Workshop
"Introducción a la Robótica con Microcontroladores"

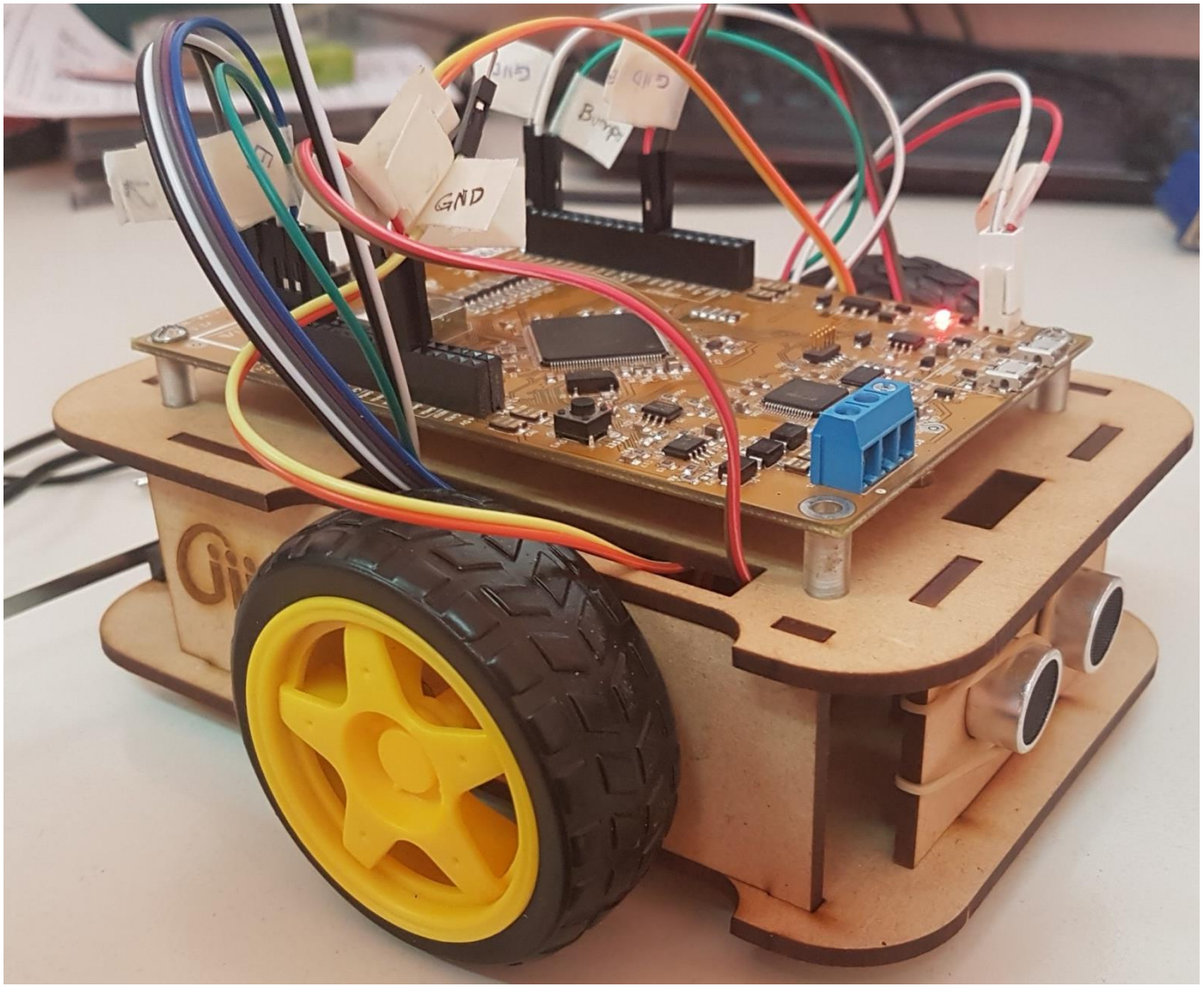
Módulo 1:
Robots Móviles

Ing. Martín Baudino: martinbaudino@gmail.com
Ing. Pablo Garrone: pdgarrone@gmail.com

Estructura del Workshop

- Módulo 1: Robots Móviles
  - Características del robot EduRoMAA
  - Motores de Corriente Continua (DC)
  - Control de motores de CC con Puente H
  - Modulación por Ancho de Pulsos (PWM)
  - Componentes de la placa Edu-CIAA-NXP
  - Entornos de Programación Integrados (IDEs)
  - Entrada y Salida de Propósito General (GPIO)
  - Abstracción del hardware: LPCOpen y sAPI
  - Ejercicio final: programar trayectoria de desplazamiento arbitraria para el robot

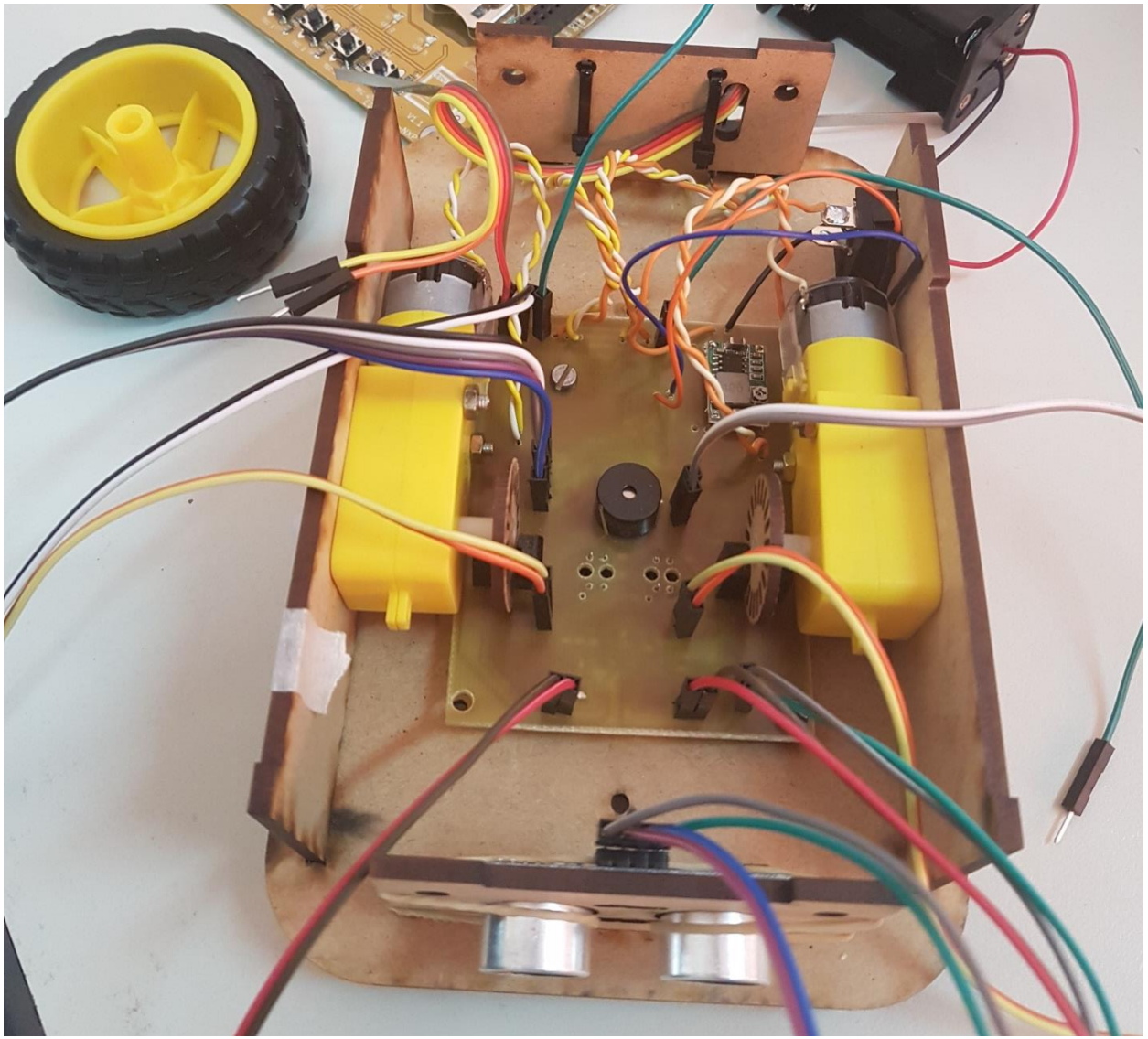
Robot EduRoMAA-CIAA



Robot EduRoMAA-CIAA

- Alimentación con baterías o tensión continua
- Sensores Digitales
  - 2x Bumpers: detección de colisiones
  - 2x Encoders ópticos: medición de velocidad
  - 1x Sonar: detección de obstáculos
- Sensores Analógicos
  - 2x Infrarrojos: detección de líneas
- Actuadores
  - 2x Motores de CC: propulsión
  - 1x Buzzer: sonidos e indicaciones

Robot EduRoMAA-CIAA



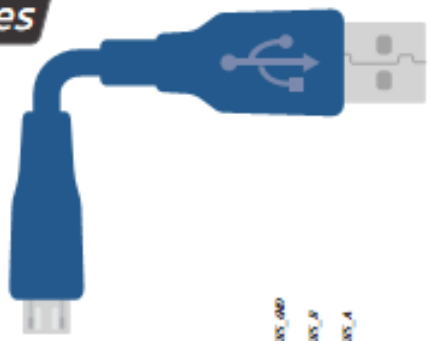
Edu-CIAA-NXP

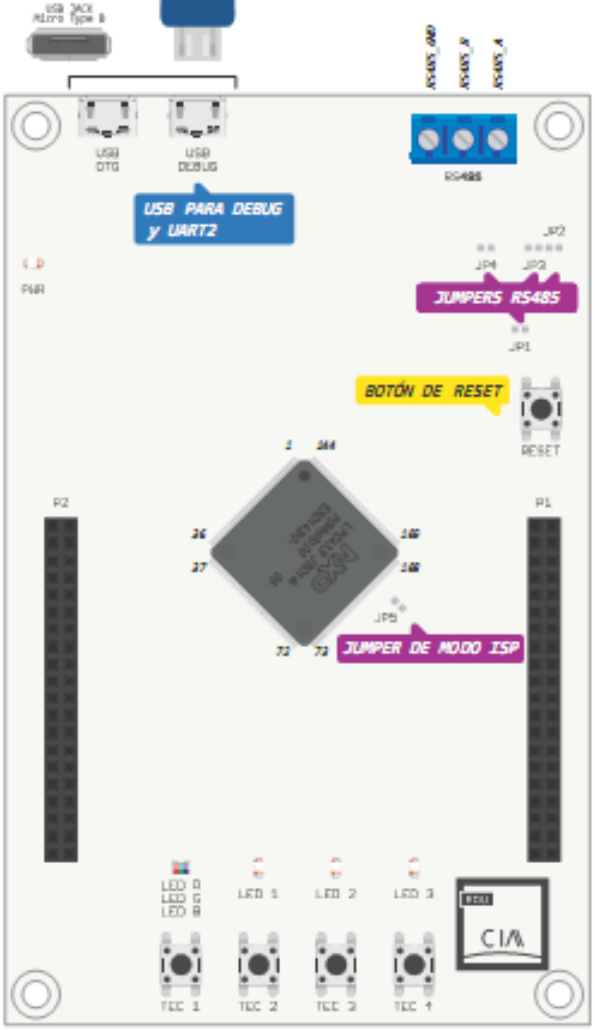
EDU-CIAA-NXP Asignación de pines

Ing.Eric Pernia

24 ABR 2016
ver 4 rev 2

- Alimentación
- Tierra Digital
- Pin E/S GPIO
- Tierra Analógica
- Pin Analógico
- Control e ISP
- Ethernet
- Pin Serie
- Pin LCD
- Pin Teclado





P2

3V3	1	2	5V
GND	3	4	RXD1
GND	5	6	TX_EN
GND	7	8	MDC
RXD0	9	10	CRS_DV
GND	11	12	MDIO
GND	13	14	TXD0
REF_CLK	15	16	TXD1
GND	17	18	SPI_MISO
GND	19	20	SPI_SCK
SPI_MOSI	21	22	LCD4
LCD_EN	23	24	LCD_RS
GND	25	26	LCD3
GND	27	28	LCD2
GPIO8	29	30	LCD1
GPIO2	31	32	GPIO1
GPIO4	33	34	GPIO3
GPIO6	35	36	GPIO5
GND	37	38	GPIO7
GND	39	40	GPIO8

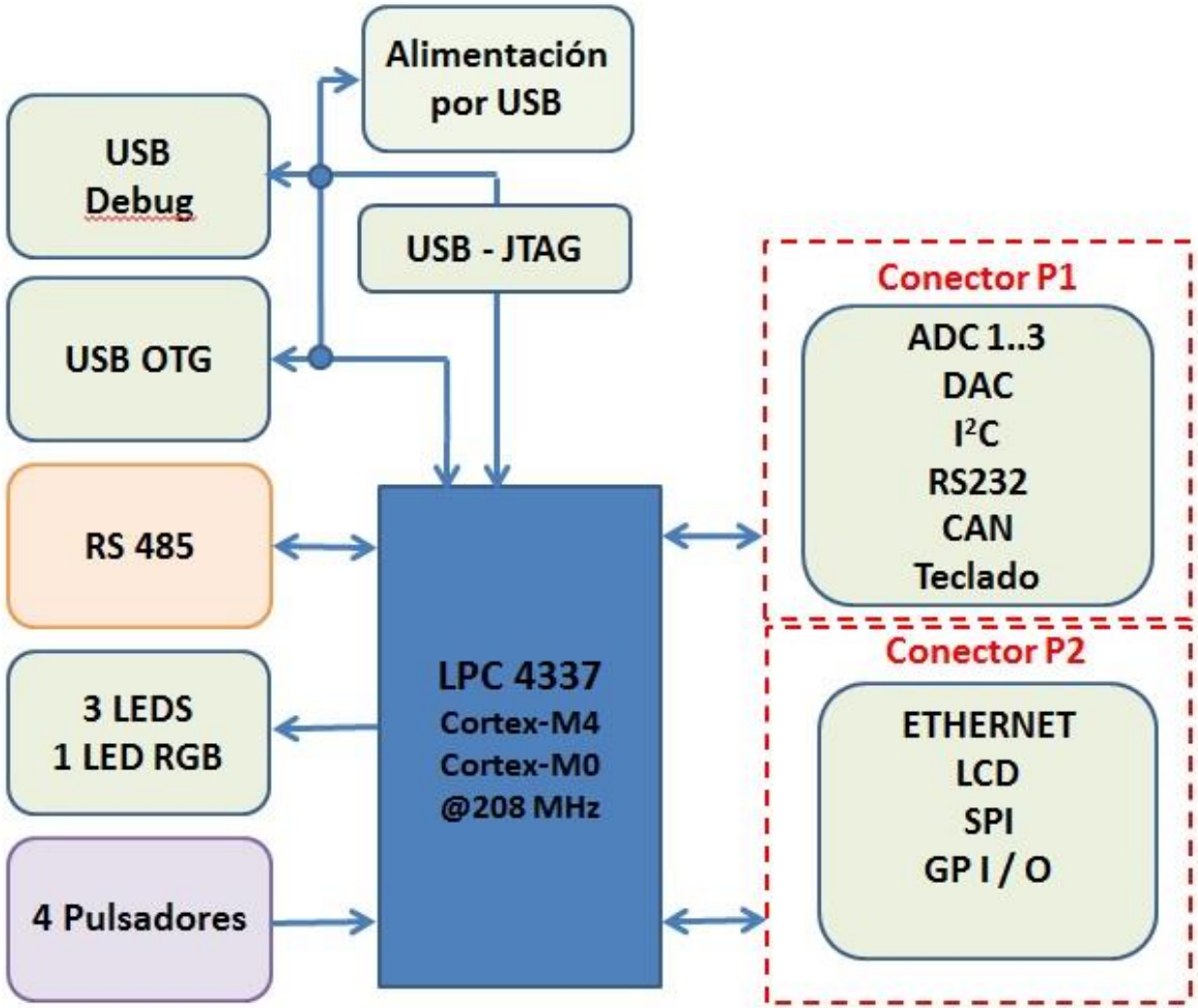
Tira de 40 pines hembra de 0.1"(2,54 mm) de espaciado

P1

3V3	1	2	5V
RESET	3	4	GND
ISP	5	6	WAKEUP
GNDA	7	8	GNDA
CH3	9	10	GNDA
CH2	11	12	GNDA
CH1	13	14	GNDA
DAC	15	16	GNDA
VDDA	17	18	GNDA
I2C_SDA	19	20	GND
I2C_SCL	21	22	GND
232_RX	23	24	GND
232_TX	25	26	GND
CAN_RD	27	28	GND
CAN_TD	29	30	GND
T_COL1	31	32	GND
T_FIL0	33	34	T_COL2
T_FIL3	35	36	T_FIL1
T_FIL2	37	38	GND
T_COL0	39	40	GND

Tira de 40 pines hembra de 0.1"(2,54 mm) de espaciado

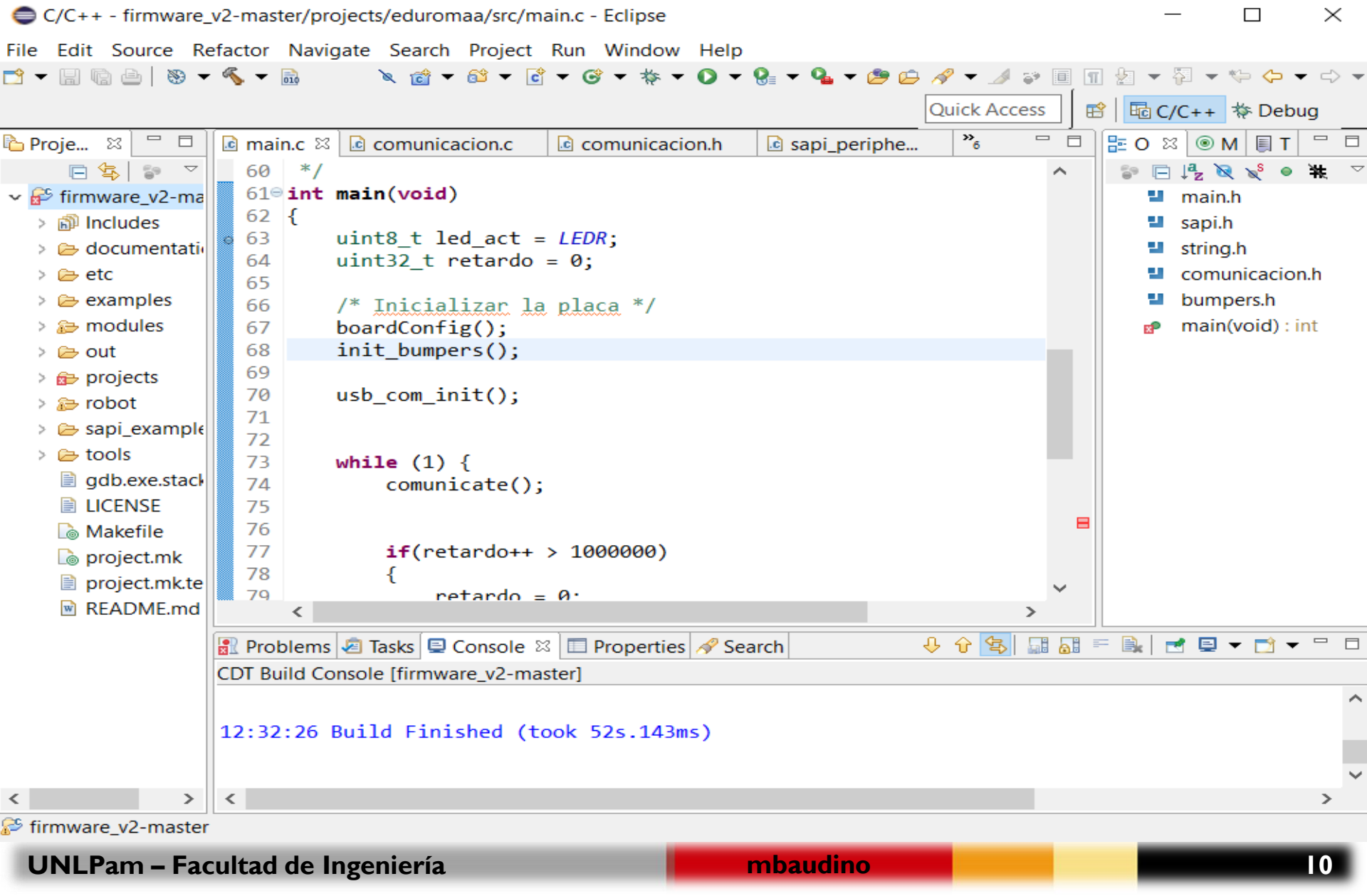
Hardware de la EduCIAA



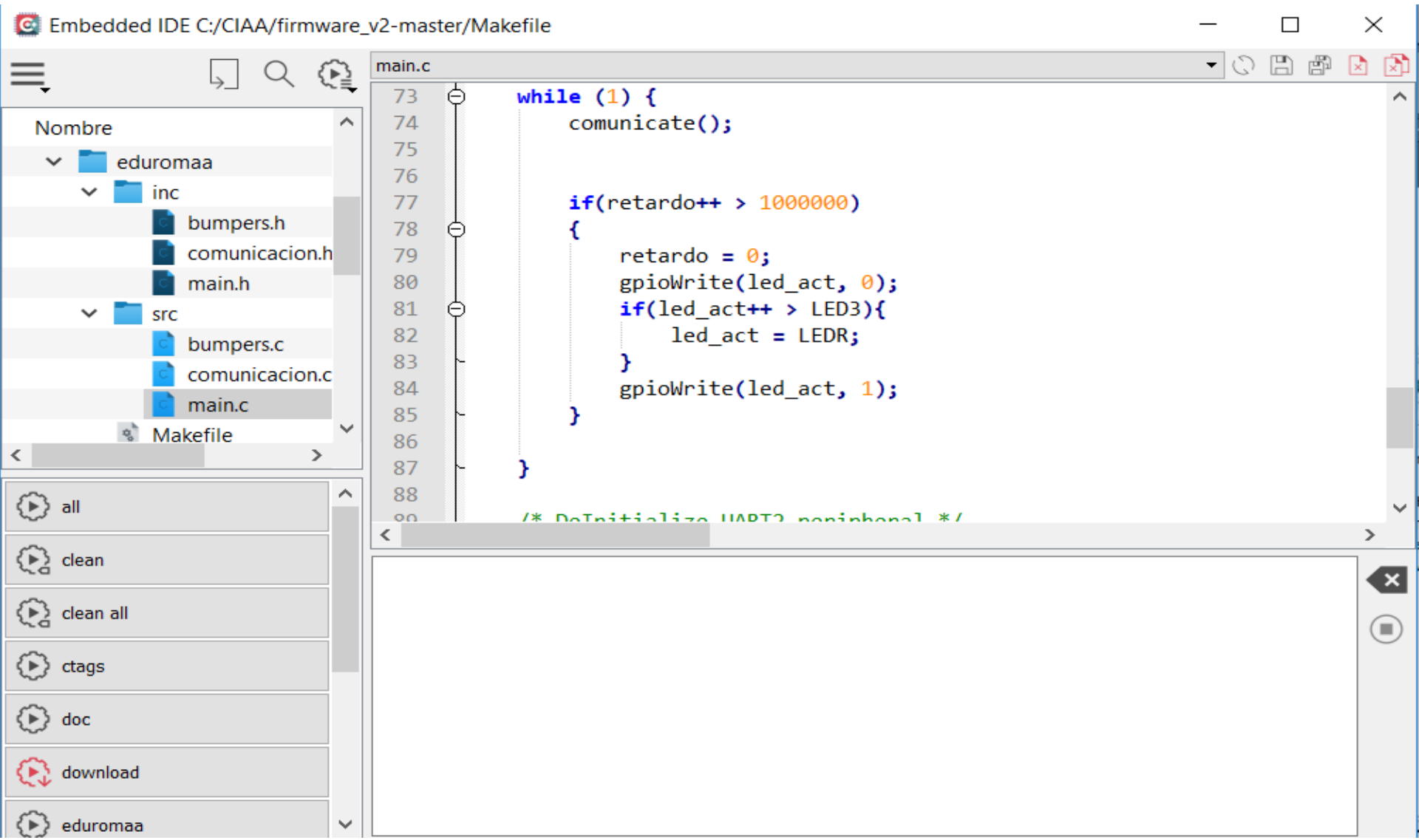
Firmware de la CIAA

- Programación sin RTOS
  - Lazo principal (big loop)
  - Separación de tareas manual
- Abstracción de acceso al Hardware
  - Librerías LPCOpen
  - Librerías sAPI
- Firmware_v2: https://github.com/ciaa/firmware_v2
  - Construcción con GNU Make
  - Proyecto activo definido en **project.mk**
  - Vamos a utilizar mucho código sin entrar en detalle de cómo funciona (!?)

Entorno de Desarrollo



Entorno de Desarrollo



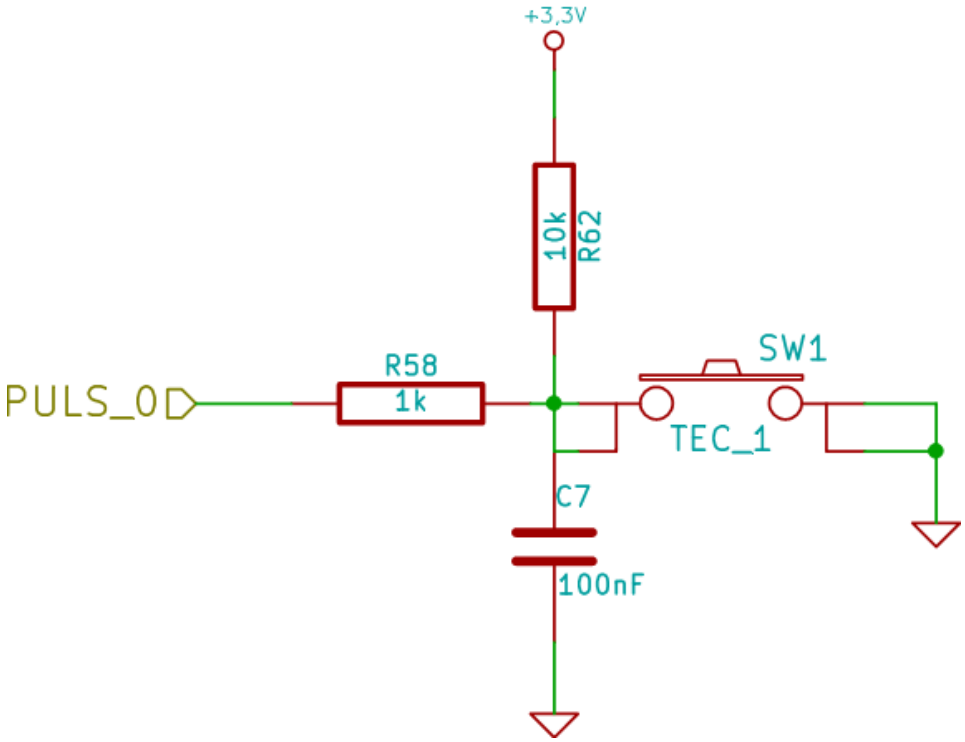
Estructura básica (big loop)

```
#include …                  //Inclusiones (ej. librerías)

extern uint8_t variable; // Declaración de variables compartidas
                         // y definidas en otro archivo
int main(void)
{
       /* Comentario de múltiples líneas
        * Aquí van primero las definiciones de variables locales
        * y luego el acceso a registros (y llamadas a funciones) de
        * configuración e inicialización (ej. de GPIO)
        */
       while(1) {     // Comentario de una línea (solo C-99)

              // Bucle Infinito.
              // Aquí va el código de la aplicación

       }
}
```

Entradas y Salidas Digitales

- Para cada pin se pueden elegir distintas funciones (GPIO, PWM, ADC, etc.)
- GPIO: General Purpose Input Output:
  - Pin funciona como entrada y/o salida digital
  - 0,0V → 0 Lógico
  - 3,3V → 1 Lógico
  - Pines tolerantes a 5V
  - Generación de eventos ante cambios en la entrada
  - Acceso de lectura y escritura a pines individuales

- Ej: TEC1 de la EduCIAA
  - Pull-up
  - Filtro anti-rebotes

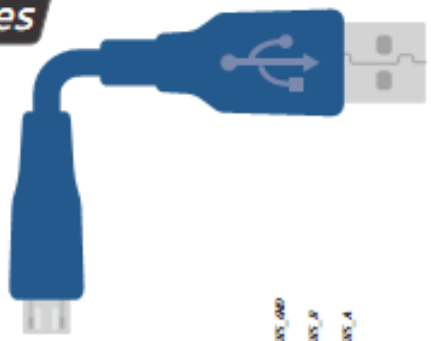


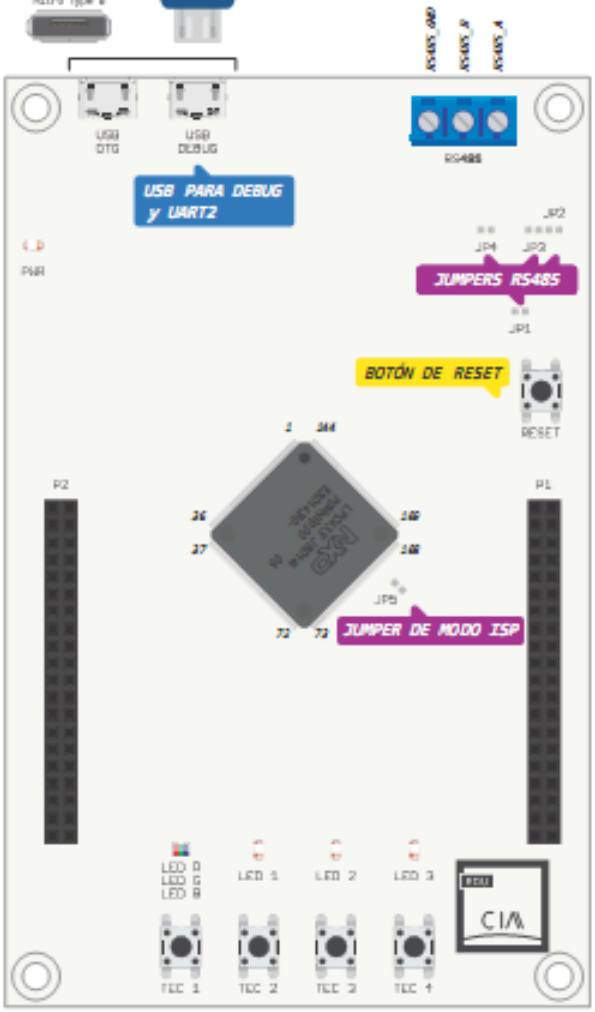
EDU-CIAA-NXP Asignación de pines

Ing.Eric Pernia

24 ABR 2016
ver 4 rev 2

- Alimentación
- Tierra Digital
- Pin E/S GPIO
- Tierra Analógica
- Pin Analógico
- Control e ISP
- Ethernet
- Pin Serie
- Pin LCD
- Pin Teclado





P2

3V3	1	2	5V
GND	3	4	RXD1
GND	5	6	TX_EN
GND	7	8	MDC
RXD0	9	10	CRS_DV
GND	11	12	MDIO
GND	13	14	TXD0
REF_CLK	15	16	TXD1
GND	17	18	SPI_MISO
GND	19	20	SPI_SCK
SPI_MOSI	21	22	LCD4
LCD_EN	23	24	LCD_RS
GND	25	26	LCD3
GND	27	28	LCD2
GPIO8	29	30	LCD1
GPIO2	31	32	GPIO1
GPIO4	33	34	GPIO3
GPIO6	35	36	GPIO5
GND	37	38	GPIO7
GND	39	40	GPIO8

Tira de 40 pines hembra de 0.1"(2,54 mm) de espaciado

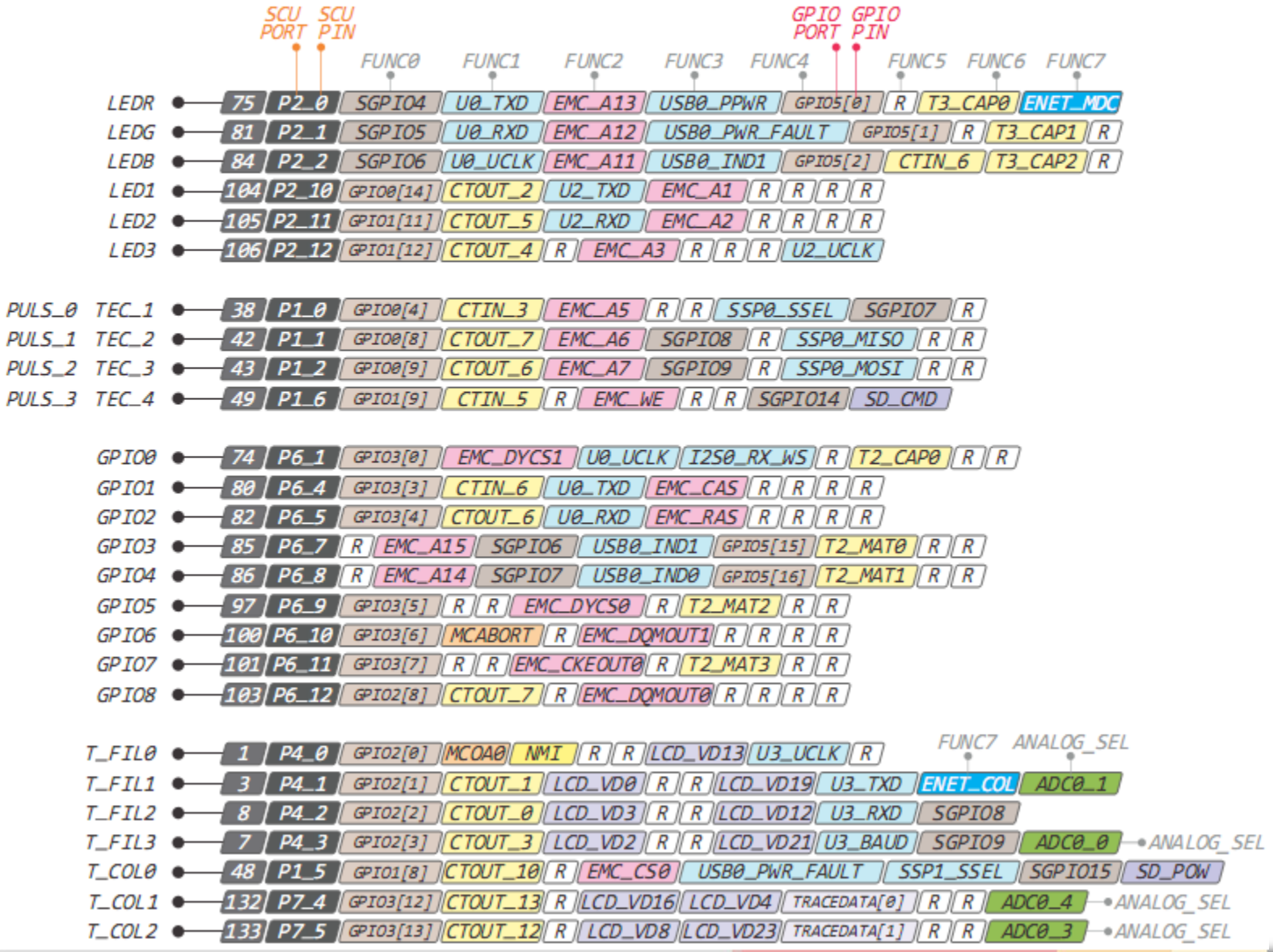
P1

3V3	1	2	5V
RESET	3	4	GND
ISP	5	6	WAKEUP
GNDA	7	8	GNDA
CH3	9	10	GNDA
CH2	11	12	GNDA
CH1	13	14	GNDA
DAC	15	16	GNDA
VDDA	17	18	GNDA
I2C_SDA	19	20	GND
I2C_SCL	21	22	GND
232_RX	23	24	GND
232_TX	25	26	GND
CAN_RD	27	28	GND
CAN_TD	29	30	GND
T_COL1	31	32	GND
T_FIL0	33	34	T_COL2
T_FIL3	35	36	T_FIL1
T_FIL2	37	38	GND
T_COL0	39	40	GND

Tira de 40 pines hembra de 0.1"(2,54 mm) de espaciado

EDU-CIAA-NXP Pines utilizados del NXP

Ing.Eric Pernia



Ejercicio 0: GPIO

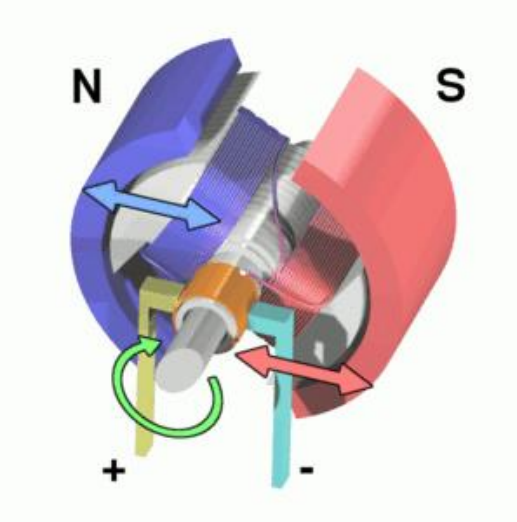
- Leer el estado de los interruptores
  - TEC1 - `if( !gpioRead( TEC1 ) ){`
  - TEC2 - `completar`
- Cuando se encuentren presionados encender el LED correspondiente
  - LED1 - `gpioWrite( LED1, ON );`
  - LED2 - `completar`
- Los LEDs deben quedar encendidos por 5 seg. utilizando el retardo bloqueante de sAPI:
  - `delay(2000);`
  - **¿Qué efecto tiene el bloqueo que produce el retardo?**

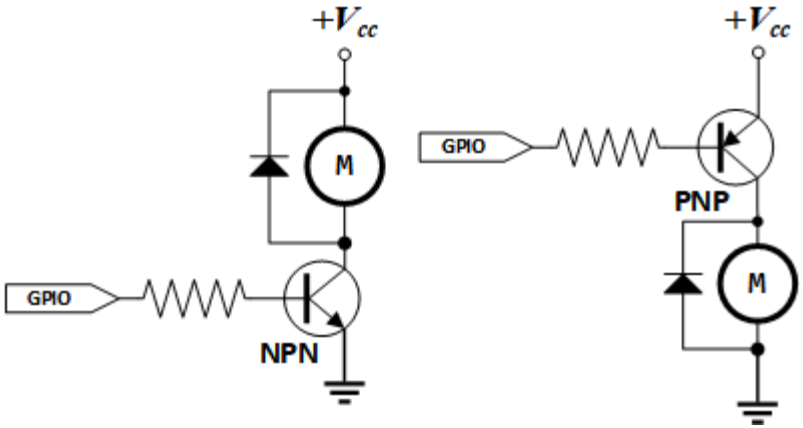
Ejercicio 1: GPIO

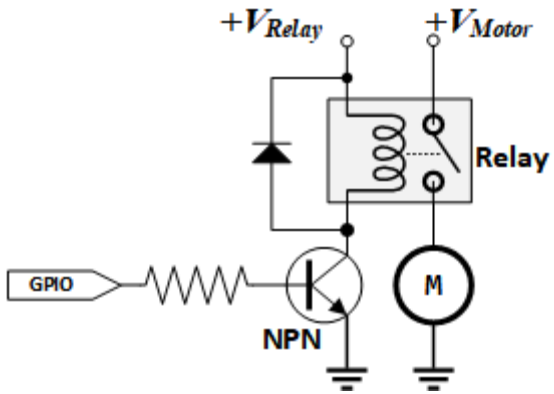
- Leer el estado de los interruptores
  - TEC1 - `if( !gpioRead( TEC1 ) ){`
  - TEC2 - `completar`
- Cuando se encuentren presionados encender el LED correspondiente
  - LED1 - `gpioWrite( LED1, ON );`
  - LED2 - `completar`
- Los LEDs deben quedar encendidos por 2 seg. utilizando el retardo **no** bloqueante de sAPI:
  - `delayConfig( &retardoTEC1, 1000 );`
  - `if ( delayRead( &retardoTEC1 ) ){`
  - **¿Cómo cambió el comportamiento respecto de delay(2000)?**

Motores de Corriente Continua

- La velocidad del motor depende del nivel tensión aplicado y del torque que requiera a carga
- Controladores con un único elemento de conmutación
  - Giro en una sola dirección

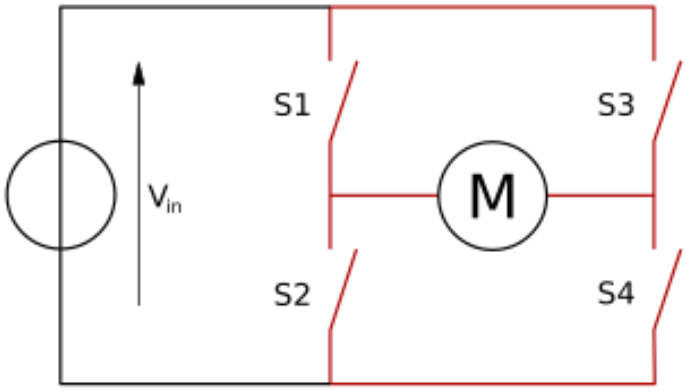






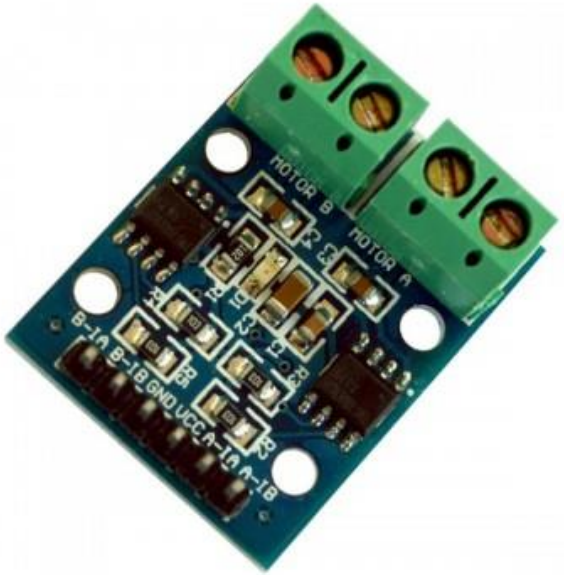
Dirección de Giro - Puente H

- Permite controlar el sentido de giro del motor y el modo de frenado



- Tabla de operación

S1	S2	S3	S4	Resultado
1	0	0	1	El motor gira en avance
0	1	1	0	El motor gira en retroceso
0	0	0	0	El motor se detiene bajo su inercia
1	0	1	0	El motor frena (fast-stop)
0	1	0	1	El motor frena (fast-stop)
1	1	0	0	Corto circuito
0	0	1	1	Corto circuito
1	1	1	1	Corto circuito

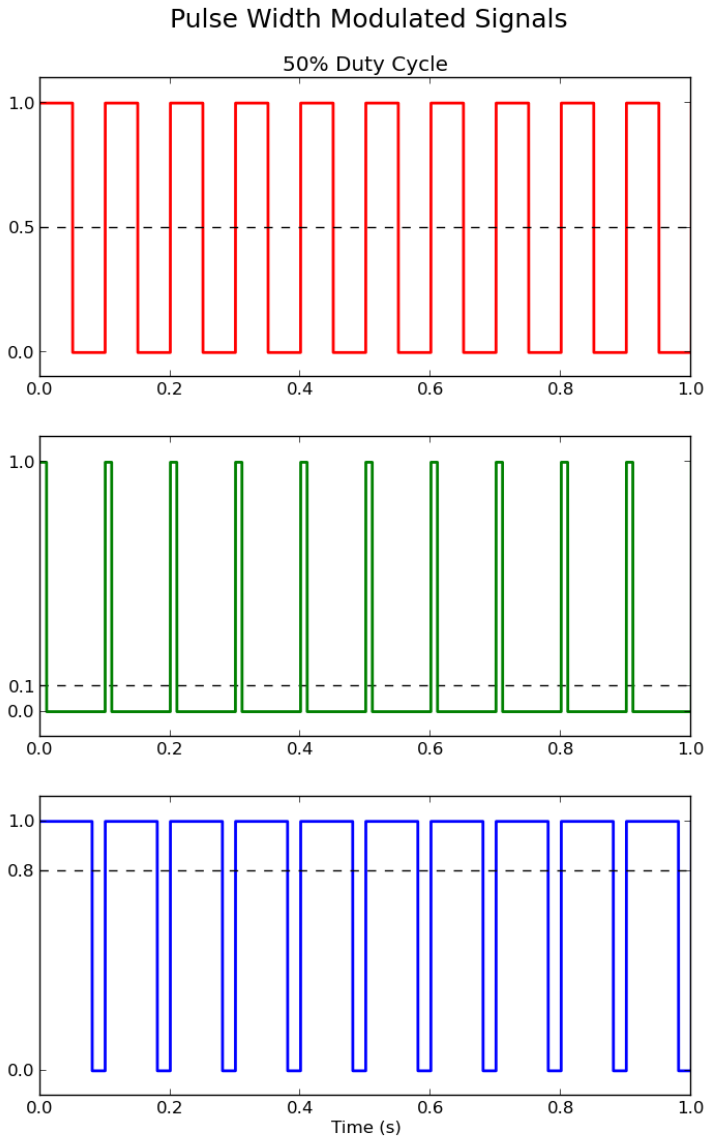


Velocidad de Giro - PWM

- Se mantiene la tensión constante
- Se alimenta al motor con una onda cuadrada de frecuencia fija a la que se le varía el ciclo de trabajo
- El valor promedio de tensión aplicada es proporcional al ciclo de trabajo

$$DC = \frac{T_{ON}}{T_{ON} + T_{OFF}}$$

$$velocidad\ [rpm] \propto v_{max} * DC$$



Desplazamiento EduRoMAA

- Señales de control para activación de los motores
  - GPIO_MOTOR_INH – Inhibición general. Cuando está en alto los motores se bloquean, sin importar las demás señales
  - GPIO_MOTOR_XXX_DIR – Dirección del motor correspondiente (IZQ o DER). Si está en alto el motor avanza y si está en bajo retrocede
  - PWM_MOTOR_XXX - PWM controla velocidad del motor de CC variando el ciclo de trabajo
- Ejemplo: Avance del chasis
  - gpioWrite(GPIO_MOTOR_INH, MOTOR_HABILITA);
  - gpioWrite(GPIO_MOTOR_IZQ_DIR, MOTOR_AVANZA);
  - gpioWrite(GPIO_MOTOR_DER_DIR, MOTOR_AVANZA);
  - pwmWrite(PWM_MOTOR_IZQ, 255);
  - pwmWrite(PWM_MOTOR_DER, 255);

Ejercicio 2: Motores

- Codificar dentro de main() los siguientes comportamientos para el robot:
  - Al presionar TEC1 → Avance a máxima velocidad durante 2 s
  - Al presionar TEC2 → Retroceso a máxima velocidad por 2 s
  - Al presionar TEC3 → Giro en sentido antihorario a 50% de velocidad máxima durante 4 s
  - Al presionar TEC4 → Giro en sentido horario a velocidad máxima durante 2 s
  - Al presionar alguno de los bumpers el robot avance con la rueda correspondiente al 70% y la otra al 40% durante 1 s.
    - ¿Qué es lo que sucede?

Abstracción del hardware

- CMSIS: Cortex Microcontroller Software Interface Standard
  - Interfaces de software para acceso y configuración de núcleos ARM Cortex-M y sus periféricos básicos.
- LPCOpen: Librerías de NXP para la familia de ICs LPC
  - Drivers específicos para cada chip de la familia
  - Configuración de bajo nivel para los dispositivos y periféricos
  - Acceso a los registros del núcleo y de los periféricos
- sAPI: Interfaz de Programación de Aplicación Simple para uC
  - Capa de Abstracción de Hardware (HAL)
  - Paquete de Soporte de la Placa EduCIAA (BSP)
  - Permite concentrarnos en el desarrollo de la aplicación, simplificando la configuración del uC
- Módulo eduromaa:
  - Funciones de alto nivel para resolver problemas de robótica

Ejercicio 3: Integración

- Modificar el ejercicio 2 para que el robot describa una trayectoria cuadrada, desplazándose marcha atrás al 80% de la velocidad máxima:
  - Todavía no tenemos parámetros importantes para la planificación de trayectorias ¿Qué método sencillo puede implementarse?
- Detección de colisiones con los bumpers
  - Detener el robot y emitir un sonido de error utilizando el buzzer hasta que se remueva el obstáculo.
  - ¿Resulta sencillo retomar la trayectoria con la alternativa implementada? ¿Por qué?
- Cuando no se presentan obstáculos en la trayectoria:
  - ¿El robot termina en el mismo lugar donde empezó?
  - ¿Qué inconvenientes no tuvimos en cuenta?