# Features of project

1. Stm communicates with other stm using uart (UART)
2. Sends Phot-registers reading (ADC + UART)
3. Send update to the computer or log file (UART)
4. Receive data from other stm and response by setting light state (UART)
5. Reveive data from computer and turns on led or off (UART)
6. Timer: a particular data will be sent in a fixed delay.

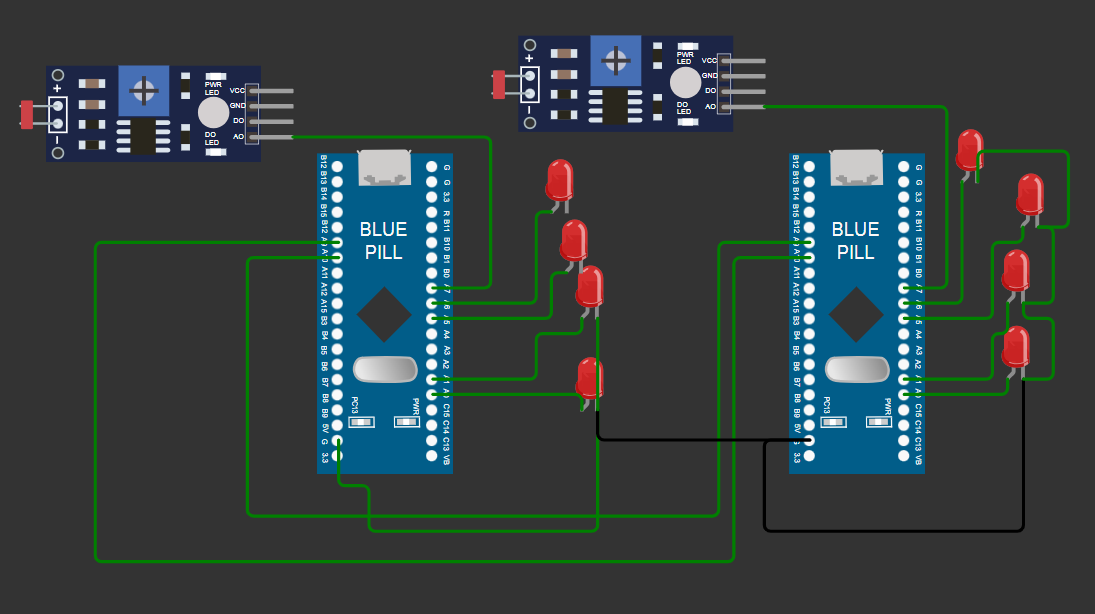
# Features of stm32 used in project

1. UART
2. ADC
3. Systick timer
4. Time gpio trigger
5. All in interrupt

# Project functions

1. Function for clock
2. Function for pin config
3. Function for systick setup
4. Function for time delay
5. Function for UART enable (both uart 1 and 2)
6. Function for ADC
7. Timer config function
8. All interrupt vector function
9. Main function.

# Circuit diagram:

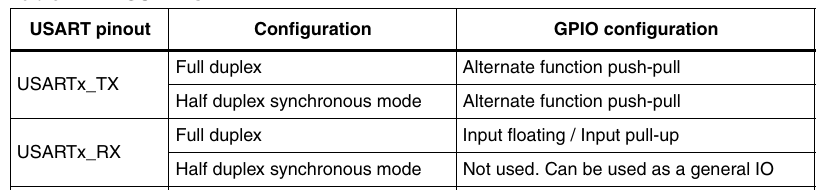


# Clock enable

* APB1ENR UART2, Timer2
* APB2ENR UART1, Systick, PA,ADC, AFIO

# Pin config:

## For UART:



* PA9: UART1 Tx : Output AF push-pull
* PA10: UART1 Rx : Input Floating
* PA2: UART2: Tx output AF push-pull
* PA3: UART2 Rx: input Floating
* PA0: TX signal for uart1 : GPIO output pushpull
* PA1: RX signal for uart1 : GPIO output pushpull
* PA4: Response of received data from USART1 : GPIO OUTput pushpull
* PA5: TxRx signal for UART2: GPIO output push pull
* pA6: Rx signal for UART2: GPIO output push pull
* PA7: Response of received data from USART2 : GPIO output pushpull

# SysTick Config:

1. SysTick Load: 72000-1;
2. Clock source cpu

# Timer setup:

1. Set counter value 0 in TIM2->CNT
2. Set the TIM2-> PSC value. CK\_CNT = fCK\_PSC / (PSC[15:0] + 1). (for 1ms, 7200-1)
3. Load autoreload register TIM2->ARR
4. Enable Update Interrupt in TIM2->DIER bit 0;
5. Set count Direction in TIM->2CR1 bit 4. Set is downcounter
6. Enable Timer in TIM2->CR1 bit 0;
7. Write interrupt vector runtime.

# UART setup

1. IN CR1:
   1. Enable RE in bit 2
   2. Enable TE in bit 3
   3. Eable RXNEIE (Rx not empty interrupt enable) in bit 5
   4. Eable UE( USART enable) in bit 13
2. Set baud rate in USART->BRR
3. Config in , Tx as AF output, RX as GPIO input push-pull
4. Enabe UART interrupt vector (NVIC)
5. Load data to USART->DR to start transmission
6. When data is received, RXNE will be set in SR and an interrupt will be call call. Write code in interrupt handeler.