"ESTACIÓN DE MONITOREO"

Descripción general

Se utilizarán 2 módulos LPCXpresso para comunicar datos en forma inalámbrica mediante protocolo 802.15.4. Cada módulo corre sistema operativo FreeRTOS portado para el microcontrolador LPC1769. Un módulo se considera **remoto** y debe transmitir datos variables al otro módulo considerado **local,** que deberá arbitrar el flujo de los datos y subirlos a una página web embebida en el mismo.

Descripción detallada

El módulo local solicita al módulo remoto que transmita el valor actual de un sensor. Mediante una interfaz web un usuario humano puede configurar que los pedidos se hagan en forma sincrónica o una única vez en forma asincrónica. El módulo remoto responde cada pedido con el envío del valor actual del sensor.

Mediante un pulsador en el módulo remoto se dispara un evento "alarma" que se debe comunicar al módulo local. El módulo remoto en modo "alarma" descarta cualquier pedido de datos y sólo responde a un comando "reset alarma" enviado desde el módulo local con el cual resume el funcionamiento normal.

El módulo local al recibir el evento "alarma", debe interrumpir el pedido de datos si estuviera en modo sincrónico y mostrar un cartel indicando la condición de alarma en la interfaz web a la espera de la intervención humana. Un usuario humano debe oprimir un botón en la interfaz web para enviar el comando "reset alarma" al módulo remoto y resumir el funcionamiento normal del sistema.

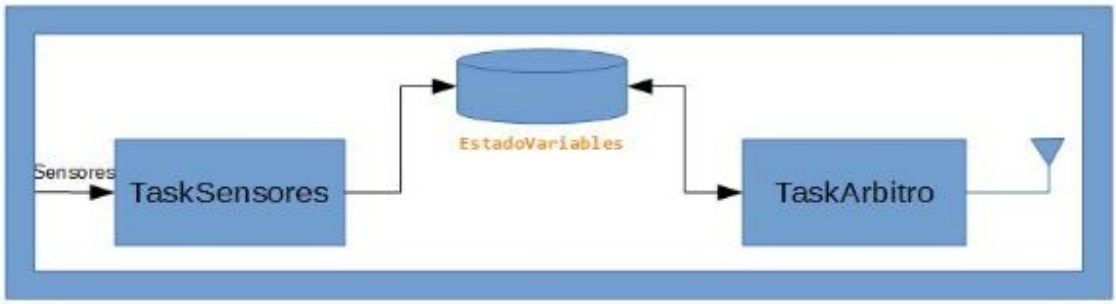
Módulo remoto (Sensores)

Cuenta con dos entradas variables a saber,

E1: Potenciómetro (fuente de datos variable)
E2: Pulsador ("Alarma")

El protocolo de comunicación 802.15.4 se implementa con un transceptor basado en el chip cc2520 de *Texas Instrument* (gentileza del Ing. Pablo Ridolfi). Se utiliza una interfaz serie SPI para comunicar el microcontrolador con el transceptor.

E/S3: Puerto SPI (interfaz directa al transceptor 802.15.4)



Se define una estructura global “EstadoVariables”, en la cual se almacenan los valores de los sensores dentro de una sección crítica, donde no pueden ocurrir cambios de contexto expropiativos, es decir, la tarea no puede ser suspendida por el scheduler, esto garantiza que los datos sean válidos al momento de solicitarlos, sin la necesidad de emplear varias colas entre las tareas. De esta manera se comunican los módulos mediante una única variable.

Este módulo contiene las funciones de setupHardware(), taskSensores, taskArbitro y vTaskStartScheduler().

SetupHardware (void) inicializa los GPIO (General Port Input/Output) para el led y el pulsador, el AIN (Analog Input) para el potenciómetro y el módulo de radio cc2520.

TaskSensores actualiza periódicamente la variable “valorPote” con la función AINread de la API-CAPI (gentileza de Alejandro Celery) de una máquina de estado finito “FSM_Pulsador” que detecta la condición del pulsador con antirrebote, en la cual se modifica el “estadoPulsador” a “ACTIVO” dentro de una sección crítica, que representa la “alarma”.

TaskArbitro es la encargada de procesar los paquetes recibidos del módulo local, basada en otra máquina de estado finito “FSM_Paquete” de dos estados: “IDLE” y “ALARM_SET”. Parte de la condición que si el “estadoPulsador” es “ACTIVO” en estado “IDLE” genera una entrada a la máquina de estado de tipo “ALARM_SET”. Las demás entradas dependen del RFPacket_type (“DATA_REQUEST” y “ALARM_RESET”) definido en la capa de control de acceso al medio.

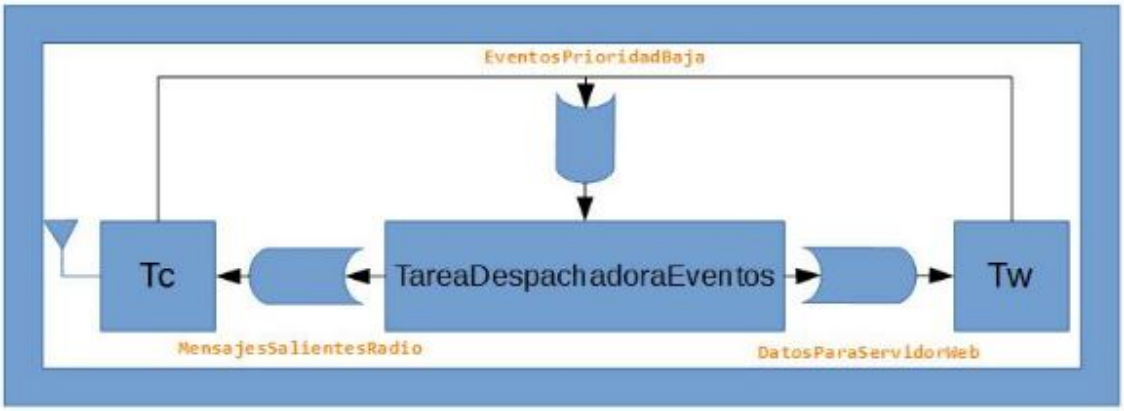
Módulo local (Control de flujo -- Webserver)

El módulo local debe implementar un stack TCP/IP, en particular se trabaja con un port de lwIP para LPC1769 que corre sobre freeRTOS. Se utiliza una página web para publicar los datos recibidos del módulo remoto con una interfaz para permitir a un usuario humano controlar variables globales del sistema, en particular el período con el cual los datos son requeridos y el envío del comando “reset alarma”. La lectura de los datos introducidos por el usuario se hace mediante un “GET request”, mecanismo que consiste en solicitar una página web al webserver y se aprovecha dicha solicitud para “colar” datos adicionales. Es en estos datos que se introduce el “período de solicitud de datos” y se modificó el código del webserver para que capture esta información adicional y la asigne a una variable global y de este modo sea utilizada.

Se utiliza el mismo transceptor basado en el chip cc2520 que el módulo remoto para la comunicación inalámbrica.

E/S1: Puerto ethernet (“PC externa” --administración del webserver)
E/S2: Puerto SPI (interfaz directa al transceptor 802.15.4)

El módulo local principalmente es un manejador de eventos, el cual se basa en el código propuesto por Alejandro Celery (Control Casa).



El módulo cuenta con tres tareas:

TareaCommRadio, encargada de transmitir o recibir información proveniente desde el transceptor basado en el chip cc2520

TareaDespachadoraEventos, que contiene la lógica principal del módulo.

TareaWeb, que contiene la lógica del manejo del stack TCP/IP lwIP.

La tarea <TareaDespachadoraEventos> es una máquina de estados finitos que contiene los diferentes estados del sistema enumerados a continuación.

- sARBITRO_INACTIVO
- sARBITRO_MIDIENDO
- sARBITRO_ALARMA
- sARBITRO_IDLE

El estado de la máquina cambia dependiendo de las señales de los eventos de entrada provenientes de la tarea <TareaCommRadio> o de las señales de los eventos desde la tarea <TareaWeb>.

Los salidas de la máquina de estados (mensajes de salida) se maneja mediante colas hacia la diferentes tareas.

El estado <sARBITRO_MIDIENDO> tiene un Timer. Cuando pasa el tiempo de desborde del timer simplemente envía la petición de información a la tarea <TareaDespachadoraEventos>, esto quiere decir que esta tarea cuando el sistema se encuentra en el estado <sARBITRO_MIDIENDO> maneja el periodo de petición de información del sistema.

Estructura del Repositorio

El código se desarrolló en forma colaborativa utilizando un repositorio público en bitbucket.org

https://bitbucket.org/rtos-fiuba/tp

ControlCasa contiene el firmware para el módulo remoto.

ProyectoRadios contiene el firmware para el módulo local