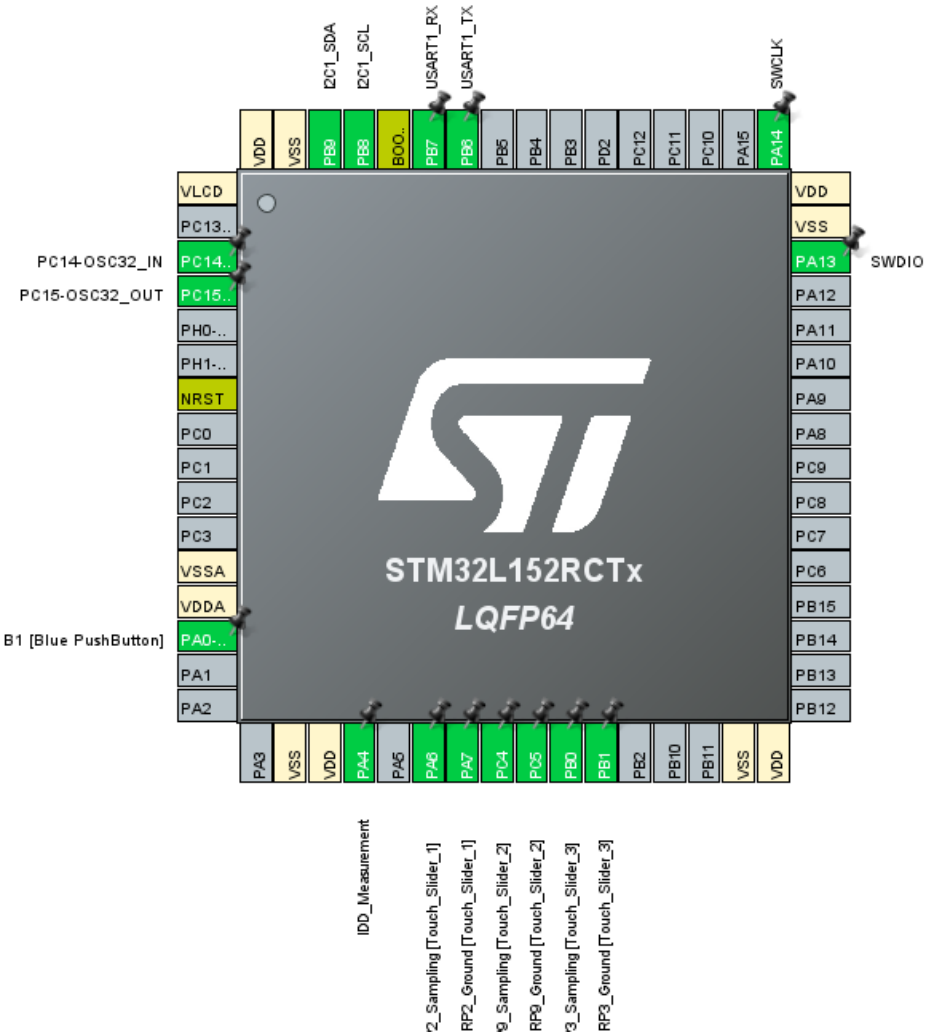
Tema 11: Comunicación Serie Síncrona

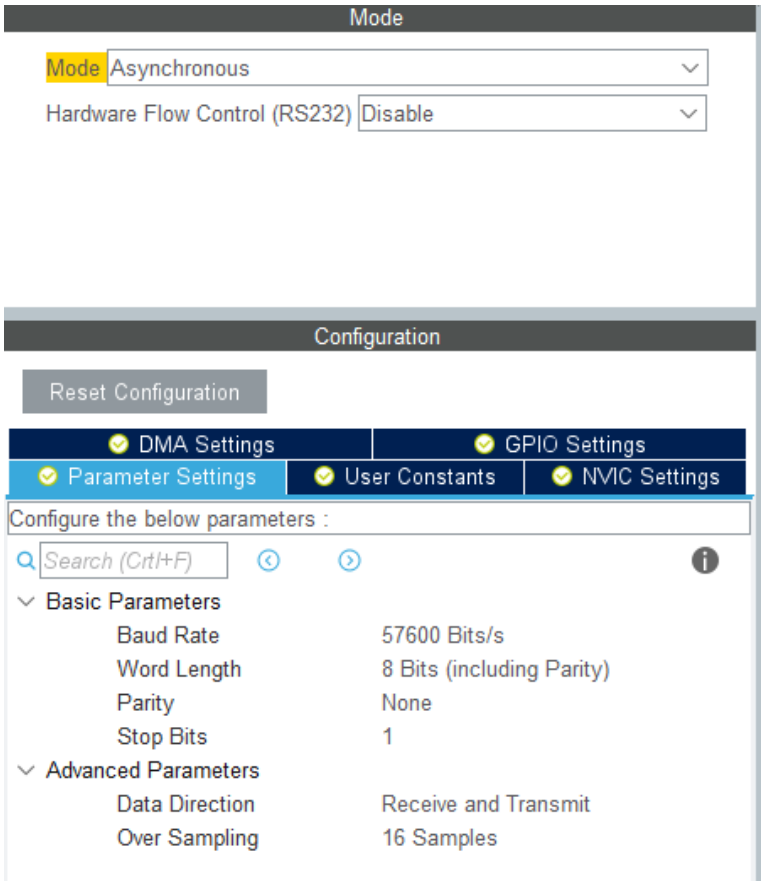
SOLUCIÓN DE EJERCICIOS PROPUESTOS

Ejercicio 1

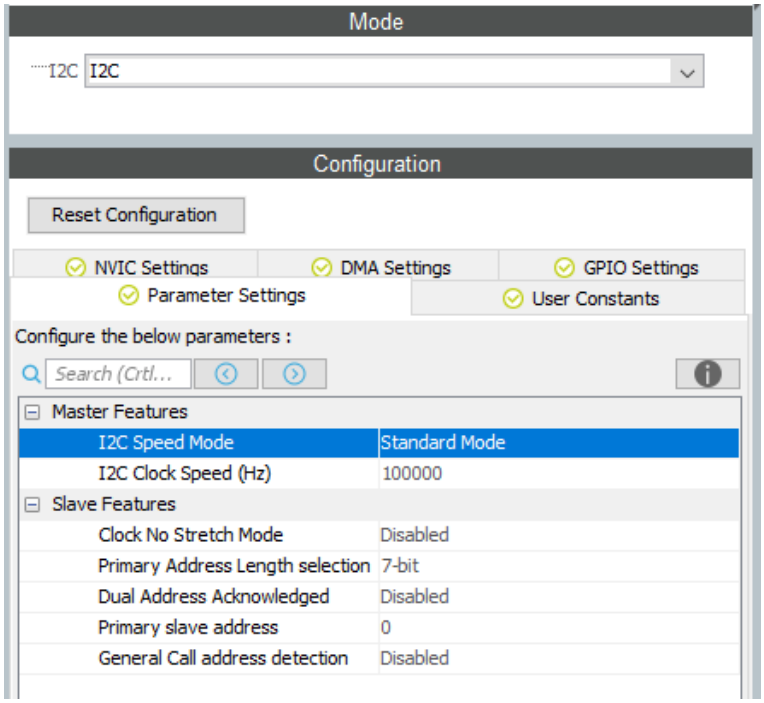
Se va a gestionar todo por interrupciones, ya que las callbacks pueden gestionar todo lo que se necesita. Para poder utilizar los pines, se va a prescindir de la conexión del LCD. De esta forma, el pinout queda de la siguiente manera:



La configuración de la USART queda así (junto con la activación del USART1 global interrupt):



La configuración del I2C queda así (junto con la activación del TIM2 global interrupt):



En cuanto al código, quedaría así:

```
/* USER CODE BEGIN PD */
#define IDLE      0
#define LECTURA   1
#define ESCRITURA 2
/* USER CODE END PD */
```

```
/* USER CODE BEGIN PV */
uint8_t texto_in[10];
uint8_t texto_out[10];
unsigned char estado=IDLE;
/* USER CODE END PV */
```

```
/* USER CODE BEGIN 0 */
// Callback para la Rx de la USART
void HAL_UART_RxCpltCallback(UART_HandleTypeDef *huart)
{
  // Recepción sin gestionar errores
  switch (estado) {
    case IDLE:
      if (texto_in[0]=='R') {
        estado = LECTURA;
        HAL_I2C_Mem_Read_IT(&hi2c1, 0x0050, 0, 256, texto_out, 10);
      }
      else if (texto_in[0]=='W') {
        estado = ESCRITURA;
        HAL_UART_Receive_IT(&huart1, texto_in, 10);
      }
      else {
        estado = IDLE;
      }
      break;
    case ESCRITURA:
      HAL_I2C_Mem_Write_IT(&hi2c1, 0x0050, 0, 256, texto_in, 10);
      break;
    default:
      estado = IDLE;
      break;
  }
}

// Callback de escritura en memoria
void HAL_I2C_MemTxCpltCallback (I2C_HandleTypeDef *hi2c) {
  HAL_UART_Receive_IT(&huart1, texto_in, 1);
  estado = IDLE;
}

// Callback de lectura de memoria
void HAL_I2C_MemRxCpltCallback (I2C_HandleTypeDef *hi2c) {
  HAL_UART_Transmit_IT(&huart1, texto_out, 10);
  HAL_UART_Receive_IT(&huart1, texto_in, 1);
  estado = IDLE;
}

/* USER CODE END 0 */
```

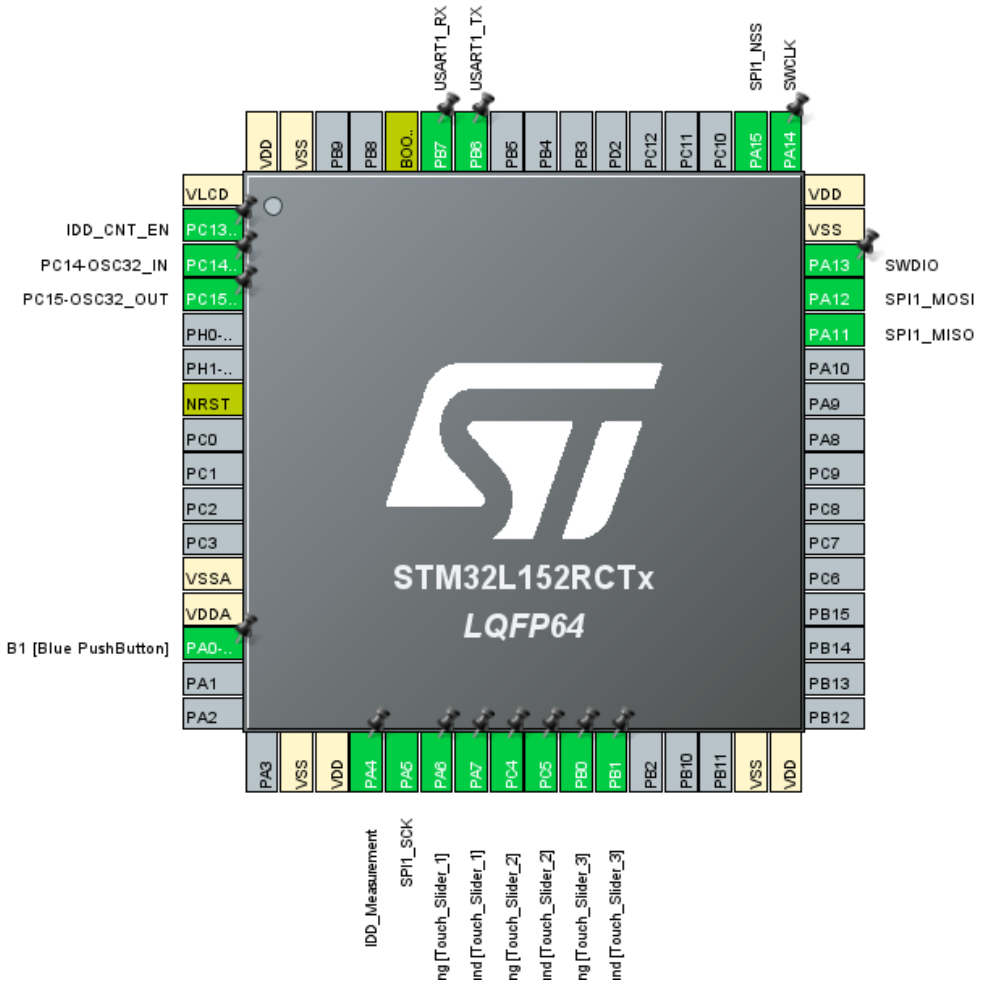
```
  /* USER CODE BEGIN 2 */
  // Init para USART en Rx
  HAL_UART_Receive_IT(&huart1, texto_in, 1);
  // Init para usar I2C
  while (HAL_I2C_IsDeviceReady(&hi2c1, 0x0050, 3, 2000)!=HAL_OK);
  /* USER CODE END 2 */
```

```
  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
```

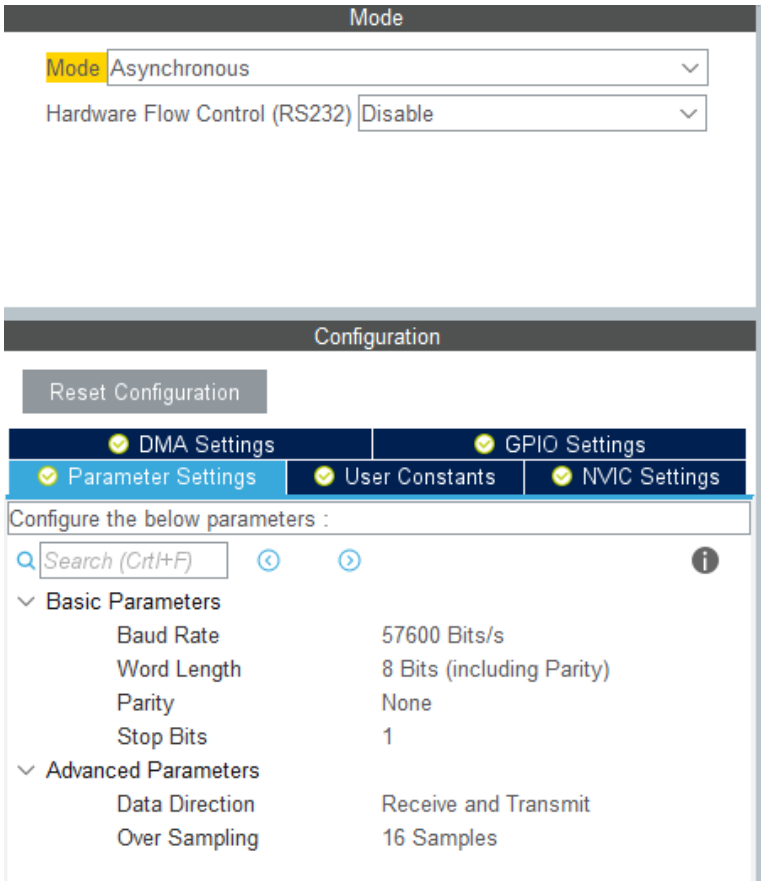
```
while (1)
{
  /* USER CODE END WHILE */
```

Ejercicio 2

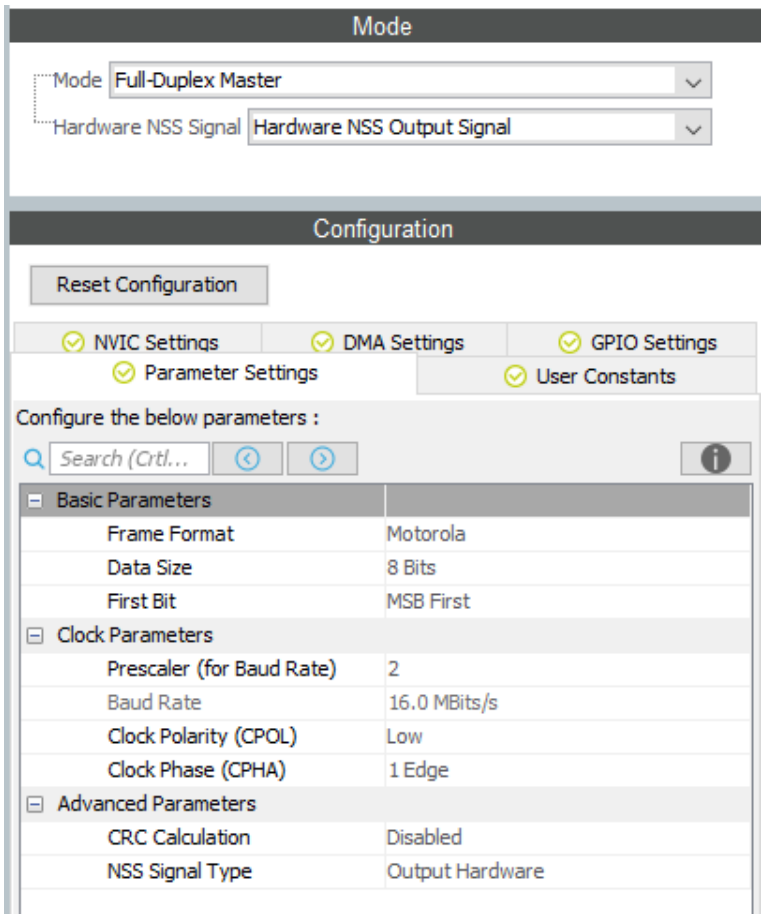
Se va a gestionar todo por interrupciones, ya que las callbacks pueden gestionar todo lo que se necesita. Para poder utilizar los pines, se va a prescindir de la conexión del LCD. De esta forma, el pinout queda de la siguiente manera:



La configuración de la USART queda así (junto con la activación del USART1 global interrupt):



La configuración del SPI queda así (junto con la activación del TIM2 global interrupt):



El código quedaría así:

```
/* USER CODE BEGIN PD */
#define IDLE      0
#define LECTURA   1
#define ESCRITURA 2
/* USER CODE END PD */
```

```
/* USER CODE BEGIN PV */
uint8_t texto_in[26] = {0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,0};
uint8_t texto_out[10];
unsigned char estado=IDLE;
/* USER CODE END PV */
```

```
/* USER CODE BEGIN 0 */
// Callback para la Rx de la USART
void HAL_UART_RxCpltCallback(UART_HandleTypeDef *huart)
{
  // Recepción sin gestionar errores
  switch (estado) {
    case IDLE:
      if (texto_in[16]=='R') {
        estado = LECTURA;
        texto_in[7]=1;
        HAL_SPI_Transmit_IT(&hspi1, texto_in, 16);
      }
      else if (texto_in[16]=='W') {
        estado = ESCRITURA;
        HAL_UART_Receive_IT(&huart1, &texto_in[16], 10);
      }
      else {
        estado = IDLE;
      }
      break;
    case ESCRITURA:
      texto_in[7]=0;
      HAL_SPI_Transmit_IT(&hspi1, texto_in, 26);
      break;
    default:
      estado = IDLE;
      break;
  }
}

// Callback de escritura en memoria
void HAL_SPI_TxCpltCallback (SPI_HandleTypeDef * hspi) {
  if (estado==LECTURA) {
    HAL_SPI_Receive_IT(&hspi1, texto_out, 10);
  }
  else {
    HAL_UART_Receive_IT(&huart1, &texto_in[16], 1);
    estado = IDLE;
  }
}

// Callback de lectura de memoria
void HAL_SPI_RxCpltCallback (SPI_HandleTypeDef * hspi) {
  HAL_UART_Transmit_IT(&huart1, texto_out, 10);
  HAL_UART_Receive_IT(&huart1, &texto_in[16], 1);
  estado = IDLE;
}

/* USER CODE END 0 */
```

```
  /* USER CODE BEGIN 2 */
  // Init para USART en Rx
  HAL_UART_Receive_IT(&huart1, texto_in, 1);
  /* USER CODE END 2 */
```

```
  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
    /* USER CODE END WHILE */
```