University of Essex

School of
Computer Science and
Electronic Engineering

CE323 Advanced Embedded Systems Design
Assignment 2

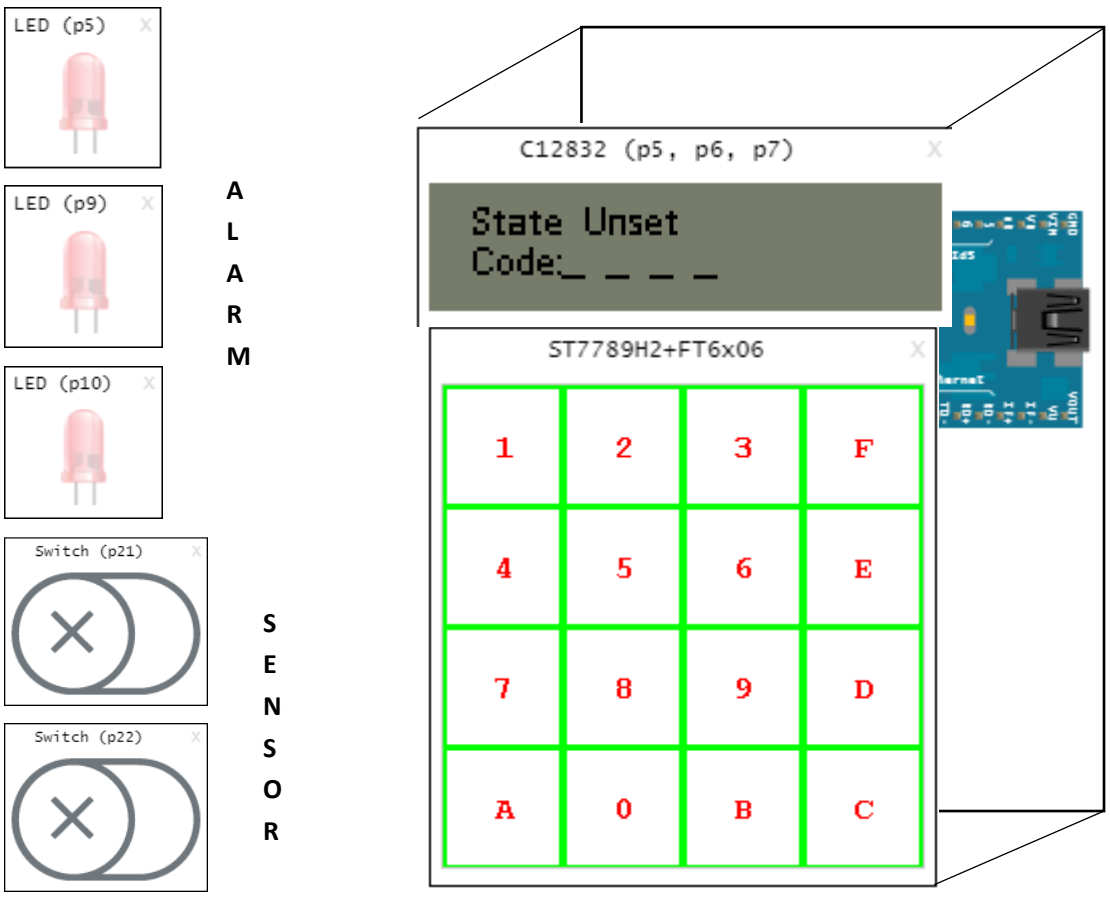
Report

Student Name: Chuan, Qin

Student ID: 1708323

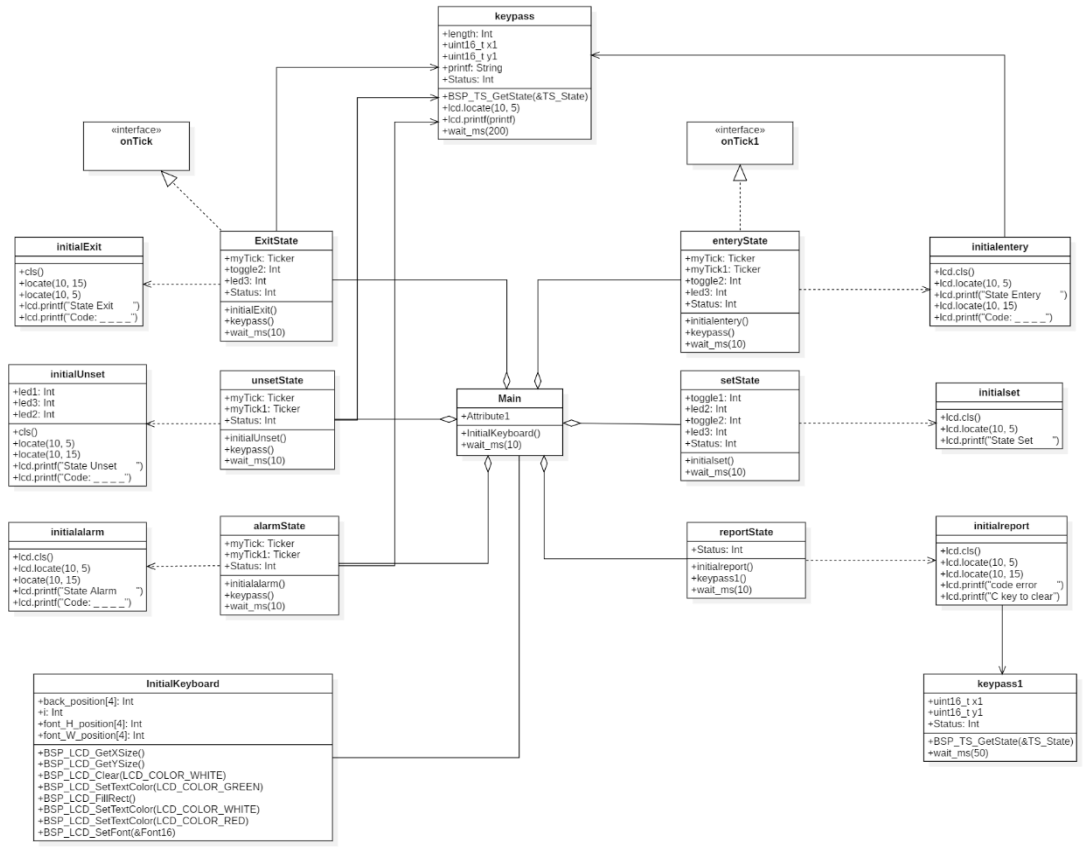
1. Requirements Form

Name	Home Alarm System
Purpose	Ensure the security of the home
Inputs	2 sensors, 1 keypad
Outputs	1 lcd, 3 led
Functions	Control home alerts by password and states
Performance	Instant alarm trigger, great sensor detection
Manufacture costs	About $85 (LPC1768 $49; LCDx2 $20; LED x3 $3; Sensors x2 $10)
Physical size/weight	No more than 5 x 10 inches, 30 oz
Power	2W



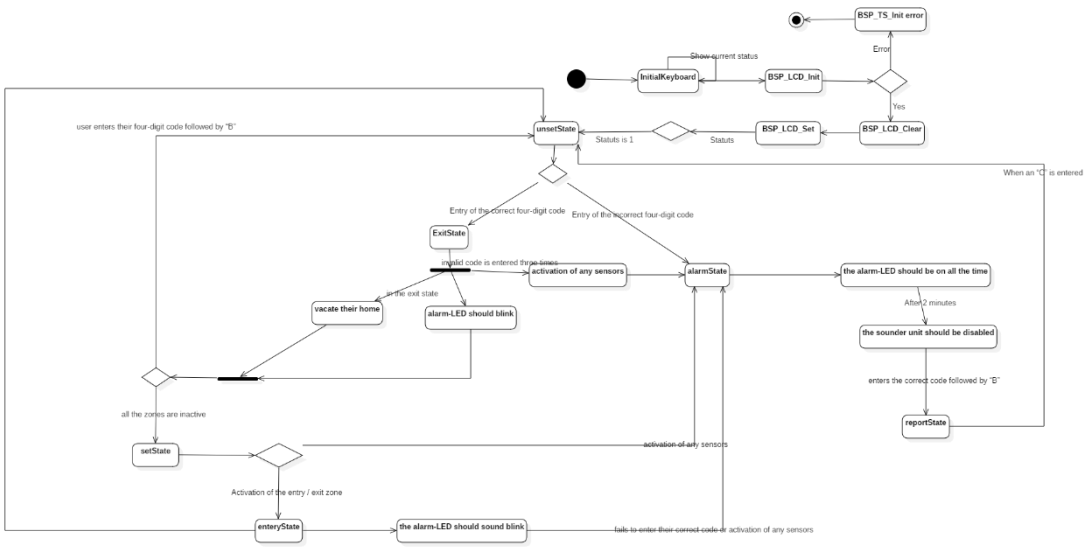
2. UML graphical representation of the system

2.1 Class



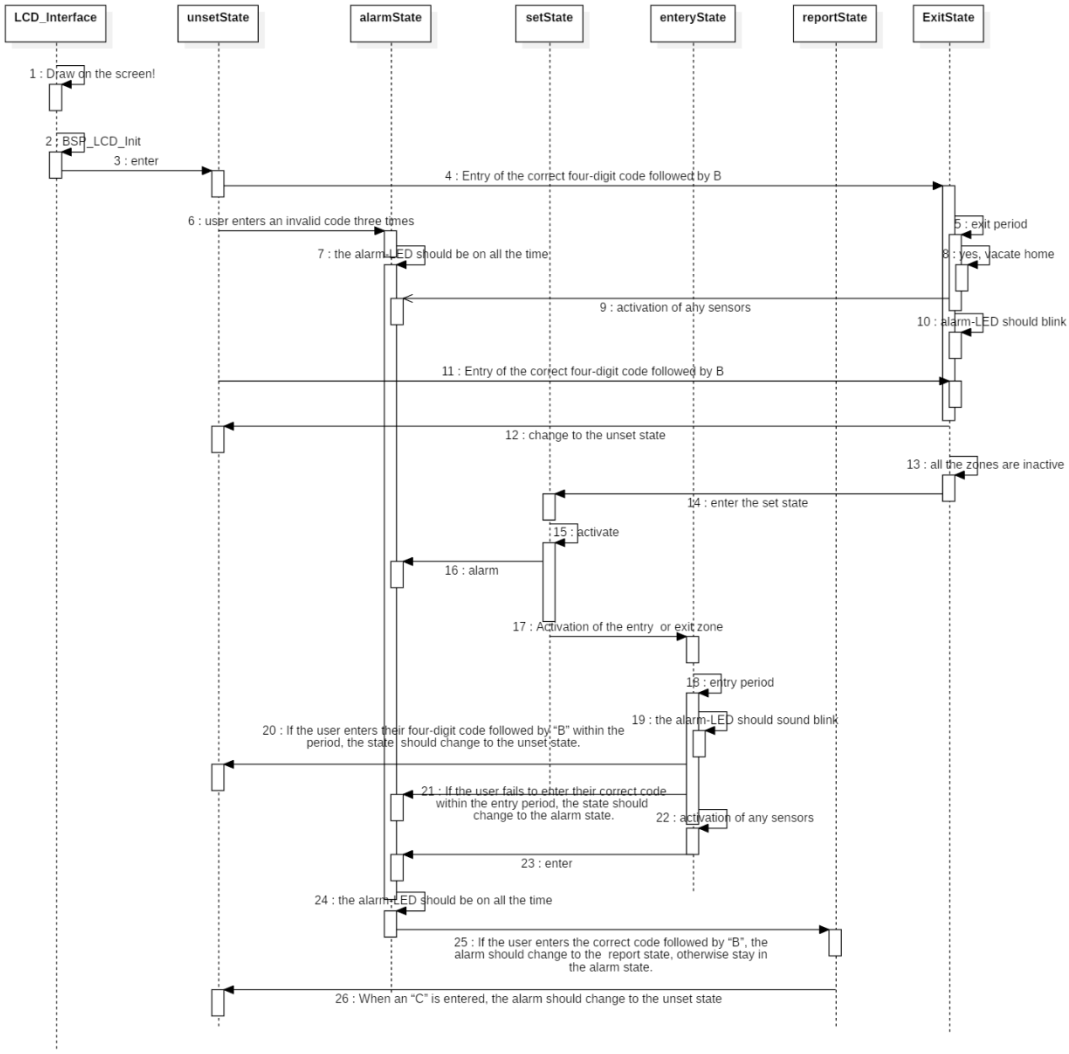
Class diagram is a static structure diagram, which describes the set of classes, the attributes of classes and the relationship between classes of the system. It can simplify people's understanding of the system.

2.2 State Machine



The state diagram is used to show the state machine (which specifies the sequence of states an object is in), the events and conditions that bring an object to those states, and the actions that take place when those states are reached.

2.3 Sequence



A sequence diagram shows dynamic collaboration between multiple objects by describing the chronological order in which messages are sent between them. It can represent the order of behavior of a use case, where each message corresponds to a class operation or trigger event in the state machine that causes a transition when a use routine is executed.

3. Source Code (Appendix)

```
	/*
	In Assignment2, I referenced the code blocks that the professor provided on Moodle, and try to restore it as much as possible,
	Call the switch function in the main function to determine the current state,
	I use the while loop in each state blocks, So i can use 'break' to jump out of the current state and back to the main function,
	And the LED flicker can be easily controlled.
	*/

	#include "mbed.h"
	#include "stm32f413h_discovery_ts.h"
	#include "stm32f413h_discovery_lcd.h"
	#include "C12832.h"

	DigitalOut led1(p5), led2(p9), led3(p10);		// define the LEDs pin
	DigitalIn toggle1(p21), toggle2(p22);		// define the toggles pin

	Ticker myTick;					// use to enter Status 4 when on Status 2
	Ticker myTick1;					// use to enter Status 3 when on Status 5
	Ticker myTick2;					// use to turn off led1 on alarm state

	C12832 lcd(SPI_MOSI, SPI_SCK, SPI_MISO, p8, p11);
	TS_StateTypeDef TS_State = { 0 };

	char* Keytable[] = {"1","2","3","F","4","5","6","E","7","8","9","D","A","0","B","C"};
	char ch[4]    = {'_','_','_','_'};		// Saves the currently entered four digits
	char password[4] = {'2','4','6','8'};		// Set password is '2 4 6 8'

	int Status = 1;					// current status
	int length = 0;					// password input length
	int try_time = 0;				// when input error, store the try time

	void onTick(void){				//    on Status 2, if nothing happen, call onTick and enter Status 4
```

```
        Status = 4;
    }

    void onTick1(void){                                     //    on Status 5, if
nothing happen, call onTick1 and enter Status 3
        Status = 3;
    }

    void disableAlarm(){                                    // when alarm shining
on Alarm Status, after 120s, close led1
        led1 = 0;
    }

    void initialset(){                                      // When enter the Set
Status, initial firstly
        lcd.cls();
        lcd.locate(10,5);lcd.printf("State Set          ");
    }

    void initialExit(){                                     // When enter the Exit
Status, initial firstly
        lcd.cls();
        lcd.locate(10,5);lcd.printf("State Exit          ");
        lcd.locate(10,15);lcd.printf("Code:_ _ _ _");
    }

    void initialUnset(){                                    // When enter the
UnSet Status, initial firstly
        led1 = 0; led2 = 0; led3 = 0;
        lcd.cls();
        lcd.locate(10,5);lcd.printf("State Unset          ");
        lcd.locate(10,15);lcd.printf("Code:_ _ _ _");
    }

    void initialentery(){                                   // When enter the
Entery Status, initial firstly
        lcd.cls();
        lcd.locate(10,5);lcd.printf("State Entery          ");
        lcd.locate(10,15);lcd.printf("Code:_ _ _ _");
    }

    void initialalarm(){                                    // When enter the Alarm
Status, initial firstly
         lcd.cls();
```

```
        lcd.locate(10,5);lcd.printf("State Alarm             ");
        lcd.locate(10,15);lcd.printf("Code:_ _ _ _");
    }

    void initialreport(){                                   // When enter the
Report Status, initial firstly
        led1 = 0;
        lcd.cls();
        lcd.locate(10,5);lcd.printf("code error 1          ");
        lcd.locate(10,15);lcd.printf("C key to clear");
    }

    void InitialKeyboard(){                                 // As same as
assignment 1
        printf("Draw on the screen!\n");
         BSP_LCD_Init();                                    // Initializes the
LCD

         if (BSP_TS_Init(BSP_LCD_GetXSize(), BSP_LCD_GetYSize()) ==
TS_ERROR)
        {
              printf("BSP_TS_Init error\n");                // If
initialization is unsuccessful, report an error
         }

         BSP_LCD_Clear(LCD_COLOR_WHITE);                    // Clear
the LCD
         BSP_LCD_SetTextColor(LCD_COLOR_GREEN);             // Set
Touchscreen description
         BSP_LCD_FillRect(0, 0, BSP_LCD_GetXSize(), 240);   // The width of
the background

         BSP_LCD_SetTextColor(LCD_COLOR_WHITE);             // Set
Rectangle description
         int back_position[4] = {2,62,122,182};             // Set the square
background position
         for (int x: back_position){
              for (int y: back_position){
                   BSP_LCD_FillRect(x,y,56,56);}
         }

         BSP_LCD_SetTextColor(LCD_COLOR_RED);               // Set
Texts description
         BSP_LCD_SetFont(&Font16);                          // Font
```

size color

```
            int i = 0;
            int font_H_position[4] ={25,85,145,205};              // Height
            int font_W_position[4] ={-85,-25,35,95};              // Wide

            for (int v: font_H_position){
                  for (int u: font_W_position){
                        BSP_LCD_DisplayStringAt(u, v, (uint8_t *)Keytable[i],
CENTER_MODE);
                        i = i+1;}
            }
      }

      void keypass(){
                  BSP_TS_GetState(&TS_State);
                   if(TS_State.touchDetected) {                    // If the screen is
detected being pressed
                        if (length<0)length = 0;                    // Since the input is
0-4 bits, the input is range-limited
                        if (length>4)length = 4;
                        /* Get X and Y position of the first touch post calibrated */
                        uint16_t x1 = TS_State.touchX[0];
                        uint16_t y1 = TS_State.touchY[0];

                        if (int(x1)>2&&int(x1)<58&&int(y1)>2&&int(y1)<62){
      // When the '1' on the screen is pressed
                              ch[length] = '1'; length = length + 1;}
 // Record '1' to 'ch', total bit length +1

                        else if (int(x1)>62&&int(x1)<118&&int(y1)>2&&int(y1)<62){
      // When the '2' on the screen is pressed
                              ch[length] = '2'; length = length + 1;}

                        else if (int(x1)>122&&int(x1)<178&&int(y1)>2&&int(y1)<62){
      // When the '3' on the screen is pressed
                              ch[length] = '3'; length = length + 1;}

                        else if (int(x1)>2&&int(x1)<58&&int(y1)>62&&int(y1)<118){
      // When the '4' on the screen is pressed
                              ch[length] = '4'; length = length + 1;}

                        else if
(int(x1)>62&&int(x1)<118&&int(y1)>62&&int(y1)<118){       // When the '5' on
the screen is pressed
```

```
                    ch[length] = '5'; length = length + 1;}

                else if
(int(x1)>122&&int(x1)<178&&int(y1)>62&&int(y1)<118){      // When the '6' on
the screen is pressed
                    ch[length] = '6'; length = length + 1;}

                else if (int(x1)>2&&int(x1)<58&&int(y1)>112&&int(y1)<178){
      // When the '7' on the screen is pressed
                    ch[length] = '7'; length = length + 1;}

                else if
(int(x1)>62&&int(x1)<118&&int(y1)>112&&int(y1)<178){      // When the '8' on
the screen is pressed
                    ch[length] = '8'; length = length + 1;}

                else if
(int(x1)>122&&int(x1)<178&&int(y1)>112&&int(y1)<178){    // When the '9' on
the screen is pressed
                    ch[length] = '9'; length = length + 1;}

                else if
(int(x1)>62&&int(x1)<118&&int(y1)>182&&int(y1)<238){      // When the '0' on
the screen is pressed
                    ch[length] = '0'; length = length + 1;}

                if (int(x1)>182&&int(x1)<238&&int(y1)>182&&int(y1)<238){
      // When the 'C' on the screen is pressed
                    ch[length - 1] = '_';
 // The previous bit is replaced to '_'
                    length = length - 1;
 // Total bit length -1
                }
               if (ch[3] != '_'){
 //Print 'Press B to Set' on the first line
                    lcd.locate(10,5);
      //if the fourth bit saved is detected.
                    lcd.printf("Press B to set");
                }
                else if (Status == 1){
                   lcd.locate(10,5);
 //if the fourth bit saved is detected.
                    lcd.printf("State Unset             ");
              }
```

```
                    else if (Status == 3){
                        lcd.locate(10,5);
//if the fourth bit saved is detected.
                            lcd.printf("State Alarm              ");
                    }
                    else if (Status == 2){
                        lcd.locate(10,5);
//if the fourth bit saved is detected.
                            lcd.printf("State Exit              ");
                    }
                      if (int(x1)>122&&int(x1)<178&&int(y1)>182&&int(y1)<238){
    // When the 'B' on the screen is pressed

if(ch[0]==password[0]&&ch[1]==password[1]&&ch[2]==password[2]&&ch[3]==p
assword[3]){
                                // If the input is equal to password, choose next
status accoarding to current status.
                                if (Status == 1){
    // Unset --> Exit
                                    Status = 2;
                                }
                                else if (Status == 2){
// Exit --> Unset
                                    Status = 1;
                                }
                                else if (Status == 5){
// Entery --> Unset
                                    Status = 1;
                                }
                                else if (Status == 3){
// Alarm --> Report
                                    Status = 6;
                                }
                                try_time = 0;
// clear try time
                            }
                            else{
    // if give wrong password, try time + 1
                                try_time += 1;
                        }
                            for(int u=0;u<4;u++){
    // reset the input
                                ch[u] = '_';
                            }
```

```
                    length = 0;
                }
                if (Status ==1 && try_time == 3 || Status ==2 && try_time ==
3){// if give wrong password 3 times
                    Status = 3;
    // Unset or Exit Status --> Alarm Status
                }
                else if (Status == 5 && try_time == 1){
//  if give wrong password 1 times
                    Status = 3;
    // Entery --> Alarm
                }
                else if (Status == 3 && try_time == 1){
//  if give wrong password 1 times
                    Status = 3;
    // Alarm --> Alarm
                }
                  lcd.locate(10,15);
    // Print the information currently entered on the second line
                  lcd.printf("Code:%c %c %c %c",ch[0],ch[1],ch[2],ch[3]);
                  wait_ms(200);
    // If 'wait' it is too small, a single touch will result in multiple inputs,
            }
    }

    void keypass1(){                                    // special use on Report
Status, because it only use 'C' button.
        BSP_TS_GetState(&TS_State);
            if(TS_State.touchDetected) {
                uint16_t x1 = TS_State.touchX[0];
                uint16_t y1 = TS_State.touchY[0];
                if (int(x1)>182&&int(x1)<238&&int(y1)>182&&int(y1)<238){
                    Status = 1;
    // if press C, Report --> Unset
                }
            }
            wait_ms(50);
    }

    void unsetState(){
        initialUnset();                                 // call to initial LCD
        myTick.detach();                                // if enter unsetState
without myTick, should detach it.
        myTick1.detach();                               // if enter
```

```
unsetState without myTick1, should detach it.

        /*  I use the while loop in each state blocks,
            So i can use 'break' to jump out of the current state and back to the
main function    */

        while(1){
            keypass();                                  // detech the input
keypass
            if (Status != 1){                           // if Status changes,
back to main function.
                break;
            }
            wait_ms(10);
        }
    }

    void alarmState(){
        myTick.detach();                                // if enter alarmState
without myTick, should detach it.
        myTick1.detach();                               // if enter
alarmState without myTick1, should detach it.
        initialalarm();
        myTick2.attach(&disableAlarm, 120);             // if nothing happen
after 120s, turn off led1.
        led1 = 1;                                       // on Alarm status, led1
blink on.
        while(1){
            keypass();
            if (Status != 3){
                break;
            }
            wait_ms(10);
        }
    }

    void enteryState(){
        myTick.detach();                            // if enter enteryState without
myTick, should detach it.
        initialentery();
        myTick1.attach(&onTick1, 10);               // if no password input after
10s, call onTick1, Rntery --> Alarm status
        int control_times = 0;
        while(1){
```

```
                /*Since I used a while loop in the function to detect changes in the
environment
                and the refresh frequency was 10ms, I set a counter with +1 for
each refresh. After 100 refresh times,
                the time passed 10*100ms=1s,
                so as to control the flashing of the LED light at the meanwhile.*/

                 if (control_times % 500 == 0){
                       led1 = !led1;
                 }
                keypass();
                if (toggle2 == 1){                      // if sensor detect, turn on led3,
enter Alarm status
                      led3 = 1;
                      Status = 3;
                      break;
                }
                if (Status != 5){
                      break;
                }
                wait_ms(10);
                control_times = control_times + 10;
          }
     }

     void setState(){
          initialset();
          while(1){
                if (toggle1 == 1){                      // if door open, turn on led2,
enter Entery status
                      led2 = 1;
                      Status = 5;
                }
                if (toggle2 == 1){                      // if sensor detect, turn on led3,
enter Alarm status
                      led3 = 1;
                      Status = 3;
                }
                if (Status != 4){
                      break;
                }
                wait_ms(10);
          }
```

```
    }

    void ExitState(){
        initialExit();
        myTick.attach(&onTick, 10);                    // if nothing happen after
10s, call onTick, Exit --> Set
         int control_times = 0;
        while(1){
             if (control_times % 500 == 0){
                  led1 = !led1;
             }
            keypass();
            if (toggle2 == 1){                         // if sensor detect, turn on led3,
enter Alarm status
                led3 = 1;
                Status = 3;
            }
            if (Status != 2){
                break;
            }
            wait_ms(10);
            control_times = control_times + 10;
        }
    }

    void reportState(){
        initialreport();
        while(1){
            keypass1();                                // call special keypass1 used
in Report state
            if (Status != 6){
                break;
            }
            wait_ms(10);
        }
    }

    int main(){
        InitialKeyboard();
        while (1) {
        printf("Current state is %d\n",Status);
        switch(Status){                                // use to change current
status
            case 1:
```

```
                unsetState();
                break;
            case 2:
                ExitState();
                break;
            case 3:
                alarmState();
                break;
            case 4:
                setState();
                break;
            case 5:
                enteryState();
                break;
            case 6:
                reportState();
                break;
            }
            wait_ms(10);
        }
}
```