Commercial Automation Using RTOS-Based Architecture

[1]Dr. Dipali Shende , [2]Mrs. Vaishali Baste, [3]Mrs. Shital More
[1]Department of E & TC, PCCOER,Ravet
[2] Department of E & TC,SKNSCOE, Vadgaon
[3] Department of E & TC, SIT Lonavala

ABSTRACT- The primary objective of this research is to introduce an RTOS-based architecture for data transmission between the host region and the industry area. RTOS is a process that takes place between hardware and software. It proposes a model which is used to monitor (Host Area) the different physical values in industrial plants and its assembling environment (Industry Area) by adding the functionality of an RTOS. In an electronics manufacturing facility, temperature control, security, and other factors are crucial. If an accident occurs, productivity will suffer. With the aid of the Wireless Module, a system built with an ARM processor, and a port of a Real Time Operating System (RTOS) based on Linux, this project seeks to address this issue by remotely monitoring the temperature and other physical parameter levels of various plant areas. Here in addition to detection of any faults, the preventive action is also taken by using the control system mechanism. These things will make the industry to operate properly by preventing the accidents in real time and to increase the level of production.

KEYWORDS—ARM-Advanced RIS C Machines; RTOS -Real Time Operating System; RTLinux-Real Time Linux; VB-Visual Basic

I. INTRODUCT ION

The RTOS-implemented system has the capacity to multitask for task monitoring and task control. By modifying the kernel, the system can also add more apps. Therefore, system updates are also feasible as needed.[1][7] The most recent ARM Controller offers more advanced applications nowadays. The system can be connected through the wireless sensor network for monitoring remote area. Due to warmth and humidity, the electronics industry and others have recently experienced significant problems. These problems include increased board oxidation and bridging, faulty soldering joints, and solder component flaws. Through these parameters, the environment, in particular a few devices like desiccators and solder paste refrigerators for storing paste and bare PCBs, respectively, also has to be controlled. In order to increase a machine's efficiency, several factories try to keep the temperature under control. However, doing so can result in unanticipated accidents, poor product quality, and other problems. Microcontroller-based interfaces and the monitoring and control system they enable are more expensive and less effective.[2][8] Also the response of the system is late which may sometimes lead to catastrophe. That means a real time response is missing part. Considering all these parameters the cost, efficiency, durability, reliability of such
system will not be optimum. So, a change in the existing system is needed. Real Time Linux (RTLinux) can be ported to the ARM platform so that the system can be implemented. The RTOS-implemented system has the capacity to multitask for task monitoring and task control. By modifying the kernel, the system can also add more apps. Therefore, system updates are also feasible as needed. The most recent ARM Controller offers more advanced applications nowadays. For monitoring a remote location, the system can be connected via a wireless sensor network. For the ARM Controller, RTLinux is a pre-emptive, hard real-time deterministic multitasking kernel. RTLinux can run additional jobs by applying patches in accordance with our needs. RTLinux manages memory, time, and inter-process communication (using semaphores, message queues, and mailboxes). By gathering data from perceived values of the electronics manufacturing sectors under the direction of engineers, it helps to monitor and control the environment. The system's entire architecture is based on three categories: communication protocols, software details, and hardware specifics. The RTOS-implemented system has the capacity to multitask for task monitoring and task control. By modifying the kernel, the system can also add more apps.[3][9]

II. CURRENT COMMERCIAL AUTOMATION SCENARIO

Globally, the new phrase The economy is quickly picking up speed. The industrial sector, which includes the automotive industry, has been altered and moulded by this new, disruptive technology, which has lately discovered its growth curve. Following Industry 4.0, this business era is seen as a collection of ICTs (information and communications technologies) and digitally enabled technologies. These include advancements in manufacturing machinery, intelligent goods, data tools, and analytics that make use of the Internet of Things (which can include, among other things, 3-D printing, prototypes, connected cars, product lifecycle management, and cyber physical production systems).[4][10]

The Government of India is giving the Indian automotive industry a boost and a push as it emphasises and concentrates on introducing new and revolutionary production techniques into the country's manufacturing system while putting ICT at the centre of development. Along with this, there is also a strong emphasis on Make in India, the implementation of GST, and FDI policies.[5][11]

III. LITERATURE SURVEY

Applications for real-time software are common today because they enable speedier completion of tasks, processes, and activities. Real Time Operating Systems and the related applications have been the subject of extensive research. A few peer-reviewed journal papers are investigated. The following is a summary of the papers' main findings and recommendations.[6][12]

We learn from the article "An intelligent architecture

for industrial automation using RTOS technology" published in "International Journal of Application or Innovation in Engineering & Management (IJAIEM)" [1],that the RTOS is responsible for managing the orderly and controlled distribution of these resources to users. Microprocessors, transceivers, displays, and analogue to digital converters make up this wireless sensor node. For the monitoring and management of industrial processes, sensor nodes are deployed. In the Master node, the sensing parameters can be shown as a graph. The main goal of this technique is to lessen the likelihood of collisions while also ensuring that timing requirements for industrial applications' data transmission are met.[7][13]

We learn how industrial parameters are tracked and handled in accordance with their priority from the study "An RTOS based Industrial Wireless Sensor Network using Multiprocessor Support" that was published in "International Journal of Computer Applications" (0975 - 8887)[2]. Data acquisition nodes are used to gather data from various nodes and transmit it wirelessly to the master node using the HART protocol in order to reduce data collision. The master node will carry out the work in accordance with the priority assigned to the various parameters. To monitor and manage the operations from the master node, several nodes can be used. Future experiments will use alternative operating systems like Linux and Neutrino.[14][15]

In the article "Comparison of Open Source RTOSs Using Various Performance Parameters" published in "International Journal of Electronics Communication and Computer Engineering" [3], that the scheduler is the most crucial parameter because it controls how the entire system, including hardware, functions. Extremely low interrupt and event latencies are necessary for hard real-time systems. A preventive system with an effective interrupt handling mechanism is required for this. To achieve more challenging real-time systems, the following generation of systems would need further reduced latencies and quicker thread switching. Out of the three RTOSs listed, RT Linux is the most adaptable since it offers a wide range of functionalities without sacrificing the core characteristics of an RTOS.[16]

The following is taken from the article "Handling of Priority Inversion Problem in RT Linux using Priority Ceiling Protocol" [4] that was published in the "International Journal of Advanced Engineering Research and Science (IJAERS)" We learn that numerous approaches, including interrupt disablement, priority inheritance protocol, priority remapping method, priority exchange, and modified priority ceiling protocol, can be employed to address the priority inversion issue in RT-Linux. Locking the scheduler and prohibiting context switches is the easiest solution. When the crucial zone is relatively small, it is very easy and effective to reduce the priority inversion problem. However, if the crucial zone is relatively long, deadlines for jobs with higher priority would frequently be missed. The design of the priority exchange approach requires modification of the RT Linux kernel.

IV. PROPOSED METHODOLOGY

The Industry Area and Host Area are the two primary portions of the proposed architecture's Block Diagram. Industry is represented by the transmitter side of the block diagram, which includes an ARM Processor, various input sensors, output indications like buzzers, GSM, motors, etc., and CC2500 for data transmission to the PC. The Host area, also known as the receiver side, uses a CC2500 to receive data from the PC side.

Each unit in the industrial sector needs to communicate with the ARM processor in order to measure real-time readings and take appropriate action. Both analogue and digital sensors are depicted. The LPC2148's integrated ADC converts analogue values into usable digital values for quick processing.

The Block Diagram of Industry Area is as shown in Fig.

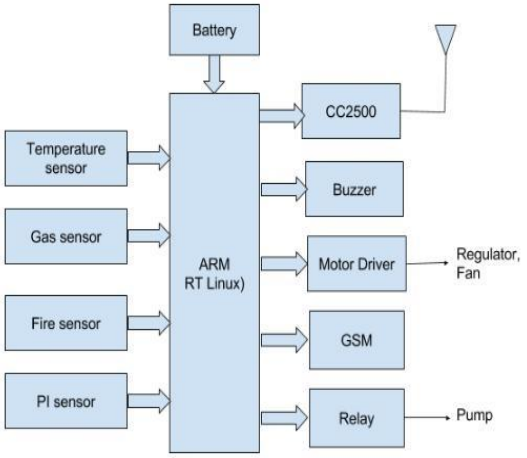


Fig. 1 Industry Area

The Block Diagram of Host Area is as shown in Fig.

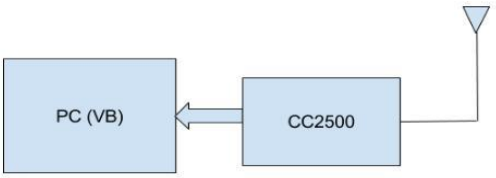


Fig. 2 Host Area

V. RTOS (REAL TIME OPERAT ING SYST EM)

A well-known definition of a real time system is as follows:
"A real-time system is one in which the timing of the results generation determines whether the system is right in addition to the logical outcome of the calculation."

The definition states that there are two sorts of real-time systems:-

Hard Real-Time:- The deadline is compulsory to be fulfilled.

Soft Real-Time:- The deadline is expected to be fulfilled but it is not always compulsory.

The concept of RTOS is not new. Different types of RTOS's are available in the market. But the selection of a proper RTOS is essential to get the maximum throughput. If an industrial application is considered, then low latency, fast task solving capabilities, hard real-time response, kernel memory specifications, multiprocessor support, inter-process communication mechanism, power requirements and cost are the major concerns.

By keeping in mind the above mentioned points, a real time version of Linux is selected, it is known as RTLinux. A hard real-time OS is RTLinux. The pre-emptive design of the Linux thread ensures that non-real-time actions never cause real-time interrupts to be postponed. Shared memory allows real-time threads to communicate with one another. This enables the system to support all currently available Linux applications. The user can also perform tasks that combine real-time and delayed chores. All Linux services, including networking, graphics, windowing systems, data analysis software, Linux device drivers, and the common POSIX API, are available to RTLinux thanks to the utilisation of shared memory.

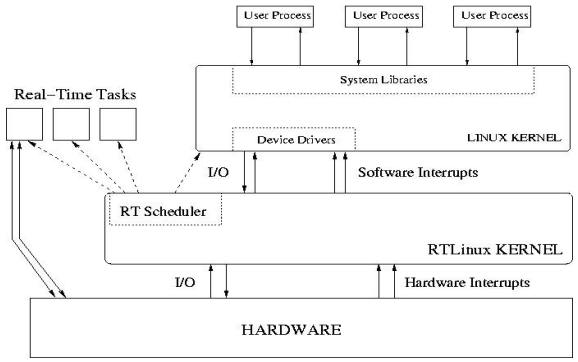


Fig. 3 Role of the RTLinux Kernel

The real-time operating system (RTOS) market lacked an effective, open source RTOS, thus the RT Linux project set out to provide one. The Linux Kernel offered a reliable foundation for achieving this objective. Linux already had a sizable user base and a substantial body of code. The Linux kernel is effective, open, and available without charge. Using the Linux kernel is practical, but it has drawbacks because Linux is a time-sharing operating system. This indicates that the interrupt latency, timing, and scheduling do not match RTOS standards. Inter-process communication is another issue brought on by the RT Linux kernel implementation.

Hard real-time OS introduced by RTAI appears to integrate some of the greatest elements of the other types. The system does this by using five complementing components. The first is the hardware abstraction layer (HAL), which offers a hardware interface for a hard real-time core and supports Linux. The second step involves integrating the new task management system into the current OS using the Linux compatibility layer, which interfaces with the Linux OS. This is an important step since hard real-time task management algorithms can differ greatly from those employed in a general purpose OS. Additionally, this layer enables the current OS to accept the new management system without detecting a change, enabling

The RT kernel and the common Linux kernel are intended to cohabit under RT Linux. The Linux kernel and non-RT processes continue to function normally in the absence of any RT processes on the system, with no interference from the RT kernel. The conventional Linux kernel typically does all that can be done in Linux, putting as little stress as possible on the RT kernel. The RT kernel pre-empts the Linux kernel and assumes control when an RT task enters the system. Data structures allow the RT kernel and the Linux kernel to communicate. This enables the inclusion of both real-time and non-real-time functionalities in programmes. Real-time interrupts are handled solely by the RT scheduler, while non-real-time interrupts are managed by the Linux kernel with the RT kernel's permission. The Linux kernel is used to manage all hardware interactions.

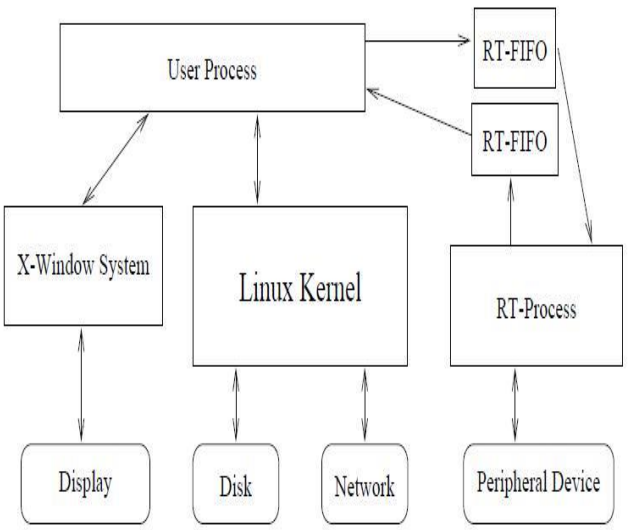


Fig. 4 Data Flow in an Application

The screenshots for the RTLinux coding are shown below:

Fig. 5 A simple Hello Program

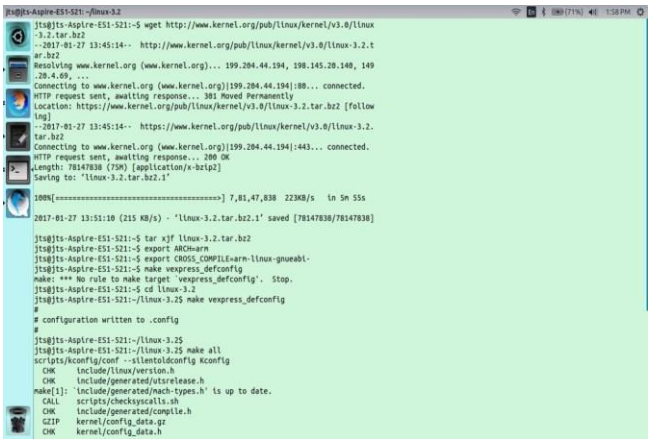


Fig. 6 First Step

Fig. 7 Second Step

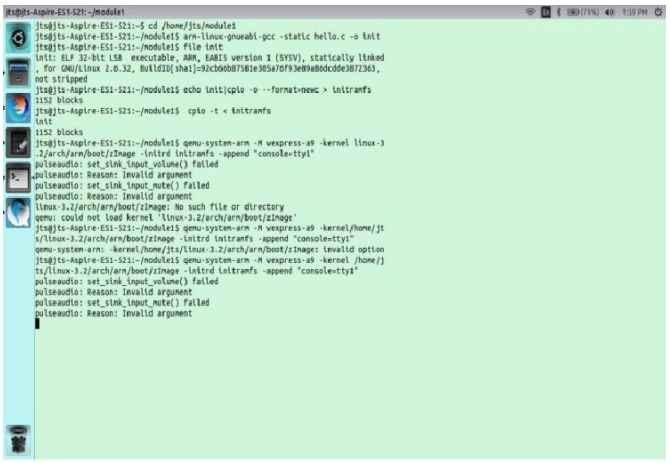


Fig. 8 Third Step

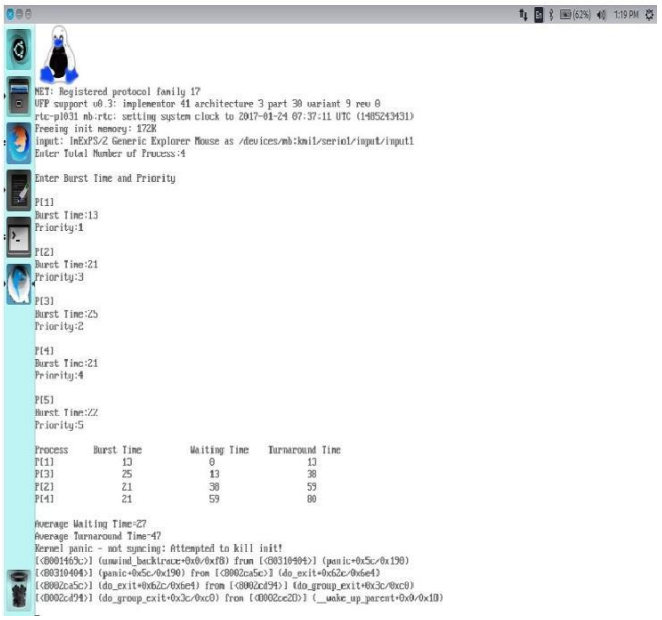


Fig. 9 Task File

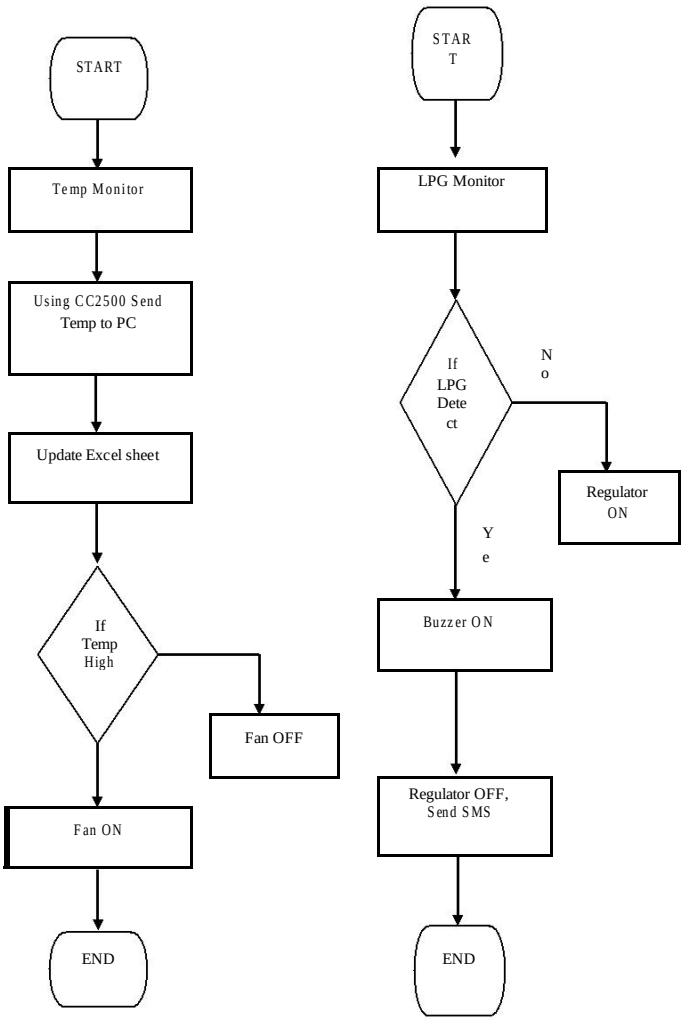
VI. HARDWARE AND SOFT WARE DESIGN

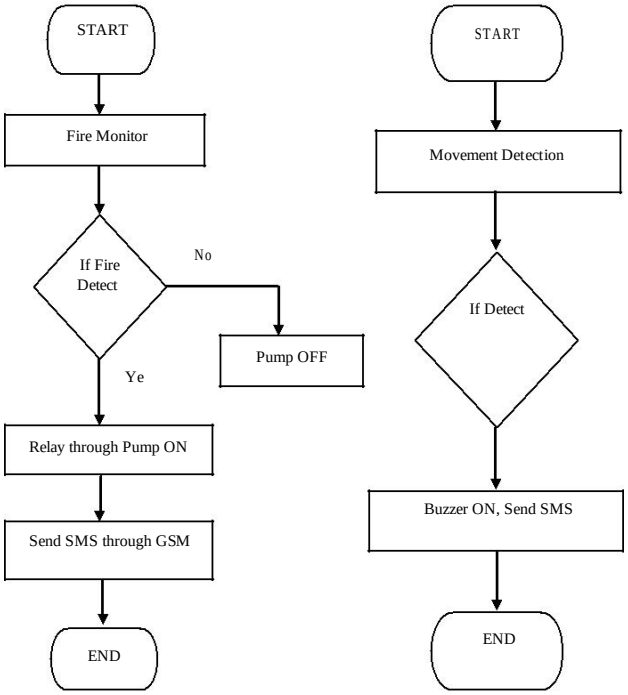
The main hardware of this project comprises of the ARM controller LPC 2148. A member of the RISC (reduced instruction set computer) family of CPUs, an ARM processor was created by Advanced RISC Machines (ARM). 32-bit and 64-bit RISC multi-core processors are produced by ARM. RISC processors are made to execute fewer different kinds of computer instructions, allowing them to run faster and handle more millions of instructions per second (MIPS). RISC processors deliver exceptional performance at a small fraction of the power consumption of CISC (complex instruction set computing) devices by eliminating unnecessary instructions and streamlining paths.

Fig. 10 ARM LPC2148 Board

The hardware requirements involves the ARM controller, different types of sensors like temperature sensor, gas sensor, PIR sensor and fire sensor, GSM, DC Motor, buzzer, CC2500, etc.

Software Flow:





Visual Basic and Environment:

You may create Windows (GUI) apps with the use of a tool called Visual Basic. The user is able to recognise the applications by their appearance. Because Visual Basic is event-driven, code sits idle until it is required to react to an event (such as a button press or menu option). The control system for Visual Basic is an event processor. Prior to the detection of an event, nothing occurs. The event procedure (the code associated with the observed event) is then run. The event processor then regains control of the programme.

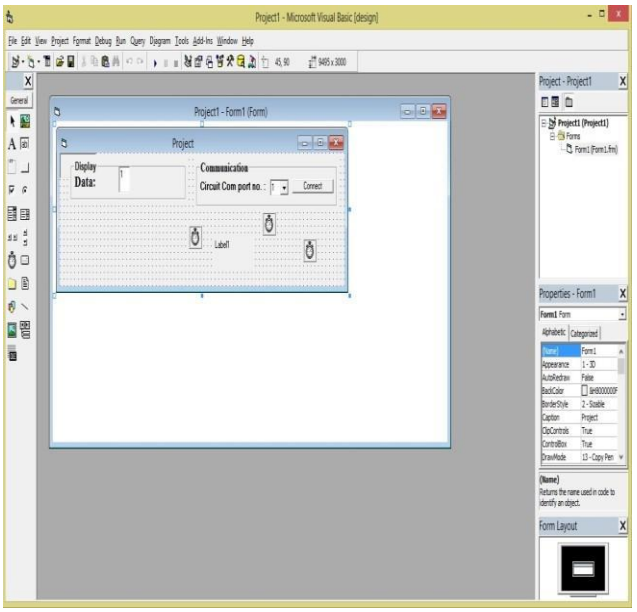


Fig. 11 Screenshot of the form in VB

VII.OBSERVATIONS AND RESULTS

In the results, we can see that the four sensors work as four tasks and then multitasking operation takes place. The controlling action takes place whenever the corresponding sensor gets a stimulus. The final proposed model is shown by figure below.

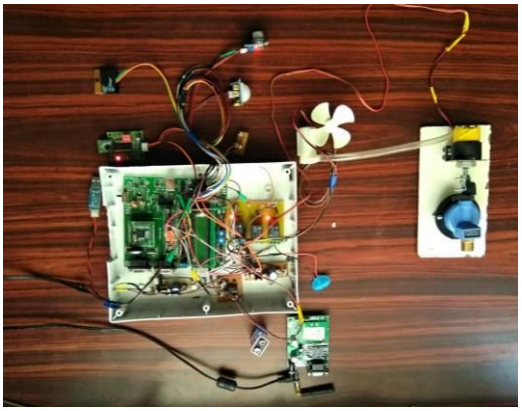
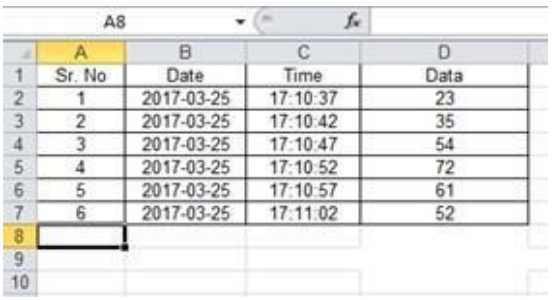


Fig. 12 Proposed Model

The GUI is developed by using Microsoft Visual Basic 6.0. The readings obtained from temperature sensor is continously updated in the excel sheet. These values are sent from the plant to the host area which is at a remote location using CC2500. The scrrenshot of Visual Basic output is as follows.



	A	B	C	D
1	Sr. No	Date	Time	Data
2	1	2017-03-25	17:10:37	23
3	2	2017-03-25	17:10:42	35
4	3	2017-03-25	17:10:47	54
5	4	2017-03-25	17:10:52	72
6	5	2017-03-25	17:10:57	61
7	6	2017-03-25	17:11:02	52
8				
9				
10				

Fig. 13 Excel file showing temperature reading

VIII.CONCLUSION AND FUT URE SCOPE

The decomposed block diagram is developed on hardware and software simultaneously. The RTLinux kernel is developed and tested for suitability of industrial applications. A User Interface using Visual Basic is developed, so that the controller can monitor the activities in an industry from a remote location. CC2500 proves to be a convenient method to transmit the data to the PC Side. Through the use of GSM, the controller can be alerted, so that timely action can be taken place if any of the sensor changes its value. Waterfall model proved to be an efficient algorithm to develop this project. In the future, the real data from sensors can be sent through network and displayed on demand through web server interfaced on board. Basically there is a variety of web page formats like PHP, .Net, HTML Page. Also, to display direct reading, board should support PHP based web page. This page can be accessed through remote terminal for controlling purpose. If external and internal memory size increases then database can be created to send the readings of sensors. For this, database support can be added to the kernel.

REFERENCES

[1] R. V. M. Silambarasan D, "Handling of Priority Inversion Problem in RT Linux using Priority Ceiling Protocol," International Journal of Advanced Engineering Research and Science (IJAERS), vol. 3, no. 6, p. 6, 2016.

[2] A. K. T. R. Aswini Bavani, "An RTOS based Industrial Wireless Sensor Network using Multiproces or Support," International Journal of Computer Applications (0975 – 8887), vol. 114, no. 4, p. 4, 2015.

[3] K. A. V. B. Rahul Sandeep, "An intelligent architecture for industrial automation using RTOS technology," International Journal of Application or Innovation in Engineering & Management (IJAIEM), vol. 3, no. 10, p. 6, 2014.

[4] D. N. N. M. Sanjay Deshmukh, "Comparison of Open Source RTOSs Using Various Performance Parameters," International Journal of Electronics Communication and Computer Engineering, vol. 2, no. 4, p. 6, 2013.

[5] M. Barabanov, A Linux based Real Time Operating System, Socorro, New Mexico: New Mexico Institute of Mining and Technology, 1997.

[6] J. P. Sam Siewert, Real-Time Embedded Systems and Components with Linux and RTOS, Dulles, VA: MERCURY LEARNING AND INFORMATION.

[7] Shende, Dipali, and Yogesh S. Angal. "Sewage water management and healthcare monitoring in IoT using Optimized deep residual network." Journal of Experimental & Theoretical Artificial Intelligence (2024): 1-24.

[8] Shende, Mrs Dipali K., and V. V. Deotare. "Internet of things enabled water quality monitoring and alert system using improved multicast routing algorithm." Design Engineering (2021): 13620-13629.

[9] Shende, Dipali K., Yogesh S. Angal, and S. C. Patil. "An iterative CrowWhale-based optimization model for energy-aware multicast routing in IoT." International Journal of Information Security and Privacy (IJISP) 16.1 (2022): 1-24.

[10] Shende, Dipali K., S. S. Sonavane, and Yogesh Angal. "A comprehensive survey of the routing

[11] Shende, D. K., and S. Smriti. "Detection of water contamination with respect to different parameters using IOT based multicast routing." J. Sci. Comput. 9 (2020): 1524-2560.Practice and Experience 21.2 (2020): 203-216.

[12] Shende, D. K., and Suryavanshi Nikhil. "IoT based

geographic multicast routing protocol with DPA through WSN." International Journal of Creative Research Thoughts 6.2 (2018): 578-584.

[13] Shende, Dipali K., and S. S. Sonavane. "CrowWhale-ETR: CrowWhale optimization algorithm for energy and trust aware multicast routing in WSN for IoT applications." Wireless Networks 26 (2020): 4011-4029.

[14] Shende, D. K., and S. Smriti. "Detection of water contamination with respect to different parameters using IOT based multicast routing." J. Sci. Comput. 9 (2020): 1524-2560.

[15] Shende, D. K., Sonawane S., "Multicast Routing for Internet of Things: A Literature Review and Challenges". In IJIACS, Volume 7, Issue 3,(2018): ISSN: 2347-8616

[16] Shende, D. K., Kale S, "Bandicoot: Drain ,Clog Detection and Prevention to support Swatch Bharat Abhiyan" , in AEGAEUM JOURNAL-UGC Care Approved Group II, Volume 8,Issue 6 (2020)