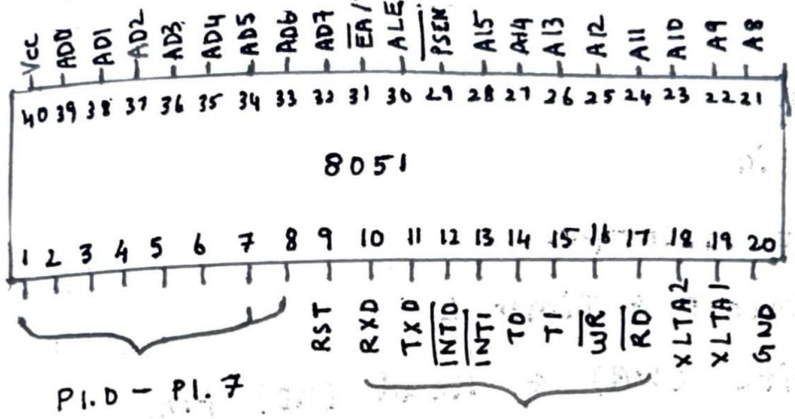
RAHUL KARTHIK.S

BECE320E - Embedded C Programming 21BEC1851

Digital Assignment - II

① P0.0 - P0.7 P2.0 - P2.7



P1.0 - P1.7

P3.0 - P3.7

★ 8051 Microcontroller have 40 Pins.

★ Pin 40 : V_{cc} is the power source. Usually +5V DC.

★ Pin 32 - 39 : P0.7 - P0.0. In addition to that, it also acts as lower order address bus, by multiplexing.

★ Pin 31 : External Access is used to allow or disallow external memory interfacing.

★ Pin 30 : Address Latch Enable. When ALE = 1 : Pins acts as Data bus, ALE = 0 : Pins acts as Address bus.

★ Pin 29 : Program Store Enable is an active low Pins. It is used along EA pin for external ROM.

★ Pin 21 - 28 : P2.7 - P2.0. In addition to that, it also acts as higher order address bus, by multiplexing.

★ Pin 20 : Ground

★ Pin 18, 19: Used to interface with external Crystal to provide clock.

★ Pin 10 – 17 : P3.0 - P3.7 I/O Pins. In addition to that,

P3.0 - Receiver Pin

P3.1 - Transmit Pin

P3.2, P3.3 - Ext. Interrupts

P3.4, P3.5 - Timer 0 & Timer 1

P3.6, P3.7 - Write (WR) & Read (RD) Pins

★ Pin 9 : Reset, It is used to reset to initial values.

★ Pin 1 - 8 : P1.0 - P1.7

② ★ SFR. Register Exists from 80H – 0FFH.

i. A or Accumulator Register :*

★ Address : 0E0H

★ Used to hold data for ALU operations.

ii. B Register :*

★ Address : 0F0H

★ Used for Multiplication and Division operation.

★ Multiplication : B is one of the operand and stores higher byte of result.

★ Division : B is divisor and it stores remainder

iii. Program Status Word Register :*

★ Contains Flag bits and Used to check the

condition of the result.

iv. Data Pointer:

★ Combination of two - 8 bit registers namely DPL and DPH.

★ Address: DPL = 82H, DPH = 83H

★ Used for addressing external memory.

v. Stack Pointer:

★ Points out top of the stack, and it indicates the next data to be accessed.

★ SP can be accessed using PUSH, POP, CALL, RET.

★ SP is 8-bit register, Address: 81H

vi. IO Ports:*

★ Ports P0, P1, P2, P3 can be used as Input / output Port.

Address:

★ P0 - 80H, P1 - 90H, P2 - 0A0H, P3 - 0B0H

vii. Power Control:

★ PCON is used to control 8051 μc Power Modes.

★ Using two bits in PCON Register, μc can be set to Idle Mode or Power down mode.

★ Address: 87H

viii Serial Control*:

* SCON used to control serial port.
* Using this, we can control operation Modes of Serial Port, Baud Rate, Send or Receive Data using Serial Port.
* Address: 98H

ix. Timer Control*:

* TCON used to start or stop the timers of 8051 μc.
* It has bits to indicate overflow.
* Address: 88H

x. Timer Mode:

* TMOD used to set operation modes of Timers T0, T1.
* Four bits used to configure Timer 0 & Timer 1.
* Address: 89H

xi. Interrupt Enable*:

* IE used to enable or disable individual interrupt.
* Address: 0A8H

xii. Interrupt Priority:*

* IP used to set priority for interrupt as low or high.
* If a bit is cleared, interrupt is assigned

low priority, if bit is set interrupt assigned to high priority.

* Address : 0B8H.

Note : * : Bit-Addressable

③

i. Timer 0 Overflow Interupt :

* Can be triggered by low-level signal or falling edge (configurable through TCON).
* This triggered by an external event on Pin3.2. It's used to handle external hardware signal such as buttons or sensors.
* Address : 0003H

ii. External Interrupt 1 (INT1) :

* Similar to INT0, it can be triggered by a low-level signal or falling edge on P3.3.
* used for external event handling.
* Address : 0013H

iii. Timer 0 Interrupt :

* Triggered when Timer 0 overflows.
* Timer 0 generates an interrupt when it overflows, allowing the CPU to handle periodic tasks, like generating delays or time-keeper.
* Address : 000BH

iv. Timer Interrupt 1:

★ Triggered when Timer 1 overflows.

★ Timer 1 generates an interrupt when it overflows, allowing periodic tasks to be handled in time-based manner.

★ Address: 001BH

v. Serial Communication Interrupt:

★ RI is ~~used~~ set when a character is received in serial port.

★ TI is set when a character finishes transmission.

★ Address: 0023H

④

i. Byte-Addressable Programming:

★ Here, data is accessed and manipulated in order of 8 bits or a byte.

★ Every register and memory location has unique byte address. E.g.: 0x30H refers to specific byte in memory.

★ It include loading, storing, manipulating full bytes of data.

ii. Bit-Addressable Programming:

★ 8051 has special region of its RAM, that is bit-addressable, that is each bits

can be accessed rather than a whole byte at once.

* It is used when dealing with status flags, control bits, or specific conditions that require manipulation of individual bits.
* The memory location range from 20H to 02FH in the RAM and bit-addressable SFR.
* SETB, CLR commands are used for bit level operations.

⑤

```
#include <reg51.h>
void delay ();
void main (){
    unsigned char i;
    while (1) {
        for (i = 0x00; i <= 0xFF ; i++) {
            P0 = i;
            delay ();
        }
    }
}
void delay (){
    unsigned int j;
    for (j=0; j< 30000; j++);
}
```

RAHUL KARTHIK.S
21BEC1851

⑥

Given, Crystal Frequency = 11.0592 MHz

Timer Frequency = $\frac{\text{Crystal Frequency}}{12}$

= 11.0592 MHz / 12

= 921.6 kHz

Time for one cycle = 1/921.6 = 1.085 μs

i. Mode 1

★ Here, timer operates as 16-bit timer, that can count from 0x0000H to 0xFFFFH (65535 in decimal).

★ Steps to Calculate:

→ Calculate the number of machine cycles.

Machine Cycles = $\frac{t}{1.085\ \mu s}$

→ Initial timer count = 65536 − Machine Cycles

→ Convert this value into two 8-bit values, and load one for THx - TLx.

★ E.g:

5 ms delay is required.

Machine Cycles = $\frac{5\ ms}{1.085\ \mu s}$ = 4608.295

≈ 4608 cycles

Initial Timer Count = 65536 - 4608

= 60298 in decimal

= EDOOH in hexadecimal

THx = EDH TLx = OOH

ii. Mode 2:

* Here, timer operates as 8-bit timer, OOH - FFH (255 in decimal).
* It automatically reloads to initial value specified by user.
* Steps to Calculate:

→ Machine cycles = $\frac{t}{1.085\ \mu s}$

→ Reload value = 256 − Machine Cycles

→ Convert this value and load into THx and TLx.

E.g:

5 ~~ms~~ μs delay is required.

→ Machine Cycles = $\frac{5\ \mu s}{1.085\ \mu s}$ = 4.608 ≈ 5 cycles

→ Reload Value = 256 - 5 = 251 = FBH

Load FBH into THx and TLx.

RAHUL KARTHIK.S
21BEC1851

⑦

Target Frequency = 100 Hz

Crystal Frequency = 11.0592 MHz

Mode 1, Timer 0

$$\text{Delay} = \frac{1}{\text{Frequency}} = 10 \text{ ms}$$

$$\text{Half-cycle delay} = \frac{10 \text{ ms}}{2} = 5 \text{ ms}$$

$$\text{Machine cycles} = \frac{5 \text{ ms}}{1.085 \ \mu s} = 4608.295$$

$$\approx 4608 \text{ cycles}$$

Initial Timer Value = 65536 − 4608

= 60298

= EDOOH

TH0 = EDH

TL0 = 00H

Code:

```
#include <reg51.h>
sbit cpin = P1^0;
void delay();
void main(){
    TMOD = 0x01;
    cpin = 0;
```

```
while(1){
    cpin = ~cpin;
    delay();
  }
}
void delay(){
   TH0 = 0xED;
   TL0 = 0x00;
   TR0 = 1;
   while(TF0 == 0);
   TR0 = 0;
   TF0 = 0;
}
```

(8)

* Timers can also be used as counters to count external event.
* Timers increments based on internal clocks whereas counter increments on external event / pulses to a pin.
* To use timer as counter, you need to configure the timer and apply external pulses to T0 (P3.4) and T1 (P3.5) for timer 0 and timer 1 respectively.

RAHUL KARTHIK.S
21BEC1851

☆ For illustration, let's consider Timer 0 as counter.

☆ Code:

```
#include <reg51.h>
void main(){
    TMOD = 0x05;
    TH0  = 0x00;
    TL0  = 0x00;
    P2   = 0x00;
    TR0 = 1;
    while (1) {
        P2 = TL0;
    }
}
```

⑨

```
# include < reg51.h>
sbit cpin = P2^7;
void main () {
    TMOD = 0x20;
    TH1 = 0x 48;
    TR1 = 1;
    cpin = 0;
    while (1) {
        while ( TF1 == 0);
        TF1 = 0;
        cpin = ~ cpin;
    }
}
```

RAHUL KARTHIK.S
21BEC1851

★ Here,

Period = 1 / 2500 Hz = 400 µs

Half - Period = 400/2 = 200 µs

Machine Cycle = 200 µs / 1.085 ≈ 184

Reload Value = 256 − 184 = 72 = 0X48

(10)

★ Here,

SCON = 0x50

TH1 = 0x FD (for Baud Rate 19200)

★ Code :

```
# include < reg51.h>
void trans (unsigned char mess);
void main (){
        int i;
  unsigned char m[] = "United we stand.";

  TMOD = 0X20;
  TH1 = 0XFD;
  SCON = 0x50;
  while (1) {
     i = 0;
     while ( m[i] != '\0') {
        trans ( m[i]);
     }
  }
}
```

RAHUL KARTHIK.S
21BEC1851

```
void seri trans (unsigned char mess){
    SBUF = mess;
    while (TI == 0);
    TI = 0;
}
```

(11)

★ Machine cycles = $\frac{70\ ms}{1.085\ \mu s}$ = 64516 cycles

Initial Value = 65536 − 64516 = 1020

= 03FCH

TH1 = 03H , TL1 = FCH

★ Code:

```
#include <reg51.h>
sbit LED = P2^3;

void tim1 ();
void ext_int ();

void main (){
    P0 = 0x00;
    LED = 0;

    tim1();
    ext_int();

    EA = 1;

    while (1);
}
```

RAHUL KARTHIK.S
21BEC1851

```
void tim1 () {
    TMOD = 0x10;
    TH1 = 0x03;
    TL1 = 0xFC;
    ET1 = 1;
    TR1 = 1;
}
void ext_int () {
    P0 = P1;
}
```

(12)

★ SCON = 0x50

Timer 1 Mode 2

Baud Rate:

TH1 = 0xFD (19200)

TH1 = 0xF4 (4800)

★
```
#include <reg51.h>
sbit SW = P1^3;
void serial_init (unsigned char baud_rate);
void serial_transmit ( unsigned char m);
void main () {
    unsigned char mess[] = "VIT-Chennai";
    int i;
    while (1) {
        if (SW==0) {
```

RAHUL KARTHIK.S
21BEC1851

```
            serial_init (19200);
        }
        else{
            serial_init (4800);
        }
        i = 0;
        while (mess[i] != '\0') {
            serial_transmit (mess[i]);
            i++;
        }
    }
}
void serial_init (unsigned char baud_rate)
{
    TMOD = 0x20;
    if (baud_rate == 19200) {
        TH1 = 0xFD;
    }
    else if (baud_rate == 4800){
        TH1 = 0xF4;
    }
    SCON = 0x50;
    TR1 = 1;
}
void serial_transmit (unsigned char m){
    SBUF = m;
    while (TI==0);
```

```
    TI = 0;
}
```

Odd registration number:

①

★ A wearable patch for monitoring biomarkers such as glucose, pH, electrolytes, cortisol, urea from sweat consists of key components: sensors, microcontroller, signal processing and wireless communi-cation.

i. System Schematic:

★ Sweat collection: Microfluidic channel or sweat-absorbing material.

★ Sensors:

→ Glucose sensors: Enzymatic or electro-chemical sensor.

→ pH Sensors: Electrochemical sensor

→ Electrolytes: Ion-selective electrodes.

→ Cortisol Sensors: Electrochemical, Immuno-sensor sensor

→ Urea Sensor: Enzyme-based biosensor

★ Signal Processing Unit: Amplifiers or ADC is used.

★ Microcontroller: For Data acquisition, control and communication.

RAHUL KARTHIK.S
21BEC1851

★ Power source : Battery or a energy harvestor (Eg : Solar)

★ Wi-Fi Module : Wireless data transmission to the cloud.

ii. Circuit Diagram :

★ Sensor Interface : Each sensor is connected to microcontroller via individual ADC input channels. Amplifiers uses to condition the sensor signals.

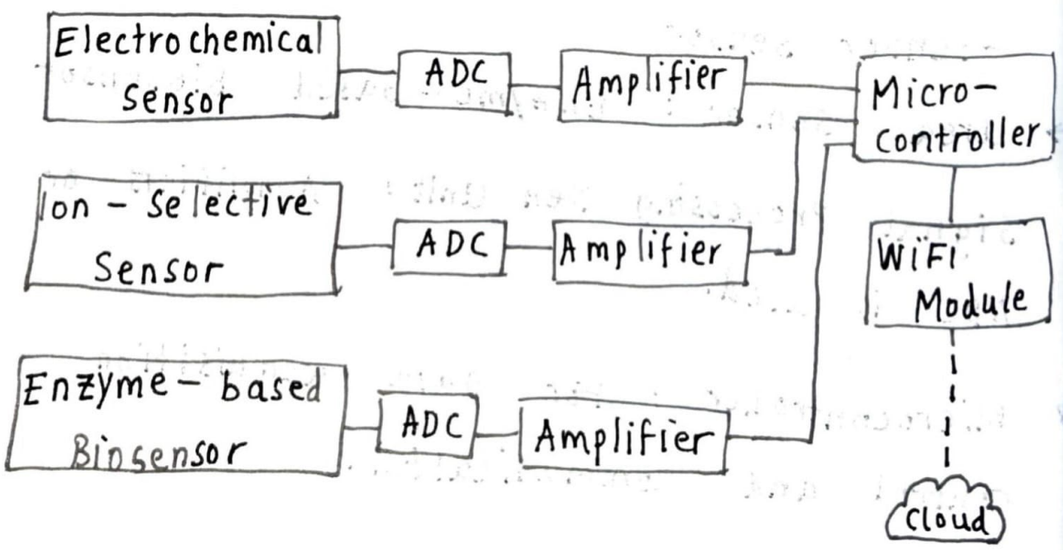
★ Microcontroller : Collect Data, Process it and transmit via Wi-Fi or Bluetooth.

★ Power Management

★ Wireless Module

iii. Cloud Communication :

★ once data is collected and digitized, with WiFi module that upload the data to the cloud.



RAHUL KARTHIK.S
21BEC1851

(2)

* An AI-enabled receptionist at hotel can handle tasks like greeting guests, answering common queries, making reservation, and providing information.

* The system involves several key components, including a voice recognition system, NLP, decision-making engine and output device.

i. Schematic Overview:

* Input Devices:

→ Microphone

→ Touchscreen / Keyboard

* Core Components:

→ Voice Recognition Module

→ NLP: Processes and interpret the text input to understand user intent.

→ Decision-Making Engine: Uses pre-trained ML models to determine response based on user query.

→ Knowledge Database: ~~Dab~~ Database about hotel services, room availability, FAQs, other details.

RAHUL KARTHIK.S
21BEC1851

★ Output Devices:

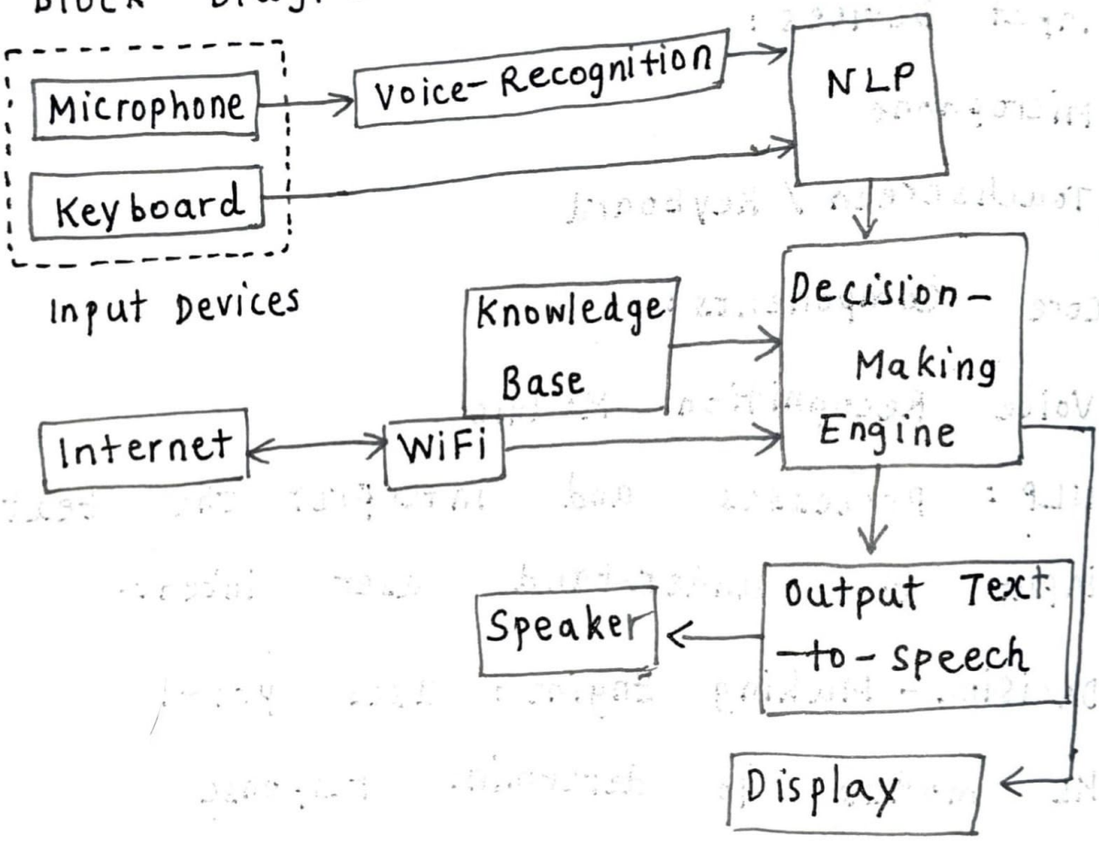
→ Text-to-Speech Module: Use TTS to convert text to natural sounding voice using speaker.

→ Display

★ Networking:

→ WiFi Module: Used to connect to the internet for external data.

Block Diagram:



(3)

j. Autonomous vehicles:

★ ~~Sen~~ Autonomous Vehicles rely on multiple sensors, control systems, computing platforms to drive, navigate

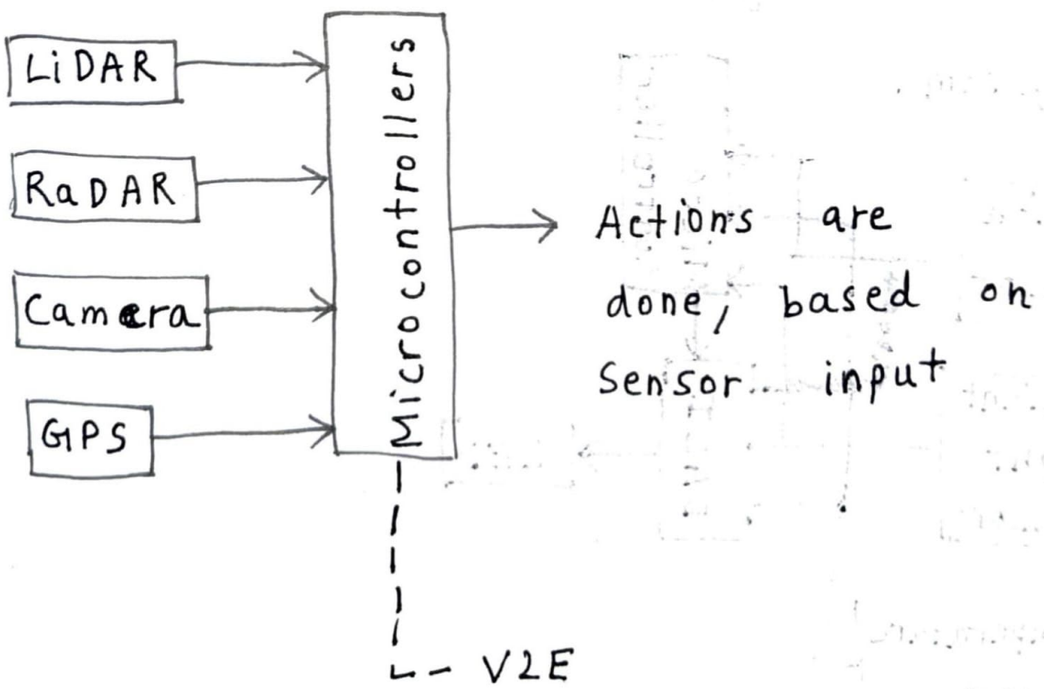
and make decisions in real-time.

* Sensors:
→ LiDAR: For mapping surroundings
→ RaDAR: For finding distance and speed
→ Camera: For sign board detection
→ GPS: For Navigation

* Control system:
→ Controllers
→ Electronic Control Unit

* Communication System:
→ Controller Area Network
→ Vehicle-to-Everything

* Decision-Making:
→ Based on the sensor information, decision is taken.



RAHUL KARTHIK.S
21BEC1851

ii.

★ Intelligent Lighting systems use embedded system to manage lighting based on a bunch of factors.

★ Sensors:

→ Motion sensors

→ Ambient Light Sensors

→ Temperature Sensors

★ Control System:

→ Microcontroller

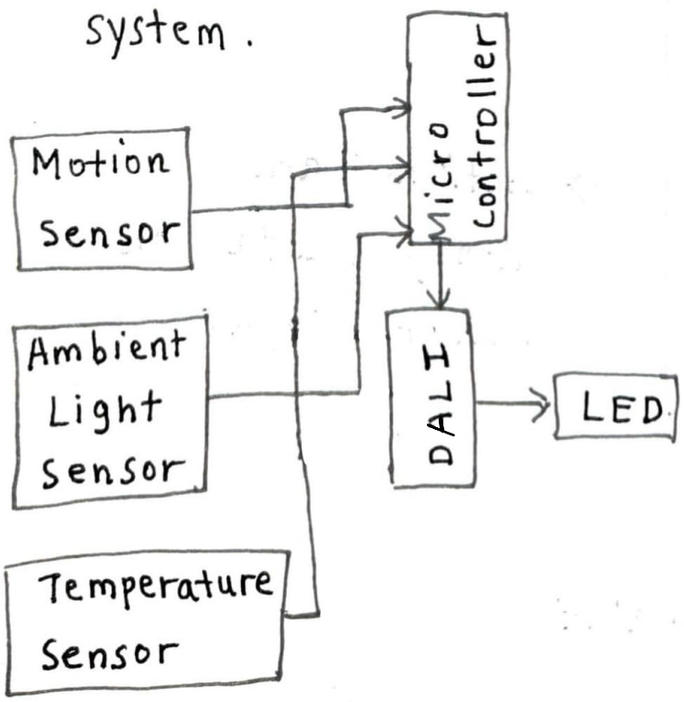
→ Digital Addressable Lighting Interface

★ Communication:

→ Wireless Connectivity like WiFi and Bluetooth.

★ Power Supply

→ Provide regulated power to lighting system.



RAHUL KARTHIK.S
21BEC1851

iii. Home Automation Systems:

* Home Automation System integrate many devices and appliances into a unified control system for convenience, security and energy management.

* Sensors:

→ Temperature and Humidity Sensors

→ Motion Sensors

→ Door / Window Sensors

→ Gas and smoke Detectors.

* control Units:

→ Micro controller: For data collection and decision making.

→ HVAC control System: For controlling temperature and Air quality in real-time.

→ Smart Switches: Embedded devices for controlling lights, fans and other appliances.

* Actuators:

→ Smart Relays: Control on/off State for application

RAHUL KARTHIK.S
21BEC1851

☆ Communication:

→ Wireless protocols like Wi-Fi & Bluetooth are used.

☆ User-Interfaces:

→ For controlling appliances from smartphones.

Block Diagram: