International Journal of Applied Engineering Research
ISSN 0973-4562 Volume 10, Number 12 (2015) pp. 31845-31854
© Research India Publications
http://www.ripublication.com

Forked Embedded System Models for Automotive Airbag Control System

B. C. Premkumar[1], Dr. P. Subbaiah[2], Dr. S. Ravi[3], Suresh. D[4]

[1]Ph. D Scholar, ECE Department,
Dr. M. G. R. Educational and Research Institute, Chennai
Email:pramukatech15@gmil. com
[2]Supervisor, ECE Department,
Dr. M. G. R. Educational and Research Institute, Chennai
[3]Prof &HOD ECE Department,
Dr. M. G. R. Educational and Research Institute, Chennai
[4]Prof &HOD EEE Department, Jain University, Bangalore

Abstract

To reduce the probability of missing events and achieve efficient functioning of a non realtime process, it is transformed into realtime using multiple tasks. In order to manage the various tasks, Priority based Preemptive Task Scheduling algorithms are required. Realtime approach has shown significant benefits on nonlinear control applications that need fast execution time and optimal interrupt response time. It improves the reliability and processing ability of the system. Existing kernel schemes are less suited to protect the data. Hence, the Algorithms need to be carefully designed to protect the data in real time control system. The Proposed Solution is to effectively use Critical section to protect data in preemptive RTOS systems. When critical section is executed task preemption and interrupts are disabled. It will protect the data access from other task.

Keywords: Nonlinear control, Preemptive scheduling, automotive electronics, Airbag control system

Introduction

This work focuses on the less unexplored area of RTOS based implementation of nonlinear systems suited to automotive sector. The primary focus is on the intrinsic integration of preemptive scheduling algorithms, use of multi-processor unit, client-server architecture etc. The essential focus is to develop RTOS based control system with performance metrics of improved speed and ability to process MIMO systems. There is also the objective to identify non realtime processes in a nonlinear control system and transform those processes into realtime tasks and Schedule these tasks using preemptive concepts. Also, there is a need to protect critical data during execution of multiple tasks and achieve synchronization in a timely and predictable fashion. In this paper, the realtime monitoring and control system for an air-bag system is built around a multicore processor in a master slave architecture that reads data from various sensors, removes artefacts and using realtime processing algorithms produce a timely output to the various actuators· The realtime systems support concurrent processing of multiple inputs. This involves correlated processing of multiple inputs over the same time period in a forked manner. Concurrent tasks are created and managed in order to fulfill the requirement of a realtime system.

Task scheduling is one of the major aspects of managing concurrency in realtime system.

Benefits of the work

The nonlinear system can be operated at higher sampling rates using master slave architecture. Design of partitioned hardware system enables faster response time and becomes well suited for fast motion system. Data corruption in multi-tasking RTOS system is protected using critical section implementation by disabling interrupt. Using μC/OS-II RTOS interrupt disable time is kept minimum.

Problem statement and proposed solution

Problem 1

Modern embedded solutions to problems demand discreteness rather than continuity. In continuous-time systems a proper choice of sample rate is difficult and this makes it impossible to observe the system performance properly and with such observed samples system instability occurs. Speed enhancement in MIMO/cascaded system is to be achieved with optimum CPU resource utilization. The Proposed Solution includes (i) Master Slave architecture where Slave module samples the input signal continuously and transmits to Master module process data and gives command to slave and the slave drive the output control.

Problem 2

(i) To implement a controller suited for realtime, using an embedded system, a non realtime process should be transformed to realtime task.

(ii) However embedded targets for nonlinear realtime applications are less suited for multiple inputs and multiple outputs with time constraint, as most of them are performed sequentially. Hence, the algorithm needs to be redesigned to exploit concurrency. The Proposed Solution is to transform Non realtime processes to multiple realtime tasks and use μC/OS-II RTOS programming and manage the various tasks using priority based preemptive task scheduling algorithm in RTOS.

Problem 3

(i) Design partition is a major issue in embedded system with multiprocessing units.

(ii) There is a need to analyse and solve issues like task dependence and input/output interface in design partition algorithm. The Proposed Solution is to design partition algorithms with help of Server-client modules both for wired and wireless enabled embedded system. Server is used to configure and monitor control system parameters.

Problem 4

(i) Communication protocol between processing units is major issue in embedded system with multiprocessing units. Also wire line communication is not preferred in industries like chemical processing and automotive electronics. The Proposed Solution is to use

(i) ZigBee based embedded solutions provide wireless end-point connectivity to devices or

(ii) Bluetooth

(iii) Wi-Fi or

(iv) IEEE 802. 15. 4 networking protocol for fast point-to multipoint or peer-to-peer networking.

Problem 5

(i) Inter task communication is a major constraint in multi-tasking RTOS system.

(ii) Synchronization is necessary for realtime tasks to share mutually exclusive resources. For multiple threads to communicate among themselves in time, predictable inter-task communication and synchronization mechanisms are required.

(iii) Need to redesign the existing communication methodology to improve the inter task communication. The Proposed Solution is to use message parsing interfaces schemes such as mailbox communication protocol.

Understanding System behavior using x(-t) and x(+t) for preemptive task management

To estimate a system behavior either in x(+t) or x(-t) a rotating cosmological model in which the controller can, in principle, travel to any point in the system's past as well as future, is desirable. This requires the states to be visualized as several non-interacting cubes. In relating the system behavior with the past inputs so as to predict the future states, paradox due to time travel in the past can occur. This is handled by splitting the system behavior into two parallel states, each with its own time track. Thus, if one, suppose go back to the time of a system response in state 1 (called the past state) and measure it, the system forks. The controller is now in state 1 and if desired can return to the present of state 2, a state in which the system had been mysteriously disturbed. The controller has to therefore acquire the knowledge as to

(i) How much would this state differ from the old one?

(ii) Would it find a duplicate of the cause of disturbance there?

The answer is maybe, maybe not. Some reported works assume that the slightest alteration of the past would introduce new causal chains that would have a multiplying effect and produce vast historical changes. Other works assume that system history is dominated by such powerful overall forces that even major alterations of the past would damp out and the future would soon be very much the same. The future state of a system is no problem. This just implies that ifthe output is retimed to the seventh state ahead, it just merely vanish for seven states and reappear in the future, seven state earlier than it would have been. This many state interpretations has many counter theories but still, will be very useful in designing a controller to predict the future states of a system. This is true as long as interactions do not occur. But when the controller actually travels to the past or the future cube (with multiple states) of a system, interacts with it and returns, enormous difficulties arise. Also, this approach restricts a zero memory model, since, if there is no memory for an nthcube, then the behavior cannot be sent back to states of nth cube from (n+1)th cube. Similarly, if states from (n+1)th cube are placed to form the future states of (n+2) cube, and when the states of (n+1) cube are returned, then care should be taken to check whether will it affect the states of (n+2) cube if any of the states of (n+1) is altered. There is infact no destruction of one cube to leave one alone. It is unnecessary to suppose that all but one are somehow destroyed, since all the separate elements of a superposition of cubes individually obey the system equation with complete indifference to the presence or absence ("actuality" or not) of any other elements. This total lack of effect of one branch on another also implies that no observer will ever be aware of any "splitting" process. i. e. process created by system forks in parallel are completely indifferent to the presence or absence of any other elements of other forked processes.

System behavior as time series models

A system behavior can be understood better if its output can be mapped to a time series model within the best accuracy limit. If perfect convergence exists, then it is said to be absolutely convergent and future state values are known exactly. However, most nonlinear systems exhibit conditional convergent behavior and this requires the determination of the golden ratio or value around which this convergence oscillates. For example, the number of states of a discrete system when quantized into independent sequences each of size ‘N’ bits and when forced with a behavior in which no two zeros occur consecutively (called satisfying states) varies as a Fibonacci number with ‘N’. This is given in Table 1.

Table 1:

N	Possible states in discrete sequence	No. of satisfying states
1	0 or 1	2
2	00 or 01 or 10 or 11	3
3	000 or 001 or 010 or 011 or 100 or 101 or 110 or 111	5
4	0000 or 0001 or 0010 or 0011 0100 or 0101 or 0110 or 0111 1000 or 1001 or 1010 or 1011 1100 or 1101 or 1110 or 1111	8

Airbag control system

The block diagram of a typical airbag control unit in automotive vehicle is shown in Figure 1. It includes the inflator assembly, the propellant, filters and the valve unit. The filter serves the purpose of cooling after post fired. The design improvement includes multiple inflators and modifying the output and making it a variable signal. Additionally occupancy detection sensors were intrinsically placed to avoid false alarm or dismissal.

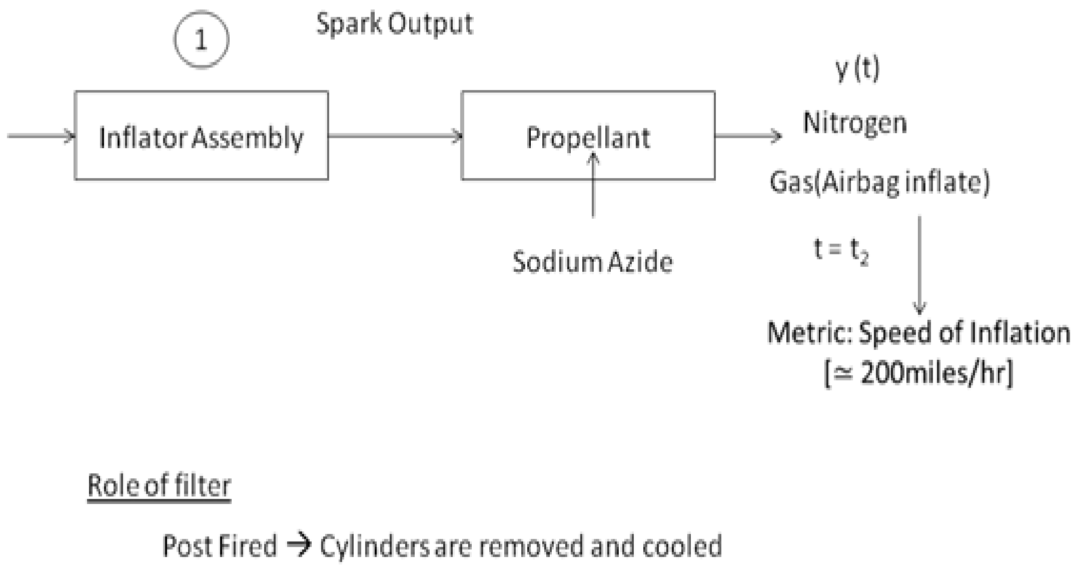


Figure: 1 Airbag control system

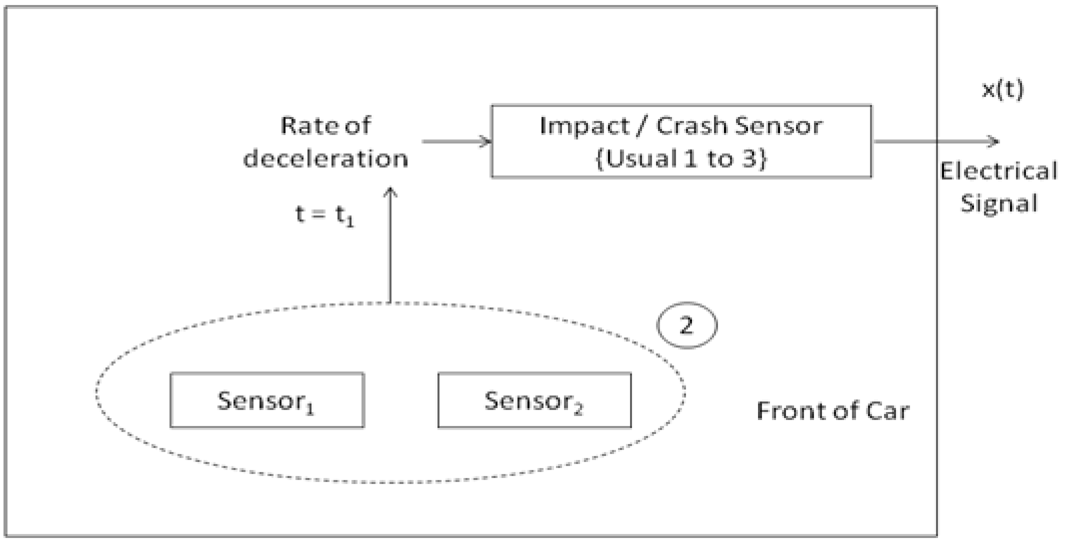


Figure: 2 Sensor section

Results and Discussion

RTOS implementation compared with non RTOS

The many state interpretation of system is best implemented using RTOS concepts and the performance of the nonlinear system is compared for both RTOS and Non-RTOS implementations. Non RTOS systems usually do not allow user programs to mask interrupts as the user program could control the CPU for as long as it wishes. RTOSs allow application itself to run in the kernel mode and permit the application to have greater control of the OS environment without requiring OS intervention. For example, if T1 is the time taken to execute task1 and T2 is the time taken to execute task2, then the total time to execute both tasks is considerably reduced in RTOS systems comparing with Non RTOS systems as shown in Table 2.

Table 2: Comparison between RTOS and Non RTOS Systems

	Task execution time	**Remarks**
RTOS	≥ Max [T1, T2]	Task execution time reduced and performance improved
Non RTOS	≥ (T1+T2)	Task execution time increased and performance degraded

The following metrics are studied in this work as shown in Table 3.

Table 3: Performance metrics

	Interrupt mechanism & synchronization schemes	**Central queue based EDF scheduling**	**Realtime comparison of RM & EDF**
Metrics included	Throughput	Packet loss	Best case response time
	Interrupt latency	Queuing delay	Worst case response time
	Interrupt response	Packet arrival rate	Response time jitter
	Task execution time	Throughput	Latency
	CPU utilization time		

From Table4 and Table 5, it is observed that task execution time is reduced for RTOS based system comparing to Non RTOS based system considerably and improves system performance by solving shared resource problem and handling critical section effectively.

Table4: Task execution time for RTOS implementation

Task number	**Macros**	**Task Entity**	**Execution time (sec)**
1	T T1 (Uctsk_Get time)	Realtime clock	User dependent
2	T2 (Uctsk_Dispdata)	Display RTC	1
		Interrupt control	1

Table 5: Task execution time for non RTOS implementation

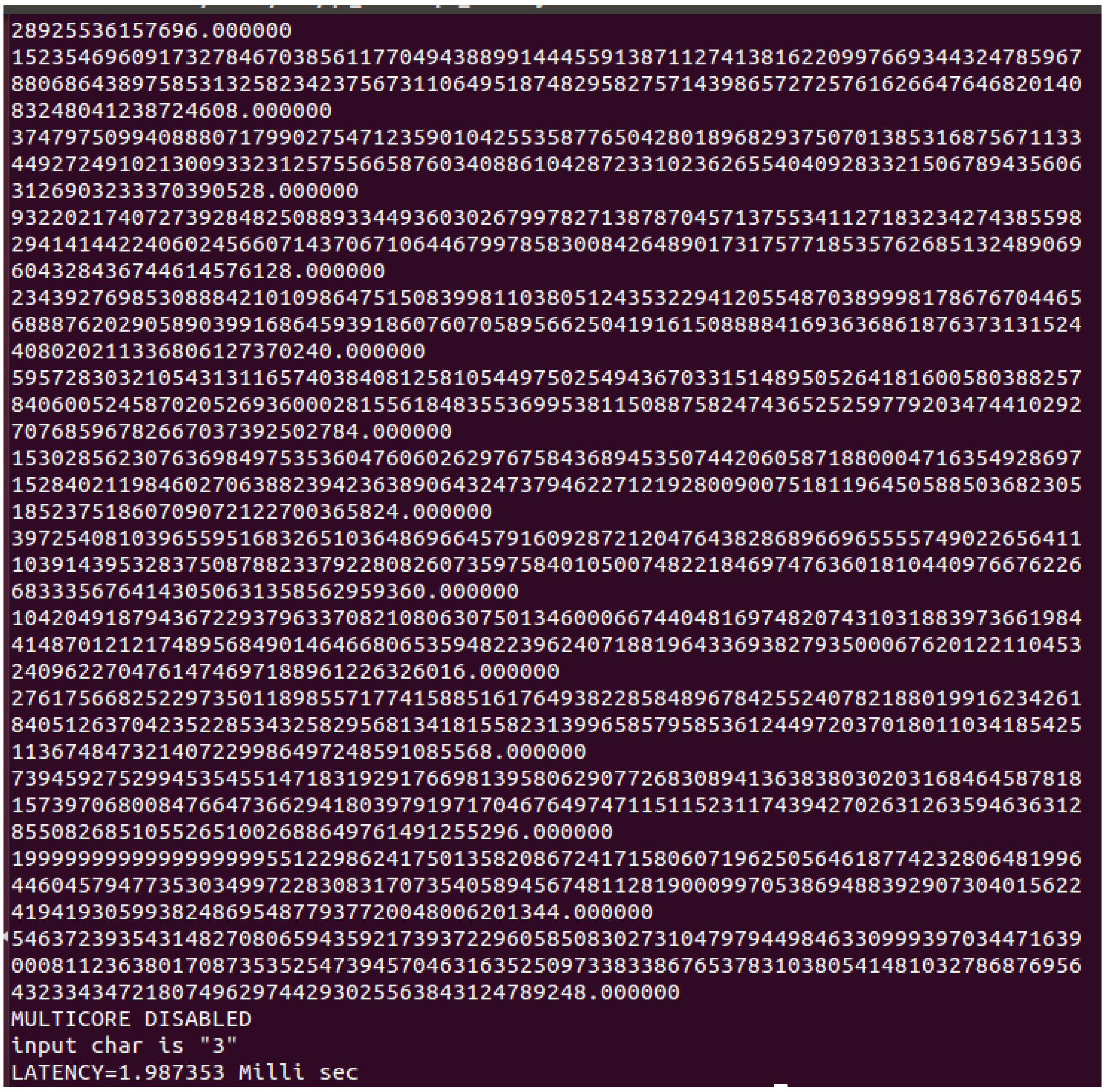
Task number	**Macros**	**Task Entity**	**Execution time (sec)**
1	T T1 (Uctsk_Get time)	Realtime clock	User dependent
2	T2 (Uctsk_Dispdata)	Display RTC	1
		Interrupt control	3

It is observed from the table shown below that system performance is improved by concurrent task handling using kernel provided calls with the enhancement in task execution time.

Table 6:Task execution time for data acquisition system implementation

Task number	**Macros**	**Task Entity**	**Execution time (sec)**
1	T T1 (Uctsk_Get ADC)	Read ADC value	0. 5
2	T2 (Uc_tsk_Disp_ADC)	Display ADC value	0. 5

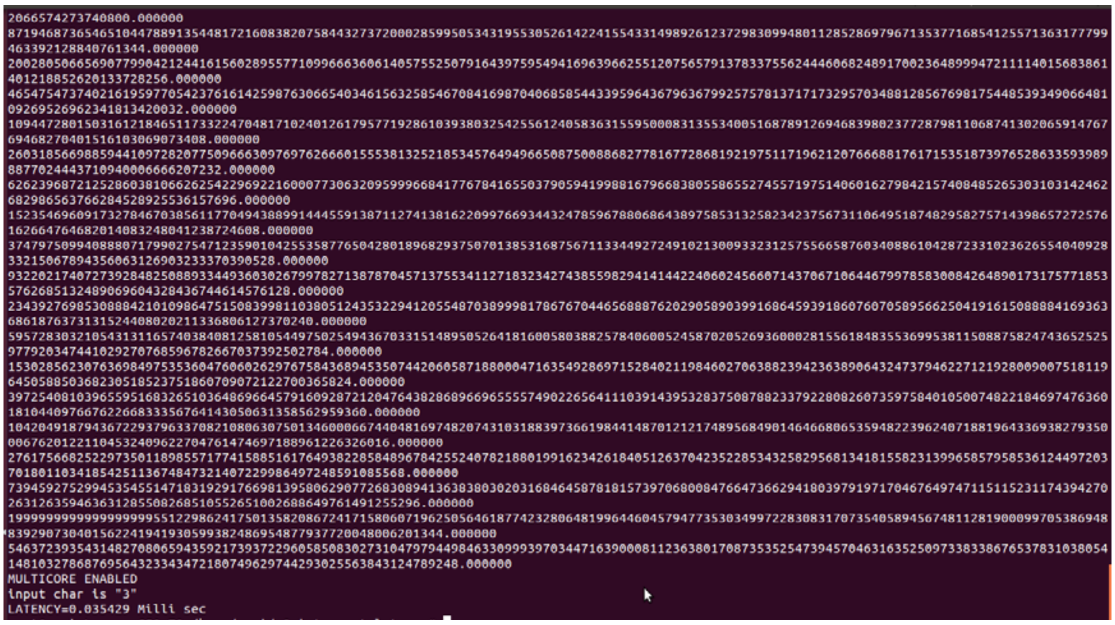
The results are obtained in multicore platform with linux kernel. Single state implementation of analyzing a nonlinear system is shown in Figure 3.



```
28925536157696.000000
15235469609173278467038561177049438899144455913871127413816220997669344324785967
```

Figure 3: Single state implementation of analyzing a nonlinear system

Multistate implementation of nonlinear system is shown in Figure 4.



```
2066574273740800.000000
871946873654651044788913544817216083820758443273720002859950534319553052614224155433149892612372983099480112852869796713537716854125571363177799
463392128840761344.000000
2002805066569077990421244161560289557710996663606140575552507916439759549416963966255120756579137833755624446068248917002364899947211114015683861
401218852620133728256.000000
4654754737402161959770542376161425987630665403461563258546708416987040685854433959643679636799257578137171732957034881285676981754485393490664 81
09269526962341813420032.000000
1094472801503161218465117332247048171024012617957719286103938032542556124058363155950008313553400516878912694683980237728798110687413020659147 67
69468270401516103069073408.000000
2603185669885944109728207750966630976976266601555381325218534576494966508750088682778167728681921975117196212076668817617153518739765286335939 89
887702444371094000666207232.000000
6262396872125286038106626254229692216000773063209599966841776784165503790594199881679668380558655274557197514060162798421574084852653031031424 62
682986563766284528925536157696.000000
1523546960917327846703856117704943889914445591387112741381622099766934432478596788068643897585313258234237567311064951874829582757143986572725 76
162664764682014083248041238724608.000000
3747975099408880717990275471235901042553587765042801896829375070138531687567113344927249102130093323125755566587603408861042872331023626554040 928
33215067894356063126903233370390528.000000
932202174072739284825088933449360302679978271387870457137553411271832342743855982941414422406024566071437067106446799785830084264890173175771853
5762685132489069604328436744614576128.000000
2343927698530888421010986475150839981103805124353229412055487038999817867670446568887620290589039916864593918607607058956625041916150888841693 63
6861876373131524408020211336806127370240.000000
5957283032105431311657403840812581054497502549436703315148950526418160058038825784060005245870205269360002815561848355369953811508875824743652 525
977920347441029270768596782667037392502784.000000
153028562307636984975353604760602629767584368945350744206058718800047163549286971528402119846027063882394236389064324737946227121928009007518119
645058850368230518523751860709072122700365824.000000
397254081039655951683265103648696645791609287212047643828689669655557490226564111039143953283750878823379228082607359758401050074822184697476360
18104409766762266833356764143050631358562959360.000000
104204918794367229379633708210806307501346000667440481697482074310318839736619844148701212174895684901464668065359482239624071881964336938279350
00676201221104532409622704761474697188961226326016.000000
276175668252297350118985571774158851617649382285848967842552407821880199162342618405126370423522853432582956813418155823139965857958536124497203
7018011034185425113674847321407229986497248591085568.000000
7394592752994535455147183192917669813958062907726830894136383803020316846458781815739706800084766473662941803979197170467649747115115231174394270
263126359463631285508268510552651002688649761491255296.000000
1999999999999999999955122986241750135820867241715806071962505646187742328064819964460457947735303499722830831707354058945674811281900099705386948
839290730401562241941930599382486954877937720048006201344.000000
5463723935431482708065943592173937229605850830273104797944984633099939703447163900081123638017087353525473945704631635250973383386765378310380 54
14810327868769564323343472180749629744293025563843124789248.000000
MULTICORE ENABLED
input char is "3"
LATENCY=0.035429 Milli sec
```

Figure 4: Multistate implementation of analyzing a nonlinear system

Conclusion

In this work, the single state implementation and the multi state implementation for correlating the system output with x(-t) and x(+t) is demonstrated. The multi state implementation offers low latency in returning to the parent state, once recalled from the superposition of past states. A realtime implementation is applied to the airbag control system using master-slave architecture. The forking based state space design enables the prediction of future states by travelling in past time. These forked states are implemented in individual cores. The results show an improvement in the task execution time with reduced number of user dependencies in the RTOS implementation.

References

[1] Leyuan Liu, Weiqiang Kong and Akira Fukuda, 2014, "Implementation and Experiments of a Distributed SMT Solving Environment", International Journal on Computer Science and Engineering (IJCSE), Vol. 6, No. 3

[2] Choi, JM, Ahn, BK & You-Sung Cha, YS 2006, 'Remote-controlled home robot server with ZigBee sensor network', *SICE-ICAS, 18-21 Oct. 2006, Bexco, Busan, Korea,* IEEE

[3] Govindaraju, K, Boopathi, S, Parvez Ahmed, F, Thulasi Ram, S, Jagadeeshraja, M 2014, 'Embedded Based Vehicle Speed Control System Using Wireless Technology', *International Journal of Innovative Research in*

Electrical, Electronic Instrumentation and Control Engineering, vol. 2, no. 8, pp.. 1841-1844

[4] Il-Kyu Hwang & Jin-Wook Baek 2007, Wireless Access Monitoring and Control System based on Digital Door Lock', *IEEE Transactions on Consumer Electronics*, vol. 54, pp. 1724-1730

[5] Indersain, Sharma, N & Dushyant Singh 2013, 'Design and implementation of μ c/Os II based embedded system using arm controller', *International Journal of Engineering and Technical Research,* vol. 1, no. 2, pp. 1-4

[6] Jae Hwan Koh & ByoungWook Choi 2013, 'Real-time Performance of Real-time Mechanisms for RTAI and Xenomai in Various Running Conditions', *International Journal of Control and Automation,* vol. 6, no. 1, pp. 235-245

[7] Jinsoo Han, Haeryong Lee & Kwang-Roh Park 2009, 'Remote-Controllable and Energy-Saving Room Architecture based on ZigBee Communication', *IEEE Transactions on Consumer Electronics*, vol. 55, no. 1, pp. 264-268

[8] Joon Heo, Choong Seon Hong, Seok Bong Kang & Sang Soo Jeon 2008, 'Design and Implementation of Control Mechanism for Standby Power Reduction', *IEEE Transactions on Consumer Electronics*, vol. 54, no. 1, pp. 179-185

[9] José Araújo, Manuel Mazo, Adolfo Anta, Paulo Tabuada & Johansson, KH 2014, 'System Architectures, Protocols and Algorithms for Aperiodic Wireless Control Systems', *IEEE Transactions on Industrial Informatics*, vol. 10, no. 1, pp. 175-184

[10] Karthikeyan, V, Dr. S. Ravi and Dr. M. Anand, 2014, 'Robust Memory Management using Real time concepts' *Journal of computer science*, vol. 10, no. 9, pp. 1480-1487

[11] Kathleen Baynes, Chris Collins, Eric Fiterman, Brinda Ganesh, Paul Kohout, Christine Smit, Tiebing Zhang & Bruce 2003, 'The Performance and Energy Consumption of Embedded Real-Time Operating Systems', *IEEE Transactions on Computers*, vol. 52,no. 11, pp. 1454-1469

[12] Rinku, DR & Mohd arshad 2013, 'Design and Implementation of Free RTOS Based Online Data Acquisition and Controlling System Using Cortex M3 Core', *International Journal of Engineering Science & Advanced Technology,* vol. 3, no. 5, pp. 259-263

[13] Xiaohong Peng & Guodong Liu 2012, 'Intelligent Water-saving Irrigation System Based on Fuzzy Control and Wireless Sensor Network', *Fourth International Conference on Digital Home, 23-25 Nov. 2012, Guangzhou,* IEEE, pp. 252-256

[14] Zhenghua Xin, Yongliang Guo & Liangyi Hu 2013, 'Research on the Implementation of Porting uC/OS-II on the STM32F103 chip, *Telkomnika Indonesian Journal of Electrical Engineering,* vol. 11, no. 7, pp. 3897-3904