VOL. 12, NO. 9, MAY 2017 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

DYNAMIC SCHEDULING OF MESSAGE FLOW WITHIN A DISTRIBUTED EMBEDDED SYSTEM CONNECTED THROUGH A RS485 NETWORK

JKR Sastry, T. Naga Sai Tejasvi and J. Aparna
Department of ECM, KL University, Vaddeswaram, Guntur District, India
E-Mail: drsastry@kluniversity.in

ABSTRACT

RS485 based networking is quite often used for establishing a distributed embedded system. The communication among the embedded systems that are connected in a network is established through a serial communication protocol. The embedded systems are identified with station numbers at the time when the embedded system gets connected to the network. Most of the communication is initiated through the master. Applications that are implemented on a distributed embedded network needs that the messages flow in a chronological sequence which can undergo a change from time to time based on the environmental conditions prevailing when the system is in running state. The criticality conditions as such keep changing from time to time. The way the messages flows across the network changes dynamically. In this paper, a dynamic scheduling algorithm has been presented that effects the communication as per the environmental conditions prevailing at any point in time.

Keywords: RS485 distributed embedded systems, dynamic message scheduling, criticality based messaging.

1. INTRODUCTION

In most of the distributed embedded systems especially when RS485 kind of networking is established, master becomes the controller for controlling the messages to move in between the master and the slaves. Most of the distributed embedded applications are designed for flow of the messages as per static priority system, priority attached to each message with which the message is communicated initiated from the master targeted to be received by a salve. But in a real time system, events occur at random. While some events require immediate control actions, some other events do not require any control actions. However most of the distributed embedded systems are real time in nature and the events happen in random fashion. As such there is no fixed structure or sequence for the events to happen.

The priority with which the events must be processed depends on the criticality of that event. Less critical events could be differed for a while. More critical events must be processed first. The processing of the events must be scheduled. Scheduling can be done either in a static way or dynamic way. Static way of scheduling will not suit real time environment as the priority of scheduling the processing the events keep changing. Dynamic priority based scheduling can be carried in many ways. Dynamic scheduling can be carried based on optimality, feasibility, the need for worst case response time, use of shared resources and release of jitters.

An embedded system can be designed in such a way that an event is processed by following end-to-end policy which could be affected through many paths of processing. For optimality one has to choose a path that processes the event according to the priority set low or high. Some times in a real time system, differing an event to be processed or speeding up the processing of the event seems in-feasible as it is difficult to find a proper schedule that befits the sequence in which the messages are to be processed considering the occurrence of the event. Another important issue that must be addresses when dynamic scheduling has to be effected is the worst case response expected for processing an event. When the messages are dynamically scheduled based on the priority set considering the environmental conditions, sometimes may lead to loss of worst case response time, defeating the design of the embedded system itself. The designer of an embedded system must consider the worst case response time along with the scheduling of the event processing, so the dynamic nature of the embedded system could be met.
Dynamic scheduling of the message processing initiated by a master to be processed by a slave is much more complicated as the events happen randomly at distributed locations. The worst-case response times of each of the events differs greatly. Processing time of any of the message also depends on network latencies, traffic flow and congestion. In this paper, dynamic scheduling of messages to be processed by a master connected to the slaves using RS485 bus based serial communication network has been presented. The RS845 based network connects a single master and several slaves and the processing takes place in a distributed manner.

2. RELATED WORK

Jean-François HERMANT *et al*., [1] have deliberated on the real time scheduling algorithms considering fixed and dynamic priorities. It is necessary to deliberated on both aspects of scheduling considering ease of implementation, efficiency and complexity of the processing built into the embedded applications. Priority scheduling becomes necessary for effecting proper controlling mechanisms and real-time guarantees especially when real-time processing has to be supported on large scale distributed embedded systems.

Mohamed Gadalla Musa *et al.,* [2] have presented a communication method to facilitate communication between a slave and the master but as such have not considered the real-time aspects in scheduling the messages to be transferred from the Server to the client. Sastry *et al*., [3] have presented message flow system

based on the static priority scheduling not considering the real-time situation that leads to dynamic scheduling.

Nicholas VunC *et al*., [4] have used RS485 communication system for implementing a LAN based system for interconnecting various RS485 based rectifier systems. The network is established based on multi drop serial RS485 line and using PC as the master station for monitoring and controlling the rectifiers through a power network. In the olden days telecommunication power supply equipment are developed based on thirstier based controls. These systems are costly and less reactive. Quite recently switch mode power supplies have been invented which are highly efficient and compact. These power supply systems are provided with various aspects of intelligence to regulate the power by interfacing the same with microcontroller based telecommunication systems. The communication line length to a maximum 1200 mtrs can be supported when RS485 communication protocol system is implemented. The maximum length can be further extended by using boosters. Data corruptions [5] which include transmission line effects, impedance imbalance in balanced pair, grounding and inadequate shielding that occur when twisted pair cable are used for transmission have been identified. Two main reasons for occurring of transmission effects include improper termination and quality of the cable. The reasons for the corruptions have been described which are primarily due to poor quality cables and improper termination. The methods for appropriate termination have been explained that avoids the data corruptions.

RS485 and fieldbus communication protocols have been used [6] to pass the process parameters and status information of field devices to monitoring computer as a mixed design. The interface between the management of control system has been achieved by ODBC (Distributed Computer Control Systems) technology. A recent trend in DCCS (Distributed Computer Control Systems) is to interconnect different distributed elements by a multipoint broadcast network wherein past point-to-point links were used. Fieldbus networks are intended to interface at the lower level of automation hierarchy for the devices like process controllers, sensors, actuators. The lowest level of industrial network in the computer communication hierarchy for both manufacturing automation and process control systems is Fieldbus. Fieldbus provides digital, bi-directional, multi-drop, and serial-bus communications within isolated field devices like supervisory computers, controllers, robots, numerically controlled (NC) machines and programmable logic controllers, transducers, actuators, sensors, and operator stations. Merits of the fieldbus are EIS openness, interoperability; interchangeability, low cost, better performances, better maintainability, better modularity etc. The disadvantage of such a system is replacing the olden 4-2Om Analog signals. True distribution of control and control processing can be achieved through Fieldbus.

The process of data acquisition and processing has been explained [7] through using networked embedded controller systems. S3C2410A has been used as microprocessor and software development was undertaken using the platform embedded Linux, Oracle Berkeley DB, and C Language. The core board consists of CPU, memory, clock, reset, and power supply. Network communication, network detection and data retransmission, calculating the system performance have been coded into the ES application. Networking of Micro controller systems which are built using RS232C and a 485 converter on top of it, has been considered. No design aspects as such are spelt out that should be considered when networking Microcontrollers based system has to be undertaken using heterogeneous embedded systems.

Daogang Peng *et al*., [8] have also used RS485 for networking several Microcontroller based systems each meant to act as data acquisition system acquiring data related to various physical parameters that include temperature, pressure etc. Some of the microcontrollers connected on to the RS485 networks are also made to communicate with remote computing stations using TCP\IP communication. Traditionally, logic controllers have been replaced by the Microcontroller based systems making it possible to acquire different physical environmental data. Issues related to heterogeneity of Microcontrollers have not been addressed that can be part of an embedded networking solution. The authors proposed dual network connectivity considering both Ethernet and RS485 thus yielding high level of redundancy leading to very high reliability. This kind of networking can however be achieved when all the ES boards used for networking have the native support of both RS485 and the Ethernet.

The design of RS485 networks involves considering various intricate parameters that include topology, cabling network matching, over voltage, transient protection, earthling, failure protection, deadlocks etc. [9].The authors have reasoned the use of topologies and the conditions with which the topologies can be used. Many networking abrasions occur that include reflection, crosstalk, attenuation, noise in ground plane etc. that occur during data transmission as the distance increases. The signal transmission rate rises as the abrasions increases, thus limiting the communication distance. Designing the termination of RS485 communication system is one of the most important design parameter. The termination has to be done considering the impedance of the networking cable so as to avoid the reflection that may disturb the voltages used for transmitting the bits. Two methods are in use for termination using resistance and RC based impedance. Various aspects that must be considered for designing effective RS485 based networking systems have been presented. They have considered networking of Microcontroller based systems which are built using RS232C and a RS 485 converter on top it. No design aspects as such are spelt out that should be considered when networking Microcontrollers based systems which are heterogeneous in nature.

Testing of electronic devices and equipment in large scale are most sought out requirements in the Industry. RS485 based master salve networking [10] have been used for undertaking testing in large scale, PC is used

as HOST and several distributed Single chip microcontrollers (SCM) are used as slave processors. The SCM are used for testing the rectifiers and the code running on PC performs data display, counting, storing, printing and other test functions. In order to support long distance transmission outdoors and to increase anti-interference performance and to resist lightning storms, voltage level conversion between SCMs' TTL and RS485 is performed by a N75LBC184 chip which can be both anti-lightning and can bear 8KV static electricity shock. Applications showed that the newly designed system has the characteristics of reliability, obviously high working efficiency, and decreasing the rectifier bridge production cost

In a stereo garage, there are many devices which need to communicate with PLC, the main controller of stereo system. The devices include such as LED screen, ID card readers, keyboard, MP3 player and so on. The protocols of the devices are so different that PLC can't communicate with the devices directly. Therefore, a network is needed to combine devices and PLC together. In order to integrate stereo garage devices in a network and realize information sharing, a communication network based on RS485 is designed [11]. The network is simple but stable and useful. Both hardware design and software design are introduced. In hardware design, some primary schematics are proposed. In software design, communication protocols and data structures are discussed. At last, the service condition is mentioned as well.

Interconnecting a PC based network (High performance network) with Microcontroller based (Single Chip based) has become a necessity for implementing applications that need high performance computing and networked low performance computing. A system design of PC and more practical master-slave control system was introduced; using Multi- computer and PC complete the communication [12]. The hardware and software system design were given. The experiment was carried out, and the result shows that this system has advanced, practical and good reliability.

Technical training and experimentation requires an automation system, which requires a communication platform which supports multiple protocols [13]. The communication platform is built around ARM Cortex -M3 microcontroller and owns abundant peripheral circuit and Interfaces. The system is primarily supported through implementing a file transfer protocol. TCP/IP is protected on Cortex board for effecting File transfer. A PC is connected on to the network through RS485. The host resident application is implemented through C# using which the files can either be uploaded or downloaded.

Clean and inexhaustible power of sunlight will be the most promising resource in mankind's quest to develop sustainable energy in the 21st century and beyond". The solar energy has several advantages for instance, it is clean, unlimited, and has potential to provides sustainable electricity in the area not served by the conventional grid power, and for this reason solar energy is the most suitable renewable energy source that will be used in entire world. The main focus is to design and simulate a DC to DC converter [14] used for supplying power to a distributed sensing system based on a multi-drop sensor network with RS485 interface. This system is used to measure temperature, voltage and current in the automotive or navy application.

Data about the process taking place within a power plant is undertaken through an embedded system. The data is processed at a centralized location and the processes results are used to control various actuators that control the process itself. A design platform has been proposed based on 32- bit ARM Coretx -MO microprocessor as its core. The overall design scheme [15] of the system and the content of Modbus Communication Protocol are introduced while the circuit interface is achieved through RS485. The communication between master station and slave station based on Modbus RTU communication protocol, and embedded real-time operating system μCOS implements and discusses the generation of Cyclic Redundancy Check in the Modbus Communication Protocol.

3. PILOT PROJECT

A distributed embedded system is generally used for monitoring and controlling the temperatures within nuclear reactor systems. Temperature sensors are connected on to the rector tubes for measuring the temperatures which are transmitted to a centralised embedded system for checking with reference temperatures and initiate control action to trigger the pumps connected to the rector for pumping the coolants if the measured temperatures are greater than the reference temperatures. The state of temperatures in the rector tubes is dynamic. The temperatures either exceed or lower than the reference temperatures depending on the way the reactions take place in the reactor tubes.

Several embedded systems are used for measuring the temperatures and pumping the coolants and the functioning of these is controlled through a centralised embedded system to which buzzing and displaying equipment are connected. The connectivity of the embedded systems is achieved through RS485 based network. Converters are used to translate the signals released through RS232C interface situated in every embedded board, MAX 232 is used for effecting the conversion.

The control of flow of messages between the server and the slaves is undertaken by the master, which is designed through a centralised embedded system. Several microcontrollers of different makes which include 89C51, AT89S52, ATmega328, PIC18F4550 and LPC2148 have been used for establishing the network. The heterogeneity among the Microcontroller at signal level is achieved through MAX232 and at the data level by data marshalling on the central server side. The connectivity of the salves and the master is shown in Figure 1.

The distributed embedded system achieved through RS485 based networking has been designed to meet the requirements shown in Table-1. The message

flow across the nodes on the network must take place in such a way that the requirements could be met.

Table-1. Requirements specification for Pilot project.

Requirement number	Requirement description
1	Read Temp-1 and write to LCD
2	Effect RS485 based communication between the 89C51 (System-1) and the Central Microcontroller (System-5)
3	Read-Temp-1 and send to Central Micro Controller
4	Read Temp-1 and measure throughput. Temperature-1 must be sensed at least 10 times per milli second
5	Effect RS485 based communication between the PIC18F4550 (System-3) and the Central Microcontroller (System-5)
6	If Temp-1 > Reference Temp-1 then Pump-1 must be on
5	If Temp-1 < Reference Temp-1 then Pump-1 must be off
6	Compare Temp-1 > temp-2 and if true assert buzzer on
7	Read Temp-1 and make buzzer off if < Temp-2
8	If Temp-1 > temp-2 then Buzzer is on
9	Response time of Temp-1 must be 10µ Seconds
10	If Temp-1 > Reference Temp-1 then Pump-1 must be on
11	If Temp-1 > Reference Temp-1 then Pump-1 must be off
12	If Temp-1 > Reference Temp-1 then Buzzer is on
13	Response between the Reading the Temp-1 and stopping the Buzzer must 10µ Seconds
14	If Temp-1 > Reference Temp-1 then buzzer off
15	Read Temp-2 and write to LCD
16	Effect RS485 based communication between the AT89S52 (System-2) and the Central Micro Controller (System-5)
17	ReadTemp-2 and send to Central Microcontroller
18	Read Temp-2 and measure throughput
19	Effect RS485 based communication between the ATmega328 (System-4) and the Central Microcontroller (System-5)
20	Read Temp-2 and make pump-2 on if Temp-2 > Reference Temp-2
21	If Temp-2 > Reference Temp-2 Pump-2 on
22	Read Temp-2 and make pump-2 off if Temp-2 < Reference Temp-2
23	If Temp-2 < Reference Temp-2 Pump-2 off
24	Read Temp-2 and make buzzer on if > Temp-1
25	If Temp-2 > temp-1 Buzzer On
26	Read Temp-2 and make buzzer off if < Temp-1
27	If Temp-2 > Temp-1 Buzzer On
28	Response between the Reading the Temp-2 and starting the pump-1 must be 10µ Secs
29	If Temp-2 > Reference Temp-2 Pump-2 On
30	Response between the Reading the Temp-2 and stopping the pump-2 must be 10µ Secs
31	If Temp-2 > Reference Temp-2 Pump-2 Off
32	The response between the Reading the Temp-2 and starting the Buzzer must be 10µ Secs
33	If Temp-2 > Reference Temp-2 Buzzer on
34	The response between the Reading the Temp-1 and stopping the Buzzer must be 10µ Secs
35	If Temp-2 > Reference Temp-2 Buzzer off

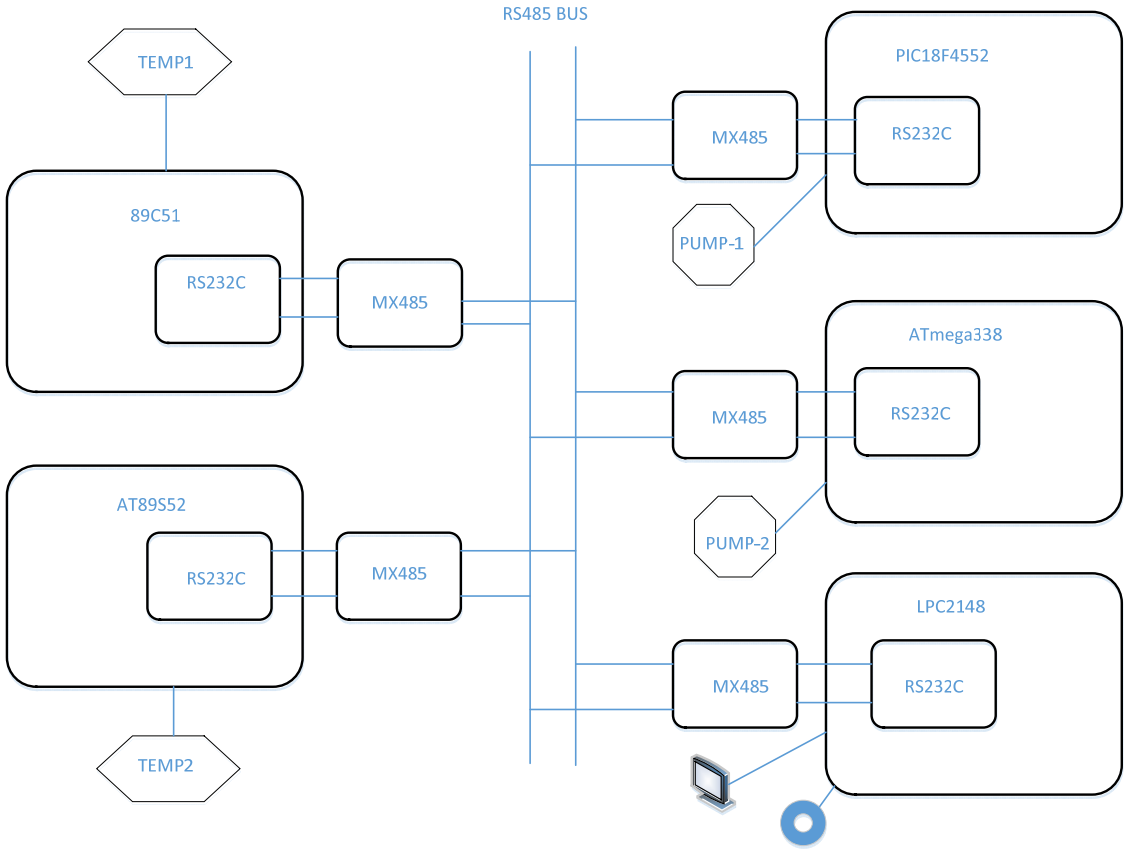

Figure-1. Distributed embedded system for monitoring and controlling temperatures within Nuclear Reactors.

The above mentioned requirements of the distributed embedded systems have been distributed to individual embedded systems based on the hardware supported on those systems.

4. MESSAGE SCHEDULING

The networking diagram shown in the Figure-1 shows the interfacing of the various heterogeneous microcontrollers based systems which are interconnected through a RS485 based protocol system using a HUB. However communication software resident in different microcontroller based systems is required for achieving application specific messaging through the middleware ported on to a Microcontroller based system. All the heterogeneous issues are handling within the middleware.

The communication has to be initiated by the master by using RTR (Remote transmission request) for want of Temperature-1 and Temperature-2 to be transmitted by 89c51 and AT89s52 in that sequence. The throughput, sequencing and timing of receipt of the temperatures are designed and developed into master device. The applications on 89C51 and AT89S52 will have software components to receive the master requests through the middleware and transmit the data to the master device through the middleware. No addresses as such are required for effecting the communication. The communication components implements RS232C serial communication system for transmitting and receiving the temperature data as RS485 signals.

The master device at the start-up receives the reference temperatures from PC which is connected to the master through RS232C serial communication system. The connection between the PC and the master device is point to point. The communication between the distributed embedded systems and the master device is achieved through the BUS to which the master and the slaves are connected.

The sensed temperatures are compared with the reference temperatures and in the event that the sensed temperatures are more than the reference temperature, a message is sent to the Microcontroller based systems that operate the pumps to be on or off. On the master side, two individual software components for each of the pump controller system are in place for transmission of the commands and reception of acknowledgement that the intended pump operation has been achieved successfully or otherwise. The communication in this case is achieved through use of RS485 interface. The software components that are designed for effecting the communication between the master and the pump control slave devices is achieved through implementation of RS485 protocol.

The master also is provided with a component that computes the temperature gradient and asserts a buzzer or otherwise if the temperature gradient is beyond

the prescribed limits. This function as such requires no communication as the entire functioning is implemented within the master device.

Static Scheduling

The software architecture that depicts the application specific communication is shown in the Figure-2.It can be seen from the Figure that the master controller system has been provided with necessary interfaces for effecting RS485 based communication with other microcontrollers based systems.

In RS485 based communication, one of the connected device will be the master and the others are slaves. The master and the slave are identified with a station number. The master provides the station number when the slave is connected to the BUS. The slave stores its own station number provided by the master.

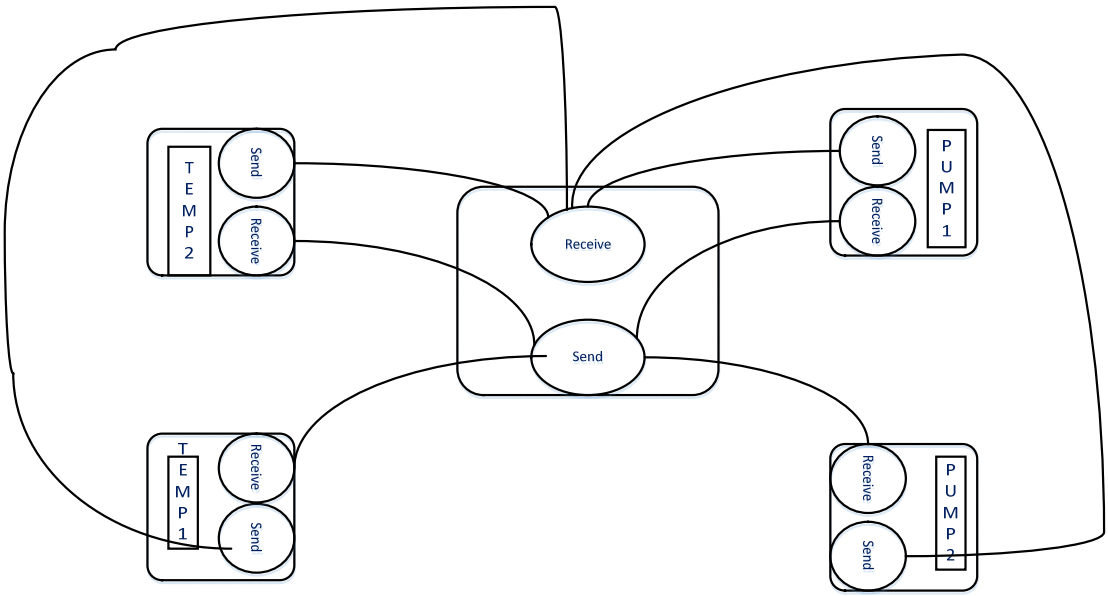

Figure-2. Software Architecture for effecting communication among distributed embedded systems using client server architecture.

Every device will transmit the data required by the master along with its station number when the slaves receive an RTS request from the master. The master will extract the slave number from the message that it has received.

Every communication is initiated from the master. Every slave is assigned with a station number which is built into it. Only one slave device will respond when requested by the master. The response from the salve could be an acknowledgment followed by the actual data requested by the master through a data packet which contains the details of the data the master is expecting.Priority numbers are assigned to the messages which must be transmitted to the slave that should receive the message concerned. The priority numbers are assigned based on the sequence of the messages that must flow from master to slave and vice versa.A typical station number allocation initiated from the masterto the salve is shown in the Table-2.

Master keeps transmitting the messages based on the priority set. There will not be any change in the sequence which is used for transmitting the messages. However the real-time environment keeps changing from time to time. The priority of the transmission of the messages keeps changing form time. While the transmission of a message to an embedded system should be higher priority when temperature read is greater than the reference temperature compared to the message to be sent to the same system for shutting of the pump when the temperature read is less than the reference temperature. The priority set to the messages should be altered dynamically at run time based on the environmental conditions that exist at the time of transmission of the message.

Table-2. Address allocation algorithm.

Serial number of the device	Type of device	Model number of the device	Station number	Priority number	Reasons for assigning priority
1	Master	LPC2148	1	999	Master has the priority over the salves Buzzer must be on If (Temp1- temp2) > 2
2	Slave-1	89C51	2	888	Temp-1 must be read first
3	Slave-2	AT89S52	3	777	Message to pump-1 must be sent if in case Temp1 > reference Temp1
4	Slave-3	PIC18F4550	4	666	Temp-2 must be read Next
5	Slave-4	ATmega328	5	555	Message to pump-2 must be sent next if Temp2 > Ref2
6	Slave-2	AT89S52	3	444	Message to pump-1 must be sent if in case Temp1 < reference Temp1
7	Slave-4	ATmega328	5	333	Message to pump-2 must be sent next if Temp2 < Ref2

Dynamic scheduling

Dynamic scheduling involves changing the priority of a message flow based on the criticality of the message. Association rules can be developed and implemented for dynamically setting the priority of messages to communicate across. Following algorithm can be used for setting the priority of the messages dynamically without effecting the real-time response requirements of the application which is implemented through effecting communication among the embedded systems which are connected through RS485 based communication system.

Algorithm

Step-1

Let the priority table is contained in a double dimension array having as many rows equivalent to the number of messages, that must be transmitted from central embedded system to distributed embedded systems.

Step-2

Each of the messages is provided with a static priority as per the design which considers normal flow of messages that must happen in an appropriate sequence. Table-3 shows the priorities that could be set as finalised at design time.

Table-3. Static priority setting to the messages.

Serial number of the message	Message	To embedded system	Priority value
1	Read temp-1	89c51	850
2	Assert pump-1	ATmega	800
3	De-assert pump-1	ATmega	750
4	Read temp-2	89c52	700
5	Assert pump-2	pic	650
6	De-assert pump-2	pic	600
7	Assert Buzzer	Lpc2148	550
8	De-assert Buzzer	Lpc2148	500

Step-3

The normal flow however might undergo change for various reasons based on environmental conditions and the flow must be controlled as per the prevailing situations some of which are quoted using the following fuzzy expressions:

a) If temp-1 < reftemp-1 and pump-1 status = “off” then the priority of the message “De-assert pump-1” is updated as 499

b) If temp-1 >reftemp-1 and pump-1 status is ON then the priority of the message “Assert pump-1” is 498

c) If temp-1 < reftemp-1 and pump-1 status = “ON” then the priority of the message “De-assert pump-1” is updated as 750

d) If temp-1 >reftemp-1 and pump-1 status is “off” then the priority of the message “Assert pump-1” is 800

e) If (temp-1) - (temp-2) > 2 and Buzzer status =”ON” then the priority of the message“Assert Buzzer” is 488

f) If (temp-1) - (temp-2) < 2 and Buzzer status =”off” then the priority of the message “De-assert Buzzer” is 477

g) If (temp-1) - (temp-2) > 2 and Buzzer status =”off” then the priority of the message “Assert Buzzer” is 550

h) If (temp-1) - (temp-2) < 2 and Buzzer status =”ON” then the priority of the “De-assert Buzzer” is 500

This algorithm dynamically changes priority of the messages and there by the messages will be executed as per the sequence required. The changes in the priority values are made dynamically at run time so that the message flow will be done as required based on the environmental status of the distributed embedded systems.

5. EXPERIMENTATION AND RESULTS

Experiments have been conducted using RS485 based network designed and the distributed embedded application system and the communication system implemented. Communication is effected by making the data flow as per the data packet standard designed for RS485 Standard. The results of the experimenting conducted are shown in the Table-4.

6. CONCLUSIONS

Static Prioritisation of the message to be transmitted in client server mode between the embedded systems that are connected on to RS485 based communication network does not respond to the real-time response time requirements. Dynamic prioritisation of the message flow is required for reducing worst-time response which can be achieved through developing and implementing association rules which can be used at run time for changing the priorities at run time.

Table-4. Experimental results.

Transaction number	Microcontroller System	Micro controller address	Bytes sent	Bytes received by micro controller system	Static priority response time in µsecs	Dynamic priority response time µsecs
1	89C51	0111100	2	LPC2148	8	5
2	AT89S52	0110010	2	LPC2148	6	4
3	LPC2148	1000110	1	PIC18F4550	3	1
4	LPC2148	1000110	1	ATmega328	2	1

REFERENCES

[1] Jean-Fran_coisHermant, Laurent Leboucher, Nicolas Rivierre, Real-Time Fixed and Dynamic Priority Driven Scheduling Algorithms: Theory and Experience, INRIA, Project REFLECS.

[2] Mohamed Gadalla Musa, Abubaker Maki Abdalla. Design and Implementation of RS485 Master-Slave Controllers Using C Language.International Journal of Engineering, Applied and Management Sciences Paradigms. 24(01).

[3] J. K. R. Sastry, A. Suresh and Smt J. SasiBhanu. 2015. Building heterogeneous distributed embedded systems through rs485 communication protocol. ARPN Journal of Engineering and Applied Sciences. 10(16): 6875- 6803.

[4] Nicholas V.C.H; Lau C.T; Lee B.S. 1993. A Power LAN for Telecommunication Power Supply Equipment. IEEE Conference Publications. 3: 24-27.

[5] Ajay Kumar V. 1995. Overcoming Data Corruption in RS485 Communication. IEEE Conference Publications. pp. 9-12.

[6] Yang Qingyu; Ren Shi. 2002. Design of a Mixed Distributed Computer Control System. IEEE Conference Publications. 4: 2782-2785.

[7] Ma Jing-rui; Dong Xue-Ren; Sun Yi-ping; Lu Qiu-xia. 2009. Data Acquisition and Processing of Embedded Network Controller. IEEE Conference Publications. 1: 285-287.

[8] Daogang Peng; Hao Zhang; Hao Zhang; Li Hui; Fei Xia. 2009. Research and Development of the Remote I/O Data Acquisition System Based on Embedded ARM Platform. IEEE Conference Publications. 341-344.

[9] Lei Shi; BaolongGuo. 2009. RS485/422 Solution in Embedded Access Control System. IEEE Conference Publications. pp. 1-4.

[10] Yanfang Wang; Wandui Mao; Jinying Li; Peng Zhang; Xiaoping Wang. 2010. A Distributed Rectifier Testing System Based on RS-485. IEEE Conference Publications. pp. 779-781.

[11] Jie Shen; Ye Yunyue; Zhuo Zheng; Weiming Hong. 2011. Communication network design of stereo garage devices based on RS485. IEEE Conference Publications. pp. 3831-3834.

[12] Shihong Qin; Kai Chen. 2011. Design of the Multi-computer Communication System Based on RS485. IEEE Conference Publications. 1318-1320.

[13] Yunzhou Zhang; Chengdong Wu. 2011. Design of A Training and Experimental Platform Based On Multi-Protocol Communication. IEEE Conference Publications. pp. 01-04.

[14] Ioana-Monica P.; Florin P.; Viorel P. 2012. Design and Simulation of DC/DC Boost Converter used for a Distributed Sensing System Based on a Multi-drop Sensor Network with RS485 Interface. IEEE Conference Publications. pp. 79-82.

[15] Kelong Wang; Daogang Peng; Lei Song; Hao Zhang. 2014. Implementation of Modbus Communication Protocol Based on ARM Cortex-MO. IEEE Conference Publications. pp. 69-73.