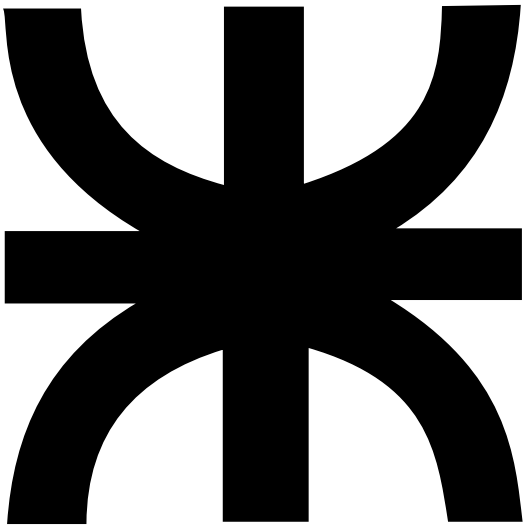
Actividades de investigación y desarrollo relacionadas con microcontroladores ARM aplicado al proyecto RoMAA del CIII

Proyecto: Plataforma Móvil de Arquitectura Abierta



Práctica Profesional Supervisada

Mario Javier Trangoni

Tutor: Ing. Gonzalo Perez Paina

Centro de Investigación en Informática para la Ingeniería

Universidad Tecnológica Nacional

Facultad Regional Córdoba

Índice

1. Introducción

La Práctica Profesional Supervisada, se desarrolló en el marco del proyecto **"Plataforma móvil de arquitectura abierta ROMAA"** del Centro de Investigación en Informática para la Ingeniería (C.I.I.I.), el cual fue presentado en las "V JORNADAS ARGENTINAS DE ROBÓTICA", Universidad Nacional del Sur, Bahía Blanca, 12 al 14 de Noviembre de 2008.
Los autores del mismo son: Ing. David Gaydou, Ing. Gonzalo Perez Paina, Ing. Javier Salomone y el Ing. Guillermo Steiner.

El proyecto surge como respuesta a la necesidad de disponer de una plataforma sobre la cual ensayar y validar las investigaciones llevadas a cabo en el C.I.I.I., las cuales se enmarcan en las áreas de **visión por computadora, control automático y robótica**.

1.1. Descripción del robot:

Se trata de un vehículo de tracción diferecial el cual se encuentra descripto en el Paper *"Plataforma móvil de arquitectura abierta", David Gaydou, Gonzalo Perez Paina, Javier Salomone, Guillermo Steiner. V Jornadas Argentinas de Robótica (JAR'08), Bahía Blanca, 12-14 de Noviembre, 2008*[1]

Se puede observar una vista en la siguiente imágen:

1.2. Descripción breve de los bloques del sistema de tracción:

Consta de dos motores de CC accionados mediante llaves H comandadas desde un controlador a través de señales moduladas en ancho de pulso.

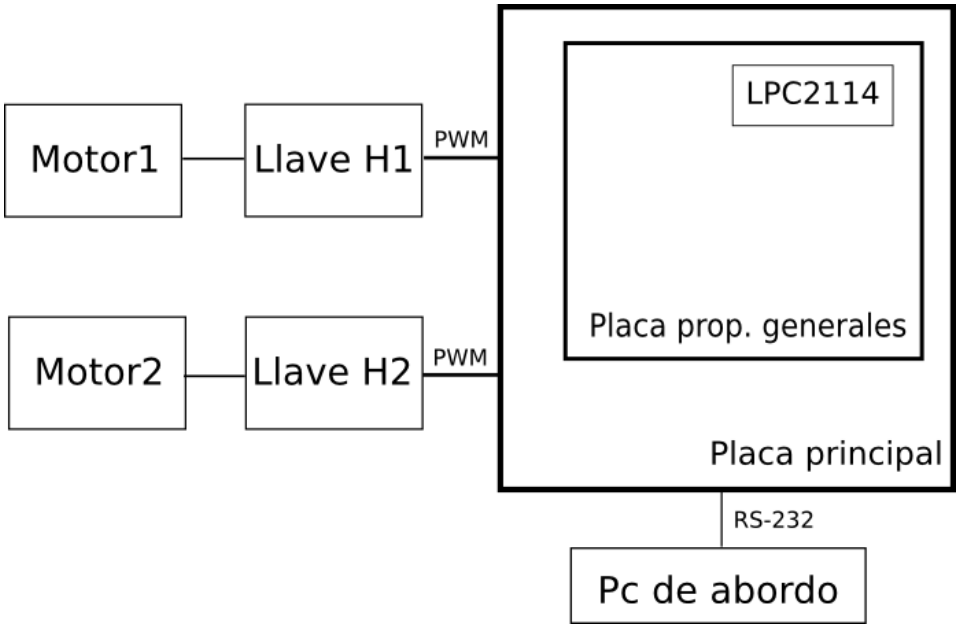
El controlador central está compuesto por dos placas. una principal que da soporte a conectores de acondicionadores de señal y a otra placa de propósitos generales con un microcontrolador LPC2114 de NXP con núcleo ARM de 32 bits.

[1] `http://lcr.uns.edu.ar/jar08/papers/paper34.pdf`

El controlador central se comunica con la computadora abordo a través de un enlace serie RS-232 (alternativamente se puede utilizar RS-485).

En el microcontrolador se procesan los comandos provenientes de la computadora abordo para generar los comandos de tracción, cerrando lazos de control digitales.

Se puede observar un diagrama general en la siguiente imágen:



2. Tareas Desarrolladas

1. **Compilado librería freeModbus bigloop**

   Permite aplicar el protocolo Modbus a la comunicación entre módulo de control y PC de abordo.

2. **Estudio Startup para Toolchain GNUARM micro LPC2114 de NXP**

   Rutinas de inicialización para el microcontrolador del módulo de control.

3. **Ciclo de prensentaciones breves: Micros ARM**

   Presentación de la arquitectura del microcontrolador en el ciclo de charlas del CIII.

4. **Estudio del simulador GDB del Toolchain GNUARM**

   Puesta a punto y testeo de herramienta GNU para debugging de Software embebido.

5. **Actualización proyecto LPC21ISP para programación sobre ARM en Linux**

   Se actualiza el programa que permite el grabado del microcontrolador ARM7 del módulo de control principal.

6. **Estudio configuración PLL para Cristal de 14,7456Mhz en LPC2114 de NXP**

   Modificación del hardware de la placa del microcontrolador y modificación del software de inicialización de perisféricos del microcontrolador.

7. **Estudio proyecto OpenOCD para hacer debug mediante Wiggler JTAG y su documentación**

   Construcción de hardware de interfaz para debugging *in-circuit*. Configuración de herramientas de software.

8. **Estudio de programa arm-elf-gdb para realizar debug de target remoto y su documentación**

   Puesta a punto y testeo de herramienta GNU para debugging de Software embebido.

9. **Modificación archivos de fuentes de proyectos para compilación en** `gcc-arm cross-compiler`

Se adaptan programas realizados sobre plataforma Microsoft a una plataforma libre GNU Linux.

10. **Codificación de lazo de control PID en sistema embebido**

Se implementa el programa del módulo de control para la tracción del vehículo, teniendo en cuenta parámetros de control.

11. **Prueba y tuning de constantes de lazo de control PID**

Se testea el código generado y se le hacen los ajustes necesarios para que funcione de forma aproximada a la teórica.

12. **Ecuaciones para integración del lazo de control PID con control en velocidad (v - ω)**

Agrega al programa principal el control de tracción de modo diferecial que permite el movimiento angular del robot.

13. **Programación de algoritmos para decodificación de encoder incremental mediante módulo de captura integrado en el microcontrolador**

Se implementa y testea el módulo de captura y se lo reemplaza por el método antes utilizado.

14. **Ajuste de Linker-Script para debugging en memoria Flash**

Mejora de codificación de archivos de configuración.

15. **Estudio de herramienta USB-JTAG para OpenOCD**

Busca el reemplazo de la herramienta por puerto paralelo Wiggler por una herramienta más rápida USB.

16. **Simulación de interrupción en entorno gdb insight, implementación de Skyeye**

Estudio de posibilidad de disparar interrupciones en el núcleo durante simulación con las citadas herramientas.

3. Descripción de las tareas

3.1. Compilado librería freeModbus bigloop

Objetivo:

Probar el desempeño en la plataforma del protocolo de comunicación serial Modbus.

Pasos intermedios:

1. Seleccionar un proyecto activo para su implementación.
2. Implementar ejemplos del mismo.
3. Adaptar programas ejemplo a la aplicación en particular.
4. Testear su desempeño.

Conclusiones:

1. Incompatibilidad con la capa BSP del microcontrolador.
2. Velocidad de transmisión insuficiente para la aplicación.

3.2. Estudio Startup para Toolchain GNUARM micro LPC2114 de NXP

Objetivo:

Hacer un análisis de las opciones a configurar, y que incidencia tienen en el comportamiento del programa. Estudiar el linker script y su desempeño.

Pasos intermedios:

Se realizó un estudio por menorizado de las distintas configuraciones de arranque del micro, que se encuentran en este archivo de assembly.
Aquí se configura:

- El vector de interrupción y los manejadores (handler) de interrupción.
- El enganche del PLL, para configurar la velocidad del uC.
- Configuración de los modos de excepción y tamaño en Stack de los mismos.
- El linkeo al código en C que se realiza según la descripción de otro archivo llamado Linker y que depende del Hardware.

Estas hacen al comportamiento futuro del código de aplicación a desarrollar.

Conclusiones:

Quedó conformado un startup optimizado y funcionando con los parámetros a modificar estudiados.

3.3. Ciclo de prensentaciones breves: Micros ARM

Objetivo:

Presentar la arquitectura ARM. Mostrar sus características y funcionalidades. Desarrollo de filminas y exposición oral.

Conclusiones:

Se dejaron en claro las perspectivas de proyectos en marcha. Se muestra información útil para otras personas interesadas en utilizar microcontroladores.

3.4. Estudio del simulador GDB del Toolchain GNUARM

Objetivo:

Evaluar esta herramienta de depuración de software embebido. Definir posibilidades y restricciones para aplicar en los desarrollos de software del centro.

Pasos intermedios:

1. Seleccionar un archivo de configuración ejemplo fijando las versiones de los programas utilizados.
2. Implementar programas ejemplo en el mismo.
3. Adaptar programas ejemplo a la aplicación en particular.
4. Testear su desempeño.
5. Documentar.

Conclusiones:

El simulador GDB, funciona en modo consola, y se observaron las siguientes características:

1. Modos de testeado: Simulador, Debugger.
2. Selección de banderas de compilación adecuadas.
3. Modo de funcionamiento de los Breakpoint, Watchpoint, etc.
4. Imposibilidad de generar señal de interrupciones en el núcleo.

3.5. Actualización proyecto LPC21ISP para programación sobre ARM en Linux

Objetivo:

Compilación de versión 1.52 de LPC21ISP. Evaluar su funcionamiento.

Pasos intermedios:

1. Obtención de fuentes con parches actualizados.
2. Compilación.
3. Testear su desempeño.
4. Documentación.

Conclusiones:

Se agregaron funciones disponibles en el programa de grabación LPC21ISP.

3.6. Estudio configuración PLL para Cristal de 14,7456Mhz en LPC2114 de NXP

Objetivo:

Disminución del error de frecuencia del generador de Baudrate de la UART.

Pasos intermedios:

1. Estudiar la hoja de datos para determinar el valor de configuración correcto.
2. Ajustes de Hardware.
3. Programación del perisférico PLL.
4. Testeo.
5. Documentación.

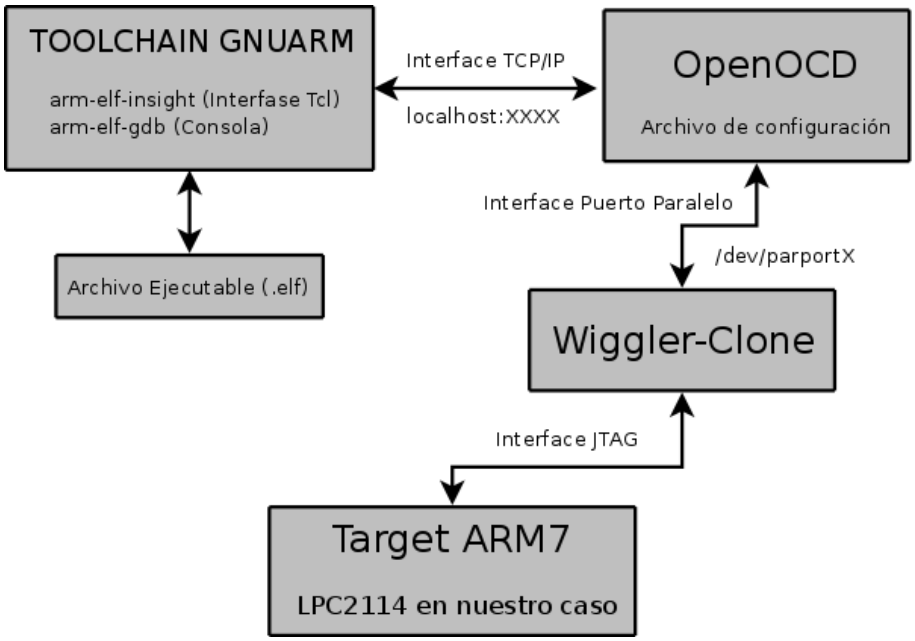
Conclusiones:

La aplicación del cristal de 14,7456 MHz permitió obtener el menor error máximo en todas las velocidades de transmición de la UART.
Se adopta la modificación para el Kit de desarrollo de sistemas embebidos del Centro.

3.7. Estudio proyecto OpenOCD para hacer debug mediante Wiggler JTAG y su documentación

Objetivo:

Según se puede ver en el esquema de la interconexión de los dispositivos:



Se busca realizar el estudio de la aplicación del protocolo JTAG a nuestro sistema mediante la aplicación del adaptador Wiggler. A su vez se busca conectar al Toolchain GNUARM con el adaptador antes mencionado mediante el software OpenOCD que realiza dicha acción.

Pasos intermedios:

1. Instalación y configuración de OpenOCD desde la página de BerliOS, desarrollador del proyecto OpenOCD.
2. Configuración de puerto paralelo.
3. Configuración de hardware para debugging.
4. Configuración de software para debugging.
5. Lograr conexión de los módulos.
6. Prueba en ejemplos simples en memoria RAM.
7. Análisis del desempeño.

Conclusiones:

Se logró comunicar con éxito los módulos.
Del estudio surgieron limitaciones dadas por el hardware, como ser la velocidad de transferencia de datos, compatibilidad con el proyecto OpenOCD y funcionamiento de las lineas de comunicación que controlan la transferencia de datos al microcontrolador.

3.8. Estudio de programa arm-elf-gdb para realizar debug de target remoto y su documentación

Objetivo:

Se busca obtener el máximo de experiencia en el funcionamiento de GDB en su función principal, el debugger del Toolchain GNUARM. Al mismo tiempo que se experimenta se busca comprobar la compatibilidad con **OpenOCD**.

Pasos intermedios:

1. Seleccionar diferentes ejemplos.
2. Implementar arm-elf-gdb.
3. Testear su funcionamiento.

Conclusiones:

Se logró un completo manual para quien quiera utilizar esta herramienta. Se detallaron todas las funciones utilizables, la forma que debe tomar el archivo de configuración principal y los mensajes que se esperan obtener. Asi mismo se obtuvo un manual adicional en complemento de **OpenOCD**.

3.9. Modificación de archivos fuentes de proyectos para compilación en `gcc-arm cross-compiler`

Objetivo:

Reemplazar instrucciones de fuentes en assembler a los proyectos existentes para q compilen con GNUARM.

Pasos intermedios:

1. Reescritura de secciones no compatibles con sintaxis GNUARM.
2. Depurar los errores y compilar el proyecto.
3. Testear funcionamiento.

Conclusiones:

Todos los proyectos compilan en GNUARM.

3.10. Codificación de lazo de control PID en sistema embebido

Objetivo:

Obtener rutinas para lazo de control de velocidad tipo PID aplicadas en programa embebido de control de tracción de RoMAA.

Pasos intermedios:

1. Estudio de los métodos propuestos en la bibliografia recibida.
2. Desarrollo de las fórmulas que describen el control.
3. Codificación del resultado obtenido.
4. Testeo y evaluación del desempeño.

Conclusiones:

Se obtuvieron las rutinas. Se aplicaron en el robot mejorando su desempeño dinámico.

3.11. Prueba y tuning de constantes de lazo de control PID

Objetivo:

Obtener los valores adecuados de las constantes del controlador para un buen desempeño dinámico al movimiento del robot RoMAA.

Pasos intermedios:

1. Estudio de métodos de sintonizaciÂºón de controlador PID disponibles en la bibliografía.
2. Implementación y medición de los resultados obtenidos.

Conclusiones:

Analizando los datos obtenidos de la medición de velocidad de los motores mediante los encoders ópticos. Se ajustaron los parámetros de control para adecuarlos a la dinámica de la planta y obtener un movimiento suave del robot.

3.12. Ecuaciones para integración del lazo de control PID con control en velocidad (v - ω)

Objetivo:

Integrar algoritmo PID en modelo de control v - *omega* (cross-coupling) para tracción diferencial.

Pasos intermedios:

1. Construcción de diagramas en bloque.
2. Desarrollo de las fórmulas que describen el modelo.
3. Codificación del desarrollo obtenido.
4. Documentación.

Conclusiones:

Se obtuvo la expresión matemática completa del modelo de control discreto v - *omega* en PID.

3.13. Programación de algoritmos para decodificación de encoder incremental mediante módulo de captura integrado en el microcontrolador

Objetivo:

Implementar el sistema de decodificación de señales provenientes de los encoders mediante módulo perisférico integrado Capture.

Pasos intermedios:

1. Estudio de las hojas de datos del perisférico.
2. Armado de prototipo para ensayos.
3. Desarrollo y prueba de programas ejemplo.
4. Utilización del módulo en la aplicación del control de tracción de RoMAA.

Conclusiones:

Se resolvió la medición de períodos de pulsos de enconder por medio de hardware obteniendo mejoras en la performance respecto al método de decodificación por software.

3.14. Ajuste de Linker-Script para debugging en memoria Flash

Objetivo:

Obtener los scripts de linkeo apropiados para debug de software embebido residente en memoria flash de *mu*C LPC2114.

Pasos intermedios:

1. Configuración OpenOCD.
2. Escritura de las sentencias de linkeo.
3. Estudio de la conexión del Hardware y verificación de la misma.
4. Prueba en ejemplos simples en memoria Flash con nueva configuración.
5. Análisis de las fallas.

Conclusiones:

La herramienta de interfaz de hardware presenta comportamiento errático. Se recomienda utilizar el modelo ” Dougle“.

3.15. Estudio de herramienta USB-JTAG para OpenOCD

Objetivo:

Estudiar la factibilidad de construir un dispositivo para debug mediante interfase USB-JTAG que sea de libre distribución y uso. A la vez que sea compatible con las herramientas libres utilizadas actualmente (Toolchain GNUARM y OpenOCD) y el hardware disponible (ARM7 LPC2114).

Pasos intermedios:

1. Busqueda de herramientas abiertas en la web.
2. Análisis del Hardware y Software.
3. Estudio de mercado local para factibilidad de compra.

Conclusiones:

Es factible la contrucción del mismo ya que los comerciantes locales disponen de los componentes requeridos, a la vez de que el proyecto seleccionado que pertenece a **Hubert Hoegl** es de libre distribución. Se encontro bibliografía que habla de velocidades de transferencia de datos 150 veces mayores y una compatibilidad absoluta con OpenOCD. Esto último permitiría hacer recidir los programas en memoria flash.

3.16. Simulación de interrupción en entorno gdb insight, implementación de Skyeye

Objetivo:

Generar señales de excepciones nIRQ y nFIQ en modelo de simulación de núcleo ARM.

Pasos intermedios:

1. Estudio de software de simulación de núcleos del cual se seleccionó Skyeye.
2. Puesta en marcha y testeo de ejemplos.
3. Documentación.

Conclusiones:

No se llegó a concluir con una herramienta que permita realizar la simulación. Para utilizar la herramienta SkyEye, se encontró que se debe adaptar la configuración ejemplo a la familia de NXP que estamos utilizando.

4. Valoración de la PPS

Desarrollar la Práctica Profesional Supervisada en el ámbito de un Centro de Investigación enriqueció mi perspectiva sobre la resolución de problemáticas técnicas, mejorando y ejercitando mis hábitos de aplicación del método científico.

Por otro lado valoro en gran medida la oportunidad de tomar contacto con una plataforma de trabajo basada enteramente en software bajo licencia GPL; ahora puedo objetivamente comparar con otras plataformas no libres que había trabajado antes y sacar mis conclusiones sobre las ventajas y desventajas de unas y otras.

A su vez he tomado contacto con profesionales que actualmente estan trabajando en distintos campos de la investigación dentro de mi especialidad, lo cual me brinda un aporte invaluable al momento de proyectar el futuro de mi vida profesional.