ISSN 2229-5518

Design and Development of New Architecture for Reconfiguration and Processing of Automation Industrial Control System

Suvidh P. Gohane, Ganesh S. Khekare

Abstract— In most Automation Industrial Process System, the processes such as manufacturing, processing, packaging etc. are performed with automation conveyor mechanism system with some particular processing pattern, If in real time there is need to use this same system to execute another process with some different modified processing pattern. There is a need of reconfiguration of the system to execute modified processing pattern. Traditionally to achieve this, new system has to be developed with new configuration, this will require programming expertise to meet modified processing pattern and are costly. So above problem taken into consideration, the design of a real time reconfigurable industrial system has been developed. The term 'reconfigurable' here indicates that the system consisting of various industrial machines and nodes can be configured to change its mode and sequence of operation, thus changing the work process. Such systems may be used in industries where a series of manufacturing operations and processes are implemented to obtain a finished product. Reconfiguration may be needed here to change the design or manufacturing process, to obtain or fulfill the requirements of another final product. To demonstrate the concept, a system resembling an industrial assembly line have been created, consisting of a conveyor belt mechanism and sensors with various other mechanisms operating alongside the conveyor over goods and objects being carried by the conveyor. To enable reconfiguration, new Motion Description Language Architecture (MDLA) schemes that allow the user to describe the set of processes and tasks in the form of a script that the machine understands have been proposed and designed, also developed Application Programming Interface (API) Graphical User Interface (GUI) based software where the user can write the scripts, which are then translated into MDL codes and sent to the machine which starts functioning accordingly.

Index Terms— Application programming interface (API), Automation, Industrial embedded System, Microcontroller, Motion Description Language Architecture (MDLA), Reconfigure, Sensors.

1 Introduction

In most of the automation industrial control system, such as a refinery, thermal power plant, automation plant paper production plant. There are present hundreds or thousands of sensors and actuators along with embedded control system which automatic monitor and control functionalities of more advanced and complicated hardware. Due to some environmental changes or any other technical changes, malfunction in devices or in sensors may occur. Traditionally, these devices must need to design again or replaced. This in turn causes, increase in the cost and very time consuming. So the systems must provide easy and convenient system reconfiguration with specially designed software for its hardware.

Usually Software for embedded control systems is designed and implemented with a set of functions, such as device drivers, control functions and algorithms. Sometimes Components threshold/reference value may need to be added, adjusted, removed, or modify in real time in such manner to meet new product requirements in industry. This emerging trend calls for reconfigurable embedded system, in which software that reuses or modifies existing hardware components to generate the reconfiguration software for each new application very quickly. This will result in turn to allow a way out for low-cost product development and Maintenance.

So proposed developed work aims to design a new Motion Description Language Architecture (MDLA) schemes for reconfiguration of WSN nodes, in most easy and without having prior knowledge of programming, in which Application programming interface (API) will have to designed wherein each nodes contains various code to be assigned to node parameter and reconfiguration can be achieved using simple reconfiguring commands.

This paper is organized as follows. Section II discusses the related work described concerning about methods/ techniques to configure and program wireless network embedded devices and to connect those to external applications to internet. In Section III, the proposed system architecture is described. It explains the architecture of wireless node components present and what are the parameters to achieve remote easy reconfiguration flexibility of any industrial wireless embedded systems. Section IV describes methodology for reconfiguration of automation conveyor process system. In Section V design and implementation of the system, hardware implementation and software implementation have been explained. What are the outcomes explained in Section VI. Finally, concludes the paper in Section VII.

2 Related Work

The approach proposed in this paper provides reconfiguration capabilities for embedded devices such as WSN nodes. Using it, distributed sensor and actuator nodes can be managed without any custom programming, with only simple configuration commands. Related work includes strategies/techniques to configure and program sensor network devices and to connect those to external applications via internet.

In the literature, there present several works that address reconfiguration. In paper [1], Author aims at designing a well-defined MidSN, standard components and formats exist that are followed by any node and in order to deploy a system architecture for deploying and configuring the servers and embedded devices with operations at the beginning of deployment, providing configuration flexibility prior to operation through remote configuration. The proposed MidSN architecture builds an intermediate computing layer which will serve as an abstraction hiding the different hardware implementations from embedded devices networked applications.

In [13], [14], HW reconfiguration is design by *ad hoc* reconfigurable devices. a new approach based on the tight coupling of processor with a dynamically reconfigurable function unit which is optimized for wireless sensor network Devices. Dynamic reconfiguration is part of the regular operation mode and the key concept to achieve a small approach that provides sufficient performance, high adaptivity and good energy-efficiency. But it is prepared to be adapted for only prerecorded applications.

Paper [2] designs and realizes a reconfigurable smart sensor interface for industrial WSN in IoT environment. This design presents many advantages, First CPLD is used as the core controller to release the restriction on the universal data acquisition interface, and realize truly parallel acquisition of sensor data. It has improved the sensor data collection efficiency of industrial WSN, Secondly, a new design method is proposed multisensor data acquisition interface that can realize plug and play for various kinds of sensors in IoT environment. The design system applies the IEEE1451 interface protocol standard that is used for smart sensors of automatically discovering network.

Paper [3] provides a detailed description of the implementation two new rateless-based OAP protocols. Shed light on the various trade-offs that arise in implementation of rateless OAP on a sensor networks, such as the tradeoff between the size of program pages and the size of the underlying finite field used for computation. It provide extensive numerical results evaluating the performance of protocols, based both on real network experiments with Tmote Sky sensors and also on simulations.

Approaches related with Macro-programming and middleware are proposed in [4] and [5]. These approaches use a middleware to reprogram the network. Most consist of mobile agents which run over virtual machines. Typically, the agent code is developed by specific frameworks. Specific communication protocols are also designed to upload the code.

Work described in [6] is the first mobile agent middleware works for reconfiguration of a WSN implemented entirely in TinyOS. This paper present an in-depth case study of Agilla using a fire tracking application. In this application, mobile agents are deployed to dynamically form and maintain a perimeter around a fire as it spreads through a network comprised of 26 MICA2 motes. This paper makes three primary contributions. First, it demonstrates how a mobile agent middleware can be used to facilitate the development and deployment of a nontrivial application. Agilla able to rapidly create and deploy the entire fire tracking application by injecting 47-byte fire agents and a 100 byte tracker agent. Second a set of application-level performance results that demonstrate the reliability and efficiency of mobile agents and tuple spaces in a highly dynamic application. Finally, it provide new insights into, and lessons about, mobile agent programming techniques for WSNs.

The works described in above papers have some issues to handle the distributed configuration. There is need for software to burn completely static and node-specific code into each device for reconfiguration. Most of the reconfiguration architecture are mostly targeted at hardware changes and fails to offer the application configuration flexibility and are costly. Also they are only for Specific operating system. There is no such a software architecture which offers computation solutions to provide flexibility to adapt to industrial process changes efficiently. Each time user needs to develop the new code, for node reconfiguration which requires programming expertise.

3 PROPOSED SYSTEM ARCHITECTURE

Proposed system set up shown in figure 3.It consists of control base station at server side and Industrial set up consists of wireless network of three nodes i.e. Sensor Alarm Monitor Unit, Actuator unit and Reconfigurable Exhaust System.

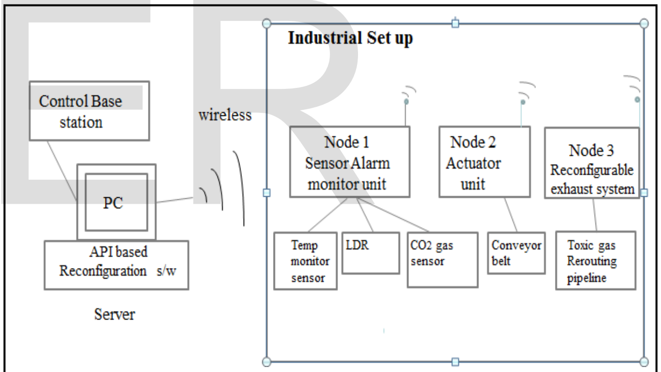

Fig. 1. System architecture

3.1 Node 1 Sensor Alarm monitor unit

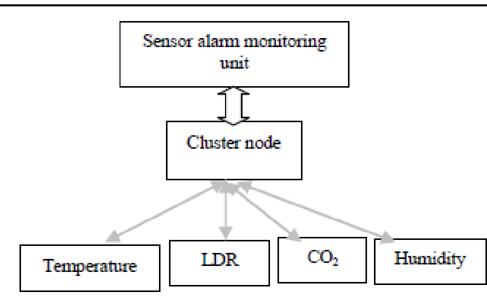

Fig. 2. Deployment of Sensors

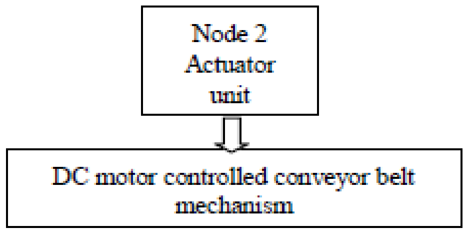

Fig. 3. Actuator node

In Industry such as thermal power plant, mining, oil refinery different sensors are used to sense the data and raise the alarm according to set threshold value. Sensor Alarm monitor unit shown in figure 4, is a collection of Controller Circuit, LCD, Battery, Zigbee wireless module for Communication Medium through cluster node, where all the sensors are connected to controller circuit. It is built of using AVR16 Microcontroller. The Parameters which reconfigure at Sensor Alarm monitor unit will be:-

1. Its Threshold value
2. Reboot the system at real time

3.2 Node 2 Actuator unit

In most of the Industrial control system such as Automation Actuator unit, AVR16 Microcontroller based DC motor controlled conveyor belt mechanism shown in figure 5, is used for manufacturing purposes. The parameters which will be reconfigured at actuator unit i.e. DC motor controlled conveyor belt mechanism will be:-

1. Speed for particular distance
2. No. of halts
3. Time duration of halts
4. Delay
5. Direction

At run time to make it reconfigurable.

3.3 Node 3 Reconfigurable exhaust pipeline system set up

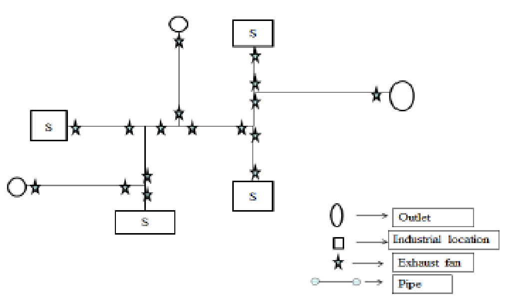

Fig. 4. Reconfigurable exhaust pipeline system set up

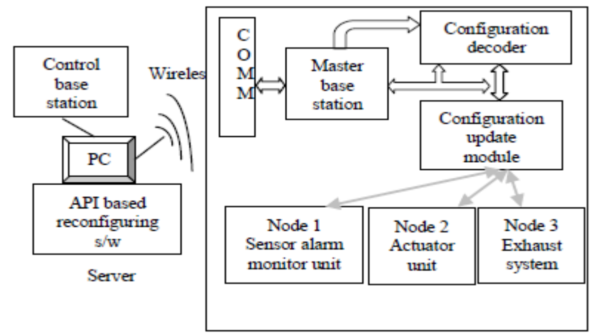

Fig. 5. Reconfiguration of Industrial control system

Figure 6 is reconfigurable exhaust pipeline system set up in this node, a network of exhaust pipes and ducts will be developed that will connect all chambers and rooms within the industry. Using multiple exhaust fans fit across every intersection, with directional control at server side API based reconfiguring software, will be able to reconfigure the route of the exhaust and suction system. In this way, the user would be able to choose the exact path from the point of suction to the point of throw-out.

3.4 Reconfiguration of Industrial control system

Proposed work shown in figure 7 will develop, the API based reconfiguration software at sever side and loaded in sever PC. This Reconfiguration software will be developed on Dotnet programming language. API will be created using Visual basic Studio (VB6). This will consist to use controls like – Button, List view and Labels Major controls were – Serial Port and Button. The advantage of using this language, it is event driven programming language and provides interface for application programming development. Control base station is nothing but a board having TX/RX Capabilities. AVR16 Microcontroller is connected to circuit using CP2101(USART) Buzzer and various input pins will also be provided in the Base station for communication and testing. Along with one LCD 2x16 will be interfaced. It continuously monitors the industrial wireless nodes functions. Zigbee wireless protocol used as COMM module for communication medium. Master base station acts as a middleware between server and industrial set up for configuration decoding and updation. Reconfiguration is done by simple API based reconfiguring commands for example R_config temp();

4 Methodology for Reconfiguration of Automation Conveyor Process System

The following Reconfiguring of Automation Conveyor Process System model to be constructed aims to provide user friendly access.

- To Develop the API Reconfiguring software on Server side.
- To create the Base station which acts as a middleware between server and nodes.

ISSN 2229-5518

- To create the embedded process nodes.
- To design these nodes as Reconfigurable.
- To provide the wireless communication between server and Reconfigurable nodes.
- To develop the configuration command and data commands.
- To test the entire system.
- To analyze the results.

The Algorithm and flow is shown below which shows how the whole system works for configuring node.

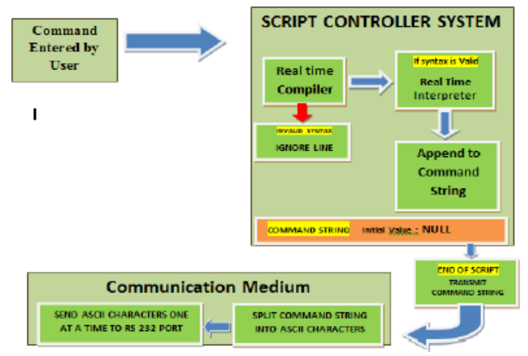

Fig. 6. Flow Diagram of MDL Architecture

ALGORITHM FOR ALARM UNIT

STEP 1: START
STEP 2: RECEIVE SENSED VALUE
STEP 3: IF RECEIVE SENSED VALUE > THRESHOLD LIMIT, YES GIVE ALARM OTHERWISE GOTO CONTROL STATE
STEP 3: CHECK CONFIG COMMAND
IF YES SET NEW THRESHOLD t VALUE OTHERWISE REPEAT
STEP 4: END

The control state diagram is explained in figure 7. It shows how the control flows from one state to another via feedback loop.

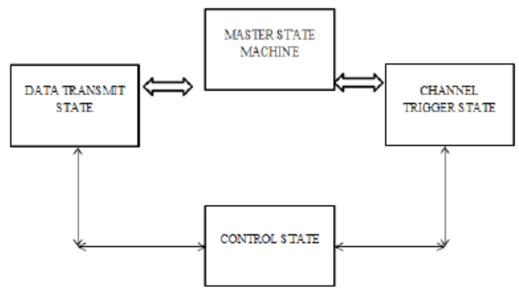

Fig. 7. Control state Diagram

Master state machine generates the reconfiguring command transfers to channel trigger state and data transmit state. Channel state triggers the pulses according to the command received by channel state. These pulses are in feedback loops with control state to data transmit state. Control state controls the pulses according to MDL codes send by users.

Algorithm For Reconfiguration of Run Time Process

1. Start
2. Check Continuously to receive command by user
   If commands received go to step 3 otherwise repeat Step 2
3. Decode the command in MDL codes & transfer it to character variable
4. Checking the entered MDL code command to perform reconfiguration process
   i. if((mybyte[q]>=65)&&(mybyte[q]<76)) movefw(mybyte[q]-65); Move conveyor belt in forward direction for specified time, if not matched go to ii
   ii. else if((mybyte[q]>=110)&&(mybyte[q]<120)) movebw(mybyte[q]-110); Move conveyor belt backward direction for specified time, if not matched go to iii
   iii. else if((mybyte[q]>=48)&&(mybyte[q]<60)); perform process 1, if not matched go to iv
   iv. else if((mybyte[q]>=77)&&(mybyte[q]<87)); perform process 2, if not matched go to v
   v. else if((mybyte[q]>=88)&&(mybyte[q]<98)); perform process 3, if not matched exit
5. Stop.

5 Design and Implementation

In this section design of hardware implementation and software implementation have been explained with specification and snapshots are shown.

5.1 Hardware Implementation

The model is divided into two parts i.e. hardware part and software part Firstly, Hardware part is also divided into two industrial test bed, sensor alarm monitoring system unit and Conveyor Process System unit.

5.1.1 Implementation of Sensor Alarm Monitoring Unit

In sensor alarm monitoring system unit a circuit diagram is designed which consist of Microcontroller (ATMEGA 16), Temperature sensor (LM35), Gas sensor (MQ-6) and Serial Communication (RS232).Two sensors are interfaced using Relay to microcontroller ATMEGA16. Port D is used in which pin no. 2 & 3 is used for connecting serial communication i.e. RS232 for transmission and reception of data. LCD is interfaced with port A of Atmega 16 on which real time corresponding reading is shown, Alarming buzzer is connected to port B.

ISSN 2229-5518

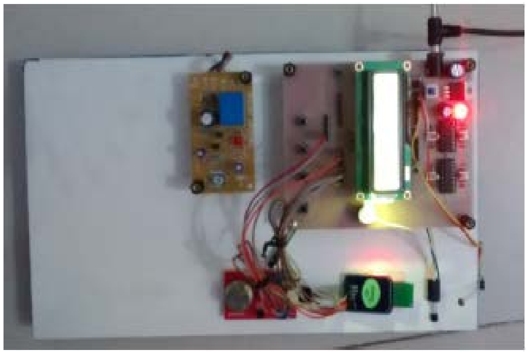

Fig. 8. Snapshot of Sensor alarm Monitoring Unit

- Sensor Node consist of two Sensors interfaced:
    1. Temperature
    2. Gas Sensor
- Temperature Gives Reading from 100-800
- Gas Sensor Gives Digital Output

5.1.2 Implementation of Actuator Unit

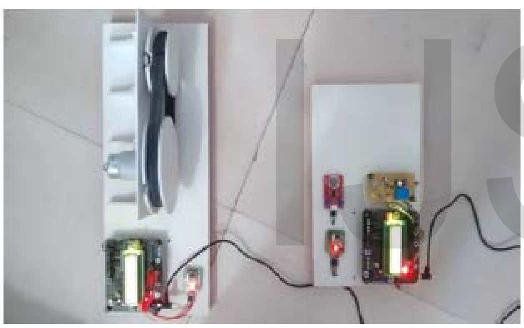

Fig. 9. Snapshot of Actuator Unit

- Node consist of Conveyor Belt Mechanism driven using 60 rpm motor
- In order to configure conveyor belt speed, direction, delay, PWM has been generated on PORT D of Atmega 16 through driver IC L293D.
- RF has been interfaced at (Tx/Rx) Pin of Controller working with 2.4 Ghz

5.1.3 Implementation of exhaust pipeline system set up

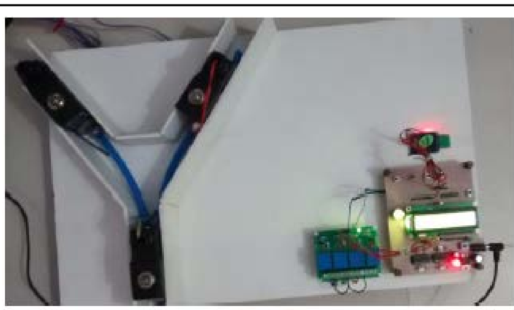

Fig. 10. Snapshot of exhaust pipeline system set up

- Gas pipeline re-routing has been shown using three solenoid Valves connected in Y arrangement.
- In and Out of Solenoid Valve has been controlled using ON/OFF, corresponding with Relay and Commands received from Base Station.

6.1 Software Implementation

6.1.1 Creation of API based GUI

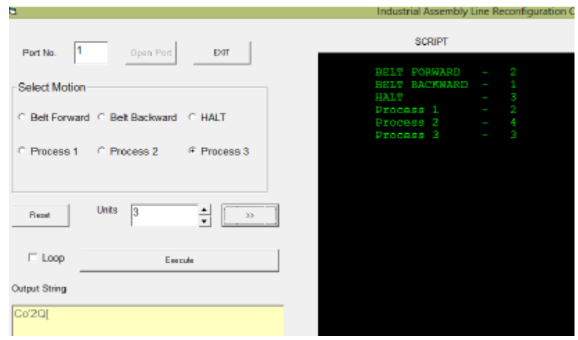

Fig. 11. Snapshot of API based GUI

- API has been created using Visual Basic Studio (VB6)
- API consist use of controls like – Button, List View and Labels
- Major controls were – Serial Port and Buttons

6 Experimental Results and Discussions

This section shows the experimental test bed for reconfiguration of Industrial assembly line System and discusses how the proposed developed system reconfigure the process pattern on real time and how effectively it achieves the reconfiguration of system through MDLA architecture.

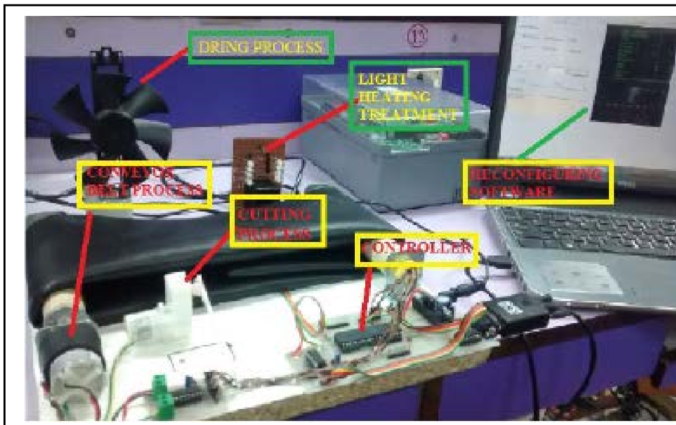

Fig. 12. Snapshot of Achieving Reconfiguration of Industrial Proessing System

Here, the paper production industrial test beds having six automation processes are defined to process with any real time pattern. Process one is a cutting process used in paper

ISSN 2229-5518

production industries, process second is drying process, process third is light heating treatment process and conveyor belt movement forward, backward and delay process. Figure 12 shows how reconfiguring software which is loaded in PC achieves the reconfiguration of system model.

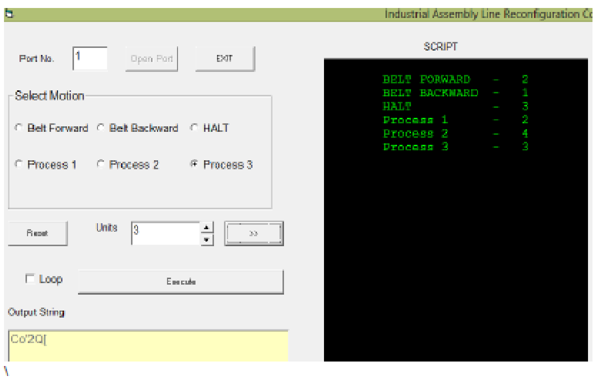

Fig. 13. Snapshot of GUI for Achieving Reconfiguration of Industrial Processing System

The GUI for reconfiguration of industrial test bed has shown in figure 13. Here anyone can easily reconfigure the processing system with this GUI without having much more knowledge of programming. Table 1 shows generation of MDL code commands run time reconfiguration of process pattern. For example if there is need to perform action of delay in between already running process pattern, simply open the reconfiguring software GUI, click on the halt caption button also can able to select how much delay want produce, selecting scroll bar of units. In this way following codes transfers to the processing model system to perform reconfiguration depending upon the command transfer.

TABLE 1
GENERATION OF MDL CODES

Actions	No. Of Units / Base Code	1	2	3	4	5
Forward	A	B	C	D	E	F
Backward	M	N	O	P	Q	R
3.5 Second Delay	$	!	@	%	^	&
Process 1	1	2	3	4	5	6
Process 2	m	n	o	p	q	r
Process 3	a	b	c	d	e	f
Loop	?					
End of Command	#					

Upon implementation, this proposed system will enable factory engineers, supervisors and even users, who do not have that much knowledge of programming to be remotely monitor / reconfigure the operation of various machines, and verify its accurate implementation, wirelessly via a remote server.

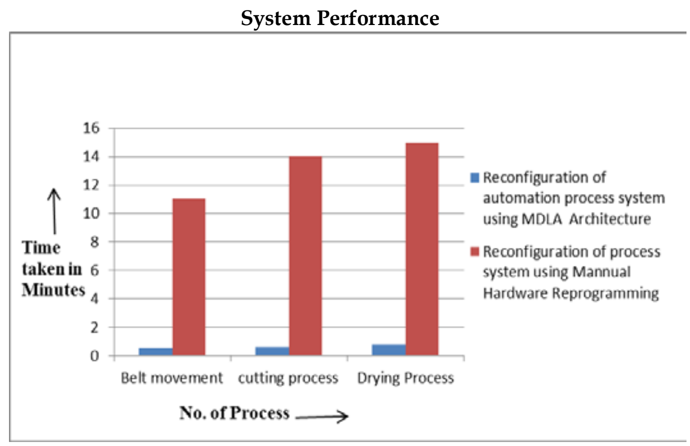

Fig. 13 Graphical Analysis

The Graphical analyses are shown with the help of Figure 13. The system has then check the performance with the existing Manual Hardware Reprogramming Architecture system, considering the parameters of command passed and time taken to respond for corresponding process. Both the above parameters are tested in the proposed system and it is found that event is triggered at same time and proposed systems, response time is very low i.e. in seconds as compared to Manual Hardware Reprogramming Architecture system.

7 CONCLUSION AND FUTURE WORK

Experimental set up achieves the effective reconfiguration through proposed MDLA architecture schemes. The main aim behind the proposed developed system approach is to provide such architecture which uses Application programming interface (API) for easy and uniform configuration and processing over wireless networks comprising embedded devices and nodes such as sensor actuator units or control stations. Proposed system has described the detail explanations about each wireless industrial node components. In wireless sensor networks, a combination of high energy-efficiency, flexibility, interoperability, low cost and user friendly interface is required, which is difficult to achieve with classical architectures. So uniform API based MDLA reconfiguration software architecture is developed, has the potential to offer a much better interface than to adapt for hardware changes solutions, which are found in most of related work.

The work presented here encourages researchers and designers to develop such protocols and algorithms that are based on fast and efficient computation to make future sensor nodes smarter and more efficient. The approach for a flexible, low cost and efficient sensor node presented in this paper is based on the combination of a controller with uniform reconfigurable functional unit, for industrial applications, can also have to consider web support applications could be imple-

ISSN 2229-5518

mented using Internet of things (IoT) concepts for reconfiguration functional unit operations. For future research it might be very useful to analyze the development of tools for heterogeneous mixed wireless sensor network and wired network that could automize and prioritizes these tasks.

REFERENCES

[1] José Cecilio and Pedro Furtado, "Architecture for Uniform (Re)Configuration and Processing Over Embedded Sensor and Actuator Networks," IEEE transactions on industrial informatics, vol. 10, no. 1, february 2014

[2] Qingping Chi, Hairong Yan, Chuan Zhang, Zhibo Pang, and Li Da Xu, Senior Member, IEEE "A Reconfigurable Smart Sensor Interface for Industrial WSN in IoT Environment," IEEE transactions on industrial informatics, vol. 10, no. 2, may 2014

[3] A. Hagedorn, D. Starobinski, and A. Trachtenberg, "Rateless deluge: Over-the-air programming of wireless sensor networks using random linear codes," in Proc. IEEE Int. Conf. Inf. Process. Sensor Networks, Apr. 2008, pp. 477–466.

[4] A. Rowe, M. Berges, G. Bhatia, E. Goldman, R. Rajkumar, L. Soibelman, J. Garrett, and J. Moura, "Sensor Andrew: Large-scale campus-wide sensing and actuation," IBM J. Res. Develop., vol. 1, pp. 1–6, 2010.

[5] A. Tavakoli, A. Kansal, and S.Nath, "On-line sensing task optimization for shared sensors," in Proc. 9th ACM/IEEE Int. Conf. Inf. Process. In Sensor Networks, Stockholm, Sweden, 2010, pp. 47 57.

[6] C.-L. Fok, G.-C. Roman, and C. Lu, "Agilla: A mobile agent middleware for self-adaptive wireless sensor networks," ACM Trans. Auton. Adaptive Syst., vol. 4, no. 3, pp. 1–26, 2009, Art. 16.

[7] H. Hinkelmann, P. Zipf, and M. Glesner, "Design concepts for a dynamically reconfigurable wireless sensor node," in Proc. 1st NASA/ESA Conf. Adaptive Hardware Syst., Jun. 2006, pp. 436–441.

[8] K. Aberer, M. Hauswirth, and A. Salehi, "The global sensor networks middleware for efficient and flexible deployment and interconnection of sensor networks," EPFL, Tech. Rep. LSIR REPORT-2006-006, 2006.

[9] P. B. Gibbons, B. Karp, Y. Ke, S. Nath, and S. Seshan, "IrisNet: An architecture for a world-wide sensorweb," IEEE Pervasive Computing, vol. 2, no. 4, pp. 22–33, Oct. 2003

[10] P. Costa, L. Mottola, A. L. Murphy, and G. P. Picco, "Programming wireless sensor networks with the TeenyLime middleware," in Proc. ACM/IFIP/USENIX Int. Conf. Middleware, 2007, pp. 429–449

[11] E. Monmasson and M. N. Cirstea, "FPGA design methodology for industrial control systems—A review," IEEE Trans. Ind. Electron., vol. 54, no. 4, pp. 1824–1842, Aug. 2007.

[12] S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, "TinyDB: An acquisitional query processing system for sensor networks," ACM Trans. Database Syst., vol. 30, pp. 122–173, 2005.

[13] S. Brown and C. J. Sreenan, Updating Software in Wireless Sensor Networks: A Survey Tech. Rep. UCC-CS-2006-13-07, 2006

[14] C.-C. Han, R. Kumar, R. Shea, and M. Srivastava, "Sensor network software update management: A survey," Int. J. Network Manag., vol. 15, pp. 283–294, 2005.M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.