```
/*
  トラ技ジュニア No.41　記事「RISC-V 採用 32bit マイコン・ボード Sipeed Longan Nano」
  制御プログラム
  2020/3/17
  Copyright (C) 2020 Eri Ikegami

  筆者のご厚意により，記事関連データを公開します.
  収録したプログラムの著作権は，著作権者(筆者)にあります.
  ※プログラムの使用により，使用者に損失が生じたとしても，著作権者とＣＱ出版(株)は，その責任を負いません.
  ※プログラムにバグや欠陥があったとしても，著作権者とＣＱ出版(株)は，修正や改良の義務を負いません.
*/

#include "lcd/lcd.h"
#include "fatfs/tf_card.h"
#include <string.h>

unsigned char image[12800];
FATFS fs;

void init_uart0(void)
{
        /* enable GPIO clock */
    rcu_periph_clock_enable(RCU_GPIOA);
    /* enable USART clock */
    rcu_periph_clock_enable(RCU_USART0);

    /* connect port to USARTx_Tx */
    gpio_init(GPIOA, GPIO_MODE_AF_PP, GPIO_OSPEED_50MHZ, GPIO_PIN_9);
    /* connect port to USARTx_Rx */
    gpio_init(GPIOA, GPIO_MODE_IN_FLOATING, GPIO_OSPEED_50MHZ, GPIO_PIN_10);

        /* USART configure */
    usart_deinit(USART0);
    usart_baudrate_set(USART0, 115200U);
    usart_word_length_set(USART0, USART_WL_8BIT);
    usart_stop_bit_set(USART0, USART_STB_1BIT);
    usart_parity_config(USART0, USART_PM_NONE);
    usart_hardware_flow_rts_config(USART0, USART_RTS_DISABLE);
    usart_hardware_flow_cts_config(USART0, USART_CTS_DISABLE);
    usart_receive_config(USART0, USART_RECEIVE_ENABLE);
    usart_transmit_config(USART0, USART_TRANSMIT_ENABLE);
    usart_enable(USART0);
```

```
    usart_interrupt_enable(USART0, USART_INT_RBNE);
}

int main(void)
{
    uint8_t mount_is_ok = 1; /* 0: mount successful ; 1: mount failed */
    int offset = 0;
    FIL fil;
    FRESULT fr;     /* FatFs return code */
    UINT br;

    rcu_periph_clock_enable(RCU_GPIOA);
    rcu_periph_clock_enable(RCU_GPIOB);
    rcu_periph_clock_enable(RCU_GPIOC);
    gpio_init(GPIOC, GPIO_MODE_OUT_PP, GPIO_OSPEED_50MHZ, GPIO_PIN_13);
    gpio_init(GPIOA, GPIO_MODE_OUT_PP, GPIO_OSPEED_50MHZ, GPIO_PIN_1|GPIO_PIN_2);

    /* IR receiver 1 */
    gpio_init(GPIOB, GPIO_MODE_IPU, GPIO_OSPEED_50MHZ, GPIO_PIN_9);
    /* IR receiver 2 */
    gpio_init(GPIOB, GPIO_MODE_IPU, GPIO_OSPEED_50MHZ, GPIO_PIN_8);

    init_uart0();

    Lcd_Init();
    LCD_Clear(WHITE);
    BACK_COLOR=WHITE;

    LEDR(1);
    LEDG(1);
    LEDB(1);

    fr = f_mount(&fs, "", 1);
    if (fr == 0)
        mount_is_ok = 0;
    else
        mount_is_ok = 1;

    if (mount_is_ok == 0)
    {
        FlagStatus bit_state0,bit_state1;
```

```
int state = 9;
int none_count = 0;
int touch_count = 0;
int flag = 0;
int omeme_time = 100;

fr = f_open(&fil, "sample.bin", FA_READ);
if (fr) printf("open error: %d!¥n¥r", (int)fr);
offset = 0;
fr = f_read(&fil, image, sizeof(image), &br);
LCD_ShowPicture(0,0,159,39);
offset += 12800;
f_lseek(&fil, offset);
LEDB_TOG;
fr = f_read(&fil, image, sizeof(image), &br);
LCD_ShowPicture(0,40,159,79);
offset += 12800;
f_lseek(&fil, offset);
LEDB_TOG;

f_close(&fil);
delay_1ms(1000);

 while(1){
     bit_state0 = gpio_input_bit_get(GPIOB, GPIO_PIN_8);
     bit_state1 = gpio_input_bit_get(GPIOB, GPIO_PIN_9);
     omeme_time = 100;
     state = 100;

     if(bit_state0 == 0 && bit_state1 == 0){
         state = 0;
     }
     if(bit_state0 == 0 && bit_state1 == 1){
         state = 1;
         omeme_time = 500;
         if(flag==0 | flag==2){
             touch_count += 1;
             flag = 1;
             if(touch_count > 4){
                 state = 3;
                 touch_count = 0;
```

```
            omeme_time = 1000;
        }
    }
}
if(bit_state0 == 1 && bit_state1 == 0){
    state = 2;
    omeme_time = 500;
    if(flag==0 | flag==1){
        touch_count += 1;
        flag = 2;
        if(touch_count > 4){
            state = 3;
            touch_count = 0;
            omeme_time = 1000;
        }
    }
}
if(bit_state0 == 1 && bit_state1 == 1){
    state = 0;
}

switch(state){
    case 0:
        fr = f_open(&fil, "nemu.bin", FA_READ);
    break;
    case 1:
        fr = f_open(&fil, "hidari.bin", FA_READ);
    break;
    case 2:
        fr = f_open(&fil, "migi.bin", FA_READ);
    break;
    case 3:
        fr = f_open(&fil, "hart.bin", FA_READ);
    break;
    default:
    break;
}

if (fr) printf("open error: %d!¥n¥r", (int)fr);
offset = 0;
fr = f_read(&fil, image, sizeof(image), &br);
LCD_ShowPicture(0,0,159,39);
```

```
            offset += 12800;
            f_lseek(&fil, offset);
            LEDB_TOG;
            fr = f_read(&fil, image, sizeof(image), &br);
            LCD_ShowPicture(0,40,159,79);
            offset += 12800;
            f_lseek(&fil, offset);
            LEDB_TOG;

            f_close(&fil);
            delay_1ms(omeme_time);
        }
    }
    else
    {
        Draw_Circle(40,40,30,BLACK);
        while(1){
            FlagStatus bit_state0,bit_state1;
            bit_state0 = gpio_input_bit_get(GPIOB, GPIO_PIN_8);
            bit_state1 = gpio_input_bit_get(GPIOB, GPIO_PIN_9);
            printf("GPIO_B8 : %d!¥n¥r", bit_state0);
            printf("GPIO_B9 : %d!¥n¥r", bit_state1);
            delay_1ms(1000);
        }

    }
}
```