Aspectos de implementación en una interfaz I2C para controladores PSoC y AVR

Integración en el diseño de un data logger

Rafael B. Oliva
Universidad Nacional de la Patagonia Austral
Área Energías Alternativas y
L&R Ingeniería
Río Gallegos, Argentina
roliva@lyr-ing.com

Nestor J. Cortez
Universidad Nacional de la Patagonia Austral
Area Energías Alternativas
Río Gallegos, Argentina
nesjaco@gmail.com

Resumen—**La utilización de la interfaz I2C está ampliamente documentada en publicaciones desde que fuera introducida por Philips como protocolo industrial, para comunicación de dispositivos en distancias cortas. Sin embargo, su complejidad subyacente y la interdependencia entre la programación y el hardware frecuentemente dificultan su implementación en circuitos reales. Aquí se presentan algunos aspectos de implementación en controladores Cypress PSoC y en controladores Atmel AVR, en modalidades maestro y esclavo, como módulo de software o con soporte de hardware. Dichas implementaciones corresponden a un proyecto con financiamiento FONTAR (ANR Nº SC002/2003, reformulado 2004) para implementación de un data logger que ha dado resultados satisfactorios.**

Palabras clave: Interfaz I2C, controladores PSoC, controladores AVR, Sistemas embebidos, lenguaje C

I. INTRODUCCION

El bus I2C (*Inter-Integrated Circuit*) es un bus serial de dos líneas desarrollado por Philips [1] a fines de los '70 para la interconexión de microcontroladores y periféricos. Su uso se enfocaba a distancias cortas (dentro de la placa o a lo sumo entre placas) y se concibió para aplicaciones de audio, televisión y electrónica de consumo. Sus principales ventajas son su simplicidad circuital (reducido número de líneas y pines de controlador) y su efectividad. Aunque Philips mantiene la propiedad intelectual del uso del bus, el mismo ha sido replicado (Intel con *SMBus* [2] , Atmel lo denomina TWI o *Two-Wire-Interface* [3]) en variantes con distintos grados de similitud. La longitud del bus, sin buffers adicionales, puede llegar a unos 6 a 8m, con hasta 40 componentes. El I2C de Philips tiene variantes de 100kHz, 400kHz y recientemente 3.4MHz. Es un bus maestro/esclavo, con previsión para maestros múltiples. Todas los dispositivos acoplados se conectan con dos líneas: SDA o *serial data*, y SCL o *serial clock*. Esta simplicidad circuital trae aparejada cierta complejidad en la programación, ya que se requiere un mecanismo de direccionamiento para que un dispositivo maestro establezca comunicación con un esclavo (o con otro maestro). Todos los dispositivos deben tener una única dirección en el bus. Los dispositivos esclavos tienen una dirección predefinida (en la mayoría de los casos), pero los bits bajos de la dirección pueden asignarse de modo de tener múltiples dispositivos del mismo tipo en el bus (así por ejemplo, un reloj de tiempo real o RTC Philips PCF8563 viene con una dirección 0xA2, y una memoria serial 24LC256 tiene una dirección 0xA0, aunque son direcciones de 7 bits que requieren manipulación por software). Cada dispositivo "mapea" a partir de su dirección base una serie de registros (o buffer de memoria) que están disponibles al resto. Una posibilidad que da mucha flexibilidad a este esquema es utilizar microcontroladores como esclavos –sea que tengan hardware específico I2C o utilicen "*bit-banging*", en cuyo caso es posible alterar por software no solo la dirección sino el mapa de registros puestos a disposición del bus. La estructura de hardware típico del bus puede verse en la Figura 1, donde se muestran las dos líneas SDA y SCL, sus respectivos resistores de *pull-up* y la configuración bidireccional de los pines en cada dispositivo.

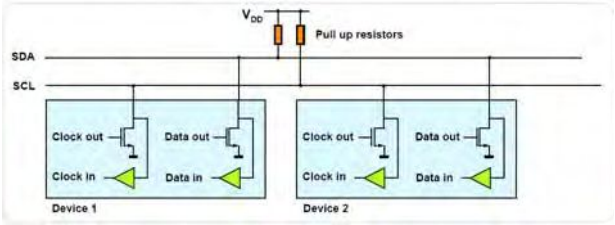

Figura 1. Estructura del bus I2C (Philips / NXP.com)

La configuración de múltiples dispositivos sobre el bus puede observarse en la Figura 2, que muestra una estructura con dos maestros y dos esclavos.

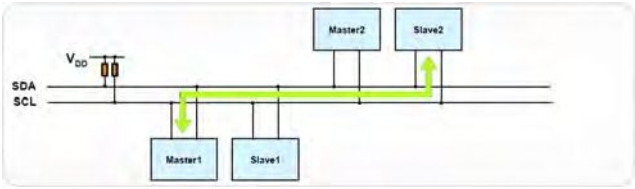

Figura 2. Bus multimaestro I2C (Philips / NXP.com)

El protocolo se inicia en el caso más sencillo con un maestro y varios esclavos. El maestro inicia la comunicación, con la siguiente secuencia:

1. El dispositivo maestro emite una condición de START o inicio, esto informa a los esclavos que deben escuchar sobre la línea serial de datos (SDA) para determinar las instrucciones.

2. El maestro emite la dirección del esclavo a quien se dirige, junto con un *flag* de R/W (lectura o escritura).

3. El esclavo que detecta su propia dirección en el bus responde con una señal de reconocimiento (ACK).

4. La comunicación procede entre maestro y esclavo sobre el bus de datos, tanto el maestro como el esclavo pueden recibir o transmitir datos dependiendo del *flag* R/W, y el que transmite emite 8 bits de datos con un ACK de 1 bit. Cuando la comunicación se completa, el *master* emite una condición de STOP para terminar la comunicación. Estas condiciones se emiten con transiciones en la línea SDA alto a bajo o bajo a alto, con la línea SCL alta (Figura 3)

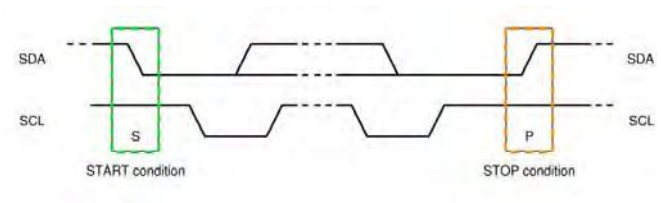

Figura 3. Temporización simplificada I2C (Philips / NXP.com)

II. IMPLEMENTACIONES CON PSOC

A. *Antecedentes en el proyecto ANR: el PSoC como maestro (por software) y dispositivos I2C convencionales*

En los inicios del proyecto ANR Nº SC002/2003 de L&R Ingeniería [8], se planteaba el desarrollo de un data-logger de arquitectura abierta en dos bases de microcontrolador distintas, con distintos niveles de costo y complejidad. Se inició con los dispositivos PSoC de la línea 8C26443, en ese momento novedosos ya que combinaban características de una FPGA, con módulos analógicos configurables y un *core* M8C propietario, con algún parecido al venerable 8051, mas una cantidad reducida de memoria Flash (16K) y RAM. Los dispositivos se configuraban con un software gratuito (PSoC Designer) en forma similar a una FPGA, y la programación del controlador en lenguaje C permitía implementar un sistema con la ventaja de poder combinara señales analógicas con funciones digitales en el mismo chip. Esa primera versión (2002-2003) se adquirió con un ICE (*In-Circuit Emulator* ó emulador) que se conectaba al puerto paralelo de una PC (Figura 4). En la primera etapa, el PSoC se conectó al bus I2C configurado como maestro a través de un módulo de software (*bit-banged*) provisto por Cypress, que permitía acceder a dos dispositivos esclavos: una memoria serial 24LC256, y un reloj de tiempo real (RTC) PCF8563 de Philips.

B. *Ensayo del acceso a EEPROM Serial vía I2C*

Los ensayos de la interfaz I2C por software implicaron un caudal importante de trabajo y repetidas consultas a Cypress y sus publicaciones [5],[6], ya que no se contaba con experiencia en la utilización de este bus. El módulo `I2Cm_1` es una API (*Application Program Interface*) que enlaza la aplicación escrita por el usuario con rutinas de bajo nivel dentro del PSoC y se basa en líneas de E/S seleccionables (una se elige como SDA, la otra como SCL). En el listado siguiente se muestra una inicialización básica de las primeras pruebas (2004):

```
// ********************************************
//Now write TxCBuf Array to EEPROM, starting from
// Adress 0000.
I2Cm_1_Start(); //I2Cm_1_Startinitializes
                //interface module..
                //31.1.04 Status chk added
if(status=I2Cm_1_fSendStart(EEPROM_ADDR,I2Cm_1_WR
-- ITE))
          {
          Comm_TxCStr ("S"); // OK
          }
               else
               {
               Comm_TxCStr ("X"); // err
               }

I2Cm_1_fWrite(0x00);  // Write AH within EEPROM
I2Cm_1_fWrite(0x00);  // Write AL within EEPROM
// A write involves directly continuing with no
// Start..
for (i=0; i< 11; i++) {  //31.1.04 Status chk
 if(status=I2Cm_1_fWrite(txCBuf[i]))
               {
               Comm_TxCStr ("1");  // OK
               }
               else
               {
               Comm_TxCStr ("0");// write fail..
               }
  ;}
I2Cm_1_SendStop();
```

Después de acomodar ciertas particularidades especiales de la EEPROM 24LC256 utilizada, el sistema funcionó correctamente, y posteriormente se avanzó con la implementación de un *File System* rudimentario para obtener una utilidad razonable de los 32KB del dispositivo I2C instalado. La placa permitía instalar hasta 2 chips de memoria en la cadena I2C, y ampliar de esa manera la memoria disponible.

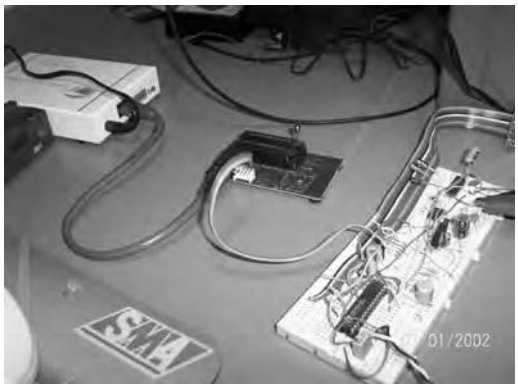

Figura 4. Detalle de conexión a ICE - Ensayo de PSoC - CY8C26443

C. *Reloj de tiempo real vía I2C y evolución posterior*

Una tarea similar requirió la interfaz con el reloj de tiempo real PCF8563 [7] , para el cual se montó un circuito con una pila de backup de 3V de litio, lo cual permitía el mantenimiento de la hora real aún después de apagar el circuito. Este dispositivo mapea registros de hora, minuto, segundo en formato BCD, y el acceso se realiza realizando una lectura a RAM del buffer completo a través del bus, y después convirtiendo los registros a números concretos y formateando de acuerdo a fecha y hora.

El circuito fue evolucionando con el avance del proyecto hasta convertirse en la placa DLCy cuyo diagrama en bloques vemos en la Figura 5, y que en sus versiones finales incorporó el microcontrolador PSoC de la línea 8C29443, con prestaciones analógicas muy superiores, 32K de Flash y 2K de RAM, lo cual lo hacía más adecuado para su uso como un data logger sencillo (aunque la memoria Flash queda prácticamente llena en la mayoría de las aplicaciones).

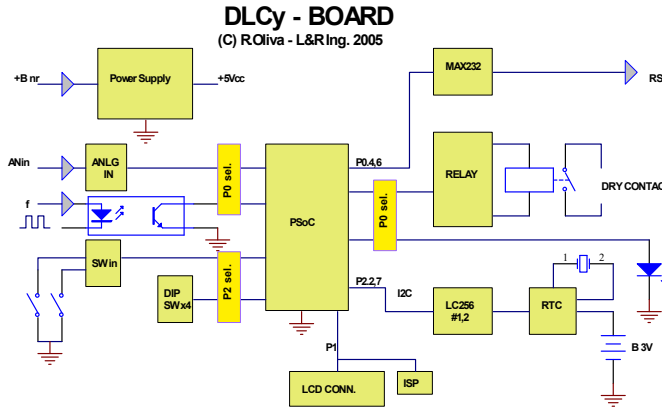

Figura 5. DLCy con PSoC - CY8C29443

Este esquema siguió utilizando dos pines elegidos P2.2, P2.7 para que el módulo de Maestro I2C por software implementara el acceso a los datos en la memoria LC256, y coordinaba las operaciones de adquisición de datos con el RTC, permitiendo las funciones básicas de un data logger.

Figura 6. DLCy con PSoC - CY82C29443

La descarga de datos se realiza a través de un puerto serie RS232. Este equipo ha dado muy buen resultado, y se han realizado múltiples aplicaciones. En la Figura 7 se muestra el registro de datos de un sistema de control de freno para un pequeño aerogenerador [4].

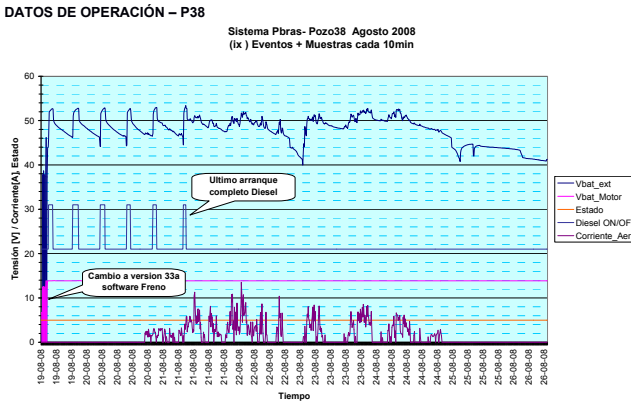

Figura 7. DLCy con PSoC - CY8C29443

En este caso, el almacenamiento es realizado en la EEPROM mencionada, y en ella se registran entre otros datos la tensión de la batería y la corriente que produce el aerogenerador.

III. Expansión I2C en arquitectura Atmel AVR (TWI)

A. *Placas de mayor capacidad con procesador AVR*

La utilización de I2C de Philips (o TWI para Atmel) se expandió en la segunda etapa del proyecto con la plaqueta CL2bm1, orientada a realizar un datalogger ampliado utilizando procesadores Atmel AVR ATMega [3] de mayores prestaciones. Incluye por ejemplo capacidad para almacenamiento en tarjetas SD hasta 2GB, puerto serie dual configurable a conexiones RS232, RS485 y/o USB y reloj de tiempo real con compensación por temperatura (Figura 8).

Figura 8. Placa CL2b-m1 con procesador AVR

Asimismo, la mayor capacidad de memoria (128K de Flash, 4K de SRAM en ATMega1284P) permite la

construcción de un firmware con funcionalidades más complejas, por ejemplo manejo de archivos y registro de eventos.

B. Acceso a periféricos con bus I2C/TWI en AVR

En las placas CL2bm1, la interfaz I2C es un elemento central de su estructura de comunicaciones. Internamente, el procesador configura su módulo I2C de hardware como maestro y accede a los periféricos RTC (el mismo PCF8563) y de I/O tipo PCA9538, que permite expandir el número de entradas o salidas discretas del procesador. Un conector específico de expansión I2C con protecciones de sobretensión (Figura 10) permite agregar módulos esclavos externos. La implementación del código para el RTC no presentó dificultad ya que la API del compilador C para AVR utilizado (CodevisionAVR [9]) contenía rutinas para el PCF8563. Para el PCA9538 se escribió un driver especial.

Se eligió utilizar para los primeros desarrollos una placa DLCy como "periférico" para conectar dicho dispositivo al CL2b, y de paso utilizar la función "I2C-Hardware-Slave" presente en los modelos más nuevos (ej 8C29443) del PSoC. Esta función implementa un módulo de hardware específico de I2C (maestro o esclavo) con su propia API, que resulta en una comunicación más veloz y confiable respecto a la versión anterior *"bit-banged*". Se encontraron algunos problemas en la conexión (el camino de GND entre ambas placas debe ser de baja resistencia), y en el software, pero el resultado ha sido por demás satisfactorio y se han implementado ya dos módulos periféricos que utilizan un PSoC 8C29443 como esclavo I2C con esta función, denominados M4/E (Figura 9) y M4/Freno.

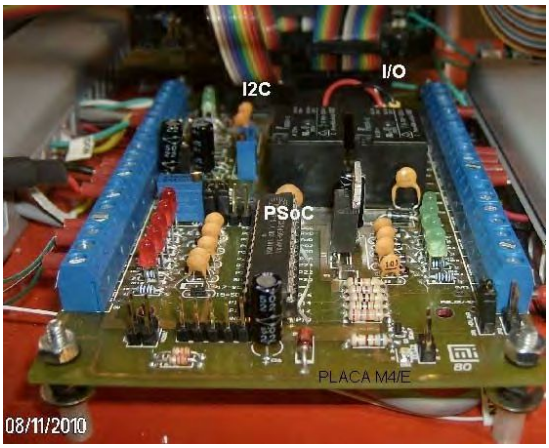

Figura 9. Placa de Expansión M4/E con PSoC como I2C/Slave

C. Reutilización de código I2C/TWI en AVR

Para el procesador ATMega644P/1284P en CL2bm1 se utilizó el módulo de hardware I2C incluido en la CPU, con pines asignados explícitamente. Se reutilizó para los drivers de los periféricos (esclavos) código de placas comerciales AVR, para lo cual fue necesario realizar una revisión exhaustiva de sus funciones. Por ejemplo, en una de las rutinas de menor nivel, *twi_step()* que se transcribe a continuación, se presentaba un error aleatorio que dejaba al bus colgado. Esto podía ocurrir ante ruido o desconexión de una segunda placa, y se debía a un *timer* que no era incrementado durante los tiempos de espera del bus. Al realizar la corrección desapareció el error. Es común que placas comerciales se suministren con módulos de código "de muestra" que no ha sido probado exhaustivamente, y su reutilización puede conducir a problemas muy difíciles de detectar.

```
/* 17.5.09 Modify options.h to access twi-cl2-2.c/.h, Add
label TIMERMAX for TWI Timer in TWIStep() routine, and
increment this timer.
/*********************************************************
    Sends a two wire interface command – para CL2b a 184kHz
    **
    ** Parameters:  twcr_mask, Command to be sent
    ** Status, Flag to see if we need to wait for the status
    **
    ** Returns: 0 - Command sent successfully
    **          1 - An error occurred
    **
    ** Modified 17.5.09 Added TWI_STEP_TIMER_MAX label,made
    ** = 100 as before, and ** make TWITimer active (there
    ** were no increments!!)
    *********************************************************/
    int8 twi_step(uint8 twcr_mask, uint8 status)
    {
        uint8 twi_status;

        TWCR = twcr_mask | TWI_ENABLE;
        if(status != TWI_NO_WAIT)
        {
            TWITimer = 0;
            while(((TWCR & 0x80) == 0) && (++TWITimer <
               TWI_STEP_TIMER_MAX))
             //17.5.09 Increment TWITimer!..
             ;

            twi_status = (TWSR & 0xFC);

            if((status != TWI_IGNORE_STATUS) && ((twi_status
                != status) || ((TWCR & 0x80) == 0)))
            {
                TWCR = TWI_SEND_STOP | TWI_ENABLE;
                TWITimer = 0;
                while(((TWCR & 0x80) == 0) && (++TWITimer <
                  TWI_STEP_TIMER_MAX))
                // 17.5.09 Increment TWITimer!..
                    ;
                return(TWI_ERROR);
            }
         }
         return(TWI_SUCCESS);
    }
```

La interfaz I2C conectada al reloj permite mantener una hora medianamente precisa con el cristal standard de 32.768kHz (Figura 10). Aún así, se observó deriva a lo largo de varios meses en productos reales como el DLCy, por lo cual la placa CL2bm1 de producción (Figura 11) incorpora la opción de un TXCO, oscilador compensado por temperatura.

D. Aplicaciones de control mas complejas con bus I2C/TWI en combinaciones AVR y PSoC

Se ha verificado la posibilidad de llevar adelante aplicaciones relativamente complejas a través de la interfaz I2C utilizando el conector de expansión de la placa CL2bm1. En la Figura 12 se aprecia uno de los sistemas que realiza el control de un aerogenerador, paneles fotovoltaicos, banco de baterías de 48V y dosificación con bombas especiales, y que funciona en un emplazamiento de clima extremo desde fines de 2010. La placa de CPU actúa como maestro, y las placas M4/E y M4/Freno como esclavas en dos direcciones distintas sobre el mismo bus. Cada una de las placas periféricas tiene un controlador PSoC 8C29466, que utiliza el módulo I2C por hardware en configuración esclavo, y las direcciones en el bus

se programan en la memoria flash del PSoC. El programa interno del periférico "publica" en el bus I2C una cantidad de registros, algunos de lectura/escritura y la mayoría de lectura. Al momento de compilar el programa principal del CL2bm1, se enlazan un par de archivos .c y .h que contienen las rutinas que constituyen la API, con las funciones de alto nivel que utiliza el programa principal para acceder a los registros de las dos placas esclavas a través del bus I2C. No se observaron problemas de compatibilidad entre las implementaciones TWI de Atmel e I2C de Cypress, y una vez resueltos los accesos de bajo nivel, la comunicación aparece al programador como una secuencia de lecturas y escrituras de registros.

Como se puede apreciar en la Figura 12, las distancias entre placas no son significativas y esto contribuye a una mayor inmunidad al ruido. Si bien Philips y otras firmas producen integrados denominados "bus extenders" para ampliar alcance físico de las conexiones I2C, su aplicación principal para reducir los costos y la complejidad sigue siendo dentro de un mismo equipo.

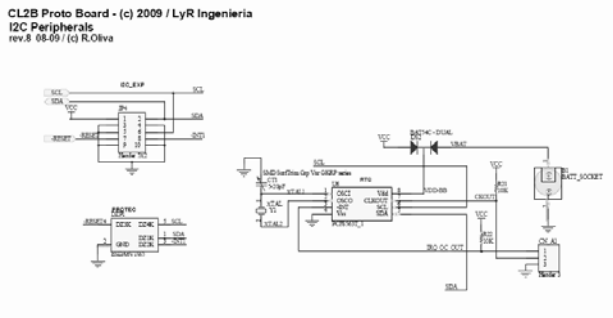

Figura 10. Conector de expansión y RTC conectada al bus I2C

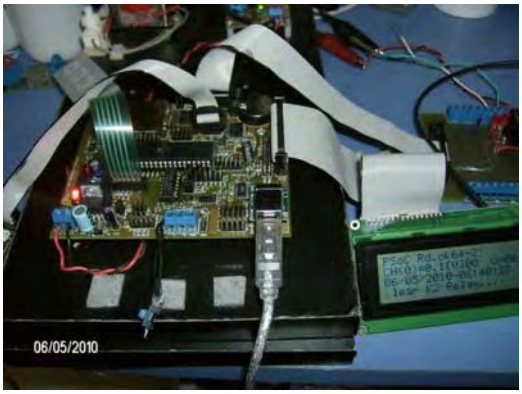

Figura 11. Placa CL2bm1 de producción en ensayos

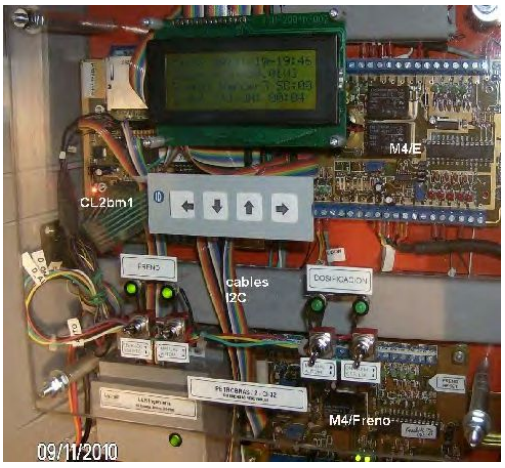

Figura 12. Aplicación de CL2bm1 con dos placas (M4/E y M4/Freno) conectadas como esclavas I2C en direcciones individuales

IV. CONCLUSIONES

La utilización del bus I2C presenta múltiples ventajas para la implementación de sistemas embebidos, aunque la complejidad subyacente y el esfuerzo inicial de programación son mayores que en otros tipos de sistema de comunicación.

Además, el diseño concreto y construcción de equipos que utilicen I2C encierra ciertos aspectos prácticos que son poco difundidos. Sin embargo, si se sortean estos obstáculos iniciales, los ahorros en costo y el incremento de confiabilidad (por tratarse de una tecnología con madurez adecuada) son muy importantes.

REFERENCIAS

.

[1] Philips/NXP The I2C-Bus Specification, Version 2.1. January 2000 http://www.semiconductors.philips.com/products/interface_control/i2c/facts

[2] SM Bus (System Management Bus – Intel 1995) http://smbus.org/specs/

[3] Atmel – AVR Processors www.atmel.com

[4] Oliva, R. y Vallejos, R , "Requerimientos para la Evaluación de Curvas de Potencia en aerogeneradores de baja potencia para carga de baterías - Diseño de su Implementación.", ASADES 2006, Buenos Aires (2006) www.asades.org.ar/asades06/horario-ponencias.xls.

[5] Van Ess, D. , A Thermistor- Based Thermometer, PSoC Style, Cypress Semiconductor App. Note AN2017, (2002)

[6] Cypress Semiconductor AN2305 – Using Cypress I2C port Expanders with Flash Storage, http://www.cypress.com

[7] Philips PCF8563 datasheet (http://www.semiconductors.philips.com)

[8] 2004-2010 Proyecto ANR-SC02/2003 -FONTAR:"Sistema de Adquisición de Datos (Data Logger) de Bajo Costo y Arquitectura Abierta para Aplicaciones Ambientales y de Energía" Presentación 03/10: http://www.lyr-ing.com/Assets/Images/Projects/ANR/PresANR-Evolucion_a_03-2010.pdf

[9] CodevisionAVR - http://www.hpinfotech.ro/html/cvavr.htm

.