleta de herramientas NI-VISA.

Utilizando los SubVI's primitivos²² de la paleta VISA es también posible comunicarse directamente con los puertos de la PC, para el caso permite la comunicación con el puerto USB, lo que representa un instrumento más directo para enlazar con la tarjeta Arduino.

Ni VISA contra LVIFA

LVIFA VENTAJAS

Permite una comunicación directa y segura con el microcontrolador Arduino moviendo información rápidamente desde pines Arduino a LabVIEW sin ajustar la comunicación, la

²¹ Para poder hacer uso de las funciones de esta paleta es necesario instalar la última versión de NI-VISA drivers las cuales se pueden descargar gratuitamente de la página de National Instruments.

²² SubVI's son VI básicos que se ofrecen como funciones a las cuales no se puede obtener acceso al código fuente ni se pueden realizar modificaciones, se reconocen por tener un color amarrillo suave.

sincronización o incluso una sola línea de código C. Y tener acceso a todas las funciones Open, Read/Write, Close en LabVIEW, y acceso a las señales digitales, analógicas, moduladas por ancho de pulso, I²C y SPI del microcontrolador Arduino.

LVIFA DESVENTAJAS

El sketch para el microcontrolador Arduino que actúa como un motor de E/S que se conecta con los VIs de LabVIEW utiliza mucho espacio en memoria, cuestión que es de suma importancia tomando en cuenta las condiciones de hardware con las que se cuenta, es decir, este proyecto dispone de una tarjeta Arduino UNO para ser ejecutado, dicha tarjeta dispone de 32KB de memoria flash, tan solo el sketch LVIFA utiliza el 80% de esta capacidad, dejando aprietos los demás requerimientos del código para controlar a la memoria, el RTC y primordialmente al ADE7753.

NI-VISA VENTAJAS

Permite una comunicación serial directa, haciendo posible la comunicación con el microcontrolador Arduino tal como la experimentada con una terminal serial, la principal ventaja es que esta opción no compromete ni un tan solo byte en el Sketch Arduino, dejando el espacio en memoria flash disponible para las librerías y el código que se requieren para el control del hardware del medidor (memoria, RTC y ADE7753), Además por ser una programación de más bajo nivel que la implementación con LVIFA se posee un mayor control y claridad sobre los aspectos de codificación de la comunicación serial.

NI-VISA DESVENTAJAS

Al no ser un protocolo predefinido ni diseñado a profundidad es necesario tomar en cuenta hasta las consideraciones más básicas de la interacción LabVIEW-Arduino como el BaudRate, la sincronización de lectura escritura, etc. Además es importante destacar que de esta manera no se dispone de una conexión directa a los pines o puertos de pines de la tarjeta Arduino, ni a todas las facilidades de comunicación que ofrece el LVIFA.

MÉTODO A UTILIZAR

Principalmente por limitaciones de espacio en la memoria flash en el desarrollo del proyecto se optó por utilizar las herramientas NI-VISA para establecer la comunicación serial con el medidor inalámbrico.

CAPÍTULO 4: HARDWARE PARA IMPLEMENTACIÓN DEL MEDIDOR.

DIAGRAMA DE CONEXIÓN DEL CIRCUITO.

Se muestra en la figura 4.1 el diagrama de conexión general del circuito. La RTC, junto con la memoria SD se encuentran en una misma shield, se han separado en el esquema en dos componentes para que quede claridad en cuales son los pines del Arduino que se están utilizando y cuáles de ellos quedan libres. La RTC es un circuito integrado que funciona con el protocolo I2C, lo que significa que necesita dos alambres para comunicarse. Estos dos alambres son usados para configurar el tiempo y conservarlo aun cuando el Arduino no se encuentre energizado, los pines están arreglados para que sean el analógico 4 y 5 del Arduino los cuales soportan el protocolo I2C.

Por otra parte la memoria SD utiliza el protocolo SPI, como ya se explicó antes y se refleja en el diagrama de conexión compartiendo los mismos pines que el ADE7753, pero el “chip selector” va al pin 10 del Arduino.

La XBEE solamente utiliza dos pines para la comunicación con el Arduino TX y RX, la shield del Arduino tiene un switch que interrumpe esta conexión y los conecta con los pines D2 y D3 del Arduino, esta conexión no se muestra porque solamente tiene lugar cuando es necesario programar el Arduino, el funcionamiento y la conexión permanente es tal como se muestra en el diagrama de conexión.

La conexión principal o más importante es la circuitería para controlar el ADE7753 que se muestra en la figura 4.2 con el objetivo de facilitar el entendimiento, es esta conexión la base para hacer la shield para el ADE7753, a partir de esta se inicia el diseño del PCB.

DISEÑO DEL PCB

El diseño se hizo con EAGLE “Easily Applicable Graphical Layout Editor” se utilizó la versión 6.1.0 para Windows, se utilizaron las librerías que trae por defecto el programa pero también fue necesario agregar librerías que los vendedores de electrónicos como “Sparkfun” ponen a disposición de sus clientes para facilitar el diseño de los PCB. En la figura 4.3 se muestra un componente de las librerías agregadas, este es un shield para el Arduino que permite disponer de las medidas reales y la distribución real que tienen los pines, dentro de este espacio se conectaron los componentes que se muestran en la figura 4.2, contando con esto el siguiente paso es contar con los componentes discretos a utilizar físicamente para el montaje y tomar las medidas correspondiente de estos y buscar su equivalente más aproximado en las librerías del Eagle.

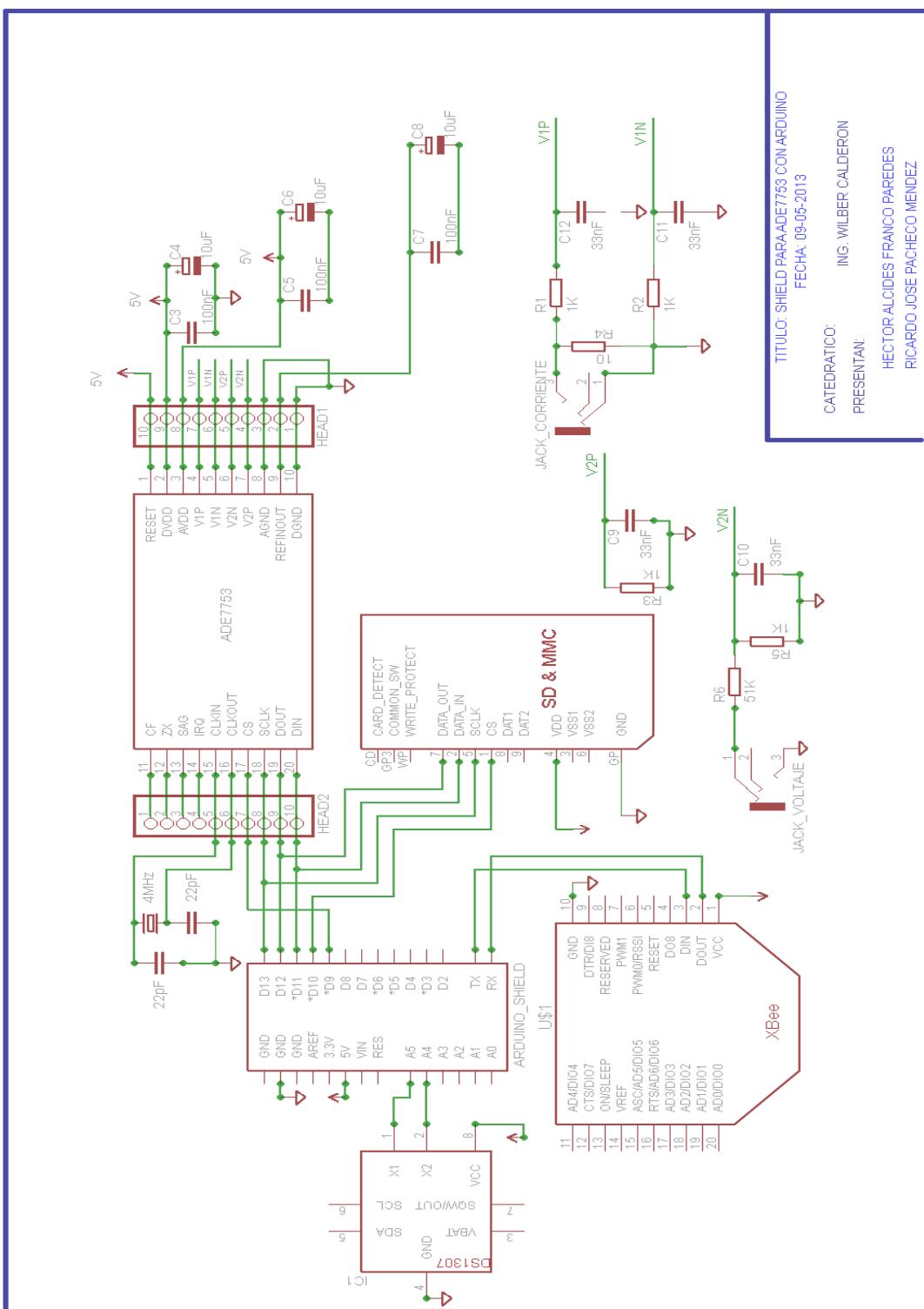


Figura 4.1 Esquema de conexión general de todo el Hardware.

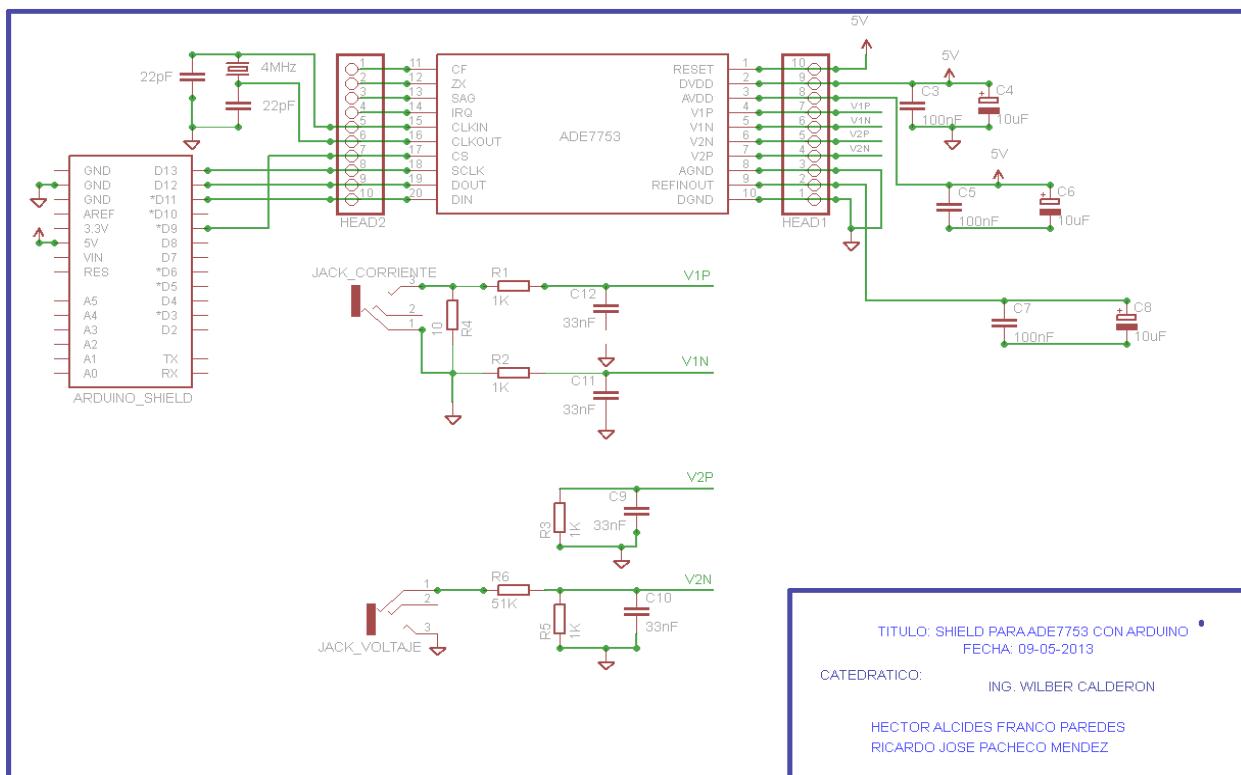


Figura 4.2 Esquema de conexión de la shield para el ADE7753.

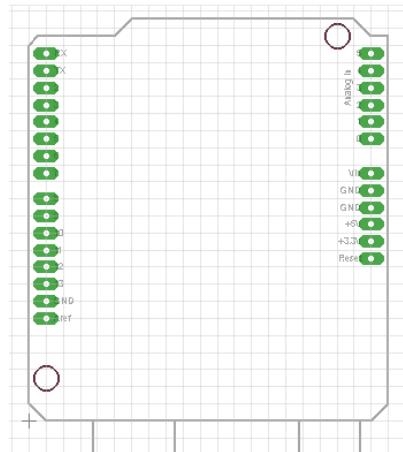


Figura 4.3 Componente para el diseño del PCB, shield para Arduino que se encuentra en las librerías de Sparkfun.

Continuando con el diseño del PCB teniendo en cuenta que el circuito integrado es de montaje superficial con encapsulado SSOP “Shrink Small Outline Plastic Packages” y con teniendo en cuenta que hacer circuitos impresos de manera artesanal imposibilita pistas de tamaños muy delgados se decidió por utilizar un adaptador de encapsulado de SSOP a DIP “Dual Inline Package” de 20 pines como el que se muestra en la figura 4.4.

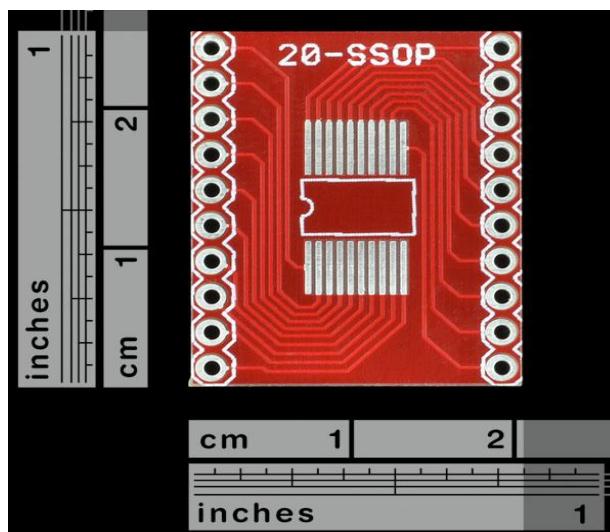


Figura 4.4 Adaptador SSOP a DIP utilizado en el montaje de la shield.

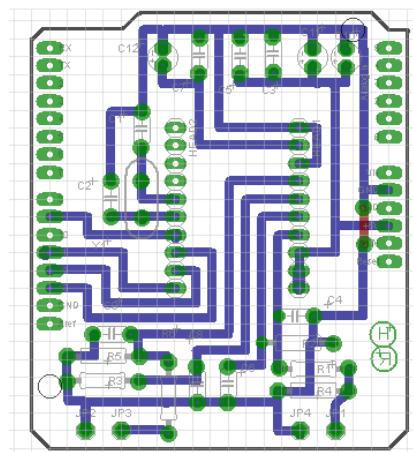


Figura 4.5 Diseño terminado del PCB para la shield del medidor.

Como es visible en la figura 4.5 en el diseño del PCB, no se utilizó el componente Eagle del circuito integrado ADE7753, en lugar de este se usaron headers para simular el adaptador utilizado.

IMPRESIÓN DEL PCB.

El esquema listo para impresión sobre una placa de cobre se muestra en la figura 4.6; algunos de los componentes utilizados en la impresión de la placa fueron:

- Placa de fibra de vidrio cobreada a una cara.
- Ácido percloruro de hierro.
- Papel couché (gramaje de 60)

- Impresión digital.
- Papel adhesivo.
- Plástico adhesivo.
- Imagen de las pistas a escala real.
- Elementos.
- Estaño.

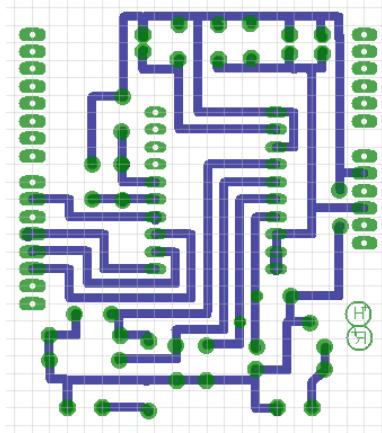


Figura 4.6 PCB listo para imprimir.

En la figura 4.7, 4.8 y 4.9 muestran un poco del proceso para llevar acabo la impresión del PCB sobre una placa.

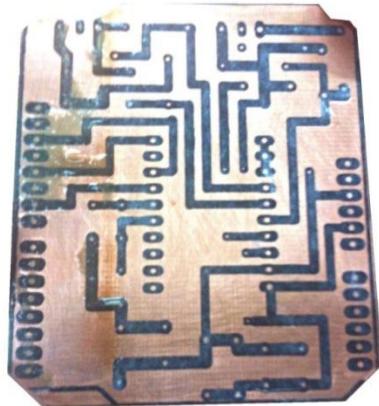


Figura 4.7 Pistas recién impresas sobre la placa de cobre.

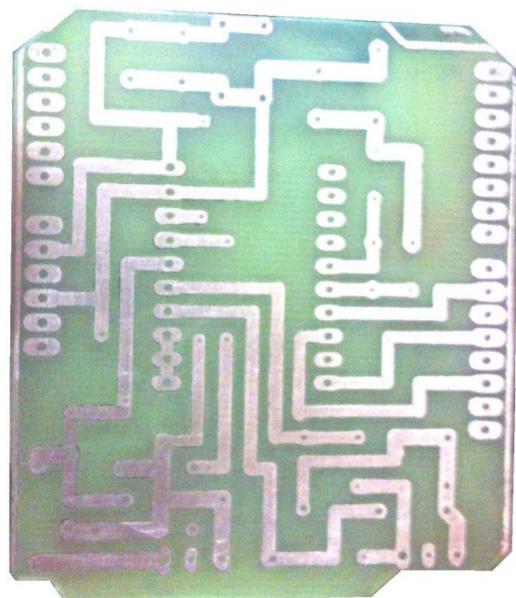


Figura 4.8 El cobre ha sido removido de la placa dejando únicamente las pistas.

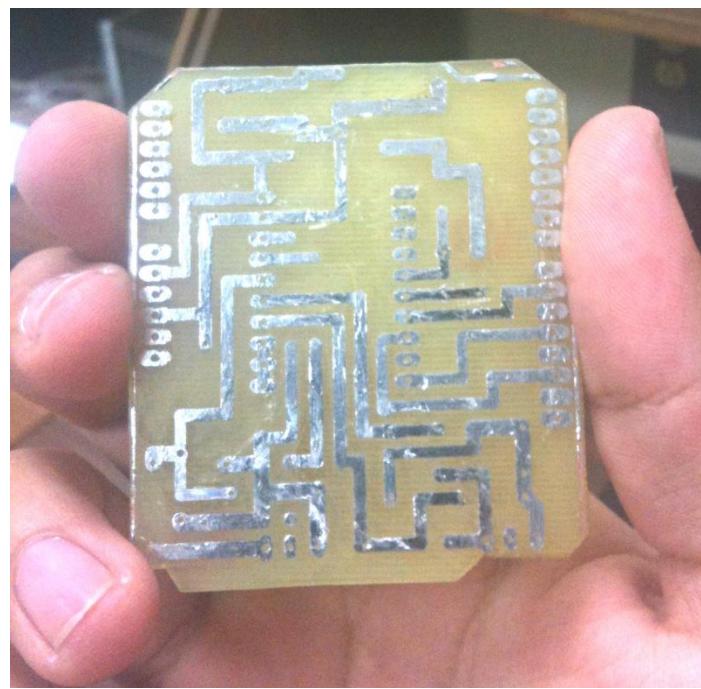


Figura 4.9 Pistas estañadas.

SHIELD ARDUINO MEDIDOR ADE7753

La shield para el ADE7753 terminada se muestra en las siguientes imágenes.

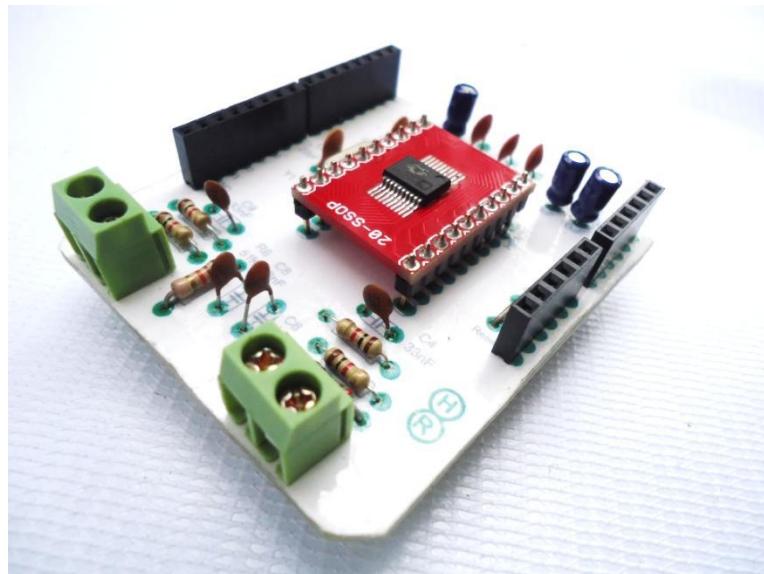


Figura 4.10 Shield Arduino medidor ADE7753

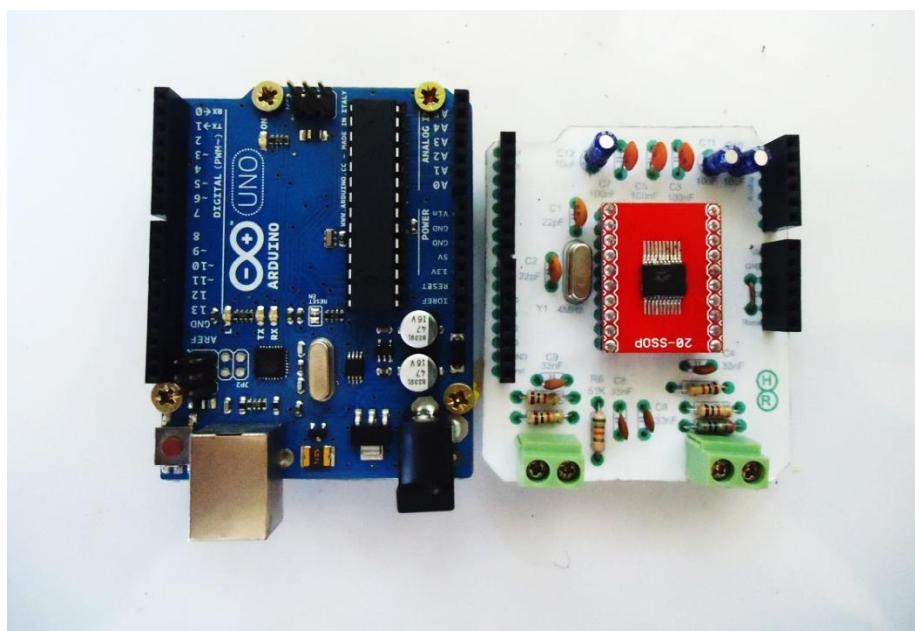


Figura 4.11 Comparación con tarjeta Arduino UNO.

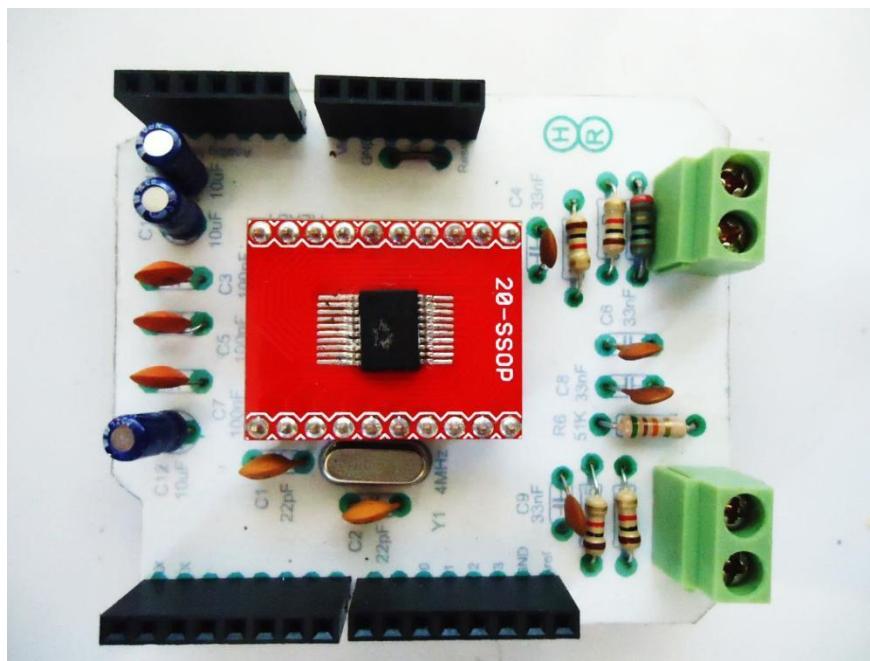


Figura 4.12 Vista superior.

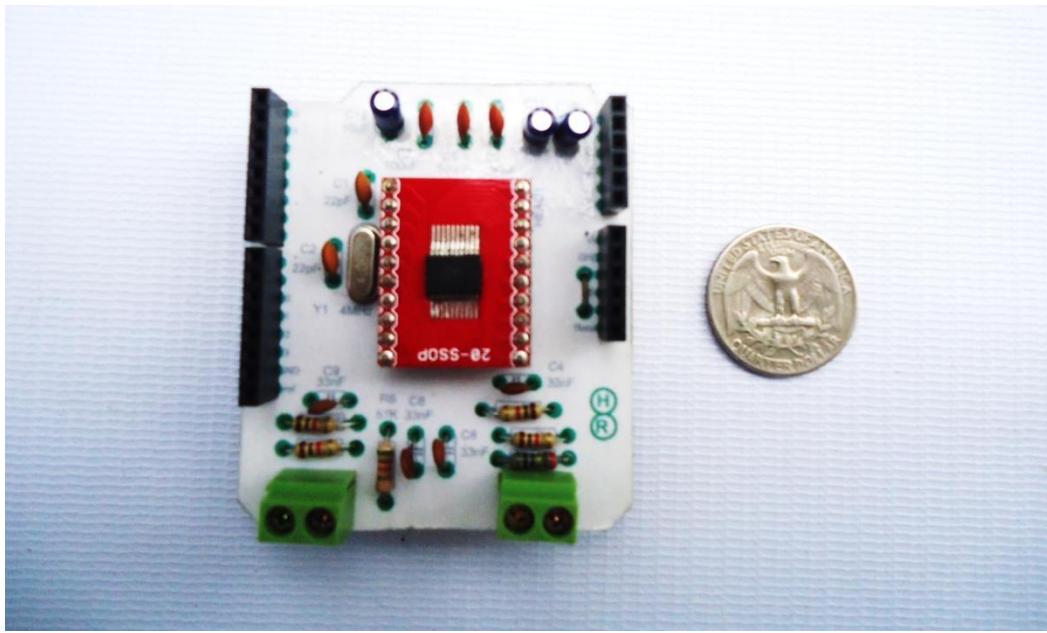


Figura 4.13 Comparación de escala con una moneda.

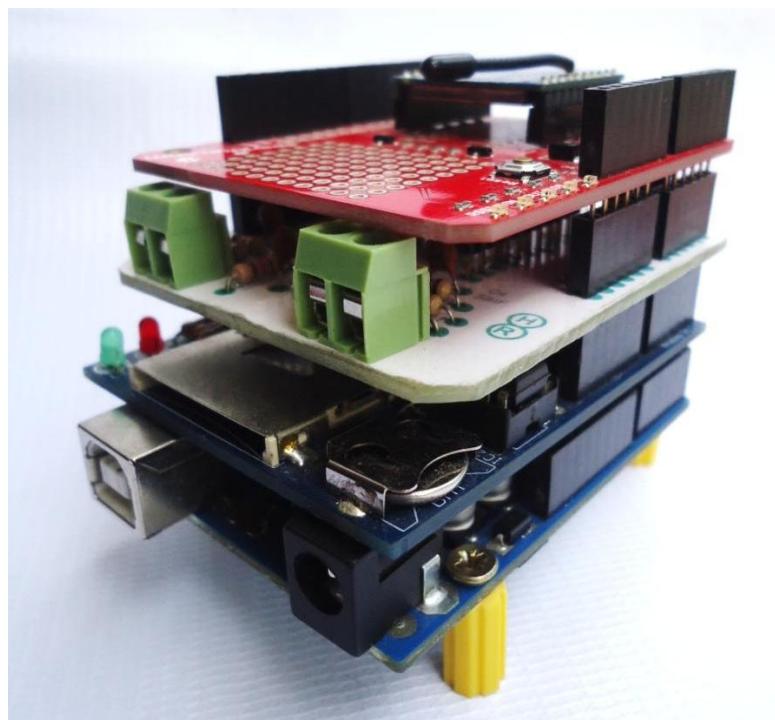


Figura 4.14 Montaje de módulos Arduino Uno, Data Logger, shield medidor ADE7753 y XBee Shield.

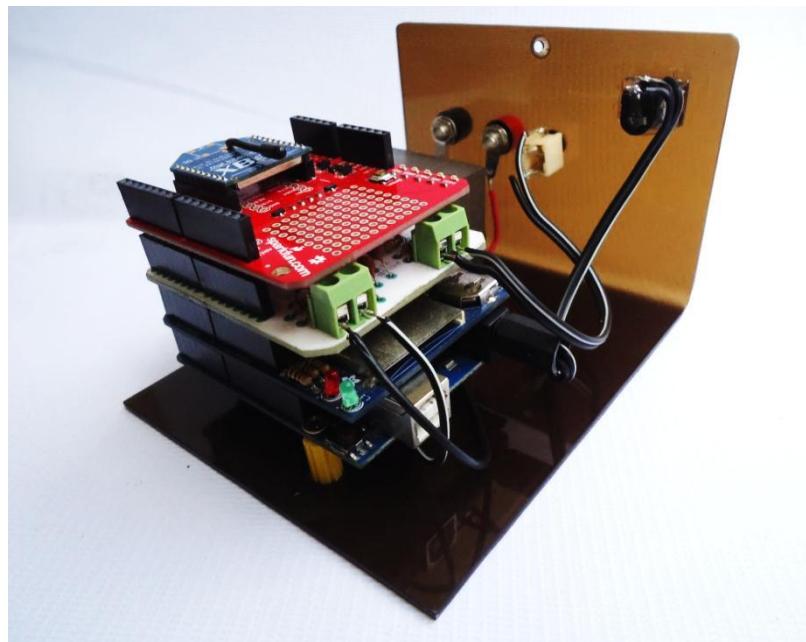


Figura 4.15 Montaje del módulo medidor en chasis.

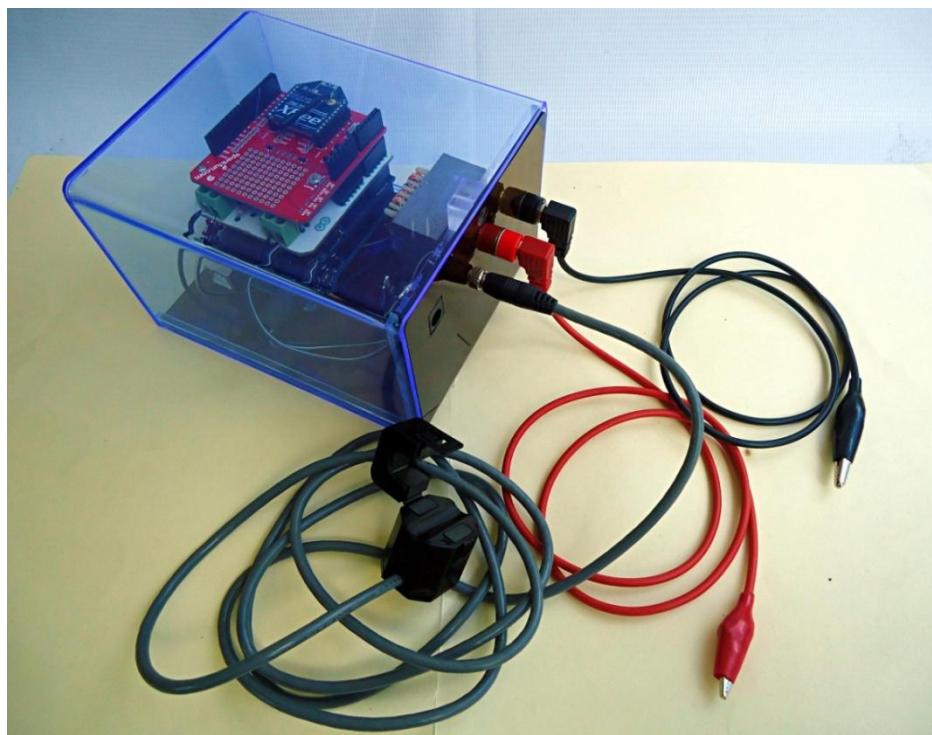


Figura 4.16 Medidor completo montado en chasis con accesorios de censado.

CAPÍTULO 5 SOFTWARE LABVIEW

INTERFAZ GRÁFICA LABVIEW

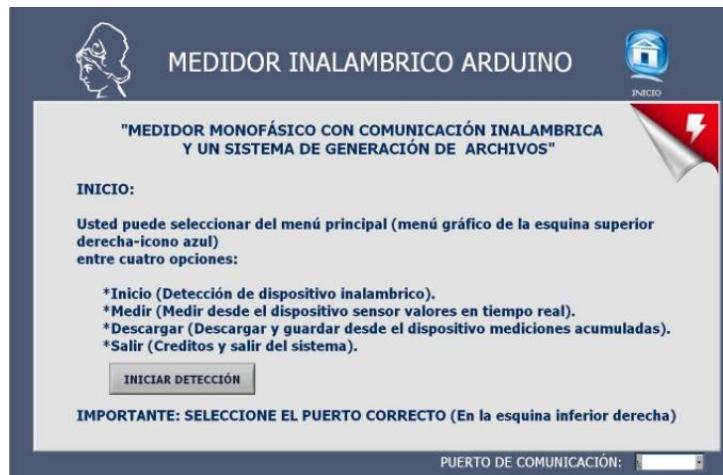


Figura 5.1 Ventana de inicio interfaz LabVIEW.

La interfaz gráfica en LabVIEW es uno de los componentes primordiales del proyecto siendo el medio principal de comunicación e interacción entre el usuario y el dispositivo.

El software diseñado en LabVIEW posee habilidades de comunicación inalámbrica con el dispositivo medidor, esto a través del módulo XBee conectado al puerto USB por medio del XBee Explorer Dongle.

Gráficamente la interfaz ha sido diseñada para brindar una experiencia agradable, con menús y cuadros de texto fácilmente legibles, explotando las cualidades de los elementos de presentación que posibilitan utilizar una misma área para presentar diferentes ventanas haciendo uso de selectores y botones, permitiendo optimizar el espacio y dar a la vez una apariencia y flujo de cuadros interesante.

El menú principal utiliza iconos gráficos haciendo la selección más intuitiva, y la mayoría de ventanas se presentan como pestañas invisibles para mostrar y ocultar grupos de controles e indicadores lo que brinda una mayor estética.

La interfaz permite al usuario una serie de opciones para tener control sobre las mediciones que se efectúan en el dispositivo medidor.

Para permitir al usuario interactuar con el dispositivo y navegar dentro de las diferentes opciones la interfaz dispone de un menú gráfico (ver figura 5.2) en el cual se puede seleccionar de entre 4 opciones diferentes:

1. INICIO: en esta ventana es posible iniciar la comunicación con el dispositivo por medio de una prueba de conectividad, que define si se está o no en el rango de alcance del medidor.
2. MEDIR: la segunda opción permite realizar mediciones en tiempo real, es decir el usuario puede visualizar las lecturas actuales en el momento que lo desee.
3. DESCARGAR: esta opción está diseñada para descargar el archivo en el cual se encuentran documentadas las lecturas realizadas desde la última vez que se borró el archivo.
4. SALIR: en esta ventana se muestran los principales datos y créditos de los desarrolladores, además de ser el menú de salida del sistema.



Figura 5.2 Menú gráfico interfaz.

INICIO

El menú inicio es una introducción a la interfaz, contiene una breve descripción de las opciones del menú y además cuenta con una aplicación que permite realizar una prueba de conectividad con el medidor.

Al conectar este dispositivo XBee Explorer de comunicación inalámbrica a la PC, inmediatamente se reconocerá como un puerto hábil²³, el primer paso para poder utilizar correctamente la interfaz es seleccionar el puerto correcto al cual se encuentra conectado el XBee Explorer, esto se realiza por medio del selector disponibles.



Figura 5.3 Selección del puerto del XBee Explorer Dongle.

DETECCIÓN DE DISPOSITIVO INALÁMBRICO

Una vez el XBee Explorer se haya conectado correctamente y se haya seleccionado el puerto USB adecuado, es posible utilizar la aplicación de detección, esto simplemente con presionar el botón “INICIAR DETECCIÓN” del menú inicio (ver figura 5.4).



Figura 5.4 Botón Iniciar detección del menú inicio.

La ventana de detección la cual se muestra en la figura 5.5 es una interfaz de espera mientras se establece comunicación con el medidor, en este momento el programa envía periódicamente un comando de reconocimiento al medidor Arduino, el cual al reconocer el comando enviará un

²³ Será necesario que la PC posea instalados los controladores (drivers) necesarios para una correcta comunicación con el XBee Explorer Dongle, estos controladores se encuentran disponibles en la página en línea de Digi X-CTU.

mensaje de respuesta a la interfaz y el programa deberá reconocer este comando de respuesta como válido para establecer que el dispositivo se encuentra dentro del rango de alcance.



Figura 5.5 Ejecución de la prueba de conectividad.

La ventana de espera pasa a un estado de aviso (ver figura 5.6) para que el usuario sepa que se está en rango de alcance para mantener la comunicación inalámbrica.



Figura 5.6 Dispositivo inalámbrico detectado.

El programa no siempre está detectando la presencia del medidor, así que es importante no alejarse del medidor una vez se esté realizando mediciones o descargando archivos.

Para poder completar esta operación de reconocimiento el programa tiene que enviar datos al medidor y leer datos desde él, para ello, como se describió en el capítulo 3 se hace uso de las funciones de la paleta NI-VISA.

BLOQUE BÁSICO DE COMUNICACIÓN SERIAL

En la figura 5.7 se muestra el bloque básico de lectura/escritura con el puerto USB, haciendo uso de las herramientas NI-VISA, este bloque se describe brevemente a continuación.

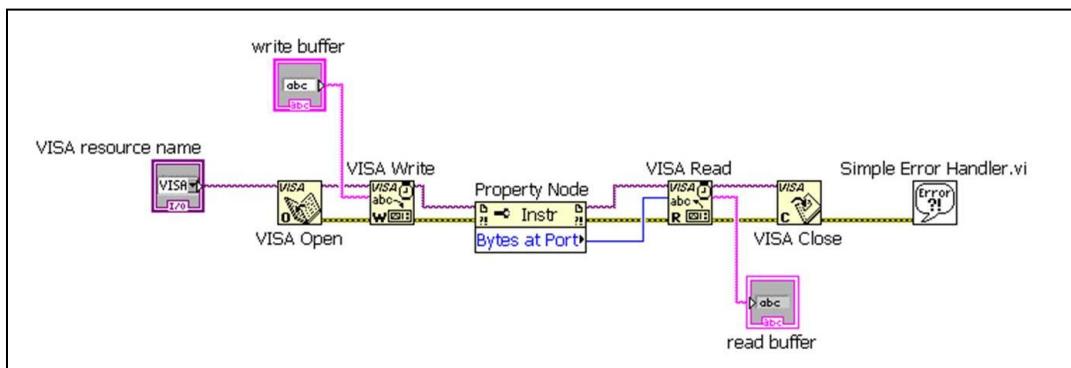


Figura 5.7 Bloque básico de comunicación serial.

VISA resource name: este es un control que permite seleccionar desde el panel frontal la dirección del puerto con el cual se desea establecer la comunicación, la apariencia de esta función en el panel frontal se muestra en la figura 5.3.

VISA Open: este SubVI primitivo habilita el flujo de datos o flujo de comunicación con el puerto seleccionado a través de “VISA resource name”.

Write Buffer: es un control que permite escribir una cadena de caracteres.

VISA Write: este SubVI permite escribir en el puerto seleccionado lo que sea que este escrito en el control “write buffer” el cual puede ser un control o una constante²⁴.

Property Node: los nodos de propiedades permiten obtener (leer) y/o setear (escribir) propiedades de una referencia, en este caso del puerto con el cual se tiene comunicación. También se puede utilizar el nodo de propiedad para acceder a los datos privados de una clase de LabVIEW.

El Nodo de Propiedad adapta automáticamente a la clase de objeto que hace referencia. LabVIEW incluye Nodos de Propiedad preconfigurados para acceder a las propiedades XML, propiedades, VISA, Propiedades de la conexión y las propiedades de ActiveX.

En la figura 5.4 el nodo de propiedad es utilizado para obtener la cantidad de bytes en el puerto serial, dato que es pasado a “VISA read” para que realice una lectura de esa dimensión.

VISA read: este SubVI posibilita la lectura del puerto seleccionado, es necesario en este caso especificar el tamaño de lo que se desea leer y el nombre del puerto del que se va a leer, los datos obtenidos de la lectura se envían al indicador “read buffer”.

Read Buffer: este indicador muestra en el panel frontal la lectura realizada por la función “VISA read”.

²⁴ Un control permite la interacción con el usuario desde el panel frontal mientras que una constante es solo editable en el diagrama de bloques.

VISA Close: pone fin al flujo de datos con el puerto especificado.

Línea transversal violeta: esta línea o cable transporta la dirección del puerto a través del flujo de interacción, esta línea es imprescindible para lograr completar con éxito dicha comunicación.

Línea transversal verde punteada: este cable detecta algún error generado en cualquier etapa de la comunicación y lo notifica a las siguientes etapas para evitar daños, además de notificarlo a través de un mensaje.

En consecuencia el protocolo de reconocimiento se basa en este bloque de comunicación serial (figura 5.7), como se mencionó antes el programa envía un comando al medidor escribiendo (con VISA write) en el puerto seleccionado (VISA resource name) este comando es una cadena de caracteres que el medidor reconoce y en consecuencia escribe en su puerto serial una cadena con una cadena de caracteres en respuesta, esta cadena debe ser leída por la interfaz (con VISA read) y discriminada como válida para determinar si la comunicación es posible, esta discriminación se realiza por medio de un algoritmo comparativo. El diagrama de bloques completo de esta sección se muestra en la figura 5.8.

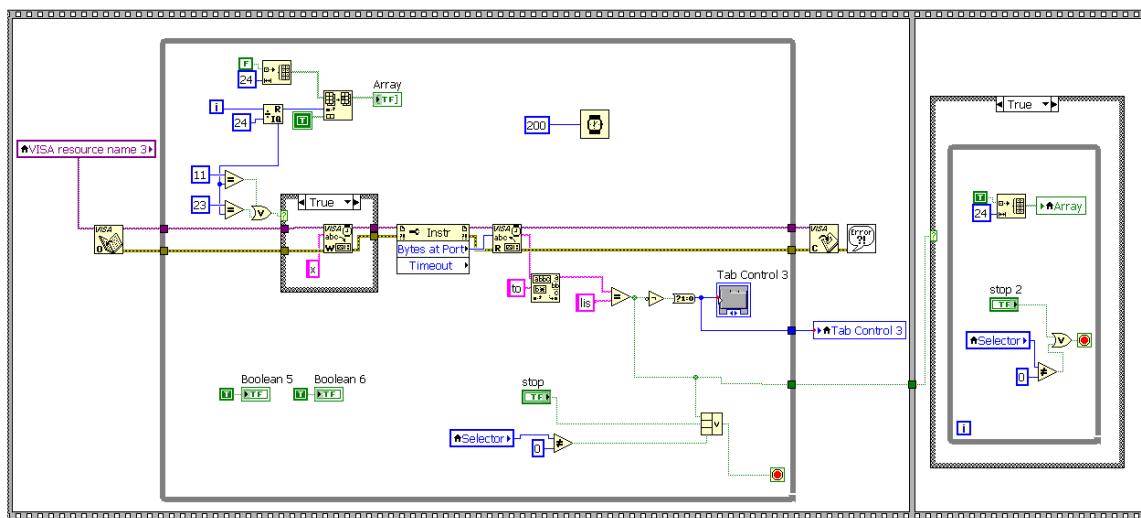


Figura 5.8 Diagrama de bloques correspondiente a la detección del dispositivo.

MEDICIÓN EN TIEMPO REAL



Figura 5.10 Menú de la ventana de medición en tiempo real.

Para desplegar cada una de las gráficas esta ventana posee un submenú selector el cual está en la parte superior de la gráfica (ver figura 5.10), desde este se puede seleccionar la gráfica que se desea visualizar, esto permite realizar un mejor aprovechamiento del espacio y observar mejor el comportamiento de cada variable.

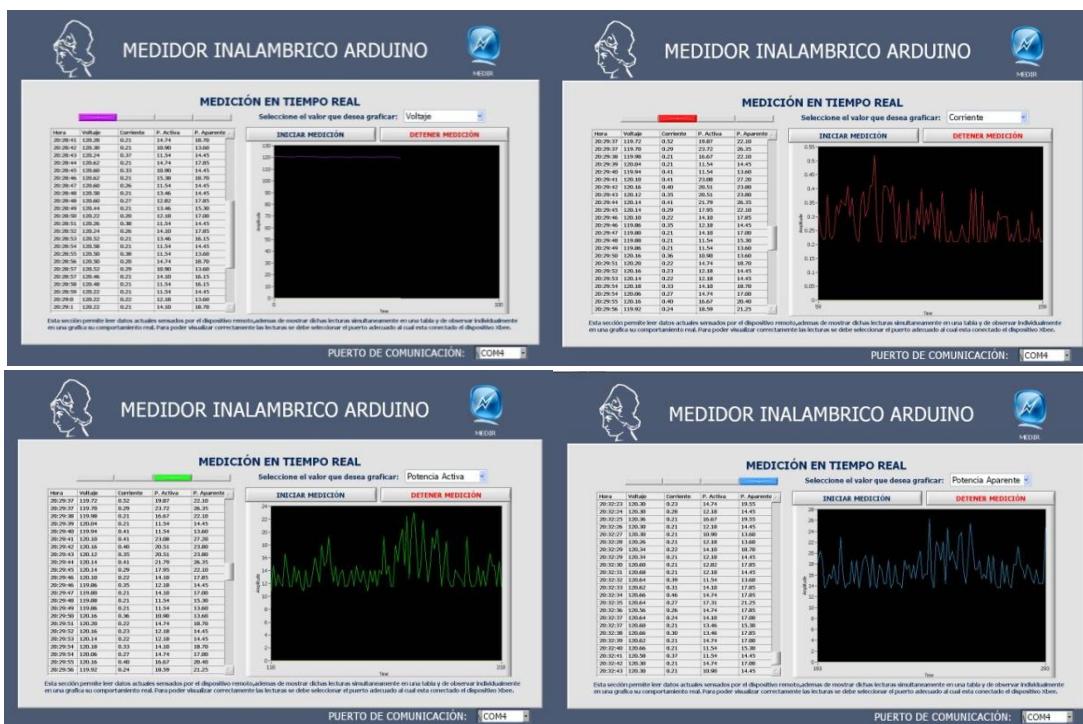


Figura 5.11 Graficas individuales de mediciones.

Al igual que en la aplicación de detección en este caso al presionar el botón “INICIAR MEDICIÓN” (figura 5.12) basado en el bloque de comunicación explicado en la sección anterior el programa enviará inalámbricamente un comando que el medidor deberá reconocer para entrar a un modo de funcionamiento específico en el cual el medidor además de realizar escrituras periódicas en la memoria, también realiza escrituras periódica de mediciones en el puerto serial, estas mediciones son leídas desde el programa en LabVIEW a través del XBee Explorer, una vez leídos estos datos el programa da formato a los datos para ser presentados de manera correcta en pantalla y poder ser graficados adecuadamente.



Figura 5.12 Botón iniciar y detener medición.

Los datos serán recibidos y graficados mientras se éste dentro de este modo de operación, para dar por terminado este modo basta con presionar el botón “DETENER MEDICÓN” (figura 5.12).

Para escribir los comandos y leer los datos se utilizan las mismas funciones expuestas anteriormente, pero ahora es importante realizar cada lectura en el momento indicado, cabe mencionar que este es uno de los momentos donde se pone de manifiesto una de las desventajas del método de comunicación empleado, y es que se debe tener una buena coordinación entre los instantes en que el medidor envía los datos al puerto y el instante en el que el programa los lee del puerto, para no realizar lectura dobles o que se den perdidas de datos, esta situación hace necesaria la implementación de un método de sincronización entre los dispositivos, a continuación se realiza una breve descripción del método utilizado.

SINCRONIZACIÓN DE LA COMUNICACIÓN

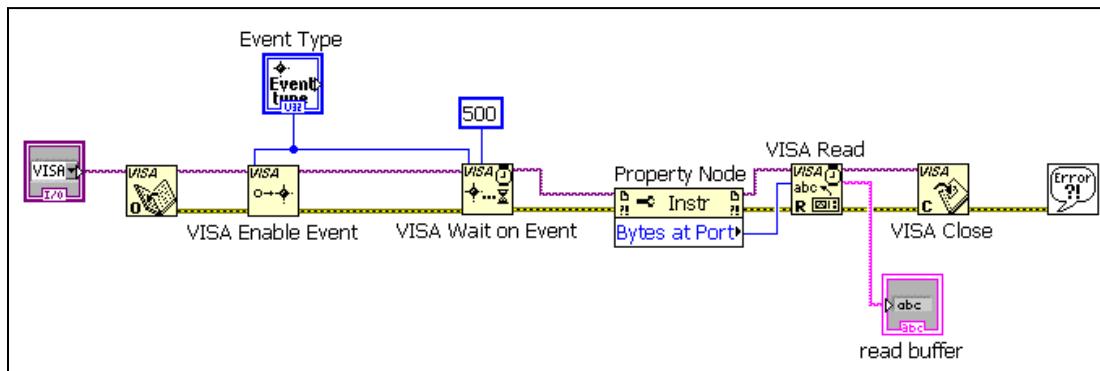


Figura 5.13 Bloque básico de lectura con implementación de sincronización.

Para la sincronización se utilizó la función “VISA wait on event”:

VISA wait on event: Suspende la ejecución de un subproceso de aplicación y espera a que un tipo de evento específico ocurra dentro de un período de tiempo no superior a la especificada por el tiempo de espera, este VI necesita de tres entradas obligatorias: el nombre del puerto, el tipo de evento, y el tiempo de espera en milisegundos (time out). Además es importante que el flujo de datos creado sea capaz de reconocer el tipo de evento especificado, de esto se encarga “VISA Enable Event”.

VISA Enable Event: Permite la notificación de un tipo de evento concreto.

Event type: es el identificador de evento lógico. Puede elegir entre los siguientes tipos de eventos VISA.

Tabla 5.1 Tipos de eventos.

Service Request (default)	0x3FFF200B
Trigger	0xBFFF200A
Clear	0x3FFF200D
VXI Signal	0x3FFF2020

Service Request (default)	0x3FFF200B
VXI/VME Interrupt	0xBFFF2021
VXI/VME Sysfail	0x3FFF201D
VXI/VME Sysreset	0x3FFF201E
GPIB CIC	0x3FFF2012
GPIB Talk	0x3FFF2013
GPIB Listen	0x3FFF2014
PXI Interrupt	0x3FFF2022
Serial Break	0x3FFF2023
Serial TermChar	0x3FFF2024
Serial CTS	0x3FFF2029
Serial DSR	0x3FFF202A
Serial DCD	0x3FFF202C
Serial RI	0x3FFF202E
Serial Character	0x3FFF2035
USB Interrupt	0x3FFF2037
Todos habilitados	0x3FFF7FFF

En resumen la lectura del siguiente dato no se realizará sino hasta que este esté listo, esto permite a la interfaz la habilidad de adaptarse al ritmo con que el medidor envíe los datos sin riesgo a que haya pérdidas o dobles lecturas.

ACONDICIONAMIENTO DE DATOS

Los datos recibidos por LabVIEW no están ordenados de manera adecuada para ser presentados en pantalla mediante una tabla ni para ser graficados, ya que lo que se obtiene es una sola cadena de caracteres con todos los datos necesarios de una lectura, por lo que se vuelve necesario implementar un bloque de acondicionamiento de datos, este se muestra en la figura 5.13.

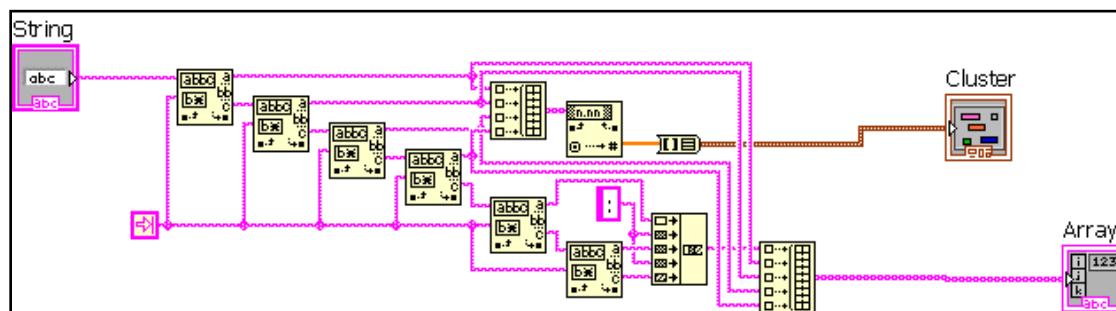


Figura 5.13 Diagrama de bloques SubVI “ConvCadena(SubVI)”.

ConvCadena(SubVI):

Este SubVI convierte una cadena de caracteres de la forma: "Voltaje \t Corriente \t Pot. Activa \t Pot. Aparente \t Hora \t minutos \t segundos" a un cluster o grupo de datos numéricos que están listos para ser presentados en una gráfica o ser utilizados para cualquier otro cálculo.

Además construye un arreglo de cadenas de caracteres de la forma: "Hora:minutos:segundos; Voltaje; Corriente; P.Activa; P.Reactiva" dicho arreglo se puede pasar directamente ahora a una tabla indicador para visualizar los datos en pantalla.

OPTIMIZANDO ESPACIO

Son cuatro valores que pueden ser graficados, para evitar tener cuatro gráficas pequeñas en pantalla se utiliza una sola gráfica de forma de onda la vez, no es conveniente colocar varias gráficas al mismo tiempo por las diferencias de los valores, es decir mientras que los valores de amperaje son normalmente menores que 1A los valores de tensión son normalmente alrededor de 120V.

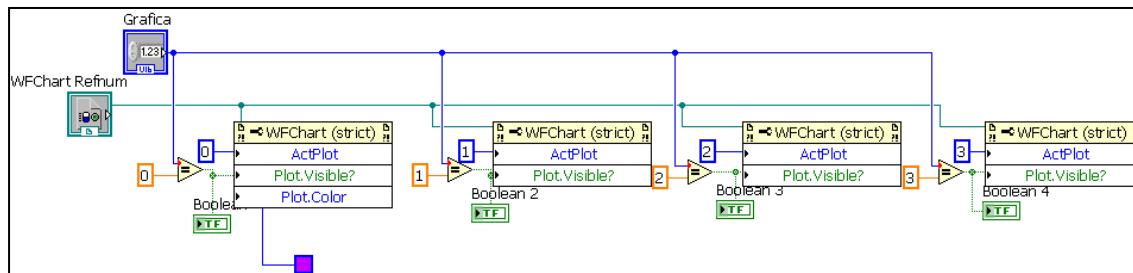


Figura 5.14 Diagrama de bloques SubVI "GraficaSelector(SubVI)".

En la figura 5.14 se muestra el diagrama de bloques correspondiente al SubVI encargado de gestionar al indicador de la gráfica, este se encarga de seleccionar qué valor será el que se mostrará como visible en la ventana, además desde aquí se podrían gestionar las características principales de la gráfica como el color las escalas el fondo, etc.

Finalmente en la figura 5.15 se presenta el diagrama de bloques del lazo de lectura y presentación de datos en tiempo real, aquí se muestran las etapas de sincronización y lectura, de acondicionamiento de datos y de gestión de gráfica, completando así las funciones de la etapa de medición, aquí simplemente no se muestra la etapa del envío de los comandos de inicio y finalización del modo de medición.

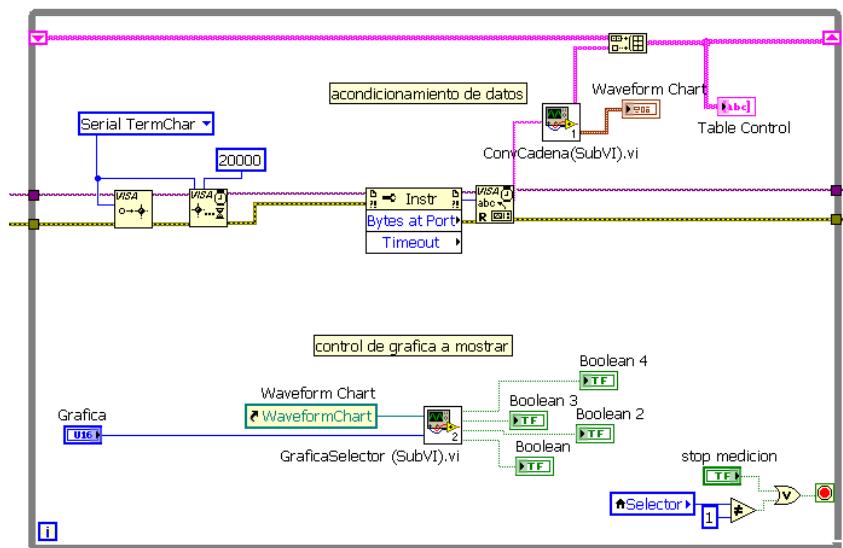


Figura 5.15 Diagrama de bloques del lazo de lectura y presentación de datos en tiempo real.

DESCARAS DE MEDICIONES

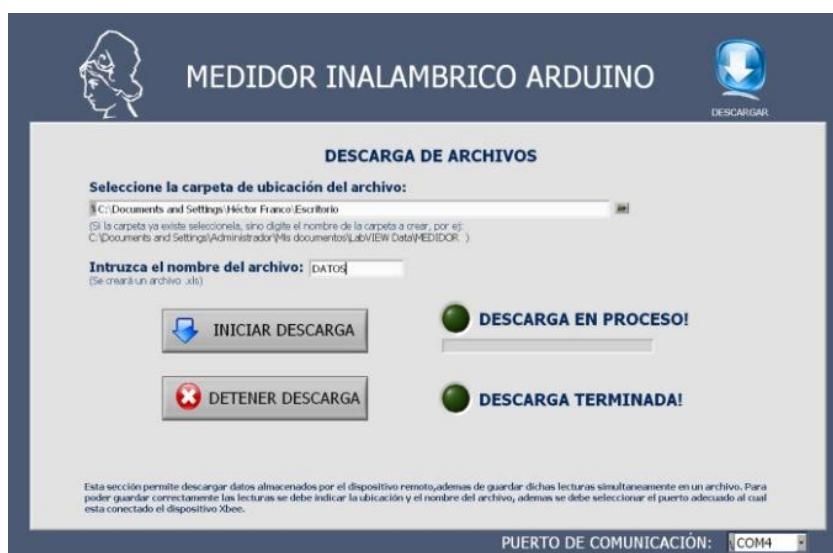


Figura 5.16 Ventana principal del menú de descarga.

El medidor ha sido diseñado principalmente para documentar mediciones de consumo en un dispositivo de memoria localizado en el hardware del medidor, esta opción está diseñada para descargar el archivo en el cual se encuentran documentadas las lecturas realizadas desde la última vez que se borró el archivo.

Para descargar correctamente el archivo es imperativo seguir los pasos siguientes:

1. Haber seleccionado el puerto correcto en el control “I/O VISA” (ver figura 5.3).
2. Seleccionar la carpeta en la cual se desea guardar el archivo (ver figura 5.17 y 5.18).



Figura 5.17 Seleccionar dirección de carpeta de descarga.

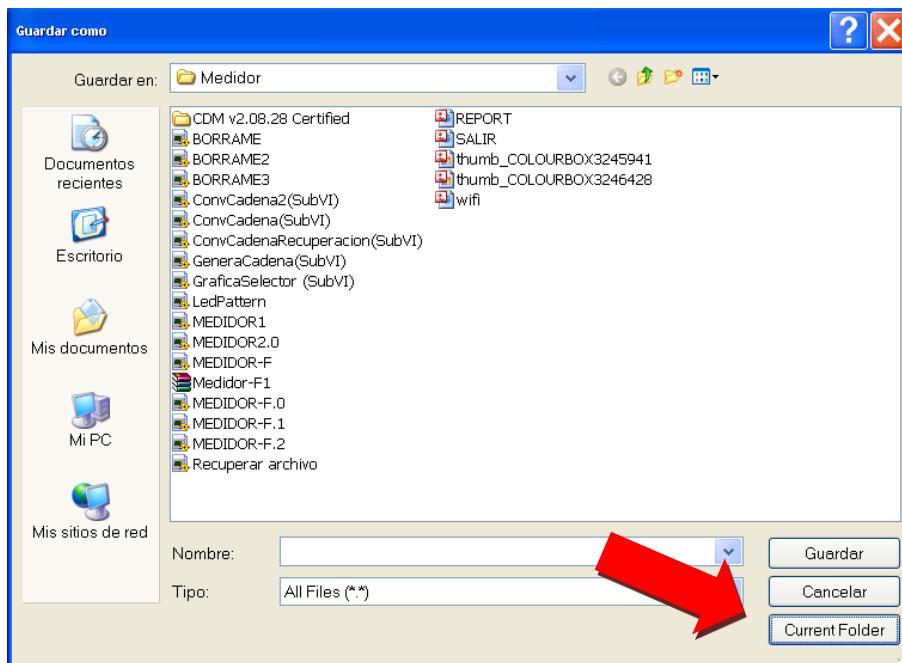


Figura 5.18 Seleccionar carpeta de descarga.

3. Escribir el nombre del archivo a crear (ver figura 5.19).



Figura 5.19 Escritura del nombre del archivo a crear.

4. Iniciar la descarga haciendo clic en el botón “INICIAR DESCARGA”.

Mientras la descarga este en proceso un indicador led estará activo y además el flujo de datos es visibles mediante el indicador serial ubicado bajo el indicador de descarga activa (ver figura 5.20), al finalizar la descarga correctamente el indicador serial debería mostrar el mensaje “outm2” y el led “Descarga terminada” debe estar activo (ver figura 5.21). Además es posible interrumpir la descarga en cualquier momento haciendo clic en el botón “DETENER DESCARGA”.



Figura 5.20 Descarga en proceso.



Figura 5.21 Descarga terminada.

Para poder realizar la descarga el medidor necesita entrara a un modo de funcionamiento especial entonces el proceso se realiza de manera similar a las otras opciones: el programa en LabVIEW envía al medidor un comando para notificar que se desea descargar los datos, el dispositivo entonces enviara una lectura a la vez hasta completar el 100% de los datos y al finalizar el proceso enviara un comando a la interfaz para notificar que los datos se han descargado por completo, la interfaz debe validar este comando y entonces detendrá las lectura del puerto, avisando al usuario que la descarga se ha finalizado.

Los datos recibidos son a una frecuencia de envío diferente que en la sección de medición pero la recepción correcta de los datos se controla con el mismo proceso de sincronización descrito anteriormente y el proceso de lectura y escritura se mantiene como en el bloque de comunicación descrito.

ACONDICIONAMIENTO DE DATOS

En este menú también se reciben los datos de una forma ordenada pero no adecuada para una documentación ordenada, nuevamente los datos son recibidos como una sola cadeana de caracteres, la cual debe ser desglosada y ordenada de la manera en que se dese guardar los datos. Este proceso se realiza mediante el SubVI “ConvCadena2(SubVI)” cuyo diagrama de bloques se muestra en la figura 5.22.

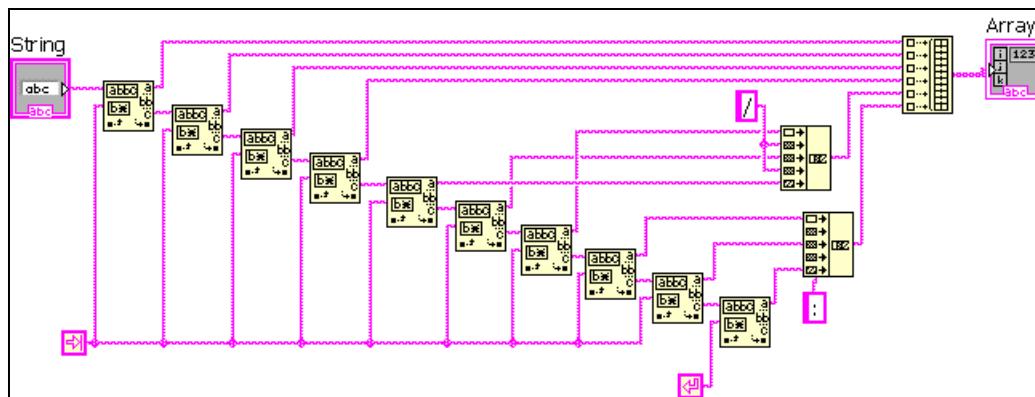


Figura 5.22 Diagrama de bloques del SubVI “ConvCadena2(SubVI)”.

El diagrama de bloques parcial de proceso de descarga se muestra en la figura 5.23, aquí se observa las etapas de sincronización, lectura, acondicionamiento de datos, validación de la finalización del proceso, activación y desactivación de los led indicadores y la formación del arreglo que será guardado en el archivo mediante el SubVI “Write To Spreadsheet File.VI”.

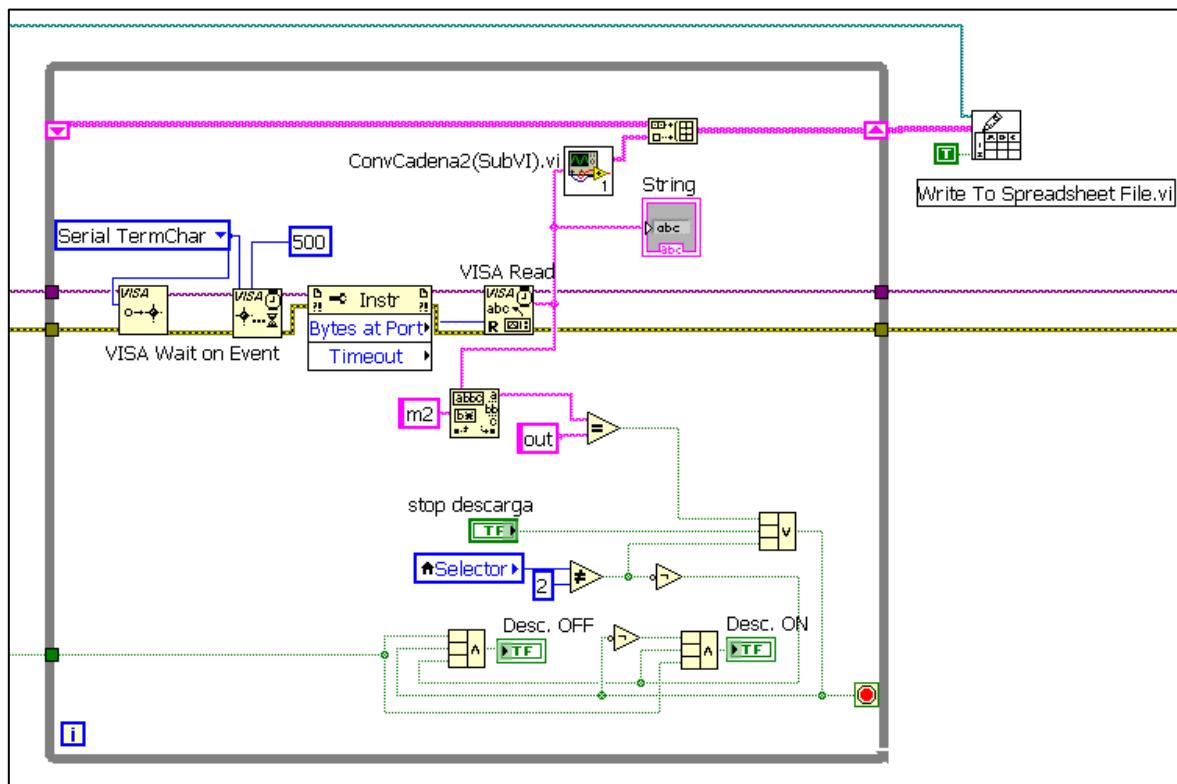


Figura 5.23 Diagrama de bloques del proceso de descarga.

Write To Spreadsheet File: Convierte una matriz 2D o 1D de cadenas, números enteros con signo, o números de doble precisión a una cadena de texto y escribe la cadena en un archivo nuevo o anexa la cadena a un archivo existente.

A este SubVI se le debe especificar el path (dirección completa del archivo) cable color celeste (ver figura 5.23), la matriz a escribir, además de poder seleccionar otras opciones como escribir al final de una archivo existente o crear un nuevo archivo o transponer en el archivo la matriz indicada y el formato con que se desea guardar los datos.

SALIR

Este menú permite finalizar y cerrar el programa.



Figura 5.24 Venta del menú salir.

CAPITULO 6: CÓDIGO ARDUINO, CONTROL DEL CIRCUITO INTEGRADO.

DETALLE DEL CÓDIGO: RUTINAS DE LECTURA, FUNCIONES MÁS IMPORTANTES, MODOS DE OPERACIÓN.

En la siguiente porción de código vemos las rutinas de lectura y escritura básicas para registros de tamaño de 8 bits, lo primero que se hace es capturar en el mismo byte la dirección del registro que debe ser un parámetro a enviar dentro de la función y además asignarle los primeros dos bits de identificación del tipo de operación (lectura o escritura) es por eso que se hace la suma lógica con reg OR leer, reg OR escribir. Además cuando ya se está listo con la información a transferir se selecciona al integrado con un csbajo (activo con un bajo) se esperan 5 microsegundos y se utiliza el comando `<SPI.transfer>` este es el que manda haciendo uso del protocolo SPI el byte correspondiente indicado dentro de sus parámetros. Despues viene una espera de 5 microsegundos (se escoge un sol tiempo para no confundirse de los tiempos de la hoja de datos, el mayor tiempo con algo de sobra), la siguiente línea es para capturar y retornar el valor contenido en el registro del integrado. Luego se desactiva el chip.

```
unsigned char leer8(char reg){  
    reg = reg | leer;  
    csbajo();  
    delayMicroseconds(5);  
    SPI.transfer(reg);  
    delayMicroseconds(5);  
    return (unsigned char)SPI.transfer(0x00);//cambiar por unsigned byte  
    csalto();  
}
```

Al igual que la porción de código anterior, la mayoría de comandos se repiten excepto porque ahora es una escritura y justo después de realizar la primera transferencia obligatoria, debemos seguir enviando los bytes que queremos separados por tiempos de 5 microsegundos.

```
void escribir8(char reg, char data){  
    reg = reg | escribir;  
    csabajo();  
    delayMicroseconds(10);  
    SPI.transfer((unsigned char)reg);      //register selection  
    delayMicroseconds(5);  
    SPI.transfer((unsigned char)data);  
    delayMicroseconds(5);  
    csalto();  
}
```

MODOS DE OPERACIÓN.

Buscando diferentes funcionalidades el usuario dispone de algunas opciones de monitoreo de energía, estos diferentes opciones se han separado en modos que significan las diferentes formas en que podemos hacer trabajar al medidor, o en su defecto las diferentes órdenes que podemos darle de forma remota e inalámbrica.

EL MODO UNO O DE MONITOREO EN TIEMPO REAL

El bloque de código correspondiente a este modo es el siguiente:

```
//obteniendo potencia activa  
long pot_act=get_pot_act();//potencia activa  
pot_act=pot_act<<8;  
pot_act=pot_act>>8;  
float pot_act1=pot_act*1.5;  
//obteniendo potencia aparente  
long pot_aparente=get_pot_aparente();  
pot_aparente=pot_aparente<<8;  
pot_aparente=pot_aparente>>8;  
float pot_aparente1=pot_aparente*1.8;  
long voltaje_rms=vrms();  
float voltaje_rms_corregido= voltaje_rms*0.000264585;  
long corriente_irms=irms();  
corriente_irms=corriente_irms-1400; //para corregir offset.  
float corriente_irms_corregido=corriente_irms*0.000048575;;  
Serial.print(voltaje_rms_corregido);  
Serial.print("\t");  
Serial.print(corriente_irms_corregido);  
Serial.print("\t");  
Serial.print(pot_act1);  
Serial.print("\t");  
Serial.print(pot_aparente1);  
Serial.print("\t");  
DateTime now = RTC.now(); //esto solo es para la rtc  
String tmpmodo1="";  
tmpmodo1=String(now.hour(),DEC)+"\t"+String(now.minute(),DEC)+"\t"+String(now.second(),DEC);
```

```

Serial.println(tmpmodo1);
int segundos=now.second();
int multiplos=segundos%5;
long tiempo=millis();
long intervalo=tiempo-lasttiempo;
if (multiplos==0 & intervalo>4000 ){
    //cssdbajo();
    guardardatos(voltaje_rms_corregido, corriente_irms_corregido, pot_act1, pot_ap1);
    //cssdalto();
    lasttiempo=tiempo;
}

```

El código comienza haciendo lecturas a los registros que guardan la acumulación de energía y haciendo rotaciones para superar el hecho que los registros son signados y poder conservar el signo cuando este se mueve a una variable temporal del Arduino que es de 32 bits, 8 bits más grande que el mayor de los registros del ADE7753. Luego se hacen las respectivas lecturas a los registros que guardan las mediciones RMS, cuando ya se han almacenado temporalmente las variables se mandan a escribir en el puerto serial junto con la hora que proporciona la RTC, mientras se encuentra en el modo uno, los datos se están almacenando en la datalogger con diferente sincronización, en bloque de código anterior por ejemplo se decidió almacenar los datos cada cinco segundos. Esa es la función que tiene el código siguiente, los 4000, significan 4 segundos, y están restringiendo que el tiempo de guardado sea solo cuando desde la última vez que se guardó hayan transcurrido más de 4 segundos y el dato a guardar sea múltiplo de 5 segundos.

```

if (multiplos==0 & intervalo>4000 ){
    //cssdbajo();
    guardardatos(voltaje_rms_corregido, corriente_irms_corregido, pot_act1, pot_ap1);
    //cssdalto();
    lasttiempo=tiempo;
}

```

MODO DOS O DE DESCARGA DE LOS DATOS ALMACENADOS

El código que permite esto es el siguiente:

```

long energia = get_energia_consumida();
energia=energia<<8;
energia=energia>>8;
float energia2 = energia*0.64;

SPI.setDataMode(SPI_MODE0);
File miarchivo = SD.open("data.txt");
//Serial.println(dataString);
if (miarchivo) {
    while (miarchivo.available()) {
        Serial.write(miarchivo.read());
    }
}

```

```
miarchivo.close();

}

// si el archivo no se abrio, muestra un error
else {
    Serial.println("error abriendo archivo de texto");
}
SPI.setDataMode(SPI_MODE2);
Serial.print("Energia\t");
Serial.println(energia2);
Serial.print("outm2 \n \n \n \n \n ");
}
```

Este código segmento de código es importantísimo, en este tiene lugar la lectura del consumo total realizado, se inicia precisamente haciendo una lectura al registro acumulador de energía que se encuentra acumulando energía desde que el medidor fue energizado. Este procedimiento tiene lugar dentro de este modo precisamente porque la información de la energía consumida total nos interesa que valla en cada descarga de datos que se realice.

Luego lo que hay en este bloque de código es un cambio en la configuración de la comunicación SPI, y los respectivos comandos para la datalogger de volcado o lectura completa del archivo donde se está almacenando la información.

MODO CINCO COMPROBACIÓN DE ALCANCE

Era necesario tener un herramienta que permitiera comprobar si el dispositivo está dentro del alcance antes de iniciar cualquier operación por tanto se incluye la dentro de los modos disponibles para el usuario.

```
//la siguiente con es para ver si se encuentra dentro del rango de cobertura
else if (letra =='x'){
    Serial.print("listo");
}
```

Los modos anteriores están disponibles en la interfaz gráfica para el usuario promedio, pero hay otras funcionalidades importantes que fueron consideradas como no muy convenientes para que cualquier usuario disponga, otras no fueron puestas por que no se alcanzaron a desarrollar lo suficiente como para ponerlas en la interfaz de usuario. Todos los modos de operación pueden ser manipulados remotamente.

MODO CUATRO O DE BORRADO DE ARCHIVO

Se vuelve útil poder hacer un borrado remoto del archivo sobre el cual se están escribiendo los datos, de modo que en las próximas descargas de información solo se encuentren los datos nuevos.

El comando que permite esto es <SD.remove>.

```
//la siguiente condicion sirve para borrar archivo cuando se desee
else if (letra =='s'){
    SPI.setDataMode(SPI_MODE0);
    SD.remove("data.txt");
    if (SD.exists("data.txt")){
        Serial.println("el archivo existe");
    }
    else {
        //delay(5000);
        Serial.print("El archivo fue borrado");

    }
    SPI.setDataMode(SPI_MODE2);
}
```

MODO CINCO O DE RESETEADO DE INTEGRADO.

```
//la siguiente condicion sirve para resetear el ADE7753
else if (letra =='r'){
    setMode(76);
    delay(3);
    setMode(12);
    delay(1);
    Serial.print("El ADE7753 fue reseteado");
}
```

El comando que hace posible el reset del IC ADE7753 es <setMode(76)> este decimal es la palabra binaria que se debe escribir para activar el bit correspondiente de reset que está dentro del registro de modo.

MODO SEIS OBTENCIÓN DE LA ENERGÍA CONSUMIDA TOTAL

```
//obteniendo energia consumida
long energia = get_energia_consumida();
energia=energia<<8;
energia=energia>>8;
float energia2 = energia*0.64;
Serial.println(energia2);
```

Se incluye este modo que no está disponible para el usuario pero consiste en monitorear la energía total consumida desde que el medidor fue energizado. La función <get_energia_consumida()> hace una lectura del registro AENEKY del ADE7753 que es un registro

de acumulación y nunca se borra a menos que ocurra una des energización o se haga un reset al integrado.

CAPÍTULO 7: RESULTADOS

En este capítulo se presentan los resultados de una prueba realizada con el medidor, dicha prueba consistió en censar el consumo de una carga de prueba durante dos días a una frecuencia de documentación de un dato cada cinco segundos. Cabe recordar que este dato no es una lectura instantánea sino que es un valor resultante del proceso de cálculo realizado por el ADE7753.

CARGA DE PRUEBA

La carga de prueba fue una refrigeradora cuyos datos técnicos se detallan en la figura 7.1

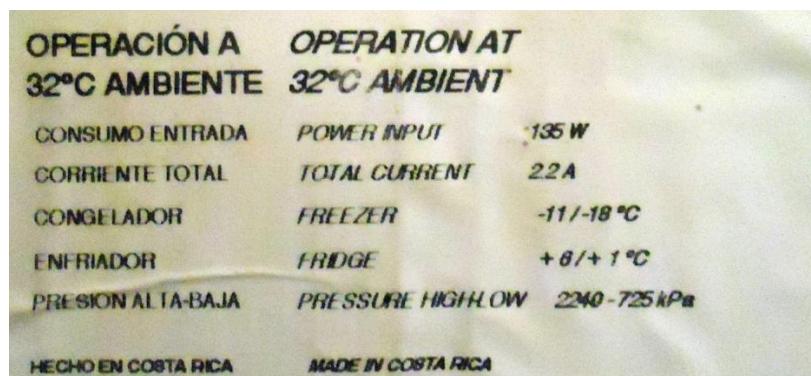


Figura 7.1 Foto de los datos técnicos de la refrigeradora (Carga de prueba).

Tabla 7.1 Resumen datos técnicos de la carga de prueba.

Dato técnico	Valor
Consumo de entrada	135 W
Corriente total	2.2 A

MEDICIONES

En total se documentaron 34,323 datos capturados cada 5 segundos la tabla 7.1 presenta una muestra de los datos documentados por el medidor, durante los dos días de prueba, estos solo representan una pequeña parte de los 34,323 datos documentados.

Tabla 7.2 Muestra de datos colectados por el medidor.

Voltaje [V]	Corriente [A]	Potencia Activa [W]	Potencia Aparente [VA]	Fecha [dd/mm/ hh]	Hora [hor:min:seg]
118.9	2	144.87	199.75	22/06/2013	20:13:40
118.92	2	144.22	199.75	22/06/2013	20:13:45
118.88	1.99	144.22	198.9	22/06/2013	20:13:50
118.86	1.99	144.22	198.9	22/06/2013	20:13:55
118.9	1.99	143.58	198.9	22/06/2013	20:14:00

Voltaje [V]	Corriente [A]	Potencia Activa [W]	Potencia Aparente [VA]	Fecha [dd/mm/hh]	Hora [hor:min:seg]
118.9	1.99	143.58	198.9	22/06/2013	20:14:05
118.88	1.99	143.58	198.9	22/06/2013	20:14:10
118.85	1.99	142.94	198.9	22/06/2013	20:14:15
118.88	1.99	143.58	198.05	22/06/2013	20:14:20
118.88	1.99	142.94	198.05	22/06/2013	20:14:25
118.86	1.98	142.94	198.05	22/06/2013	20:14:30
118.94	1.98	143.58	198.05	22/06/2013	20:14:35
118.92	1.98	142.94	198.05	22/06/2013	20:14:40
118.88	1.98	142.3	198.05	22/06/2013	20:14:45
118.91	1.98	142.94	197.2	22/06/2013	20:14:50
118.91	1.98	142.94	198.05	22/06/2013	20:14:55
118.88	1.98	142.94	197.2	22/06/2013	20:15:00
118.88	1.98	142.94	197.2	22/06/2013	20:15:05
118.9	1.97	142.3	198.05	22/06/2013	20:15:10
118.96	1.97	141.66	197.2	22/06/2013	20:15:15
118.88	1.97	141.66	197.2	22/06/2013	20:15:20
118.89	1.97	142.3	196.35	22/06/2013	20:15:25
118.9	1.97	142.3	197.2	22/06/2013	20:15:30
118.87	1.97	142.3	196.35	22/06/2013	20:15:35
118.92	1.97	141.66	197.2	22/06/2013	20:15:40
118.93	1.97	141.66	196.35	22/06/2013	20:15:45
118.89	1.97	141.66	196.35	22/06/2013	20:15:50
118.9	1.97	141.66	196.35	22/06/2013	20:15:55
118.89	1.97	141.02	196.35	22/06/2013	20:16:00
118.88	1.96	141.66	196.35	22/06/2013	20:16:05
118.92	1.96	141.02	196.35	22/06/2013	20:16:10
118.91	1.96	141.02	196.35	22/06/2013	20:16:15
119.02	1.96	140.38	195.5	22/06/2013	20:16:20
119.02	1.96	140.38	196.35	22/06/2013	20:16:25
119.04	1.96	140.38	196.35	22/06/2013	20:16:30
119.02	1.96	141.02	195.5	22/06/2013	20:16:35
119.05	1.96	141.02	196.35	22/06/2013	20:16:40

GRÁFICAS

Del total de datos capturados se presentan en las imágenes 7.2 a 7.5 las gráfica de las lecturas obtenidas. Estas presentan el comportamiento de los valores de tensión, corriente, potencia activa y potencia aparente contra tiempo en horas,

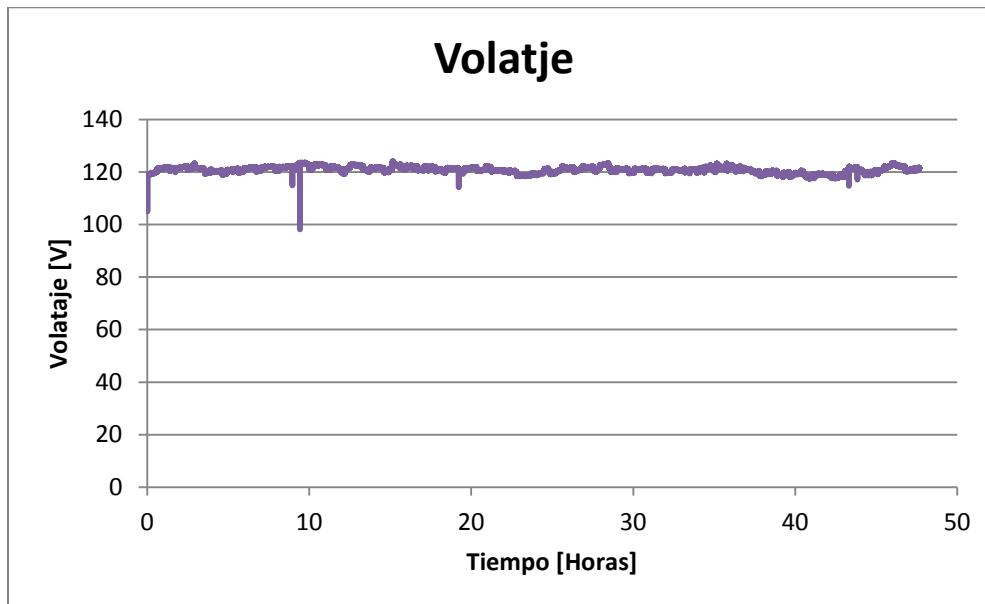


Figura 7.2 Gráfico de Voltaje vs tiempo.

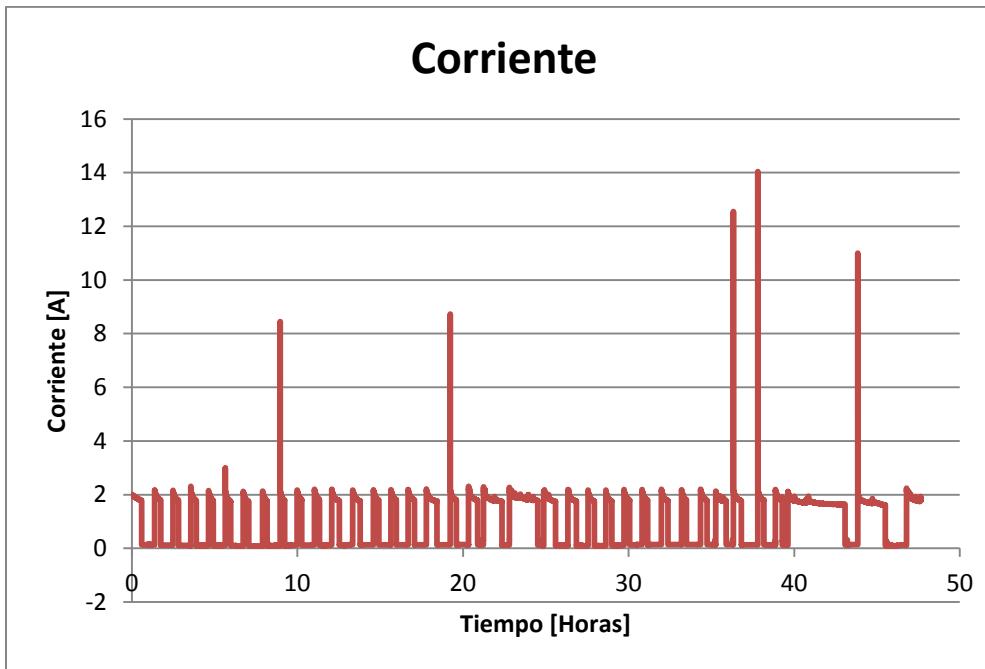


Figura 7.3 Gráfico de Corriente vs tiempo.

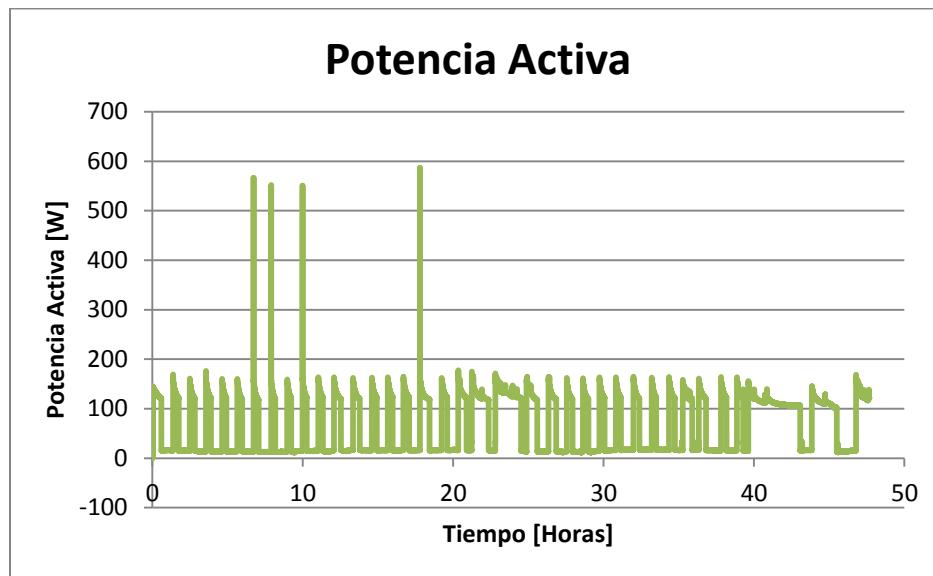


Figura 7.4 Gráfico de Potencia Activa vs tiempo.

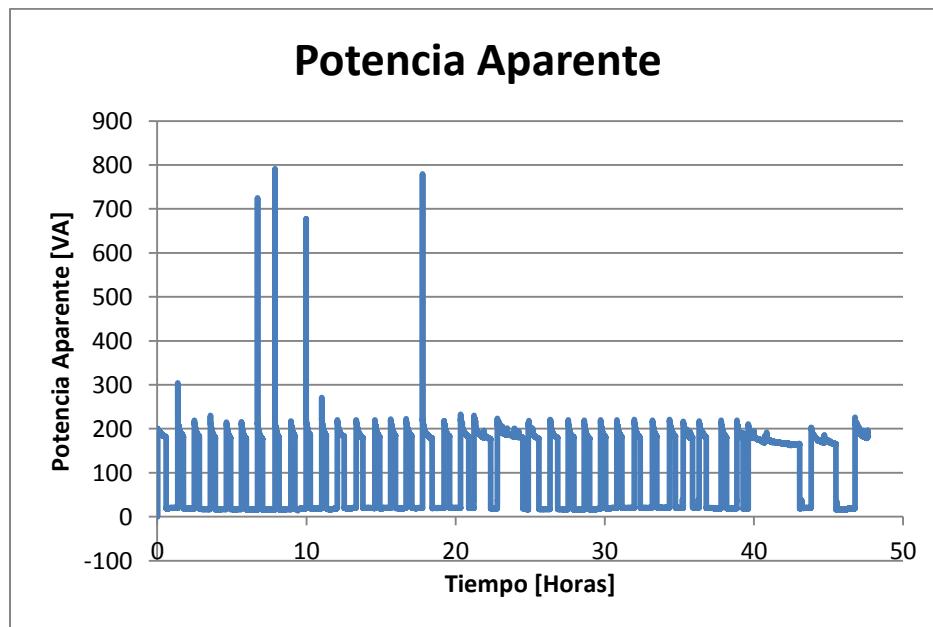


Figura 7.5 Gráfico de Potencia Aparente vs tiempo.

CONCLUSIONES

- ✚ En síntesis, el medidor de energía diseñado y construido a lo largo de este proyecto permite realizar mediciones de valores rms de tensión y corriente eléctrica, además de valores de potencia activa y aparente junto con la energía total consumida durante un periodo de tiempo determinado; el usuario es capaz de acceder a estos valores de manera inalámbrica a través de una interfaz tanto en tiempo real como a datos documentados en la memoria del medidor, estos datos almacenados son importados a un archivo de reporte que contiene las mediciones y la energía total consumida en W/h. Cumpliendo de esta manera los objetivos planteados al inicio del proyecto.
- ✚ Para alcanzar exitosamente los puntos clave del proyecto, utilizando al medidor de energía monofásico ADE7753 como eje principal, se concluyó que el hardware y software apropiados para lograr el control sobre el medidor, sería mediante la integración de la plataforma de hardware Arduino, la cual posee un protocolo de comunicación compatible con el ADE7753, para adquirir la capacidad de comunicación inalámbrica se utilizan dispositivos XBee cuyas características permiten el flujo de datos entre el microcontrolador Arduino y el puerto USB de la PC de forma transparente, para permitir al usuario la interacción con el medidor se seleccionó el software LabVIEW de National Instruments que permite controlar el puerto USB para procesar y mostrar resultados de las mediciones, mientras que para almacenar de forma autónoma las mediciones, se utiliza una memoria SD y un RTC de respaldo que proporciona la hora y fecha para cada lectura. Superando de manera exitosa los requerimientos principales trazados a priori para el proyecto.

ANEXOS

ANEXO 1

EJEMPLO1: CONTROL BASICO DEL ADE7753.

Generalidades: El ADE7753 es un IC medidor de energía que puede ser utilizado para obtener información sobre valores RMS de las señales de corriente y voltaje y valores de potencia y energía.

Componentes necesarios:

- ✚ Un Arduino.
- ✚ Un hardware controlador del ADE7753, (shield del medidor para el Arduino).
- ✚ Pinza de corriente 30.
- ✚ Transformador 120/12 V.

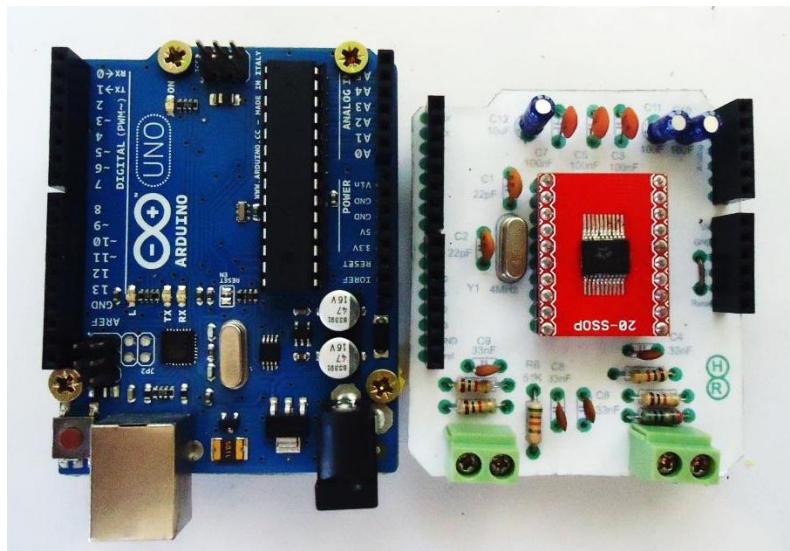


Figura A1. Arduino y la shield del medidor para Arduino usando el ADE7753.



Figura A2. Pinza de corriente y transformador para censar corriente y voltaje respectivamente.

ADE7753 INTERFACE SERIAL.

Todas las funcionalidades del ADE7753 son accesibles a través de muchos registros que ya están dentro del chip, ver figura A3. Los contenidos de estos registros pueden ser actualizados o leídos usando la interface serial del chip. Después de encender, mandar a RESET un pulso bajo o en el borde de bajada de *cs*. El ADE7753 es puesto en modo de comunicación; en este modo el IC espera que se escriba en su registro de comunicación. El dato escrito en el registro de comunicación determina si la siguiente operación es una lectura o una escritura y además indica la dirección del registro a acceder. Entonces todas las transferencias de datos no importando si es lectura o escritura deben iniciar con una escritura en el registro de comunicaciones.

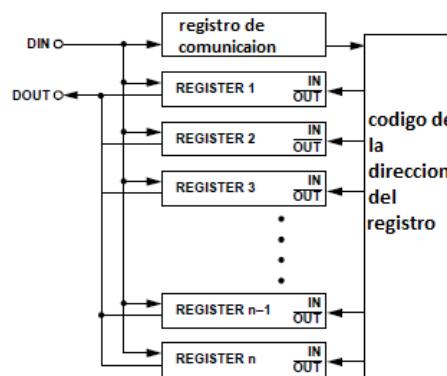


Figura A3. Direccionando los registros del ADE7753 vía registro de comunicación.

Para la conexión se debe tener en cuenta solamente que la pinza de corriente debe de llegar donde se encuentra la resistencia de $10\ \Omega$ dentro de la shield.

Configuración básica del ADE7753.

```

//declarando librerias a utilizar
#include <SPI.h>
//declarando algunas constantes principales
const int cs=9; //debe ser este
const byte leer=0b00000000;
const byte escribir=0b10000000;
    
```

En la primera parte del código se inicia la librería necesaria para trabajar con el integrado ñuego se inicializan unas constantes como el selector de chip, y la constante principal que sirve para declarar el tipo de operación a realizar (di de leer o de escribir).

```

void setup(){
  Serial.begin(9600);
  //configuracion estandar de la comunicacion SPI del IC.
  SPI.setDataMode(SPI_MODE2);
    
```

```
SPI.setClockDivider(SPI_CLOCK_DIV32);
SPI.setBitOrder(MSBFIRST);
SPI.begin();
pinMode(9, OUTPUT);//pin de integrado
iniciarconstantes();
csalto();
delay(10);
}
```

En la última porción de código se inicia configurando el baud rate (velocidad de comunicación serial pines TX y RX) luego se debe trabajar en el MODO2 y con MSBFIRST, según especificaciones de la hoja de datos, mientras que la frecuencia de reloj con que trabajara la comunicación Arduino integrado la podemos dejar con un DIV32.

Se declara como salida el selector de chip y luego se llama a una función que sirve para asignar y declarar los registros, esto tiene como única finalidad hacer entendible el código y en lugar de poner números, poner el nombre al registro que se hace referencia.

<csalto> es una pequeña función junto con <csbajo>, útil cuando se trabaja con más dispositivos conectados a los mismos pines.

```
void csalto(void){
digitalWrite(cs,HIGH);
}

void cssdbajo(void){
digitalWrite(cssd,LOW);
}
```

En la siguiente porción de código vemos las rutinas de lectura y escritura básicas para registros de tamaño de 8 bits, en un pequeño esbozo de estas se puede decir que lo primero que se hace es capturar en el mismo byte la dirección del registro que debe ser un parámetro a enviar dentro de la función y además asignarle los primeros dos bits de identificación del tipo de operación (lectura o escritura) es por eso que se hace la suma lógica con reg OR leer, reg OR escribir. Además cuando ya se está listo con la información a transferir se selecciona al integrado con un csbajo (activo con un bajo) se esperan 5 microsegundos y se utiliza el comando <SPI.transfer> este es el que manda haciendo uso del protocolo SPI el byte correspondiente indicado dentro de sus parámetros. Después viene una espera de 5 microsegundos (se escoge un solo tiempo para no confundirse de los tiempos de la hoja de datos, el mayor tiempo con algo de sobra), la siguiente línea es para capturar y retornar el valor contenido en el registro del integrado. Luego se desactiva el chip.

```
unsigned char leer8(char reg){
reg = reg | leer;
csbajo();
delayMicroseconds(5);
SPI.transfer(reg);
delayMicroseconds(5);
return (unsigned char)SPI.transfer(0x00);//cambiar por unsigned byte
csalto();
```

}

Al igual que la porción de código anterior, la mayoría de comandos se repiten excepto porque ahora es una escritura y justo después de realizar la primera transferencia obligatoria, debemos seguir enviando los bytes que queremos separados por tiempos de 5 microsegundos.

```
void escribir8(char reg, char data){  
    reg = reg | escribir;  
    csabajo();  
    delayMicroseconds(10);  
    SPI.transfer((unsigned char)reg);      //register selection  
    delayMicroseconds(5);  
    SPI.transfer((unsigned char)data);  
    delayMicroseconds(5);  
    csaltol();  
}
```

Con todo esto listo podemos pasar a hacer algo útil con el integrado, esto es obtener valores RMS de corriente y voltaje censados.

Se explica solamente el código para obtener la corriente, y se inicia describiendo las funciones que invoca esta función.

```
int getStatus(void){  
    return leer16(STATUS);  
}
```

La función anterior es por decirlo así de segundo nivel pues ya utiliza las funciones leer16 que invocan toda la operación de lectura descrita anteriormente.

Además la forma en que se definen variables en las constantes es la siguiente:

```
#define ZX 0x0010 // bit 4 -
```

Las variables <lastupdate> <contando> y <salirse> solo sirven para evitar atascos por largos tiempos pero para propósitos de funcionamiento ideal debemos analizar el código sin tomarlos en cuenta.

La línea while (! (getStatus() & ZX)) tiene como objetivo sincronizar la operación de lectura con la frecuencia de la señal, ZX es uno en los ciclos positivos de la señal de voltaje, por lo tanto, getStatus() tiene como objetivo verificar en qué ciclo está la señal y dependiendo de eso permite salir de <while> cuya única restricción es esta. Haciendo esto logramos que los cálculos de RMS sean de acuerdo a la frecuencia de la señal, sino obtuviéramos cantidades innecesarias de lecturas repetidas pues el cálculo que el integrado realiza ni siquiera se ha llevado a cabo porque la señal es lenta.

```
//obtener valor IRMS
```

```

long getIRMS(void){
long lastupdate=0;
resetStatus();
lastupdate=millis();
while (! (getStatus() & ZX ))
long contando=millis();
long salirse=contando-lastupdate;
if ( salirse > 200) break;
}
return leer24(IRMS);
}

//obtener valor VRMS
long getVRMS(void){
long lastupdate=0;
resetStatus();
lastupdate=millis();
while (! (getStatus() & ZX ))
{//Serial.println("encerrado en vrms");
long contando=millis();
long salirse=contando-lastupdate;
if ( salirse > 200) break;
}
return leer24(VRMS);
}

//LA SIGUIENTE FUNCION
//IGNORA LAS PRIMERAS GETRMS Y SACA EL PROMEDIODE 50 LECTURAS
long vrms(){
char i=0;
long v=0;
getVRMS(); //ignora la primera lectura para evitar basura
getVRMS();
getVRMS();
for(i=0;i<50;++i){
v+=getVRMS();
}
return v/50;
}
//ahora para la corriente
long irms(){
char n=0;
long i=0;
getIRMS(); //ignora la primera lectura para evitar basura
getIRMS();
getIRMS();
for(n=0;n<50;++n){
i+=getIRMS();
}
return i/50;
}

```

La anterior porción de código es la definitiva que sirve, para empezar ignora las primeras lecturas y luego toma 50 lectura todo esto tiene lugar en ciclo cuando la señal está en la parte negativa, por lo tanto al descartar y sacar un promedio de unas cuantas lecturas lo que se quiere lograr es la fiabilidad de los datos.

Las únicas líneas que deben ir en loop principal son:

```
long voltaje_rms=vrms();
float voltaje_rms_corregido= voltaje_rms*0.000264585;
//Serial.println(voltaje_rms);// solo para calibrar, intercambiar según se explica
//Serial.println(voltaje_rms_corregido);//intercambiar según se necesite
```

Las tercera línea de la porción de código anterior es la que debemos des-comentar la primera vez que lo pongamos en funcionamiento, esto nos arrojara un entero que corresponde al valor binario guardado en el registro.

Para este caso no hay que tener un especial cuidado con los signos, puesto que los registros RMS son siempre positivos.

Si se desea mostrar en consola un valor real de la medición, se aplica una simple regla de tres, con un valor conocido de voltaje o corriente vemos cuanto es el valor entero del registro y ahora aplicamos la regla de tres para encontrar la constante por la cual debemos multiplicar el registro y obtener el valor de acuerdo a lo que estamos midiendo.

ANEXO 2

EJEMPLO 2: LVIFA

LabVIEW Interface For Arduino (LVIFA), presenta una herramienta para enlazar la comunicación directa entre las propiedades graficas de LabVIEW, y las aplicaciones de hardware de Arduino, como se explicó, en las aplicaciones de Arduino basadas en LVIFA el código es diseñado gráficamente en LabVIEW y desde este se controlan todos los pines de entrada y salida de la tarjeta, para ello se utilizan las funciones de la paleta del LVIFA, entonces cabe recordar que para que estas funciones actúen sobre los pines de la tarjeta se debe montar el Sketch "LVFA_Base" en el Arduino.

El ejemplo presentado a continuación es una aplicación sencilla de comunicación entre LabVIEW y la tarjeta Arduino haciendo uso de LVIFA, dicha aplicación consiste en un slide que sirve como selector de entre diferentes números, los números seleccionados son mostrados en un presentador de siete segmentos conectado al Arduino, además para esta aplicación se utiliza comunicación inalámbrica a través de antenas XBee que no es de uso necesario, sino solo para mostrar otras posibilidades de comunicación.

En la figura LVFA.1 presenta la interfaz de usuario para la aplicación creada, como se puede ver en el slide del lado izquierdo de la imagen se puede optar de los números entre 0 y 11, pero en el presentador solo se pueden mostrar dígitos del 0 al 9, los números 10 y 11 realizan funciones diferentes que incluyen manejo de patrones en el código de LabVIEW y que se muestran como patrones de luces giratorias en el presentador de 7 segmentos.

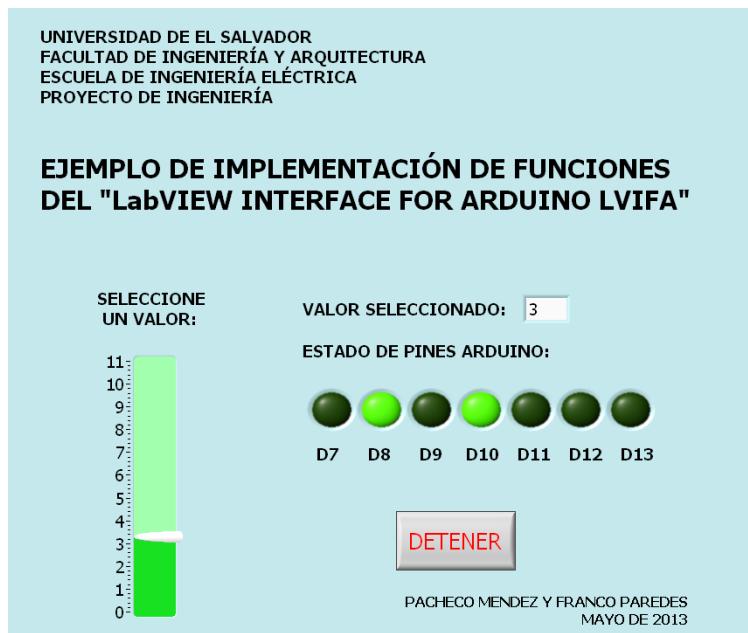


Figura LVFA.1 Panel frontal de la aplicación ejemplo.

ASPECTOS DE HARDWARE:

Para poder implementar este ejemplo se necesita conectar a los pines digitales del Arduino un presentador de siete segmentos, y dependiendo de si este es ánodo común o cátodo común así deberán ser los estados de los pines para encender o apagar cada segmento, en la figura LVFA.2 se muestra la conexión para un presentador de 7 segmentos que puede ser utilizada para esta aplicación.

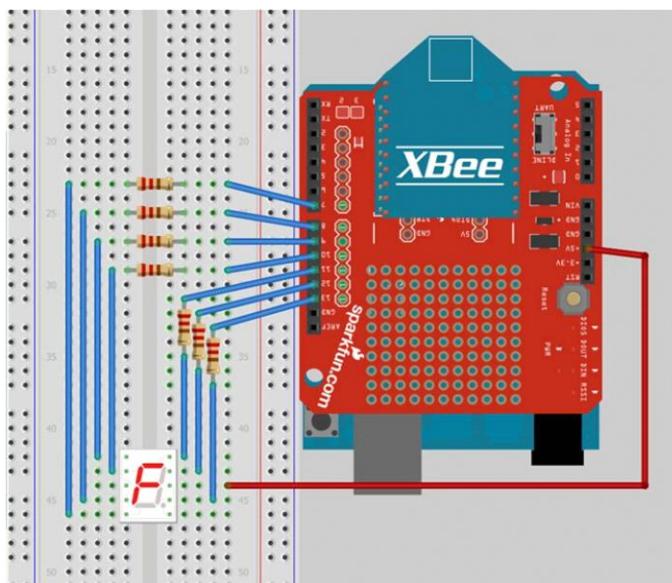


Figura LVFA.2 Diagrama de conexión para la aplicación de este ejemplo.

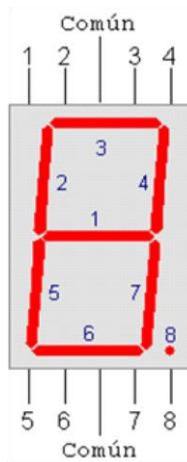


Figura LVFA.3 Descripción general de los segmentos de un presentador.

El diseño de esta aplicación se ha realizado para un presentador o display de 7 segmentos de ánodo común, a continuación se muestra una tabla generaliza de conexión para cualquier display de 7 segmentos de ánodo común:

Tabla LVFA.1 Conexión del presentador de 7 Segmentos.

Segmento del display	Pin correspondiente del XBee Shield
1	D9
2	D10
3	D11
4	D12
5	D8
6	D7
7	D13
8	Sin conexión
Común	5Vdc

ASPECTOS DE SOFTWARE

El software diseñado para este ejemplo se muestra en la figura LVFA.4, aquí se presenta el diagrama de bloques general, el cual será explicado con más detalle a continuación.

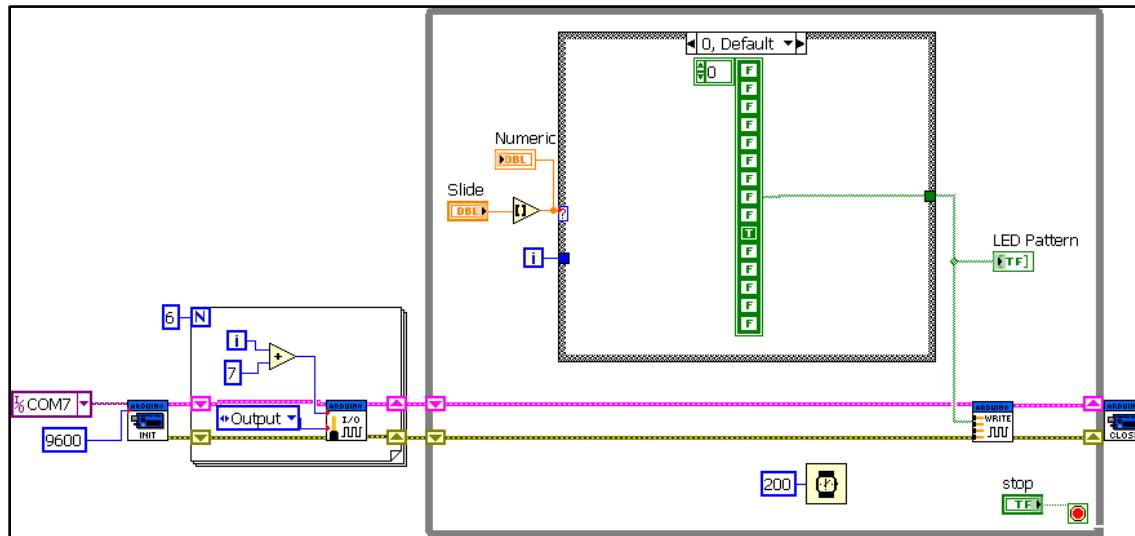


Figura LVFA.4 Diagrama de bloques general la aplicación.

Para poder crear un flujo de comunicación con la tarjeta Arduino es necesario utilizar la función “Init.vi” a esta función se le debe pasar el puerto en el cual está conectado la tarjeta Arduino mediante una función “VISA Resource name”²⁵, además del BaudRate al cual esta trabajando el dispositivo, normalmente 9600. Al igual que en un sketch normal de Arduino antes de poder lograr una comunicación con los pines se debe declarar a cada pin como entrada o salida, esto se logra

²⁵ La función “VISA resource name” es explicada en el capítulo 5.

en este caso mediante la función “Set Digital Pin Mode.vi” ya que solo se utilizan los pines digitales de la tarjeta, a esta función se le debe indicar el número de pin y además el modo en el que será utilizado, para evitar utilizar esta función varias veces se utiliza un “lazo for” para declarar como salidas los pines D7 a D13, este proceso se muestra en la figura LVFA.5.

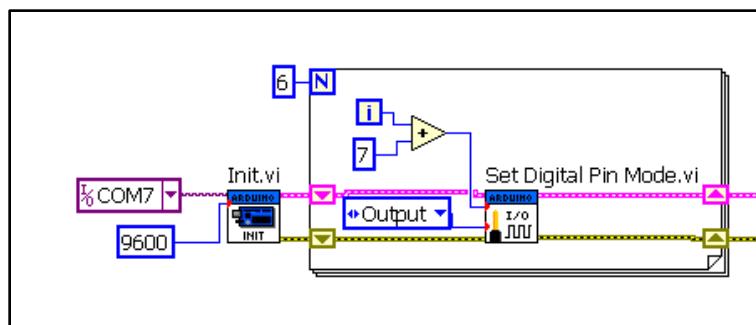


Figura LVFA.5 Inicialización de los pines digitales Arduino.

Una vez inicializados los pines se escribe directamente en el puerto digital utilizando la función “Digital Write Port.vi” la cual recibe un arreglo de una dimensión el cual contiene la información a ser escrita en el puerto digital. La figura LVFA.7 presenta el lazo de escritura implementado.

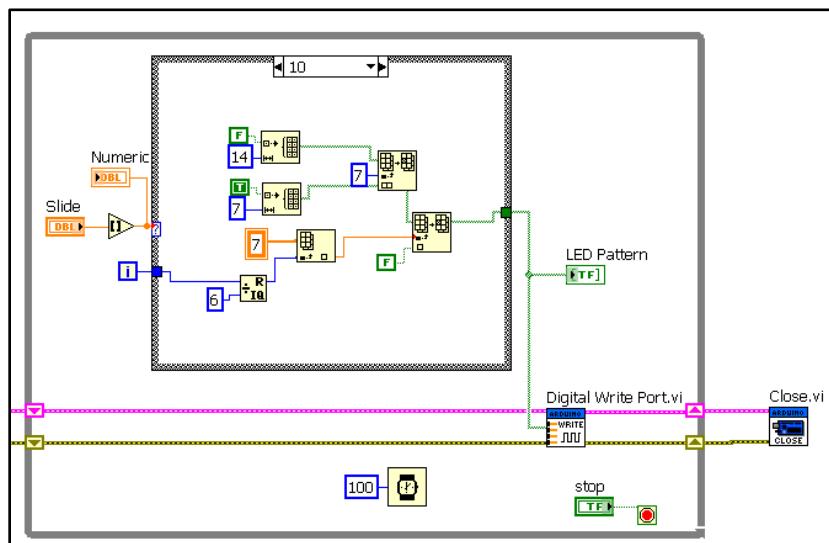


Figura LVFA.6 Lazo de escritura puerto digital Arduino.

Para escribir en el puerto el arreglo correspondiente para el número seleccionado se utiliza una estructura case y para formar el arreglo se utiliza una constante “arreglo”, esta se muestra en la figura LVFA.7, cada bit es cambiado al estado requerido para cada caso “True o False”.

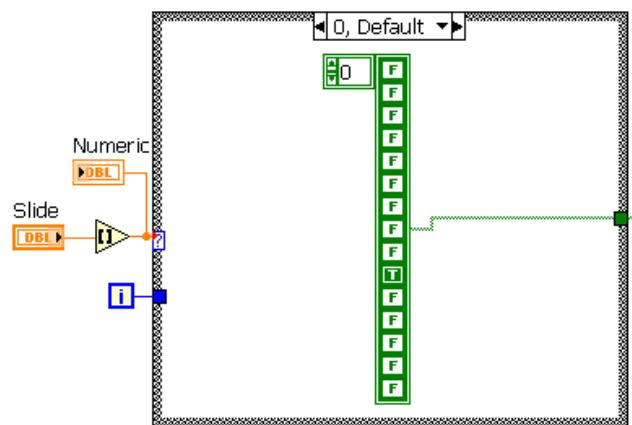


Figura LVFA.7 Constante arreglo a ser escrita en el puerto digital.

Finalmente para finalizar el flujo de comunicación creado se debe utilizar la función “Close.vi” (Ver figura LVFA.6), esta aplicación muestra solo el uso de las funciones básicas del LVFA ya que su principal propósito es mostrar la funcionalidad de las herramientas ofrecidas por LVFA.

ANEXO 3

HOJA DE DATOS DE LA PINZA DE CORRIENTE

Echun

ECHUN Electronic Co., Ltd

Split Core Current Transformer ECS1030-L72

The ECS10 Series are split-core current transformers. The CT can be mounted to existing panels, such as control centers or load centers, to measure or monitor wattage. These CTs can be mounted without removing existing cables for easier installation



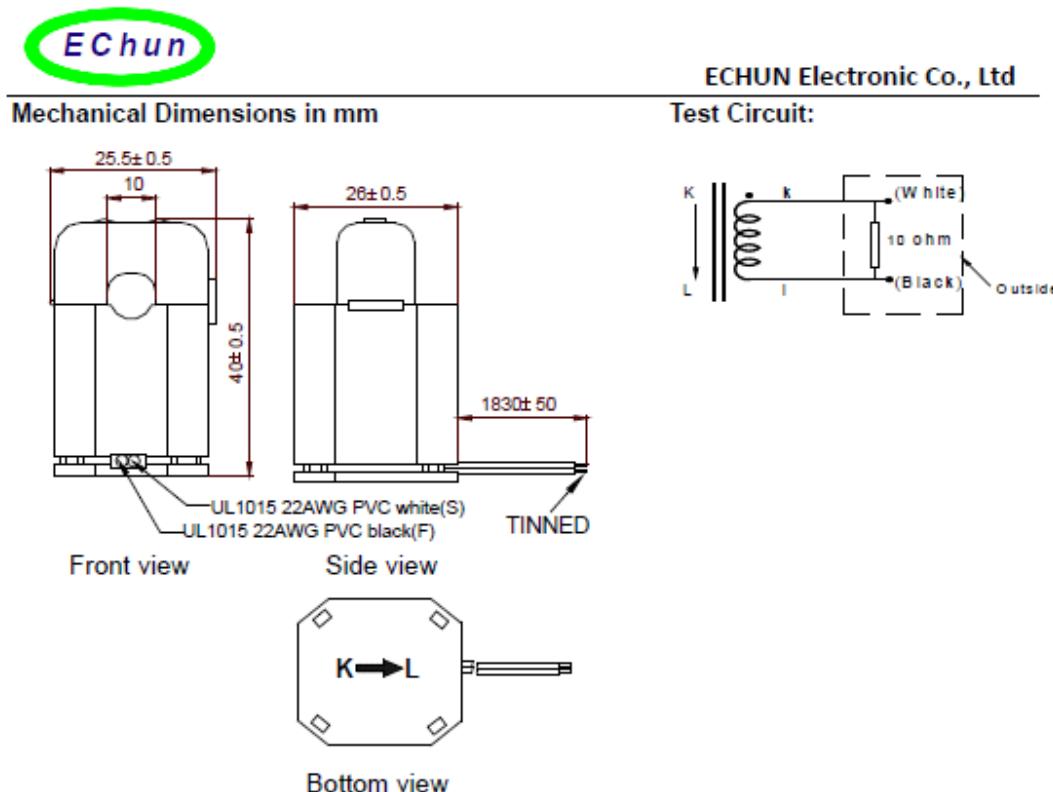
RoHS

Electrical Specifications

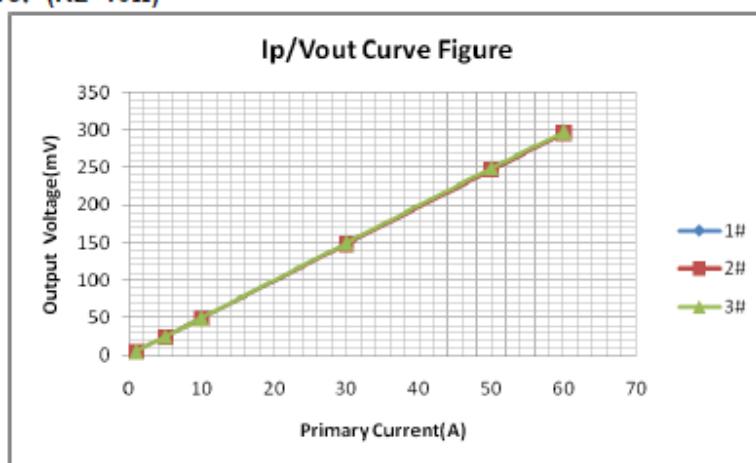
Rated Primary Current(Amp.) 50/60Hz	30nom(1~60A max)
Turnn ratio	Np:Ns=1:2000
Current Ratio	30A/15mA
D.C.Resistance at 20 °C	250 Ω
Accuracy @RL≤10Ω	2%
Linearity @RL≤10Ω	0.5%
Phase error at rated current range	≤4°
Operating Temperature Range	-40~65°C
Storage Temperature Range	-45~85°C
Dielectric Withstanding Voltage(Hi-pot)	2.5KV/1mA/1min
Insulation Resistance	DC500V/100MΩ min

Mechanical Specifications

CUP	PBT
Opening Dimensions	>10mm
Output type	UL1015 22AWG PVC WIRE(doubling wire)
Approx.Weight	60g



Output curve: ($RL=10\Omega$)



EChun

ECHUN Electronic Co., Ltd

Applications:

- Current Measurement
- Monitoring & Protection

Advantage:

- Low cost
- Mount easy
- Excellent linearity over the whole range
- Fully RoHS Compliant.

Material List:

NO	ITEM	DESCRIPTION	UL Card	MATERIAL	NOTE
1	CORE	UU20.6	/	N07	
2	CASE	PBTRG301	E171666	PBT	
3	BOBBIN	PBTRG301	E171666	PBT	
4	WIRE	2UEW Ø0.1mm	E164502	UEW/U	
5	LEAD WIRE	UL1015 22AWG	E77981	PVC	105°C 600V
6	SOLDER	99.3%	/	Sn	

CUSTOMER APPROVED

Approved by	Checked by	Acknowledged by

CUSTOMER P/ N(VER): _____