

# Uso del CPLD para Control de Motores vía SPI

Este documento explica cómo codificar los dos bytes que debe enviar el micro maestro (por ejemplo, un PIC18F4550) para controlar dos motores a través del CPLD.

## 1. Byte **CONTROL** (8 bits)

b7	b6	b5 b4	b3	b2	b1	b0
0	0	en_code	sel <sub>2</sub>	dir <sub>2</sub>	sel <sub>1</sub>	dir <sub>1</sub>

b7–b6 siempre 00 (reservados).

b5–b4 en\_code:

- 00 = ningún motor habilitado.
- 01 = sólo Motor 1.
- 10 = sólo Motor 2.
- 11 = ambos motores.

b3 sel<sub>2</sub>:

- 0 → PWM motor 2 a **R\_PWM\_M2**.
- 1 → PWM motor 2 a **L\_PWM\_M2**.

b2 dir<sub>2</sub>:

- 0 = *forward* (hacia delante).
- 1 = *reverse* (hacia atrás).

b1 sel<sub>1</sub>:

- 0 → PWM motor 1 a **R\_PWM\_M1**.
- 1 → PWM motor 1 a **L\_PWM\_M1**.

b0 dir<sub>1</sub>:

- 0 = *forward*.
- 1 = *reverse*.

## 2. Byte DUTY (8 bits)

b7	b6	b5	b4	b3	b2	b1	b0
duty_M1	duty_M1	duty_M1	duty_M1	duty_M2	duty_M2	duty_M2	duty_M2

b7-b4 duty\_M1: valor 0-15  $\rightarrow$  0

b3-b0 duty\_M2: valor 0-15  $\rightarrow$  0

## 3. Ejemplos de Pares CONTROL/DUTY

Acción	CONTROL	DUTY	Salida esperada
Apagar ambos	0x00	0x00	Ningún enable, sin PWM.
Motor 1 derecha al 100 %	0x10	0xF0	R_EN_M1=1, R_PWM_M1=100 %.
Motor 1 izquierda al 50 %	0x12	0x80	L_EN_M1=1, L_PWM_M1=50 %.
Motor 2 derecha al 100 %	0x20	0x0F	R_EN_M2=1, R_PWM_M2=100 %.
Motor 2 izquierda al 25 %	0x28	0x04	L_EN_M2=1, L_PWM_M2=25 %.
Ambos motores opuestos 60 %	0x3E	0xAA	Giros opuestos al 60 %.

## 4. Código de Ejemplo en C (MAESTRO PIC)

```

1 #include <xc.h>
2 #include <stdint.h>
3 #include "spi_master.h"
4
5 #define _XTAL_FREQ 48000000UL
6
7 void send_pair(uint8_t ctrl, uint8_t duty) {
8     LATAbits.LATA5 = 0;           // SS = 0 (activo)
9     SPI_Master_TxRx(ctrl);        // envi a CONTROL
10    LATAbits.LATA5 = 1;           // SS = 1
11    __delay_us(50);
12
13    LATAbits.LATA5 = 0;           // SS = 0
14    SPI_Master_TxRx(duty);        // envi a DUTY
15    LATAbits.LATA5 = 1;           // SS = 1
16    __delay_us(200);              // espera para ver efecto
17 }
18
19 void main(void) {

```

```

20     SPI_Master_Init(SPI_CLK_DIV_64, SPI_MODE_0);
21     TRISAbits.TRISA5 = 0;    // SS como salida
22     LATAbits.LATA5 = 1;     // SS inactivo
23
24     // Ejemplo: Motor1 izquierda al 50%
25     send_pair(0x12, 0x80);
26
27     while(1) {
28         // bucle principal...
29     }
30 }

```

## 5. Configuración y Velocidades Requeridas

- **Reloj del CPLD:** 50 MHz (entrada en el pin asignado).
- **SPI Maestro:**  $F_{osc}/64$  ( $\approx 750$  kHz SCLK).
- **Modo SPI:** MODE\_0 (CKP=0, CKE=1).
- **Pull-up en SS:** 10 k para mantener SS=1 cuando el PIC no lo esté conduciendo.
- **CPLD PWM:** genera 20 kHz con divisor interno ( $50 \text{ MHz}/(155+1)/16$ ).

Con estas dos palabras y la configuración indicada, tu CPLD interpretará cada par como un comando completo para encender, dirigir y ajustar la velocidad de ambos motores.