微算機原理及應用實習

Tetris battle

任課教授:李仁貴 教授

班級:電資三

許瀚允 111820021

大綱

- 專題簡介
- 功能介紹
- 情境說明 or（遊戲說明）
- 流程圖
- 重點程式說明
- 展示影片
- DEMO

專題介紹

- 平常在讀完書後都會來上一局TETR.IO放鬆身心，因此在學習LCD這個單元時立刻就想到了已經典俄羅斯方塊遊戲作為我們的期末專題。此專案使用STM3210E-EVAL, 結合多種周邊功能實現經典的簡易版俄羅斯方塊。

功能清單

- GPIO：使用GPIO按鈕，控制方塊的左右、旋轉，並使用LED提示玩家方塊的狀態。
- UART：在Realterm中，向玩家顯示遊玩時間、即時的下降速度。
- ADC ：利用可變電阻調整下降速度。
- TIMER ：控制下降頻率、同時顯示時間及速度。
- LCD ：將遊戲畫面顯示在螢幕上。

情境說明 or（遊戲說明）

- **情境說明：**

我的Tetris Battle 是一種單人俄羅斯方塊對戰遊戲，玩家需要將掉落的方塊快速排列並消除橫列，消除越多橫列就能給自己提供更多情緒價值，若沒有成功消除則會產生一些垃圾。遊戲設計主要是透過速度、策略和反應來進行較量。

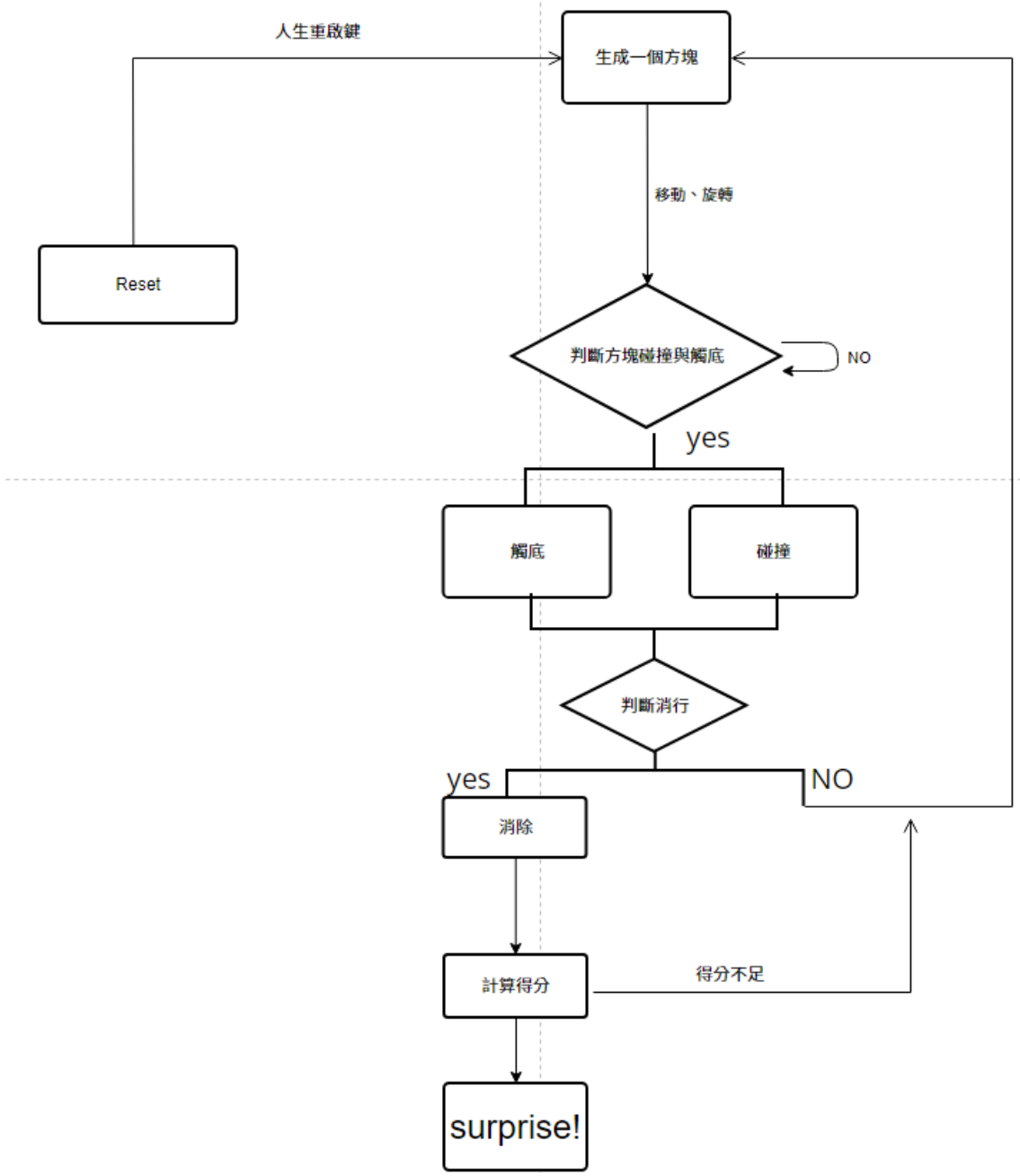
情境說明 or（遊戲說明）

- **遊戲所需功能：**

1. **基本遊戲功能：**
    1. **方塊生成：** 使用經典的seven bag的方式，來生成方塊，使之不會重複掉落
    2. **橫列消除機制：** 方塊堆疊滿一行即消除該行，並加分，達到4分即可獲得讓你爪出來的獎勵。

        BONUS機制： 消除一行得 1 point
        消除四行解鎖隱藏福利
2. **遊戲模式：**
    1. **消消樂：** 挑戰在最短時間，獲得4分。
    2. **碰頂端即結束遊戲**
3. **客製化選項：**
    1. 因為螢幕大小限制，因此我將旋轉時會同時掉落的規則改成不掉落
    2. 因應每個人的反應速度，我們可以隨時在遊戲中調整下降速度

流程圖


人生重啟鍵
生成一個方塊
移動、旋轉
Reset
判斷方塊碰撞與觸底
NO
yes
觸底
碰撞
判斷消行
yes
NO
消除
計算得分
得分不足
surprise!

重點程式說明

- LCD 方塊與基底矩陣定義(方塊共十九塊)

```
int base[12][12] = {0};
//11&8
int tetr[19][4][4]={{{0,0,1,1},{0,0,1,1},{0,0,0,0},{0,0,0,0}},//O       0
                    {{1,1,1,1},{0,0,0,0},{0,0,0,0},{0,0,0,0}},//I1      1
                    {{0,0,0,1},{0,0,0,1},{0,0,0,1},{0,0,0,1}},//I2      2
                    {{0,0,0,1},{0,0,1,1},{0,0,0,1},{0,0,0,0}},//T1      3
                    {{0,1,1,1},{0,0,1,0},{0,0,0,0},{0,0,0,0}},//T2      4
                    {{0,0,1,0},{0,0,1,1},{0,0,1,0},{0,0,0,0}},//T3      5
                    {{0,0,1,0},{0,1,1,1},{0,0,0,0},{0,0,0,0}},//T4      6
                    {{0,0,1,1},{0,0,0,1},{0,0,0,1},{0,0,0,0}},//L1      7
                    {{0,1,1,1},{0,1,0,0},{0,0,0,0},{0,0,0,0}},//L2      8
                    {{0,0,1,0},{0,0,1,0},{0,0,1,1},{0,0,0,0}},//L3      9
                    {{0,0,0,1},{0,1,1,1},{0,0,0,0},{0,0,0,0}},//L4      10
                    {{0,0,0,1},{0,0,0,1},{0,0,1,1},{0,0,0,0}},//J1      11
                    {{0,1,1,1},{0,0,0,1},{0,0,0,0},{0,0,0,0}},//J2      12
                    {{0,0,1,1},{0,0,1,0},{0,0,1,0},{0,0,0,0}},//J3      13
                    {{0,1,0,0},{0,1,1,1},{0,0,0,0},{0,0,0,0}},//J4      14
                    {{0,0,0,1},{0,0,1,1},{0,0,1,0},{0,0,0,0}},//S1      15
                    {{0,1,1,0},{0,0,1,1},{0,0,0,0},{0,0,0,0}},//S2      16
                    {{0,0,1,0},{0,0,1,1},{0,0,0,1},{0,0,0,0}},//Z1      17
                    {{0,0,1,1},{0,1,1,0},{0,0,0,0},{0,0,0,0}}};//Z2     18
```

重點程式說明

- 當前 base 矩陣製作（無碰撞與觸底時先把方塊暫放base）

```
void drawi(int init_x, int init_y,int type){
    for(int i = init_x; i < init_x+4; i++){
            for(int j = init_y; j < init_y+4; j++){
                if(base[i][j] ==0 && tetr[type][i-init_x][j-init_y] ==1 ){
                    base[i][j]=1;
                }
            }
    }

}
```

重點程式說明

- LCD方塊自定義

```
0xFF ,0xFF ,0xFF ,0xFF ,0xFF ,0xFF ,0xFF ,0xFF ,0x00 ,0x00 ,0x00 ,0x00 ,0x00 ,0x00 ,0x00,0x00 , //all 22
0x00 ,0x00 ,0x00 ,0x00 ,0x00 ,0x00 ,0x00 ,0x00 ,0xFF ,0xFF ,0xFF ,0xFF ,0xFF ,0xFF ,0xFF ,0xFF ,//half 23
0x00 ,0x00 ,0x00 ,0x00 ,0x00 ,0x00 ,0x00 ,0x00 ,0x00 ,0x00 ,0x00 ,0x00 ,0x00 ,0x00 ,0x00 ,0x00 , //void 24
0xFF , 0x00 , 0x00 , 0x00 , 0x00 , 0x00 , 0x00 , 0x00 , 0xFF , 0x00 , 0x00 , 0x00 , 0x00 , 0x00 , 0x00 , 0x00 , //right side 25
0x00 , 0x00 , 0x00 , 0x00 , 0x00 , 0x00 , 0x00 , 0x00 , 0xFF , 0x00 , 0x00 , 0x00 , 0x00 , 0x00 , 0x00 , 0x00 , 0xFF , //left side 26
0xFE , 0xFF , 0x83 , 0x83 , 0x83 , 0x83 , 0x83 , 0x02 , 0x40 , 0xC1 , 0xC1 , 0xC1 , 0xC1 , 0xC1 , 0xFF , 0x7F , //S 27
0x00 , 0xFF , 0x02 , 0x1C , 0xE0 , 0x1C , 0x02 , 0xFF , 0x00 , 0xFF , 0x00 , 0x00 , 0xFF , 0x00 , 0x00 , 0xFF , //M 28
0xFF , 0xFF , 0x00 , 0x00 , 0x00 , 0x00 , 0xFF , 0xFF , 0x1F , 0x7F , 0x60 , 0xC0 , 0xC0 , 0x60 , 0x7F , 0x1F , //U 29


//Display English letter
/*--  文字:  B  --30*/
0xFF , 0xFF , 0x83 , 0x83 , 0xC7 , 0xC6 , 0x3C , 0x00 , 0xFF , 0xFF , 0xC1 , 0xC1 , 0xC3 , 0x63 , 0x7E , 0x00 , //B

/*--  文字:  0  --31*/
0xFF , 0xFF , 0x03 , 0x03 , 0x03 , 0x03 , 0xFF , 0xFF , 0xFF , 0xFF , 0xC0 , 0xC0 , 0xC0 , 0xC0 , 0xFF , 0xFF , //0
/*--  文字:  1  --32*/
0x08 , 0xC , 0xE , 0xFF , 0xFF , 0x00 , 0x00 , 0x00 , 0xC0 , 0xC0 , 0xC0 , 0xFF , 0xFF , 0xC0 , 0xC0 , 0xC0 , //1

/*--  文字:  5  --33*/
0xFF , 0xFF , 0x83 , 0x83 , 0x83 , 0x83 , 0x83 , 0x83 , 0xC1 , 0xC1 , 0xC1 , 0xC1 , 0xC1 , 0xC1 , 0xFF , 0xFF , //5
```

重點程式說明

- 當前方塊 LCD 繪製（包含邊界繪製）

```
void printall()//print base
    {
        LCD_Clear();
        int x=16;
        int y=22;
        for(int i = 0; i < 11 ; i++){
            for(int j = 0; j < 10; j++ ){
                if(base[i][j] == 1){
                    LCD_DrawString(x+j, y+(8*i) , String1,sizeof(String1));
                }
            }
        }
        LCD_DrawString(x, y-8 , String5,sizeof(String5));
        LCD_DrawString(x, y+(8*11) , String4,sizeof(String4));
        LCD_DrawString(x+2, y-8 , String5,sizeof(String5));
        LCD_DrawString(x+2, y+(8*11) , String4,sizeof(String4));
        LCD_DrawString(x+4, y-8 , String5,sizeof(String5));
        LCD_DrawString(x+4, y+(8*11) , String4,sizeof(String4));
        LCD_DrawString(x+6, y-8 , String5,sizeof(String5));
        LCD_DrawString(x+6, y+(8*11) , String4,sizeof(String4));
        LCD_DrawString(x+8, y-8 , String5,sizeof(String5));
        LCD_DrawString(x+8, y+(8*11) , String4,sizeof(String4));
        LCD_DrawString(x+10, y-8 , String5,sizeof(String5));
        LCD_DrawString(x+10, y+(8*11) , String4,sizeof(String4));
    return;
}
```

重點程式說明

- 清除目前方塊（下墜時要先清除目前方塊以免重疊）

```
void qingqiang(int init_x, int init_y,int type){  //clean

    for(int i = init_x; i < init_x+4; i++){
        for(int j = init_y; j < init_y+4; j++){
            if(base[i][j] ==1 && tetr[type][i-init_x][j-init_y] ==1 ){
                base[i][j]=0;
            }
        }
    }
}
```

重點程式說明

- 當旋轉方塊型態改變

```
int rozhao(int type){
    if(type == 0){
    }else if(type == 1){
        type = 2;
    }else if(type == 2){
        type = 1;
    }else if(type == 3){
        type = 4;
    }else if(type == 4){
        type = 5;
    }else if(type == 5){
        type = 6;
    }else if(type == 6){
        type = 3;
    }else if(type == 7){
        type = 8;
    }else if(type == 8){
        type = 9;
    }else if(type == 9){
        type = 10;
    }else if(type == 10){
        type = 7;
    }else if(type == 11){
        type = 12;
    }else if(type == 12){
        type = 13;
    }else if(type == 13){
        type = 14;
    }else if(type == 14){
        type = 11;
    }else if(type == 15){
        type = 16;
    }else if(type == 16){
        type = 15;
    }else if(type == 17){
        type = 18;
    }else if(type == 18){
        type = 17;
    }
    return type;
}
```

重點程式說明

- 判斷左右移動以及是否旋轉（當旋轉時不會下墜）

```
int testicle(int type){

      if(HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13)==0){
            px++;
            HAL_Delay(30);
        }

        if(HAL_GPIO_ReadPin(GPIOA, GPIO_PIN_0)){
            px--;
            //HAL_Delay(30);
        }

        if(!HAL_GPIO_ReadPin(GPIOB, GPIO_PIN_10)){
            type = rozhao(type);
            HAL_Delay(30);
        }else{
            py++;
        }

    return 0;
}
```

重點程式說明

- 橫排消除

```
void cum(){    //clear cum
    int test[12][12];
    int count = 0;
    int full[12] = {0};
    for(int j = 0; j < 12; j++ ){ //put base in the array //put in main? //copy
        for(int i = 0; i < 12; i++){
            test[i][j] =0;
        }
    }
    for(int j = 0; j < 12; j++ ){ //put base in the array //put in main? //copy
        for(int i = 0; i < 12; i++){
            test[i][j] = base[i][j];
        }
    }
    for(int j = 0; j < 12; j++ ){ //count the full row //////scan the full 1 row
        for(int i = 0; i < 12; i++){
            if(base[i][j]==1){
                count = count + 1;
            }
        }
        if(count >= 11){ //the full row is 1
            full[j]=1;
            surprise++;
        }
        count = 0;
    }
    for(int j = 11; j >= 0; j-- ){ //buttom to top the whole row, up row replace the buttom
        if(full[j]==1){
            for(int q = j; q >= 0 ; q--){ //let the upper row get down
                for(int k = 0; k < 12; k++){

                    base[k][q] = 0;
                    base[k][q] = test[k][q-1];
                    base[k][q-1] = 0;
                }
            }
        }
    }
}
```

重點程式說明

- 自動換方塊（觸底或是碰撞後生成下一個形狀方塊）

```
if(i == size){
    i = 0;
}
type = number[i++];
```

重點程式說明

- 碰撞判斷與執行

```
int collanpa(int init_x, int init_y, int type){
        for(int i = 0; i <4  ; i++){
            for(int j = 0; j < 4; j++){    //need to change
                    if(base[i+init_x][j+init_y+1] ==1 && tetr[type][i][j] ==1 ){
                            return 1;
                }
            }
     }
     return 0;
}
```

重點程式說明

- 如果碰撞條件成立執行下列程式(包含消行、消除行數判斷、生成下一個方塊)

```
if(collanpa(px, py, type)){
    drawi(px, py, type);
    cum();
    cum();
    cum();
    cum();
    if(surprise >= 1){
        LCD_Clear();
        LCD_DrawString(18, 22+(8*0) , String6,sizeof(String6));
        LCD_DrawString(18, 22+(8*1) , String7,sizeof(String7));
        LCD_DrawString(18, 22+(8*2) , String8,sizeof(String8));
        LCD_DrawString(18, 22+(8*3) , String9,sizeof(String9));
        LCD_DrawString(18, 22+(8*4) , String10,sizeof(String10));
        LCD_DrawString(18, 22+(8*5) , String11,sizeof(String11));
        LCD_DrawString(18, 22+(8*6) , String12,sizeof(String12));
        return 0 ;
    }
    printall();
    HAL_Delay(100);
    px = 4;
    py = -1;
    if(i == size){
        i = 0;
    }
    type = number[i++];
```

```
    HAL_GPIO_TogglePin(GPIOF,GPIO_PIN_6);
    HAL_GPIO_TogglePin(GPIOF,GPIO_PIN_7);
    HAL_GPIO_TogglePin(GPIOF,GPIO_PIN_8);
    HAL_GPIO_TogglePin(GPIOF,GPIO_PIN_9);
    HAL_Delay(30);
    HAL_GPIO_TogglePin(GPIOF,GPIO_PIN_6);
    HAL_GPIO_TogglePin(GPIOF,GPIO_PIN_7);
    HAL_GPIO_TogglePin(GPIOF,GPIO_PIN_8);
    HAL_GPIO_TogglePin(GPIOF,GPIO_PIN_9);
    HAL_Delay(30);
    HAL_GPIO_TogglePin(GPIOF,GPIO_PIN_6);
    HAL_GPIO_TogglePin(GPIOF,GPIO_PIN_7);
    HAL_GPIO_TogglePin(GPIOF,GPIO_PIN_8);
    HAL_GPIO_TogglePin(GPIOF,GPIO_PIN_9);
    HAL_Delay(30);
    HAL_GPIO_TogglePin(GPIOF,GPIO_PIN_6);
    HAL_GPIO_TogglePin(GPIOF,GPIO_PIN_7);
    HAL_GPIO_TogglePin(GPIOF,GPIO_PIN_8);
    HAL_GPIO_TogglePin(GPIOF,GPIO_PIN_9);
    continue;
}
```

重點程式說明

- ADC控制方塊掉落速度

```
if(a > 0 && a<1365){
speed=1500;
}else if(a>= 1366 && a <= 2730){
speed = 1000;
}else{
speed = 500;
}
```

重點程式說明

- 中斷Timer 計算遊玩時間

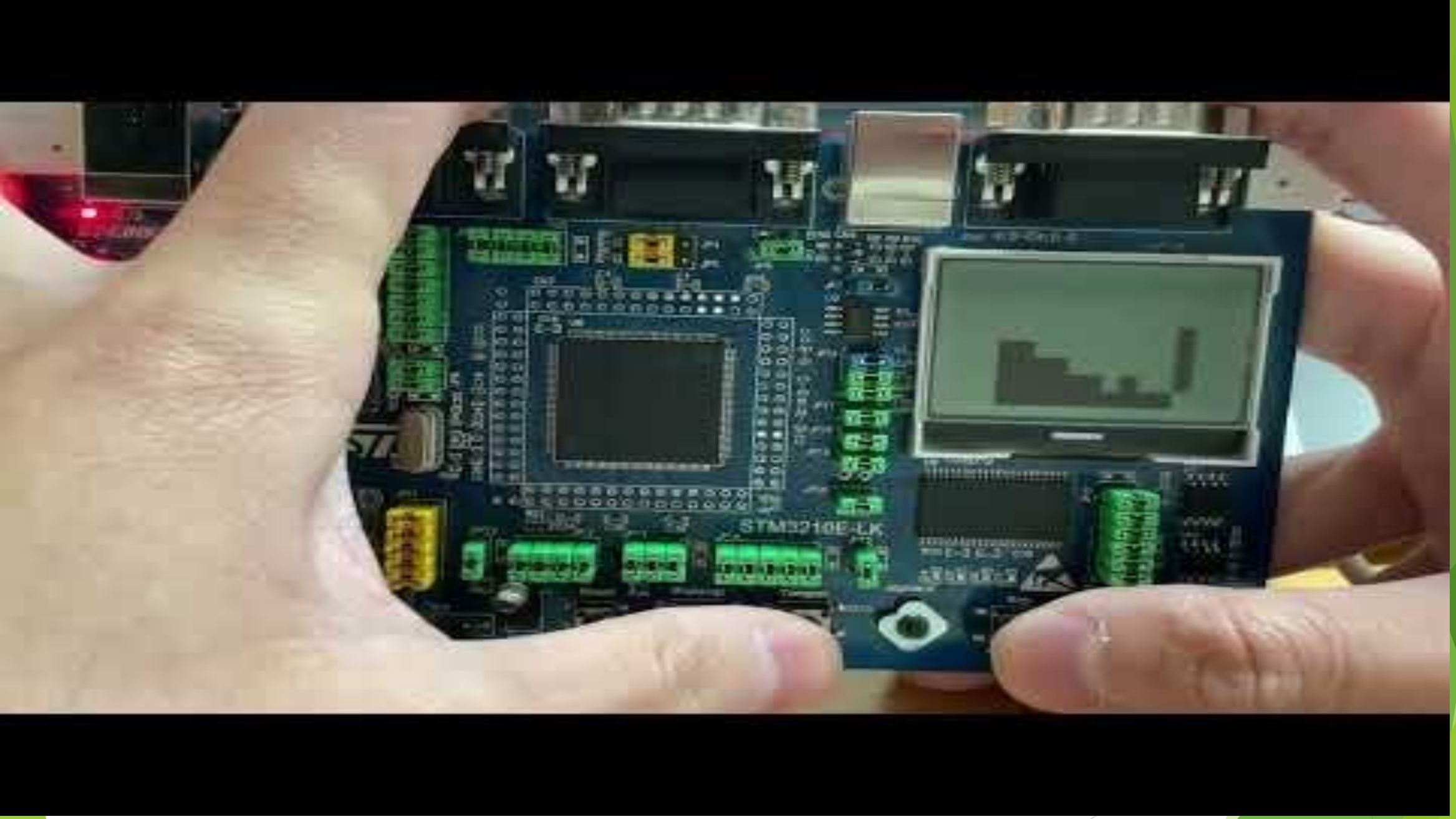
```
void TIM1_UP_IRQHandler(void)
{

  if(second == 59){
    second = 0;
    minute++;

  }
  printf("%d : %d   ", minute, second);
  printf("speed = %d\n\r",speed);

  second++;

    HAL_TIM_IRQHandler(&htim1);
  }
```



https://youtu.be/0FonHs35lU0