Benchmarking Real-Time Linux Implementation on Embedded Platform

Jay Kothari
VLSI & Embedded Systems Design
GTU PG School
Ahmedabad, India
jaikothari10@gmail.com

Mr. Babu Krishnamurthy
Principal Faculty,
School of Embedded
Bangalore,India
babu_krishnamurthy@yahoo.com

Abstract—This paperdeals with design, implementation and testing of real time drivers for I2C and UART processor controllers on Beaglebone Black. Embedded Board runs with Linux 3.8.13 and real time co-kernel, Xenomai-2.6.3. Beaglebone Black has cortex A8 processor with 1GHz frequency. Xenomai Real time driver Model(RTDM) drivers are made for I2C and UART processor controller and their performance parameters were tested.

Keywords—RTOS,Xenomai, Beaglebone Black, I2C RTDM,UART RTDM.

*****

I. INTRODUCTION

An embedded systems are mainly used in real world application. Where in real world application, there are time constrains, so Linux would not be competent for such application. Hence Real Time operating systems(RTOS) were raised, which are competent for such application due to different design and implementation. RTOS are classified into Hard real time and soft real time operating system. In Hard real time operating system, it should absolutely hit every deadline otherwise it may lead to failure of the system. In soft real time operating system, it can miss some of its deadlines. This never leads to a failure of system, but it may be possible performance of the system may diminish. Choiceof RTOS depends on the demand of application the developer wants and performance parameter of the RTOS.

Embedded system consists of embedded platformwhich has many processor controllers which in turn areinterfaced to peripherals. Device Drivers are an important component in the embedded system as theycontrol the interconnection of hardware and software.

This paper would design, implement and Test, Xenomai Real Time Device Model(RTDM) for I2C andUART processor controller. Testing with various performance parameters is the goal of this paper. Various performance parameters are described in section

II. RELATED WORK

Some of the papers referred to achieve goals for this paper are Scheduling Policy and its Performance for the Embedded Real time system by Mr. P.S.Prasad, Dr.Akhilesh Upadhyay have talked about variousscheduling policies of Real time system and Linux.Linux has Fair scheduling Algorithm which gives achance to every process to execute while RTOS haveFIFO for different priority task and RR for the samepriority task. This paper analyzed the scheduling policy in order to select the right policy for the specificembedded application and also suggested the performance for advantage[2].

Performance Comparison of VxWorks, Linux, RTAI and Xenomai in a Hard Real-time Application byA.Barbalacel, A. Luchetta, G. Manduchi, M. Moro,ASoppelsa and C. gives various results for comparison. The test was done on VMEbus MVME5500boards equipped with a MPC7455 PowerPC processor.Various operating system are running of thisboard for testing Interrupt Latency, Rescheduling andinterprocess communication. Performance measurement shows that the tested open-source software issuitable for hard real-time application. The resultshows that RTAI and Xenomai are noted to be considerable. RTAI has a slightly higher performance thanXenomai because the Layered Approach. While onthe other hand Xenomai has better structure andsupports many embedded platforms. Network performance represents a strong point in favor of theRTAI and Xenomai. Otherwise Vxworks has better performance [3].

On Performance of Kernel Based andEmbedded RealTime Operating System: Benchmarking and Analysis by MasturaD. Marieska, Paul G.Hariyanto, M.Firda Fauzan, Achmad Imam Kistijantoro This paper explains various types of RTOS. Itcompares RT Patch Linux , Xenomai and eCos forembedded RTOS.Benchmarking is done by comparingfour performance metrics and assessing performancemetrics.The four performance metrics are Interruptinglatency, task switching time, pre-emption time, anddeadlock break time. Testing result shows that embedded RTOS is appropriate for executing one task application while kernel based RTOS is suitable to execute multitasking applications. Testing result alsoshows that the network packet processing of embedded RTOS is faster than kernel based RTOS [1].

III. XENOMAI

Xenomai is a Real Time subsystem which tightly mingleswith the Linux kernel to give Hard Real Timeperformance. It is co-kernel Approach with a smallco-kernel running side by side with Linux for providing determinism . Xenomai was created for migrationof industrial application to the GNU/Linux environment. Xenomai user space interface is released under LGPL2.1 and Xenomai core is released under GPL 2.

- **Interrupt pipeline**:The Linux kernel and Xenomai Core needed to talk to each other. This is done by interrupt pipeline (I-pipe).
- **Hardware abstraction layer**: This written inmachine level language. HAL is platform dependent code so that high end software remains platform independent.
- **Xenomai Nucleus and Core**:Xenomai Core isthe abstract RTOS. These building blocks aregathered to form a Xenomai Nucleus.

 ISSN: 2321-8169
2676 – 2678

- **Xenomai Skin**: Skins are various API Xenomai provides using same coreimplementation.Xenomai provides various skinslike POSIX,VxWorks,pSOS+ ,VRTX ,uITRON,RTAI 3.x, Native API, and Real-Time DriverModel (RTDM)[6].

There are two modes in Xenomai Real Time operating system.

1. **Primary mode**: In this mode kernel is deterministic and has minimum latency.
2. **Secondary mode**: In this mode kernel is non-deterministic Switching between these modesautomatically handle. No intervention of programmers is needed.

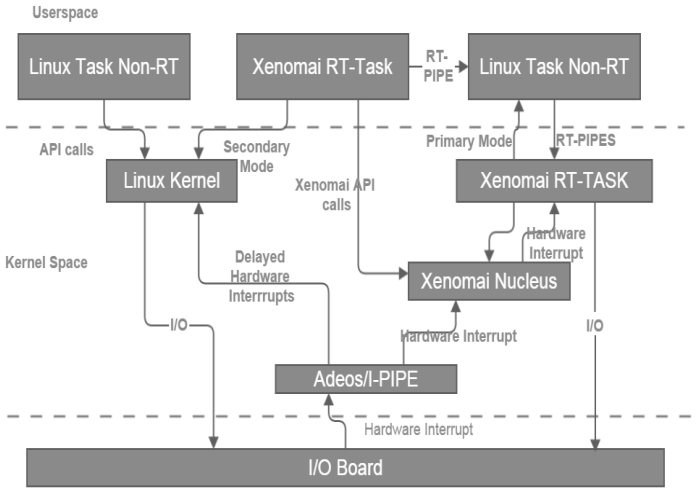

Figure 1 Xenomai Architecture

Xenomai RTDM is a common framework that would define a normalized interface between the real-time kernel and the device driver as well as between user process and device driver.

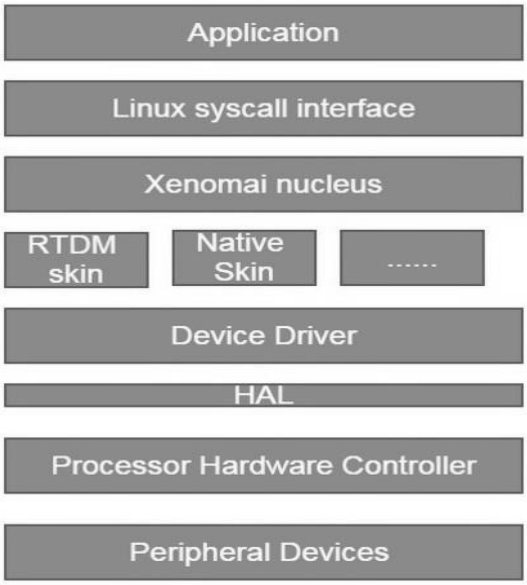

Figure 2: RTDM Driver

IV. PERFORMANCE PARAMETER

Here we expect that Hardware has no issue with performance Then Performance measurement can be divided into

1. **Throughput**: This is not popular in the hard realtime system but still it needs to be taken care.

2. **Responsiveness**: It means speed with whichembedded system responds to an event. Theremay be some latency due to
   - **Interrupt Latency**: The amount of time it takes embedded system to get the first instruction of the interrupt routine to a particular interrupt.
   - **Dispatch Latency**: The interval of time fromthe last instruction of your interrupt handler until the first interrupt caused to become runnable.
   - **Context switch time**: It's time interval between two consecutive processes.
   - **Lazy context switching**: When switchingfrom old to new process, not all the registers are saved If new process needs resourceregisters than resources are given at time ofrunning processes. This might not include inthe performance measurement for all processor Processors like 80x86,SPARC andMIPS chips does this.

3. **Determinism**:Determinism means what verse case embedded system can agree upon. Throughput and Responsiveness are inversely proposal,which means if there is an increase in throughputthen Responsiveness is poor.

V. RTDM DRIVER FOR I2C PROCESSOR CONTROLLER

In BeagleBone Black there are three I2C controllers out of one I2C is used with MMC communication andother are disabled. To use them, necessary settingis required to be done in the device tree. A Linux framework for I2C can be studied from [4].

For programming I2C processor controller

- Program the pre-scaller.
- Enable interrupt mask.
- Check for bus busy. I2C bus is in busy state until stop condition is not occurred after start condition.
- Enable the controller in master mode and start condition in the control register.
- Configure the master address and number of bytes for i2c transaction.
- If writing to the I2C device transmit mode is enabled from control register.
- If reading from I2C device receive mode is enabledfrom control register.
- Stop condition would be generated by the controller itself when count in the data count register getsread or write[5].

The I2C RTDM driver was tested by interfacing BMP085 which is pressure,temperature and altitudesensor.

VI. RTDM DRIVER FOR UART PROCESSOR CONTROLLER

In BeagleBone Black there are six UART controllers.One is used for serial console and other are disablewhich needs to be enabled. A Linux framework forUART can be studied from [4].

For programming UART processor controller

- Enable interrupt on Receiver and Transmitter byInterrupt Enable register(IER).
- Write to FIFO control register(FCR) for clearingFIFO of Rx and Tx.
- Program Line control register(LCR) with variousparameters like parity bit sets,number of stopbit,word length to transmit and Receive.
- Write values to divisor latch low and high register (DLL and DLH) according to baud rate selection.
- Enable Flow control: Hardware or Software according to requirement.
- Interrupt are identified by Interrupt identificationregister(IIR) and necessary work is done.
- Hence, data are sent and received by the interrupt[5].

The UART RTDM driver was tested by connecting to BeagleBone Black via serial communication cross wires.In this paper modem control register is not programmed because work does not use other pins exceptTransmit and Receive.

VII. RESULTS

This paper aims to measure interrupt latency andscheduling latency of RTDM driver and Non-RTDMdriver.

	RTDM Driver for I2C	*RTDM Driver for UART*
Interrupt Latency	*11.958*	*9.5*
Scheduling Latency	*16.2583*	*13.084*

Table 1. Results

All units are in micro-second.
These Latency is far less than normal Linux, which might get up to mili-second.

VIII. CONCLUSION

In this paper,Xenomai RTDM driver for I2C andUART hardware controller in Cortex A8 processor isdesigned and implemented and Tested. The RTDM drivers areable to register with Xenomai and Xenomai kernelwhich make it visible to user space applications. AlsoRTDM drivers are verified by interfacing an I2C sensor and UART device. Various arguments are passedfrom user space application and necessary error arechecked properly so that it ensures reliable communication. Results shows than Xenomai RTDM drivers are far more deterministicand can be used for real time application. Future work would to make RTDM driver formore processor controllers in Cortex A8 processor.

REFERENCES

[1] MasturaD. Marieska, Paul G. Hariyanto, M. Firda Fauzan, Achmad Imamand Afwarman Manaf, "OnPerformance of Kernel Based and Embedded RealTime Operating System: Benchmarking and Analysis," Advanced Computer Science and Information System(ICACSIS), pp. 401-406, Dec 2011

[2] Mr.P.S.Prasad1,Dr.Akhilesh Upadhyay, "Scheduling Policy and its Performance for the EmbeddedReal time System," International Journal of Advanced Research in Computer and Communication Engineering,Vol.2,April 2013.

[3] A.Barbalacel, A.Luchetta, G.Manduchi, M. Moro,A.Soppelsa and C.Taliercio, "Performance Comparison of VxWorks, Linux, RTAI and Xenomai in a Hard Real-time Application," IEEE,Vol:55,pp.435-439,February 2008.

[4] Sreekrishnan Venkateswaran, Essential Linux Device Drivers, Prentice Hall,March 2008.

[5] Sreekrishnan Venkateswaran, AM335x SitaraTM Processors Technical Reference Manual, Texas Instruments,June 2014.

[6] Karim Yaghmour,Jon Jason Masters,Brittain Gilad and Ben-Yossef, Ian F. Darwin and PhilippeGerum, Building Embedded Linux Systems Tomcat, O Reilly Media, Inc,August 2008.

[7] Jonathan Corbet, Alessandro Rubini, and GregKroah-Hartman, Linux Device Driver 3rd edition, O Reilly Media, Inc,Jan 2005.

[8] www.xenomai.org,