International Journal for Research in Applied Science & Engineering Technology (IJRASET)

RT-Linux Based Schedule Priority System for Remote Location

A.Venkateswarlu[1], P. Sreenivasulu[2], M.Tech(ph.d)
Department of Electronics and communication Engineering
Audisankara College of Engineering & Technology, Gudur (Autonomous)

Abstract- online task programming in heterogeneous multi core system-on-a-chip could be a difficult drawback because of precedence constraints and non pre-emptive task execution within the synergistic processor core. This study 1st proposes an internet heterogeneous dual-core programming framework for dynamic workloads with period constraints. The overall purpose processor core and therefore the synergistic processor core square measure dedicated to separate schedulers with completely different programming policies, and precedence constraints among tasks square measure restrained through interaction between the 2 schedulers. This framework is additionally configurable for low priority inversion and High system utilization. We tend to then extend this framework to heterogeneous multi core systems with well-known dispatcher schemas.
This paper presents a true case study to indicate the practicable ness of the projected methodology, and presents a series of in depth simulations to get comparison studies victimization completely different workloads and programming algorithms.
Index Terms – Real-Time Scheduling, Heterogenous multicore system, Web-baesd monitoring.

I. INTRODUCTION

In the past decade, several studies have extended synchronization protocols to resolve task programming issues in heterogeneous multi core systems by managing the non-preemptive coprocessor as a resource. The management of priority inversion ends up in poor system utilization. this can be as a result of lower priority task execution within the processor is impermissible even once the processor is idle, if a better priority task is dead within the coprocessor. Previous analysis has extended digital computer programming algorithms for unvaried multi core SoCs to avoid the matter of priority inversion management. However, these algorithms haven't been tailored for heterogeneous multi core systems. The explanation behind this can be that the processor and coprocessor square measure uneven, and therefore the coprocessor isn't fitted to preventive task execution thanks to important preemption overhead. This overhead comes from the quantity of registers, pipeline stages and cache flushes. The previous may be results of the look of coprocessors, and also the latter is because of the many size of knowledge and directions on transmission platforms. This work is actuated by the requirement for web computer hardware for heterogeneous multi core systems, and also the issue obligatory by the trade-off between priority inversion management and system utilization improvement. in a very heterogeneous multi core atmosphere, the execution time of a package task depends on that processor core it executes on. as an example, a package task activity camera work rendering, simulating physics, or estimating trajectories of flying objects runs abundant quicker on a graphics processor than on a standard processor. Conversely, some package tasks area unit inherently ordered and can\'t take pleasure in the graphics processor; they execute abundant quicker on a standard processor. as an example, a package task with several branches and no inherent correspondence runs abundant quicker on a standard processor than on a graphics processor. Ideally, every task would be assigned to the processor wherever it executes with the best speed, however sadly the work is usually not utterly balanced to the categories of processor cores accessible. Economical use of process capability within the new generation of microprocessors so needs that tasks area unit assigned to processors showing intelligence. During this context, “intelligently” implies that the resources requested by the program area unit those possessed by the processor. Moreover, the will for brief style cycles, speedy fielding, and upgrades necessitates that task assignment be done mechanically with algorithms and associated tools.

II. REAL-TIME SCHEDULING

Scheduling is the process of determining which task to run at a given time. Scheduling is a key task for any modern operating

system, but especially for a real-time operating system (RTOS). A real-time scheduling System is composed of the scheduler, clock and the processing hardware elements. In a real-time system, a process or task has schedulability; tasks are accepted by a real-time system and completed as specified by the task deadline depending on the characteristic of the scheduling algorithm. Modeling and evaluation of a real-time scheduling system concern is on the analysis of the algorithm capability to meet a process deadline. A deadline is defined as the time required for a task to be processed.

A. Basic RT Scheduling Rules

The following rules summarize the operation of Real-Time scheduler when a new thread enters the run queue.

1) Perform preemptive scheduling among threads of different priority.
2) Perform round-robin scheduling among threads of equal priority.
3) Time-critical threads always run to completion.

B. Understanding how RT Scheduling Works

At any given time, each thread in the system is either running or blocked.

1) *Running*—the thread is in the run queue, which means it is either ready to execute or is currently executing.
2) *Blocked*—the thread cannot execute until some event occurs, such as the completion of an I/O operation or a timing function.

When a thread begins executing, it runs until one of the following conditions arises:
The thread becomes blocked by a Wait within a While Loop, the built-in timing mechanism of a Timed Loop, or a blocking function such as the Read Variable with Timeout function.
The thread is interrupted by a higher-priority thread.
The thread finishes executing.
When a thread becomes unblocked, it enters the run queue. The run queue is always sorted in priority order, so when a thread enters this queue, it immediately moves ahead of all lower-priority threads. If a thread entering the run queue is higher-priority than a currently-running thread, the higher-priority thread either runs on a different CPU core or interrupts the currently-running thread in a process called preemptive scheduling.

C. Preemptive Scheduling

Preemptive planning is that the method of ordering thread execution supported the relative priorities of the threads. Once a thread of upper priority than this thread enters the run queue, the computer hardware interrupts this thread if necessary in order that the higher-priority thread will execute straightaway. The interrupted thread then returns to the run queue behind the higher-priority thread.

D. Round Robin Scheduling

Round robin programming is that the method of alternating between threads specified every thread receives roughly equivalent central processing unit time. Throughout spherical robin programming, the hardware interrupts and switches between the threads at regular intervals, keeping track of the central processing unit time given to every thread.

E. Priority Based Scheduling

Although it appears intuitive to line the priority of every loop supported the perceived importance of the loop, this strategy will cause disturbance. Set the priority of every loop not supported however crucial the task is at intervals the appliance however rather supported however necessary it\'s that the task fulfills an exact temporal order guarantee. For instance, in a very knowledge work application with an information acquisition loop and a work loop, you would possibly be tempted to form the work loop the higher-priority loop. However, knowledge acquisition should occur at regular intervals to avoid incomprehensible knowledge points. However, it doesn't matter after you log every information to disk, as long as you log every purpose eventually. During this case, you must assign the next priority to the information acquisition loop, despite the fact that knowledge work is that the primary purpose of the appliance..

III. A WEB-BASED MONITORING

The exponential growth of web and technology allows the event of complicated, hybrid systems that offers bigger concern in maintenance and has additional flexibility in union and fault finding. With the advanced technology industries have an interest in automation by introducing remote observance and system for the menstruation management of commercial method parameters terribly exactly and accurately for the standard product. The local area network provides an affordable entranceway through that to knowledge transfer for period interaction of the remote observance and management of the parameter provide several benefits.

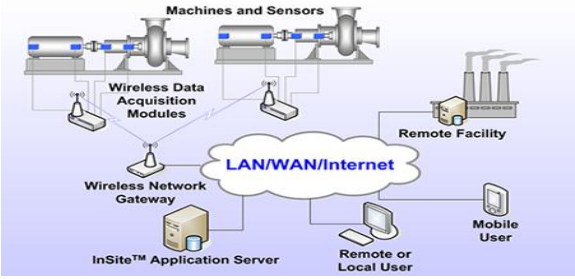


Fig1. Web-based Monitoring of Machine and Sensor

Ethernet/internet-based information watching brings new capabilities and new access to method activity and management. victimization commonplace method sensors, like thermocouples or RTDs (temperature), pressure transducers, flow meters or different sensors that manufacture a regular analog or pulse output, you\'ll be able to monitor, management or log information in virtually any location - across the hall, on the opposite aspect of the road, across city, on opposite ends of the country, or anyplace round the world. With Ethernet-based, internet-enabled instrumentation, remote access are often anyplace a Smartphone incorporates a signal. From the only application, viewing information through the web-browser on your iPhone, Blackberry device or laptop computer, to additional subtle uses, like causation a text or e-mail message once AN alarm happens, or transmittal a knowledge log file over the net from a foreign location to a home office. A user will access this information anytime, anywhere, twenty four hours every day, 12 months a year, where you\'ve got web access.

IV. SYSTEM ARCHITECTURE

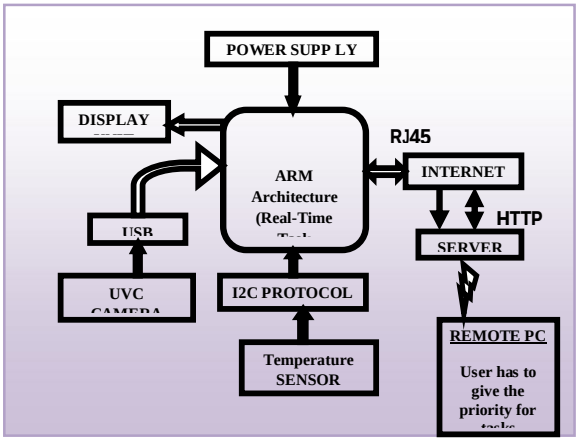


Fig2. Framework for online real time task scheduling

The framework of projected system consists of ARM thirty two bit small controller that acts as period of time task hardware for heterogeneous multi-core system. The devices like USB camera, Temperature device and LDR device that were interfaced to ARM board square measure the coprocessors. Once system is turned ON the devices interfaced to the USB host and I2C protocol starts operating. The devices connected to USB host (ex: USB camera) and I2C (ex: temperature device, LDR) endlessly transmits information to controller. The controller transmits information that is coming back from USB and I2C to server through web by victimization FTP. FTP may be a protocol through those users will transfer files from their systems to server. Once information is

placed at server we are able to read the information at remote laptop (with internet) on online page with distinctive information science address provided. we are able to read continuous streaming of video, temperature information moreover as LDR device information.

If we would like amendment to vary to alter} the programming of the system we are able to change through online page from remote location victimization protocol. Protocol endlessly requests the server for {the information the info the information} if any changes had been done at online page like dynamic the programming of either USB or I2C and receives data from server and modifies the programming of the tasks within the controller. Currently take into account highest priority is given to USB at online page. Once modification is finished then the server sent the changed information to controller by victimization protocol and controller changes the programming of tasks within it according that changed information. Once programming is modified with success, the information that is coming back from USB can update endlessly on online page however the information from I2C can stop change its price, merely it shows last monitored condition. During this approach we are able to amendment priority condition on remote laptop.

A. Hardware Requirements for Implementation

1) MINI2440 Development board

Fig3. Arm9 Board

The ARM9 processor is good for several time period embedded applications with stern size constraints and cost-sensitive issues. The improved DSP extensions within the ARM9 processor take away the necessity for a separate DSP within the SoC style, leading to further savings in chip complexness, power consumption, and time-to-market. Today, the ARM family accounts for roughly seventy fifth of all embedded 32-bit computer architecture computer hardware are, creating it the foremost wide used 32-bit design. ARM CPUs ar found in most corners of client physical science, from moveable devices (PDAs, mobile phones, iPods and different digital media and music players, hand-held vice units, and calculators) to pc peripherals (hard drives, desktop routers)

2) UVC Camera Driver

Fig4. UVC Driver Camera

A UVC (or Universal Video Class) driver may be a USB-category driver. A driver allows a tool, like your digital camera, to speak along with your computer's OS. And USB (or Universal Serial Bus) may be a common style of association that permits for high-speed knowledge transfer. Most current operational systems support UVC.

3) Ethernet LAN

Ethernet could be a family of laptop networking technologies for native space networks (LANs). local area network was commercially introduced in 1980 and standardized in 1983 as IEEE 802.3. local area network has mostly replaced competitive wired LAN technologies like token ring, FDDI, and ARCNET.The local area network standards comprise many wiring and signal variants of the OSI physical layer in use with local area network. knowledge rates were sporadically redoubled from the first ten megabits per second to one hundred gigabits per second

Volume 3, Special Issue-1, May 2015
ISSN: 2321-9653
International Journal for Research in Applied Science & Engineering Technology (IJRASET)

B. Temperature Sensor

Fig5. Temperature Sensor

LM35 is a precision IC temperature sensor with its output proportional to the temperature (in ^{o}C). With LM35, temperature can be measured more accurately than with a thermistor. The operating temperature range is from -55°C to 150°C. The output voltage varies by 10mV in response to every ^{o}C rise/fall in ambient temperature, *i.e.*, its scale factor is 0.01V/^{o}C.

C. Software Requirements for Implementation

1) *Rt Linux Operating system:* RT-Linux could be a arduous real-time RTOS microkernel that runs the whole UNIX system OS as a totally preventative method. it\'s one in all the arduous time period variants of UNIX system, among many, that produces it potential to regulate robots, knowledge acquisition systems, producing plants, and alternative time-sensitive instruments and machines. RT-Linux controls instrumentation for independent agency, high-speed Active Magnetic Bearings (AMBs) for the University of Virginia Rotating Machinery and Controls Laboratory, animatronics puppets at the Jim Henson Creature look and satellite base stations for Japan Post and Telegraph..

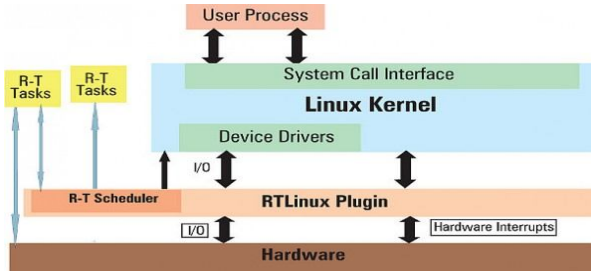


Fig4. RTLinux Operating Sytem

2) *Qt for embedded Linux:* Qt for Embedded UNIX could be a C++ framework for interface and application development for embedded devices. It runs on a range of processors, typically with Embedded UNIX. Qt for Embedded UNIX provides the quality Qt API for embedded devices with a light-weight window system.

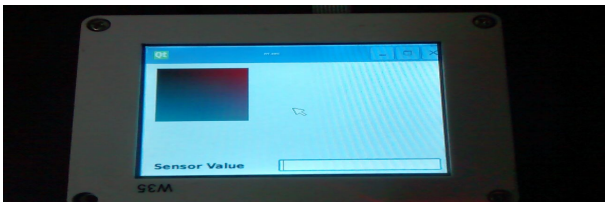
3) *HTTP(Hypertext Transfer Protocol):* HTTP is associate degree application protocol for distributed, cooperative, object-oriented database management system info systems. HTTP is that the foundation of information communication for the planet Wide internet. machine-readable text is structured text that uses logical links (hyperlinks) between nodes containing text. HTTP is that the protocol to exchange or transfer machine-readable text.

V. RESULTS

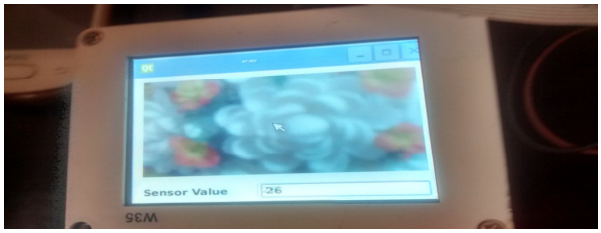
(1)Fig: Board Setup:

(2)Fig: GUI on LCD Disply of Board

Volume 3, Special Issue-1, May 2015
ISSN: 2321-9653
International Journal for Research in Applied Science & Engineering Technology (IJRASET)

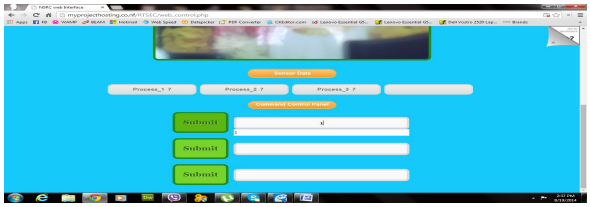


(3)Fig: Coprocessors transmits data to Controller board

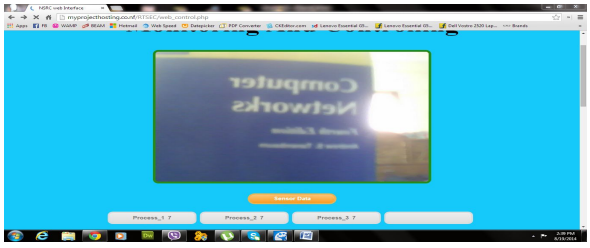


(4)Web Page GUI displays the data sent by controller

(5)Web page consists of Command Control panel in which user has to specity the priority to tasks



(6)Controller upadets the data sent by coprocessor on web page based on priority of task



VI. CONCLUSION

The project titled “RT-Linux Based Schedule Priority System for Remote Location” has been successfully designed and tested. It has been developed by integrating features of all the hardware components and software used. Presence of every module has been reasoned out and placed carefully thus contributing to the best working of the unit. Secondly, using highly advanced ARM9 board and with the help of growing technology, the project has been successfully implemented.

REFERENCES

[1] Texas Instruments, Inc., "OMAP3 Platform," technical report, Texas Instruments, http://www.ti.com/lit/ml/swpt024b/swpt024b.pdf, 2009.
[2] Texas Instruments, Inc., "OMAP4 Platform," technical report, Texas Instruments, http://www.ti.com/lit/ml/swpt034b/swpt034b.pdf, 2011.
[3] Qualcomm, Inc., "Snapdragon," technical report, Qualcomm,
http://www.qualcomm.com/media/documents/snapdragons4-processors-system-chip-solutions-new-mobile-age, 2011.
[4] L. Sha, R. Rajkumar, and J. Lehoczky, "Priority Inheritance Protocols: An Approach to Real-Time Synchronization," IEEE Trans. Computers, vol. 39, no. 9, pp. 1175-1185, Sept. 1990.
[5] T.P. Baker, "Stack-Based Resource Allocation Policy for Real-Time Process," Proc. Real Time Systems Symp. 1990.
[6] P. Gai, L. Abeni, and G. Buttazzo, "Multiprocessor dsp scheduling
in System-on-a-Chip Architecture," Proc. Euromicro Conf. Real- Time Systems, 2002.
[7] K. Kim, D. Kim, and C. Park, "Real-Time Scheduling in Heterogeneous Dual-Core Architecture," Proc. Conf. Parallel and Distributed Systems, 2006.
[8] S. Kato, K. Lakshmanan, R. Rajkumar, and Y. Ishikawa, "Timegraph: Gpu Scheduling for Real-Time Multi-Tasking Environments," Proc. USENIX Ann. Technical Conf., 2011.
[9] S. Kato, K. Lakshmanan, Y. Ishikawa, and R. Rajkumar, "Resource Sharing in gpu-Accelerated Window Systems," Proc. Real-Time and
Embedded Technology and Applications Symp., 2011.
[10] S. Saewong and R. Rajkumar, "Cooperative Scheduling of Multiple Resources," Proc. Real-Time Systems Symp., 1999.
[11] Y.-S. Chen and L.-P. Chang, "A Real-Time Configurable Synchronization Protocol for Self-Suspending Process Sets," Real-Time Systems, vol. 42, no. 1, pp. 34-62, 2009.
[12] L. Benini, D. Bertozzi, A. Guerri, and M. Milano, "Allocation, Scheduling and Voltage Scaling on Energy Aware MPSoCs," Proc.Conf. Integration of AI and OR Techniques in Constraint Programming
for Combinatorial Optimization Problems, 2006.
[13] M. Kim, S. Banerjee, N. Dutt, and N. Venkatasubramanian, "Design Space Exploration of Real-Time Multi-Media mpsocs with Heterogeneous Scheduling Policies," Proc. Conf. Hardware/
Software Codesign and System Synthesis, 2006.
[14] C.-F. Kuo and Y.-C. Hai, "Real Time Task Scheduling on Heterogeneous Two-Processor Systems," Proc. Conf. Algorithms and Architectures for Parallel Processing, 2010.

AUTHORS

A.Venkateswarlu received his B.TECH degree in Electronics and Communication Engineering from Priyadarshini institute of Technology, tirupati, Chithoor (Dist), affiliated to JNTU Anantpur. He is currently pursuing M.Tech Embedded systems in Audisankara college of Engineering and Technology, Gudur(Autonomous), Nellore (Dist), affiliated to JNTU Anantpur.

P.Sreenivasulu He received his M.Tech in ECE from S.V UNIVERSITY,Tirupati. He has 11 years teaching experience. He Presented 4 International Conferences. Presently working as Assoc. Professor in the department of ECE, Audisankara College of Engineering and Technology, Gudur (Autonomous), Affiliated to JNTU, Anantpur.