1. PRIEDAS

Gamybos procesų eiga bei įranga naudota Twente universitete Olandijoje CMUT struktūrų gamybai, receptų kodai.

100mm Silicio P/B <100> plokštelė su 300nm Terminiu SiO_2 Standartinis valymas (clean1001)

1

Padengimas rezistu Ti35 ES (reversinis procesas) (lith140)

2

Eksponavimas ir ryškinimas (lith140) Fotošablonas #1

3

Ėsdinimas SiO_2 (BOE/BHF) (etch1020)

4

Chromo garinimas 100nm (film128)

5

Atkėlimas (lith144)

6

1st Si_xN_y sluoksnis 250nm, PECVD Oxford 80, vidiniai įtempiai 40-60 MPa (film133)

7

ž ~20 min
Temperatūra ~300˚STS PECVD receptas:
HF 13.56MHz ~5min
MF 13.56MHz÷380kHC

Padengimas Oir 907-17 pozityviniu rezistu (lith1002)

8

Eksponavimas ir ryškinimas (lith1002) Fotošablonas #2

9

Įsdinimas SiO_2 ir Si_xN_y su RIE, Si_xN_y su CHF3-02 (Tetske) (etch193)

10

Rezisto šalinimas (lith1142) (su metalais)

11

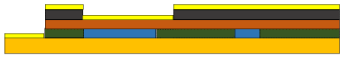
Dengimas Ti35 ES (reversinis procesas) (lith140)

12

Eksponavimas ir ryškinimas (lith140) Fotošablonas #3

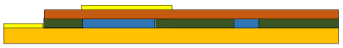
13

Elektrodų formavimas Al/Ti/Al 25/150/25nm garinimu BAK 600 (film127)



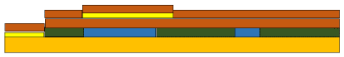
14

Atkėlimas (lith144)



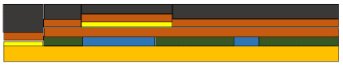
15

2nd Si_xN_y sloksnio formavimas 100nm, PECVD Oxford 80, (film133)



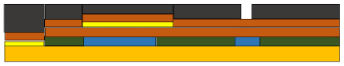
16

Dengimas Oir 907-17 pozityviniu rezistu (lith1002)



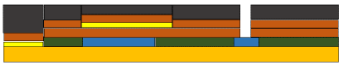
17

Eksponavimas ir ryškinimas (lith1002) Fotošablonas #4 IW



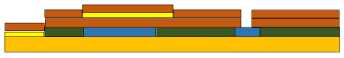
18

Ėsdinimas Si_xN_y su RIE, SiO_2, Si_xN_y sluoksnių su CHF3-02 (Tetske) (etch193)



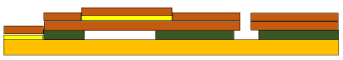
19

Rezisto pašalinimas (lith1142)



20

CR-7 aukojamojo sluoksnio atlaisvinimas ir kritinio taško džiovinimas (etch1023) (clean126)

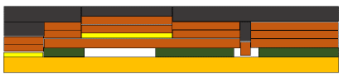


21

3rd Si_xN_y sluoksnio padengimas 300nm, PECVD Oxford 80, (film133)

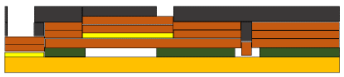
22

Dengimas Oir 907-17 pozityviniu rezistu (lith1002)



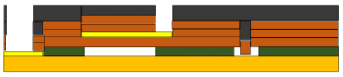
23

Eksponavimas ir ryškinimas (lith1002) Fotošablonas #5



24

Ėsdinimas Si_xN_y su RIE SiO_2, Si_xN_y sluoksniu naudojant CHF3-02 (Tetske) (etch193)

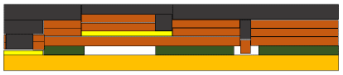


25

Rezisto panaikinimas (su metalais) (lith1142)

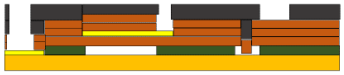
26

Dengimas Ti35 ES (reversinis procesas) (lith140)



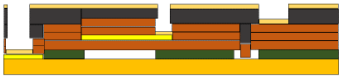
27

Eksponavimas ir ryškinimas (lith140) Fotošablonas #6



28

Aukso garinimas 100-120nm BAK 600 (film144)

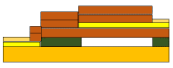


29

Atkėlimas (lith144)

30

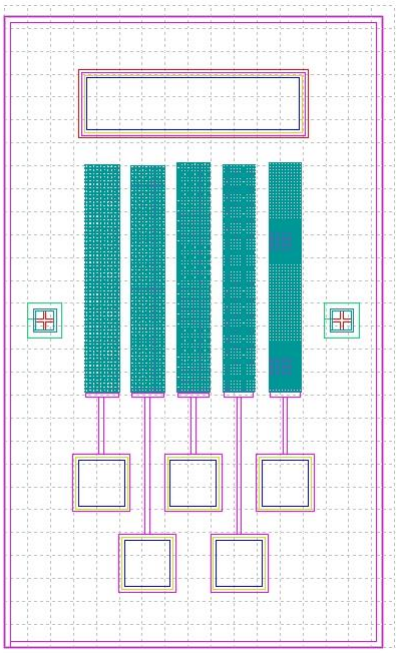
Pjaustymas (back101)



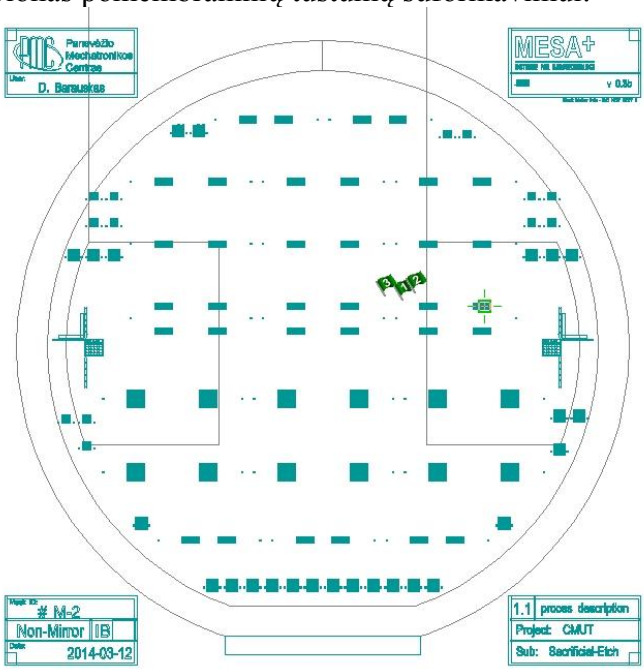
31

2. PRIEDAS

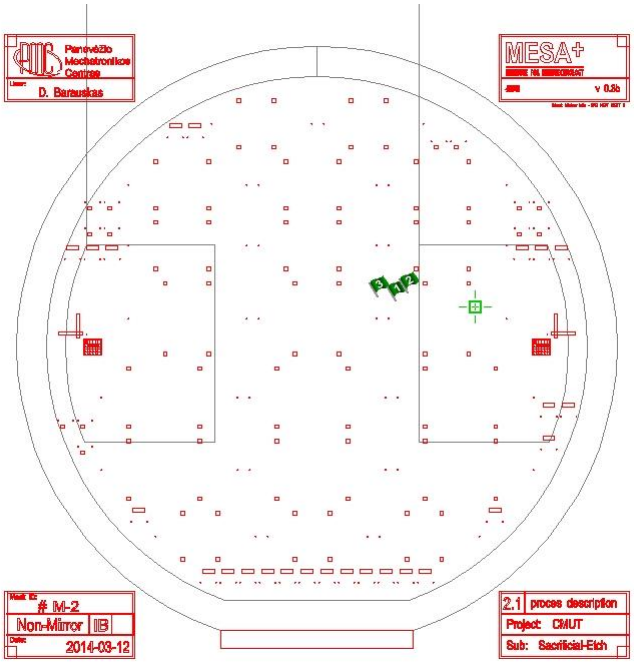
Fotošablonų komplektas. Slėgio jutiklis (sudėti visi sluoksniai, sluoksnių spalvos pagal šablono spalvą):



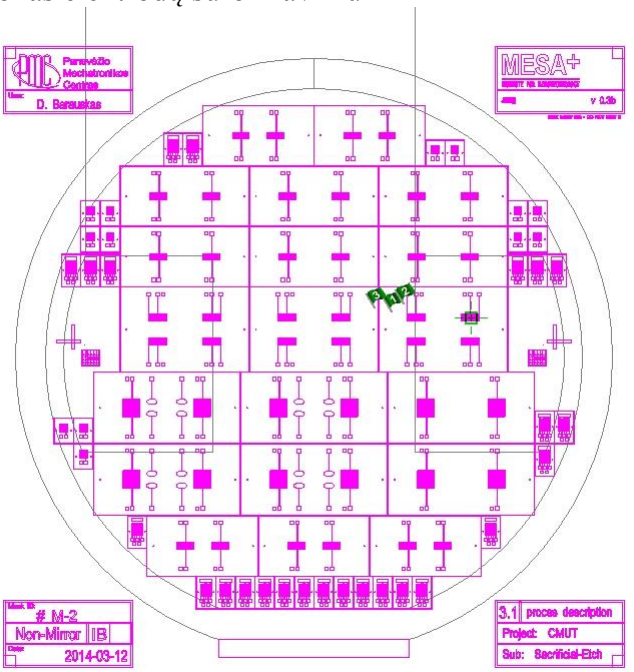
1. Šablonas pomembraninių tuštumų suformavimui:



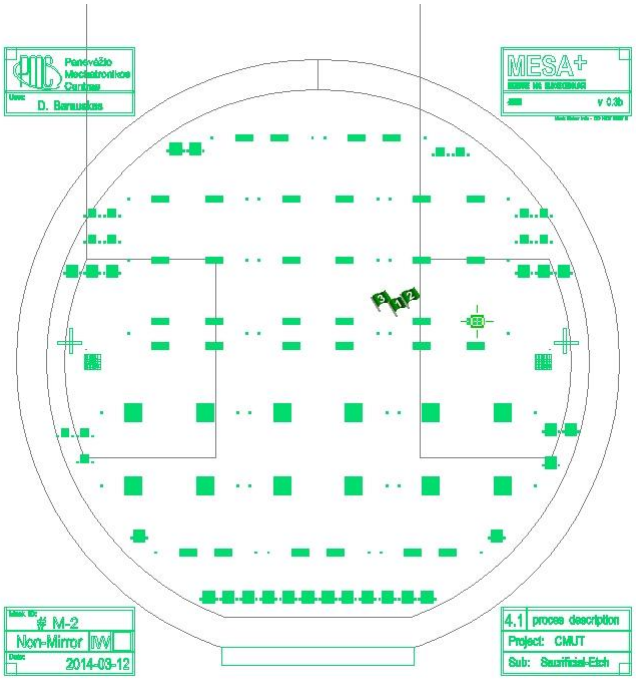
2. Šablonas žemės elektrodo atvėrimui



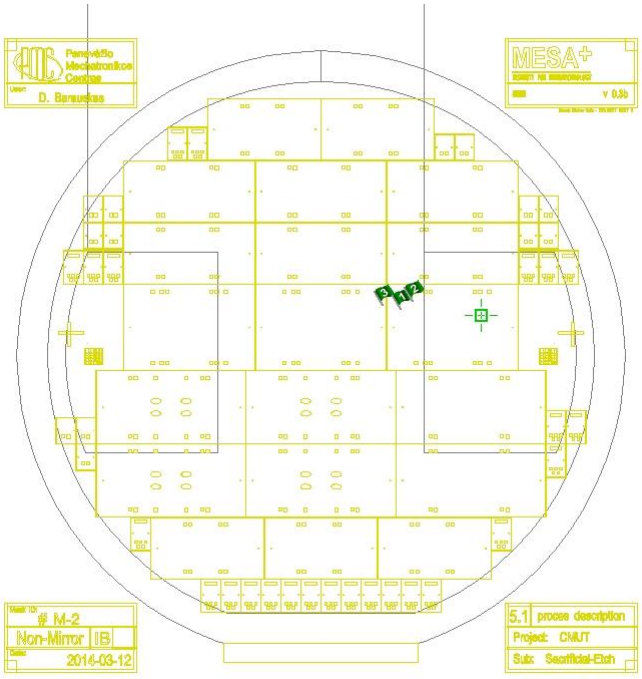
3. Šablonas elektrodų suformavimui



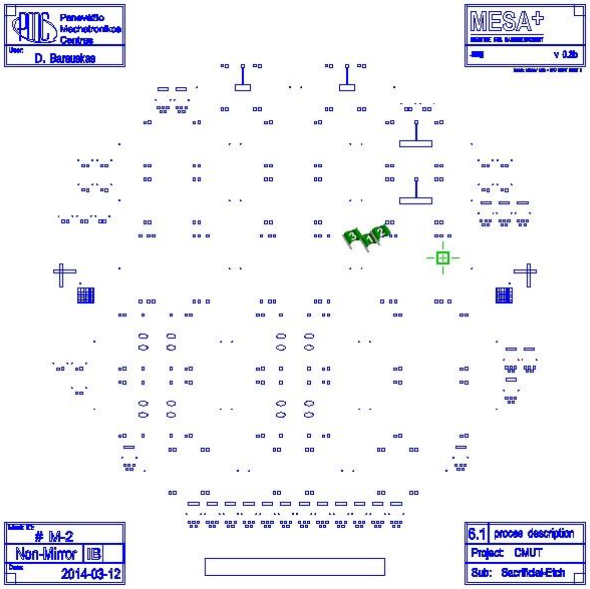
4. Šablonas ėsdinimo skylių suformavimui



5. Šablonas elektrodų atvėrimui



6. Šablonas elektrodų metalizavimui



3. PRIEDAS

Realaus laiko matavimo duomenų programos kodas MatLab aplinkoje

```
if exist('g2','var') == 0
    clear
    g2=0;
end
if exist('g3','var') == 0
    g3=0;
end
if g3==1
    clear
    g2=1;
    i=1;
    g3=0;
    pradinis_laikas=clock;
pradinis_laikas_sec=pradinis_laikas(6)+pradinis_laikas(5)*60+pradinis_laikas(
4)*3600;
end
if g2==0
    figure('units','normalized','outerposition',[0 0 1 1]);
    i=1;

end
btn2 =
uicontrol('style','push','string','Tęsti','callback','g2=1;,Vieno_Daznomacio_
nuskaitymas;','Position', [10 600 100 30]);
btn3 = uicontrol('style','push','string','Pradėti iš
naujo','callback','g3=1;,Vieno_Daznomacio_nuskaitymas;','Position', [10 570
100 30]);
btn113 = uicontrol('style','push','string','reset','callback','q=duomenys(i-
1,2)','Position', [10 370 100 30]);
btn11 = uicontrol('style','push','Position', [10 340 100 30]);
g = 0;
uzdelsimas=0.1;%nuskaitymo intervalas
btn =
uicontrol('style','push','string','Stabdyti','callback','g=1;','Position',
[10 650 100 30]);
   sld = uicontrol('Style', 'slider',...
        'Min',1,'Max',1e7,'Value',10,...
        'SliderStep', [0.000001 0.000010], ...
        'Position', [1270 80 20 550]) ;
xlabel('Laikas');
ylabel('Dažnis, Hz');
if g2==0
pradinis_laikas=clock;
pradinis_laikas_sec=pradinis_laikas(6)+pradinis_laikas(5)*60+pradinis_laikas(
4)*3600;
pause(0.001);
end
q=3075811.6;
while g==0
dabar=clock;
praejo=dabar(6)+dabar(5)*60+dabar(4)*3600;
laikas_praejes=praejo-pradinis_laikas_sec;
laikas(i,1)=laikas_praejes;
duomenys(i,1:2)  = kodas_out(labwM('daznis',0));
pause(uzdelsimas);
asisy=get(sld,'value');
asis=duomenys(i,1);
[ax]=plot(laikas(:,1),duomenys(:,1));
axis([0 laikas(i,1) asis-asisy asis+asisy]);
asis=duomenys(i,2);
```

```
xlabel('Laikas, s');
ylabel('Apatinio dažnomačio parodymai, Hz');
title(['Dažnomačio parodymai']);
virsutinis=num2str(duomenys(i,1)*10);
apatinis=num2str(duomenys(i,2)*10);
set(btn11,'String', [virsutinis ' Hz'])
x=duomenys(i,2);
i = i+1;
 slegis = ((-q+x)/(-2376.8)+1)*101325;
 lol=num2str(slegis);
duomenys2 = 'duomenys';
duomenys3 = 'skirtumas';
duomenys4 = 'absskirtumas';
duomenys5 = 'vidurkis';
duomenys6 = 'laikas';
set(btn113,'String', [lol ' Pa'])
pause(1);
end
btn5 = uicontrol('style','edit','Position', [10 475 100 20]);
g4 = get(btn5, 'String');
vardas=[g4 ' ' datestr(now, 'yyyy-mmm-dd HH-MM-SS')];
btn4 =
uicontrol('style','push','string','Išsaugoti','callback','saugojimas','Positi
on', [10 440 100 30]);
btn6 = uicontrol('style','text','string','Failo
Pavadinimas','callback','','Position', [15 500 90 15]);
```

4. PRIEDAS

STM32 mikrovaldiklio programos kodas

```
#define HSE_VALUE ((uint32_t)8000000) /* STM32 discovery uses a 8Mhz external
crystal */
#include "stm32f4xx_conf.h"
#include "stm32f4xx.h"
#include "stm32f4xx_gpio.h"
#include "stm32f4xx_rcc.h"
#include "stm32f4xx_exti.h"
#include "usbd_cdc_core.h"
#include "usbd_usr.h"
#include "usbd_desc.h"
#include "usbd_cdc_vcp.h"
#include "usb_dcd_int.h"
#include "stm32f4xx_usart.h"
#include "stdio.h"
#include "stdlib.h"
#include "string.h"
volatile uint32_t ticker, downTicker;
__ALIGN_BEGIN USB_OTG_CORE_HANDLE  USB_OTG_dev __ALIGN_END;
void init();
void ColorfulRingOfDeath(void);
#ifdef __cplusplus
 extern "C" {
#endif
void SysTick_Handler(void);
void NMI_Handler(void);
void HardFault_Handler(void);
void MemManage_Handler(void);
void BusFault_Handler(void);
void UsageFault_Handler(void);
void SVC_Handler(void);
void DebugMon_Handler(void);
void PendSV_Handler(void);
void OTG_FS_IRQHandler(void);
void OTG_FS_WKUP_IRQHandler(void);
#ifdef __cplusplus
}
#endif
void init()
{
          GPIO_InitTypeDef LED_Config;
RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOA, ENABLE);
          LED_Config.GPIO_Pin =  GPIO_Pin_1 | GPIO_Pin_2 | GPIO_Pin_3 |
GPIO_Pin_4 | GPIO_Pin_5 | GPIO_Pin_6 | GPIO_Pin_7  | GPIO_Pin_10 |
GPIO_Pin_13 | GPIO_Pin_14 | GPIO_Pin_15 ;
          LED_Config.GPIO_Mode = GPIO_Mode_IN;
          LED_Config.GPIO_OType = GPIO_OType_PP;
          LED_Config.GPIO_Speed = GPIO_Speed_100MHz;
          LED_Config.GPIO_PuPd = GPIO_PuPd_NOPULL;
          GPIO_Init(GPIOA, &LED_Config);
          RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOB, ENABLE);
          LED_Config.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_Pin_8
| GPIO_Pin_9 | GPIO_Pin_10 | GPIO_Pin_11 | GPIO_Pin_12 | GPIO_Pin_13 |
GPIO_Pin_14 | GPIO_Pin_15  ;
          LED_Config.GPIO_Mode = GPIO_Mode_IN;
          LED_Config.GPIO_OType = GPIO_OType_PP;
          LED_Config.GPIO_Speed = GPIO_Speed_100MHz;
          LED_Config.GPIO_PuPd = GPIO_PuPd_NOPULL;
          GPIO_Init(GPIOB, &LED_Config);
          RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOC, ENABLE);
```

```
            LED_Config.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_Pin_8 |
GPIO_Pin_9 | GPIO_Pin_10 | GPIO_Pin_11 | GPIO_Pin_12 | GPIO_Pin_13 ;
            LED_Config.GPIO_Mode = GPIO_Mode_IN;
            LED_Config.GPIO_OType = GPIO_OType_PP;
            LED_Config.GPIO_Speed = GPIO_Speed_100MHz;
            LED_Config.GPIO_PuPd = GPIO_PuPd_NOPULL;
            GPIO_Init(GPIOC, &LED_Config);
            RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOD, ENABLE);
            LED_Config.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_Pin_8 |
GPIO_Pin_9 | GPIO_Pin_10 | GPIO_Pin_11;
            LED_Config.GPIO_Mode = GPIO_Mode_IN;
            LED_Config.GPIO_OType = GPIO_OType_PP;
            LED_Config.GPIO_Speed = GPIO_Speed_100MHz;
            LED_Config.GPIO_PuPd = GPIO_PuPd_NOPULL;
            GPIO_Init(GPIOD, &LED_Config);
            RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOE, ENABLE);
            LED_Config.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_Pin_8
| GPIO_Pin_9 | GPIO_Pin_10 | GPIO_Pin_11 | GPIO_Pin_12 | GPIO_Pin_13 |
GPIO_Pin_14 | GPIO_Pin_15;
            LED_Config.GPIO_Mode = GPIO_Mode_IN;
            LED_Config.GPIO_OType = GPIO_OType_PP;
            LED_Config.GPIO_Speed = GPIO_Speed_100MHz;
            LED_Config.GPIO_PuPd = GPIO_PuPd_NOPULL;
            GPIO_Init(GPIOE, &LED_Config);
            if (SysTick_Config(SystemCoreClock / 1000))
            {
                        ColorfulRingOfDeath();
            }
            USBD_Init(&USB_OTG_dev,
                        USB_OTG_FS_CORE_ID,
                        &USR_desc,
                        &USBD_CDC_cb,
                        &USR_cb);
            return;
}
void ColorfulRingOfDeath(void)
{
            uint16_t ring = 1;
            while (1)
            {
                        uint32_t count = 0;
                        while (count++ < 500000);

                        GPIOD->BSRRH = (ring << 12);
                        ring = ring << 1;
                        if (ring >= 1<<4)
                        {
                                    ring = 1;
                        }
                        GPIOD->BSRRL = (ring << 12);
            }
}
int pirm_nusk()
{
            int sk1 , sk2 , sk3 , sk4, suma ;
            sk1 = 0;
            sk2 = 0;
            sk3  = 0;
            sk4 = 0;
            suma = 0;
            sk1 = GPIO_ReadInputDataBit(GPIOC, GPIO_Pin_9);
```

```
            sk2 = GPIO_ReadInputDataBit(GPIOC, GPIO_Pin_8);
            sk3 = GPIO_ReadInputDataBit(GPIOA, GPIO_Pin_13);
            sk4 = GPIO_ReadInputDataBit(GPIOA, GPIO_Pin_10);

            suma = sk1*1+sk2*2+sk3*4+sk4*8;
            return suma ;
}
int pirm_nusk2()
{
            int sk1 , sk2 , sk3 , sk4, suma ;
            sk1 = 0;
            sk2 = 0;
            sk3  = 0;
            sk4 = 0;
            suma = 0;
            sk1 = GPIO_ReadInputDataBit(GPIOC, GPIO_Pin_1);
            sk2 = GPIO_ReadInputDataBit(GPIOC, GPIO_Pin_3);
            sk3 = GPIO_ReadInputDataBit(GPIOC, GPIO_Pin_2);
            sk4 = GPIO_ReadInputDataBit(GPIOA, GPIO_Pin_1);

            suma = sk1*1+sk2*2+sk3*4+sk4*8;
            return suma ;
}
int antr_nusk()
{
            int sk1 , sk2 , sk3 , sk4, suma ;
            sk1 = 0;
            sk2 = 0;
            sk3  = 0;
            sk4 = 0;
            suma = 0;
            sk1 = GPIO_ReadInputDataBit(GPIOA, GPIO_Pin_15);
            sk2 = GPIO_ReadInputDataBit(GPIOA, GPIO_Pin_14);
            sk3 = GPIO_ReadInputDataBit(GPIOC, GPIO_Pin_11);
            sk4 = GPIO_ReadInputDataBit(GPIOC, GPIO_Pin_10);

            suma = sk1*1+sk2*2+sk3*4+sk4*8;
            return suma ;
}
int antr_nusk2()
{
            int sk1 , sk2 , sk3 , sk4, suma ;
            sk1 = 0;
            sk2 = 0;
            sk3  = 0;
            sk4 = 0;
            suma = 0;
            sk1 = GPIO_ReadInputDataBit(GPIOA, GPIO_Pin_3);
            sk2 = GPIO_ReadInputDataBit(GPIOA, GPIO_Pin_2);
            sk3 = GPIO_ReadInputDataBit(GPIOA, GPIO_Pin_5);
            sk4 = GPIO_ReadInputDataBit(GPIOA, GPIO_Pin_4);
            suma = sk1*1+sk2*2+sk3*4+sk4*8;
            return suma ;
}
int trec_nusk()
{
            int sk1 , sk2 , sk3 , sk4, suma ;
            sk1 = 0;
            sk2 = 0;
            sk3  = 0;
            sk4 = 0;
            suma = 0;
            sk1 = GPIO_ReadInputDataBit(GPIOD, GPIO_Pin_0);
            sk2 = GPIO_ReadInputDataBit(GPIOC, GPIO_Pin_12);
```

```
            sk3 = GPIO_ReadInputDataBit(GPIOD, GPIO_Pin_2);
            sk4 = GPIO_ReadInputDataBit(GPIOD, GPIO_Pin_1);
            suma = sk1*1+sk2*2+sk3*4+sk4*8;
            return suma ;
}
int trec_nusk2()
{
            int sk1 , sk2 , sk3 , sk4, suma ;
            sk1 = 0;
            sk2 = 0;
            sk3  = 0;
            sk4 = 0;
            suma = 0;
            sk1 = GPIO_ReadInputDataBit(GPIOA, GPIO_Pin_7);
            sk2 = GPIO_ReadInputDataBit(GPIOC, GPIO_Pin_5);
            sk3 = GPIO_ReadInputDataBit(GPIOC, GPIO_Pin_4);
            sk4 = GPIO_ReadInputDataBit(GPIOB, GPIO_Pin_1);
            suma = sk1*1+sk2*2+sk3*4+sk4*8;
            return suma ;
}
int ketv_nusk()
{
            int sk1 , sk2 , sk3 , sk4, suma ;
            sk1 = 0;
            sk2 = 0;
            sk3  = 0;
            sk4 = 0;
            suma = 0;
            sk1 = GPIO_ReadInputDataBit(GPIOD, GPIO_Pin_4);
            sk2 = GPIO_ReadInputDataBit(GPIOD, GPIO_Pin_3);
            sk3 = GPIO_ReadInputDataBit(GPIOD, GPIO_Pin_6);
            sk4 = GPIO_ReadInputDataBit(GPIOD, GPIO_Pin_5);

            suma = sk1*1+sk2*2+sk3*4+sk4*8;
            return suma ;
}
int ketv_nusk2()
{
            int sk1 , sk2 , sk3 , sk4, suma ;
            sk1 = 0;
            sk2 = 0;
            sk3  = 0;
            sk4 = 0;
            suma = 0;
            sk1 = GPIO_ReadInputDataBit(GPIOB, GPIO_Pin_0);
            sk2 = GPIO_ReadInputDataBit(GPIOB, GPIO_Pin_2);
            sk3 = GPIO_ReadInputDataBit(GPIOE, GPIO_Pin_7);
            sk4 = GPIO_ReadInputDataBit(GPIOE, GPIO_Pin_8);
            suma = sk1*1+sk2*2+sk3*4+sk4*8;
            return suma ;
}
int penkt_nusk()
{
            int sk1 , sk2 , sk3 , sk4, suma ;
            sk1 = 0;
            sk2 = 0;
            sk3  = 0;
            sk4 = 0;
            suma = 0;
            sk1 = GPIO_ReadInputDataBit(GPIOB, GPIO_Pin_3);
            sk2 = GPIO_ReadInputDataBit(GPIOD, GPIO_Pin_7);
            sk3 = GPIO_ReadInputDataBit(GPIOB, GPIO_Pin_5);
            sk4 = GPIO_ReadInputDataBit(GPIOB, GPIO_Pin_4);
            suma = sk1*1+sk2*2+sk3*4+sk4*8;
```

```
            return suma ;
}
int penkt_nusk2()
{
            int sk1 , sk2 , sk3 , sk4, suma ;
            sk1 = 0;
            sk2 = 0;
            sk3  = 0;
            sk4 = 0;
            suma = 0;
            sk1 = GPIO_ReadInputDataBit(GPIOE, GPIO_Pin_9);
            sk2 = GPIO_ReadInputDataBit(GPIOE, GPIO_Pin_10);
            sk3 = GPIO_ReadInputDataBit(GPIOE, GPIO_Pin_11);
            sk4 = GPIO_ReadInputDataBit(GPIOE, GPIO_Pin_12);

            suma = sk1*1+sk2*2+sk3*4+sk4*8;
            return suma ;
}

int sest_nusk()
{
            int sk1 , sk2 , sk3 , sk4, suma ;
            sk1 = 0;
            sk2 = 0;
            sk3  = 0;
            sk4 = 0;
            suma = 0;
            sk1 = GPIO_ReadInputDataBit(GPIOB, GPIO_Pin_7);
            sk2 = GPIO_ReadInputDataBit(GPIOB, GPIO_Pin_6);
            sk3 = GPIO_ReadInputDataBit(GPIOB, GPIO_Pin_9);
            sk4 = GPIO_ReadInputDataBit(GPIOB, GPIO_Pin_8);

            suma = sk1*1+sk2*2+sk3*4+sk4*8;
            return suma ;
}

int sest_nusk2()
{
            int sk1 , sk2 , sk3 , sk4, suma ;
            sk1 = 0;
            sk2 = 0;
            sk3  = 0;
            sk4 = 0;
            suma = 0;
            sk1 = GPIO_ReadInputDataBit(GPIOE, GPIO_Pin_13);
            sk2 = GPIO_ReadInputDataBit(GPIOE, GPIO_Pin_14);
            sk3 = GPIO_ReadInputDataBit(GPIOE, GPIO_Pin_15);
            sk4 = GPIO_ReadInputDataBit(GPIOB, GPIO_Pin_10);

            suma = sk1*1+sk2*2+sk3*4+sk4*8;
            return suma ;
}

int sept_nusk()
{
            int sk1 , sk2 , sk3 , sk4, suma ;
            sk1 = 0;
            sk2 = 0;
            sk3  = 0;
            sk4 = 0;
            suma = 0;
            sk1 = GPIO_ReadInputDataBit(GPIOE, GPIO_Pin_1);
            sk2 = GPIO_ReadInputDataBit(GPIOE, GPIO_Pin_0);
            sk3 = GPIO_ReadInputDataBit(GPIOE, GPIO_Pin_3);
```

```
	sk4 = GPIO_ReadInputDataBit(GPIOE, GPIO_Pin_2);

	suma = sk1*1+sk2*2+sk3*4+sk4*8;
	return suma ;
}

int sept_nusk2()
{
	int sk1 , sk2 , sk3 , sk4, suma ;
	sk1 = 0;
	sk2 = 0;
	sk3  = 0;
	sk4 = 0;
	suma = 0;
	sk1 = GPIO_ReadInputDataBit(GPIOB, GPIO_Pin_11);
	sk2 = GPIO_ReadInputDataBit(GPIOB, GPIO_Pin_12);
	sk3 = GPIO_ReadInputDataBit(GPIOB, GPIO_Pin_13);
	sk4 = GPIO_ReadInputDataBit(GPIOB, GPIO_Pin_14);

	suma = sk1*1+sk2*2+sk3*4+sk4*8;
	return suma ;
}

int ast_nusk()
{
	int sk1 , sk2 , sk3 , sk4, suma ;
	sk1 = 0;
	sk2 = 0;
	sk3  = 0;
	sk4 = 0;
	suma = 0;
	sk1 = GPIO_ReadInputDataBit(GPIOE, GPIO_Pin_5);
	sk2 = GPIO_ReadInputDataBit(GPIOE, GPIO_Pin_4);
	sk3 = GPIO_ReadInputDataBit(GPIOC, GPIO_Pin_13);
	sk4 = GPIO_ReadInputDataBit(GPIOE, GPIO_Pin_6);

	suma = sk1*1+sk2*2+sk3*4+sk4*8;
	return suma ;
}

int ast_nusk2()
{
	int sk1 , sk2 , sk3 , sk4, suma ;
	sk1 = 0;
	sk2 = 0;
	sk3  = 0;
	sk4 = 0;
	suma = 0;
	sk1 = GPIO_ReadInputDataBit(GPIOB, GPIO_Pin_15);
	sk2 = GPIO_ReadInputDataBit(GPIOD, GPIO_Pin_8);
	sk3 = GPIO_ReadInputDataBit(GPIOD, GPIO_Pin_9);
	sk4 = GPIO_ReadInputDataBit(GPIOD, GPIO_Pin_10);

	suma = sk1*1+sk2*2+sk3*4+sk4*8;
	return suma ;
}

void SysTick_Handler(void)
{
	ticker++;
	if (downTicker > 0)
	{
		downTicker--;
	}
```

```
}

void NMI_Handler(void)        {}
void MemManage_Handler(void) { ColorfulRingOfDeath(); }
void BusFault_Handler(void)   { ColorfulRingOfDeath(); }
void UsageFault_Handler(void){ ColorfulRingOfDeath(); }
void SVC_Handler(void)        {}
void DebugMon_Handler(void)   {}
void PendSV_Handler(void)     {}
void OTG_FS_IRQHandler(void)
{
  USBD_OTG_ISR_Handler (&USB_OTG_dev);
}
void OTG_FS_WKUP_IRQHandler(void)
{
  if(USB_OTG_dev.cfg.low_power)
  {
    *(uint32_t *)(0xE000ED10) &= 0xFFFFFFF9 ;
    SystemInit();
    USB_OTG_UngateClock(&USB_OTG_dev);
  }
  EXTI_ClearITPendingBit(EXTI_Line18);
}
int main(void)
{
           init();
printf("sudas \n\r");
           while (1)
           {
                      if (500 == ticker)
                      {
                                 GPIOD->BSRRH = GPIO_Pin_13;
                      }
                      else if (1000 == ticker)
                      {
                                 ticker = 0;
                                 GPIOD->BSRRL = GPIO_Pin_13;
                      }
                      uint8_t theByte;
                      uint8_t buferis[6];
                      uint8_t len;
                      int aInt1 = pirm_nusk();
                      int aInt2 = antr_nusk();
                      int aInt3 = trec_nusk();
                      int aInt4 = ketv_nusk();
                      int aInt5 = penkt_nusk();
                      int aInt6 = sest_nusk();
                      int aInt7 = sept_nusk();
                      int aInt8 = ast_nusk();
                      int bInt1 = pirm_nusk2();
                      int bInt2 = antr_nusk2();
                      int bInt3 = trec_nusk2();
                      int bInt4 = ketv_nusk2();
                      int bInt5 = penkt_nusk2();
                      int bInt6 = sest_nusk2();
                      int bInt7 = sept_nusk2();
                      int bInt8 = ast_nusk2();

                      int aInt = aInt1*1 + aInt2*10 + aInt3*100 + aInt4 * 1000
+ aInt5 *10000 + aInt6 * 100000 + aInt7 * 1000000 + aInt8 *10000000;
                      int bInt = bInt1*1 + bInt2*10 + bInt3*100 + bInt4 * 1000
+ bInt5 *10000 + bInt6 * 100000 + bInt7 * 1000000 + bInt8 *10000000;
                      char str[20];
                      char str2[20];
```

```
                        sprintf(str, "%d", aInt);
                        sprintf(str2, "%d", bInt);
                        strcat(str," ");
                        strcat(str,str2);
                        len = VCP_get_string(&buferis);
                        if(len)
                        {
                                VCP_send_str(strcat(str, "\r\n"));
                        }
                }
                return 0;
}
```