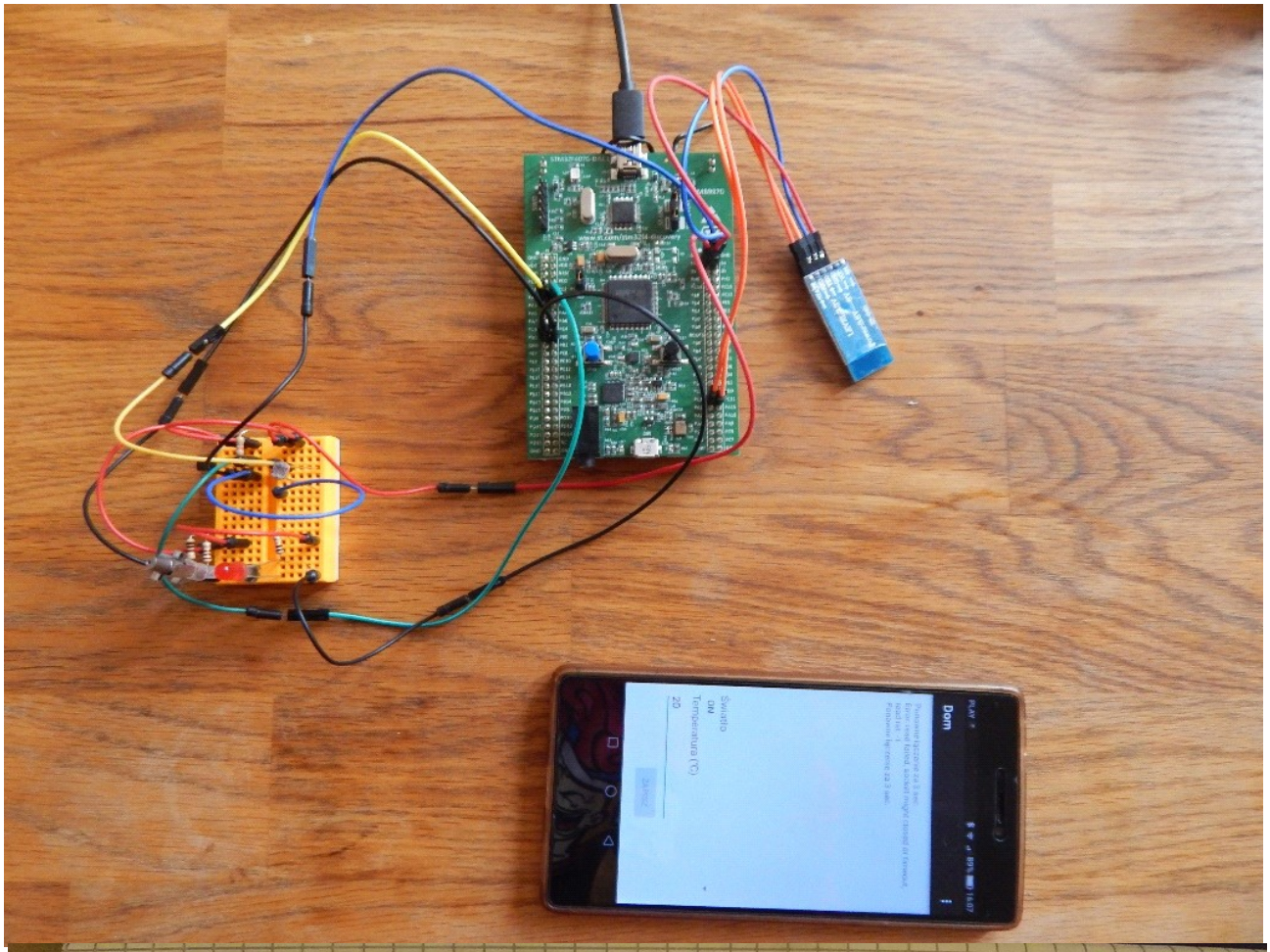
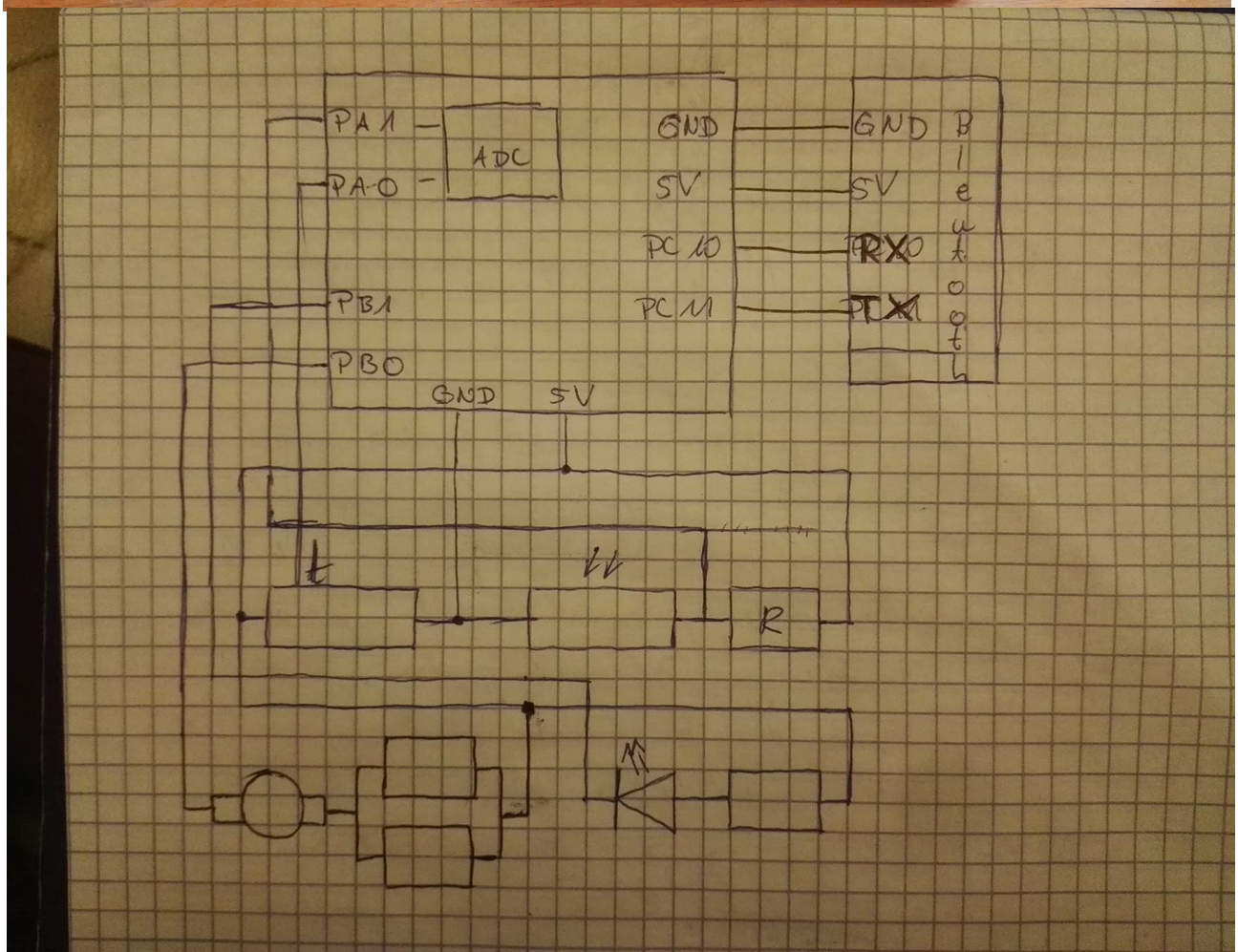
Dokumentacja Projektu
Inteligentny Dom

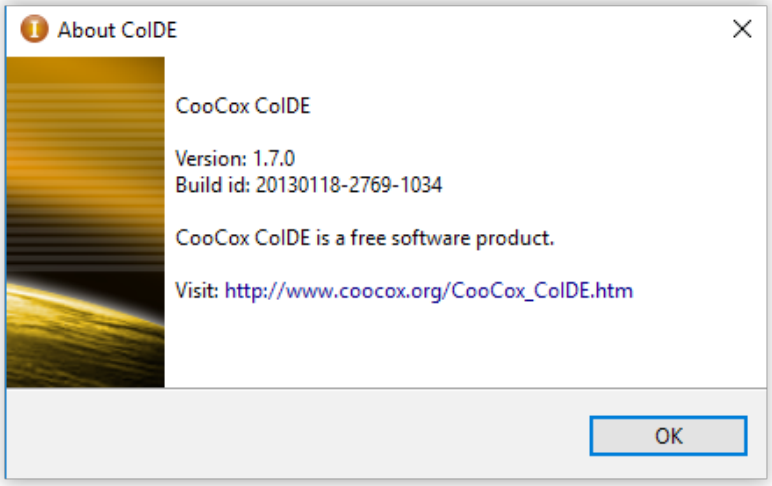
Opis ogólny :



Za pomocą aplikacji mobilnej odbywa się sterowanie domem, czyli oświetleniem i temperaturą. Oświetlenie można ustawić w trzech pozycjach – włączone na stałe, wyłączone na stałe oraz auto. Tryb auto to tryb, którym w zależności od oświetlenia zewnętrznego włącza lub wyłącza się oświetlenie domowe. Temperaturą sterujemy zadając temperaturę graniczną. Gdy czujnik wyczuje, że w domu jest cieplej niż zadana temperatura wtedy włącza chłodzenie. Komunikacja między urządzeniami a aplikacją odbywa się za pomocą Bleutooth.

Wersja środowiska :



```c
#include "stm32f4xx_conf.h"
#include "stm32f4xx_gpio.h"
#include "stm32f4xx_rcc.h"
#include "stm32f4xx_adc.h"
#include "stm32f4xx_usart.h"
#include "stm32f4xx_exti.h"
#include "misc.h"
#include <stdio.h>

float ADC_Result1, ADC_Result2;
float temp;
volatile uint32_t a;
float C;
int PL=1;
int Dioda=2500;
int Liczba1,Liczba2;
```

Dołączanie bibliotek i inicjalizacja zmiennych globalnych ,należy pamiętać o załączeniu potrzebnych bibliotek w sekcji Repository.

```c
int Liczba1,Liczba2;
void USART3_IRQHandler(void)
{
    // sprawdzenie flagi zwiazanej z odebraniem danych przez USART
    if(USART_GetITStatus(USART3, USART_IT_RXNE) != RESET)
      {
            a=USART3->DR;
                switch(a){
                case '0':
                case '1':
                case '2':
                case '3':
                case '4':
                case '5':
                case '6':
                case '7':
                case '8':
                case '9':
                case '*':
                    {
                        if(PL==1)
                        {
                            Liczba1=(int)a-48;
                            PL=0;
                        }
                        else
                        {  if(a=='*'){
                            C=Liczba1;
                        }else
                        {
                            Liczba2=(int)a-48;
                            C=Liczba1*10+Liczba2;
                        }
                            PL=1;
                        }
                        break;
                        }
                case'a':{
                        Dioda=0;
                        break;
                    }
                case'b':{
                        Dioda=2500;
                        break;
                        }
                case'c':{
                        Dioda=4095;
                        break;
```

Funkcja obsługująca przerwanie USART która wykonuje się przy wysłaniu danych do urządzania

```c
int main(void)
{
    SystemInit();

    RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOA , ENABLE);
    RCC_APB2PeriphClockCmd(RCC_APB2Periph_ADC1, ENABLE);
    RCC_APB2PeriphClockCmd(RCC_APB2Periph_ADC2, ENABLE);

    GPIO_InitTypeDef GPIO_InitStructure;
    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_0|GPIO_Pin_1;
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_AN;
    GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_NOPULL;
    GPIO_Init(GPIOA, &GPIO_InitStructure);

    ADC_CommonInitTypeDef ADC_CommonInitStructure;
    ADC_CommonInitStructure.ADC_Mode = ADC_Mode_Independent;
    ADC_CommonInitStructure.ADC_Prescaler = ADC_Prescaler_Div2;
    ADC_CommonInitStructure.ADC_DMAAccessMode = ADC_DMAAccessMode_Disabled;
    ADC_CommonInitStructure.ADC_TwoSamplingDelay = ADC_TwoSamplingDelay_5Cycles;
    ADC_CommonInit(&ADC_CommonInitStructure);

    ADC_InitTypeDef ADC_InitStructure;
    ADC_InitStructure.ADC_Resolution = ADC_Resolution_12b;
    ADC_InitStructure.ADC_ScanConvMode = DISABLE;
    ADC_InitStructure.ADC_ContinuousConvMode = ENABLE;
    ADC_InitStructure.ADC_ExternalTrigConv = ADC_ExternalTrigConv_T1_CC1;
    ADC_InitStructure.ADC_ExternalTrigConvEdge = ADC_ExternalTrigConvEdge_None;
    ADC_InitStructure.ADC_DataAlign = ADC_DataAlign_Right;
    ADC_InitStructure.ADC_NbrOfConversion = 1;
    ADC_Init(ADC1, &ADC_InitStructure);

    ADC_InitStructure.ADC_Resolution = ADC_Resolution_12b;
    ADC_InitStructure.ADC_ScanConvMode = DISABLE;
    ADC_InitStructure.ADC_ContinuousConvMode = ENABLE;
    ADC_InitStructure.ADC_ExternalTrigConv = ADC_ExternalTrigConv_T1_CC1;
    ADC_InitStructure.ADC_ExternalTrigConvEdge = ADC_ExternalTrigConvEdge_None;
    ADC_InitStructure.ADC_DataAlign = ADC_DataAlign_Right;
    ADC_InitStructure.ADC_NbrOfConversion = 1;
    ADC_Init(ADC2, &ADC_InitStructure);

    ADC_RegularChannelConfig(ADC1, ADC_Channel_1, 1, ADC_SampleTime_84Cycles);
    ADC_RegularChannelConfig(ADC2, ADC_Channel_2, 0, ADC_SampleTime_84Cycles);

    ADC_Cmd(ADC1, ENABLE);
    ADC_Cmd(ADC2, ENABLE);

    RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOB, ENABLE);
```

Konfiguracja używanych urządzeń takich jak ADC, USART,PINY(A1,C10,C11)

```
    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_0 | GPIO_Pin_1;
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_OUT;
    GPIO_InitStructure.GPIO_OType = GPIO_OType_OD;
    GPIO_InitStructure.GPIO_Speed = GPIO_Speed_100MHz;
    GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_NOPULL;
    GPIO_Init(GPIOB, &GPIO_InitStructure);

    RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOC, ENABLE);
    RCC_APB1PeriphClockCmd(RCC_APB1Periph_USART3, ENABLE);

    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_10 | GPIO_Pin_11;
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_AF;
    GPIO_InitStructure.GPIO_OType = GPIO_OType_PP;
    GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_UP;
    GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
    GPIO_Init(GPIOC, &GPIO_InitStructure);

    GPIO_PinAFConfig(GPIOC, GPIO_PinSource10, GPIO_AF_USART3);
    GPIO_PinAFConfig(GPIOC, GPIO_PinSource11, GPIO_AF_USART3);

    USART_InitTypeDef USART_InitStructure;
    USART_InitStructure.USART_BaudRate = 9600;
    USART_InitStructure.USART_WordLength = USART_WordLength_8b;
    USART_InitStructure.USART_StopBits = USART_StopBits_1;
    USART_InitStructure.USART_Parity = USART_Parity_No;
    USART_InitStructure.USART_HardwareFlowControl = USART_HardwareFlowControl_None;
    USART_InitStructure.USART_Mode = USART_Mode_Rx | USART_Mode_Tx;
    USART_Init(USART3, &USART_InitStructure);

    USART_Cmd(USART3, ENABLE);

    NVIC_InitTypeDef NVIC_InitStructure;
    USART_ITConfig(USART3, USART_IT_RXNE, ENABLE);
    NVIC_InitStructure.NVIC_IRQChannel = USART3_IRQn;
    NVIC_InitStructure.NVIC_IRQChannelPreemptionPriority = 0;
    NVIC_InitStructure.NVIC_IRQChannelSubPriority = 0;
    NVIC_InitStructure.NVIC_IRQChannelCmd = ENABLE;
    NVIC_Init(&NVIC_InitStructure);
    NVIC_EnableIRQ(USART3_IRQn);
```

Konfiguracja używanych urządzeń takich jak ADC, USART,PINY(A1,C10,C11)

```
    while(1)
    {

        ADC_SoftwareStartConv(ADC1);
        while(ADC_GetFlagStatus(ADC1, ADC_FLAG_EOC) == RESET);
        ADC_Result1 = ADC_GetConversionValue(ADC1);
        if(ADC_Result1<Dioda) GPIO_SetBits(GPIOB,GPIO_Pin_1);
            else GPIO_ResetBits(GPIOB,GPIO_Pin_1);

        ADC_SoftwareStartConv(ADC2);
        while(ADC_GetFlagStatus(ADC2, ADC_FLAG_EOC) == RESET);
        ADC_Result2 = ADC_GetConversionValue(ADC2);
        temp=(293*ADC_Result2/4095)-50;
        if(temp<C) GPIO_SetBits(GPIOB,GPIO_Pin_0);
            else GPIO_ResetBits(GPIOB,GPIO_Pin_0);
    }
}
```

Pętla funkcji main w której zmieniane są wartości pinów sterujących w zależności od odczytanych danych

Aplikacja:

Aplikacja opiera się na implementacji znajdującej się :
https://github.com/omaflak/Bluetooth-Terminal

Dokonane zmiany dotyczą głównie wyglądu ,najistotniejszą zmianą w działaniu logiki aplikacji jest zmiana sposobu wysyłania danych :

```
send.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        String msg = temp.getText().toString();
        String msg1=dioda;
        String msg2 ="*";
        temp.setText("");
        b.send(msg1);
        b.send(msg);
        if(msg.length()<2) {
            b.send(msg2);
        }
      //  Display("You: "+msg);
    }
});
```

W przypadku kiedy wysyłana przez nas temperatura jest liczbą składającą się tylko z jednej cyfry wysyłamy dodatkowo znak końca ‘*’ który przez przerwanie na płytce jest ignorowany pozwala to uniknąć błędów aplikacji.