Invention Journal of Research Technology in Engineering & Management (IJRTEM)
ISSN: 2455-3689
www.ijrtem.com Volume 2 Issue 4 ǁ April 2018 ǁ PP 08-30

IOT BASED MEDICATION SYSTEM

[1,] Mrs. Urvashi Gupta, [2,] M. Nivedhitha , [3,]Likhitha , [4,] B. Ashiwini
1,2,3,4, (Department ECE, SRIDEVI WOMENS, ENGINEERING COLLEGE)

ABSTRACT : The category of patient involves all human beings-teachers, students, businessmen, housewives, children and also all of us have a busy hectic schedule. Today's life is full of responsibilities and stress. So, people are prone to diseases of different types and it is our duty to make ourselves stay fit and healthy. If the patient stays at home then he or she might get someone to look after him/her but when one is not at home, is out of the city or state away from home then it is hard for the family members to call them and remind them their dosage timings every time. In our developing and technology dependent life we totally rely on gadgets especially smart phones. Today everyone has a smart phone. With this we get an opportunity to use technology in a better way so that it can be made useful to us. And it plays an important part in our daily life and helps us staying fit in many ways. So, we are introducing an ARM7 and IOT application whose objective is to remind the patients of their dosage timings through Alarm Ringing system so that they can stay fit and healthy. This application focuses on the people who forget to take medicines on time. It allows users to set an alarm along with the fields of date, time and medicine description which will allow them to set alarm for multiple medicines at different time intervals. Medication reminders help in decreasing medication dispensing errors and wrong dosages. It is life-saving, money saving and time saving application which is easy to use and provides a good user interface.

KEYWORDS: *IR SENSOR,ARM 7, IOT Module, Buzzer.*

I. INTRODUCTION

In day-to-day life most of the people need to take medicines which was not there in past couple of years and the reason behind this is diseases are increasing in large amount. So sooner or later many people come in contact with these diseases. Some diseases are temporary diseases while many are permanent life threatening diseases. Life threatening diseases gets mixes with the human body in such a way that they can't leave the body ever and they increase in rapid time. Life span of humans became less because of such diseases and to overcome or to live a better life we need to take medicines regularly and also in large amount. We need to be in advice of Doctor who tells us to take desired pills in desired way so that patients face problems like forgetting pills to take at right time and also when Doctor changes the prescription of medicine patients have to remember the new schedule of medicine. This problem of forgetting to take pills at right time, taking wrong medicines and accidentally taking of expired medicine causes health issues of patient and this leads to suffer from unhealthy life. Our project is to made IOT based Smart medicine reminder box which uses Real time clock. The new awaited feature in our project is our system is sensible that patient has taken medicine or not and thus the patient can't postpone the time on which he needs to take pills. It is compulsory for the patient to take pills from the box at the right time otherwise our systems continue to make large sound until the medicine is taken out from the box. This notification feature adds life years to the patient and thus this thing is not available in any device which is the necessity for present days.

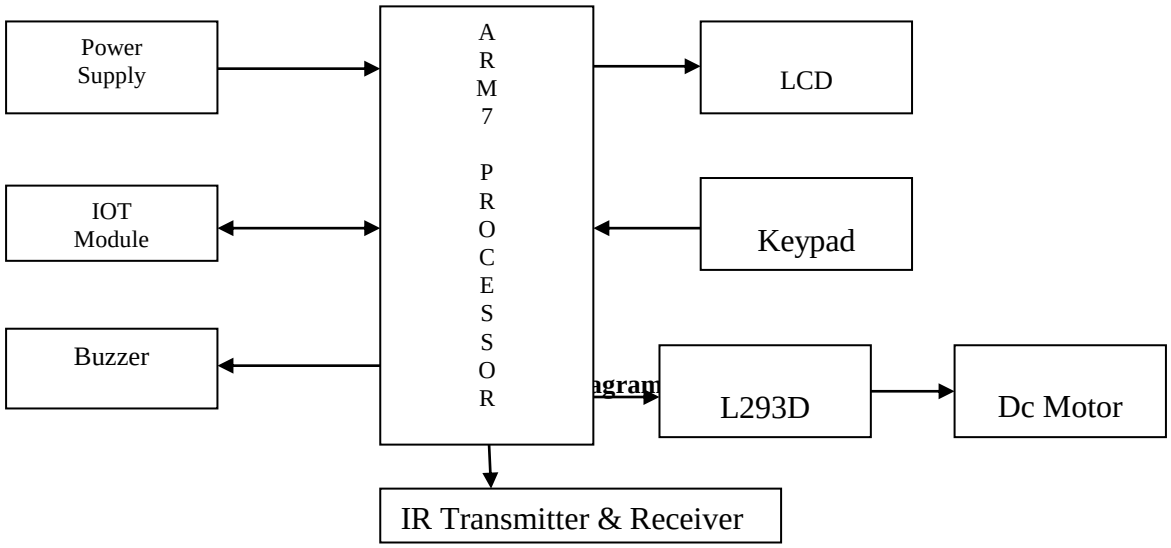
II. LITERATURE REVIEW

The background of the project will be revealed. The past researches on the same topic and some scientific writings will be discussed and compared to this project. The literature review is made so that ones will understand the theory of the Smart e-Pills Medication Reminder. The researches have been done on the common and disable people about their medication time. From survey, majority of them forget to take the medicine on time. The survey shows that around 50% of the people that were involved in the survey had or has problem remembering their medication time. Worst, these disable people especially blind people are having a time taking the medicine. That is why I have to come out with a design of a portable medicine box with reminder for these people to help them with their medication. This idea later leads to creating the Smart e-Pills Medication Reminder. One of the reasons on making this device is to set the alarm to indicate the medication time for people. The box will vibrate and the buzzer will be activated during the time when they have to consume their medicine. Thus, it will be easier for blind people to remember to consume their pills. Misplacing their medicine is also one of the problems that the patients usually faces.

Thus, the MEDBOX Reminder is also equipped with a remote finder that will emit a song when the remote finder button is push and this will help to locate the MEDBOX Reminder.

III. HARDWARE REQUIREMENTS

Block Diagram:



Arm 7 Processor: ARM documentation set for the ARM7 family of CPU processor cores, including ARM7TDMI, ARM7TDMI-S, ARM7EJ-S, and ARM720T. The ARM7 family is a range of low-power 32-bit RISC microprocessor cores optimized for cost and power-sensitive consumer applications. The ARM7 family incorporates the Thumb 16-bit instruction set - enabling 32-bit performance at 8/16-bit system cost.

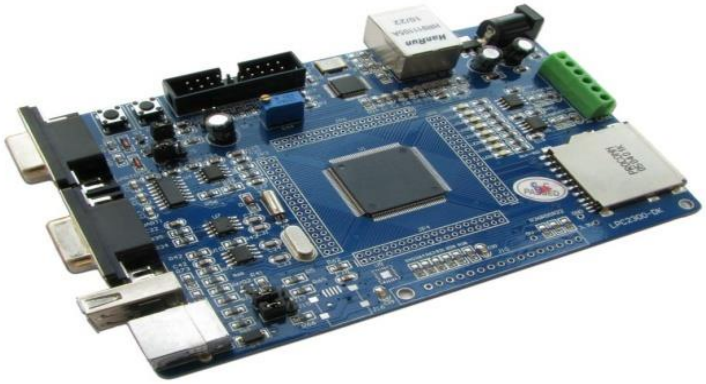


Figure 2: Arm 7 Processor

Regulated power supply: A variable regulated power supply, also called a variable bench power supply, is one where you can continuously adjust the output voltage to your requirements. Varying the output of the power supply is the recommended way to test a project after having double checked parts placement against circuit drawings and the parts placement guide. This type of regulation is ideal for having a simple variable bench power supply. Actually, this is quite important because one of the first projects a hobbyist should undertake is the construction of a variable regulated power supply. While a dedicated supply is quite handy e.g. 5V or 12V, it's much handier to have a variable supply on hand, especially for testing. Most digital logic circuits and processors need a 5-volt power supply. To use these parts, we need to build a regulated 5-volt source. Usually you start with an unregulated power supply ranging from 9 volts to 24 volts DC (A 12-volt power supply is included with the Beginner Kit and the Microcontroller Beginner Kit.). To make a 5-volt power supply, we use a LM7805 voltage regulator IC.

Block Diagram:

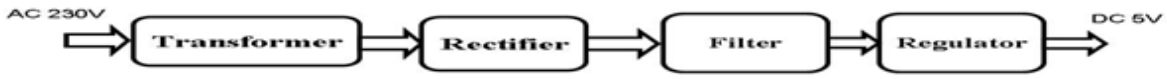


Figure.3.3. a Regulated Power Supply

The basic circuit diagram of a regulated power supply (DC O/P) with led connected as load is shown in fig:

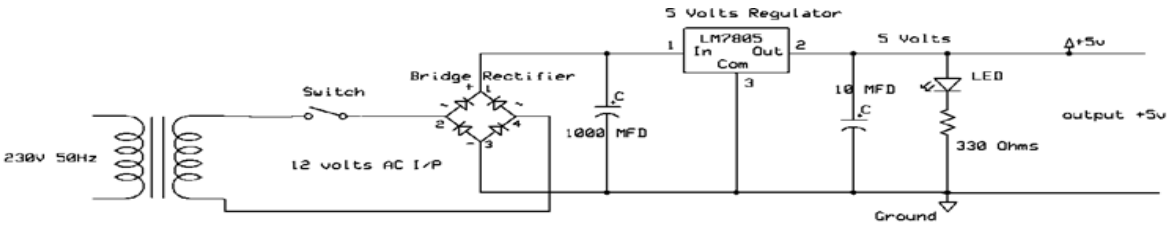


Figure.3.3. b. Circuit diagram of Regulated Power Supply with Led connection

The components mainly used in above figure are

- 230V AC MAINS
- TRANSFORMER
- BRIDGE RECTIFIER(DIODES)
- CAPACITOR
- VOLTAGE REGULATOR (IC 7805)
- RESISTOR
- LED (LIGHT EMITTING DIODE)

The detailed explanation of each and every component mentioned above is as follows:

1Transformers: A transformer is a device that transfers electrical energy from one circuit to another through inductively coupled conductors without changing its frequency. A varying current in the first or primary winding creates a varying magnetic flux in the transformer's core, and thus a varying magnetic field through the secondary winding. This varying magnetic field induces a varying electromotive force (EMF) or "voltage" in the secondary winding. This effect is called mutual induction. If a load is connected to the secondary, an electric current will flow in the secondary winding and electrical energy will be transferred from the primary circuit through the transformer to the load. This field is made up from lines of force and has the same shape as a bar magnet. If the current is increased, the lines of force move outwards from the coil. If the current is reduced, the lines of force move inwards. If another coil is placed adjacent to the first coil then, as the field moves out or in, the moving lines of force will "cut" the turns of the second coil. As it does this, a voltage is induced in the second coil. With the 50 Hz AC mains supply, this will happen 50 times a second. This is called MUTUAL INDUCTION and forms the basis of the transformer. The input coil is called the PRIMARY WINDING; the output coil is the SECONDARY WINDING. Fig: 3.3.4 shows step-down transformer.

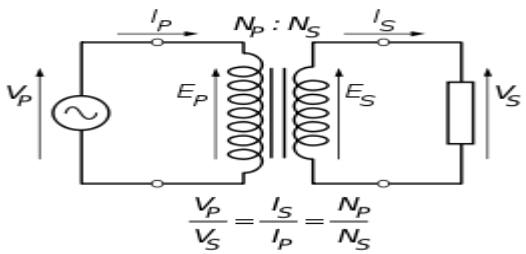


Figure 3: Step-Down Transformer

Rectifiers: A rectifier is an electrical device that converts alternating current (AC) to direct current (DC), a process known as rectification. Rectifiers have many uses including as components of power supplies and as detectors of radio signals. Rectifiers may be made of solid-state diodes, vacuum tube diodes, mercury arc valves, and other components.

Bridge full wave rectifier: The Bridge rectifier circuit is shown in figure, which converts an ac voltage to dc voltage using both half cycles of the input ac voltage. The Bridge rectifier circuit is shown in the figure. The circuit has four diodes connected to form a bridge. The ac input voltage is applied to the diagonally opposite ends of the bridge. The load resistance is connected between the other two ends of the bridge.

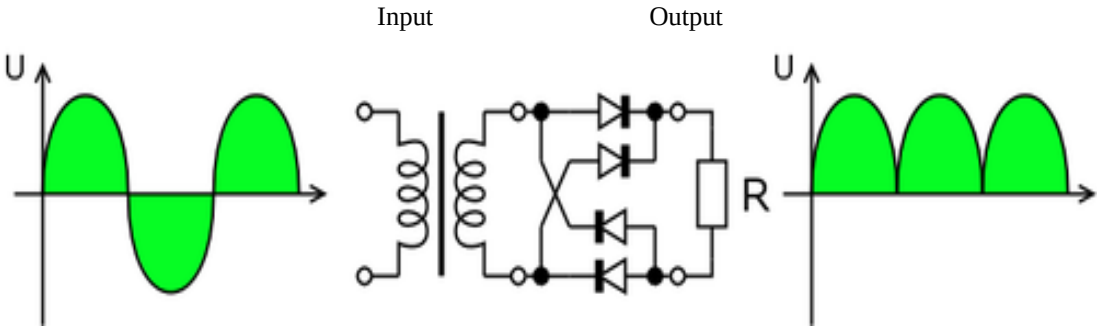


Figure 4: Bridge rectifier: a full-wave rectifier using 4 diodes

Filters: Electronic filters are electronic circuits, which perform signal-processing functions, specifically to remove unwanted frequency components from the signal, to enhance wanted ones. The **Capacitor** or sometimes referred to as a Condenser is a passive device, and one which stores energy in the form of an electrostatic field which produces a potential (static voltage) across its plates. In its basic form a capacitor consists of two parallel conductive plates that are not connected but are electrically separated either by air or by an insulating material called the Dielectric. When a voltage is applied to these plates, a current flows charging up the plates with electrons giving one plate a positive charge and the other plate an equal and opposite negative charge. This flow of electrons to the plates is known as the Charging Current and continues to flow until the voltage across the plates (and hence the capacitor) is equal to the applied voltage V_{cc}. At this point the capacitor is said to be fully charged and this is illustrated below.

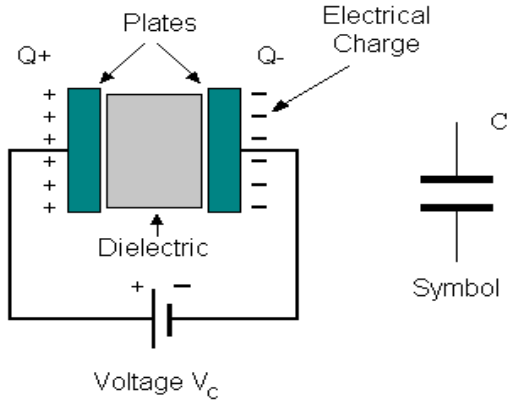


Fig 5:Construction Of a Capacitor

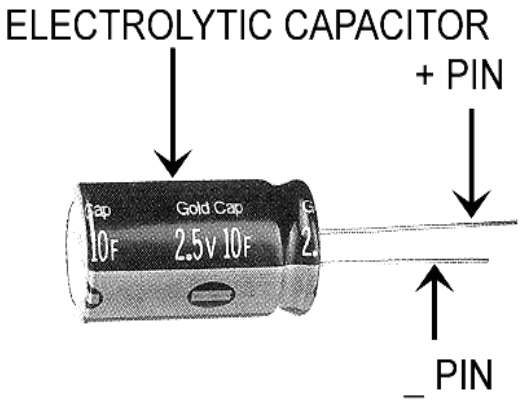


Fig 6:Electrolytic Capaticor

Voltage Regulator : A voltage regulator (also called a 'regulator') with only three terminals appears to be a simple device, but it is in fact a very complex integrated circuit. It converts a varying input voltage into a constant 'regulated' output voltage. Voltage Regulators are available in a variety of outputs like 5V, 6V, 9V, 12V and 15V. The LM78XX series of voltage regulators are designed for positive input. For applications requiring negative input, the LM79XX series is used. Using a pair of 'voltage-divider' resistors can increase the output voltage of a regulator circuit. It is not possible to obtain a voltage lower than the stated rating. You cannot use a 12V regulator to make a 5V power supply. Voltage regulators are very robust. These can withstand over-current draw due to short circuits and also over-heating. In both cases, the regulator will cut off before any damage occurs. The only way to destroy a regulator is to apply reverse voltage to its input. Reverse polarity destroys the regulator almost instantly.

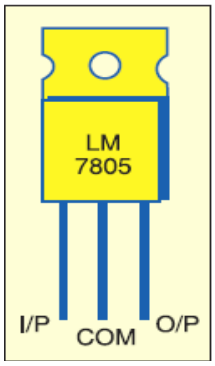
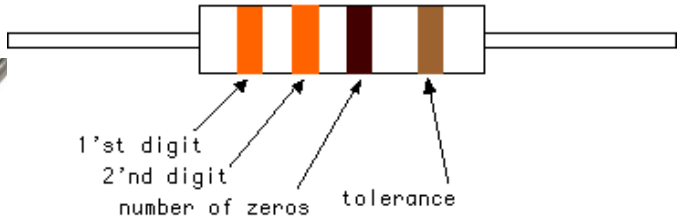


Fig 7: Voltage Regulator

Resistors: A resistor is a two-terminal electronic component that produces a voltage across its terminals that is proportional to the electric current passing through it in accordance with Ohm's law:
$V = IR$

Fig 8: Resistor

Resistor color Coding



Digit	color
0	Black
1	Brown
2	Red
3	Orange
4	Yellow
5	Green
6	Blue
7	Violet
8	Grey
9	White

Tolerance	color
20%	nothing
10%	Silver
5%	Gold
2%	Red
1%	Brown

Figure 9: Color Bands in Resistor

IOT Module: The IOT concept was coined by a member of the Radio Frequency Identification (RFID) development community in 1999, and it has recently become more relevant to the practical world largely because of the growth of mobile devices, embedded and ubiquitous communication, cloud computing and data analytics.[12] Imagine a world where billions of objects can sense, communicate and share information, all interconnected over public or private Internet Protocol (IP) networks. These interconnected objects have data regularly collected, analyzed and used to initiate action, providing a wealth of intelligence for planning, management and decision making. This is the world of the Internet of Things (IOT).

Figure 10: IOT Module

Buzzer

Figure 11: Buzzer

A buzzer or beeper is an audio signaling device, which may be mechanical, electromechanical, or piezoelectric. Typical uses of buzzers and beepers include alarm devices, timers and confirmation of user input such as a mouse click or keystroke. Buzzer is an integrated structure of electronic transducers, DC power supply, widely used in computers, printers, copiers, alarms, electronic toys, automotive electronic equipment, telephones, timers and other electronic products for sound devices. Active buzzer 5V Rated power can be directly connected to a continuous sound, this section dedicated sensor expansion module and the board in combination, can complete a simple circuit design, to "plug and play."

KEYPAD : Punch in your secret key into this numeric matrix keypad. This keypad has 12 buttons, arranged in a telephone-line 3x4 grid. It's made of a thin, flexible membrane material with an adhesive backing (just remove the paper) so you can attach it to nearly anything. The keys are connected into a matrix, so you only need 7 microcontroller pins (3-columns and 4-rows) to scan through the pad. We include a 7-pin extra-long header strip so you can plug this into a breadboard with ease.

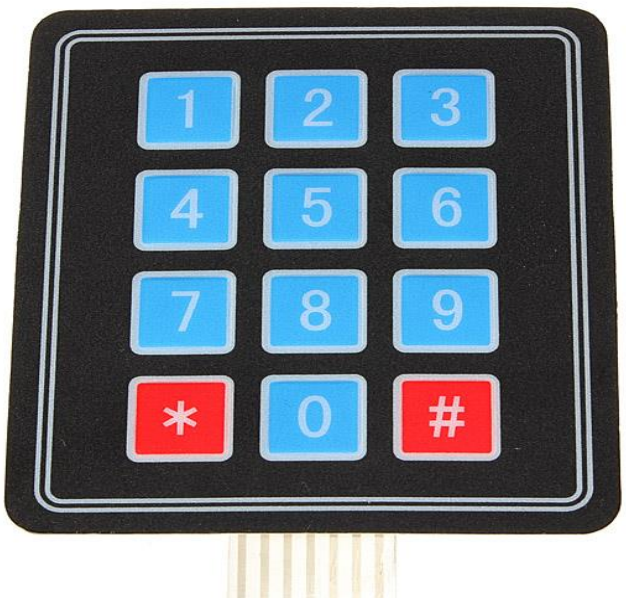


Figure 12: keypad

LCD (LIQUID CRISTAL DISPLAY): A liquid crystal display (LCD) is a thin, flat display device made up of any number of color or monochrome pixels arrayed in front of a light source or reflector. Each pixel consists of a column of liquid crystal molecules suspended between two transparent electrodes, and two polarizing filters, the axes of polarity of which are perpendicular to each other. Without the liquid crystals between them, light passing through one would be blocked by the other. The liquid crystal twists the polarization of light entering one filter to allow it to pass through the other. A program must interact with the outside world using input and output devices that communicate directly with a human being. One of the most common devices attached to an controller is an LCD display. Some of the most common LCDs connected to the controllers are 16X1, 16x2 and 20x2 displays. This means 16 characters per line by 1 line 16 characters per line by 2 lines and 20 characters per line by 2 lines, respectively.

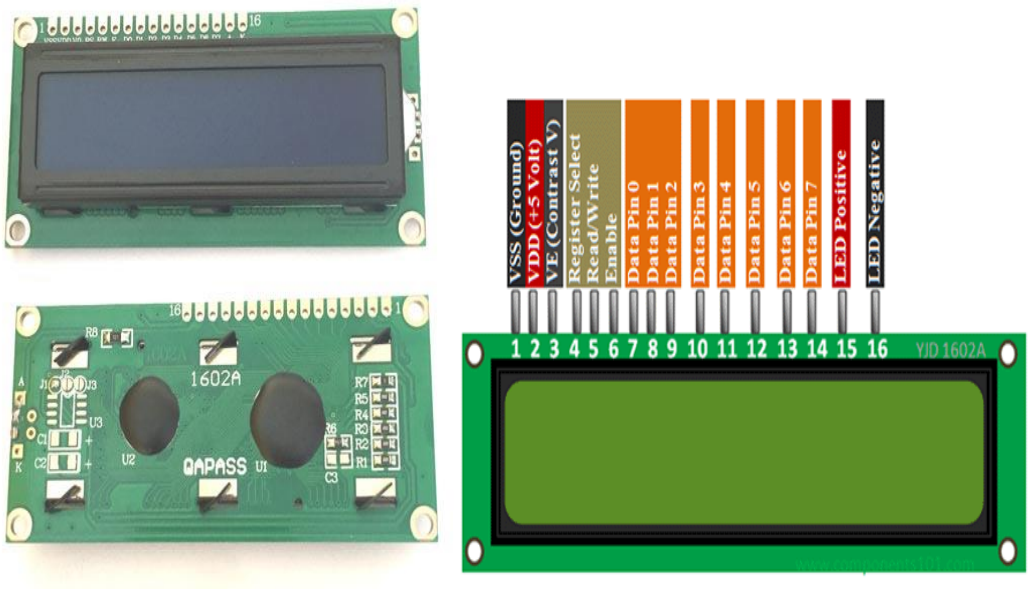


Figure 13: LCD

Push button : Push button switches are those which can be made to work with the force of a finger or two. Not only vehicles but camera, lifts and several other common and uncommon interactions with machines/gadgets involve push button switches applications. The image above shows the external view of a conventional SPST push button switch. Almost all the parts of the switch can be figured out by observing its external structure.

Figure 14: Push button

L293D : L293D is a dual H-bridge motor driver integrated circuit (IC). Motor drivers act as current amplifiers since they take a low-current control signal and provide a higher-current signal. This higher current signal is used to drive the motors. L293D contains two inbuilt H-bridge driver circuits. In its common mode of operation, two DC motors can be driven simultaneously, both in forward and reverse direction. The motor operations of two motors can be controlled by input logic at pins 2 & 7 and 10 & 15. Input logic 00 or 11 will stop the corresponding motor. Logic 01 and 10 will rotate it in clockwise and anticlockwise directions, respectively. Enable pins 1 and 9 (corresponding to the two motors) must be high for motors to start operating. When an enable input is high, the **associated** driver gets enabled. As a result, the outputs become active and work in phase with their inputs. Similarly, when the enable input is low, that driver is disabled, and their outputs are off and in the high-impedance state.

Shapes and S

IR Transmitter and Receiver

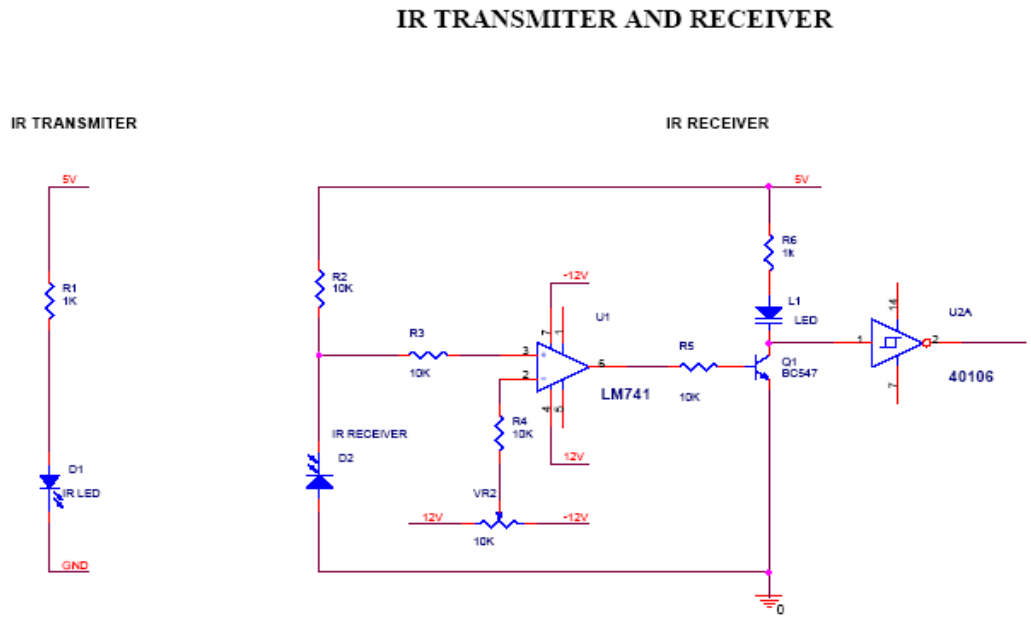


Figure 15: IR Transmitter and Receiver.

Infrared transmitter is one type of LED which emits infrared rays generally called as IR Transmitter. Similarly, IR Receiver is used to receive the IR rays transmitted by the IR transmitter. One important point is both IR transmitter and receiver should be placed straight line to each other. The transmitted signal is given to IR transmitter whenever the signal is high, the IR transmitter LED is conducting it passes the IR rays to the receiver. The IR receiver is connected with comparator.

GEAR MOTOR: Gear motors are complete motive force systems consisting of an electric motor and a reduction gear train integrated into one easy-to-mount and -configure package. This greatly reduces the complexity and cost of designing and constructing power tools, machines and appliances calling for high torque at relatively low shaft speed or RPM. Gear motors allow the use of economical low-horsepower motors to provide great motive force at low speed such as in lifts, winches, medical tables, jacks and robotics. They can be large enough to lift a building or small enough to drive a tiny clock.

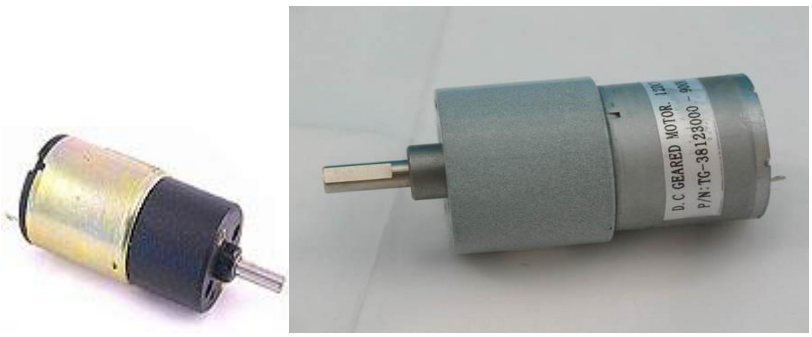


Figure 16: ***High Torque Dc Gear Motor***

Most synchronous AC electric motors have output ranges of from 1,200 to 3,600 revolutions per minute. They also have both normal speed and stall-speed torque specifications. The reduction gear trains used in gear motors are designed to reduce the output speed while increasing the torque. The increase in torque is inversely proportional to the reduction in speed

Project Flowchart`

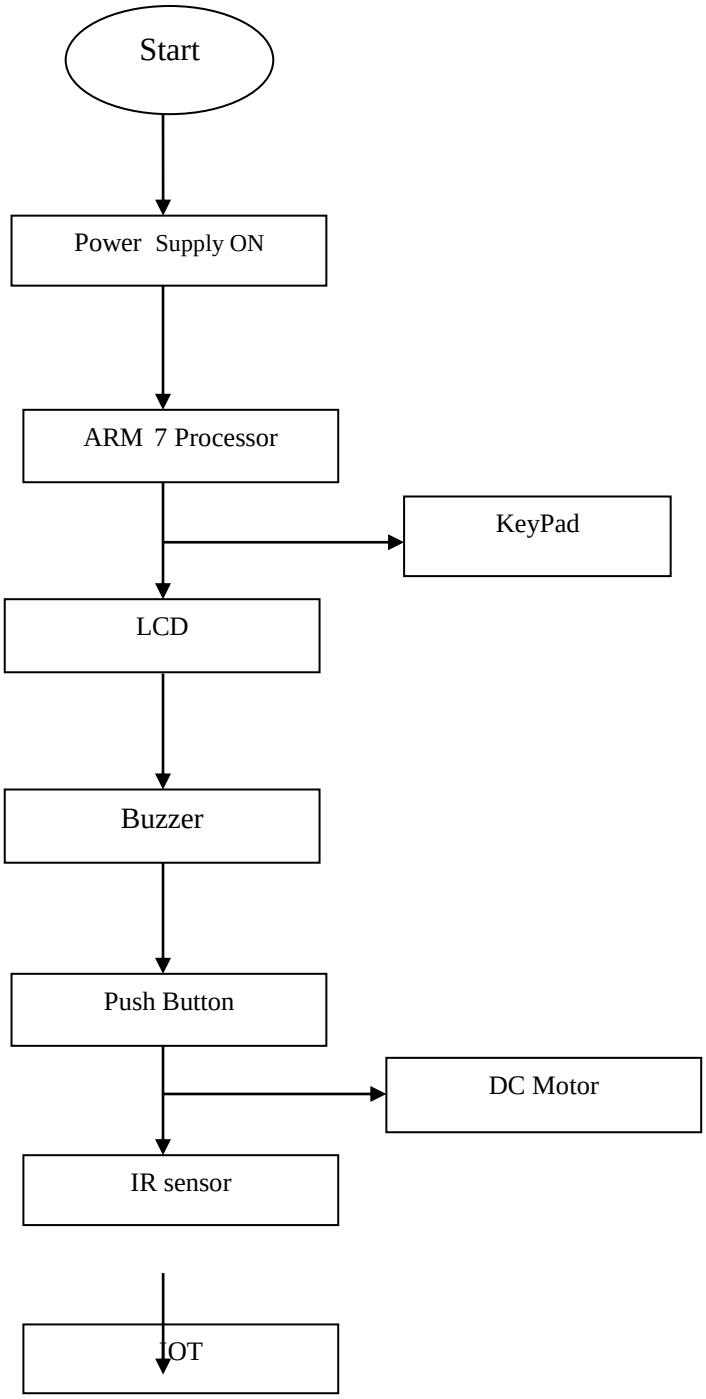


Figure 17: Flowchart

Watch part not only matches with main part forming "Enhance each other's beauty", but also is in the future development of real-time monitor forming a "Close security guards". Currently in this system has realized watch synchronization of medication remind, and vibration, etc. Mainly through the parameters Settings of the wireless module, and then make it connected to the wireless module of the main part. After received the signal, the watch drives Audi on conduction, the motor start shaking, to realize reminder. Follow-up work will also be implemented using the watch to realize time display and reminds about medicines which are to be taken on time.

IV. SOFTWARE REQUIREMENTS

Keil Software: In this chapter the software used and the language in which the program code is defined is mentioned and the program code dumping tools are explained. The chapter also documents the development of the program for the application. This program has been termed as "Source code". Before we look at the source code we define the two header files that we have used in the code.

V. RESULT AND DISCUSSION

In this project the code for advanced helmet system is programmed into Arduino using Arduino software.

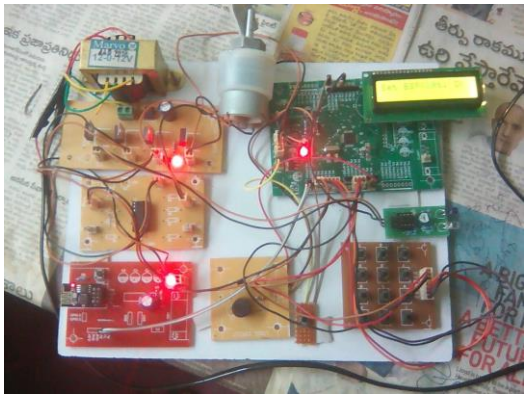


Figure 19: When Power Supply is on

Figure 20: Interval Required

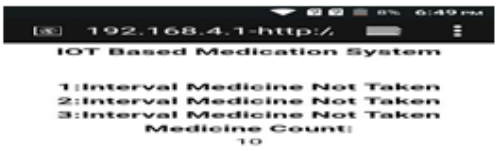


Figure 21: Count of Medicines

Figure 22: When Medicines Are Finished

Figure 23: Indicates About Taking the Medicines on the Web Page

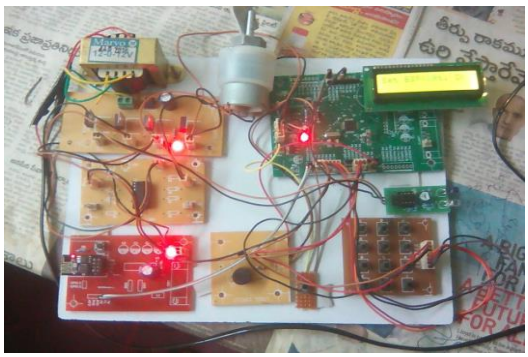


Figure 24: Over View

Applications

- Patients can take multiple medicines
- Complex medication schedule
- Can be a life savior at times, as it reminds patients to take medicines.
- Can be used as smart medical equipment

- Helps blind and partial deaf person to take medicines without any assistance

Advantages
- Cost efficient: Our product cost is affordable compare to other product available in market.
- User friendly: User can set time table of medicine by himself.
- Highly reliable: Good in quality and performance; able to be trusted for patients & old age people.
- Provide comfort and health: Comfortable for old age people and provide healthy life for patients who are regularly take medicines.
- Long-Lasting: The product can be used for long time.
- Easy to use and manufacture: It is very easy to use and manufacture.
- Accurate result: Alarm will ring at proper time which is set by user previously.
- Easy to maintain: It need less Maintenance. It is one time investment afterwards it can be used continuously.

Limitations
- It remains confined to dosage of only a week or two.
- More complex structure

VI. CONCLUSION

- The Automatic Medication Reminder For People ToTake Medicines In Time is a useful resource for those who need technological help in completing or need help in working through day-to-day tasks and taking care of their health.
- It is a smart and organized system that is designed with helping the elderly people in our homes, but we have not put any restrictions that stop an everyday user from using the system.

Future scope
- In the future, we hope that the application can be to linked to med karts, if the tablets are empty it directly sends a prescription message to the med kart in which they can help us delivering the prescribed tablets to our door step.
- Scanning of prescription to load the app can be done using image processing technology.
  There are several aspects we need to work on our device in the future to meet the user needs.

REFERENCES

Journals:

1. Ananda Mohon Ghosh, Debashish Halder, S K Alamgir Hossain, "Remote health monitoring system through IoT 5th International Conference on Informatics", *Electronics and Vision (ICIEV)*.
2. R. Kumar, M. Pallikonda Rajasekaran, "An IoT based patient monitoring system using raspberry Pi", *2016 International Conference on Computing Technologies and Intelligent Data Engineering (ICCTIDE'16)*.
3. Sarfraz Fayaz Khan, "Health care monitoring system in Internet of Things (IoT) by using RFID", *2017 6th International Conference on Industrial Technology and Management (ICITM)*.
4. Freddy Jimenez, Romina Torres, "Building an IoT-aware healthcare monitoring system", *2015 34th International Conference of the Chilean Computer Science Society (SCCC)*.

Appendix
Project Code

```
#include<LPC2103.H>
#include<string>
#define GPIO_Port0s_IODIR    IODIR
#define Set_Port0s           IOSET
#define Clear_Port0s         IOCLR
#define Port0_Set            IOPIN
#define button (1<<27)
#define buzzer (1<<15)
#define motor1 (1<<2)
#define motor2 (1<<3)
#include "LCD"
#include "Serial_Uart0.c"
```

```
#include "Serial_Uart1.c"
#include "keypad_4x3.c"
#include "ESP.c"
#include "rtc.c"
#include "GPS.c"
#include "app.c"
int PinStatus_Port(unsigned char ,unsigned char);
void Door_OpenClose(void);
void Medi_Taken(void);
unsigned char x;//,LCD_CLEAR=0x01;
//unsigned char ok1flag=0,ok2flag=0, ok3flag=0,ok4flag=0;
main()
{
GPIO_Port0s_IODIR=  ~(coloums|button);      //input=0     3 cols
GPIO_Port0s_IODIR =  (rows|LCD_Data|RS|EN|buzzer|motor1|motor2);
Clear_Port0s=buzzer;
Clear_Port0s=motor1;
Clear_Port0s=motor2;
Lcd_Init();
Init_UART0 (9600);
Init_UART0_Interrupt ();
RTC_Init();
RTC_Parameters_Setting();
ESP_GetReady();
Lcd_Data_Str(1,1,"Mediction       ");
Lcd_Data_Str(2,1,"System          ");
Delay(200);
Lcd_Data_Chr(0,0,0,LCD_CLEAR);
//Lcd_Data_Str(1,1,"Prss Butn");
//Lcd_Data_Str(2,1,"To Config Intrvl");
    Set_Periods();
          while(1)
{
RTCTime();
Medi_Taken();
}
}
int PinStatus_Port(unsigned char port, unsigned char pin)
{
if(port==0)
{
  x=(Port0_Set& (1<<pin))?1:0;
}
return x;
}
void GPSTime(void)
{
Gps_Getdata();
Gps_Time_Disp();
}
void Medi_Taken(void)
{
if(!PinStatus_Port(0,27))
{
if(i1flag)
{
ok1flag=1;
```

```
Door_OpenClose ();
}
if(i2flag)
{
ok2flag=1;
Door_OpenClose ();
}
if(i3flag)
{
ok3flag=1;
Door_OpenClose();
}
}
}
void Door_OpenClose(void)
{
Set_Port0s=motor1;
Clear_Port0s=motor2;
Delay (200);
Clear_Port0s=motor1;
Clear_Port0s=motor2;
Delay(200);
Set_Port0s=motor2;
Clear_Port0s=motor1;
Delay(200);
Clear_Port0s=motor1;
Clear_Port0s=motor2;
}
#define Fosc   12000000
#define Cclk   Fosc*5
#define Pclk   Cclk /4
void Init_UART0 (unsigned int baud rate);
void UART0_TX_Chr (unsigned char ch);
char UART0_RX_Chr (void);
char UART0_RX_Chr1 (void);
void UART0_TX_Str (unsigned char *str);
void Init_UART0_Interrupt(void);
void ISR_UART0(void)__irq;
void ISR_Timmer0(void)__irq;
void Get_IPD(void);
void Message_Reading(void);
void Misscall(void);
void Disable_UART_Interrupt(void);
unsigned int Divisor;
unsigned char rx_serial_data;
unsigned char flag=0;
  /* Initialize Uart0 Interface       */
void Init_UART0 (unsigned int baud rate)
  {
  PINSEL0 = 0x00050005;                   /* Selecting the GPIO to UART pins */
  Divisor=Pclk/(16*baud_rate);
  U0LCR = 0x83;
  U0DLL = Divisor;//%256;                     /* This value must be loaded (into U0DLL & U0DLM which forms
a DIVISOR) have 9600 Baud Rate @ 11.0592 MHz VPB Clock  */
 // U0DLM= Divisor/256;                     /* DLAB=1,8 bits, no Parity, 1 Stop bit                              */
   U0LCR = 0x03;                   /* DLAB = 0                                                          */
}
```

```
void UART0_TX_Chr (unsigned char ch)  {              /* Write character to Serial Port    */
 while (U0LSR == 0x20);
 // return (U0THR = ch);
 U0THR = ch;
 }
 void UART0_TX_Str (unsigned char *str)
  {              /* Write character to Serial Port    */
    while(*str!='\0')
   {
   while (U0LSR == 0x20);
   U0THR = *str;
   str++;
   Delay(1);
   }
}
char UART0_RX_Chr (void)  {                 /* Read character from Serial Port   */
 while (!(U0LSR & 0x01));
 return (U0RBR);
}
char UART0_RX_Chr1 (void)  {                 /* Read character from Serial Port   */
while (!(U0LSR & 0x01));
    {
  rx_serial_data=U0RBR;
  }
  return (rx_serial_data);
}
void Init_UART0_Interrupt(void)
{
   U0IER = 0x01;
  VICVectCntl0  |= 0x00000026;                  //select a priority slot for a given interrupt
  VICVectCntl1  |= 0x00000024;
  VICVectAddr0   = (unsigned long)ISR_UART0;       //pass the address of the IRQ into the VIC slot
  //VICVectAddr1  = (unsigned long)ISR_Timmer0;
  VICIntEnable  |= 0x00000050;  //40                 //enable interrupt2
}
void ISR_UART0(void)__irq
{
rx_serial_data=U0RBR;
VICVectAddr = 0x00000000;
if(rx_serial_data=='+')
{
Disable_UART_Interrupt();
Get_IPD();
rx_serial_data=0x00;
}
return;
}
void Enable_UART_Interrupt(void)
{
U0IER = 0x01;
}
void Disable_UART_Interrupt(void)
{
U0IER = 0x00;
}
#define rows    0x0F<<4
#define col1 1<<10
```

```
#define col2 1<<11
#define col3 1<<12
unsigned char keypad(void);
#define columns  (col1|col2|col3)
unsigned char Keypad Reading(void);
unsigned char x,i,Keypad_nos[15],KeyPad_value,passwd[5]="1235";
unsigned char passwd1=0xff, passwd2=0xff, passwd3=0xff;
unsigned char keypad_vlaues[4][3]=
{'1', '2', '3',
'4', '5', '6',
'7', '8', '9',
'*', '0', '#'};
unsigned char rowloc,colloc,key_count;
Keypad_Getdata ()
{
unsigned int ptr;
ptr=3;
for(i=0;i<4;i++)
{
Keypad_nos[i]=Keypad_Reading ();
Lcd_Data_Chr (1,2, ptr++, Keypad_nos[i]) ;//'*');
Delay (100);
}
ptr=3;
}
unsigned char Keypad_Reading(void)
{
   unsigned char i,j,key[4][3]={'1','2','3','4','5','6','7','8','9','*','0','#'};
   Delay (100);
   while (1)
   {
      j=1;
      for (i=0; i<4; i++)
      {
         IOSET=rows;
         IOCLR=rows&j<<4;
         if(! (IOPIN&col1))
         {
            while(! (IOPIN&col1));
            return key[i][0];
         }
         if(!(IOPIN&col2))
         {
            while(!(IOPIN&col2));
            return key[i][1];
         }
         if(!(IOPIN&col3))
         {
            while(!(IOPIN&col3));
            return key[i][2];
         }
         j=j*2;
      }
   }
}
void Send_AT_Cmd(unsigned char *);
void GSM_Modem_Init(void);
```

```
void Check_Response_GPRS_Modem(void);
void Clear_UART_Buffer(void);
void Check_SIM_Registration (unsigned char *);
void Check_Signal_Strength (unsigned char *);
void Check_Client_Request(void);
void Up_Date_WebPage(void);
void Up_Traffic_To_Page(void);
void Check_Traffic(void);
void Check_String(void);
int PinStatus_Port (unsigned char, unsigned char);
unsigned char Channel_Id, Serial_Int_Flag;
unsigned char Connection_ID;
unsigned char command [5], Server Data [25];
unsigned char flag1=1, flag2=1, flag3=1;
unsigned char temp;
unsigned char i1flag=0, i2flag=0, i3flag=0,i4flag=0,i5flag=0;
unsigned char ok1flag=0, ok2flag=0, ok3flag=0, ok4flag=0;
void ESP_GetReady(void)
{
Disable_UART_Interrupt ();
Lcd_Data_Str (1,1,"ESP Initing...");
Delay (200);
Lcd_Data_Str (1,1,"ESP Initing.....");
UART0_TX_Str("AT+RST");
UART0_TX_Chr(0x0D);
UART0_TX_Chr(0x0A);
Delay (600);
UART0_TX_Str("ATE0");
UART0_TX_Chr(0x0D);
UART0_TX_Chr(0x0A);
Delay (400);
UART0_TX_Str("AT");
UART0_TX_Chr(0x0D);
UART0_TX_Chr(0x0A);
Delay (400);
Lcd_Data_Str (1,1," ESP Inited Ok ");
Delay(100);
Lcd_Data_Str(1,1,"Set ESP as A.Pt.");
UART0_TX_Str("AT+CWMODE=2");
UART0_TX_Chr(0x0D);
UART0_TX_Chr(0x0A);
Delay(400);
Lcd_Data_Str(1,1,"ESP as Apt. OK ");
Delay(100);
Lcd_Data_Str(1,1,"Set ESP Name &Pwd");
UART0_TX_Str("AT+CWSAP=\"WIFI-Medi\",\"12345678\",1,3");
UART0_TX_Chr(0x0D);
UART0_TX_Chr(0x0A);
Delay(600);
Lcd_Data_Str(1,1,"ESP Name&Pwd OK");
Delay(100);
Lcd_Data_Str(1,1,"Set ESP as Servr");
UART0_TX_Str("AT+CIPMUX=1");
UART0_TX_Chr(0x0D);
UART0_TX_Chr(0x0A);
Delay(400);
UART0_TX_Str("AT+CIPSERVER=1,80");
```

```
UART0_TX_Chr(0x0D);
UART0_TX_Chr(0x0A);
Delay(400);
Lcd_Data_Str(1,1,"ESP as Servr RDY");
Enable_UART_Interrupt();
}
void Get_IPD(void)
{
//while(Get_Serial_Data()!='I');
while(UART0_RX_Chr()!='P');
while(UART0_RX_Chr()!='D');
while(UART0_RX_Chr()!=',');
Channel_Id=UART0_RX_Chr();
while(UART0_RX_Chr()!='G');
while(UART0_RX_Chr()!='E');
while(UART0_RX_Chr()!='T');
while(UART0_RX_Chr()!='/');
for(j=0;j<15;j++)
{
Server_Data[j]=UART0_RX_Chr();
if(Server_Data[j]=='/')
{
Server_Data[j]=0x00;
break;
}
}
Lcd_Data_Chr (1,2,16, Channel_Id);
//Lcd_Data_Str(1,14,Server_Data);
Up_Date_WebPage ();
}
void Up_Date_WebPage(void)
{
Disable_UART_Interrupt ();
UART0_TX_Str("AT+CIPSEND=");
UART0_TX_Chr(Channel_Id);
UART0_TX_Str(",242");//size of bytes to transfer -->
UART0_TX_Chr(0x0D);
UART0_TX_Chr(0x0A);
while(UART0_RX_Chr()!='>');
UART0_TX_Str("<!DOCTYPE html>");//15
UART0_TX_Str("<html>");//6
UART0_TX_Str("<meta http-equiv=\"refresh\" content=\"10\">");//39
UART0_TX_Str("<center>");//8
//All above=29+39=68
//-----------------------------------------------------------------------------------------------
UART0_TX_Str("<h1>IOT Based Medication System</h1>");//36
UART0_TX_Str("<br>");//4
UART0_TX_Str("<br>");//4
if(ok1flag)
UART0_TX_Str("<h1>1:Interval Medicine Taken    </h1>");//38
if(!ok1flag)
UART0_TX_Str("<h1>1:Interval Medicine Not Taken</h1>");
if(ok2flag)
UART0_TX_Str("<h1>2:Interval Medicine Taken    </h1>");//38
if(!ok2flag)
UART0_TX_Str("<h1>2:Interval Medicine Not Taken</h1>");//
```

```
if(ok3flag)
UART0_TX_Str("<h1>3:Interval Medicine Taken    </h1>");//38
if(!ok3flag)
UART0_TX_Str("<h1>3:Interval Medicine Not Taken</h1>");//
//-------------------------------------------------------------------------------------------------------------
UART0_TX_Str("</center>");//9
UART0_TX_Str("</html>");//7
//16
void Check_String(void)
{
if(strcmp(Server_Data,"time1 HTTP")==0)
{
}
else if(strcmp(Server_Data,"time2 HTTP")==0)
{
}
else if(strcmp(Server_Data,"time3 HTTP")==0)
{
Lcd_Data_Str(2,1,"Time-Slot:3     ");
}
else
{
Lcd_Data_Str(2,3,"               ");
Lcd_Data_Chr(1,2,1,Channel_Id);
Lcd_Data_Str(2,3,Server_Data);
}
}
 #define Fosc   12000000
#define Cclk   Fosc*5
#define Pclk   Cclk /4

void Init_UART1 (unsigned int baud_rate);
void UART1_TX_Chr (char ch);
char UART1_RX_Chr (void);
void UART1_TX_Str (char *str);
/*****************************Initialize Uart0 Interface*************************************/
void Init_UART1 (unsigned int baud_rate)
  {
  unsigned int Divisor;
   PINSEL0 = 0x00050000;                        /* Selecting the GPIO to UART pins */
  Divisor=Pclk/(16*baud_rate);
  U1LCR = 0x83;
  U1DLL = Divisor;//%256;                       /* This value must be loaded (into U0DLL & U0DLM which forms
a DIVISOR) have 9600 Baud Rate @ 11.0592 MHz VPB Clock  */
 // U1DLM=  Divisor/256;                        /* DLAB=1,8 bits, no Parity, 1 Stop bit                    */
  U1LCR = 0x03;                               /* DLAB = 0                                                */
}
/*****************************************************************************************
*/
/*****************************Write character to Serial Port*************************************/
void UART1_TX_Chr (char ch)  {
while (U1LSR == 0x20);
U1THR = ch;
 Delay(15);
}
/*****************************************************************************************
*/
```

```
/***********************************Writin string to Serial Port*****************************/
void UART1_TX_Str (char *str)
 {
     while(*str!='\0')
  {
  while (U1LSR == 0x20);
  U1THR = *str;
  str++;
  Delay(5);
  }
}
/*****************************************************************************************************
*/
/***********************************Read character from Serial Port*****************************/
char UART1_RX_Chr (void)  {
 while (!(U1LSR & 0x01));
    return (U1RBR);
}
/*****************************************************************************************************
*/
void Set_Periods(void);
void Get_Periods(void);
void RTCTime(void);
//0000000000000000000000000000000000000000000000000000
unsigned char periods,LCD_CLEAR=0x01,pcnt,lcnt1;
unsigned char P1h[5],P1m[5],p1hr,p1mn,p1flag=1/*,i1flag=0*/;
unsigned char P2h[5],P2m[5],p2hr,p2mn,p2flag=1/*,i2flag=0*/;
unsigned char P3h[5],P3m[5],p3hr,p3mn,p3flag=1/*,i3flag=0*/;
unsigned char P4h[5],P4m[5],p4hr,p4mn,p4flag=1/*,i4flag=0*/;
unsigned char P5h[5],P5m[5],p5hr,p5mn,p5flag=1;
unsigned char P6h[5],P6m[5],p6hr,p6mn,p6flag=1;
unsigned char P7h[5],P7m[5],p7hr,p7mn,p7flag=1;
unsigned char P8h[5],P8m[5],p8hr,p8mn,p8flag=1;
void Set_Periods(void)
{
Lcd_Data_Chr(0,0,0,LCD_CLEAR);
Lcd_Data_Str(1,1,"Intrvls Req");
Lcd_Data_Str(2,1,"");
periods=Keypad_Reading();
Lcd_Data_Chr(1,2,1,periods);
Delay(300);
if(periods<'4')
{
Get_Periods();
}
else
{
Lcd_Data_Chr(0,0,0,LCD_CLEAR);
Lcd_Data_Str(1,1,"OutBound  ");
Lcd_Data_Str(2,1,"Pls Select Again");
Delay(300);
}
}
void Get_Periods(void)
{
periods -='0';
while(pcnt<periods)
```

```
{
pcnt++;
//Lcd_Data_Chr (0,0,0, LCD_CLEAR);
//Lcd_Data_Str(1,1,"P:");
//Lcd_Data_Chr (1,1,3, periods+48);
/********* PERIOD:1 ***************/
/************************************/
if(pcnt==1){
Lcd_Data_Chr (0,0,0, LCD_CLEAR);
RTCTime();
Lcd_Data_Str(1,1,"I:");
Lcd_Data_Chr(1,1,3,pcnt+48);
Lcd_Data_Str(2,1,"Enter HR:");
lcnt1=9;
for(i=0;i<2;i++)
{
P1h[i]=Keypad_Reading ();
Lcd_Data_Chr(1,2, ++lcnt1,P1h[i]);
}
Delay(200);
p1hr=(((P1h[0]-48)*10)+(P1h[1]-48));
Lcd_Data_Chr(1,2,10,p1hr/10+48);
Lcd_Data_Chr(1,2,11, p1hr%10+48);
Delay(300);
Lcd_Data_Str(2,1,"Enter MN:");
Lcd_Data_Chr (1,2,10,' ');
Lcd_Data_Chr (1,2,11,' ');
lcnt1=9;
for(i=0;i<2;i++)
{
P1m[i]=Keypad_Reading ();
Lcd_Data_Chr(1,2,++lcnt1,P1m[i]);
}
Delay (200);
p1mn=(((P1m[0]-48)*10)+(P1m[1]-48));
Lcd_Data_Chr(1,2,10,p1mn/10+48);
Lcd_Data_Chr(1,2,11,p1mn%10+48);
Delay (300);
Lcd_Data_Chr (0,0,0, LCD_CLEAR);
RTCTime();
Lcd_Data_Str(1,1,"1#Alarm");
Lcd_Data_Str(2,1,"At:");
Lcd_Data_Chr(1,2,4,p1hr/10+48);
Lcd_Data_Chr(1,2,5,p1hr%10+48);
Lcd_Data_Str(2,6,":");
Lcd_Data_Chr(1,2,7,p1mn/10+48);
Lcd_Data_Chr(1,2,8,p1mn%10+48);
Delay(300);
}
/*****************************************************/
/*****************************************************/
/********************* PERIOD:2 ******************/
/*****************************************************/
if(pcnt==2){
Lcd_Data_Chr(0,0,0,LCD_CLEAR);
RTCTime();
Lcd_Data_Str(1,1,"I:");
```

```
Lcd_Data_Chr(1,1,3,pcnt+48);
Lcd_Data_Str(2,1,"Enter HR:");
lcnt1=9;
for(i=0;i<2;i++)
{
P2h[i]=Keypad_Reading();
Lcd_Data_Chr(1,2,++lcnt1,P2h[i]);
}
Delay(200);
p2hr=(((P2h[0]-48)*10)+(P2h[1]-48));
Lcd_Data_Chr(1,2,10,p2hr/10+48);
Lcd_Data_Chr(1,2,11,p2hr%10+48);
Delay(300);
Lcd_Data_Str(2,1,"Enter MN:");
Lcd_Data_Chr(1,2,10,' ');
Lcd_Data_Chr(1,2,11,' ');
lcnt1=9;
for(i=0;i<2;i++)
{
P2m[i]=Keypad_Reading();
Lcd_Data_Chr(1,2,++lcnt1,P2m[i]);
}
Delay(200);
p2mn=(((P2m[0]-48)*10)+(P2m[1]-48));
Lcd_Data_Chr(1,2,10,p2mn/10+48);
Lcd_Data_Chr(1,2,11,p2mn%10+48);
Delay(300);
Lcd_Data_Chr(0,0,0,LCD_CLEAR);
RTCTime();
Lcd_Data_Str(1,1,"2#Alarm");
Lcd_Data_Str(2,1,"At:");
Lcd_Data_Chr(1,2,4,p2hr/10+48);
Lcd_Data_Chr(1,2,5,p2hr%10+48);
Lcd_Data_Str(2,6,":");
Lcd_Data_Chr(1,2,7,p2mn/10+48);
Lcd_Data_Chr(1,2,8,p2mn%10+48);
Delay(300);
}
/*****************************************************/
/*****************************************************/

/********************* PERIOD:3 ******************/
/*****************************************************/
if(pcnt==3){
Lcd_Data_Chr(0,0,0,LCD_CLEAR);
RTCTime();
Lcd_Data_Str(1,1,"I:");
Lcd_Data_Chr(1,1,3,pcnt+48);
Lcd_Data_Str(2,1,"Enter HR:");
lcnt1=9;
for(i=0;i<2;i++)
{
P3h[i]=Keypad_Reading();
Lcd_Data_Chr(1,2,++lcnt1,P3h[i]);
}
Delay(200);
p3hr=(((P3h[0]-48)*10)+(P3h[1]-48));
```

```
Lcd_Data_Chr(1,2,10,p3hr/10+48);
Lcd_Data_Chr(1,2,11,p3hr%10+48);
Delay(300);
Lcd_Data_Str(2,1,"Enter MN:");
Lcd_Data_Chr(1,2,10,' ');
Lcd_Data_Chr(1,2,11,' ');
lcnt1=9;
for(i=0;i<2;i++)
{
P3m[i]=Keypad_Reading();
Lcd_Data_Chr(1,2,++lcnt1,P3m[i]);
}
Delay(200);
p3mn=(((P3m[0]-48)*10)+(P3m[1]-48));
Lcd_Data_Chr(1,2,10,p3mn/10+48);
Lcd_Data_Chr(1,2,11,p3mn%10+48);
Delay(300);
Lcd_Data_Chr(0,0,0,LCD_CLEAR);
RTCTime();
Lcd_Data_Str(1,1,"3#Alarm");
Lcd_Data_Str(2,1,"At:");
Lcd_Data_Chr(1,2,4,p3hr/10+48);
Lcd_Data_Chr(1,2,5,p3hr%10+48);
Lcd_Data_Str(2,6,":");
Lcd_Data_Chr(1,2,7,p3mn/10+48);
Lcd_Data_Chr(1,2,8,p3mn%10+48);
Delay(300);
}
if(pcnt==periods)
{
Lcd_Data_Chr(0,0,0,LCD_CLEAR);
Lcd_Data_Str(1,2,":Intervls");
Lcd_Data_Chr(1,1,1,periods+48);
Lcd_Data_Str(2,1,"Set Successfully");
Delay(300);
}
}
}
void Check_Bells(void)
{
if(HOUR==p1hr&&MIN==p1mn&&SEC==0)
{
Set_Port0s=buzzer;
Delay(1000);
Clear_Port0s=buzzer;
i1flag=1;
}
if(HOUR==p2hr&&MIN==p2mn&&SEC==0)
{
Set_Port0s=buzzer;
Delay(1000);
Clear_Port0s=buzzer;
i1flag=0;
i2flag=1;
}
if(HOUR==p3hr&&MIN==p3mn&&SEC==0)
{
```

```
Set_Port0s=buzzer;
Delay(1000);
Clear_Port0s=buzzer;
i1flag=0;
i2flag=0;
i3flag=1;
}
if(HOUR==p4hr&&MIN==p4mn&&SEC==0)
{
i4flag=1;
Set_Port0s=buzzer;
Delay(500);
Clear_Port0s=buzzer;
}
```