ใบงานครั้งที่ 12

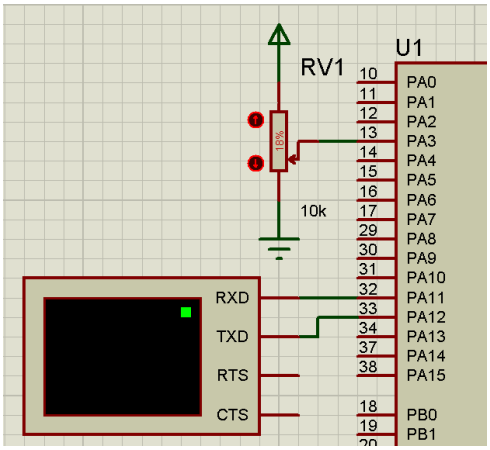
Analog-to-Digital Converter (ADC)

<u>อุปกรณ์ที่ใช้ในการทดลอง</u>

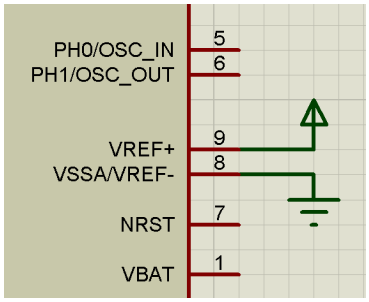
1. STM32CubeIDE
2. Proteus 8.9 SP 2

<u>ขั้นตอนการทดลอง</u>

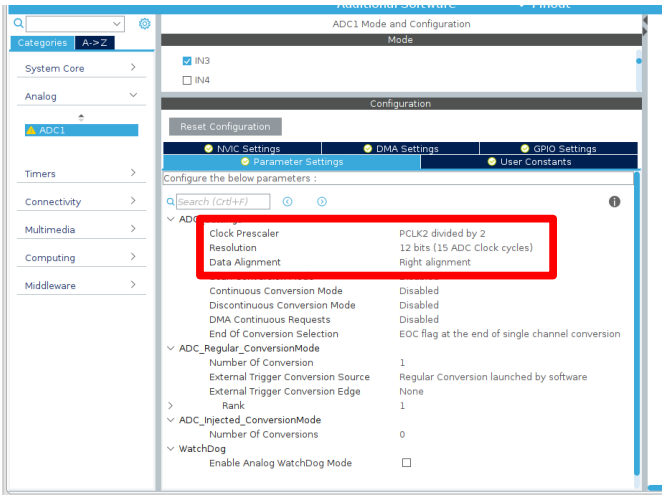
1. ต่อวงจรตามรูปด้านล่าง



2. เนื่องจากโมดูล ADC ต้องการแรงดันอ้างอิง (VREF) ในการเปรียบเทียบแรงดันจึงต้องต่อแรงดันที่ VREF ด้วย



3. ทำการ configure ADC โดยทำการ tick ที่ IN3 ซึ่งความละเอียดสามารถกำหนดได้ที่ Resolution จำนวนบิตความละเอียดสูงสุดอยู่ที่ 12 bits



4. โค้ดที่ generate ที่เกี่ยวข้องกับ ADC

```c
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 */
  /** Configure the global features of the ADC (Clock, Resolution, Data Alignment and number of conversion)
  */
  hadc1.Instance = ADC1;
  hadc1.Init.ClockPrescaler = ADC_CLOCK_SYNC_PCLK_DIV2;
  hadc1.Init.Resolution = ADC_RESOLUTION_12B;
  hadc1.Init.ScanConvMode = DISABLE;
  hadc1.Init.ContinuousConvMode = DISABLE;
  hadc1.Init.DiscontinuousConvMode = DISABLE;
  hadc1.Init.ExternalTrigConvEdge = ADC_EXTERNALTRIGCONVEDGE_NONE;
  hadc1.Init.ExternalTrigConv = ADC_SOFTWARE_START;
  hadc1.Init.DataAlign = ADC_DATAALIGN_RIGHT;
  hadc1.Init.NbrOfConversion = 1;
  hadc1.Init.DMAContinuousRequests = DISABLE;
  hadc1.Init.EOCSelection = ADC_EOC_SINGLE_CONV;
  if (HAL_ADC_Init(&hadc1) != HAL_OK)
  {
    Error_Handler();
  }
  /** Configure for the selected ADC regular channel its corresponding rank in the sequencer and its sample time.
  */
  sConfig.Channel = ADC_CHANNEL_3;
  sConfig.Rank = 1;
  sConfig.SamplingTime = ADC_SAMPLETIME_3CYCLES;
  if (HAL_ADC_ConfigChannel(&hadc1, &sConfig) != HAL_OK)
  {
    Error_Handler();
  }
  /* USER CODE BEGIN ADC1_Init 2 */

  /* USER CODE END ADC1_Init 2 */

}
```

5. เนื่องจากจะใช้งาน function ที่เกี่ยวข้องกับการจัดการข้อความ จึงทำการ include library ทั้งสองตัว

```c
  */
/* USER CODE END Header */

/* Includes ------------------------------------------------------------------*/
#include "main.h"

/* Private includes ----------------------------------------------------------*/
/* USER CODE BEGIN Includes */
#include <stdio.h>
#include <string.h>
/* USER CODE END Includes */

/* Private typedef -----------------------------------------------------------*/
/* USER CODE BEGIN PTD */
```

6. ทำการสร้างตัวแปรขึ้นมาเพื่อใช้ในการจัดการข้อความและเก็บค่าที่ได้จาก ADC

```c
int main(void)
{
  /* USER CODE BEGIN 1 */
    uint32_t raw_adc = 0;
    char msg[100];

  /* USER CODE END 1 */

  /* MCU Configuration--------------------------------------------------------*/

  /* Reset of all peripherals, Initializes the Flash interface and the Systick. */
  HAL_Init();
```

7. ทำการ convert แรงดันที่เข้ามาและเก็บค่าที่ได้ลงตัวแปร raw_adc (12 bits)

- HAL_ADC_Start: ทำการเริ่ม convert แรงดัน
- HAL_ADC_PollForConversion: ทำการตรวจสอบการ convert เสร็จสิ้น
- HAL_ADC_GetValue: อ่านค่าที่ได้จาก ADC ที่เสร็จสิ้นแล้ว

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

    /* USER CODE BEGIN 3 */
      HAL_ADC_Start(&hadc1);
      HAL_ADC_PollForConversion(&hadc1, HAL_MAX_DELAY);
      raw_adc = HAL_ADC_GetValue(&hadc1);
      sprintf(msg,"Raw ADC: %d %%\r\n", (int)(raw_adc*100/4095));
      HAL_UART_Transmit(&huart6, (uint8_t*)msg, strlen(msg), 100);

      HAL_Delay(500);
  }
  /* USER CODE END 3 */
```

8. ทำการ simulate แล้วปรับค่าความต้าน สังเกตุค่าที่อ่านได้

<u>คำถามท้ายใบงาน</u>

1. ให้ออกแบบวงจรที่อ่านค่าจากวงจรแบ่งแรงดันผ่าน ADC จากนั้นให้แสดงค่าข้อมูลดิบ (raw value) ที่ได้ทางหน้าจอ LCD และตรวจสอบค่าที่อ่านมาได้ หากข้อมูลที่มาได้มีค่ามากกว่า 60% ของแรงดัน ให้ LED สีเขียวติด แต่ถ้าน้อยกว่าให้ LED สีแดงติด