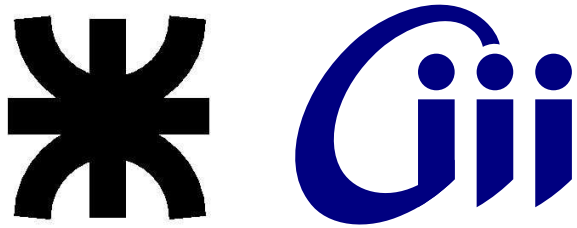
Librerías embebidas para microcontroladores LPC2000 de aplicación en robótica

Perez Paina Gaydou Palomeque Martini

Centro de Investigación en Informática para la Ingeniería
Universidad Tecnológica Nacional, F.R.C.
http://ciii.frc.utn.edu.ar
Córdoba, Argentina



Congreso Argentino de Sistemas Embebidos
2 al 4 de marzo de 2011

Contenido

Objetivos

Generales

Obtener librerías flexibles para el desarrollo de sistemas embebidos en el laboratorio del CIII, aplicables en áreas como robótica, control y sensorística

que brinden

- acceso transparente a los periféricos del μC
- rutinas o algoritmos de uso común

para

- reutilizar código existente
- agilizar el desarrollo de sistemas embebidos
- tener un marco de desarrollo de constante crecimiento

Objetivos

Generales

Obtener librerías flexibles para el desarrollo de sistemas embebidos en el laboratorio del CIII, aplicables en áreas como robótica, control y sensorística

que brinden

- acceso transparente a los periféricos del μC
- rutinas o algoritmos de uso común

para

- reutilizar código existente
- agilizar el desarrollo de sistemas embebidos
- tener un marco de desarrollo de constante crecimiento

Objetivos

Generales

Obtener librerías flexibles para el desarrollo de sistemas embebidos en el laboratorio del CIII, aplicables en áreas como robótica, control y sensorística

que brinden

- acceso transparente a los periféricos del μC
- rutinas o algoritmos de uso común

para

- reutilizar código existente
- agilizar el desarrollo de sistemas embebidos
- tener un marco de desarrollo de constante crecimiento

Objetivos

Generales

Obtener librerías flexibles para el desarrollo de sistemas embebidos en el laboratorio del CIII, aplicables en áreas como robótica, control y sensorística

que brinden

- acceso transparente a los periféricos del μC
- rutinas o algoritmos de uso común

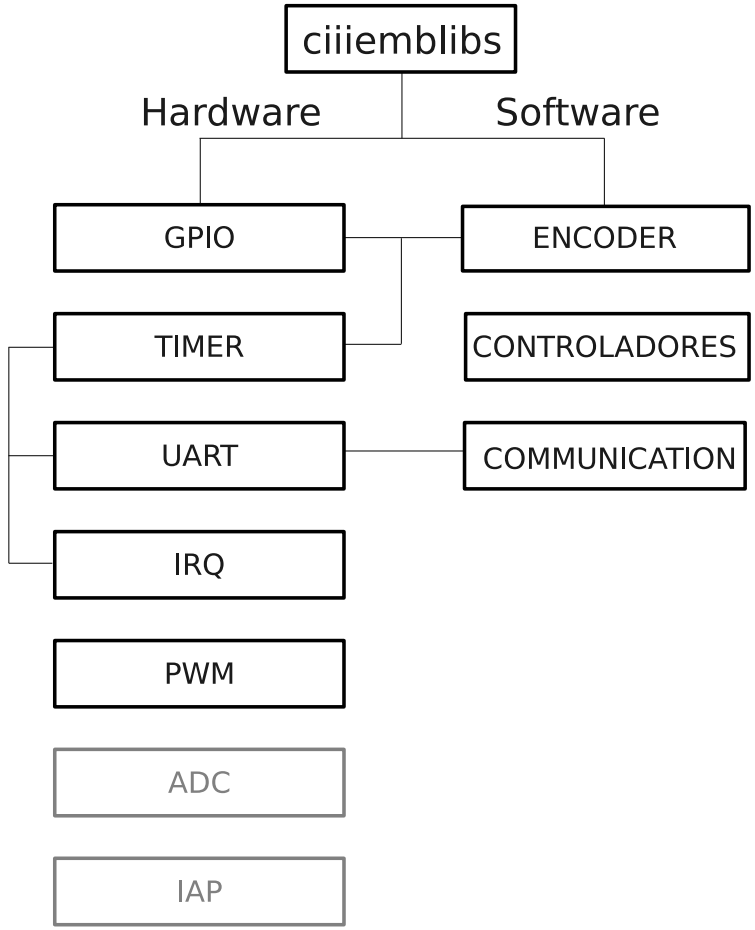
para

- reutilizar código existente
- agilizar el desarrollo de sistemas embebidos
- tener un marco de desarrollo de constante crecimiento

Descripción general

- Desarrolladas en lenguaje ANSI-C
- Separadas en módulos (.c y .h)
- Divididas en dos grandes grupos
  - módulos para periféricos o hardware
  - módulos especiales o de software
- Estilo de nombre unificado

Ejemplos: `gpio_init()`, `pwm_init()`, `com_init()`, `timer_init()`



Módulo para periféricos

Módulo GPIO

Permite configurar y controlar los pines correspondientes a puertos de propósito generales del μC.

Algunas funciones son:
`gpio_init()`, `gpio_set()`, `gpio_toggle()`

Módulo TIMER

Permite configurar y controlar los módulos TIMER del μC. Tiene implementada el modo Capture.

Algunas funciones son:
`timer_init()`, `timer_get()`, `timer_enable_interrupt()`, `timer_disable_interrupt()`

Módulo para periféricos (cont.)

Módulo UART

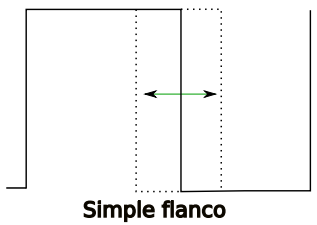
Permite configurar y controlar los módulos UART del μC. Permitiendo elegir el BaudRate, la cantidad de bytes, paridad y bits de stop de la comunicación, y tamaño de la memoria FIFO. También puede interactuar con el Módulo software de IRQ.

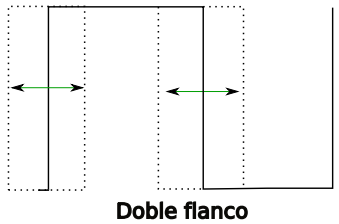
Algunas funciones son:
`uart_init()`,
`uart_set_baudrate()`,
`uart_send_byte()`,
`uart_receive_byte()`

Módulo PWM

Permite configurar y controlar los módulos de PWM del μC.

Algunas funciones son: `pwm_init()`, `pwm_set_on()`, `pwm_set_off()`, `pwm_set_value()`





Módulo para periféricos (cont.)

Módulo IRQ

EXPLICACION

Algunas funciones son:
`irq_vect_init()`,
`irq_vect_enable()`,
`irq_vect_disable()`

Módulos en desarrollo: IAP y ADC

El módulo **IAP** (In-Application Programming) permite almacenar datos en la memoria flash del μC.

El módulo **ADC** (Analog to Digital Converter) permite realizar mediciones sobre señales analógica.

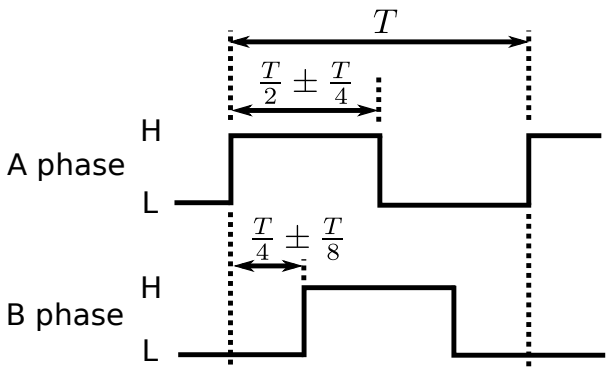
Módulos especiales

Módulo ENCODER

Decodificación de la señal de encoders ópticos incrementales, para determinar

- cantidad absoluta de pulsos
- sentido de avance

Algunas funciones son: `encoder_init()`, `encoder_update()`

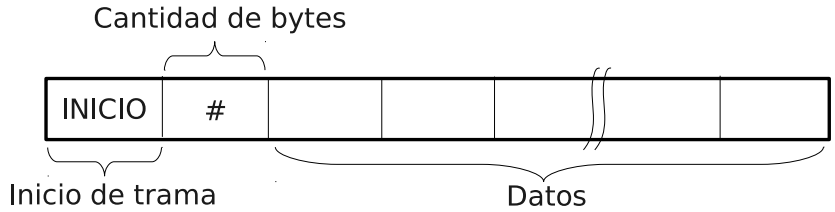


Módulo COMMUNICATION

Comunicación punto a punto entre dos dispositivos

- transferir datos empaquetados
- hace uso de callbacks para...

Algunas funciones son: `com_init()`, `com_send_packet()`, `com_receive_packet()`



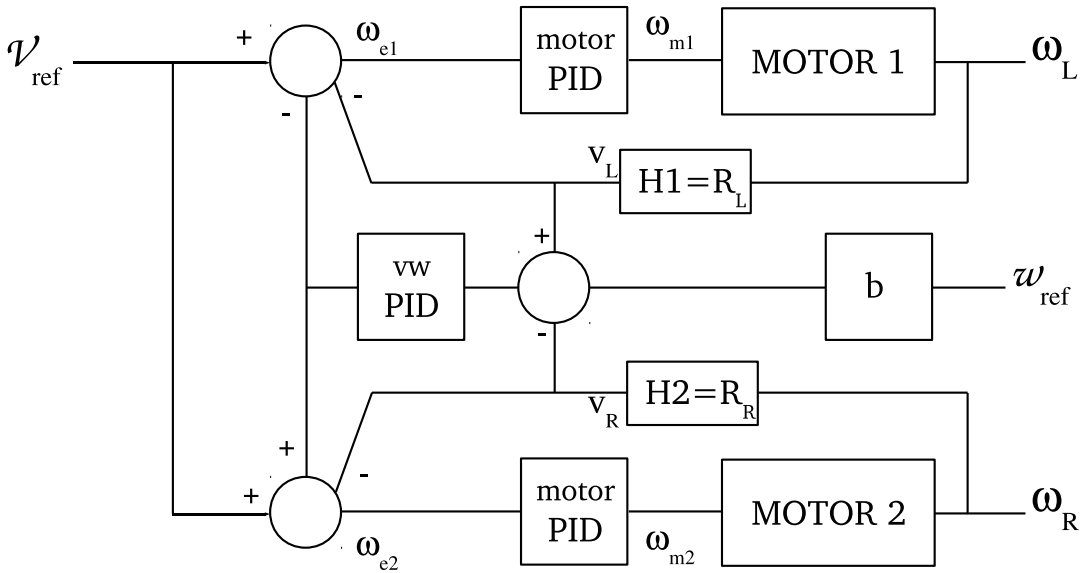
Módulos especiales (cont.)

Módulo PID

Realiza el cálculo de un controlador Proporcional-Integral-Derivativo

Algunas funciones son: `pid_init()`, `pid_update()`

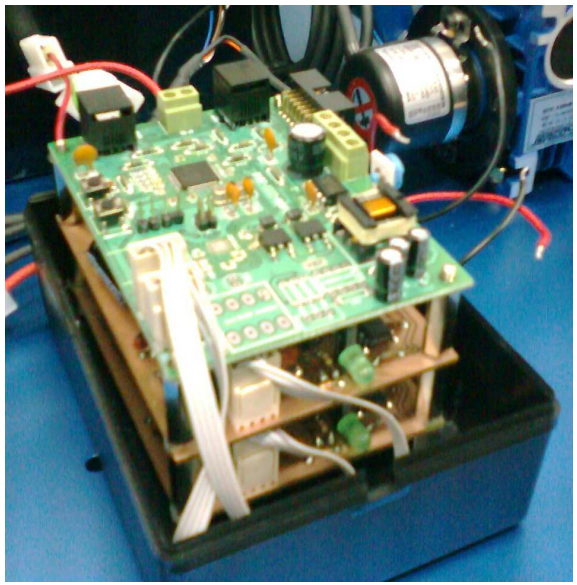
$$
\begin{aligned}
R_n &= R_{n-1} + K_P(e_n - e_{n-1}) \\
&+ K_I\left(\frac{e_n + e_{n-1}}{2}\right) \\
&+ K_D(e_n - 2e_{n-1} + e_{n-2})
\end{aligned}
$$

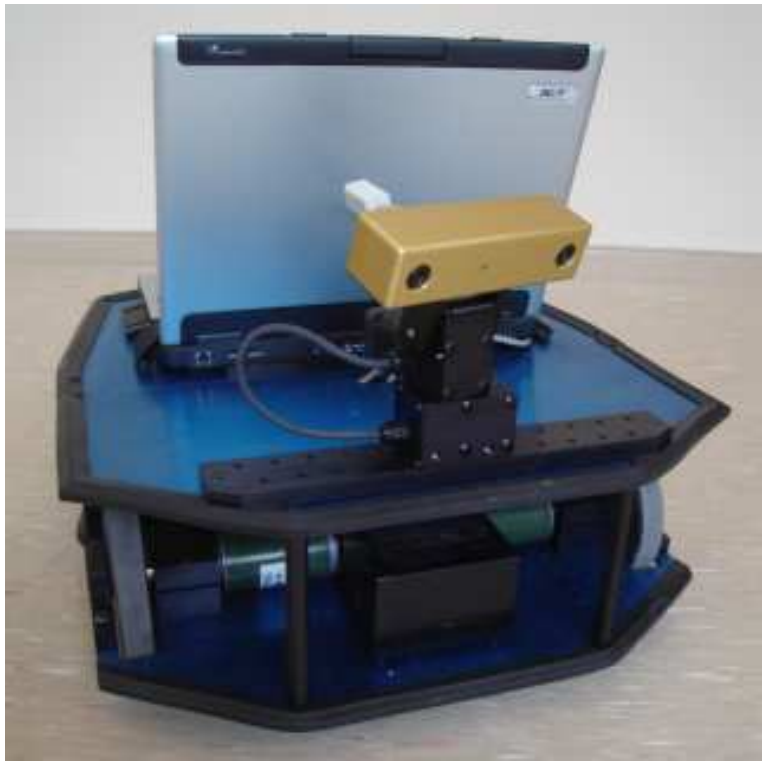


Ejemplo de aplicación - control embebido de robot móvil

Robot Móvil de Arquitectura Abierta

- robot móvil de tracción diferencial
- drivers de potencia en llave H
- encoders ópticos incrementales
- lazo cerrado de velocidad
- cálculos en punto flotante





Módulos utilizados

- GPIO, TIMER, IRQ, UART
- ENCODERS, COMMUNICATION, PID

Conclusiones

- Actualmente se tiene un conjunto de librerías para el desarrollo de sistemas embebidos con ARM7
- Estas permiten mayor agilidad y flexibilidad en la implementación de aplicaciones
- Lo cual ha sido demostrados en casos específicos como el Robot Móvil de Arquitectura Abierta (RoMAA) y un robot Cuadricóptero
- Además, se dispone de un marco de desarrollo de constante evolución y crecimiento

Gracias!

Preguntas?