Diseño de una aplicación Web embebida para el control de la carga útil de un satélite geoestacionario

Ing. Gabriel Rodrigo Caballero
gcaballero@iua.edu.ar
Dr. Pedro E. Colla
pcolla@iua.edu.ar
Instituto Universitario Aeronáutico, Departamento Postgrado Sistemas Embebidos
Av. Fuerza Aérea 6500 (5010) Córdoba, Córdoba, Argentina

Abstract. Este trabajo presenta una solución real de diseño e implementación de una aplicación de control del payload de un satélite geoestacionario construida como un ambiente web embebido operando en una plataforma de microcontroladores Rabbit. La misma permite enviar comandos y leer telemetría de la carga útil durante la integración al modelo de ingeniería y vuelo. Se plantea en primer lugar el diseño del hardware que provee al sistema de las interfaces adecuadas para llevar a cabo la conectividad entre el payload y el módulo. El resultado final es un dispositivo capaz de realizar el monitoreo y control, a través de una red Ethernet, de distintas variables de estado, proveyendo al usuario de una interfaz gráfica que puede ser ejecutada en cualquier terminal remota con conexión a Internet.

Keywords: Carga útil, satélite, web, sistemas embebidos, telecomandos, telemetría, geoestacionario.

Introducción

En el proceso de construcción de un satélite uno de los principales objetivos es lograr comandar y conocer el estado de distintas variables de la *carga útil* (payload) durante la integración con los modelos de ingeniería y vuelo. En el proyecto referido en este trabajo surgió la necesidad de explorar las diversas alternativas que permitirían implementar conectividad TCP/IP (Transmission Control Protocol/Internet Protocol), proveer el acceso a los subsistemas mediante un servicio Web y gestionar, en una capa más baja del stack del protocolo, el hardware subyacente de la plataforma por medio de interfaces estrictamente definidas de la carga útil. El módulo encargado de comandar el payload es la unidad de interfaz de distribución de carga útil (PIDU por su nombre en inglés *Payload Interface Distribution Unit*), en este diseño se busca sustituir el PIDU de vuelo en la etapa de integración del payload al modelo de ingeniería del satélite.
En tecnología satelital se crea con éste propósito un equipo eléctrico de soporte en tierra (EGSE por sus siglas en inglés *Electrical Ground Support Equipment*) el cual sirve de apoyo durante la fase de integración y ensayo de cada uno de los subsistemas del satélite, tanto en el modelo de ingeniería, como en el modelo de vuelo. En este trabajo se describe el diseño de un equipo de ésta índole que sirve de apoyo en la fase de integración del payload de un satélite geoestacionario el que ha sido especificado para trabajar en banda Ku con canales de 36MHz y 72MHz de ancho de banda, como así también con un cierto número de entradas/salidas digitales y analógicas. Uno de los desafíos que debe resolver exitosamente el diseño propuesto es la implementación de la solución mediante herramientas estandar y que permita entregar desde una plataforma embebida una GUI (GUI por las siglas en inglés Graphical User Interface) flexible para su operación. Los alcances del proyecto requieren que este deba ser realizado bajo estrictos y restrictivos requerimientos de costo, calendarios y calidad claramente especificados.

Desarrollo del módulo y aplicación embebida

En base a las interfaces del PIDU y a las necesidades de los usuarios en la fase de integración, es que se desprenden los requerimientos de la plataforma embebida, principalmente en lo que respecta al hardware de la misma. Para el desarrollo se eligió una plataforma RCM3365 [R01] basada en microprocesadores Rabbit[R03], la que cumple con los requisitos de cantidad de entradas/salidas, conectividad Ethernet, stack TCP/IP integrado y prestaciones de espacio en memoria adecuadas para este desarrollo.
Una vez seleccionada la plataforma, se desarrollaron los restantes componentes de hardware y software, actividades que involucraron:

- Diseño del la plataforma de hardware de acuerdo con los requerimientos de interfaces del fabricante del *payload*.
- Diseño y fabricación del circuito impreso, listados de materiales y documentación del desarrollo.
- Poblado de la placa (630 componentes) y soldadura por olas y horno de termo-fusión.
- Diseño e implementación de la aplicación embebida en la plataforma seleccionada (RCM3365) mediante Dynamic C para el envío de telecomandos y recepción de telemetría a través de la plataforma de hardware con el payload.
- Implementación de un servicio Web donde la información a desplegar se genere utilizando el formato HTML y se despliegue utilizando *RabbitWeb*[R02].
- Definición del protocolo de comunicación para una aplicación remota, usando TCP/IP.

Definición de interfaces y requerimientos de diseño

De acuerdo con las especificaciones de la misión serán necesarios los siguientes tipos de interfaces:

- Low Level Commands (LLC) – Envío de comandos digitales
- Digital Relay (DR) – Lectura de Telemetría digital
- Analog Ouput (AN) – Lectura de Telemetría analógica
- Temperature Sensing (PT) – Lectura de telemetría analógica

Estrategia de diseño

En la Tabla 1 se especifican las interfaces LLC, para el caso de DR se corresponden entradas digitales TTL y el rango de las lecturas analógicas es de 0 a 5V.

Analog acquisition	< 0.5% full scale
Low Level Command characteristics (LLC)	26V (+3V/-2V) / 45ms (±5ms)
High Level Command characteristics (HLC)	26V (+3V/-3V) / 515ms (±15ms)
Power consumption	50W @ 100V (without Power distribution)

Tabla 1 Especificaciones de interfaces LLC

Teniendo en cuenta los requerimientos de diseño y las especificaciones del payload la plataforma de hardware debe contar con 16 salidas LLC, 6 entradas DR, 2 AN y 2 PT. El envío de comandos es vía interfaces LLC y la lectura de telemetría discreta mediante interfaces DR, por lo tanto asociamos *telecomandos* (TC) con LLC y *telemetría* (TM) con DR.

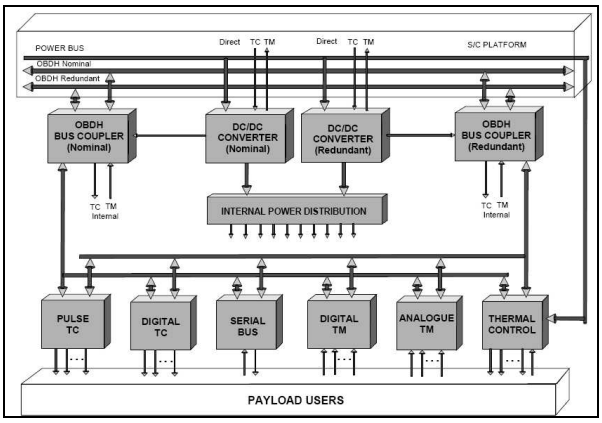

Figura 1 Diagrama de arquitectura general de interfaces del PIDU

La arquitectura de interfaces es la descripta en la Figura 1 donde se puede apreciar un diagrama del PIDU, para el cual, la plataforma embebida debe conectarse como un usuario del payload.

A partir del diseño de alto nivel del satélite se define el entorno de integración así la configuración base de los EGSE. La arquitectura seleccionada utiliza armarios de 20" resistentes a las vibraciones conteniendo una PC industrial, una pantalla de 17" y de la electrónica adicional involucrada en cada subsistema a ser integrado. El software que corre en la PC tiene como propósito controlar la electrónica del EGSE para que se vincule con cada subsistema. Surge entonces la necesidad de contar con una GUI con la que puedan configurarse parámetros establecidos durante la etapa de integración y ensayo.

El enfoque convencional es implementar este diseño mediante una FPGA, pero para satisfacer los requerimientos de conectividad con la *plataforma de simulación* (SIMPLAT por sus siglas en inglés *Simulation Platform*) es mandatorio disponer de un stack Ethernet en 10Mbps.

Como estrategia de diseño todos los módulos EGSE se comunicarán con el SIMPLAT por medio de TCP/IP, con lo cual es evidente la importancia de seleccionar una plataforma que cuente con soporte para este protocolo. Una posible solución explorada en el diseño explicado en este trabajo es recurrir a una interfaz basada en Web donde la información se codifique en HTML. Se aprecia la flexibilidad de configuración de la misma y por la facilidad de acceso remoto desde cualquier PC conectada al ambiente de desarrollo del satélite.

En este marco, considerando restricciones en los tiempos del proyecto, disponibilidad de herramientas de desarrollo y costos, la elección de un microprocesador de la familia Rabbit proporciona la flexibilidad necesaria puesto que esta provee capacidad de conexión TCP/IP, programación en una versión propietaria de C (*Dynamic C*) y abundantes recursos de desarrollo.

El diseño de interfaces con el payload, LLC, DR, AN y PT es, por tratarse de un satélite geoestacionario, propietario de la empresa proveedora de la carga útil. Está implementado con interfaces muy robustas y resistentes a fallas. Esos mismos atributos por otra parte hacen que sea imposible encontrar en el mercado hardware de control que se adapte para los requerimientos de la misión. Fue necesario entonces diseñar un hardware de interfaces a nivel electrónico específico que contemplara al mismo tiempo conectividad con el usuario y con el SIMPLAT.

Selección de la plataforma embebida RCM3365

De acuerdo con las prestaciones de diseño de hardware, y la posibilidad de que el módulo a ser desarrollado fuera usado en la integración de otros subsistemas del satélite como potencia, control de actitud, navegación y otros, se tuvieron en cuenta la disponibilidad de un número importante de entradas/salidas LLC, DR adicionales para expansión futura.
En el otro extremo las características de comunicación TCP/IP servicio web y la posibilidad de acceso vía Telnet, requieren cantidades de memoria significativas lo que resulta clave al momento de elección de la plataforma de desarrollo. Conjugando espacio en memoria (512K programa, 512K datos), capacidad de almacenamiento masivo y la cantidad de E/S provista por 52 líneas, se seleccionó para el diseño el microprocesador RCM3365 de la familia Rabbit.
La plataforma seleccionada soluciona varios problemas de diseño de hardware al proveer una capa física Ethernet, tolerancia al régimen de alimentación y memoria incorporada en dispositivo [R03].
Al seleccionar una plataforma embebida basada en RCM3365 se facilita, además, la solución de restricciones de implementación puesto que el módulo de desarrollo se monta en un zócalo de interfaz. De esta forma se puede trabajar de manera simultánea a la puesta en marcha del hardware para posteriormente ser integrado en la plataforma final.
Otro problema resuelto con la elección, es solucionar la capa física de Ethernet, puesto que el módulo cuenta con un conector RJ45 y el hardware asociado para operar con este protocolo.

Diseño del Hardware de interfaz

Una vez finalizada la etapa de definición de requerimientos de diseño se pudo avanzar con la elaboración de los planos esquemáticos-eléctricos, del hardware de interfaz que hace de nexo entre la plataforma embebida RCM3365 y las interfaces de la carga útil.
El diseño fue sometido a dos procesos de revisión por estar involucrado directamente con un subsistema de un satélite y de manera de realizar la detección de defectos lo más temprano posible en el ciclo de vida del proyecto. Los procesos de revisión típicos son la revisión preliminar de diseño (PDR por *Preliminary Design Review*) y revisión crítica de diseño (CDR por *Critical Design Review*). Una vez establecida la línea de base de los requerimientos de diseño, se realiza la PDR con el objetivo de saber si se han comprendido esos requerimientos y se está en condiciones de avanzar con la fase de diseño de detalle. En este análisis, se presenta el sistema como un diagrama en bloques.
Finalizada la PDR comienza la etapa de desarrollo y antes de fabricar la placa se realiza la CDR donde se estudia a nivel de circuitos y en detalle cada una de las interfaces involucradas y los posibles modos de fallo. En este análisis se presentan planos esquemáticos muy detallados de cada circuito.
Por último se realiza un análisis independiente de modos de falla (FMEA *Failure mode and effects analysis*). En caso de detectar riesgos de que el fallo de un componente se propague y dañe el hardware de vuelo, se identifica la mitigación a realizar mediante estrategias de protección como interfaces opto-acopladas o de aislamiento galvánico.
Se utilizan en la implementación, prácticas de Ingeniería de Software [C04,P01] consistentes con la complejidad del desarrollo realizado, la criticidad de los entregables y la necesidad de alcanzar estrictos requerimientos de calidad.
Diseñado el esquemático se desarrolla el diseño del circuito impreso (PCB) teniendo en cuenta aspectos de Interferencia y compatibilidad electromagnética (EMI/EMC) e integridad de las señales entre otros.

Diseño e implementación de la aplicación embebida

Para la programación embebida se utilizaron los recursos de desarrollo provistos por el lenguaje Dynamic C propietaria de la plataforma RCM3365. Este lenguaje es un derivado de C adecuado y mejorado para los requerimientos de los microprocesadores de la familia Rabbit. El fabricante de los módulos entrega junto con el kit de desarrollo de la plataforma, el software para programación [R05] y el cable de conexión con la PC utilizada como ambiente de desarrollo.
Una vez finalizada la puesta en marcha a nivel electrónico de la plataforma de interfaz de hardware se comienza a trabajar en el componente de software. En primera medida se validan todas las entradas/salidas digitales, LLC y DR, comprobando que el software sea capaz de comandar y leer todos los puertos del microprocesador que habían sido asignados a las distintas interfaces del PIDU. Este proceso se simplifica notablemente aplicando una técnica de "*stubs*" [C04] (funciones simuladas no operativas) donde pequeños segmentos de código son aplicados a verificar un aspecto en particular.
A continuación se presenta en el Ejemplo 1, la programación que muestra como se escribe y lee un puerto en Dynamic C [C01] para un módulo RCM3365.

```
main()
                                        {
    BitWrPortI(PDDDR, &PDDDRShadow,0,1);          // switch, input
    BitWrPortI(PDDDR, &PDDDRShadow,1,0);          // LED, output
    BitWrPortI(PDDR, &PDDRShadow,1,0);                    // apaga LED
    while(1){
          if(!BitRdPortI(PDDR,1))
            BitWrPortI(PDDR,&PDDRShadow,0,0);  /* Prende el LED DS1 */
          else
            BitWrPortI(PDDR,&PDDRShadow,1,0);  /* Apaga el LED DS1 */
             }
    }
```

Ejemplo 1 Lectura de puerto en Dynamic C

Los canales de entrada analógicos AN y PT provienen de un conversor analógico-digital (ADC *Analog To Digital Converter*) de Analog Devices AD7993AR [C03] que posee una interfaz serial I2C [I01].
El lenguaje de programación Dynamic C incluye una interfaz de desarrollo API [D01] llamada I2C.LIB [R04], que maneja los aspectos genéricos de la interfaz I2C y proporciona funciones simplificadas [C02] para que el diseñador utilice el módulo RCM3365 como un Master del BUS según lo mostrado en la Figura 2.

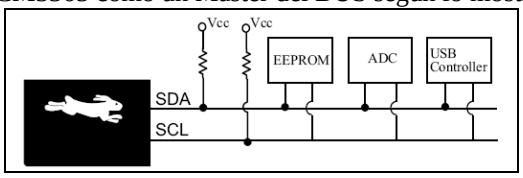

Figura 2 Modulo de arquitectura Rabbit como master de Bus I2C

Los principales elementos del API proporcionado por la librería pueden ser vistos en la Tabla 2. Una vez corroborado el correcto funcionamiento de todas las entradas/salidas del la plataforma embebida, LLC, DR, AN y PT, se pone el énfasis en la interfaz con el usuario. El software Dynamic C provee un entorno de trabajo, para el cual, una de sus principales ventajas es poder realizar una depuración del programa paso a paso, esto constituye una herramienta muy importante a la hora de realizar cualquier implementación en una plataforma embebida de este tipo.

• **Inicialización de los pines de I2C**
i2c_init() // Configura los pines SCL y SDA como salidas de colector abierto y constantes de retardo.
• **Envío de la condición de Comienzo**
i2c_start_tx() // Inicializa la transmisión mediante el envío de un comienzo (start)
i2c_startw_tx() // Inicializa la transmisión mediante el envío de un Start (S)
// Inserta un delay después del pulso S
• **Envío de un Byte de datos**
i2c_write_char() // Envía 8 bits al dispositivo esclavo
i2c_wr_wait() // Reintenta escribir una variable char hasta que el esclavo responde
• **Escucha de un ACK**
i2c_check_ack() // Chequea si un dispositivo esclavo pone un bajo en SCL
• **Recepción de un Byte de datos**
i2c_read_char() //Lee 8 bits de datos de un dispositivo esclavo
• **Envío de un ACK**
i2c_send_ack() // Envía una secuencia ACK al esclavo.
i2c_send_nak() // Envía una secuencia NAK
• **Envío de una condición de parada**
i2c_stop_tx() // Envío de P (STOP) al esclavo

Tabla 2 Principales elementos del API de gestión de bus I2C

Implementación de un servicio Web utilizando Rabbit Web

El servidor HTTP es el encargado de resolver las solicitudes GET y POST direccionada a la dirección IP y puerto donde está escuchando el servidor [H01]; las mismas son respondidas mediante páginas HTML generadas dinámicamente. De esta forma un usuario puede entre otras facilidades:

- Configurar distintos parámetros de funcionamiento como dirección IP y usuario del dispositivo
- Obtener registros parciales de las muestras de temperatura y valores analógicos de tensión propiciados ambos por el ADC.
- Obtener valores de telemetría DR del payload.
- Enviar comandos LLC al payload.

```
#define TCPCONFIG 0
#define USE_ETHERNET                1
#define MY_IP_ADDRESS   "192.168.1.54"
#define MY_NETMASK      "255.255.255.0"
#define MY_GATEWAY                  "192.168.1.1"
#ximport "index.html"     index_html
#ximport "rabbit1.gif"    rabbit1_gif
#memmap xmem
#use "dcrtcp.lib"
#use "http.lib"
const HttpType http_types[] =
{   { ".html", "text/html", NULL},          // html
   { ".gif", "image/gif", NULL} };
const static HttpSpec http_flashspec[] =
{ { HTTPSPEC_FILE,  "/",            index_html,    NULL, 0, NULL, NULL},
   { HTTPSPEC_FILE,  "/index.html",    index_html,    NULL, 0, NULL, NULL},
   { HTTPSPEC_FILE,  "/rabbit1.gif",   rabbit1_gif,   NULL, 0, NULL, NULL} };
main()
{sock_init();
        http_init();
  while(1){
        http_handler(); } }
```

Ejemplo 2 Implementación de HTML en RCM3365

En el Ejemplo 2 se presenta de manera simplificada la estructura del software en Dynamic C implementada para montar un servicio web en un módulo RCM3365.
Para el desarrollo de la página HTML se utilizó el software Macromedia DreamWeaver 8; en realidad cualquier editor puede ser utilizado con éste propósito puesto que el diseño de las páginas es relativamente sencillo y el volumen de información desplegada bajo.

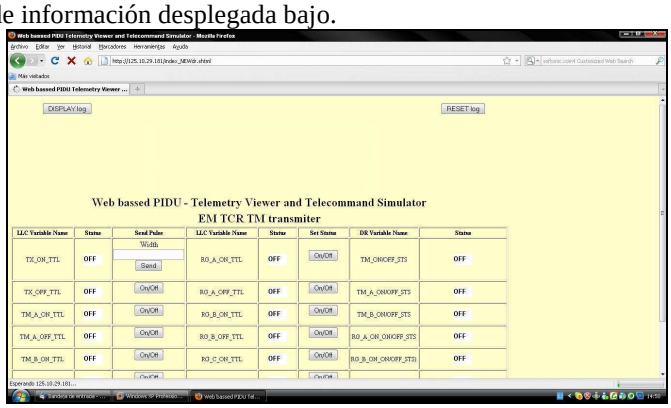

Figura 3 Ejemplo de interfaz Web embebida

El principal uso de este servicio es proveer una interfaz de usuario que se capaz vincular las variables leídas desde el bus I2C del ADC provenientes de los canales AN y PT, y procesar las mismas para que puedan ser mostradas al usuario en formato de página HTML.

Implementación en RabbitWeb

RabbitWeb es una facilidad que ofrece Dynamic C, con la cual se puede crear una interfaz web para los dispositivos de la familia Rabbit. La principal ventaja respecto a servidores HTTP convencionales consiste en la eliminación de la necesidad de utilizar programación CGI [G01]. El resultado obtenido es una página Web que puede ser accedida desde cualquier PC visible mediante ruteos TCP/IP de manera que permita al usuario enviar comandos LLC y leer telemetría DR y analógica. En la Figura 3 puede apreciarse el aspecto final de la página que relaciona al usuario con la electrónica del payload en la etapa de integración del modelo de ingeniería. Como fuere mencionado anteriormente, la recolección de telemetría y el envío de telecomandos debe ser llevado a cabo por el SIMPLAT que emula la aviónica del satélite y no su carga útil. Al realizarlo es necesario implementar un protocolo de comunicación entre éste y la plataforma embebida que replique la que tendrá luego en la plataforma de ingeniería y de vuelo.

Protocolo de comunicación para una aplicación remota, vía TCP/IP.

El protocolo de comunicación entre la aplicación embebida y el simulador de plataforma se implementa sobre un socket TCP/IP, en el campo de datos del segmento, donde la plataforma RCM3365 operará como cliente. El control de flujo propio de TCP/IP garantiza una entrega de paquetes en orden y libres de errores, lo cual constituye una ventaja para la integridad de la comunicación. El SIMPLAT inicia la conexión con la plataforma embebida enviando un paquete "*Request*" (*TM* o *TC)* formado por 4 Bytes (32bits), que incluye: número de secuencia, un identificador de unidad (Payload Nominal o Redundante) y un campo de datos, donde se especifica el comando "*TC Variable"* que el SIMPLAT desea enviar o el valor de telemetría "*TM Variable* "que se pretende leer. Para una solicitud de telemetría "TM Request", la plataforma embebida devuelve un paquete "*TM Response"* con el mismo número de secuencia con el cual se originó la petición y

el campo "*TM Value*" con un valor de TM DR, AN o PT según lo requerido, de tratarse de una solicitud de telecomandos "*TC Request*", la plataforma, retorna un paquete "*TC Response*" (eco), tras haber ejecutado el comando indicado en los campos "*LLC Status*" y "*TC Variable*". En caso de que la conexión no pueda ser establecida dentro de 10 intentos se asumirá una condición de error en el SIMPLAT. Las principales ventajas del protocolo bidireccional implementado son simplicidad y robustez. La estructura de los paquetes de datos del protocolo de comunicación puede ser observada en la Tabla 3.

TM REQUEST / RESPONSE (DR, AN,PT)			
Header (1 Byte)	Data Field (3 Bytes)		
N° Seq	Unit ID (1 Byte)	TM Value* (1 Byte)	TM Variable (1 Byte)
0x00 - 0xFF	Nominal - 0x00	DR - (0x00/0x01)	DR - (0x00-0x0F)
	Redundant - 0x01	AN - (0x00-0xFF)	AN - (0x10-0x13)
		PT - (0x00-0xFF)	PT - (0x14-0x17)

*Para el paquete TM Request, TM Value = 0xoo
TM Value es completado por la plataforma embebida en el paquete TM Response

TC REQUEST / RESPONSE** (LLC)			
Header (1 Byte)	Data Field (3 Bytes)		
N° Seq	Unit ID (1 Byte)	LLC Status (1 Byte)	TC Variable (1 Byte)
0x00 - 0xFF	Nominal - 0x00	ON 0x01	LLC - (0xF0-0xFF)
	Redundant - 0x01	OFF 0x00	

**El paquete TC RESPONSE es igual al paquete TC REQUEST

Tabla 3 Estructura de los paquetes de datos del protocolo de comunicación Implementado

Cada 500 mSeg el SIMPLAT establecerá una comunicación con la plataforma embebida realizando el envío de telecomandos (*TC Resquest*) y/o pedidos de telemetría (*TM Request*) que estén determinados en ese ciclo. Se ha implementado en la memoria de programa del módulo RCM3365 una tabla de asignación de telecomandos y variables de telemetría, que vincula TC y TM con valores hexadecimales. Estos valores son enviados en el campo "*TM Variable*" o "*TC Variable*" de los paquetes "*TM Request*" y "*TC Request*" implementados para este protocolo de comunicación En la figura 4 se puede observar el funcionamiento del protocolo.

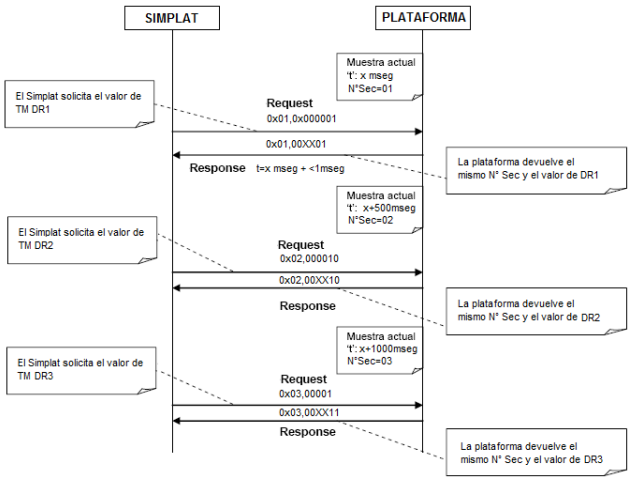

Figura 4 Funcionamiento del Protocolo de Comunicación Simplat-Plataforma

Si la plataforma embebida encuentra un error en el sub-campo "*TM Variable*" o "*TC Variable*" del campo de datos de un paquete "*Request*", devolverá un paquete "*retry*" con número de secuencia 0x00 y el campo de datos con un valor 0x000000, lo cual indicará al SIMPLAT un pedido de retransmisión. Si la plataforma no devuelve un número de secuencia esperado por el SIMPLAT, se generará automáticamente desde el SIMPLAT un reenvío de la solicitud TM o TC según corresponda. Una vez finalizado el período de comunicación con la plataforma, la misma estará esperando hasta que comience un nuevo ciclo de Telecomandos y pedidos de Telemetría que esté determinado por parte del SIMPLAT.

Validación y Verificación

La *validación y verificación* (V&V) del diseño requiere consideraciones especiales. A nivel de verificación se procede a revisar unitariamente cada interfaz en particular, los niveles de tensión, corrientes, tiempos de transitorios, flancos y tolerancias requeridas por el fabricante del payload las que son estrictas.
Para los ensayos a nivel unitario de recepción de comandos y emisión de telemetría mediante "stubs" que simularon el payload a ser controlado de acuerdo a la especificación de su fabricante. Posteriormente se realizó la integración con el payload de manera de verificar la recolección de telemetría DR. Se probaron 100 casos de estados de cambios de variables controladas por un operador y su correcto reflejo en la información HTML proporcionada por la interfaz Web.
Una estrategia similar se utilizó para verificar unitariamente los comandos LLC controlando con instrumental apropiado que los parámetros eléctricos y de tiempo estuvieran dentro de especificación al mismo tiempo que se verifica que las lecturas son correctamente reflejadas en la GUI. Los resultados del test fueron documentados e incluidos en la documentación del sistema.
La integración con la PC industrial se resuelve también con un criterio de verificación unitaria primero para luego ser validado con mediante ciclos extendidos de mediciones del SIMPLAT, verificando la conectividad integrada entre la plataforma embebida bajo desarrollo y el simulador de la plataforma.
La integración final del EGSE con el subsistema ha quedado para una etapa posterior del proyecto y no se ha completado al momento de formular esta publicación.

Resumen del proyecto

La duración total del proyecto implicó aproximadamente un año de trabajo desde la definición de requerimientos a nivel sistema, hasta la concepción de un entregable como el descripto en este trabajo.
El enfoque utilizado cumple con todas las etapas de un ciclo de vida de tipo iterativo incluyendo requerimientos, diseño, construcción, etapas de testing, integración de hardware y software y validación del EGSE. El esfuerzo total de desarrollo es del orden de 3 personas-año donde algunas etapas fueron tercerizadas tales como la fabricación de PCB, poblado de la placa, soldadura por horno y por olas, este diseño constituye una pieza de relevante importancia en la cadena de tests de integración y ensayo de un satélite geoestacionario.

Conclusiones

Este trabajo refleja la creación de una plataforma embebida como parte del proyecto de desarrollo de un satélite geoestacionario. Este tipo de desarrollo apela a la sencillez de un entorno HTML/Web para implementar la interfaz de usuario. El proceso de desarrollo se sostiene en prácticas de ingeniería de software establecidas tales como ciclo de requerimientos, diseño por etapas, establecimiento de línea de base de configuración e inspecciones confirmando la utilidad de estas técnicas en el desarrollo de sistemas embebidos de esta complejidad. Queda confirmada la hipótesis de las ventajas relativas de la plataforma elegida respecto a un desarrollo funcionalmente equivalente basado en arquitecturas FPGA. Estas ventajas se ven en aspectos tales como capacidad de integración de hardware y software como una solución integrada así la posibilidad de actualización flexible del *firmware*, lo cual transforma al módulo desarrollado en una estructura tolerante al cambio de requerimientos de diseño. Las aplicaciones de este módulo en las etapas de

integración y ensayo del satélite geoestacionario son diversas, de acuerdo con los requerimientos de cada subsistema a ser integrado, razón por la cual, para cada uno de ellos el software embebido en el módulo RCM3365 deberá ser actualizado y modificado. Este desarrollo se extenderá entonces hasta fines del 2011, fecha estipulada para finalizar con la integración del modelo de ingeniería del satélite.
Una ventaja del diseño utilizado fue desarrollar en un entorno de programación amigable, basado en las estructuras del lenguaje C, lo cual permitió abordar la tecnología con una curva de aprendizaje modesta comparada a la que hubiera sido necesaria para adquirir los conocimientos de VHDL para haber podido utilizar una implementación en FPGA.
La disponibilidad de gran cantidad de ejemplos, notas de aplicación y manuales del fabricante sirvieron de guía en este desarrollo y una importante cantidad de aplicaciones similares en el mundo de los entornos embebidos indicaban que la alternativa elegida fuera la correcta.
Como aprendizaje se puede mencionar que el compilador de Dynamic C 9.62 no es totalmente compatible con ANSI C y si bien puede ser portado entre diferentes microprocesadores de la familia Rabbit, se requiere un trabajo extra para implementarlo. Usos específicos como RabbitWeb con Z Server son propietarios y no son reutilizables en otros microprocesadores o plataformas embebidas.
Quedará como trabajo a futuro culminar con la integración del payload al modelo de ingeniería y actualizar el software embebido de cada módulo de acuerdo a los requerimientos de cada subsistema a ser integrado.
La integración final del EGSE con el subsistema será realizada a fines de Julio del corriente año y se prevé hacer en primera medida una recepción e inspección visual del subsistema montaje en el modelo de ingeniería del satélite y posterior encendido, apagado del transmisor, receptor. Se busca en primera instancia verificar las funcionalidades vitales. El plan de integración y ensayo tiene estipulado en la segunda etapa conectar un simulador de canal, y realizar pruebas específicas, para lo cual la plataforma aquí desarrollada es de vital importancia para el control de la carga útil.

Referencias

[C01] Caprile S.R.: Desarrollo con procesadores y módulos Rabbit 2ª Ed. (2005) ISBN: 987-21834-2-2
[C02] Caprile S.R.: EL camino del conejo. 2ª Ed. ISBN: 978-987-1301-28-7
[C03] Conversor Analógico Digital AD7993AR- http://www.analog.com
[C04] Colofello J. S."Introduction to Software V&V" SEI Curriculum Module SEI-CM-13-1.1 1988
[D01] Daughtry J.M.,Farooq U et al "API Usability:Report on special group interest at CIDH" SIGSoft V34N4 pp 27-29 2009
[G01] Gundavaram, S "Common Gateway Interface" O'Reilly Open Book Project
[H01] Hyder K,Perrin B "Embedded Systems Design using the Rabbit 3000 Microprocessor" ISBN: 0-7506-7872-0
[I01] I2C: Inter-Integrated Circuit (http://www.nxp.com/acrobat_download/literature/9398/39340011.pdf)
[P01] Pressman, R. "Software Engineering: A Practitioner's Approach",7th Ed McGraw-Hill ISBN 0071267824
[R01] Rabbit RCM3365 Data sheet- http://www.rabbit.com/products/rcm3365/rcm3365.pdf
[R02] RabbitWeb - http://www.rabbit.com/documentation/docs/modules/RabbitWeb/RabbitWeb.pdf
[R03] Rabbit Technical Notes and white Papers http://www.rabbit.com
[R04] Rabbit Referencia manual librería I2C.lib- http://www.rabbit.com/documentation
[R05] Rabbit Software Dynamic C version 9.6 http://www.rabbit.com/support