LAMPIRAN

Pengujian Software

```
#include"stm32f4xx.h"

#include"stm32f4xx_rcc.h"

#include"stm32f4xx_gpio.h"

#include"stm32f4xx_adc.h"

#include"lcd_16x2.h"

#include"delay.h"

#include"math.h"

#include"stdio.h"


void stabilisasiclock();

void initadc();

uint16_t adc_Read(uint8_t channel);

void delay (int a)

 {

        volatile int i,j;

                 for (i=0 ; i < a ; i++)

                 {

                 j++;

                 }


       return;

 }


 int main(void) {


       RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOD, ENABLE);

                  GPIO_InitTypeDef  GPIO_InitStructure;

       GPIO_InitStructure.GPIO_Pin                                   =
GPIO_Pin_12|GPIO_Pin_13|GPIO_Pin_14|GPIO_Pin_15;

                  GPIO_InitStructure.GPIO_OType = GPIO_OType_PP;

                  GPIO_InitStructure.GPIO_Mode = GPIO_Mode_OUT;
```

```
                        GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_NOPULL;
                        GPIO_InitStructure.GPIO_Speed = GPIO_Speed_100MHz;
                        GPIO_Init(GPIOD, &GPIO_InitStructure);    /// pind




      uint16_t TMP=0;
    uint16_t RH=0;
 float_t tmp=RH;
 char buf[20];
 char TMP1=0;
 char RH1=1;

     SysTick_Init();
     stabilisasiclock();
     initadc();
     lcd16x2_init();
     lcd16x2_cls();

    lcd16x2_strxy(0,0,"Suhu:");
    lcd16x2_strxy(0,1,"RH:");

while (1){

// memilihADC1 channel (0-7)  port A
if (TMP1==0)  TMP = adc_Read(ADC_Channel_0); //PA0
if (RH1==1)  RH = adc_Read(ADC_Channel_1); //PA1
//if (p==2)  temp = adc_Read(ADC_Channel_2); //PA2
//if (p==3)  temp = adc_Read(ADC_Channel_3); //PA3
//if (p==4)  temp = adc_Read(ADC_Channel_4); //PA4
//if (p==5)  temp = adc_Read(ADC_Channel_5); //PA5
//if (p==6)  temp = adc_Read(ADC_Channel_6); //PA6
//if (p==7)  temp = adc_Read(ADC_Channel_7); //PA7



        //suhu
        lcd16x2_locate(5,0);
        lcd16x2_DisplayNumber(TMP*0.068);
        tmp=TMP*0.068;
        sprintf(buf,"%0.2f", tmp*10);
        //lcd16x2_strxy(11,0,buf);
       // delay_nms(300);

        lcd16x2_locate(3,1);
        lcd16x2_DisplayNumber(RH*0.010);
        RH=RH*0.010;
        sprintf(buf,"%0.2f", RH*10);
       /* RH=(float_t)(RH*2.9)/4095;
        sprintf(buf, "%0.2f", RH);*/
       // delay_nms(300)
          if (tmp<=38)
          { GPIO_SetBits(GPIOD,GPIO_Pin_12);
        GPIO_ResetBits(GPIOD,GPIO_Pin_13);}//|GPIO_Pin_14|GPIO_Pin_15);

        else if (tmp>=40)
        {  GPIO_SetBits(GPIOD,GPIO_Pin_13);
         GPIO_ResetBits(GPIOD,GPIO_Pin_12);}/*|GPIO_Pin_14|GPIO_Pin_15);*/

         /* if (RH>=60)
                   { GPIO_SetBits(GPIOD,GPIO_Pin_14);

GPIO_ResetBits(GPIOD,GPIO_Pin_15);}//|GPIO_Pin_14|GPIO_Pin_15);

                 else if (RH<=50)
```

```
                {  GPIO_SetBits(GPIOD,GPIO_Pin_15);
                 GPIO_ResetBits(GPIOD,GPIO_Pin_14);}*/
    /*   else if (tmp<=30)
               {  GPIO_SetBits(GPIOD,GPIO_Pin_15);}
       else if (tmp>=50)
               {  GPIO_SetBits(GPIOD,GPIO_Pin_14); }*/


}
{

                  GPIO_SetBits(GPIOD, GPIO_Pin_14);
                  delay(17000);
                  GPIO_ResetBits(GPIOD,GPIO_Pin_14);
                  delay(28800000);
                  GPIO_SetBits(GPIOD, GPIO_Pin_14);
                  delay(17000);
                  GPIO_ResetBits(GPIOD,GPIO_Pin_14);
                  delay(28800000);
                  GPIO_SetBits(GPIOD, GPIO_Pin_14);
                  delay(17000);
                  GPIO_ResetBits(GPIOD,GPIO_Pin_14);
                  delay(28800000);

}


}


uint16_t adc_Read(uint8_t channel)
{
      ADC_RegularChannelConfig(ADC1, channel, 1, ADC_SampleTime_480Cycles);
      ADC_SoftwareStartConv(ADC1);
      while (ADC_GetFlagStatus(ADC1, ADC_FLAG_EOC) == RESET);
      return ADC_GetConversionValue(ADC1);
}


void initadc(){
      GPIO_InitTypeDef GPIO_InitStructure;
      ADC_InitTypeDef ADC_InitStructure;
      ADC_CommonInitTypeDef ADC_CommonInitStructure;
```

```
	RCC_APB2PeriphClockCmd(RCC_APB2Periph_ADC1, ENABLE);

	RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOA, ENABLE);


	GPIO_InitStructure.GPIO_Pin = GPIO_Pin_0 | GPIO_Pin_1 | GPIO_Pin_2 |
GPIO_Pin_3 | GPIO_Pin_4 | GPIO_Pin_5 | GPIO_Pin_6 |GPIO_Pin_7 ;

	GPIO_InitStructure.GPIO_Mode = GPIO_Mode_AN;

	GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_NOPULL;

	GPIO_Init(GPIOA, &GPIO_InitStructure);

	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_InitStructure.ADC_Resolution = ADC_Resolution_12b;

	ADC_InitStructure.ADC_ScanConvMode = DISABLE;

	ADC_InitStructure.ADC_ContinuousConvMode = DISABLE;

	ADC_InitStructure.ADC_ExternalTrigConvEdge                         =
ADC_ExternalTrigConvEdge_None;

	ADC_InitStructure.ADC_ExternalTrigConv = DISABLE;

	ADC_InitStructure.ADC_DataAlign = ADC_DataAlign_Right;

	ADC_InitStructure.ADC_NbrOfConversion = 1;

	ADC_Init(ADC1, &ADC_InitStructure);

	ADC_Cmd(ADC1, ENABLE);

}

void stabilisasiclock(){

	SystemInit();

	RCC_HSEConfig(RCC_HSE_ON);

	while (!RCC_WaitForHSEStartUp());

}

void SysTick_Handler(void) {

    TimeTick_Decrement();
```

Pengujian Mikrokontroler ARM STM32

```
#include"stm32f4xx.h"
#include"stm32f4xx_rcc.h"
#include"stm32f4xx_gpio.h"
#include"lcd_16x2.h"
#include"delay.h"
#include"math.h"
#include"stdio.h"

intmain(void)
{
   SysTick_Init();
         SystemInit();
      lcd16x2_init();
while (1)
   {
         lcd16x2_cls();
         lcd16x2_strxy(0,0,"Suhu");
         lcd16x2_strxy(5,1,"RH");
         delay_nms(10000);
         lcd16x2_cls();

   }
}

voidSysTick_Handler(void)
{
   TimeTick_Decrement();
}
```