Universidad de Buenos Aires
Facultad de Ingeniería

Seminario de Sistemas Embebidos

Cortacorriente para Equipo Autónomo de Red

2^{do} cuatrimestre de 2012

Alumnos: Matías Petrella (68405)
Patricio Bos (81163)
Fecha: 22/02/13

Índice

1. Introducción

Problema inherente a los dispositivos autónomos de red

Los equipos de red "autónomos" –es decir, aquellos que en condiciones normales de trabajo no interactúan con humanos, tales como: switches, routers, camaras IP, servidores no asistidos, almacenamiento RAID directo a red, puntos de acceso inalámbricos, etc.– son susceptibles de interrumpir su servicio en forma repentina. Normalmente la solución a este problema es tan simple como cortar el suministro de corriente durante unos segundos y luego restituirlo. Este procedimiento fuerza al equipo "colgado" a reiniciarse, en cuyo proceso se restablecen las variables de trabajo y (posiblemente) se realiza un autodiagnostico tras lo cual vuelven a prestar servicio en forma normal.

Las ventajas de disponer de un equipo igualmente autónomo que pueda detectar estas salidas de servicio y accionar sobre el suministro eléctrico son múltiples:

- detección temprana de fallas
- aumento en la disponibilidad de los servicios
- ahoro en los costos de mantenimiento
- posibilidad de instalación en zonas de dificil acceso
- etc.

El LPCXpresso es un toolchain completo de desarrollo y evaluación para microcontroladores de NXP. Se compone de los siguientes elementos:

- LPCXpresso IDE y herramientas de desarrollo
- LPCXpresso target board (stick)
- LPCXpresso Baseboard EA-XPR-021

Se propone utilizar el entorno de desarrollo LPCXpresso para implementar un **Corta-corriente para equipos autónomos de red**. Se pretende desarrollar una interfaz web para configurar el sistema

El proyecto se segmenta en los siguientes 10 items.

1. Determinar que stack de TCP/IP se va a utilizar y probarlo. (LwIP, uIP, NicheLite, etc)
2. Diseñar e implementar una API por sobre el stack para abstraerse del mismo y poder realizar los pings.
3. Diseñar e implementar la página HTML para administrar el sistema.
4. Utilizar el stack seleccionado para poner en marcha un web server y montar la página.

5. Implementar o buscar implementaciones de la funcionalidad PING y probarla.
6. Diseñar e implementar una API para poder manejar las salidas digitales y otra para el Watchdog Timer.
7. Implementar la funcionalidad para chequear paginas web por telnet y actuar en consecuencia. (ó implementación del PCB para el módulo de control)
8. Implementar una interfaz de potencia.
9. Combinar los items anteriores para implementar el sistema, pensar en la utilización de un RTOS.
10. Entrega de Informe Final.

$\left(\frac{x^2+15}{10\rho}\right)$

2. Desarrollo

2.1. PCB interfaz de potencia

Para que el MCU pueda administrar el suministro eléctrico de los equipos autónomos de red, se implementa una interfaz de potencia 5VDC/220VAC de 4 canales. Para tal fin se utlizan optoacopladores MOC3021 y triacs BTA12 aislados, de propositos generales.

La conmutación se realiza sobre el neutro de la línea de 220VAC, mientras que la carga se conecta al vivo.

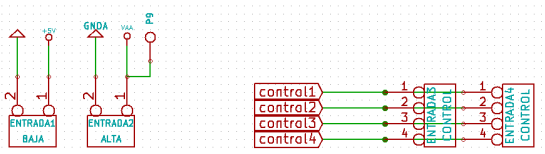

Figura 1: conectores

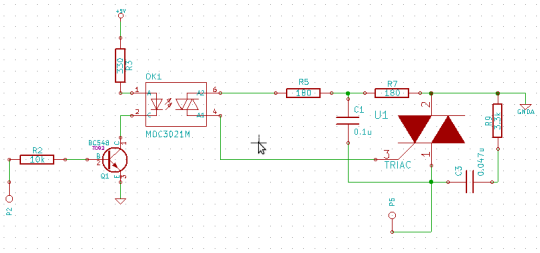

Figura 2: interfaz 5VDC/220VAC

Al desconocerse el tipo de carga que será conectada deben tomarse precausiones para proteger el triac y el optoacoplador. Se utilizan dos redes snubber compuestas por un capacitor de $0{,}047\mu F$ y un resistor de $3{,}3k\Omega$ para el triac y un capacitor de $0{,}1\mu F$ y un resistor de 180Ω para el optoacoplador.

Para aislar la excitación del optoacoplador de la capacidad de suministro de corrientes del MCU, la misma se realiza con un transistor bipolar NPN BC548.

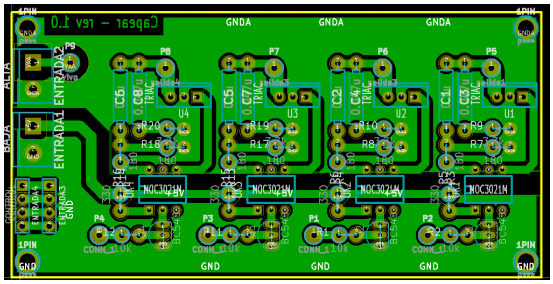

Figura 3: PCB layout

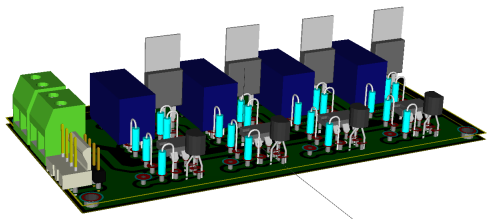

Figura 4: vista 3D

3. algo

4. Conclusiones

Se pudo desarrollar un sistema que cumpla con todos los objetivos propuestos haciendo uso extensivo de las funciones propias del *port* de freeRTOS para LPC1769, así como también de la librería CMSIS provista por NXP.
Se utilizaron funciones ISRsafe dentro de las rutinas de interrupción y se forzó el llamado al schedule al finalizar las mismas para asegurar el cambio de contexto como práctica de buena programación. En cuanto a la elección de las prioridades, se optó por asignar prioridades distintas a cada tarea. La mas baja a la tarea periódica y la mayor a la tarea asociada al RTC. No obstante lo cual, todas las tareas hacen uso de la UART cuya disponibilidad esta restringida a un mutex. Por este motivo ninguna tarea es *pre-empted* ya que tomar el mutex le asegura permanecer en estado running.