Ejemplo con cube IDE

La placa blue pill es una placa china por lo que requiere un programador para poder subir el programa, en este caso utilizaremos stLink v2.0 el cual se conecta al micro a través del siguiente diagrama :

3.3V – 3.3V

GND – GND

SWIN – SWIN

SWCLK – SWCLK

Luego de conectarlos se procede a abrir cubeIDE y se crea un nuevo proyecto en donde dependiendo del microcontrolador que se tenga se debe seleccionar la placa o el micro

Por ejemplo para una nucleo 64 se puede seleccionar la tarjeta con tranquilidad, pero para nuestro micro blue pill se debe seleccionar el micrcontrolador ya que no es una tarjeta oficial de tm32

Tarjeta Nuclel – F410RB

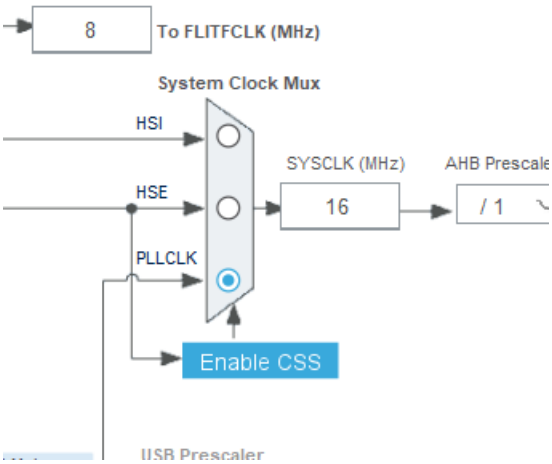
Blue pill – STM32F103T6

Se le da un nombre al proyecto y aparece la ventana de configuración de puertos en donde podemos generar código para cada puerto – Ejemplo para BLink

Para la blue pill todos los puertos aparecen desahabilitados sin configuraciones extras

Primero se configura el clock en clock Configuration y como se aprecia todo a la derecha es la configuración del clock, no se puede configurar el clock por que los puertos aun no están habilitados por lo que primero debemos de ir a la configuración de pines, en pinout configuration colocar en system core - > RCC -> Habilitar high speed clock

Dentro de la configuración de clock se puede trabajar hasta 16Mhz haciendo los cambios respectivos en las fuentes del microcontrolador



Luego seleccionar PC13 y colocarlo como output

Para configurar el programador en pinout configuration en system en debug seleccionar serial wire

Finalmente se procede a transformar toda la configuración a código

Esta herramienta grafica nos permitirá utilizar capas como HAL y poder generar nuestro código de forma predeterminada

Las funciones en este ejemplo se realizaran utilizando HAL, para lo cual usaremos las funciones HAL_GPIO_TogglePin(Puerto , PIN)

HAL_GPIO_TogglePin(GPIOC , GPIO_PIN_13)

Para cargar el código ir al PLAY verde :

Antes de hacer la primera carga lo mas probable es que les pidan actualizar su firmware, reconectar su programador y darle actualizar

Finalmente el programa estará subido y se vera el led parpadeando

Ejemplo Base:

```
  while (1)
  {
    /* USER CODE END WHILE */
        HAL_GPIO_TogglePin(GPIOC,GPIO_PIN_13);
        HAL_Delay(500);
        HAL_GPIO_TogglePin(GPIOC,GPIO_PIN_13);
        HAL_Delay(500);

    /* USER CODE BEGIN 3 */
  }
```

Ahora leer un pulsador con GPIO y HAL

Par configurar un puerto con HAL se debe de modificar el pin en la parte grafica como GPIOINPUt, verificar el uso de las resistencias PULL up O Pull Down EN LA VENTANA DE CONFIGURACION, NO OLVIDAR el tipo de entrada que se esta utilizando

Finalmente Se tienen las siguiente funciones con HAL:

HAL_GPIO_TogglePin(Puerto , PIN)

HAL_GPIO_WritePin(Puerto,PIN, estado)

PUERTO = GPIOC, PIO= GPIO_PIN_12, estado -> GPIO_PIN_SET, GPIO_PIN_RESET

PARA LEER UN PIN SE TIENE

HAL_GPIO_ReadPin(Puerto, PIN)

Funcion importante en donde se habilitan los clocks para luego comenzar con la configuración de los puertos , verificar como se configuran las salidas

```
static void MX_GPIO_Init(void)
{
  GPIO_InitTypeDef GPIO_InitStruct = {0};

  /* GPIO Ports Clock Enable */
  __HAL_RCC_GPIOC_CLK_ENABLE();
  __HAL_RCC_GPIOD_CLK_ENABLE();
  __HAL_RCC_GPIOA_CLK_ENABLE();

  /*Configure GPIO pin Output Level */
  HAL_GPIO_WritePin(GPIOC, GPIO_PIN_13, GPIO_PIN_RESET);

  /*Configure GPIO pin : PC13 */
  GPIO_InitStruct.Pin = GPIO_PIN_13;
  GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
  HAL_GPIO_Init(GPIOC, &GPIO_InitStruct);

}
```

Verificar que luego de configurar una entrada se puede ver como se borra todo el código y se reescribe con las nuevas configuraciones

```
static void MX_GPIO_Init(void)
{
  GPIO_InitTypeDef GPIO_InitStruct = {0};

  /* GPIO Ports Clock Enable */
  __HAL_RCC_GPIOC_CLK_ENABLE();
```

```
  __HAL_RCC_GPIOD_CLK_ENABLE();
  __HAL_RCC_GPIOA_CLK_ENABLE();

  /*Configure GPIO pin Output Level */
  HAL_GPIO_WritePin(GPIOC, GPIO_PIN_13, GPIO_PIN_RESET);

  /*Configure GPIO pin : PC13 */
  GPIO_InitStruct.Pin = GPIO_PIN_13;
  GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
  HAL_GPIO_Init(GPIOC, &GPIO_InitStruct);

  /*Configure GPIO pin : PC14 */
  GPIO_InitStruct.Pin = GPIO_PIN_14;
  GPIO_InitStruct.Mode = GPIO_MODE_INPUT;
  GPIO_InitStruct.Pull = GPIO_PULLUP;
  HAL_GPIO_Init(GPIOC, &GPIO_InitStruct);

}
```

Para configurar un ADC en Analog, seleccionar el ADC correspondiente y seleccionar el canal correspondiente, se observara como el puerto asociado se configurara en el microcontrolador y se podrá realizar las configuraciones pertinentes

En la parte inferior se habilitara el modo de conversión continua, realizar dicha configuración con PB1 y generar el código correspondiente.

Luego agregar las funciones pertinentes para poder leer el valor de la entrada digital

Verificar que se ha agregado esta función con la cofiguracion del ADC

```
static void MX_ADC1_Init(void)
{

  /* USER CODE BEGIN ADC1_Init 0 */

  /* USER CODE END ADC1_Init 0 */

  ADC_ChannelConfTypeDef sConfig = {0};

  /* USER CODE BEGIN ADC1_Init 1 */

  /* USER CODE END ADC1_Init 1 */

  /** Common config
  */
  hadc1.Instance = ADC1;
  hadc1.Init.ScanConvMode = ADC_SCAN_DISABLE;
  hadc1.Init.ContinuousConvMode = ENABLE;
```

```
  hadc1.Init.DiscontinuousConvMode = DISABLE;
  hadc1.Init.ExternalTrigConv = ADC_SOFTWARE_START;
  hadc1.Init.DataAlign = ADC_DATAALIGN_RIGHT;
  hadc1.Init.NbrOfConversion = 1;
  if (HAL_ADC_Init(&hadc1) != HAL_OK)
  {
    Error_Handler();
  }

  /** Configure Regular Channel
  */
  sConfig.Channel = ADC_CHANNEL_9;
  sConfig.Rank = ADC_REGULAR_RANK_1;
  sConfig.SamplingTime = ADC_SAMPLETIME_1CYCLE_5;
  if (HAL_ADC_ConfigChannel(&hadc1, &sConfig) != HAL_OK)
  {
    Error_Handler();
  }
  /* USER CODE BEGIN ADC1_Init 2 */

  /* USER CODE END ADC1_Init 2 */

}
```

Para leer el ADC se tiene la siguiente función

```
  MX_GPIO_Init();
  MX_ADC1_Init();
  /* USER CODE BEGIN 2 */
  HAL_ADC_Start(&hadc1);
  /* USER CODE END 2 */

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
    /* USER CODE END WHILE */
        HAL_ADC_PollForConversion(&hadc1,1000);
        readValue = HAL_ADC_GetValue(&hadc1);
    /* USER CODE BEGIN 3 */
  }
  /* USER CODE END 3 */
}

/**
```

Se tiene que iniciar el ADC con HAL_ADC_START y pasar como parámetro el adc configurado, luego se debe esperar para laconversion con HAL_ADC_pollforconversion (el 1000 es el timeout) y finalmente se debe leer el valor con HAL_ADC_GetValue

Por el momento no podemos ver cosas físicas aun, estamos realizando las configuraciones

Hacer ejemplo blink con registros -- rE

Para la configuración a través de los registros se tiene los siguientes ejemplos

Los ejemplos de la raspberry pi pico hacerlos con micropython

```
# Paso 1
from machine import Pin
# Paso 2
import time
# Paso 3
led_1 = Pin(25, Pin.OUT)
while True:
    # Paso 4
    led_1.value(1)
    time.sleep(0.5)
    led_1.value(0)
    time.sleep(0.5)
```

```
import machine
import utime

analog_value = machine.ADC(28)

while True:
    reading = analog_value.read_u16()
    print("ADC: ",reading)
    utime.sleep(0.2)
```

https://docs.micropython.org/en/latest/rp2/quickref.html#pins-and-gpio