Home Security System

Real Time Embedded Systems

Vaishnavi Nandedkar (vn549)
Nikunj Rajput (nr1536)

Introduction :

From many recent surveys it can be seen that, almost 70 percent fatal incidences taking place inside house, like fire, gas leakage, theft has been happening when no one is home. As it is not possible for a person to stay home every time , we can develop a technology which would alert the person about these incidences.

Idea of this project is to develop a system which would alert the user through an alarm.

The home security system would have three main parts

1. Fire Detection
2. gas leakage detection
3. theft control.

For fire detection we would interface a temperature detector with the STM32 board. So whenever the sensor gives a positive output the board will actuate a signal which would be used by the alarm to alert the user.
Similarly for gas leakage we will use a gas detector with whose output the controller would actuate the alarm to alert the user. Also for the theft control part of the security system we will use a proximity sensor which, on having someone around, would give a signal to the controller. Based on this signal the controller would turn on the alarm.

Components :

1. STM32F4 Discovery Board

2. Sensors :

• Temperature Sensor (MCP9700A-E/TO) for fire detection.
(Datasheet — https://www.sparkfun.com/datasheets/DevTools/LilyPad/MCP9700.pdf)

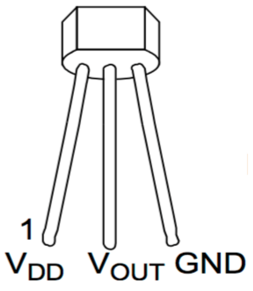
• Gas Sensor (MQ5) for gas detection.
(Datasheet — https://www.seeedstudio.com/depot/datasheet/MQ-5.pdf)

• Proximity Sensor (GP2Y0A21YK0F Sharp IR).
(Datasheet — http://www.sharpsma.com/Webfm_Send/1208)

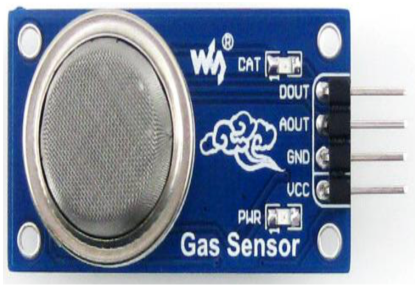
<u>Design</u> :

The following diagram shows pin configurations of each sensor :

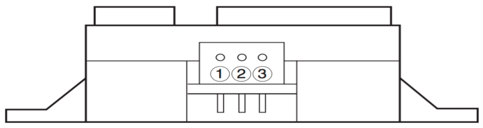
Temperature sensor



Gas Sensor

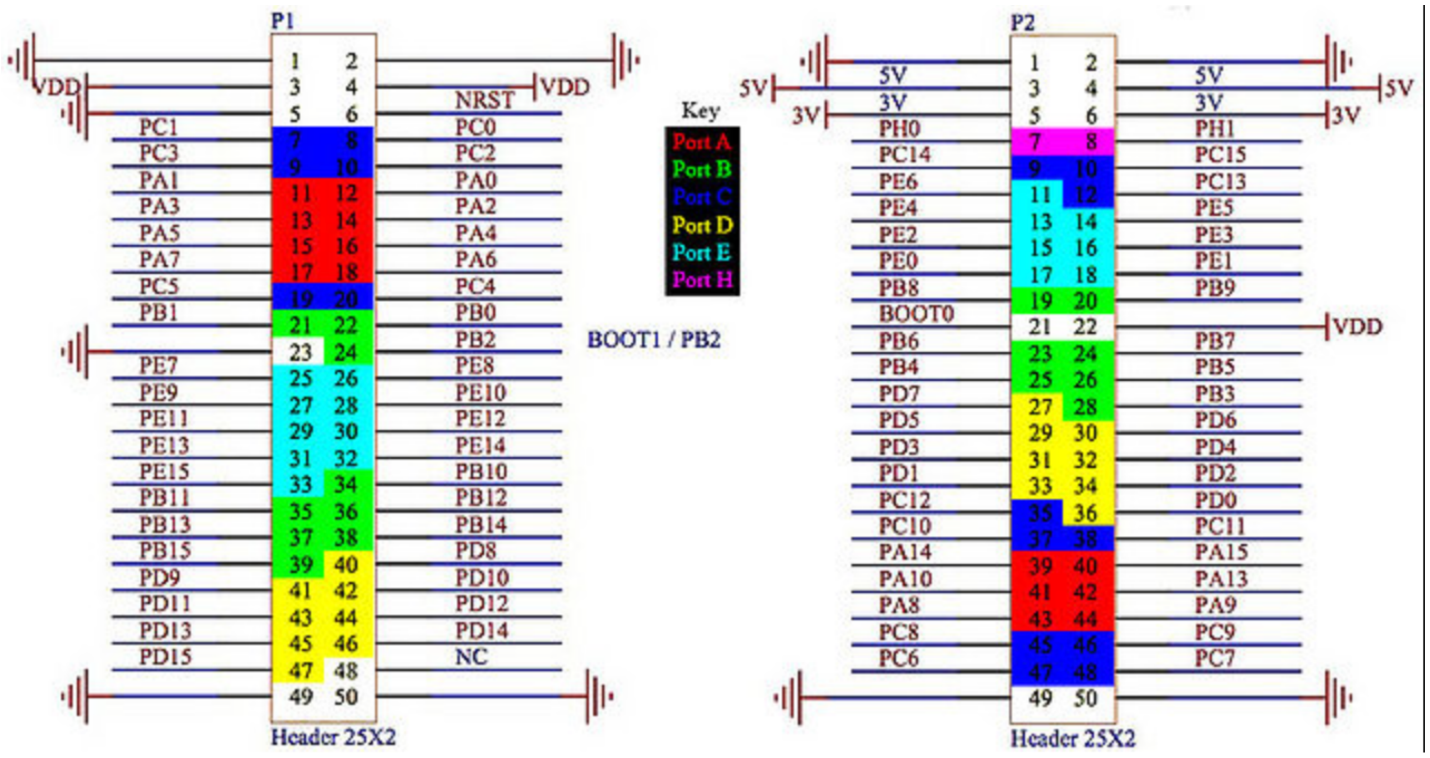


Proximity Sensor



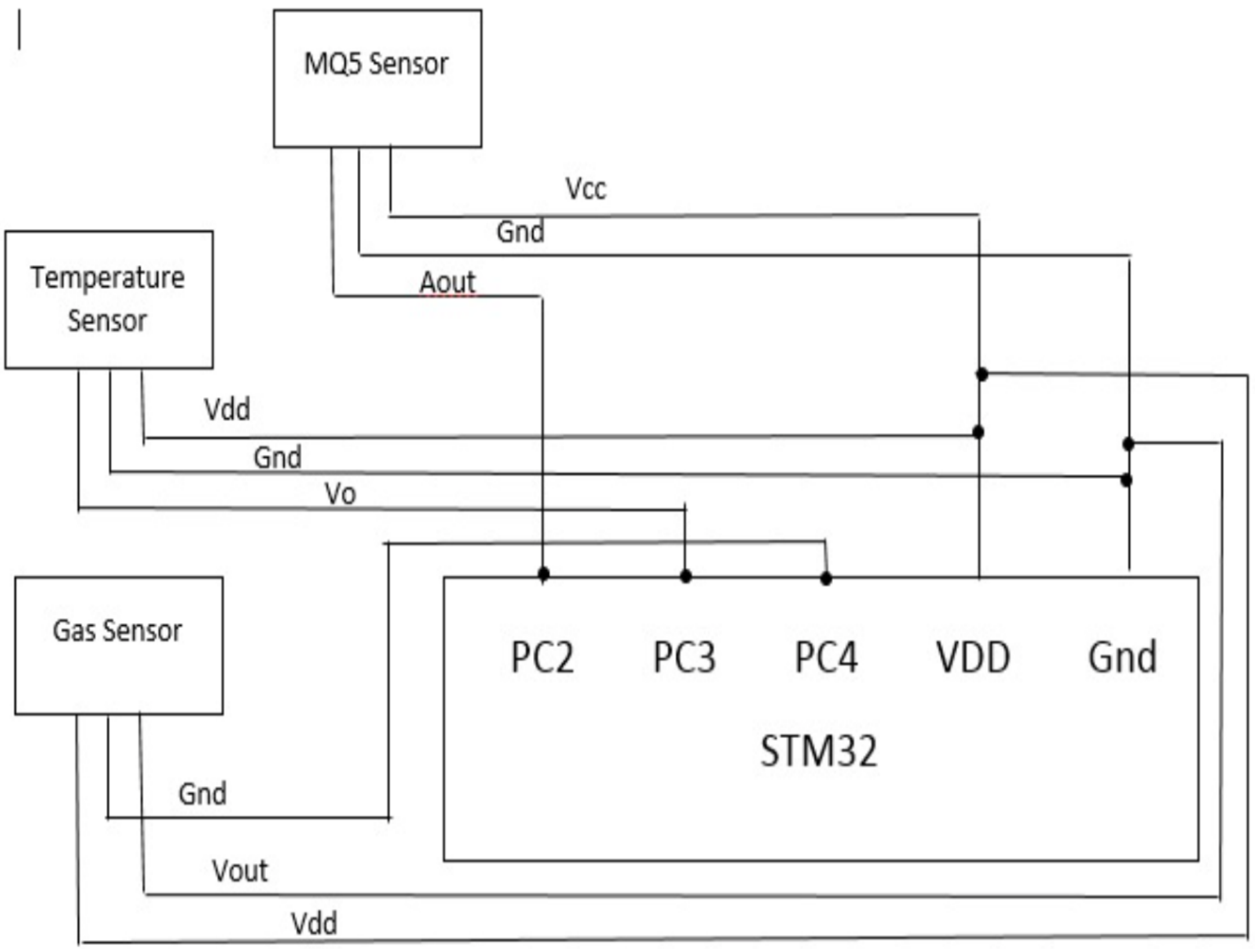
PIN	SIGNAL NAME
①	V_O
②	GND
③	V_{CC}

STM32 Pin Configuration :



Pin Connections :
PC2 is connected to Aout of MQ5 (gas sensor) used as Channel_12.
PC3 is connected to Vo of Proximity Sensor used as Channel_13.
PC4 is connected to Vout of Temperature Sensor used as Channel_14.

Circuit Diagram :



Working :

As shown in the circuit diagram above, the output pins of the sensors are connected to the STM32 board GPIO pins. We have configured the pins as ADC channels,a s the output data coming out from the sensor is analog data and to process it we have to use its digital format. Hence we are using ADC to convert Analog Sensor data to digital values.

Gas Sensor :

The gas sensor is connected to pin PC2 which we have configured as the Channel_12 for ADC 1. MQ5 senses Methane in the atmosphere and gives values in PPM. The algorithm of this system is such that whenever the value converted by the ADC, read from MQ5 is greater than some threshold value, then the user will be alarmed. We have collected a set of values from the readings of the gas sensor in atmosphere with controlled amount of methane and then we have taken a different set of values in the atmosphere with increased level of methane. From these values we decided one threshold depending on which the alarm will be turned on.
The values taken are as below :

With controlled amount of methane :
0.029, 0.025, 0.0316, 0.024, 0.0264, 0.03009, 0.0327, 0.0255

With increased level of methane :
0.045, 0.048, 0.056, 0.064, 0.0653, 0.0655, 0.067, 0.059

From these values we decided to take reference value as 0.035.

Temperature Sensor :

The temperature sensor is connected to pin PC4 which we have configured as the Channel_14 for ADC 1. This sensor senses the temperature values and provides the analog values to channel number 14 of ADC 1. Then these analog values are converted to digital values for processing. Used the calibration procedure to take the values of temperature in degree Celsius. Depending on the values noted at Room temperature and higher temperature we have decided a reference temperature above which the system will alarm the user.

The values taken are as below :

At Room temperature :
0.253968, 0.253968, 0.253724, 0.253968, 0.253724

At higher temperature :
0.256899, 0.257143, 0.257143, 0.257143, 0.257387

From these values we decided to take reference value as 0.2560.

Proximity Sensor :

The Proximity sensor is connected to pin PC3 which we have configured as the Channel_13 for ADC1. This sensor which we are using is an IR sensor, which sends waves and based on the reflected values it gets it gives output in terms of voltage. We have taken the reference voltage to be 0.300. Whenever the measured voltage is greater than the reference voltage the system alerts the user.

Code Blocks :

Sensor and ADC Initialization function :

```
   // initialize the gas sensor
   void init_gas_sensor(){
   ADC_InitTypeDef ADC_InitStruct;
   ADC_CommonInitTypeDef ADC_CommonInitStruct;

   // RCC Initialization
   RCC_APB2PeriphClockCmd(RCC_APB2Periph_ADC1, ENABLE);

   // Common ADC Initialization
   ADC_CommonInitStruct.ADC_Mode = ADC_Mode_Independent;  //ADC works in Independent mode
   ADC_CommonInitStruct.ADC_Prescaler = ADC_Prescaler_Div8; //Set the prescaler
   ADC_CommonInitStruct.ADC_DMAAccessMode = ADC_DMAAccessMode_Disabled;
   ADC_CommonInitStruct.ADC_TwoSamplingDelay = ADC_TwoSamplingDelay_5Cycles;
   ADC_CommonInit(&ADC_CommonInitStruct);

   ADC_InitStruct.ADC_Resolution = ADC_Resolution_12b;
   ADC_InitStruct.ADC_ScanConvMode = DISABLE;
   ADC_InitStruct.ADC_ContinuousConvMode = ENABLE;  //ADC Works in continuous mode
   ADC_InitStruct.ADC_ExternalTrigConvEdge = ADC_ExternalTrigConvEdge_None; //No trigger
   ADC_InitStruct.ADC_ExternalTrigConv = ADC_ExternalTrigConv_T1_CC1;
   ADC_InitStruct.ADC_DataAlign = ADC_DataAlign_Right;
   ADC_InitStruct.ADC_NbrOfConversion = 3; //Number of convergence
   ADC_Init(ADC1, &ADC_InitStruct);


   // Enable ADC
   ADC_Cmd(ADC1, ENABLE);

   // ADC Channel 1 Configuration
   ADC_RegularChannelConfig(ADC1, ADC_Channel_gas, 1, ADC_SampleTime_144Cycles);//Channel
                                                                     //number and configuration
}
```

The above code shows the initialization of gas sensor.
Similarly we have two more blocks for temperature sensor and proximity sensor with their respective channel number in the ADC_RegularChannelConfig function.

Read sensor values :

```
float read_gas_sensor(){
   float gas;  // gas reading

   ADC_SoftwareStartConv(ADC1);  //Start the conversion

   while (ADC_GetFlagStatus(ADC1, ADC_FLAG_EOC) == RESET);  //Wait for ADC conversion to finish
   gas = (float)ADC_GetConversionValue(ADC1);  //Get one ADC reading -- as a number from 0 to 4095 (12
                                                                   // bits)

   // Convert ADC reading to voltage
   gas /= 4095.0f;
   return gas;
}
```

Similarly we have two more blocks for temperature sensor and proximity sensor with their respective parameters.

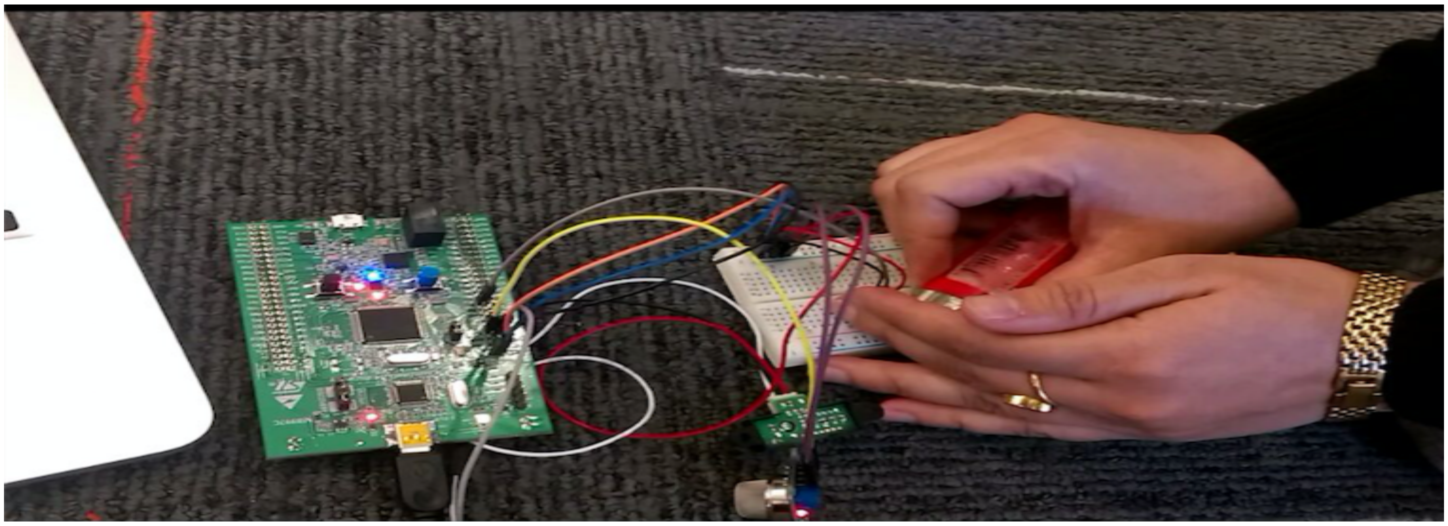
Main Loop :

```
 while (1)
   {
      msTicks_var = msTicks;
      SystemInit();
      float gas1 = read_gas_sensor(); // Reads the value from gas sensor ADC channel

      SystemInit();

      if (gas1 > 0.0350) //checks with reference value for gas
      {


         if(msTicks_var - msTicks_var1>=8000) //checks for initial delay
         {
            msTicks_var1 = msTicks;
            msTicks_var2 = 0;
            GPIOD->BSRRL |= (0x1 << (1*12)); //turns on the alarm

          }
       }

      msTicks_var = msTicks;
      SystemInit();
      float temperature1 = read_temperature23_sensor(); //Reads the value from temperature sensor ADC
                                                                   //channel
      SystemInit();
      if (temperature1 > 0.2650)//checks with reference value for temperature
      {
        if(msTicks_var - msTicks_var3>=2000) //checks for initial delay
        {
            msTicks_var3 = msTicks;
            msTicks_var4 = 0;

            GPIOD->BSRRL |= (0x1 << (1*13));//turns on the alarm

            GPIOD->BSRRL |= (0x1 << (1*14));   //turns on the alarm

        }

    }

    msTicks_var5 = msTicks;
    SystemInit();

    float distance = read_proximity_sensor();//Reads the value from Proximity sensor ADC
                                              //channel


    SystemInit();
    if (distance > 0.0300)//checks with reference value for temperature
    {
        if(msTicks_var - msTicks_var5>=2000)//checks for initial delay
        {
            msTicks_var5 = msTicks;
            msTicks_var6 = 0;
            GPIOD->BSRRL |= (0x1 << (1*15));//turns on the alarm
        }
    }

}
```

Results :

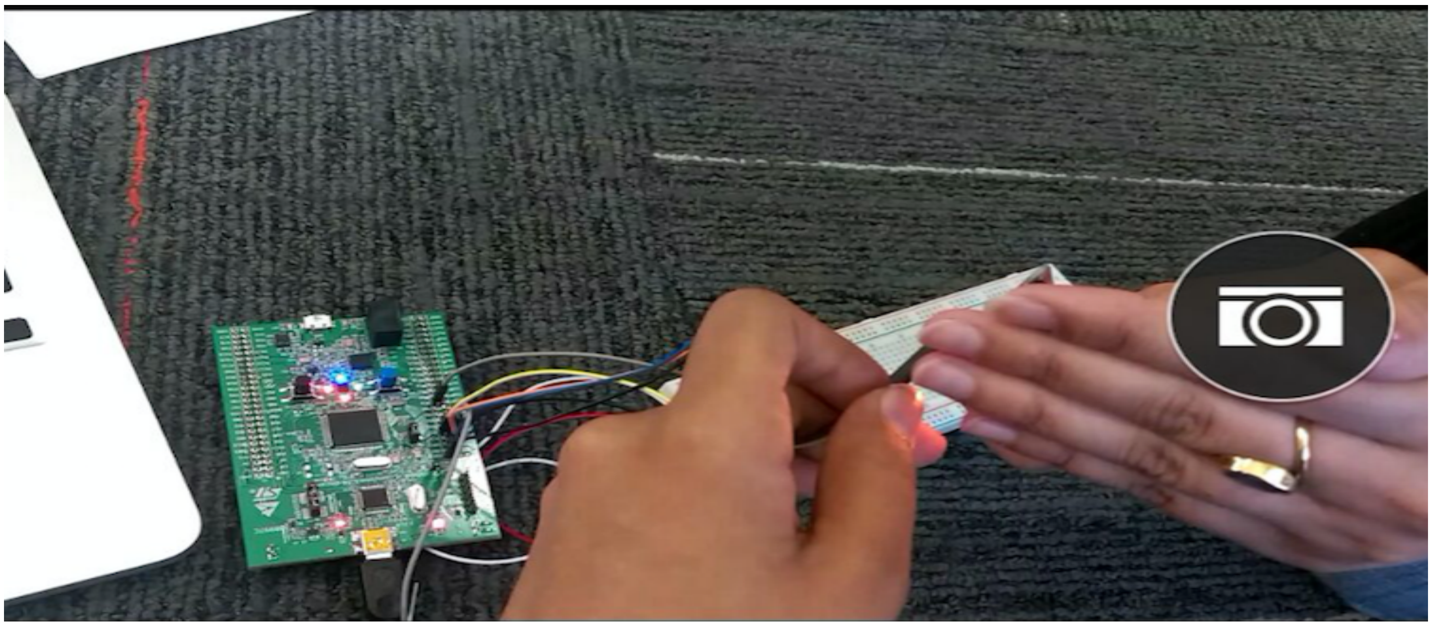
Working of Temperature sensor and Proximity Sensor :

The two red LEDs are showing that the temperature sensed by the temperature sensor is above reference temperature value.



Working of Gas Sensor :

The green LED shows that the methane gas value sensed by the gas sensor is above the threshold value.



Summary :

Designed a home security system based on three sensors, temperature sensor, gas sensor and proximity sensor with the help of ARM Cortex processor on STM32 board and coded using Embedded c. The code has been compiled using GCC compiler and then taken the results based on run time values.

Acknowledgments :

Our sincere thanks to professor Prashanth Krishnamurthi in Embedded applications and the TAs for their help and support.

References :

Datasheet temperature Sensor — https://www.sparkfun.com/datasheets/DevTools/LilyPad/MCP9700.pdf
Datasheet Gas Sensor — https://www.seeedstudio.com/depot/datasheet/MQ-5.pdf
Datasheet Proximity Sensor — http://www.sharpsma.com/Webfm_Send/1208
STM32 Reference manual — http://www.st.com/web/en/resource/technical/document/reference_manual/DM00031936.pdf