Smart Home Project

Project Description:

- This project is Smart Home-based Bluetooth where we want to control home appliance like a door, an air conditioner, 5 LEDs, extra Dimming LED remotely using Mobile App via Bluetooth or from a PC terminal or for emergencies controlled by LCD and Keypad.
- In case of emergencies only non-remoted users can access the system by Keypad.
- Two ECU's Communicate with each other the first is a control ECU which takes the input from Bluetooth and send it to the Actuator ECU via SPI to interpret which action should be taken.
- The project consists of two microcontrollers (Atmega32) on is the master and the other one is the salve which are connected via SPI.
- The master microcontroller interface with the user of the system (admin or any user) and sends the data to the salve to control the system (5 on/off lamps, one dimming lamp, door, air-condition).
- The login system consists of admin and users: Admin is remoted only and can access to all applies and user can access to all applies except the door opening.
- Admin mode can also register any user or remove.
- Usernames and password are kept into memory even if the system is powered off via the EEPROM.
- If LCD wasn't in use it operates in the IDLE state and displays the running devices.

Prerequisites:

1. Download Atmel studio or SDK (eclipse) for the code.
2. Download Proteus for the simulation or use the components below for hardware.
3. Download Suitable application to be used as a terminal for Bluetooth Module HC-05 which can be easily downloaded from Google play.

NOTE: You may need to pair with the Bluetooth first then try to connect.

MASTER Code Guide:

To fully understand and follow the proceed of the code take in consider the Program status flag variable as it is what runs the program from state to another and these states are:

1. **FIRST TIME USE STATE:** this state operates only for the first time to register admin username and password using Keypad and LCD then store in the EEPROM that the Program Status Flag is Load login page until reset.

2. **LOAD LOGIN SYSTEM DATABASE STATE:** this state loads the Usernames and Passwords of The ADMIN and Registered Users from the EEPROM to be ready for use then change program state to the IDLE State.

3. **IDLE STATE:** The Program stays in this state as long as the LCD Keypad System and The Remoted System aren't used to display the running devices and changes the program state according to the interrupts caused in case of any login attempt either remoted or by keypad.

4. **REMOTED LOGIN PAGE STATE**: it runs if there was a connection detected by UART and asks for a username and a password then compares the input with the loaded data base if it didn't match any, it shows that the input was wrong and decrease the remaining trials of insertion and if it matches any code goes to the remoted system menu state with respect to the result of match if it was a user or the admin as the menu differs.

5. **REMOTED SYSTEM MENU STATE:** It displays the menu of the Smart Home to the remoted user and displays additional options to the admin only.

6. **LCD_KEYPAD LOGIN PAGE STATE:** Only if the button was pressed the program state changes to this by the EXTI then It takes the username and password from the user and compare it with non-remoted users database.

7. **LCD_KEYPAD MENU STATE**: It is exactly the same as the remoted user only it is displayed on the LCD controlled by Keypad.

SLAVE Code Guide:

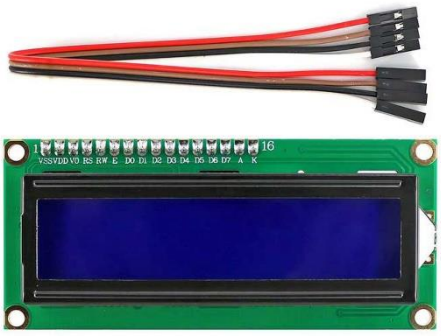
Slave main code is more simple than the master as it just operates according to the data sent by the Master using a switch statement to check the data sent then operates according to it.

Components:

1. Two microcontrollers (ATmega32).
2. LCD 16x2.
3. Keypad 4x4.
4. Bluetooth module HC-05.
5. LM35 temperature sensor.
6. DC motor.
7. Servo motor.
8. 6 LEDs.
9. Transistors like 2N2222.
10. Button.

Specifications:

1. **LCD& keypad:**



They are used to login to system for the first time use to register admin then it is used by a non-remoted user for emergencies by entering username and password on keypad and display it on lcd.
They can control the system even if any user login by remoted mode except the admin himself or a remoted user taking the exception from the admin to access over the lcd-keypad user.

Interface for keypad: 8-pin access to 4×4 matrix

The LCD is used in 4-pins data mode.

16x2 LCD Pinout Configuration

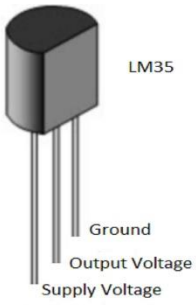
Pin No:	Pin Name:	Description
1	Vss (Ground)	Ground pin connected to system ground
2	Vdd (+5 Volt)	Powers the LCD with +5V (4.7V – 5.3V)
3	VE (Contrast V)	Decides the contrast level of display. Grounded to get maximum contrast.
4	Register Select	Connected to Microcontroller to shift between command/data register
5	Read/Write	Used to read or write data. Normally grounded to write data to LCD
6	Enable	Connected to Microcontroller Pin and toggled between 1 and 0 for data acknowledgement
7	Data Pin 0	Data pins 0 to 7 forms a 8-bit data line. They can be connected to Microcontroller to send 8-bit data. These LCD's can also operate on 4-bit mode in such case Data pin 4,5,6 and 7 will be left free
8	Data Pin 1	
9	Data Pin 2	
10	Data Pin 3	
11	Data Pin 4	
12	Data Pin 5	
13	Data Pin 6	
14	Data Pin 7	
15	LED Positive	Backlight LED pin positive terminal
16	LED Negative	Backlight LED pin negative terminal

2. RC SERVO MOTOR:

• The operating voltage from 4.8 to 6 v (5v is the mostly used)
• Stall current = 0.7 A
• Advantages of RC servo motor:

1. Small
2. Cheap
3. More suitable for the door control job.

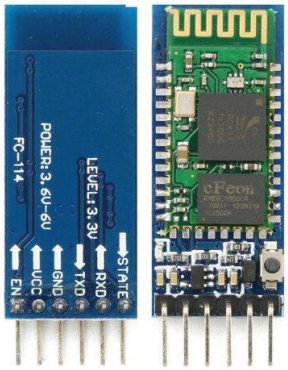
3. LM35 temperature sensor:



Operates from 4 V to 30 V
It is used to control the air conditioner
In the automatic mode.

4. Bluetooth module HC-05:

It is a 6-pin module.
It works on serial communication (USART).
The module works on 5V or 3.3V.
It is used to control the system remotely by admin or a remoted user.



5. DC motor

DC motors normally have just two leads, one positive and one negative
It is used as an air conditioner that
Is controlled by lm35 sensor as there is no need to reverse its direction for example a simple NPN transistor is used to control it depending on the pin signal connected to the microcontroller.

Simple Explanation of Some used Functions:

The master controls the LCD, KEYPAD and UART used module (TTL or HC-05).

The slave controls all the other HAL layer presented in the home applies.

MCAL layer in master:

1. DIO:
   ES_t DIO_enuSetPinDirection(u8 Copy_u8PortID , u8 Copy_u8PinID , u8 Copy_u8Direction)
   In this function it returns an error state to detect the error and it take the port id and pin id you want to set the direction of and sets the direction to be input or output

2. EEPROM:
   void EEPROM_vWriteByteToAddress(const u16 uiAddress, const u8 uiData);
   this function writes a bite into the address you want so it takes two parameters one is the address the second is the that in byte you want to write in EEPROM.

3. SPI:
   u8 SPI_ui8TransmitRecive (u8 Copy_u8Data);
   this function is used to send or receive data between the two microcontrollers so it takes one input that is the data to send in case of master is sending and returns the received data in case of receiving from slave

4. UART:
   ES_t UART_enuSendChar(u8 Copy_u8CharData);
   This function returns an error state to detect the error it is used to send a character by UART communication protocol so it takes one parameter the char data you want to sent

In MCAL layer in slave:

1. ADC:
   int ADC_Read(char channel);
   this function reads the digital value from ADC channel and returns the ADC read so it takes one parameter that is the channel number.

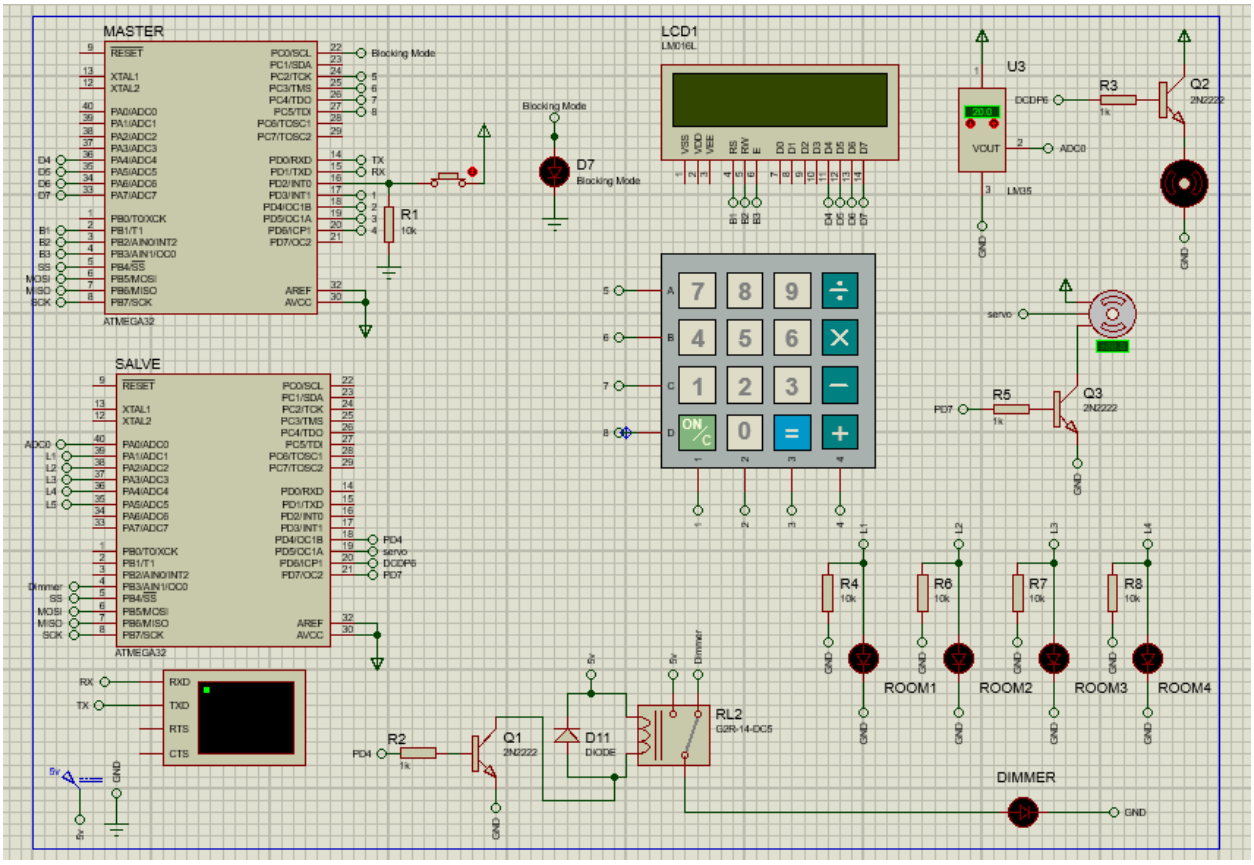
2. Timer1:
   ES_t Timer_PWM(float Copy_u16Freq ,float Copy_u8DutyCycle);
   This function generates a timer with the frequency and duty cycle you want so it returns an error state to detect the error and takes two parameters the frequency and duty cycle values

In HAL layer:

1. KEYPAD:
   ES_t Keypad_enuGetPressedKey(u8 * Copy_pu8KeyValue);
   This function gets the pressed value on keypad and it returns an error state to detect the error

2. LCD:
   ES_t LCD_enuSendData(u8 Copy_u8Data);
   This function is used to display a character on the lcd so it takes the character to be displayed and returns an error state to detect the error

The Connection of components on proteus:



Link of GitHub for the project:

https://github.com/hamdy24/Smart-Home-Project