Embedded Patient Monitoring System

[1]V.Ramya, [2]B.Palaniappan,[3]Anuradha Kumari

[1]Asst. professor, Department of CSE, Annamalai University, Chidambaram, Tamilnadu.
ramyshri@yahoo.com
[2]Dean, FEAT, H.O.D, Department of CSE, Annamalai University, Chidambaram, Tamilnadu.
bpau2002@yahoo.co.in
[3]BE [IT], Department of CSE, Annamalai University, Chidambaram, Tamilnadu.
itanuradha577@gmail.com

ABSTRACT

The aim of this project is to inform the doctor about the ICU patient condition through wireless. For the medical professionals it becomes important to continuously monitor the conditions of a patient. In a large setup like a hospital or clinical center where a single doctor attends many patients, it becomes difficult to keep informed about the critical conditions developed in each of the patients. This project provides a device which will continuously monitor the vital parameters to be monitored for a patient and do data logging continuously. If any critical situation arises in a patient, this unit also raises an alarm and also communicates to the concerned doctor by means of an SMS to the doctor.

Keywords:

Embedded System, Microcontroller, NTC Thermistor, Sensor.

1. Introduction

This is an attempt to provide a device which will continuously monitor the body temperature and status of drip status of the patient. If either the temperature goes high or if the drip administration fails, this device will raises an alarm and communicate the concerned doctor by means of sending SMS to the doctor. The major part of this project is the hardware model consisting of sufficient sensor with embedded system.

1.1. Embedded System

An embedded system is a computer system designed to do one or a few dedicated and/or specific functions often with real-time computing constraints. It is *embedded* as part of a complete device often including hardware and mechanical parts. By contrast, a general-purpose computer, such as a personal computer (PC), is designed to be flexible and to meet a wide range of end-user needs. Embedded systems control many devices in common use today. Embedded systems contain

processing cores that are typically either microcontrollers or digital signal processors (DSP). The key characteristic, however, is being dedicated to handle a particular task. They may require very powerful processors and extensive communication, for example air traffic control systems may usefully be viewed as embedded, even though they involve main frame computers and dedicated regional and national networks between airports and radar sites (each radar probably includes one or more embedded systems of its own).

In general, "embedded system" is not a strictly definable term, as most systems have some element of extensibility or programmability. For example, hand held computers share some elements with embedded systems such as the operating systems and microprocessors that power them, but they allow different applications to be loaded and peripherals to be connected. Moreover, even systems that do not expose programmability as a primary feature generally need to support software updates. On a continuum from "general purpose" to "embedded", large application systems will have subcomponents at most points even if the system as a whole is "designed to perform one or a few dedicated functions", and is thus appropriate to call "embedded".

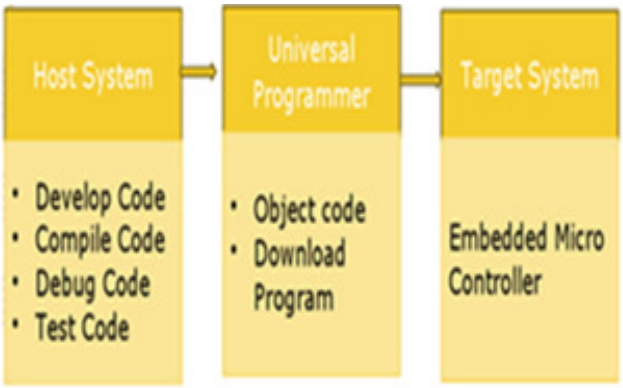


Figure 1: Embedded software design

1.2. Characteristics of Embedded system

1. Embedded systems are designed to do some specific task, rather than be a general-purpose computer for multiple tasks. Some also have real-time performance constraints that must be met, for reasons such as safety and usability; others may have low or no performance requirements, allowing the system hardware to be simplified to reduce costs.

2. Embedded systems are not always standalone devices. Many embedded systems consist of small, computerized parts within a larger device that serves a more general purpose. Similarly, an embedded system in an automobile provides a specific function as a subsystem of the car itself.

3. The program instructions written for embedded systems are referred to as firmware, and are stored in read-only memory or flash memory chips. They run with limited computer hardware resources: little memory, small or non-existent keyboard and/or screen.

1.3. Embedded Debugging

Embedded debugging may be performed at different levels, depending on the facilities available. Embedded software design is shown in figure 1. From simplest to most sophisticate they can be roughly grouped into the following areas:

- Interactive resident debugging, using the simple shell provided by the embedded operating system (e.g. Forth and Basic)

- External debugging using logging or serial port output to trace operation using either a monitor in flash or using a debug server like the Remedy debugger which even works for heterogeneous multicore systems.

- An in-circuit debugger (ICD), a hardware device that connects to the microprocessor via a JTAG or Nexus interface. This allows the operation of the microprocessor to be controlled externally, but is typically restricted to specific debugging capabilities in the processor.

- An in-circuit emulator (ICE) replaces the microprocessor with a simulated equivalent, providing full control over all aspects of the microprocessor. A complete emulator provides a simulation of all aspects of the hardware, allowing all of it to be controlled and modified and allowing debugging on a normal PC.

Unless restricted to external debugging, the programmer can typically load and run software through the tools, view the code running in the processor, and start or stop its operation. The view of the code may be as HLL source-code, assembly code or mixture of both. Because an embedded system is often composed of a wide variety of elements, the debugging strategy may vary. For instance, debugging a software- (and microprocessor-) centric embedded system is different from debugging an embedded system where most of the processing is performed by peripherals (DSP, FPGA, co-processor). An increasing number of embedded systems today use more than one single processor core. A common problem with multi-core development is the proper synchronization of software execution. In such a case, the embedded system design may wish to check the data traffic on the busses between the processor cores, which requires very low-level debugging, at signal/bus level, with a logic analyzer, for instance.

2. Hardware Design

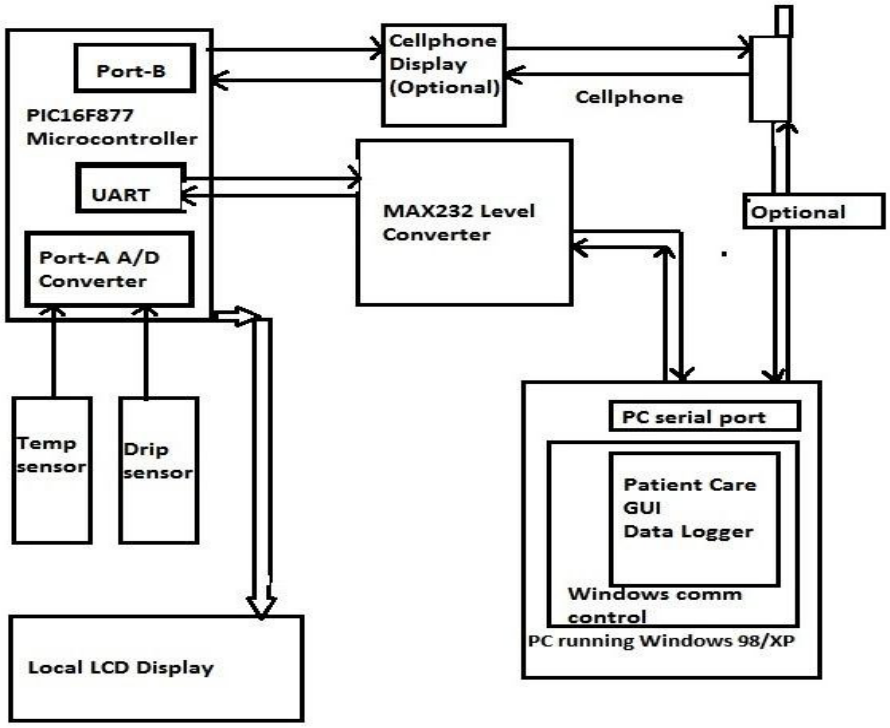


Figure 3: Block diagram of the proposed system

The block diagram of the system is given in the figure 3.The project is implemented with microchip PIC16F877A micro controller, and sensors were used to sense the temperature and drip status. The sensors are hooked to the in-built Analog to digital converter of the microcontroller. The PIC16F877A micro controller also has in-built UART which can interface to a PCs serial port. Level conversion of the signals is done before connecting the UART signals of the controller to the PC. MAX232 level converter is used for level conversion. Visual basic program is used to provide the GUI program for displaying the temperature values and drip status. Visual basic contains a control called windows communication control used for communication of peripheral equipment to PC. Hence when the micro controller runs continuously, it sends the temperature and drip data to PC, and the application program continuously receive the signals and display it on the PCs display. Any discontinuity of signals from the micro controller to the serial port of the PC is also taken care of by the Windows communication control. A mobile phone is hooked to microcontroller/PC and F-bus command set is used for issuing the Send SMS command from the micro controller/PC. If any critical situation arises the micro controller issues the appropriate F-bus commands to the concerned doctor's mobile number. These hardware modules contain various integrated chips which are used to control the system.

The System is divided into four hardware modules.

- Temperature detector
- Drip status detector

- PIC microcontroller
- Analog to digital converter
- PC and PIC interface
- Mobile and PIC interface

2.1. Temperature Detector

NTC thermistor is used to detect the temperature. This thermistor has 10k resistance in room temperature and 100 to 200ohms at 125c. This thermistor is connected in series. NTC thermistor detects the temperature with a 10k resistor. One end of the thermistor is connected to 5v and the other end of resistor is connected to ground. A tapping is made at junction of thermistor and resistor and connected to the analog input port of PIC. The voltage at the tapping will be varying between 2.5v and 5v depending on the temperature prevailing at the thermistor. The circuit diagram of temperature sensor is shown in figure 4. For this project, 98F is assumed as normal temperature.

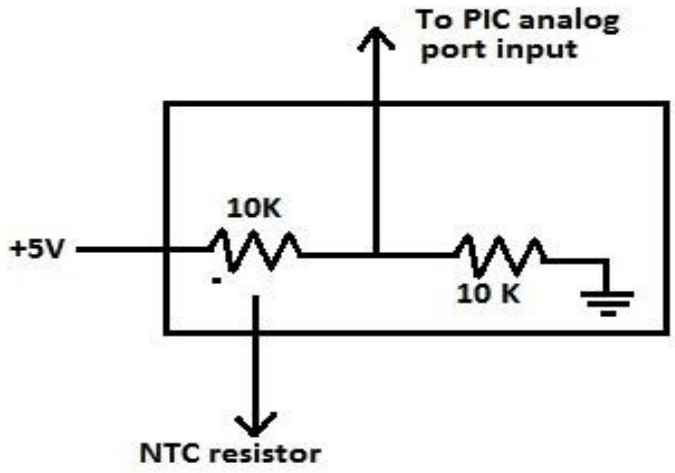


Figure 4: Temperature Sensor

2.2. Drip Sensor

For checking the drip status, two metallic probes are taken and one of the probes is connected to the ground. The other probe is connected to a 10K resistor. The other terminal of the 10K resistor is connected to 5V. A tapping is made at the junction between the probe and the resistor. When there is drip fluid present in the tube, the fluid will conduct. The 5V applied at the resistor is passed through the probes and get grounded. Hence the voltage available for the PIC will be low (Detected as 0 by PIC) but when the drip fluid is not present in the tube, the probes will not conduct. Because of this a 5V is applied to the resistor which will be available for the PIC (Detected as 1). If a 0 is detected, it indicates that the Drip status is normal. If a 1 is detected by the PIC, it indicates drip status as abnormal. For raising an alarm, 1 is sent through the PIC's port

(RB7). This port is connected to an LED. When a 1 is received by the LED (that is 5V) the LED will light up. Similarly through another port of PIC (RB6) a 1 is sent to start the alarm. When the drip status or temperature becomes normal, we send a 0 through the PIC's ports which will put off the LED and Alarm. Figure 5 shows the drip sensor.

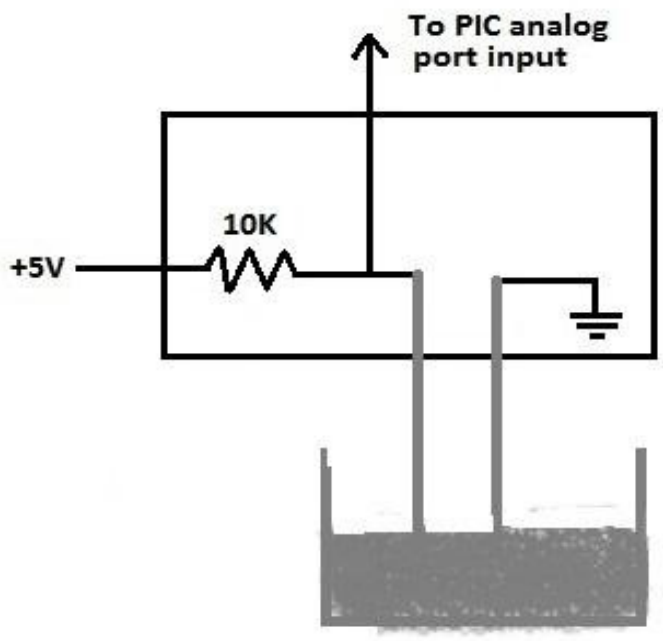


Figure 5: Drip Sensor

2.3. PIC Microcontroller

The Microcontroller used here is the PIC16F877. PIC (Peripheral Interface Controller) is a family of microcontrollers. It has attractive features and they are suitable for a wide range of application. It consists of I/O parts, 3 timers, ROM, RAM, Flash memory and inbuilt ADC. PIC channel 10 bit inbuilt ADC which convert the analog value into 10 bit digital data. PIC is programmed to convert 10 bit data into an 8 bit data and to transmit the data into a transistor driver. Figure 2 shows the architecture of PIC microcontroller.

2.3.1.Features

- High performance RISC CPU
- Only 35 single word instructions to learn.
- All single cycle instructions except for program Branches which are two cycle.
- Operating speed: 20MHz clock input,200 ns instruction cycle.
- Up to 8k x 14 words of FLASH program memory, up to 368 x 8 bytes of Data memory(RAM). Wide operating voltage range:2.0V to 5.5V
- Low-power consumption:
- -0.6 mA typical @ 3V,4MHz

- -<1μA typical standby current
- Timer0: 8-bit timer/counter with 8-bit prescaler.
- Timer1: 16-bit timer/counter with prescaler,can be incremented during SLEEP mode.
- Timer2: 8-bit period register, prescaler and postscaler
- Timer0: 8-bit timer/counter with 8-bit prescaler.
- Timer1: 16-bit timer/counter with prescaler,can be incremented during SLEEP mode.
- Timer2: 8-bit period register, prescaler and postscaler

2.4. Analog to digital converter(A/D)

The Analog-to-Digital (A/D) Converter module has five inputs for the 28-pin devices and eight for the other devices. The A/D conversion of the analog input signals results signal results in a corresponding 10-bit digital number. The A/D convert has a unique feature of being able to operate while the device is in SLEEP mode. To operate in SLEEP, the A/D clock must be derived from the A/D's internal RC oscillator. The A/D module has four registers, these four register are:

- A/D Result High Register(ADRESH)
- A/D Result Low Register(ADRESL)
- A/D Control Registe0r(ADCON0)
- A/D Control Register1(ADCON1)

The ADCON0 register controls the operations of the A/D module. The ADCON1 register configures the function of the port pins. The port pins can be configured as analog inputs, or as digital I/O, Additional on using the A/D module can be found in the PIC micro Mid-range MCU family.

2.5. PC and PIC Interface

The RC6 line of PIC is connected to pin 10 of MAX232 and pin 7 of MAX232 is connected to pin2 of DB-9 connector of PC.RC7 line of PIC is connected to PIN 9 of MAX232 and pin 8 of MAX232 is connected to pin 3 of DB-9 connector of PC. The PC to PIC interface is shown in Figure 6.

2.6. Mobile and PIC Interface

A mobile phone is hooked to microcontroller/PC and F-bus command set is used for issuing the send SMS command from the microcontroller. F-bus is a high speed full duplex bus. It uses pin 1 for MBUS, pin 2 as ground, pin 3 for receiving data and pin 4 for transmitting the data. The F-bus is bi-directional serial type bus running at 115,200 bps, 8 data bits. The serial cable contains electronics for level conversion and therefore requires power. For this the DTR(data terminal ready) pin is connected to +3 to +12 Volts supply and RTS pin is connected to -3 to -12volts supply by using MAX232 for the RS232 TX and RX pins. The next step is to synchronize the UART in the phone with the PC or microcontroller. This is done by sending a string of 0x55 or 'U' 128 times.

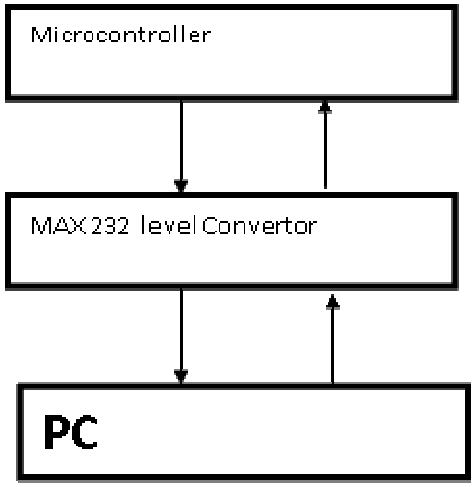


Figure 6: Mobile and PIC interface

3. Software Description

3.1. PIC programming procedure

1) Open MicroCode Studio

Double click on the MicroCode Studio desktop icon or select from the Start menu:
Programs | MicroCode Studio (MCSX) | MicroCode Studio (MCSX).

2) Create or Open PicBasic Program

Select File | New to start from scratch. For Editing an existing project, select File | Open and browse to your code file. The file can be created initially in any text editor (e.g., Windows NotePad or Microsoft Word, saving the file as "Plain Text: *.txt").

3) Save and Name Project File

The file is saved and stored in desired location. PICBASIC PRO file (*.pbp) file type is used.

4) Choose the PIC Device you are using

Select the appropriate PIC microcontroller (usually the 16F87) from the pull-down box in the *Microcontroller* (MCU) toolbar. Microcode Studio and the U2 Programmer support only the Devices listed.

5) Check for Errors

To make sure there are no errors in the code, Compile button is clicked on the *Compile and Program Toolbar*. If there are any errors, Microcode Studio will identify and locate them. To have the line #'s appear in the editor window (if they are not there already), *View | Editor Options*

is selected and check the *Show line numbers in left gutter* box. The errors are corrected and *compiled* again until there are no more errors. After a Successful compile, the status line at the bottom of the window will read "Success" and indicate how much memory that the program is using on the PIC.

6) Preparing the PIC for Programming

USB cable is plugged into the U2 Programmer. The green LED in the device is switched on. The metal lever on the U2 Programmer ZIF socket is in the up position. PIC is inserted into the socket with pin 1 in the position indicated on the socket board. The "Pin 1" position is different depending on the # of pins on your PIC, as indicated on the green U2 socket board. The socket is pivoted lever down to lock the PIC in place. PIC programming is shown in figure 7.

Figure 7: Programming the PIC

7) Prepare the Code for Download onto the PIC

The *Compile Program* button is clicked to compile the code and generate the files needed for Programming the PIC. This will launch the meProg utility that allows to store the code on the PIC.

8) Identify the PIC Model Number

The PIC device number should transfer from Microcode Studio, but we should still verify this and change it if necessary in the meProg window pull-down list.

9) Select the Appropriate Configuration Bit Settings

Again in the meProg window, *View | Configuration button is* clicked. Click on the down-arrows to select the desired or appropriate choice for each feature listed.

10) Changing Configuration Settings in Code

The code is added for the settings for which the default values are different from what our system is having. The settings available for a given PIC can be found in the appropriate *.INFO file for the device.

11) Download the Code on to the PIC

After all of the configuration choices have been set to the desired values, Program icon is clicked. The U2 programmer LED will glow red while the code is being downloaded, and it should glow green again when the process is completed. After the program is written and verified, a *Program Verify complete* dialog box should appear, indicating that everything worked properly. Then click *OK*.

12) Remove and Test the Programmed PIC

The lever is lifted on the programmer to release the pin clamp. The PIC is removed from the socket and inserted into your circuit for testing.

13) Shutdown the Software and Logoff

The MicroCode Studio application is closed (Exit). The programmer and configuration windows will close automatically with MicroCode Studio.

4. Implementation

Figure 8 shows the prototype of the developed system. This project can also be used to monitor temperature in industrial scale were the temperature in the upper limit may not exceed about 110 C. Utilization of the cell phone to message and the alert the user was another advantage. Hence this interface could as well be used to remotely monitor and gets alerts in case the process value exceeds the limit. The only limits observed in this project are that we must ensure that cell phone is functional during the operation of this module. That is appropriate backup power supply or charging of the cell phone batteries is needed. For this, we can always leave the charger usually provided with the cell phone, connected to the handset. In the Visual Basic SDK, we have to configure the project properties to include the MS Comm Control in this project. The properties of Comm Control (communication control) is set to have 9600 baud, no parity , 8 data bits and 1 stop bit is provided. There after we can call the Comm Control port open, Comm Control input, Comm Control Output and Comm Control port close functions.

Figure 8: Prototype of the developed System

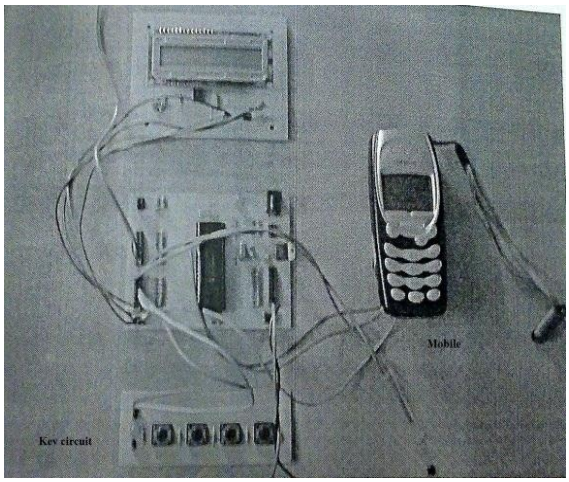
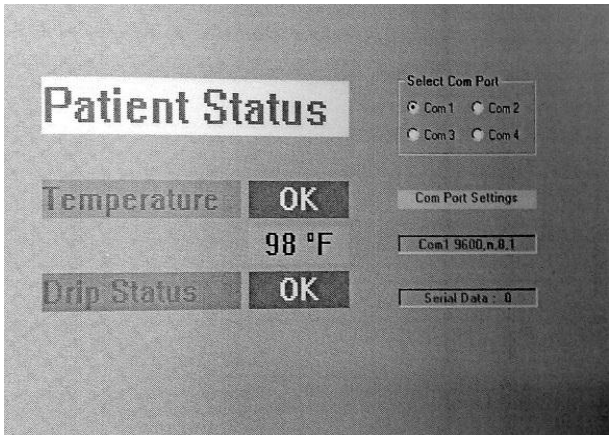


Figure 9: VB Interface for system



The VB interface of the proposed system is given in figure 9. The developed system uses visual basic for displaying the temperature and drip status. Visual basic is a beginner programming language for authoring Windows – based software. It provides the GUI interface for displaying the sensed parameters. At first the temperature is sensed and taken as input through NTC thermistor. Then the temperature is computed and converted from analog to digital. In parallel to it drip input is taken by using metallic probes. Both temperature and drip status are sent to PC. If the temperature is not greater than the set value (98 F) then the System restarts the sensing. If the temperature is greater than the set value, then an alert to cell phone and PC is sent. If any key is not pressed in the cell phone in response to the alarm then the sensing process restarts and the alarm keep on ringing until the doctor responds. If any key is pressed in response to the alarm then the preset value for temperature changes to the sensed value and the alarm is automatically switched off. Then the system restarts to take the temperature and drip input. Figure 10 shows the flowchart of the developed system.

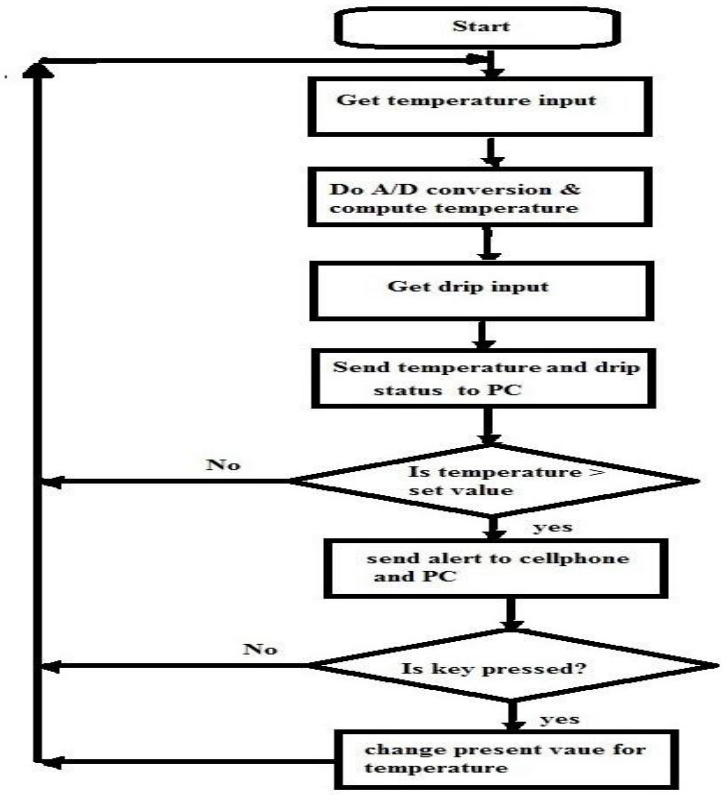


Figure 10: Flow chart of the proposed System

5. Conclusion

This paper presents the embedded intensive care unit using PIC microcontroller. The project is monitoring the patient's body temperature and the status of drip administered and makes data logging (on PC) and reporting/alerting (using cell phone).The availability of in-built A/D converter inPIC16F877A has been very useful in the easy implementation of the digital temperature measurement. The chip used in this project (PIC16F877A) contains 8 analog channels, of which we have used only one for temperature measurement. In the actual scenario in a hospital , there are many other vital parameters to be monitored in a patient like heartbeat, pulse rate, breathing and ventilator activity etc. this project can further be enhanced or improved by adding facilities to monitor the above mentioned parameters too. In that case the additional analog input channels will be of great use.

REFERENCES

[1] Cyber-Physical Medical and Medication Systems by Albert M. K. Cheng, 2008.

[2] Wireless Transfusion Supervision and Analysis Using Embedded System Nivedita Daimiwal, DipaliRamdasi, RevathiShriram, AsmitaWakankar, 2010.

[3] A low cost model for patient monitoring in Intensive care unit using a micro web-server by JoãoBosco da MotaAlver Juarez Bento da Silva ,SuenoniPaladini.

[4] Steve Heath, 'Embedded system and design' butterworth-heinemann publications, New Delhi, first edition, 1997.

[5] Microchip company, 'EmbeddedSolutions', microchip publications, first edition, 1999.

[6] TammyNoergaArdewnes, 'EmbeddedSystems Architecture', first edition 1999.

[7] Paul Sherriff, 'visual basic 6', prenticehall publication, New Delhi, first edition1999.

[8] Arnold Berger,' Embedded System Design', first edition 1997.

[9] http://www.microchip.com[pic microcontroller]

[10] http://www.gnokii.org [mobile interface]