Seminario de Electrónica: Sistemas Embebidos - Trabajo Práctico N° 2

LPC43xx Entradas y Salidas (Digitales) de Propósito General (GPIO) – Diagrama de Estado

Objetivo:

- **Uso del IDE** (edición, compilación y depuración de programas)
- **Uso de GPIO & Diagrama de Estado** (manejo de Salidas y de Entradas Digitales en Aplicaciones)
- **Documentar lo que se solicita en c/ítems**

Referencias (descargar del Campus Vitual del curso a fin de usarlas durante la realización del TP)**:**

- **Diagrama de Estado:** http://campus.fi.uba.ar/mod/resource/view.php?id=51884
- **DE-LPCXpresso & Yakindu SCT:** http://campus.fi.uba.ar/mod/resource/view.php?id=79378
- **LPC435X_3X_2X_1X Product Data Sheet:** http://campus.fi.uba.ar/mod/resource/view.php?id=28519
- **LPC43XX User Manual** (Chapter 1, 18 & 19)**:** http://campus.fi.uba.ar/mod/resource/view.php?id=77765
- **EDU-CIAA-NXP (web site):** http://proyecto-ciaa.com.ar/devwiki/doku.php?id=desarrollo:edu-ciaa:edu-ciaa-nxp
- **EDU-CIAA-NXP (esquemático):** http://www.proyecto-ciaa.com.ar/devwiki/lib/exe/fetch.php?media=desarrollo:edu-ciaa:edu-ciaa-nxp:edu-ciaa-nxp_color.pdf
- **EDU-CIAA-NXP (pinout):** http://proyecto-ciaa.com.ar/devwiki/lib/exe/fetch.php?media=desarrollo:edu-ciaa:edu-ciaa-nxp_pinout_a4_v4r2_es.pdf

1. Uso del **IDE** (Integrated Development Environment) **LPCXpresso**
    a. En TP1.1.a ya se Registró, Descargó, Instaló, Ejecutó y Licenció **LPCXpresso IDE v8.2.0** (o posterior)
        i. En TP1.1.b ya instaló **LPCXpresso** y agregó el plug-in **OpenOCD Debugging**
        ii. Dentro de **LPCXpresso**, agregar el plug-in **Yakindu StateChart Tools**
        Menú **Help → Install New Software …** Work with: http://updates.yakindu.org/sct/mars/releases/
        Seleccione el plug-in y luego siga las instrucciones del asistente (**Yakindu SCT**)
        iii. Antes de ejecutar asegúrese tener conectada la placa **EDU-CIAA-NXP** a su PC (recuerde conectarla **siempre al mismo puerto USB**) a través de la interfaz **Debug**
            1. Seleccionar como nombre de Workspace: **workspace-EDU_CIAA_NXP-TP2**
            2. Mediante **Import project(s)** de los archivo: **LPCXpresso-Yakindu SCT-Examples.zip** importar los proyectos:
                a. **lpc_chip_43xx** (librería p/chips LPC43xx de NXP)
                b. **lpc_board_nxp_lpcxpresso_4337** (librería p/placas NXP)
                c. **periph_statechart** (ejemplo de aplicación)
            3. Mediante **Project** → **Clean** → **OK** (Build the entire workspace) compilar los proyectos importados (en Debug se genera el archivo: **periph_statechart.axf**)
            4. Verifique tener en la carpeta: **periph_systick/example/** el archivo:
                a. **lpc4337.cfg** (opciones de configuración de OpenOCD)
            5. Copiar la configuración de **Debug** de **periph_systick** (ejemplo de aplicación) y adecuarla a **periph_statechart**
            6. Ejecute la **secuencia de comandos**: Clean **periph_statechart** -> Build **periph_statechart** -> Debug **periph_statechart** -> Ejecutar **periph_statechart** (ejemplo de aplicación)
            7. **Documentar** mediante tablas c/texto e imágenes la estructura de **archivos**, su tipo/contenido (especialmente **readme.txt**) de c/proyecto importado
    b. **Documentar** mediante tablas c/texto e imágenes la secuencia de **funciones** invocadas durante la ejecución del ejemplo de aplicación, en qué archivo se encuentran, su descripción detallada, qué efecto tiene la aplicación sobre el hardware (identificar circuitos, puertos, pines, niveles, etc.) así como la interacción entre las mismas
    c. **Idem b** pero con **datos** (definiciones, constantes, variables, estructuras, etc.)

2. Uso del **IDE** (Integrated Development Environment) **LPCXpresso &** plug-in **Yakindu SCT**
    a. Verifique tener en la carpeta **periph_systick/example/src/** los archivos:
        i. **prefix.sct** Yakindu SCT **Statechart Model** file
        ii. **pregix.sgen** Yakindu SCT **Code Generator Model** file
    b. Para Editar el modelo: Doble clic sobre **prefix.sct**
    c. Para Simular el modelo: Clic derecho sobre **prefix.sct** -> **Run Us** -> **1 Satechart Simulation**
    d. Para Editar la generación de código: Doble clic sobre **pregix.sgen**
    e. Para Generar el código del modelo: Clic derecho sobre **pregix.sgen** -> **Generate Code Artifacts** (Artifacts => **Prefix.c**, **Prefix.h**, **PrefixRequired.h** y **sc_types.h**)
    f. Para concluir resta **modificar la aplicación** y probar la modificación:
        i. Modificar **main()**, **Systick_Handler()** (otros handlers) a fin de excitar al modelo y atender las operaciones invocadas por el modelo (**interface**: interfaz entre el código generado por el modelo y el escrito por nosotros)

ii. Ejecutar la **secuencia de comandos**: Clean **periph_statechart** -> Build **periph_statechart** -> Debug **periph_statechart** -> Ejecutar **periph_statechart** (ejemplo de aplicación)
iii. **Implementar** el modelo de control de **panel de control de un generador de señales** (tensión de 0 a 10V, frecuencia de 20 a 20.000Hz y 3 formas de señal)

3. **Implementar** el modelo de control de **puerta corrediza** automatizada (motor con movimiento en dos sentidos, sensor de presencia y fines de carrera)

4. **Implementar** el modelo de control de **portón de cochera** automatizado (motor con movimiento en dos sentidos, control remoto de apertura/cierre, fines de carrera y señalización luminosa)

5. **Implementar** el modelo de control de **escalera mecánica** unidireccional automatizada (motor c/movimiento en un sentido y dos velocidades, sensores de ingreso, egreso y señalización luminosa)

6. **Implementar** el modelo de control de **horno microondas** (3 modos de cocción seleccionable por botón de modo, botón de comenzar/terminar y sensor de apertura de puerta)