Team B
[Abdelrhman Walaa - Moamen Mamdouh - Mohamed Ahmed]

SHS System Design

Outline

Smart Home System is an application that *Controls Home Temperature* in real-time, and displays it on the LCD. The system also *Controls Home Lighting System.*

This application allows the user to *Access Doors* usings password, if the password is wrong, after a certain number of trials it'll *Fire a Door Alarm*, and *Send a Notification* on mobile.

System Design - Hardware Design

Block Diagram

A *Block Diagram* (*Figure 1*) is a specialized flowchart used to visualize systems and how they interact.

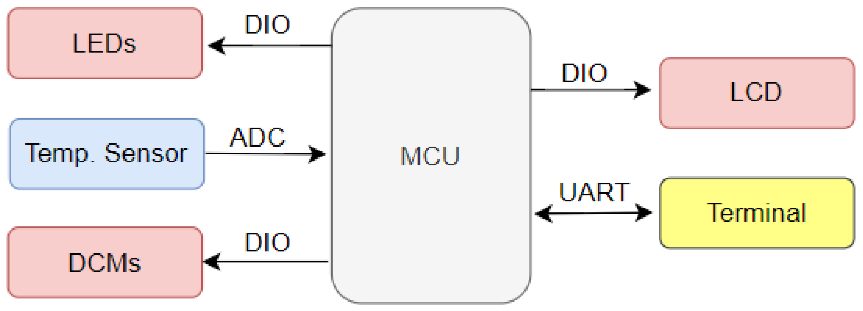


Figure 1. Block Diagram

System Input: Blue | *System Output*: Red | System Input/Output: Yellow

Block Diagrams give you a high-level overview of a system so you can account for major system components, visualize inputs and outputs, and understand working relationships within the system.

System Design - Software Design

Layered Architecture

Layered Architecture (*Figure* 2) describes an architectural pattern composed of several separate horizontal layers that function together as a single unit of software.

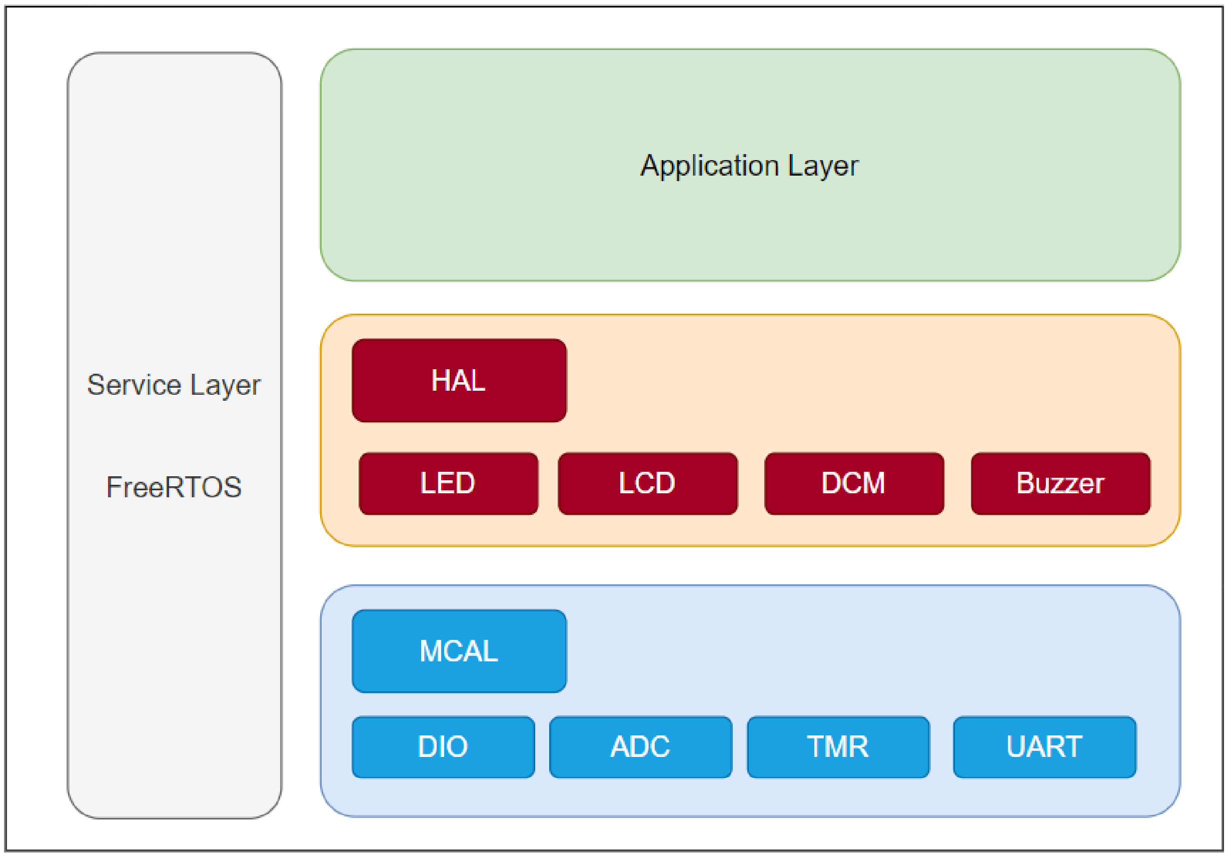


Figure 2. Layered Architecture

Task Design

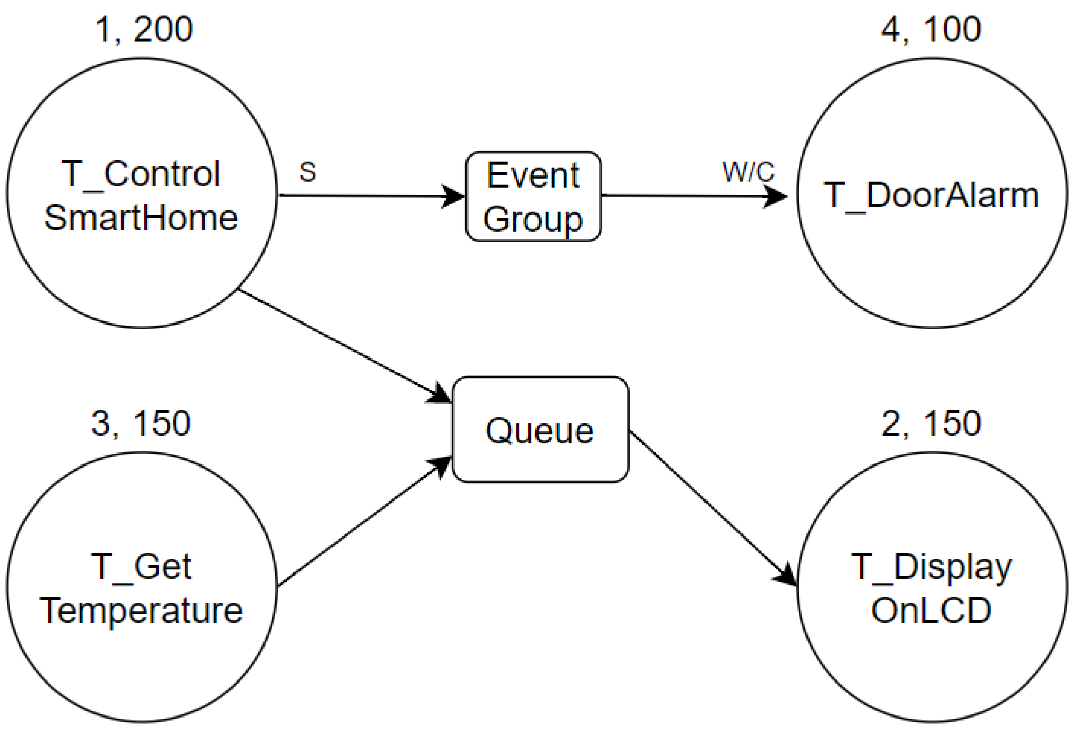


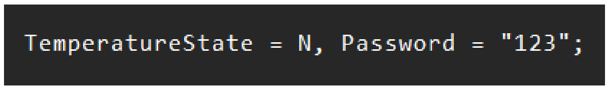
Figure 3. Task Design

Component Level Design

Component Level Design is the definition and design of components and modules after the architectural design phase.

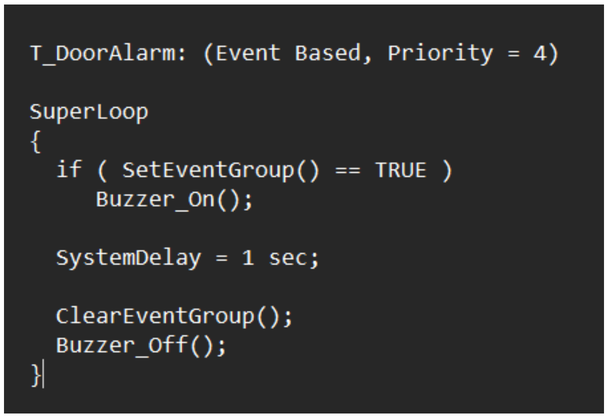
It defines the data structures, algorithms, interface characteristics, and communication mechanisms allocated to each component for the system development.

First of all, we define *Global Variables*.



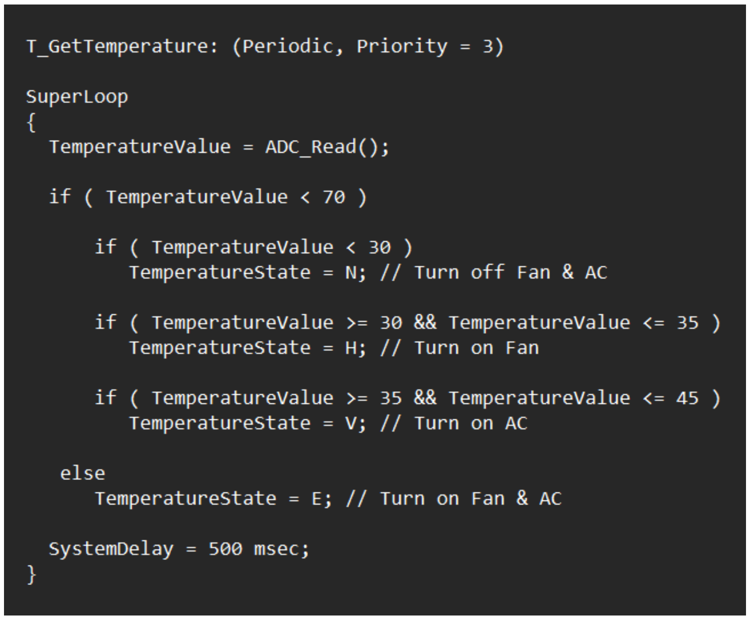
```
TemperatureState = N, Password = "123";
```

Then, identifying the door alarm task "*T_DoorAlarm*", it is an *Event Based* task, and with highest *Priority* equals to 4.



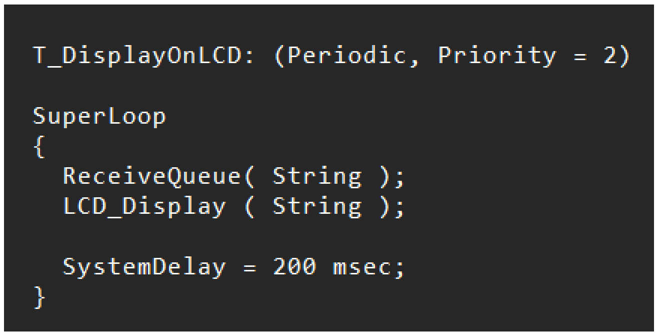
```
T_DoorAlarm: (Event Based, Priority = 4)

SuperLoop
{
  if ( SetEventGroup() == TRUE )
     Buzzer_On();

  SystemDelay = 1 sec;

  ClearEventGroup();
  Buzzer_Off();
}
```

After that, we create a *Periodic* task to get the temperature ”T_GetTemperature” in real-time, its *Priority* equals to 3.



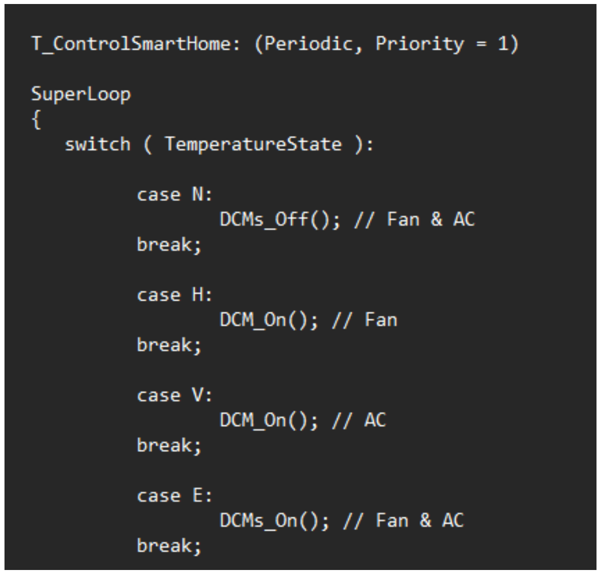
```
T_GetTemperature: (Periodic, Priority = 3)

SuperLoop
{
  TemperatureValue = ADC_Read();

  if ( TemperatureValue < 70 )

      if ( TemperatureValue < 30 )
         TemperatureState = N; // Turn off Fan & AC

      if ( TemperatureValue >= 30 && TemperatureValue <= 35 )
         TemperatureState = H; // Turn on Fan

      if ( TemperatureValue >= 35 && TemperatureValue <= 45 )
         TemperatureState = V; // Turn on AC

   else
      TemperatureState = E; // Turn on Fan & AC

  SystemDelay = 500 msec;
}
```

Then, we implement a *Periodic* task which displays different system modes on LCD “*T_DisplayOnLCD*”, it has *Priority* equals to 2.

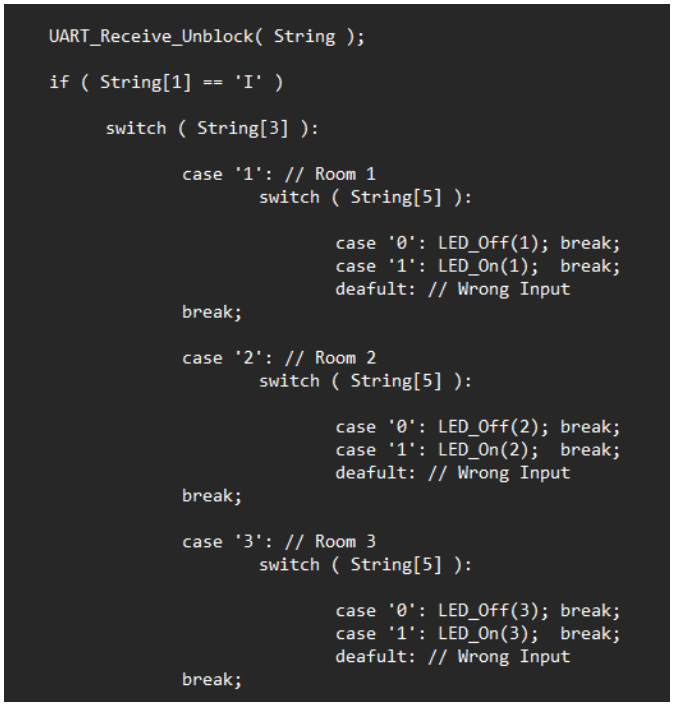


```
T_DisplayOnLCD: (Periodic, Priority = 2)

SuperLoop
{
  ReceiveQueue( String );
  LCD_Display ( String );

  SystemDelay = 200 msec;
}
```

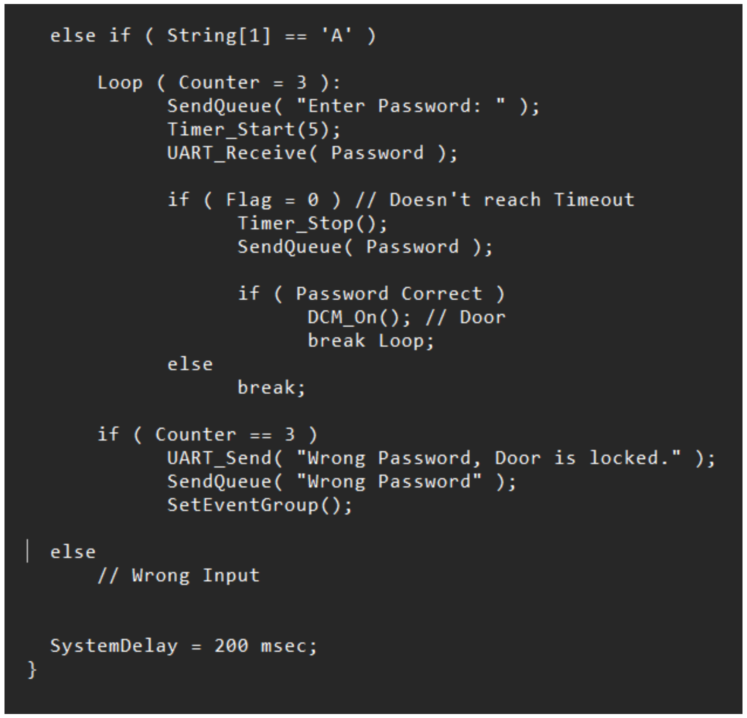
Last but not least, this pseudocode describes a *Periodic* task to control and configure the system “*T_ControlSmartHome*”, with the least *Priority* equals to 1.



```
T_ControlSmartHome: (Periodic, Priority = 1)

SuperLoop
{
   switch ( TemperatureState ):

         case N:
                DCMs_Off(); // Fan & AC
         break;

         case H:
                DCM_On(); // Fan
         break;

         case V:
                DCM_On(); // AC
         break;

         case E:
                DCMs_On(); // Fan & AC
         break;
```



```
UART_Receive_Unblock( String );

if ( String[1] == 'I' )

     switch ( String[3] ):

            case '1': // Room 1
                   switch ( String[5] ):

                          case '0': LED_Off(1); break;
                          case '1': LED_On(1);  break;
                          deafult: // Wrong Input
            break;

            case '2': // Room 2
                   switch ( String[5] ):

                          case '0': LED_Off(2); break;
                          case '1': LED_On(2);  break;
                          deafult: // Wrong Input
            break;

            case '3': // Room 3
                   switch ( String[5] ):

                          case '0': LED_Off(3); break;
                          case '1': LED_On(3);  break;
                          deafult: // Wrong Input
            break;
```



```
  else if ( String[1] == 'A' )

      Loop ( Counter = 3 ):
            SendQueue( "Enter Password: " );
            Timer_Start(5);
            UART_Receive( Password );

            if ( Flag = 0 ) // Doesn't reach Timeout
                  Timer_Stop();
                  SendQueue( Password );

                  if ( Password Correct )
                        DCM_On(); // Door
                        break Loop;
            else
                  break;

      if ( Counter == 3 )
            UART_Send( "Wrong Password, Door is locked." );
            SendQueue( "Wrong Password" );
            SetEventGroup();

  else
      // Wrong Input


  SystemDelay = 200 msec;
}
```

References:

1. *diagrams.net*
2. *Block Diagram Maker | Free Block Diagram Online | Lucidchart*
3. *Layered Architecture | Baeldung on Computer Science*
4. *SOFTWARE ENGINEERING COMPONENT LEVEL DESIGN Component level design is the definition and design of components and modules after*