

微算機原理及應用實習

期末專題報告書

隨機抽籤機

任課教師：李仁貴 老師

班級：電子二丙

學生：游鎮遠107360734

學生：李彥霖107360741

學生：陳柏宇107360740

組別：第5組

前言：

目錄：

動機與目的：

在國、高中的課程中，老師常會透過抽籤的方式來進行分組或抽人問問題，這樣人為的抽籤往往會有利弊，每個人都當過學生，都會有些投機取巧的心態不想被抽到而把自己的籤偷偷先藏起來，為了讓往後的抽籤更有公平性，我們想到一個辦法來製造隨機抽籤機，讓每個人都有被抽到的機會！

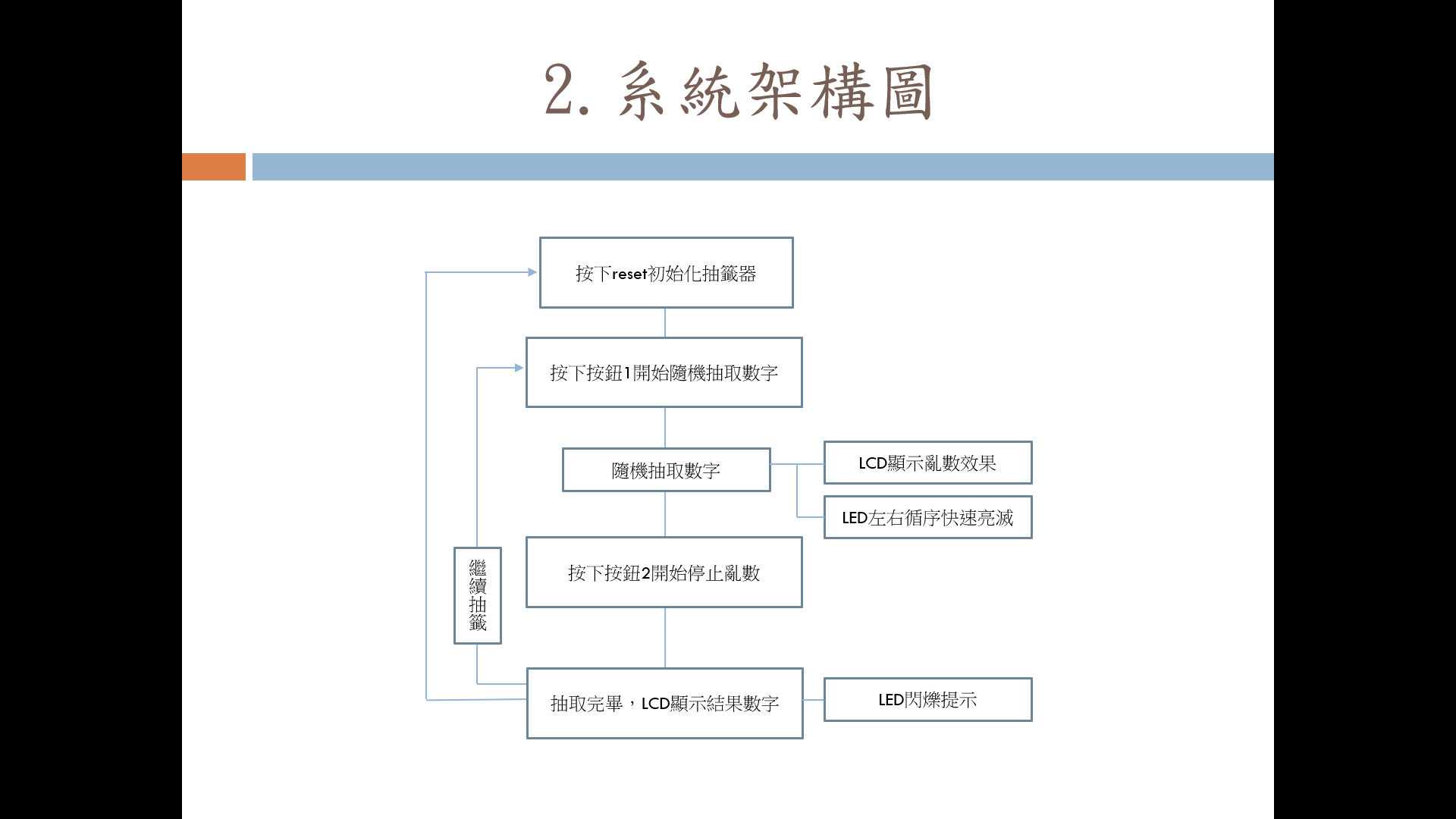
功能原理：

利用開發版上的各項功能，配合程式的編寫來做到抽籤器的效果。按鈕實現輸入。

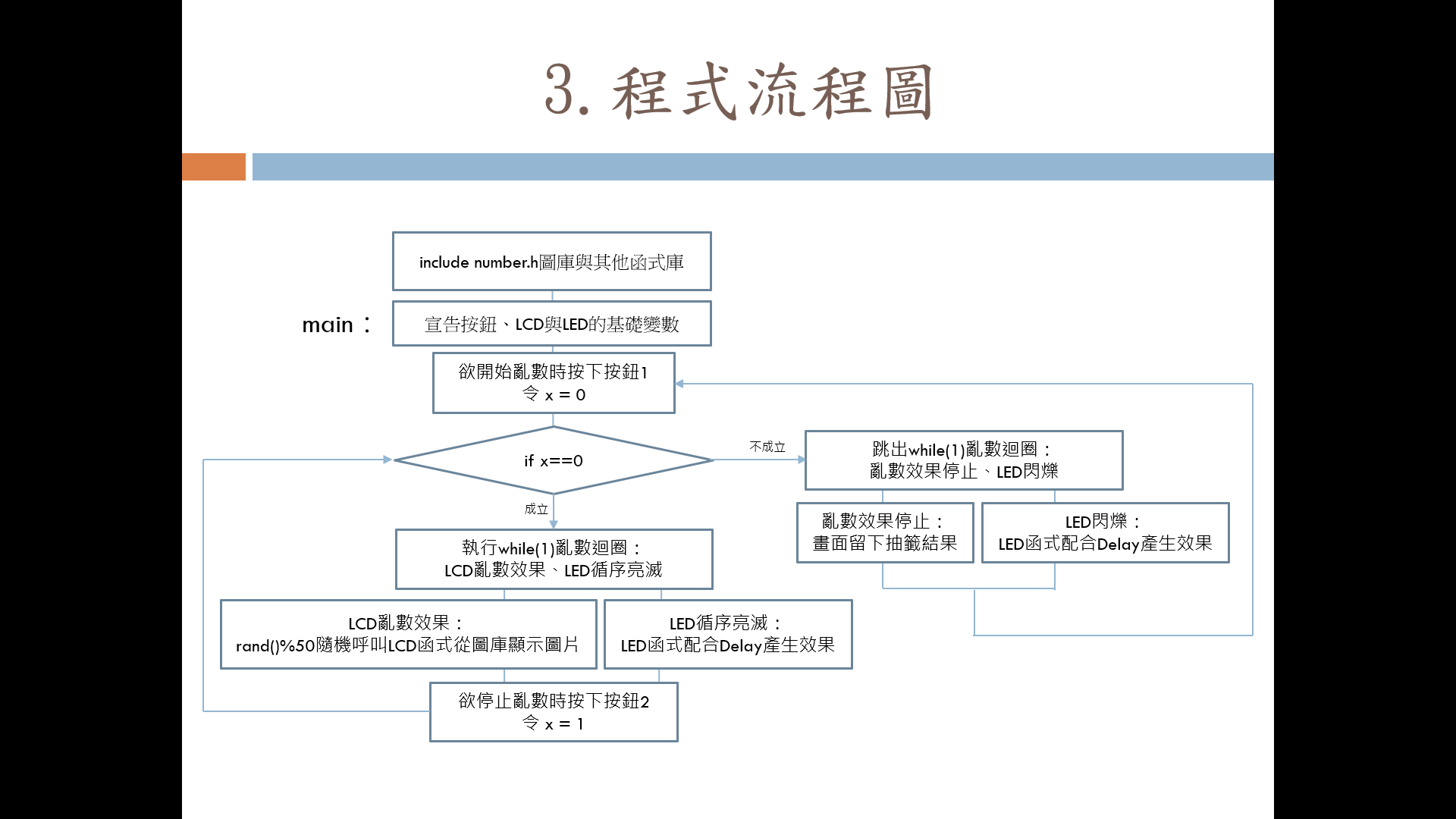
LCD做為顯示面板，顯示抽籤效果與最終的抽選結果。

LCD亮滅則做為氣氛營造的方式。

系統架構圖：



程式流程圖：



成果展示及說明：

心得：這次的專題報告，剛開始我們有很多想做的東西，但很可惜都被其他組選走了，但最後我們有了抽籤機的想法，雖然這個東西很普通，但也是考驗了我們幾個禮拜，透過程式不斷的更改以及討論，過程中也有助教的幫忙，在最後才能呈現這完美的成果。\

未來展望 :

程式碼：

/\* Includes ------------------------------------------------------------------\*/

#include "stm32f1xx\_hal.h"

#include "main.h"

#include "lcd.h"

#include "number.h"

#include "stdlib.h"

#include "stdio.h"

#ifdef \_\_GNUC\_\_

#define PUTCHAR\_PROTOTYPE int \_\_io\_putchar(int ch)

#else

#define PUTCHAR\_PROTOTYPE int fputc(int ch, FILE \*f)

#endif /\* \_\_GNUC\_\_ \*/

/\* USER CODE BEGIN Includes \*/

/\* USER CODE END Includes \*/

/////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

/\* Private variables ---------------------------------------------------------\*/

UART\_HandleTypeDef huart1;

/////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

ADC\_HandleTypeDef hadc1;

/\* USER CODE BEGIN PV \*/

/\* Private variables ---------------------------------------------------------\*/

static GPIO\_InitTypeDef GPIO\_InitStruct;

/\* USER CODE END PV \*/

/\* Private function prototypes -----------------------------------------------\*/

void SystemClock\_Config(void);

static void MX\_GPIO\_Init(void);

///////////////////////////////////////////////////////////////////////////////////////////////////

static void MX\_USART1\_UART\_Init(void); //+codeAa-E?Y-n|33o|a

///////////////////////////////////////////////////////////////////////////////////////////////////

static void MX\_ADC1\_Init(void);

int main(void)

{

int z;

int x=0;

GPIO\_InitStruct.Pin = TAMPER\_BUTTON\_PIN;

GPIO\_InitStruct.Mode = GPIO\_MODE\_INPUT;

GPIO\_InitStruct.Pull = GPIO\_PULLUP;

GPIO\_InitStruct.Speed = GPIO\_SPEED\_FREQ\_HIGH;

HAL\_GPIO\_Init(TAMPER\_BUTTON\_GPIO\_PORT, &GPIO\_InitStruct);

TAMPER\_BUTTON\_GPIO\_CLK\_ENABLE();

GPIO\_InitStruct.Pin = WAKEUP\_BUTTON\_PIN;

GPIO\_InitStruct.Mode = GPIO\_MODE\_INPUT;

GPIO\_InitStruct.Pull = GPIO\_PULLDOWN;

GPIO\_InitStruct.Speed = GPIO\_SPEED\_FREQ\_HIGH;

HAL\_GPIO\_Init(WAKEUP\_BUTTON\_GPIO\_PORT, &GPIO\_InitStruct);

WAKEUP\_BUTTON\_GPIO\_CLK\_ENABLE();

GPIO\_InitStruct.Pin = WAKEUP\_BUTTON\_PIN;

GPIO\_InitStruct.Mode = GPIO\_MODE\_INPUT;

GPIO\_InitStruct.Pull = GPIO\_PULLDOWN;

GPIO\_InitStruct.Speed = GPIO\_SPEED\_FREQ\_HIGH;

HAL\_GPIO\_Init(WAKEUP\_BUTTON\_GPIO\_PORT, &GPIO\_InitStruct);

WAKEUP\_BUTTON\_GPIO\_CLK\_ENABLE();

#define c GPIOA

#define v GPIO\_PIN\_0

#define n GPIOC

#define m GPIO\_PIN\_13

HAL\_Init();

/\* Configure the system clock \*/

SystemClock\_Config();

#if 0

LED1\_GPIO\_CLK\_ENABLE();

LED2\_GPIO\_CLK\_ENABLE();

LED3\_GPIO\_CLK\_ENABLE();

LED4\_GPIO\_CLK\_ENABLE();

GPIO\_InitStruct.Pin = LED1\_PIN;

GPIO\_InitStruct.Pin = LED2\_PIN;

GPIO\_InitStruct.Pin = LED3\_PIN;

GPIO\_InitStruct.Pin = LED4\_PIN;

GPIO\_InitStruct.Mode= GPIO\_MODE\_OUTPUT\_PP;

GPIO\_InitStruct.Pull= GPIO\_PULLUP;

GPIO\_InitStruct.Speed=GPIO\_SPEED\_FREQ\_HIGH;

HAL\_GPIO\_Init(LED1\_GPIO\_PORT, &GPIO\_InitStruct);

HAL\_GPIO\_Init(LED2\_GPIO\_PORT, &GPIO\_InitStruct);

HAL\_GPIO\_Init(LED3\_GPIO\_PORT, &GPIO\_InitStruct);

HAL\_GPIO\_Init(LED4\_GPIO\_PORT, &GPIO\_InitStruct);

#endif

/\* Initialize all configured peripherals \*/

MX\_GPIO\_Init();

////////////////////////////////////////////////////////////////////////

MX\_USART1\_UART\_Init();

//////////////////////////////////////////////////////////////////////////

MX\_ADC1\_Init();

LCD\_Init();

LCD\_Clear();

start:

if(HAL\_GPIO\_ReadPin(c, v)){

while (x==0){

z=rand()%50;

if(z==0)

LCD\_Draw\_Logo(Logo0);

else if(z==1)

LCD\_Draw\_Logo(Logo1);

else if(z==2)

LCD\_Draw\_Logo(Logo2);

else if(z==3)

LCD\_Draw\_Logo(Logo3);

else if(z==4)

LCD\_Draw\_Logo(Logo4);

else if(z==5)

LCD\_Draw\_Logo(Logo5);

else if(z==6)

LCD\_Draw\_Logo(Logo6);

else if(z==7)

LCD\_Draw\_Logo(Logo7);

else if(z==8)

LCD\_Draw\_Logo(Logo8);

else if(z==9)

LCD\_Draw\_Logo(Logo9);

else if(z==10)

LCD\_Draw\_Logo(Logo10);

else if(z==11)

LCD\_Draw\_Logo(Logo11);

else if(z==12)

LCD\_Draw\_Logo(Logo12);

else if(z==13)

LCD\_Draw\_Logo(Logo13);

else if(z==14)

LCD\_Draw\_Logo(Logo14);

else if(z==15)

LCD\_Draw\_Logo(Logo15);

else if(z==16)

LCD\_Draw\_Logo(Logo16);

else if(z==17)

LCD\_Draw\_Logo(Logo17);

else if(z==18)

LCD\_Draw\_Logo(Logo18);

else if(z==19)

LCD\_Draw\_Logo(Logo19);

else if(z==20)

LCD\_Draw\_Logo(Logo20);

else if(z==21)

LCD\_Draw\_Logo(Logo21);

else if(z==22)

LCD\_Draw\_Logo(Logo22);

else if(z==23)

LCD\_Draw\_Logo(Logo23);

else if(z==24)

LCD\_Draw\_Logo(Logo24);

else if(z==25)

LCD\_Draw\_Logo(Logo25);

else if(z==26)

LCD\_Draw\_Logo(Logo26);

else if(z==27)

LCD\_Draw\_Logo(Logo27);

else if(z==28)

LCD\_Draw\_Logo(Logo28);

else if(z==29)

LCD\_Draw\_Logo(Logo29);

else if(z==30)

LCD\_Draw\_Logo(Logo30);

else if(z==31)

LCD\_Draw\_Logo(Logo31);

else if(z==32)

LCD\_Draw\_Logo(Logo32);

else if(z==33)

LCD\_Draw\_Logo(Logo33);

else if(z==34)

LCD\_Draw\_Logo(Logo34);

else if(z==35)

LCD\_Draw\_Logo(Logo35);

else if(z==36)

LCD\_Draw\_Logo(Logo36);

else if(z==37)

LCD\_Draw\_Logo(Logo37);

else if(z==38)

LCD\_Draw\_Logo(Logo38);

else if(z==39)

LCD\_Draw\_Logo(Logo39);

else if(z==40)

LCD\_Draw\_Logo(Logo40);

else if(z==41)

LCD\_Draw\_Logo(Logo41);

else if(z==42)

LCD\_Draw\_Logo(Logo42);

else if(z==43)

LCD\_Draw\_Logo(Logo43);

else if(z==44)

LCD\_Draw\_Logo(Logo44);

else if(z==45)

LCD\_Draw\_Logo(Logo45);

else if(z==46)

LCD\_Draw\_Logo(Logo46);

else if(z==47)

LCD\_Draw\_Logo(Logo47);

else if(z==48)

LCD\_Draw\_Logo(Logo48);

else if(z==49)

LCD\_Draw\_Logo(Logo49);

else if(z==50)

LCD\_Draw\_Logo(Logo50);

HAL\_GPIO\_TogglePin(LED1\_GPIO\_PORT,LED1\_PIN);

HAL\_Delay(5);

HAL\_GPIO\_TogglePin(LED1\_GPIO\_PORT,LED1\_PIN);

HAL\_Delay(10);

HAL\_GPIO\_TogglePin(LED2\_GPIO\_PORT,LED2\_PIN);

HAL\_Delay(5);

HAL\_GPIO\_TogglePin(LED2\_GPIO\_PORT,LED2\_PIN);

HAL\_Delay(10);

HAL\_GPIO\_TogglePin(LED3\_GPIO\_PORT,LED3\_PIN);

HAL\_Delay(5);

HAL\_GPIO\_TogglePin(LED3\_GPIO\_PORT,LED3\_PIN);

HAL\_Delay(10);

HAL\_GPIO\_TogglePin(LED4\_GPIO\_PORT,LED4\_PIN);

HAL\_Delay(5);

HAL\_GPIO\_TogglePin(LED4\_GPIO\_PORT,LED4\_PIN);

HAL\_Delay(10);

HAL\_GPIO\_TogglePin(LED3\_GPIO\_PORT,LED3\_PIN);

HAL\_Delay(5);

HAL\_GPIO\_TogglePin(LED3\_GPIO\_PORT,LED3\_PIN);

HAL\_Delay(10);

HAL\_GPIO\_TogglePin(LED2\_GPIO\_PORT,LED2\_PIN);

HAL\_Delay(5);

HAL\_GPIO\_TogglePin(LED2\_GPIO\_PORT,LED2\_PIN);

HAL\_Delay(10);

if(HAL\_GPIO\_ReadPin(n, m)==0){

x=1;

};

};

while(x==1){

HAL\_GPIO\_TogglePin(LED1\_GPIO\_PORT,LED1\_PIN);

HAL\_GPIO\_TogglePin(LED2\_GPIO\_PORT,LED2\_PIN);

HAL\_GPIO\_TogglePin(LED3\_GPIO\_PORT,LED3\_PIN);

HAL\_GPIO\_TogglePin(LED4\_GPIO\_PORT,LED4\_PIN);

HAL\_Delay(500);

HAL\_GPIO\_TogglePin(LED1\_GPIO\_PORT,LED1\_PIN);

HAL\_GPIO\_TogglePin(LED2\_GPIO\_PORT,LED2\_PIN);

HAL\_GPIO\_TogglePin(LED3\_GPIO\_PORT,LED3\_PIN);

HAL\_GPIO\_TogglePin(LED4\_GPIO\_PORT,LED4\_PIN);

HAL\_Delay(500);

if(HAL\_GPIO\_ReadPin(c, v)){

x=0;

};

};

};

goto start;

};

PUTCHAR\_PROTOTYPE

{

/\* Place your implementation of fputc here \*/

/\* e.g. write a character to the USART1 and Loop until the end of transmission \*/

HAL\_UART\_Transmit(&huart1, (uint8\_t \*)&ch, 1, 0xffff);

return ch;

}

/\*\* System Clock Configuration

\*/

void SystemClock\_Config(void)

{

RCC\_OscInitTypeDef RCC\_OscInitStruct;

RCC\_ClkInitTypeDef RCC\_ClkInitStruct;

RCC\_OscInitStruct.OscillatorType = RCC\_OSCILLATORTYPE\_HSI;

RCC\_OscInitStruct.HSIState = RCC\_HSI\_ON;

RCC\_OscInitStruct.HSICalibrationValue = 16;

RCC\_OscInitStruct.PLL.PLLState = RCC\_PLL\_NONE;

HAL\_RCC\_OscConfig(&RCC\_OscInitStruct);

RCC\_ClkInitStruct.ClockType = RCC\_CLOCKTYPE\_SYSCLK;

RCC\_ClkInitStruct.SYSCLKSource = RCC\_SYSCLKSOURCE\_HSI;

RCC\_ClkInitStruct.AHBCLKDivider = RCC\_SYSCLK\_DIV1;

RCC\_ClkInitStruct.APB1CLKDivider = RCC\_HCLK\_DIV1;

RCC\_ClkInitStruct.APB2CLKDivider = RCC\_HCLK\_DIV1;

HAL\_RCC\_ClockConfig(&RCC\_ClkInitStruct, FLASH\_LATENCY\_0);

HAL\_SYSTICK\_Config(HAL\_RCC\_GetHCLKFreq()/1000);

HAL\_SYSTICK\_CLKSourceConfig(SYSTICK\_CLKSOURCE\_HCLK);

}

/\* ADC1 init function \*/

void MX\_ADC1\_Init(void)

{

ADC\_ChannelConfTypeDef sConfig;

/\*\*Common config

\*/

hadc1.Instance = ADC1;

hadc1.Init.ScanConvMode = ADC\_SCAN\_DISABLE;

hadc1.Init.ContinuousConvMode = DISABLE;

hadc1.Init.DiscontinuousConvMode = DISABLE;

hadc1.Init.ExternalTrigConv = ADC\_SOFTWARE\_START;

hadc1.Init.DataAlign = ADC\_DATAALIGN\_RIGHT;

hadc1.Init.NbrOfConversion = 1;

HAL\_ADC\_Init(&hadc1);

/\*\*Configure Regular Channel

\*/

sConfig.Channel = ADC\_CHANNEL\_14;

sConfig.Rank = 1;

sConfig.SamplingTime = ADC\_SAMPLETIME\_1CYCLE\_5;

HAL\_ADC\_ConfigChannel(&hadc1, &sConfig);

}

//////////////////////////////////////////////////////////////////

/\* USART1 init function \*/

void MX\_USART1\_UART\_Init(void)

{

huart1.Instance = USART1;

huart1.Init.BaudRate = 9600;

huart1.Init.WordLength = UART\_WORDLENGTH\_9B;

huart1.Init.StopBits = UART\_STOPBITS\_1;

huart1.Init.Parity = UART\_PARITY\_NONE;

huart1.Init.Mode = UART\_MODE\_TX\_RX;

huart1.Init.HwFlowCtl = UART\_HWCONTROL\_NONE;

huart1.Init.OverSampling = UART\_OVERSAMPLING\_16;

HAL\_UART\_Init(&huart1);

}

/////////////////////////////////////////////////////////////////////

/\*\* Pinout Configuration

\*/

void MX\_GPIO\_Init(void)

{

// /\* GPIO Ports Clock Enable \*/

// \_\_GPIOA\_CLK\_ENABLE();

// \_\_GPIOF\_CLK\_ENABLE();

GPIO\_InitTypeDef GPIO\_InitStruct;

/\* GPIO Ports Clock Enable \*/

\_\_GPIOF\_CLK\_ENABLE();

\_\_GPIOC\_CLK\_ENABLE();

\_\_GPIOA\_CLK\_ENABLE();

/\*Configure GPIO pins : PF6 PF7 PF8 PF9 \*/

GPIO\_InitStruct.Pin = GPIO\_PIN\_6|GPIO\_PIN\_7|GPIO\_PIN\_8|GPIO\_PIN\_9;

GPIO\_InitStruct.Mode = GPIO\_MODE\_OUTPUT\_PP;

GPIO\_InitStruct.Speed = GPIO\_SPEED\_LOW;

HAL\_GPIO\_Init(GPIOF, &GPIO\_InitStruct);

WAKEUP\_BUTTON\_GPIO\_CLK\_ENABLE();

GPIO\_InitStruct.Pin = WAKEUP\_BUTTON\_PIN;

HAL\_GPIO\_Init(WAKEUP\_BUTTON\_GPIO\_PORT, &GPIO\_InitStruct);

}

/\* USER CODE BEGIN 4 \*/

/\* USER CODE END 4 \*/

#ifdef USE\_FULL\_ASSERT

/\*\*

\* @brief Reports the name of the source file and the source line number

\* where the assert\_param error has occurred.

\* @param file: pointer to the source file name

\* @param line: assert\_param error line source number

\* @retval None

\*/

void assert\_failed(uint8\_t\* file, uint32\_t line)

{

}

#endif

/\*\*

\* @}

\*/

/\*\*

\* @}

\*/