ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 7, July 2015

Real Time Task Scheduling

Pallavi D. Deore[1], Prof. H. B. Mali[2]

Student, Department of Electronics & Telecommunications, SITS, Pune, India[1]

Professor, Department of Electronics & Telecommunications, SITS, Pune, India[2]

Abstract: Online task scheduling in arm core is a challenging problem due to precedence constraints and no pre-emptive task execution in the synergistic processor core. This learning first proposes an online heterogeneous dual-core scheduling structure for dynamic workloads with real-time constraints. This structure is also suitable for low priority inversion and high system operation. We then spread this structure to various multicore systems by interfacing multiple co-processors to the main processor and scheduling the tasks of co-processors by using a single chip which acts as a controller to control the tasks of co-processors based on the data sent by remote pc using HTTP and FTP Protocols.

Keywords: Real time task Scheduling, ARM11, HTTP, and FTP.

I. INTRODUCTION

Generally multimedia embedded systems are fortified with a heterogeneous multicore system-on-a-chip [1], for example, one general purpose processor core and one or more synergistic processor cores (coprocessors) to advance performance and power efficiency. Therefore, a fast and expected online scheduling algorithm is an important system specification for a heterogeneous multicore system-on-a-chip. However, the scheduling problems of heterogeneous multicore systems are problematical by precedence constraints of tasks and no pre-emptive task executions in the coprocessor. Previous research has extended multiprocessor scheduling algorithms [2] for similar multicore to avoid the problem of significance inversion management. However, these algorithms have not been modified for heterogeneous multicore systems.

The rational Pre-emptive task implementation due to significant pre-emption overhead. This overhead comes from the number of registers and pipeline stages and cache flushes [3]. The earlier indicates the design of coprocessors [1], and the latter is due to the significant size of data and directives on multimedia platforms. This work is determined by the necessity for an online scheduler for heterogeneous multicore systems [1], and the difficulty imposed by the cooperation between priority inversion management and system utilization enhancement.

II. EXISTING SYSTEM

Generally in industries systems are located at different locations each has its own processor with its own tasks running elite it. We cannot control or change the programming of these tasks running in a processor from regulator room so, every time person should move to particular location and change its scheduling of the tasks.

Bjorn Anderson et al have proposed the technique for assigning Real-Time Tasks on Heterogeneous Multiprocessors with Two Unrelated Types of Processors. In this paper they present a new algorithm is FF-3C, which offers low time-complexity as well as good performance. They also present number of extensions to FF-3C; all extensions offer the same Time complexity and better performance as that of FF-3C. And they also offer better.

Average-case performance. I-Hong Hou and P. R. Kumar has proposed Scheduling Periodic Real-Time tasks with Heterogeneous Reward Requirements. In this paper they have explained the problem of scheduling periodic real-time tasks which has exact minimal reward requirements. They also consider positions where tasks generate instance that can be provided Subjective service times before their deadlines, and get rewards based on the service times received by the instance of the task. They also wind up that this model is appropriate with the estimated computation models also increasing reward with growing service models.

III.SYSTEM ARCHITECTURE

We are focusing on real-time online scheduling for a heterogeneous multicore system-on-a-chip, in which a general purpose processor core is supported by one or more synergistic processor core(s), such as Digital Signal Processors (DSPs) for signal processing or Graphics Processing Units for graphics. For ease, we assume the processor represents the general purpose processor core, and the coprocessor signifies the synergistic processor core. To emphasis on the scheduling problem of processors, we have further predicted that the processor and the coprocessor communicate using dedicated shared memory and a mailbox.

The mailbox is used for sending requests from the processor to the coprocessor and vice versa. Under the shared memory ideal, the worst case communication time can be bounded by or analysed as a part of the worst case performance time. This paper simply considers the communication cost as a part of the execution time without special identification. For more feature regarding

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

the statement planning implementation, refer to the approaches proposed in for DSPs and, for GPUs. In most heterogynous multicore systems, the processor performs managing control signals, and coprocessors are in charge for data computation. Therefore, a task executed on such a system is a collection of subtasks with a partial organization. If a subtask is placed ahead of another in the partial direction, then the latter cannot start until the former completes execution. Each subtask is statically assigned to be executed on a processor or a coprocessor. The subtasks of a task executed in the processor and coprocessor are referred to as processor subtasks and coprocessor subtasks. Moreover, we undertake that the execution of subtasks of a task is divided between the processor and coprocessor. For example, when the subtask is executed in the processor, and then the subtask is executed in the coprocessor. Most requests are executed in a varied system with timing constraints, which are the influx time, poorest case computation time, response time, and relative limit of a task noted, by ai, ci, ri, and Di in this paper. Task scheduling in the processor is preventative, but task execution in a coprocessor is usually non pre-emptive to avoid significant context-switch overhead.

For this paper, the context switch overhead includes register saving/restoring, pipeline stage flushes, and cache flushes. However, non-pre-emptive scheduling might introduce a longer blocking time for an emergency task such that the task may violate its timing constraint time. This paper simply considers the communication paper, the context switch overhead includes register redeemable/restoring, pipeline stage flushes, and cache flushes. Even though, non-pre-emptive scheduling might introduce a longer blocking time for an emergency task such that the task may violate its timing constraint.

IV. PROPOSED SYSTEM

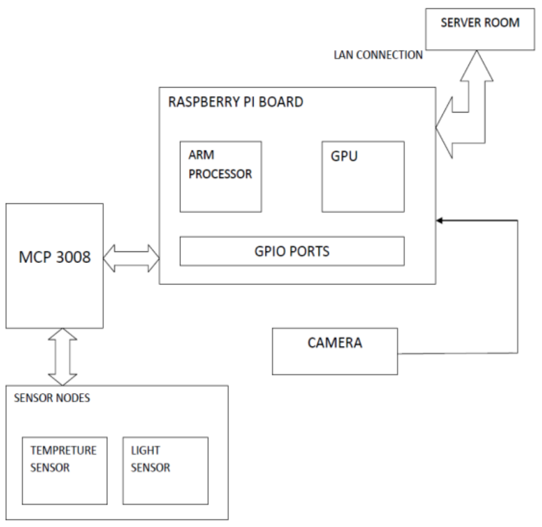

Fig. 1 ARM system architecture

ARM is a 32-bit RISC processor architecture developed by the ARM Corporation. ARM processors possess a unique combination of features that makes ARM the most popular embedded architecture today. First, ARM cores are very simple compared to most other general-purpose processors, which means that they can be manufactured using a comparatively small number of transistors, leaving plenty of space on the chip for application specific macro cells. A typical ARM chip can contain several peripheral controllers, a digital signal processor, and some amount of on-chip memory, along with an ARM core. Second, both ARM ISA and pipeline design are aimed at minimizing energy consumption a critical requirement in mobile embedded systems. Third, the ARM architecture is highly modular: the only mandatory component of an ARM processor is the integer pipeline; all other components, including caches, MMU, floating point and other co-processors are optional, which gives a lot of flexibility in building application-specific ARM-based processors. Finally, while being small and low-power, ARM processors provide high performance for embedded applications. For example, the PXA255 scale processor running at 400MHz provides performance comparable to Pentium 2 at300MHz, while using fifty times less energy.

V. RESULT

Following are the result and experimental setup. These have been analysed with thorough testing with precise inputs

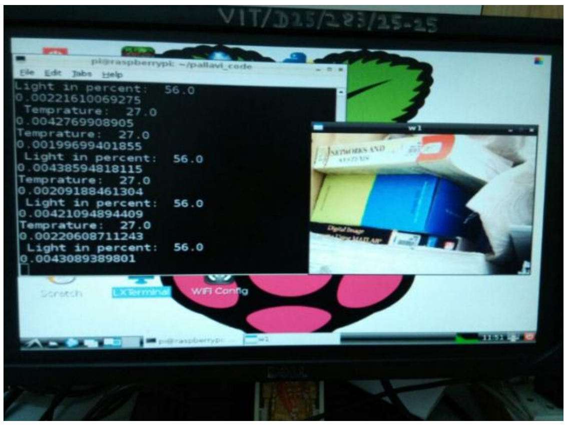
```
Light in percent:  56.0
0.00221610069275
 Temprature:  27.0
0.0042769908905
Temprature:  27.0
0.00199699401855
 Light in percent:  56.0
0.00438594818115
Temprature:  27.0
0.00209188461304
 Light in percent:  56.0
0.00421094894409
Temprature:  27.0
0.00220608711243
 Light in percent:  56.0
0.0043089389801
```

Fig.2 Result

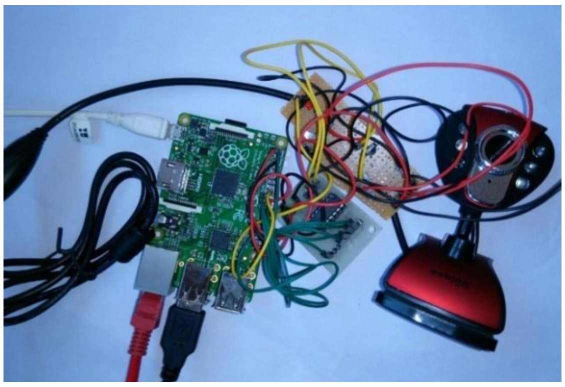

Fig.3 Experimental setup

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 7, July 2015

TABLE I RESULT READING ANALYSIS

Sr. No.	**Task 1 (Temp)**	**Task2 (Light)**	**Task 3 (PWM)**
1	1.3 m sec	0	0
2	0	0	998.3 m sec
3	1.3 m sec	0	0
4	0	0	998.3 m sec
5	0	1.3 m sec	0
6	0	0	998.3 m sec
7	1.3 m sec	0	0

VI. CONCLUSION

This paper sees the sights on the online real-time task scheduling problem in heterogeneous multicore systems, and considers tasks with priority constraints and no pre-emptive task execution. We firstly propose a heterogeneous dual-core scheduling framework to provide fast admission control able to accept active workload. A pre-emption point-based solution is presented as a way to configure task blocking time and context switch overhead in the coprocessor. Future research should explore the online energy efficiency polices of heterogeneous multicore systems, the utilization certain of heterogeneous multicore systems and the communication architecture of heterogeneous multicore systems. Additional research and development in these areas may prove to be very important for future mobile system designs.

VII. 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] Texas Instruments, Inc., "OMAP3 Platform," technical report, Texas Instruments, http://www.ti.com/lit/ml/swpt024b/ swpt024b.pdf, 2009.
[5] Texas Instruments, Inc., "OMAP4 Platform," technical report,Texas Instruments, http://www.ti.Com/lit/ml/swpt 034b/s wpt 034 b.pdf, 2011.
[6] Qualcomm, Inc., "Snapdragon," technical report, Qualcomm,http://www.qualcomm.com/media/documents/sna pdrago s4-processors-system-chip-solutions-new-mobile-age, 2011