# 3200 Project

# Project Features

## UART Communication

The STM32 communicates with another STM32 using UART.

* When data is sent, PA0 blinks.
* When data is received, PA1 blinks.

## Timer-Based Data Sending

A general-purpose timer sends data (from 0 to 3 in sequence) at regular intervals (3s). PA4 and PA7 act as indicators for the received data:

* PA4 shows the LSB (Least Significant Bit)
* PA7 shows the MSB (Most Significant Bit)

## Photoresistor Reading and Threshold Detection (ADC + UART)

The system reads values from a photoresistor.

* If the value goes above a high threshold, PA5 turns ON.
* If the value goes below a low threshold, PA5 turns OFF on the other STM32.

## External Interrupt Handling

When an external interrupt occurs on PA2, a signal is sent.

* The other STM32 responds by toggling PA6.

# STM32 Features Used in the Project

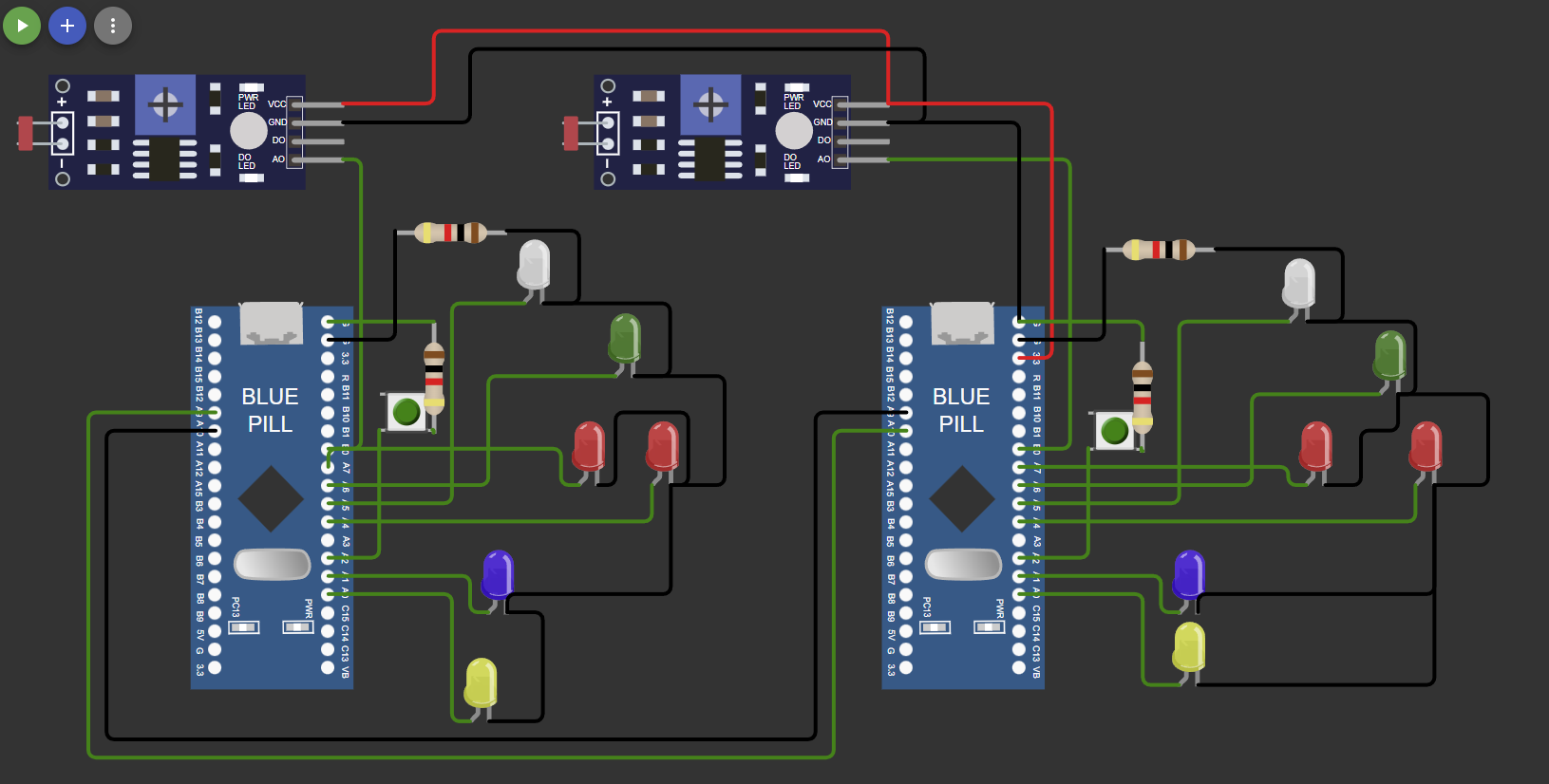
* UART
* ADC
* SysTick timer
* Timer GPIO trigger
* External interrupt
* All handled via interrupts

# Project functions

### **Project Functions**

1. **Function for clock**
   * void En\_clock(void)
2. **Function for pin config**
   * void gpio\_setup(void)
3. **Function for systick setup**
   * void systick\_config(void)
4. **Function for time delay**
   * void delay\_ms(void)
   * void delay(uint32\_t count)
5. **Function for UART config**
   * void Uart1\_config(void)
6. **Function for ADC config**
   * void ADC\_config(void)
7. **Timer config function**
   * void timer\_config(void)
8. **All interrupt vector functions**
   * void TIM2\_IRQHandler(void)
   * void USART1\_IRQHandler(void)
   * void ADC1\_2\_IRQHandler(void)
   * void EXTI2\_IRQHandler(void)
9. **Main function**
   * int main(void)

# Circuit diagram:



# Pin config:

|  |  |  |  |
| --- | --- | --- | --- |
| **Pin Name** | **Purpose** | **Config** | **CNF[1:0] + MODE[1:0]** |
| **PA0** | TX indication (blink on send) | GPIO Output Push-Pull | 0011 |
| **PA1** | RX indication (blink on receive) | GPIO Output Push-Pull | 0011 |
| **PA2** | External interrupt input | GPIO Input with Pull-up/down | 1000 |
| **PA4** | Display LSB of data send by Timer interrupt | GPIO Output Push-Pull | 0011 |
| **PA5** | Output control (based on photoresistor) | GPIO Output Push-Pull | 0011 |
| **PA6** | Toggle on external interrupt response | GPIO Output Push-Pull | 0011 |
| **PA7** | Display MSB of data send by Timer interrupt | GPIO Output Push-Pull | 0011 |
| **PA9** | UART Transmission (USART1\_TX) | Alternate Function Push-Pull | 1011 |
| **PA10** | UART Reception (USART1\_RX) | Input Floating | 0100 |
| **PB0** | Photoresistor input (ADC) | Analog Input | 0000 |

# 1. Function for Clock Enable

void En\_clock(void)

* Enables peripheral clocks:
  + APB1ENR: USART2, TIM2
  + APB2ENR: AFIO, IOPA, IOPB, USART1, ADC1

# 2. Function for Pin Config

void gpio\_setup(void)

* Configures GPIO pins:
  + PA0–PA7: Various input/output configurations for UART, ADC, EXT, response signals
  + PA9: UART1 Tx → AF Output Push-Pull (1011)
  + PA10: UART1 Rx → Input floating (0100)
  + PB0: Analog input (ADC) → 0000

# 3. Function for SysTick Setup

void systick\_config(void)

* SysTick LOAD = 72000-1 (for 1 ms delay at 72 MHz)
* Clock source: CPU
* Enables SysTick

# 4. Function for Time Delay

void delay\_ms(void) and void delay(uint32\_t count)

* delay\_ms() waits for the COUNTFLAG of SysTick
* delay(count) repeats delay\_ms() for count times

# 5. Function for UART Config

void Uart1\_config(void)

* CR1:
  + TE (bit 3): Transmitter enable
  + RE (bit 2): Receiver enable
  + RXNEIE (bit 5): Rx not empty interrupt enable
  + UE (bit 13): USART enable
* Baud Rate: USART1->BRR = 0x1DCC (9600 for 72 MHz)
* GPIO Config: PA9 (Tx), PA10 (Rx)
* NVIC interrupt enable: USART1\_IRQn
* Transmit: Write to USART1->DR
* Receive: USART1\_IRQHandler handles reception using RXNE flag

# 6. Function for ADC Config

void ADC\_config(void)

* CR1:
  + AWDEN (bit 23): Analog Watchdog enable
  + AWDIE (bit 6): Analog Watchdog interrupt enable
  + AWDCH = 8 (B0 = Channel 8 → 01000)
* CR2:
  + CONT (bit 1): Continuous conversion
  + ADON (bit 0): ADC enable (called twice as per manual)
* SQR3: Set channel 8 (B0)
* HTR/LTR: High/Low threshold for watchdog
* Interrupt: ADC1\_2\_IRQHandler
  + SR.AWD checked
  + Action based on ADC1->DR compared to thresholds
  + AWD flag cleared after handling

# 7. Timer Config Function

void timer\_config(void)

* CNT = 0
* PSC = 7200 - 1 ( CK\_CNT = 72MHz / 7200 = 10KHz)
* ARR = 50000 ( 5s period)
* DIER bit 0: Enable update interrupt
* CR1 bit 4: Downcount (DIR)
* CR1 bit 0: Timer Enable
* NVIC: TIM2\_IRQn enabled
* ISR: Toggle PA0, send UART1 Sdata and increment cyclically

# 8. All Interrupt Vector Function

## USART1\_IRQHandler

* Check RXNE
* Read from USART1->DR
* Conditional actions using GPIO depending on Rdata
* Clear RXNE flag
* Delay and reset output pin

## TIM2\_IRQHandler

* Check UIF, clear it
* Toggle PA0
* Send Sdata via UART1
* Cycle Sdata from 0 to 3

## ADC1\_2\_IRQHandler

* Read ADC1->DR
* If AWD:
  + Compare with HTR and LTR
  + Set present flag
  + If changed from past, send 'B' or 'C' via UART1
* Clear AWD flag at the end

## EXTI2\_IRQHandler

* Clear pending bit
* Toggle PA0
* Send 'A' via UART1

# 9. Main Function

int main(void)

* Calls:
  + En\_clock()
  + gpio\_setup()
  + systick\_config()
  + timer\_config()
  + Uart1\_config()
  + Exi\_config()
  + ADC\_config()
* Infinite loop: waits for interrupts to perform tasks