

Performance Improvement In Nonlinear Control System Using RTOS

B. C. Premkumar.¹,

¹Ph. D Scholar, ECE Department Dr. M. G. R. Educational and Research Institute Dr. MGR University, Chennai
Email: pramukhatech15@gmail.com

Dr. P. Subbaiah.²,

²Supervisor, Principal, Sri Sunflower College of Engineering & Technology, Lankapalli, AP

Dr. S. Ravi.³,

³Prof &HOD ECE Department, Dr. M. G. R. Educational and Research Institute, Dr. MGR University, Chennai

Suresh D.⁴

⁴Professor, EEE Department, Jain University, Bangalore

Abstract

In this paper, processes that are presently non real-time in an existing control system are identified and transformed into real-time. Real time approach has shown significant benefits on process control applications and offer higher sampling rates, fast execution time and optimal interrupt response time. Long term embedded solutions to problems demand discreteness in precise form. However stand alone embedded targets for nonlinear Real-time implementations are less suited for multiple inputs and multiple outputs with time constraint, as most of algorithms are still performed sequentially. Hence, the algorithm needs to be redesigned to exploit concurrency. A Hardware/software is the solution proposed for nonlinear control system to raise the system performance and uses master slave architecture is used. It is inferred that RTOS based embedded system with hardware-software design can be managed critical section and multiple tasks

Keywords: Non linear time, Real Time, Master-Slave Architecture

Introduction

This work focuses on developing RTOS based multi process unit in a control system. A hardware implementation in MIMO/cascaded systems that improve the system processing speed is proposed. For example the Methodology is to Identify non real-time processes in a control system and transform those processes into real-time tasks. Schedule these tasks using preemptive concepts.

1. 1. Benefits of the research work

The nonlinear system can be operated at higher sampling rates using master slave architecture. Design of partitioned hardware system can be implemented with faster response time and becomes well suited for fast motion system. Reduce the probability of missing events. For efficient functioning of control system a non realtime process is transformed into realtime using multiple tasks. In order to manage the various

tasks, Priority based Preemptive Task Scheduling algorithm in RTOS is used. Realtime approach has shown significant benefits on process control applications that need fast execution time and optimal interrupt response time. It improves the reliability and processing ability of the system.

Problem statement and proposed solution

Problem 1

- Modern embedded solutions to problems demand discreteness rather than continuity and discussion on discreteness.
- 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.

Proposed Solution

Master Slave architecture is proposed for continuous-time systems. Where Slave TTCAN implementation module samples the input signal continuously and transmits to Master module process data and gives command to slave and the slave drive output control.

Problem 2

However embedded targets for nonlinear Real-time applicants 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.

1. 2. Implementation of the proposed work

The purpose of this work is effective hardware implementation of multiprocessor unit in an embedded control system so that the system can be operated instantaneously. This system is suitable for fast motion system. In this work real-time transformation of an existing non real-time process control system is proposed.

2. LITERATURE SURVEY

Yan Xijun et al. (2009) has proposed a high performance automatic irrigation system using low-power technology and wireless sensor network. The node gathers the information such as water-level, gate position and rainfall and the sink node receives the real-time data. The information centre stores and processes those data which are transmitted from the sink node through the GPRS network. It is replaced from wired transmission into a wireless network. This network reduces the system's cost, enhance the system's extension and improve the system's performance. The system can be used in water resources dispatch, flood prevention since it has a high stability.

Rinku and Mohdarshad (2013) have attempted Free-RTOS based online data acquisition and controlling system using Cortex-M3 core. It has Ethernet device for Cortex-M3 core to transmit the monitored sensor data to internet. The System support remote monitoring and maintenance operations of equipment through the network using Web browser.

Shinde and PravinZaware (2014) proposed the design and prototype implementation of new Wi-Fi technology based monitoring and controlling the field equipments (or) machines using Phone, Tablet Laptop (or) and desktop pc having Wi-Fi utility with IEEE 802. 11b, g and n standards. The proposed system is divided into two parts for better implementation; the first part is monitoring the embedded system (field equipments) data and the second part controls the field equipments. It is a very easy, reliable, secure, fast wireless solution.

Fu Zhang and Kang (2003) have designed an embedded signal acquisition system for real time according to the mechanical failure which occurred with high frequency in the rotating machines. The system is based on a low cost.

low cost and high sensitivity, and the acquisition data from axis x, y, and z. The system analyses the working principle of data acquisition module. The proposed system has $\mu\text{C}/\text{OS-II}$ to realize the data task management and scheduling, and its compacted with structure and low cost.

Ajith Singh et al. (2010) has proposed a round robin scheduling algorithm to improve the CPU efficiency in real time and time sharing operating system. The proposed algorithm is superior as it has less waiting response time, usually less pre-emption and context switching thereby reducing the overhead and saving of memory space.

José Araújo et al. (2014) has proposed a periodic implementation of control systems that are specially designed for wireless sensor and actuator networks (WSANs). Also he identified limitations of the current IEEE 802. 15. 4 MAC protocol and proposed some modifications to increase its flexibility and enable our implementations. The result shows that the efficient communication mechanisms implemented with respect to control and communications performance.

2. 1. Inference from literature review

Active research work is required only on multi-tasking preemptive scheduling algorithms, design partition for multi target embedded system, wireless communication between client and server system and their performance improvement. Inter task communication is also a major constraint in multi-tasking RTOS system. Master and slave modules can communicate using high speed serial interface and plug & play bus protocol and make the control system faster and scalable.

Table 1. Performance metrics

Sl. No.	METRICS	DEFINITIONS	REMARKS
1.	Through put	Number of task executive in a unit amount of time.	Should be high
2.	Context Switch	Storing and restoring state of a preempted process	Should be low
3.	Turnaround Time	Time taken to complete the process and how long it takes the time to execute that process. Turnaround time = Time of process completion – Time of process submission.	Should be low
4.	Waiting Time	Waiting time = Time of process scheduled – Time of process ready to executes	Should be low
5.	Interrupt Latency	The amount of time interrupts are disabled.	Should be low
6.	Response Time	Response time = Time of response received – Time task submitted	Response should be quick

microcontroller STM32, Vibration signals are picked by the three axis acceleration sensor which has the performance of

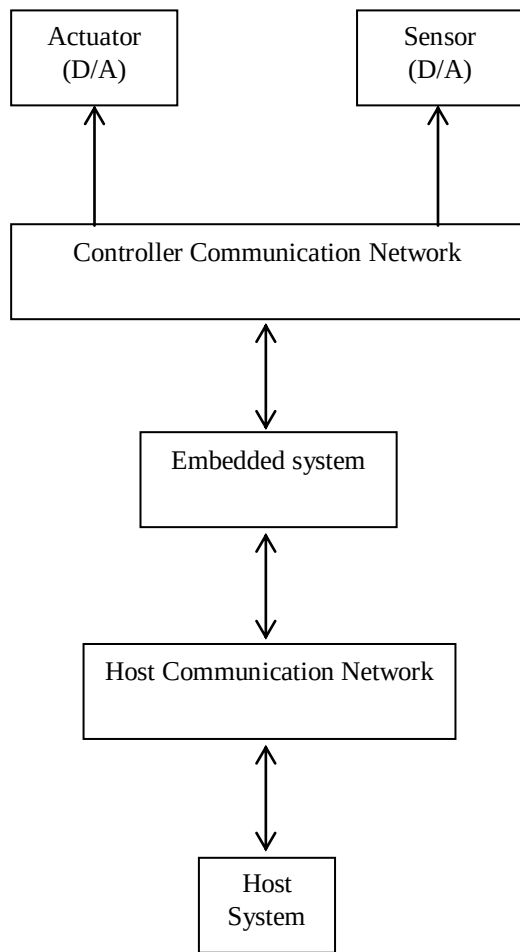


Figure 1: Block diagram of Network Control System

3. REALTIME SYSTEM

A real-time system is one whose applications are mission-critical, where real-time tasks should be scheduled to be completed before their deadlines. Most real-time systems control unpredictable environments and need operating systems that can handle unknown and dynamic tasks. Example: Determining the port/multiple users in a UDP protocol.

In real time systems each task should be invoked after the ready time and must be completed before its deadline and is achieved in this work. The real-time system performance can be improved by real-time operating system (RTOS). There are two main types of real-time systems,

- a. Hard Real-Time System
- b. Firm (or) Soft Real-Time System

Difference between hard and soft real time system is shown in Table 2.

Table 2. Hard and Soft real time system Comparison

Soft Real-Time System	Hard Real-Time System
Missing an occasional deadline is undesirable, but nevertheless tolerable.	Fixed deadlines must be met
Performance is degraded but system is not destroyed by failure to meet response time constraints. (e. x) online data bases	System is destroyed by failure to meet response time constraints, (e. x) flight control system

The various RTOS features are shown in Table 3.

Table 3. RTOS features

Metric	Definitions
Synchronization	To share mutually exclusive resources.
Interrupt Handling	Time delay between the occurrence of an interrupt and the running of the corresponding Interrupt Service Routine (ISR).
Timer and clock	Vital part of every real-time operating system
Real-Time Priority Levels	Once a priority value is assigned to a task, the operating system does not change it by itself.
Fast Task Preemption	Whenever a high priority critical task arrives, an executing low priority task should be made to instantly yield the CPU to it.
Memory Management	RTOS uses small memory size by including only the necessary functionality for an application while discarding the rest.
Predictable Response	Execution time of all system function calls is predictable.
Dynamic Scheduling	Scheduler decision points occur a regular clock frequency.
Double linked list	Tasks of the same priority are organized.
Scheduler suspension	When exclusive access to the MCU is required without jeopardizing the operation of ISRs.
Blocking and deadlock avoidance	Non-transitive approach deadlock example (team round robin schedule drawback)

4. TTCAN

In this work, a TTCAN (time-triggered CAN) protocol is built a higher layer protocol on top of the CAN (Controller Area Network) data link layer. The benefits include

- 1) Triggering activities by the elapsing of time segments.
- 2) All points of time of message transmission are defined during the system development.
- 3) Suits applications with periodic nature of data traffic

The higher-layer protocol is done without changing the standard CAN protocol using synchronization and scheduler for all existing CAN nodes in a network using global system. Thus if the synchronized communication & schedule is delinked, the TTCAN nodes get converted standard CAN protocol.

Advantages of Integrating higher layer protocol

CAN messages are scheduled both in a time-triggered & event-triggered way. This suits vehicle data traffic since, in vehicles, data traffic must usually be both event.

Two layer Architecture Implementation

Typical C_CAN consists of the components CAN Core, Message RAM, Message Handler, Control Registers, and Module Interface as shown in the Figure. 2

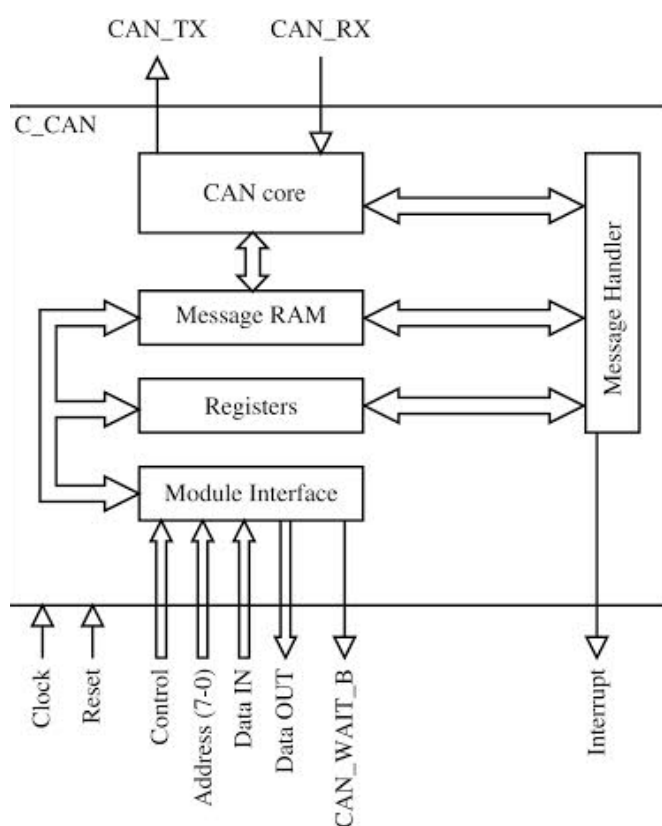


Figure 2: Block Diagram of C_CAN

Compared to the C_CAN, the TT_CAN is expanded by two functional blocks as shown in the following figure 3, the Trigger Memory and the Frame Synchronization Entity FSE. The TTCAN implementation is done in two steps. In the first step, only level 1 is implemented, without the global system time and drift compensation of level2. In the second step, after the evaluation of the TT_CAN IP module in a TTCAN network, a global time unit will be added to the module.

Optimization Requirements

1. Safety

2. Comfort
3. Pollution prevention
4. Low cost
5. Reducing the number of wire harnesses
6. Transferring large amount of data at high speed via multiple LAN's.

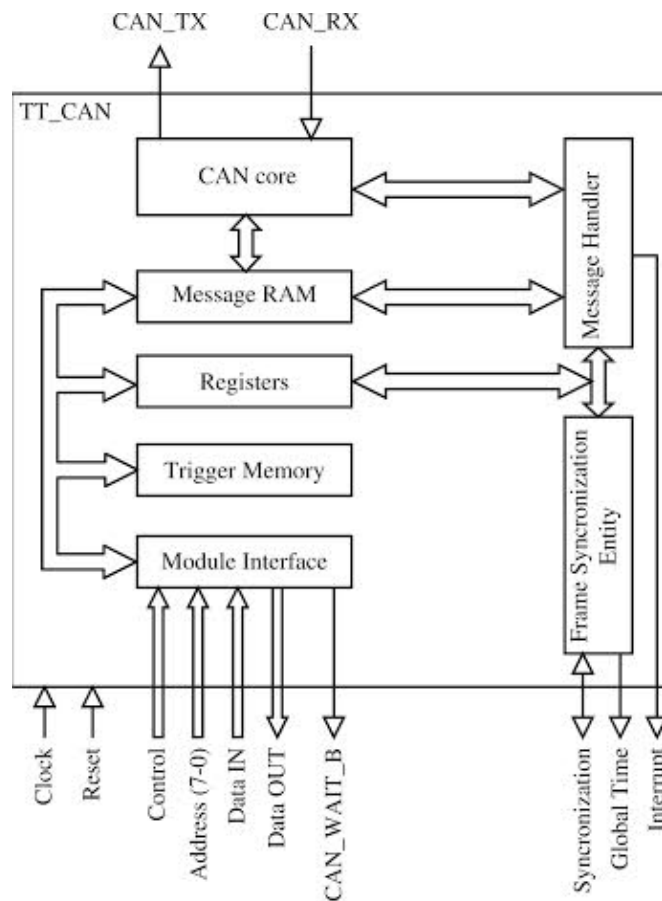


Figure 3: Block Diagram of TT_CAN

Table 4. Functions of CAN

Sl no	Terms	Functions
1	CAN Core	communication performs according to the CAN protocol version 2.0,
2	Message RAM	Message Objects and Identifier Masks for acceptance filtering of received messages are stored
3	Registers	Message Objects and Identifier Masks for acceptance filtering of received messages are stored
4	Module Interface	It is controlled by many control signals
5	Message Handler	All functions concerning the handling of messages are implemented

Table 5. Important Features of Existing vehicle control protocol

Sl no	Terms	Remarks
1	Multimaster	Regulates units connection depending on bus availability
2	Message transmission	Allows messages to be transmitted in predetermined format
3	System flexibility	Re-Configurability is introduced
4	Communication speed	Maximum Signal frequency is fixed and is a performance matrix in session layer
5	Remote error recovery request	Converts the data in physical layer to a form suitable for data link layer
6	Error detection	Performs channel coding to handle errors during routing and is a performance matrix in physical layer and DLL
7	Connection	multiple units can be connected concurrently with master slave architecture
8	Length	Network layer performance metric
9	Maximum latency	Performance metric is session layer

Table 6. Two dedicated wires CAN_H & CAN_L

no.	Status	CAN _H	CAN _L	$\Delta V = [CAN_H - CAN_L]$	Advantages
1.	Idle	2. 5v	2. 5V	0	Insensitive to inductive spikes
2.	Transmission	3. 75v	1. 25v	2. 5v	Sensitive to inductive spikes

CAN benefits

- low cost
- Broadcast Communication scalable & includes plugging of intelligent non-transmitting nodes dynamically.
- Priority based
- Cyclic Redundancy Code (CRC) 32 bit CRC intrinsically generated using hardware.

Slave –master communication

First alternative of passing data is when the slave receives out a complete message frame i. e. both header and response for one or more slaves and is usually referred to as a command frame (CMD frame). The CAN benefits are illustrated in Figure4 and Data transfer from slave to master is presented in figure 5

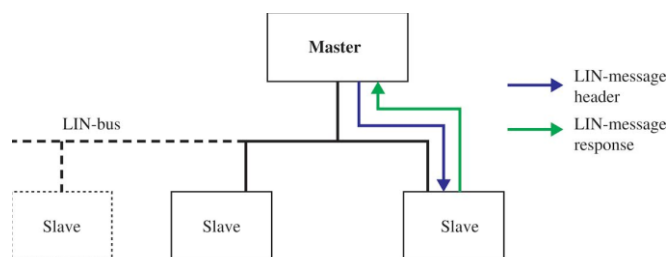


Figure: 5 Data from Slave to Master

CAN Benefits



Figure : 4 CAN Benefits

The slave nodes on the network are addressed by the ID- field. The nodes do not have physical addresses (e. g. MAC - address or any other hardware address) but instead use a pre-programmed list of valid IDs in memory that they use to filter out which messages to respond to. This way the ID can have different meaning for different nodes and it enables3 ways of passing data between the master and its slaves:

Slave –slave(s) communication: the last alternative is when one slave sends its response to one or more slaves and can be used when there is no need for data processing by the master e. g. information sent between sensor and actuator.

The features of LIN and CAN are compared in Table 7. LIN has drawback of lower bandwidth and also less effective bus access scheme with themaster- slave configuration.

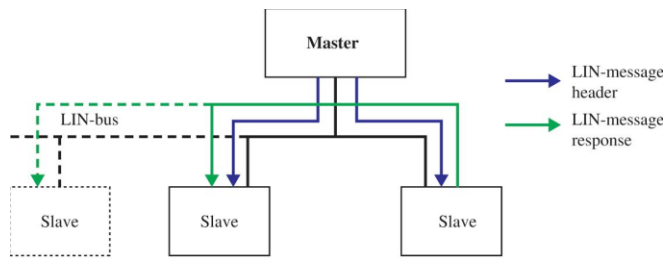
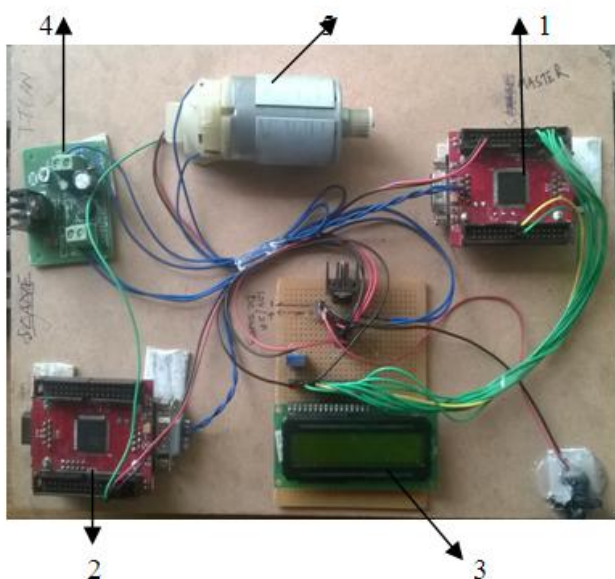


Figure : 6 Data from Slave to Slave

Table 7. LIN and CAN Features

Features	LIN	CAN
Media Access	Single Master	Multiple Masters
Bus Speed	2, 4, 9, 6 and 19.2 Kbps	Up to 1 Mbps
Multicast Message Routing	6 bit identifier	11/29 bit identifier
Network Size	2 to 16 nodes	4 to 20 nodes
Data bytes per frame	0 to 8	2 to 8
Transmission time for 4 data bytes	6 ms at 19.2Kbps	0.8 ms at 125Kbps
Error detection(Data field)	8 bit checksum	15 bit CRC
Physical Layer	Single wire, 12 v, 40 meter bus length.	Shielded, Twisted pair, 5V
Quartz/Ceramic resonator	For Master only	For all nodes

. Hardware Implementation



The Hardware consists of the following Modules: 1. Master CPU, 2. Slave CPU, 3. Display 4. TTCAN Based Control Interface 5. Engine

Figure 7. Hardware Implementation setup

The Engine speed control and monitoring is done by the slave controller. Whenever, the master controller requests for the value of Engine speed, a trigger is given to the slave in a specified time slot and in response to the trigger, the slave places the speed data on the bus along with its node ID. The timing diagram is shown in figure 8

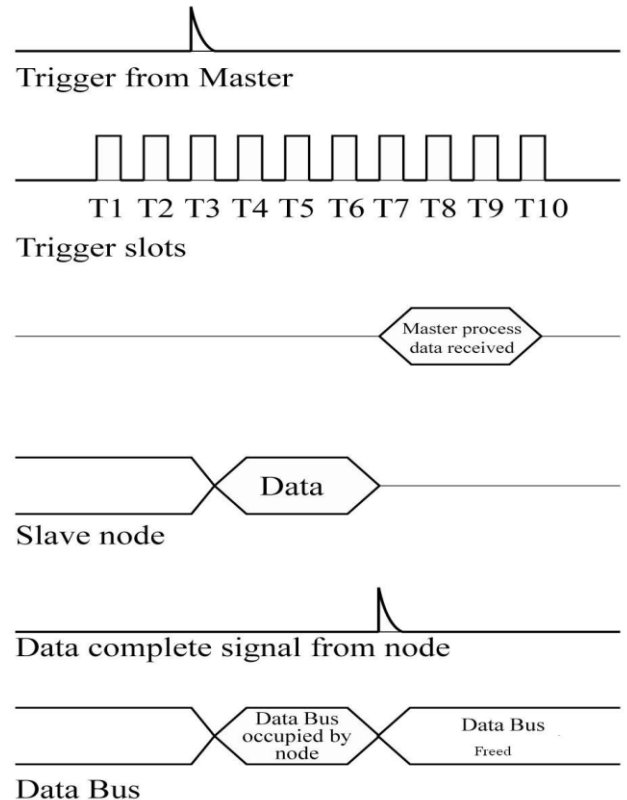


Figure 8. Input and output waveforms

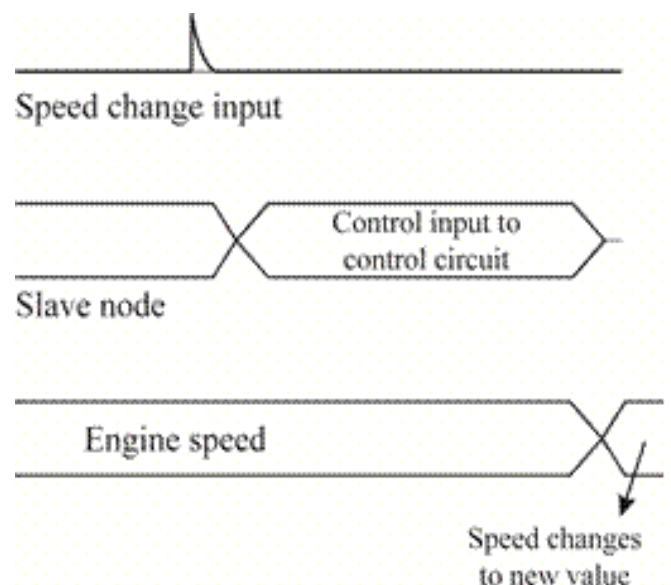


Figure 9 Request for speed change from user

Advantages

Master process unaffected due to different task states on slave side. Each task has multiple states like ready, wait, dormant, etc. and the behavior & effect of these states are localized to the slave and does not affect the states of the task interacting with master node.

5. RESULTS AND DISCUSSION

Placing Task priorities in a Queue using Scheduler to satisfy condition 1.

Condition 1: Let there be two groups of tasks. These tasks are to be placed in a queue such that at any point in the queue, the Task A should be executed more in number compared to Task B. Scheduler does this task allocation into the queue.

For Example: If there are 6 Tasks, 3-from Task A & 3- from Task B.

Let Task A be assigned bit 1

Task B be assigned bit 0

Then, the scheduler, schedules the Tasks in any one of the five following ways. The scheduled tasks and their coded implementation is shown in Table 8. This result can be made generic, and has fourteen ways for 8-Tasks [4 each from A & B]. The number of possible ways of arranging queue to satisfy condition (1) is shown in Table 9. The results obtained in realtime is shown in Figure 7.

Table 8. Scheduled Tasks and their implementation

Possibility	Scheduled tasks	Coded implementation
1	Task A Task B Task A Task B Task A Task B	101010
2	Task A Task A Task B Task B Task A Task B	110010
3	Task A Task A Task A Task B Task B Task B	111000
4	Task A Task A Task B Task A Task B Task B	110100
5	Task A Task B Task A Task A Task B Task B	101100

Table 9. Arranging Tasks in Queue

n	1 2 3 4 5 6
P(n) possible ways of arranging tasks In queue to satisfy condition (1)	1 2 5 14 42 132

6. CONCLUSIONS AND FUTURE ENHANCEMENT

A Hardware/software is the solution proposed for nonlinear control system to raise the performance of the system. The research of hardware design is to enhance the response time of control system. For this purpose master slave architecture is used. Data acquisition logic and control algorithm is driven by

slave module and master module respectively. The data acquisition rate is increased greatly and it is used for fast motion system. All the non-real time activities have been effectively transformed into real time using task based implementation to enhance the speed. It is established that RTOS based embedded system can manage any critical section containing multiple tasks. To support data at high sample rate the System performance can be improved by redesign processing algorithm to improve the processing speed. This includes moving control driving logic to slave module for high speed SPI interface. The improvement of system performance is achieved through redesigning critical section algorithm & scheduler.

REFERENCES

- [1]. AswinSayeraman, Ramesh, PS, 2012, ZigBee and GSM Based Secure Vehicle Parking Management and Reservation System, Journal of Theoretical and Applied Information Technology, vol. 37, no. 2, pp. 199-203
- [2]. Franklin Richard, M, SudheerSukumaran, S, 2012, A Real Time Industrial Automation System with Task Scheduling International Journal of Computer Science and Technology, vol. 3, no. 2, pp. 474-477
- [3]. Govindaraju, K, Boopathi, S, Parvez Ahmed, F, Thulasi Ram, S, Jagadeeshraja, M 2012, 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]. 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
- [5]. 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
- [6]. Jian Feng &HongmeiJin 2011, μ C/OS-II Port for STM32F107VC Processor, Information Engineering Letters, vol. 1, no. 1, pp. 1-7
- [7]. Kaibin Wang, Ping Li, Jingke Liu, DianyanNing 2011, Application of μ c/os - II in the Design of Mine dc Electrical Prospecting Instrument', International Conference on Fine Exploration and Control of Water & Gas in Coal Mines, vol. 3, pp. 485-492
- [8]. Naga srikanth, C, Veda chary, M, Sudhakar, M 2012, Development of microkernel for multitasking with ARM11, International journal of Engineering Science and innovation technology, vol. 2, no. 2, pp. 12-17
- [9]. RaanaSyeda, Manju Ahuja, SnehaKhatwani, SwaraPampatwar 2014, Priority Based Assignment of Shared resources in RTOS, International Journal

- of Engineering Research and Applications, vol. 1, pp. 84-91
- [10]. Rinku, DR, Mohdarshad 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
- [11]. Srikanth, K, NarayanarajuSamunur 2013, RTOS Based Priority Dynamic Scheduling for Power Applications through DMA Peripherals, International Journal of Engineering Trends and Technology, vol. 4, no. 8, pp. 3660-3664