Universidad Tecnológica de Torreón
Organismo Público Descentralizado del Gobierno del Estado de Coahuila

MICROCONTROLADORES

Práctica No. 21. USART.

1. Objetivo

- Uso del ADC y el módulo USART del STM32F103.

2. Material y Equipo.

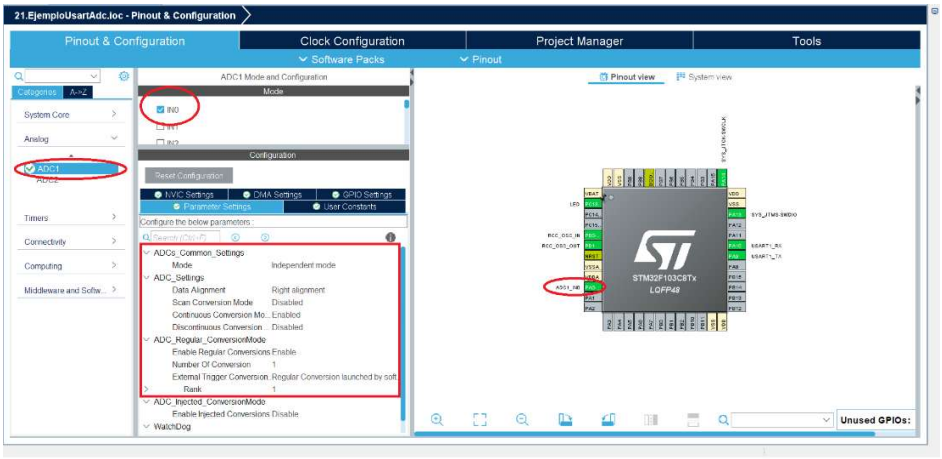
- Computador o laptop con STM32CubeIDE.
- Un Sensor de temperatura LM35.
- Un diodo LED.

3. Marco de Referencia.

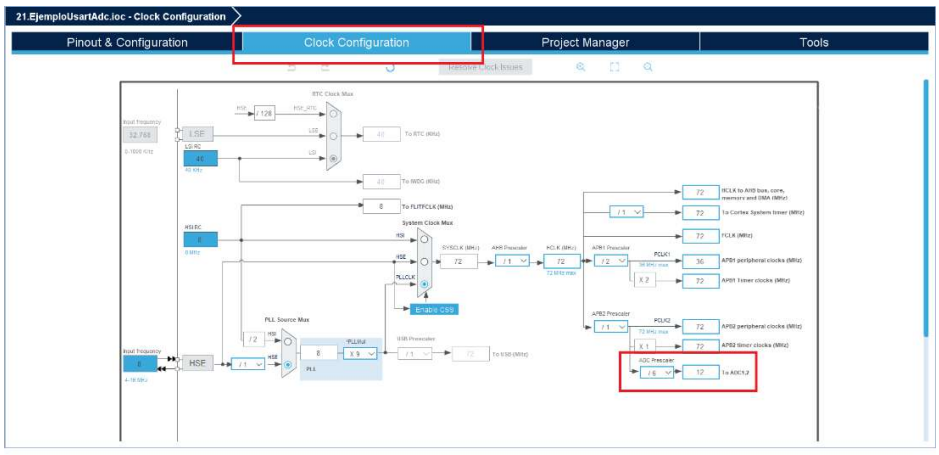
La USART es un modulo del STM32F103 que tiene dos modos de operación el modo síncrono y el modo asíncrono para este modulo solo usaremos el modo asíncrono ya que el síncrono se cubrirá en practicas posteriores con los módulos SPI e I^2C. La USART nos permite comunicarnos con otros dispositivos el más común una computadora, pero también es posible comunicarse con módulos celulares, modulo GPS, algunas pantallas LCD, TFT y módulos diseñados para esta interfaz especifica. Para usar la USART tenemos que configurar la velocidad, los bits a transmitir, la paridad y los bits de parada. Para configurar la USART lo haremos desde el STM32CubeMX, donde configuraremos velocidad, paridad y la interrupción por recepción de datos seriales.

4. Desarrollo y Procedimiento.

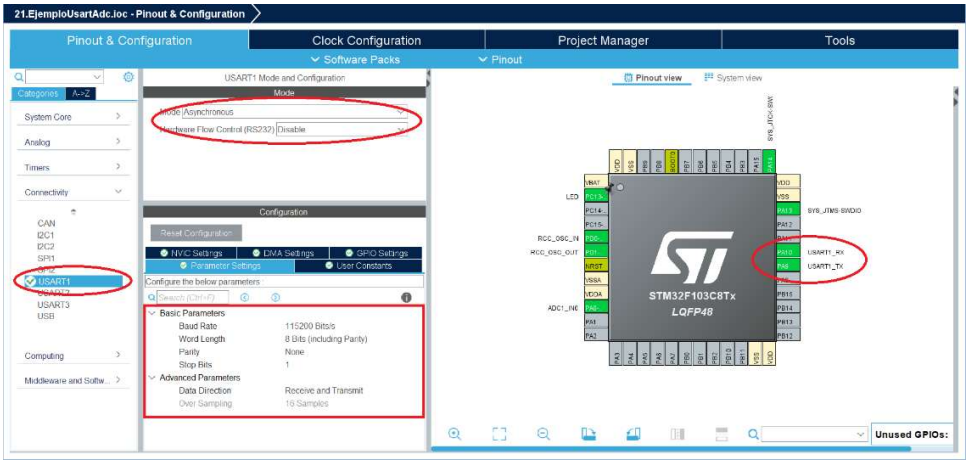
Se creará un proyecto en el STM32CubeIDE como se indicó anteriormente. Primero configuramos el ADC usando la misma configuración de la práctica anterior, como se muestra en la siguiente figura.



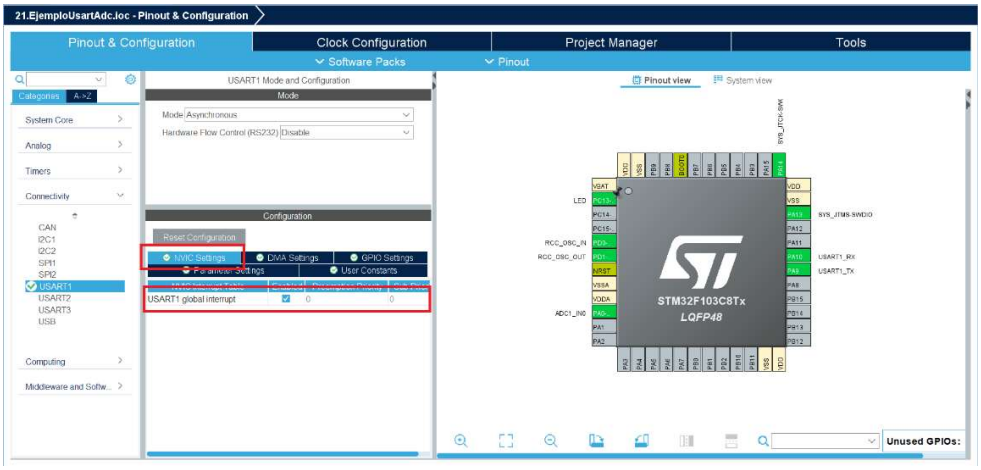
Después configuramos el prescaler del ADC.



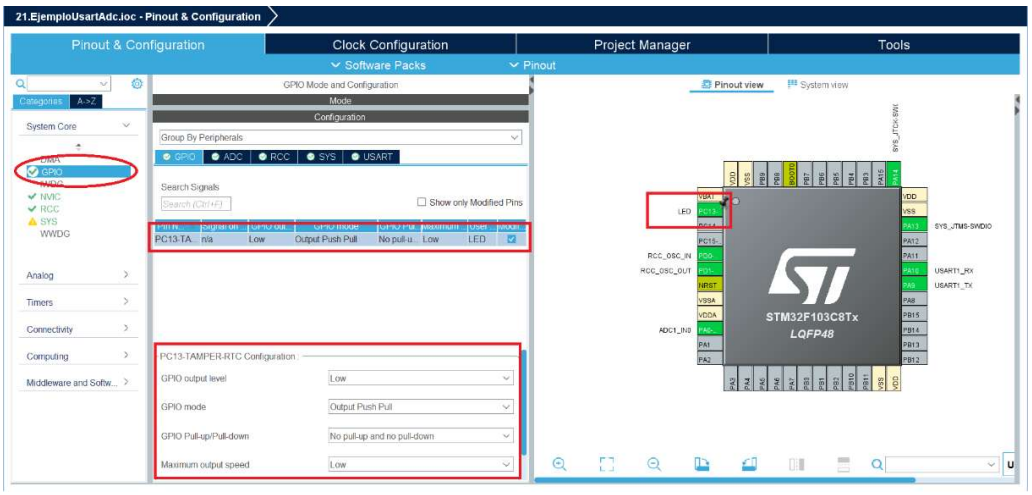
Ahora configuramos la USART1.



Ahora habilitamos la interrupción por recepción de la USART1.



Por último, habilitamos el pin PC13 como salida para el LED.

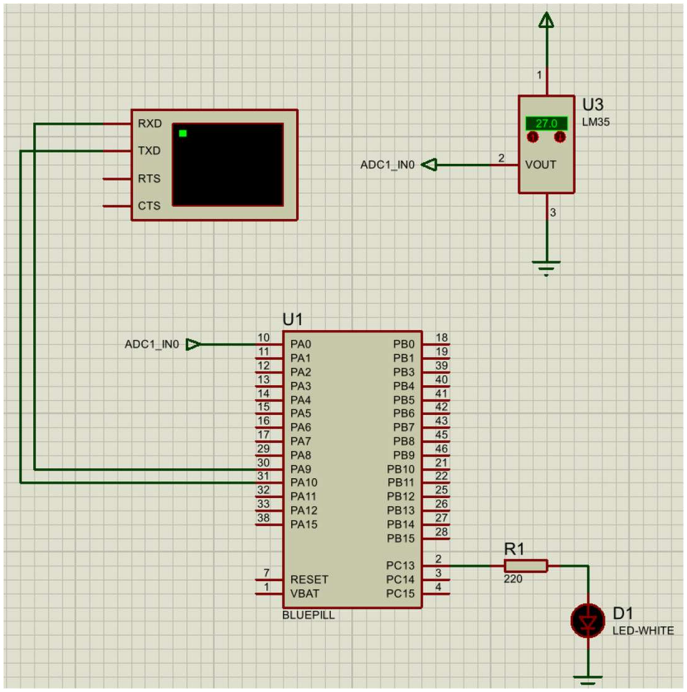


El código de la práctica es el siguiente en el archivo main.c.

```c
#include "main.h"
#include "adc.h"
#include "usart.h"
#include "gpio.h"
#include <stdio.h>
#include <string.h>

uint8_t datoUsart;
volatile uint8_t flagUsart;
uint32_t adcResult;
float voltaje, temperatura;
char strUsart[50];

void SystemClock_Config(void);

void HAL_UART_RxCpltCallback(UART_HandleTypeDef *huart){
    flagUsart = 1;
    HAL_UART_Receive_IT(&huart1, &datoUsart, 1);
}
int main(void)
{
    HAL_Init();
    SystemClock_Config();
    MX_GPIO_Init();
    MX_ADC1_Init();
    MX_USART1_UART_Init();
    HAL_ADC_Start(&hadc1);
    HAL_UART_Receive_IT(&huart1, &datoUsart, 1);

    while (1)
    {
        if(flagUsart){
            flagUsart = 0;
            switch(datoUsart){
                case 'A':
                    HAL_GPIO_WritePin(LED_GPIO_Port, LED_Pin, GPIO_PIN_SET);
                    break;
                case 'a':
                    HAL_GPIO_WritePin(LED_GPIO_Port, LED_Pin, GPIO_PIN_RESET);
                    break;
                case 'H':
                    HAL_ADC_PollForConversion(&hadc1, 100);
                    adcResult = HAL_ADC_GetValue(&hadc1);
                    voltaje = (((float)adcResult) * 3.3) / 4095.0;
                    temperatura = (float)(voltaje * 100.0);
                    sprintf(strUsart, "%0.2f\r\n", temperatura);
                    HAL_UART_Transmit(&huart1, (uint8_t*)strUsart, strlen(strUsart), HAL_MAX_DELAY);
                    break;
            }
        }
    }
}
```

5. Esquemático del circuito.

El esquemático de la práctica se muestra a continuación.



6. Mejora.

La mejora de esta practica es desplegar la temperatura en grados Centígrados, Fahrenheit y tendrá un led que indique una alarma de temperatura alta este límite se establecerá en una interfaz de Visual Basic.

7. Observaciones.

Esta sección es para que el alumno anote sus observaciones.

8. Conclusiones.

Esta sección es para que el alumno anote sus conclusiones.

9. Importante.

La práctica deberá ser validad en el salón de clases antes de anexar el reporte al manual de prácticas. Una vez validad realizar el reporte de practica como se anteriormente y anexar al manual de prácticas que se entregara a final del curso.