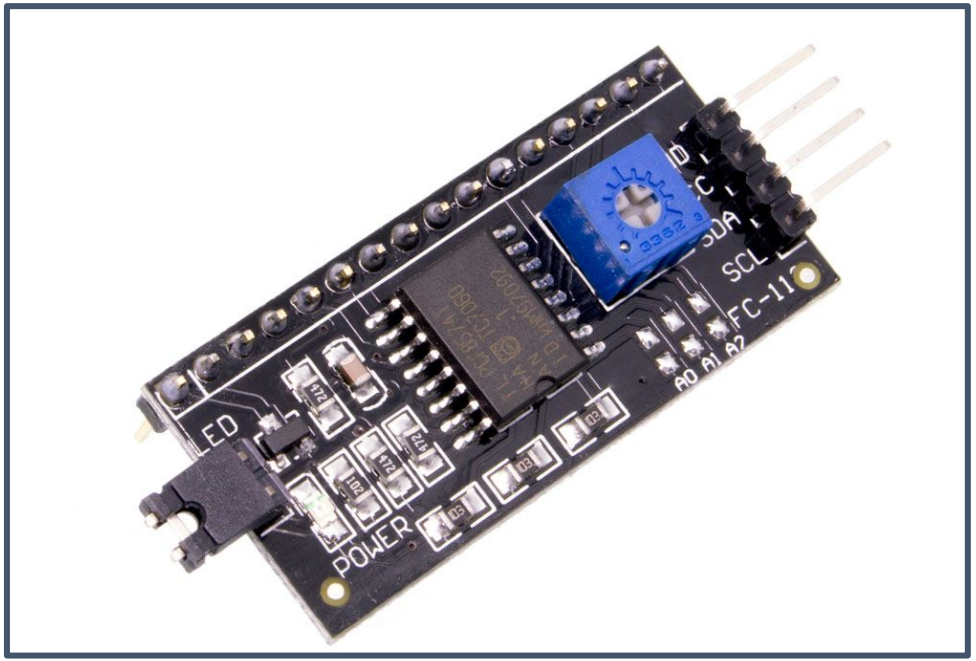
Wels

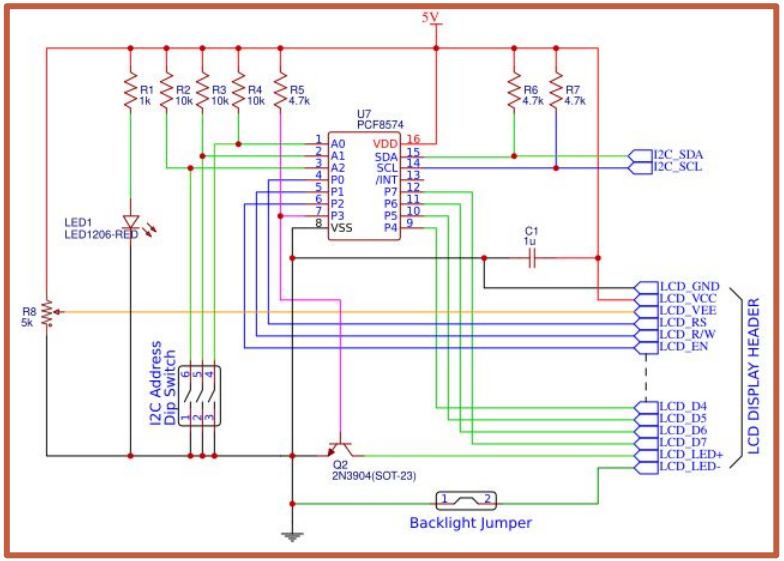
C Embebido

Wels



PCF8574

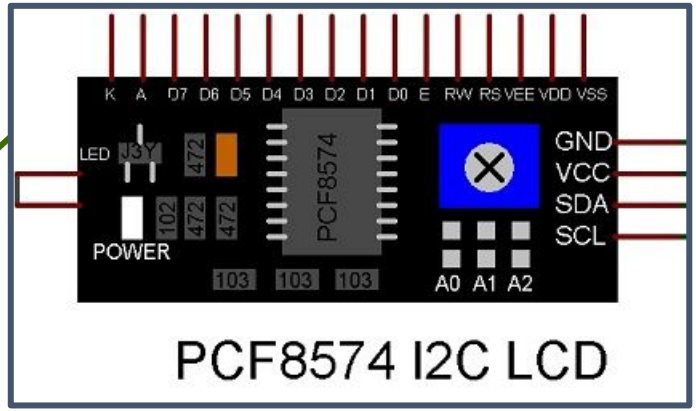
PCF8574

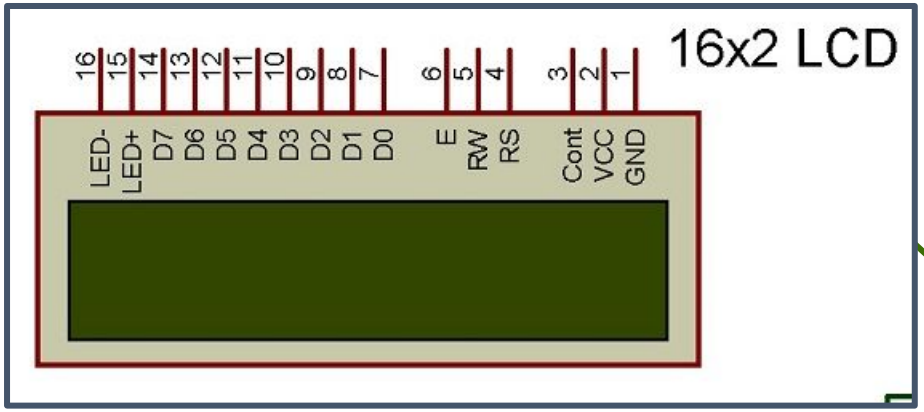


- LCD a 4 Bits.
- 4 Pines MSB del PCF8574 envía los datos.
- 4 Pines LSB del PCF8574 controlan el LCD.
- Dirección del PCF8574: 0x4E
- Utiliza el protocolo I2C

Wels

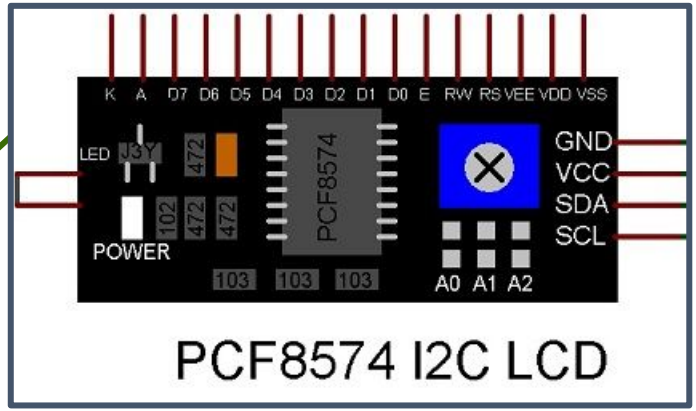
- P0 -> LCD_RS
- P1 -> LCD_RW
- P2 -> LCD_EN
- P3 -> LCD_BK_L

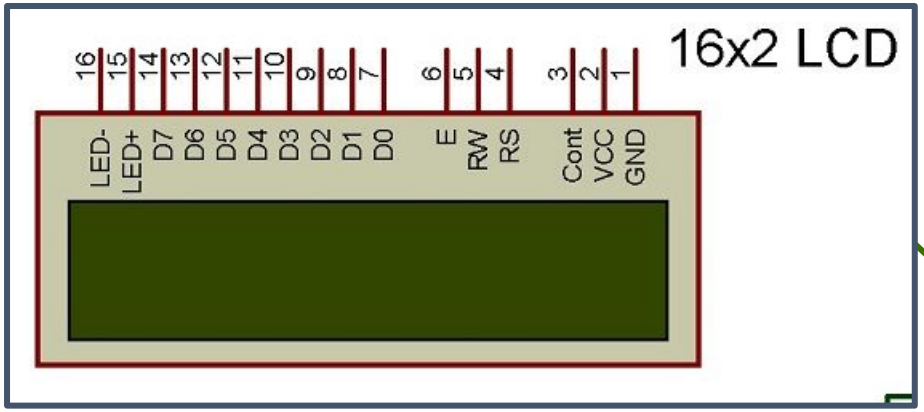




PCF8574

- P4 -> LCD_D4
- P5 -> LCD_D5
- P6 -> LCD_D6
- P7 -> LCD_D7





PCF8574

Wels

```
void LCD_Init(void)
{
    LCD_Gpio();
    LCD_Comando(0x30);
    Delay_ms(5);
    LCD_Comando(0x30);
    Delay_ms(1);
    LCD_Comando(0x32);// 4 bits
    LCD_Comando(0x28);// 2 filas 5x8
    LCD_Comando(0x0C);// Display ON
    LCD_Comando(0x01);// Limpia LCD
    LCD_Comando(0x06);// Incremento
    Delay_ms(100);
}
```

Data
Para
Configurar

LCD Configuración

Wels

#Define

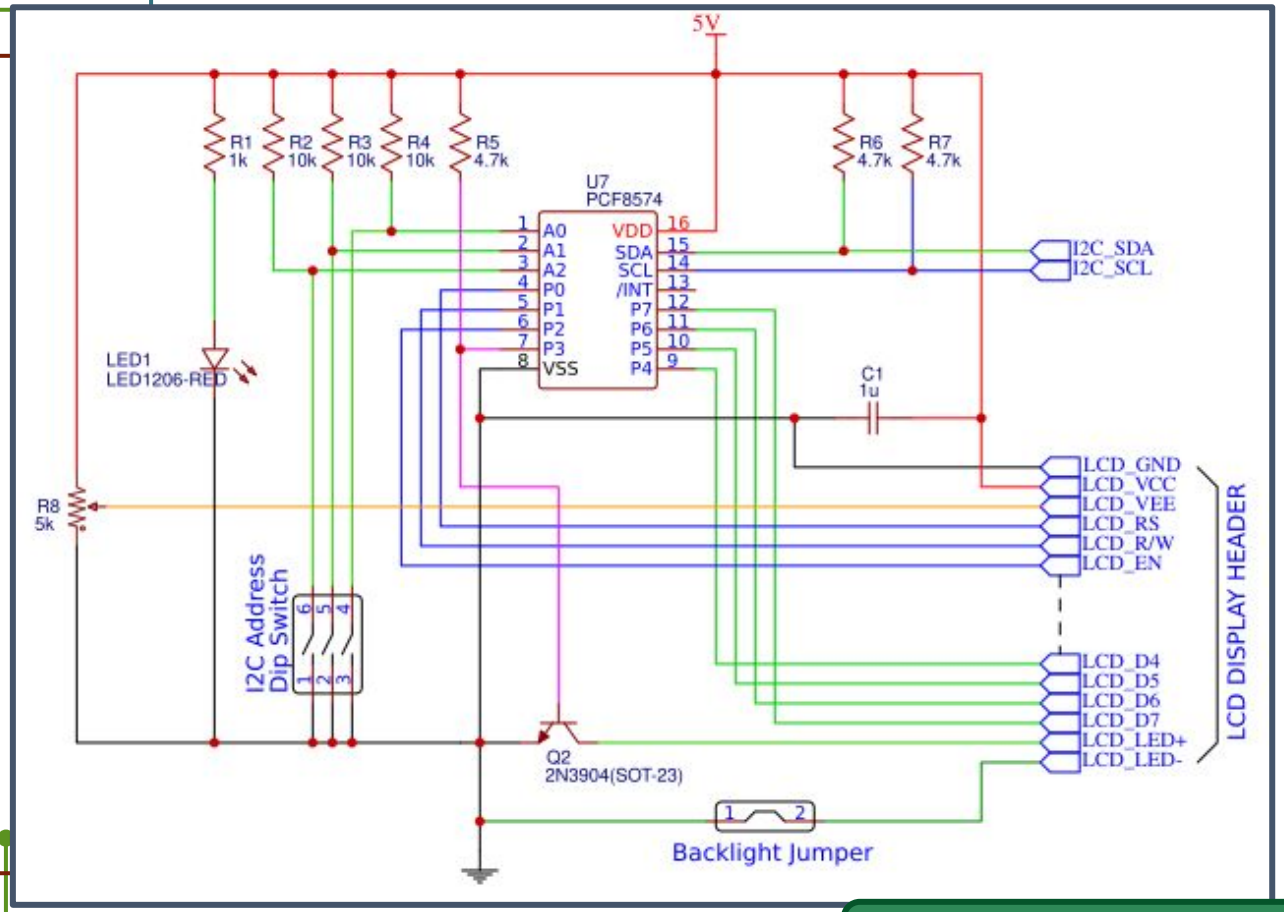
Es una directiva de preprocesador, se procesan antes de la compilación y permite reemplazar un identificador por un valor específico en todo el código.

```
LCD_Comando(0x01);// Limpia LCD
```

```
#define LCD_Borrar 0x01
```

Ahora cada vez que coloquemos LCD_Borrar el programa lo reemplazará por 0x01

Wels

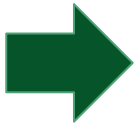


- LCD_RS -> P0
- LCD_RW -> P1
- LCD_EN -> P2
- LCD_BK_L -> P3
- LCD_D4 -> P4
- LCD_D5 -> P5
- LCD_D6 -> P6
- LCD_D7 -> P7

PCF8574

Estructura LCD

```
void LCD_Init(void)
{
    LCD_Gpio();
    LCD_Comando(0x30);
    Delay_ms(5);
    LCD_Comando(0x30);
    Delay_ms(1);
    LCD_Comando(0x32);// 4 bits
    LCD_Comando(0x28);// 2 filas 5x8
    LCD_Comando(0x0C);// Display ON
    LCD_Comando(0x01);// Limpia LCD
    LCD_Comando(0x06);// Incremento
    Delay_ms(100);
}
```



```
typedef struct
{
    int32_t id_carro;
    char color[1];
    char conductor[10];
}uber_t;
```

Wels

Estructura LCD

```
/* ESTRUCTURA PARA CONFIGURAR EL LCD */
typedef struct
{
    uint8_t lcd_address;    /* DIRECCIÓN DE LCD*/
    uint8_t lcd_4bit;       /* CONFIGURAR COMO 4 BITS */
    uint8_t lcd_font;       /* CONFIGURAR LINEAS Y CARACTERES */
    uint8_t lcd_display;    /* CONFIGURAR DISPLAY, CURSOR Y PARPADEO */
    uint8_t lcd_inc_ddram;  /* CONFIGURAR INCREMENTO DE DDRAM */
    uint8_t lcd_clear;      /* CONFIGURAR LIMPIEZA DE LCD */
    uint8_t data[4];        /* DATA A ENVIAR POR EL I2C */
}lcd_t;
```

Estructura para cada dato de configuración

Wels

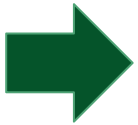
Estructura LCD

```
/* LCD LIBRERIA */
lcd_t LCD_Data;
LCD_Data.lcd_4bit = LCD_4BITS;
LCD_Data.lcd_address = LCD_I2C_ADDRESS_0;
LCD_Data.lcd_clear = LCD_BORRARLCD;
LCD_Data.lcd_display = LCD_DISPLAY_ON;
LCD_Data.lcd_font = LCD_4BIT_2LINEAS;
LCD_Data.lcd_inc_ddram = LCD_ENTRY_INCREMENTO_;
```

Estructura para la configuración

Estructura LCD

```
void LCD_Init(void)
{
    LCD_Gpio();
    LCD_Comando(0x30);
    Delay_ms(5);
    LCD_Comando(0x30);
    Delay_ms(1);
    LCD_Comando(0x32);// 4 bits
    LCD_Comando(0x28);// 2 filas 5x8
    LCD_Comando(0x0C);// Display ON
    LCD_Comando(0x01);// Limpia LCD
    LCD_Comando(0x06);// Incremento
    Delay_ms(100);
}
```



```
void LCD_I2C_Init(lcd_t *LCD_Data)
{
    LCD_I2C_Comando(LCD_Data, LCD_CONFIG);
    delay_ms(100);
    LCD_I2C_Comando(LCD_Data,LCD_CONFIG);
    delay_ms(5);
    LCD_I2C_Comando(LCD_Data, LCD_CONFIG | LCD_Data->lcd_4bit);
    delay_ms(1);
    LCD_I2C_Comando(LCD_Data, LCD_Data->lcd_font);
    LCD_I2C_Comando(LCD_Data, LCD_Data->lcd_display);
    LCD_I2C_Comando(LCD_Data, LCD_Data->lcd_clear);
    LCD_I2C_Comando(LCD_Data, LCD_Data->lcd_inc_ddram);
    delay_ms(100);
}
```

Wels

Comando y Caracter

Creamos **LCD_I2C_Comando** para el envío de comandos al LCD y **LCD_I2C_Caracter** para el envío de data al LCD.

```
void LCD_I2C_Comando(lcd_t *LCD_Data, uint8_t cmd)
{
    I2C_SendCommand(LCD_Data,cmd);
}

void LCD_I2C_Caracter(lcd_t *LCD_Data, uint8_t data)
{
    I2C_SendData(LCD_Data,data);
}
```

Pero dentro de cada función llamo a otras funciones para la transmisión.

I2C_SendCommand

- LCD_RS -> P0
- LCD_RW -> P1
- LCD_EN -> P2
- LCD_BK_L -> P3
- LCD_D4 -> P4
- LCD_D5 -> P5
- LCD_D6 -> P6
- LCD_D7 -> P7

El dato se envía en dos partes (*nible alto y nible bajo*) y se le agrega los pines **LCD_EN** y **LCD_BK_LIGHT**.

```
static void I2C_SendCommand(lcd_t *LCD_Data, uint8_t data)
{
    uint8_t low_data = (0xF0 & (data<<4));
    uint8_t high_data = (0xF0 & data);
    LCD_Data->data[0] = high_data | LCD_EN | LCD_BK_LIGHT;
    LCD_Data->data[1] = high_data | LCD_BK_LIGHT;
    LCD_Data->data[2] = low_data | LCD_EN | LCD_BK_LIGHT;
    LCD_Data->data[3] = low_data | LCD_BK_LIGHT;
}
```

Wels

I2C_SendData

- LCD_RS -> P0
- LCD_RW -> P1
- LCD_EN -> P2
- LCD_BK_L -> P3
- LCD_D4 -> P4
- LCD_D5 -> P5
- LCD_D6 -> P6
- LCD_D7 -> P7

El dato se envía en dos partes (*nible alto y nible bajo*) y se le agrega los pines **LCD_RS**, **LCD_EN** y **LCD_BK_LIGHT**.

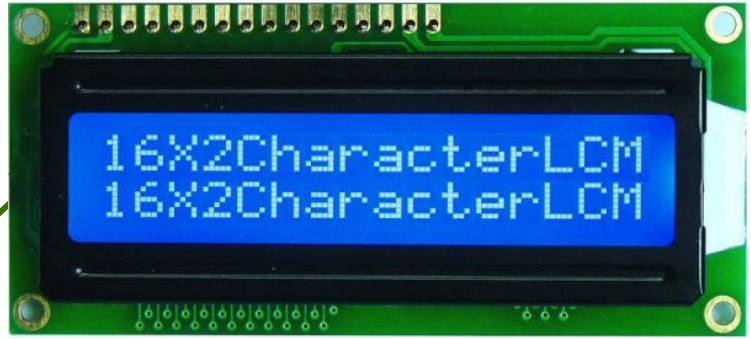
```
static void I2C_SendData(lcd_t *LCD_Data, uint8_t data)
{
    uint8_t low_data = (0xF0 & (data<<4));
    uint8_t high_data = (0xF0 & data);
    LCD_Data->data[0] = high_data | LCD_EN | LCD_BK_LIGHT | LCD_RS;
    LCD_Data->data[1] = high_data | LCD_BK_LIGHT | LCD_RS;
    LCD_Data->data[2] = low_data | LCD_EN | LCD_BK_LIGHT | LCD_RS;
    LCD_Data->data[3] = low_data | LCD_BK_LIGHT | LCD_RS;
}
```

Wels

```
void LCD_I2C_Cadena(lcd_t *LCD_Data, char *cadena)
{
    for(uint8_t i = 0; i < strlen(cadena); i++)
    {
        LCD_I2C_Caracter(LCD_Data, cadena[i]);
    }
}
```

Función para enviar cadena

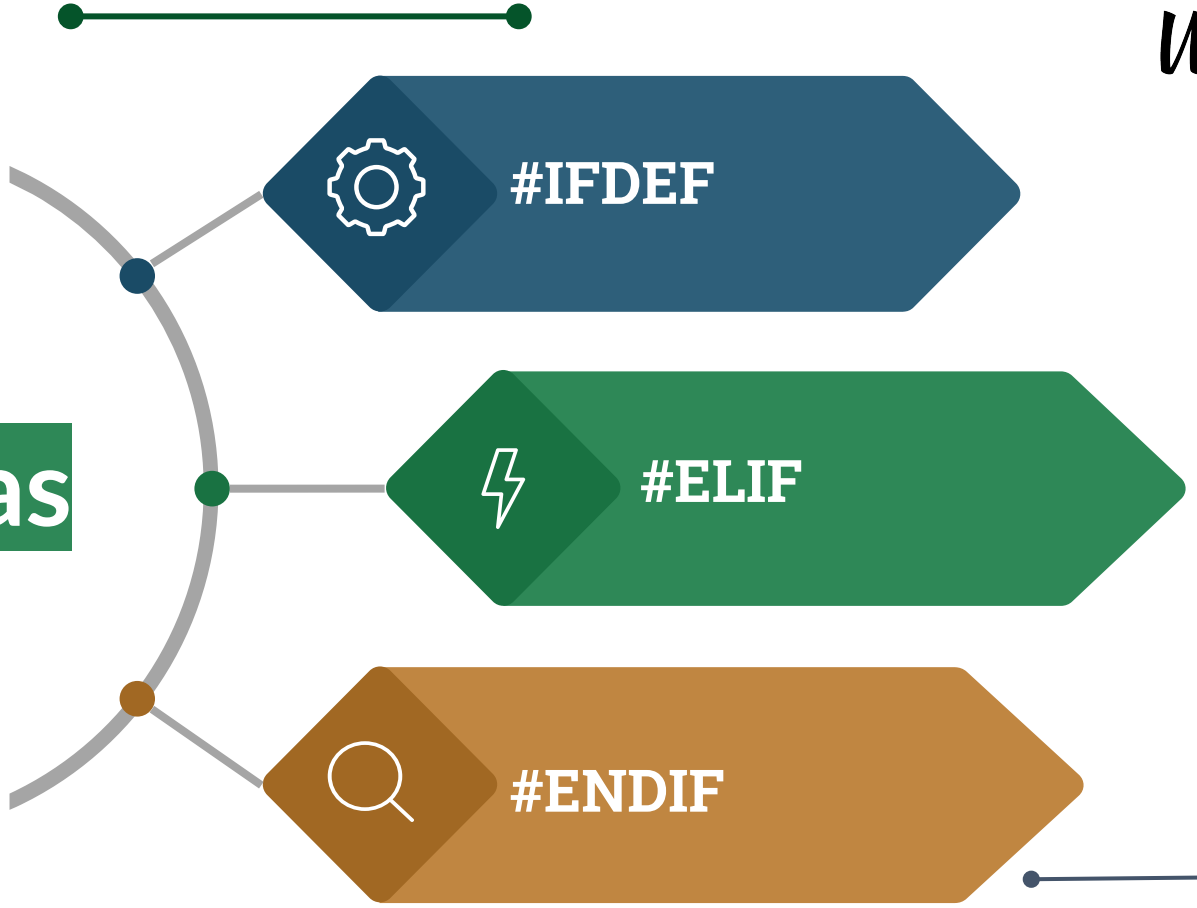
Wels



¿CUAL LCD UTILIZAR?

Wels

Directivas



#IFDEF o #IF DEFINED

Wels

se utiliza para verificar si una definición o macro particular ya ha sido definida en el código

```
#ifdef LCD_2X16
    // Código relacionado con el LCD
#endif
```

Si está definido se ejecutará lo que está adentro

#ELIF

Se utiliza cuando hay múltiples condiciones para comprobar.

```
#ifdef LCD_2X16
    // Código relacionado con el LCD
#elif defined(LCD_4X20)
    // Código relacionado con el LCD
#else
    // Código relacionado con el LCD
#endif
```

Puede ser seguido de un #else y un #endif

#ENDIF

Marca el final de un bloque condicional.

```
#ifdef LCD_2X16
    // Código relacionado con el LCD
#elif defined(LCD_4X20)
    // Código relacionado con el LCD
#else
    // Código relacionado con el LCD
#endif
```

Veamos cómo aplicarlo

Wels

Definir el LCD

```
/* LCD LINEAS*/
#define LCD_LINEA1                  0x80
#define LCD_LINEA2                  0xC0
#define LCD_LINEA3                  0x94
#define LCD_LINEA4                  0xD4
```

```
/* DEFINIR EL LCD A UTILIZAR */
#define LCD_4X20
//#define LCD_2X16
```

Wels

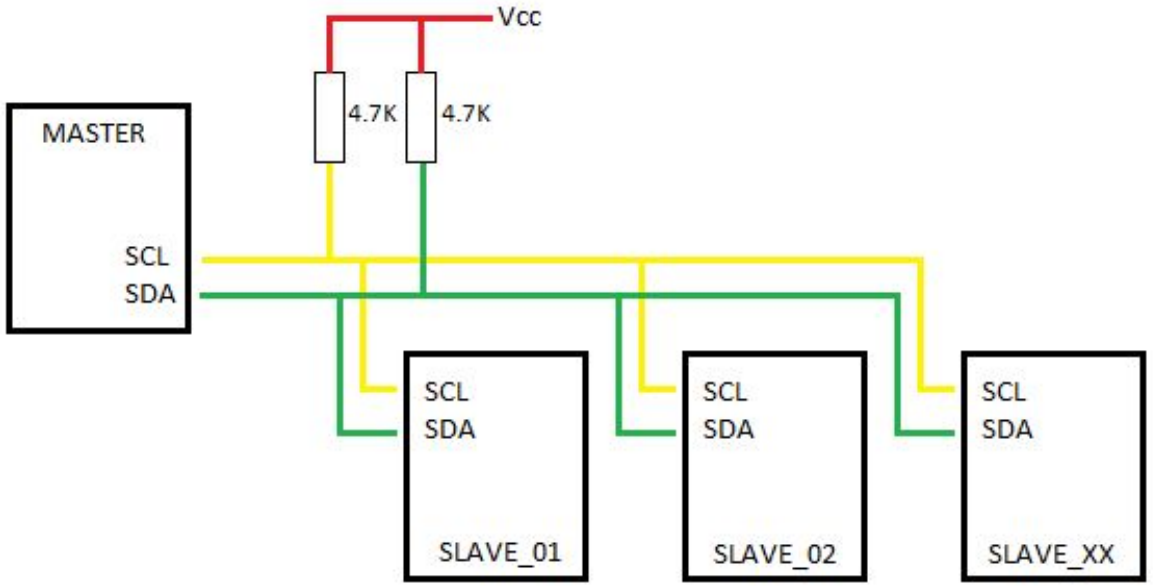
Función de Posición

```
void LCD_I2C_XY(lcd_t *LCD_Data, uint8_t x, uint8_t y)
{
#if defined(LCD_2X16)
    if(x > 0)
    {
        LCD_I2C_Comando(LCD_Data, LCD_LINEA2 + y);
    }
    else
    {
        LCD_I2C_Comando(LCD_Data, LCD_LINEA1 + y);
    }
```

```
#elif defined(LCD_4X20)
    if(x == 0)
    {
        LCD_I2C_Comando(LCD_Data, LCD_LINEA1 + y);
    }
    else if(x == 1)
    {
        LCD_I2C_Comando(LCD_Data, LCD_LINEA2 + y);
    }
    else if(x == 2)
    {
        LCD_I2C_Comando(LCD_Data, LCD_LINEA3 + y);
    }
    else
    {
        LCD_I2C_Comando(LCD_Data, LCD_LINEA4 + y);
    }
#endif
```

Código para una LCD de 2x16 o 4x20

Wels


Vcc
4.7K
4.7K
MASTER
SCL
SDA
SCL
SDA
SLAVE_01
SCL
SDA
SLAVE_02
SCL
SDA
SLAVE_XX

¿Y EL I2C?

Wels

Gracias

@welstheory
hola@welstheory.com
+51 918 899 684