Implementation of Real Time Systems on FreeRTOS Platform for Industrial Automation Applications

Vicent Rutagangibwa[1] Babu Krishnamurthy[2]
[1]P.G Student (M.E) [2]Visiting Faculty
[1,2]Department of Electronics & Communication Engineering
[1]Mbarara University of Science & Technology (MUST) – Uganda & GTU PG School, Ahmedabad, India [2]CDAC, Pune

Abstract— Industrial Automation is one of the leading research areas in Embedded Systems. With the advent of new processors and methods of processing, communications and infrastructure, modern industrial automation systems require high real-time control capabilities. Real-time systems are designed to provide a timely response to real world events. FreeRTOS platform is chosen as the implementation platform because it is designed to be simple, portable, and concise and it can as well be used to develop multitasking applications suitable for real time industrial applications.
Keywords: real-time, Industrial Automation, infrastructure, multitasking applications

I. INTRODUCTION

Applications particularly for industries can be automated to guarantee response within strict time deadlines. FreeRTOS can be implemented for real time applications with low interrupt latency, simple to use, portable across many platforms and a deterministic scheduling mechanism [5]. The major motivating factor towards this research work was to implementing a working prototype on FreeRTOS platform that can serve as a benchmark for Industrial automation application. The current challenging problem in industries is how to formally define the automation program and the design of a real time scheduler algorithm to handle conflicts in selecting the actual output in a multitasking environment. The PLC implementation model has been criticized for Industrial Automations in accordance with [1], [3].

II. RELATED WORK

A. Real-Time System Behavior:

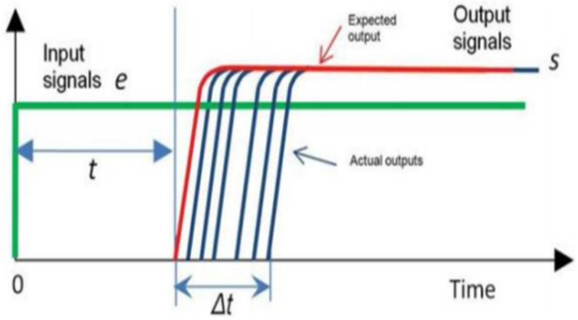

Fig. 1: Real-Time System Behavior [3]

To determine whether a computer system is really a real-time system, we repeatedly (but not periodically) apply the same input signal to the system, expecting to get the same output signal every time, at exactly *t* seconds after the input signal is applied. Unfortunately, this is not what happens in actual controllers. The same output signal will be generated by the controller every time, but the time *t* it takes the controller to produce each output may increase slightly as shown in *Fig. 1*. Repeating the same experiment, results in the controller response times falling into a variation interval, this interval helps in deducing the behavior of the system.

B. RTOS Implementation:

In recent years, the number of embedded devices in the marketplace has increased substantially due to significant reductions in size and costs of microprocessors. As a result, the safety of the real-time operating systems (RTOSs) that are traditionally used by embedded devices is becoming increasingly important. The industry has already recognized the importance of providing safe and reliable RTOSs and the academic community is actively working on tools and methods that can improve the current standards of software quality [4].

In the last decade, there have been significant studies to design platforms for intelligent industrial automation systems [3]. These automation systems rely heavily on a distributed computer-based infrastructure, where smart sensors and actuators, intelligent machines, robots, and other automation devices can interact using industrial protocols and take decisions in real-time. In these systems, system level communication, device synchronization, and integration of new devices to the system are extremely challenging issues.

Depending on the physical plant and the type of control to be performed, the controller may be classified as "hard real time" or "soft real time." If the specific characteristics of the plant or process to be controlled are such that noncompliance of the constraint will produce a malfunction or a failure, then the controller must be a "hard real-time" controller. If, on the other hand, the specific characteristics to be controlled are such that noncompliance of its constraint will generate a degradation of the plant functionality, but will not produce any malfunctions or failures, then a "soft real-time" controller may be used [4].

C. Industrial Automation:

The new paradigm shift towards industrial automation indicates the limitations of relying on OOP in programming large distributed systems [6]. In industrial automation, objects are often related to physical devices built in a certain specific industrial domain context, which calls for intelligent designs and implementation of systems that involve careful I/O handling and hardware specific integration of different modules to support the real time demands of the industrial application.

In reference to [6], the following methods can be employed in ensuring meeting real time constraints of an industrial automation application; Event-Triggered execution of Functional Blocks (FB), Reusability and adaptability implementation of the FB objects the basic functionalities of such physical devices to perform a certain working cycle, use of supervisory control in the context of industrial control, to solve the problem of programming the concurrent FB behaviors so as to satisfy the desired sequencing and logical constraints.

PLCs are often considered as the main workhorses of industrial automation systems. They are connected with the plant peripherals, such as sensors and actuators, via remote I/O modules which communicate with the PLC via field area networks (fieldbuses) [6]. With the implementation of real-time systems on FreeRTOS platform, the challenges of PLC's implementation especially its centralized nature can be minimized by having some of the controller's functionalities which are more critical be performed by the FreeRTOS system, and therefore have the PLCs perform more complex and power hungry related tasks. The good news is that this implementation provides connections to the main system that can then synchronize the outputs as inputs to the HMI (Human Machine Interface).

In one of the proposed neural network model [4], instead of having controllers loaded with specific algorithm, a general neural network based controller is assigned and trained to do the action of any type of controller. Communication with the external device via RS-232 interface (serial port) and UART can then be used for transmission of characters controlled by a special integrated circuit designed for both Tx and Rx ends.

D. Free RTOS Platform:

FreeRTOS system's real time architectural implementation and the stack structural model were analyzed in [7] and the execution strategy demonstrated in order to solve the problem of stack overflow and interrupt processing issues. The reference [7] discusses the advantages of using FreeRTOS system to support time slice rotation and priority scheduling, but also the open source can easily be migrated to different processors such as ARM, AVR [8] The paper demonstrated how to overcome the challenges with the current systems i.e. task switching and discussed how to solve it using resource scheduling mechanism, and memory allocation techniques that make use of FreeRTOS platform as the main advantages of using such architecture.

Different methods for implementation of real-time systems for industrial automation applications are discussed and fully analyzed in our survey paper [10]. The methods discussed include; Event-trigged, Service-oriented, Formal methods and the Object-oriented approaches.

III. CORE TECHNOLOGIES

The LPC1769 LPCXpresso board with NXP's ARM Cortex-M3 microcontroller Fig.2 is part of NXP's low-cost development toolchain for LPC families. It has been jointly developed by Embedded Artists, Code Red, and NXP [2]. It is an end-to-end solution for creating applications all the way from evaluation through to production. The microntroller was used together with LPC1769 embedded base board that consists of a number of peripherals for prototyping real time applications.

Fig. 2: LPC1769 development board [2]

IV. METHODS OF IMPLEMENTATION

The block diagram shown in Fig. 3 illustrates the building blocks of the system implementation;

- LPC1769 development board ported with FreeRTOS
- Sensor module that consists of light, temperature, and the accelerometer sensors for representing industrial environment sensing.
- Industrial Ethernet for the Networking module that help monitor the in real time context via a network.
- The control panel referred to as the HMI that consists of the switch buttons and the OLED display

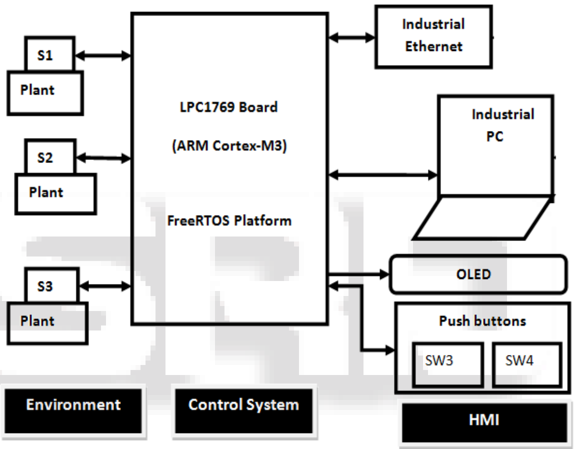

Fig. 3: System Design Block Diagram

Fig. 4 shows the program execution flow used in the implementation stage of the designed prototype. Normally, the FreeRTOS program is implemented in form of task modules that are executed only when required.

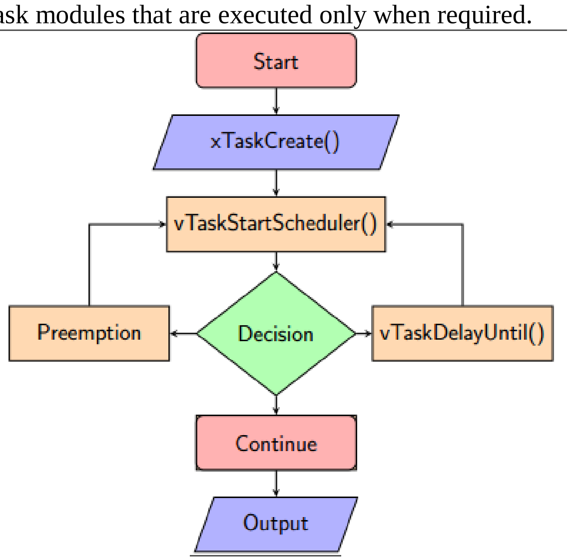

Fig. 4: Program Execution Flow Chart

V. RESULTS AND ANALYSIS

The Industrial environmental monitoring was achieved by having sensor modules capture the industrial behavior and then the results displayed on theOLED display considered to be part of the HMI section of the research. Theexperimental setup was subjected to different environmental conditions to determine the behavior of the system, and all wasconsistent and deterministic. Fig. 5 shows the snapshot of the results as being monitored on the OLED display embedded on the LPC1769 Baseboard. The implemented system can as well be monitored via a network and run-time statistics can be analyzed to ascertain the real-time behavior of the system as shown in Fig. 6.

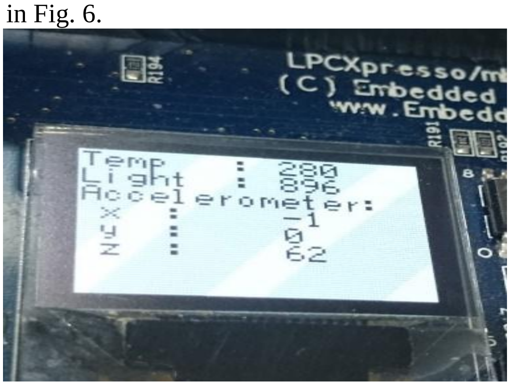

Fig. 5: Sensor Data Displayed on An OLED

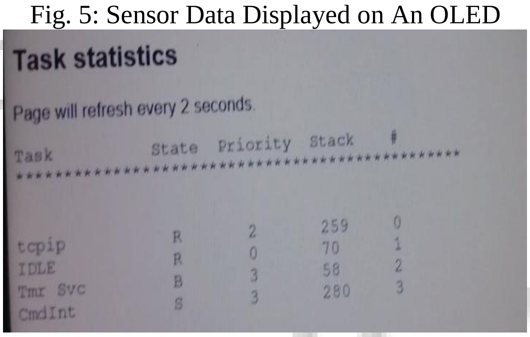

Fig. 6: Run-Time Statistics of The System

VI. CONCLUSION

Implementation of a Real Time Systems (Prototype) on FreeRTOS Platform for Industrial Automation Applications serves as a benchmark for future research work to standardize industrial control systems with FreeRTOS platforms for better performance and guarantee of response within strict time constraints.

VII. ACKNOWLEDGMENT

First, I would like to thank the almighty for giving me strength and courage in carrying out this academic research. I acknowledge my guide Mr. Babu Krishnamurthy, the co-guide Mr. Rajesh Sola for supporting me with the technical guidance. I do thank GTU and C-DAC administrators for the wonderful environment and facilities provided to us. Lastly to Mbarara University of Science and Technology (MUST), ICCR India, My Guardian Mr. Muhangi Alex, my wife Olivia Rutagangibwa for all the support financially, spiritually and encouragements given to me in achieving my career goals.

REFERENCES

[1] T. Cucinotta, A. Mancina, G. F. Anastasi, G. Lipari, L. Mangeruca, R. Checcozzo, and F. Rusin, “A real-time service-oriented architecture for industrial automation,” vol. 5. No. 3. August 2009.

[2] C.R. technologies Ltd. (2014) LPC1769 processor implementation specifications. [Online].http://www.lpctools.com/lpc1768.lpcxpresso.aspx.

[3] H. Kopetx, “Real time systems: Design principles for distributed embedded applications.” [Text book]

[4] ValeriyVyatkin, Software engineering in industrial automation: State-of-the-art review, IEEE Transactions on Industrial Informatics, Vol. 9, August 2013.

[5] Richard. Barry, Real time engineers ltd: FreeRTOS main website, http://www.freertos.org, October 2014.

[6] Francesco Basile, Pasquale Chiacchio, and Diego Gerbasio, “On the implemen- tation of industrial automation systems based on PLC”, IEEE, Transaction science and Engineering, vol. 10, October 2013, pp. 990 – 1003

[7] PankajSagar, Abdul Nazer K. H, and Likhin M, “Implementation of cmnn based industrial controller using VxWorks RTOS ported to MPC826”, Third International Conference on Advances in Computing and Communications, 2013.

[8] Pengcheng Huang, “Implementation of porting RTOS FreeRTOS to ARM7”, J. Chinese electronic business communications market, June 2009, pp. 59 – 64.

[9] VicentRutagangibwa and Babu Krishnamurthy. "A Survey on the Implementation of Real Time Systems for Industrial Automation Applications", International Journal for Scientific Research and Development 1.7 (2014): 174-177