AN ADAPTIVE SCHEDULER OF OPERATING SYSTEMS USED IN IOT DEVICES

Prachi C. Khanzode, Assistant Professor Sipna College of Engineering and Technology, Amravati, India.
Dr. D.G. Harkut, Associate Professor Prof Ram Meghe College of Engineering and Management, Badnera Amravati, India

Abstract:
Nowadays most of the devices work with Internet of Things. Internet of Things can be defined as the connectivity of smart devices which are able to connect and communicate with the help of Internet. These IoT devices perform various tasks while executing. As IoT devices must be efficient in working hence it is very important that the internal design of devices must be effective. Always the working of any device depends on its architecture hence the architecture of IoT devices should work properly. The center of any IoT device is its operating system. IoT devices chiefly exist in tiny size; use less memory, runs with less energy having high out turn. These Scant resources pass specially designed dedicated tiny size operating system to bolster IoT devices with varied applications and multifariously helpful specifications. In IoT devices operating system plays a vital role to manage the needed elements of operating system in a correct manner. This paper reviews the operating systems of IoT devices.
***Keywords*:** Internet of things, Sensors, Smart devices, Low end Devices

I. Introduction

Internet of Things can be defined as the connectivity of smart devices which are able to connect and communicate with the help of Internet. IoT Environment is consisting of a number of various components such as actuators, sensors, camera and embedded code which is used to feel sound, light, distance and motion [1]. Because the World is enhancing all its technologies, we tend towards the advancement of automation with smart homes; smart cities and smart planet allare accomplished with very smart IoT devices having the aptitude of acting varied tasks in their own manner [2, 3]. In this technological era the use of IoT devices may go beyond expectations in coming ten years [4].

With the Evolution in IoT devices, it is necessary that evolution of operating system also takes place with it. Whenever advancement in hardware takes place to cope up with desired changes OS hast to evolve accordingly. Typical OS like windows, UNIX cannot match into IoT devices due to their restricted capabilities.

Based on performance and potential of IoT devices are classified in 2 sorts [5]. The primary kind consists of high end IoT devices that are engineered on one board like Smartphone and Raspberry Pi [6]. High end devices are capable with adequate characteristics and ample resources. This class consists of general-purpose OS like BSD, UNIX or Windows. Another class consists of Low end IoT devices that have limitations of resources due to its tiny size which cannot pass ancient OS. The advancement in number of IoT applications is fast in several areas; chiefly smart cities, smart factories, smart homes, smart industrial management, and smart environment monitoring. With the forceful increase in smart applications and also the generation of heaps of knowledge each second, higher cognitive process is of significant importance in smart devices and systems [7]. As the embedded systems and autonomous technology has evolved drastically and its working is managed by real time system hence IoT devices is recent R&D area. In Real time embedded devices, code and hardware systems are subjected to numerous constraints and these embedded devices ought to respond among time constraints or deadlines [8]. Hence, the embedded devices ought to be expedited with sensible decision-making capabilities at the task computer hardware; so scheduler will best decide the order of tasks to be run and also take proper use of available resources.

Real time systems are may be hard real time systems or soft-real time systems. If the tasks within

ISSN: 2278-4632
Vol-13, Issue-10, No.03, October: 2023

the system have rigid deadlines the system is taken into account of hard real time system with the strict constraint of tasks executing before their deadline. If the tasks within the system have soft/flexible point then the system is termed as a soft real time system [9].

Low-end IoT devices have to be compelled to match the resource management challenges and for that purpose organized management of resources and its operating mechanism ought to be mentioned. Whereas working with operating system the resources plays the very important role within the out turn of the system, therefore it's obligatory to check the small print of operating system resources and its importance. The researchers study the resources like task management, File Management, Energy Management and Memory Management.

II. Literature Review

In real time embedded devices task scheduler plays an important role as it schedules processes according to specific order of execution. Any real time embedded system works in two ways hard real time systems and soft real time systems. How the processes are scheduled by scheduler is totally depends on scheduler [10]. Any Process in real time scheduling get attached with an identifier, description of process and deadline

Operating system works in two ways for scheduling purpose named as preventive and non-preemptive. Preemptive scheduling has restriction for time quantum while non-preventive approach is not limited to time limits. Because the real time OS has time constraints and task should execute among deadline and should reach to point, it follows the manner of preventive planning. Computer hardware continuously must style in such the simplest way that it should be energy economical and should be versatile in nature which boosts the performance of OS [11].

The varied programming algorithms for such systems square measure projected. Priority-based preventive programming approach, here tasks have priorities which will be statically or dynamically allotted and at any time, the task with the best priority executes. It's the latter demand that necessitates preemption: if a low-priority task is in execution and a better priority task arrives, the previous is preempted, and also the processor is given to the new arrival. If priorities square measure allotted consistently in such how that temporal order constraints is taken into consideration, then the ensuring computer hardware may be used for period of time systems [12].

Cyclic programming is a type of programming which follows the concept of various large scale periods of time systems [SI]. It could be a combination of priority programming and table-driven programming. Here, tasks square measure allotted one amongst a collection of harmonic periods. Amongst every amount, tasks square measure sent in keeping with a table that simply lists the order during which the tasks execute [13].

Static Table-Driven programming is associate degree approach that is impelled by the actual fact that resources required to satisfy the deadlines of safety-critical tasks should be reallocated in order that they will be secure a priori. These tasks square measure sometimes statically regular such their deadlines are going to be met even beneath worst case conditions [14].

Contiki OS is developed in C language. The OS runs with each programming approaches as preemptive and non-preemptive. Here Power Management is done by using sleep mode. It doesn‟t contain memory protection unit however it's supported by dynamic memory allocation. Contiki uses low flash file management [15]. It's a partial RTS. Contiki supports IPv6 protocol. Contiki OS is helpful in some ways. It's applicable in remote house observation, observation streetlights, warning device and observation the town sound. On the contrary it's certain limitations in power management and memory management. As contiki has sleep mode it ends up in wastage of energy and absence of memory protection unit, minimizes the effectiveness in memory management.

RIOT software is that the Dominant IoT OS. it's launched in 2013 with Open source Model. It's microkernel design and encompasses a multithreaded programming model. It provides a preventive programming approach. To supply power management, RIOT OS uses the Tickless computer hardware [16]. RIOT implements deep sleep mode. TinyOS is associate degree open supply

ISSN: 2278-4632
Vol-13, Issue-10, No.03, October: 2023

software that is largely the part based mostly OS. It‟s popularly used OS. It has an event driven model for multithreading. The design comes up in monolithic structure. Its partial implementation of cooperative threading approach, the programming models supported by TinyOS square measure each preventive and non-preemptive [17].

To overcome the deficiency of TinyOS, programming principle supported cooperation [18] is mentioned. Once programming a task, the high priority task is often thought of 1st. However, once one task is running, it'll possess the system resource unless it itself voluntarily provides up. Even a better priority task got to run, it's to attend for finishing the lower task. Once the lower running task finishes, the upper priority task is regular to run.

Early point first formula [19] premised that each period of time task had a point death penalty time, the priority of tasks trusted the death penalty point time. The less absolutely the point time was, the upper the priority of task was. Otherwise, the additional absolutely the point time was, the lower the priority of task was. The priority of tasks could also be required to regulate consequently once a replacement task was prepared. Therefore EDF formula was dynamic priority programming mechanism compared to straightforward priority programming mechanism.

The class-conscious programming Framework used for FreeRTOS (HSF) [20] may be a promising technique for integration advanced time period elements on one processor. It provides associate degree mechanism to supply temporal partitioning among elements and supports freelance development and analysis of time period systems. In HSF, the CPU is divided into variety of subsystems. Every scheme contains a group of tasks which generally would implement associate degree application or a group of elements. Every task is mapped to a scheme that contains area hardware to schedule the inner tasks of the scheme. Every scheme will use a unique programming policy, and is regular by a worldwide hardware.

Behnam et al. [21] gift associate degree implementation of a 2 level HSF in a very business package VxWorks with the stress on not modifying the underlying kernel. The implementation supports each independent agency and EDF at each world and native level of programming and a one-shot timer is employed to trigger schedulers. R.Rathna et al. projected associate degree algorithmic program to conserve energy victimization TDMA (Time Division Multiple Access) primarily based sleep/wake-up programming by reducing the quantity of times; a node has got to rouse, throughout a slot, to be in active mode [22].

M.Saleh et al. mentioned the important time programming algorithmic program with optimization of international intelligence agency in step with network dynamics. The time period programming uses the differentiated the earliest point in time point intime hardware (EDF) [23].

M.Tan et al. given a message transmission model within which switch and also the end-nodes management the time period with The Earliest point in time 1st (EDF) programming [24].

This paper [25] has projected associate degree electronic communication programming algorithmic program for IoT surroundings. Here IoT subgroups area unit having objects/sensors in it, and master node operating as a broker is arbitrarily designated. Hardware is running at the broker that is choosing the messages sent from the incoming stream of messages from all the members of its subgroup.

Following table highlights some of the reviewed scheduling strategies.

Table 1. Various Methods used in scheduling.

Method	**Parameters**
Preemptive Scheduler	Power Saving
Priority based soft real-time scheduling strategy	Throughput
Preemptive scheduling with Tickless scheduler	Power Saving
Earliest deadline first Scheduling	Improved message transmission model
Table-driven scheduling and priority	Throughput

ISSN: 2278-4632
Vol-13, Issue-10, No.03, October: 2023

scheduling	
Priority-based preemptive scheduling approach	Turnaround Time.
Adaptive Double Ring Scheduling.	Power Consumption
Differentiated earliest deadline first scheduler	security levels

III. Analysis of Research gap

An IoT device is supported by a small operating system specifically designed to support a variety of applications and operational requirements of an IoT device. IoT OS is also responsible for managing limited IoT device resources in real time and efficiently. IoT Operating system has several key issues such as architecture, scheduling capabilities, programming model, and memory footprint and power management. A lot of work already has done on the above mentioned key issues.

Decision making for scheduling in operating system is a trending research area. The selection of scheduling algorithm depends on various parameters which should match real time constraints, way of communication and priority of task. Designing an Optimal scheduling algorithm should maximize the CPU utilization and throughputand minimize the latency and power consumption.

As the throughput of the device totally depends on the working of operating system hence it is very necessary to have efficient algorithm.

There are many existing scheduling algorithm which provide quality of service to the end devices. But these algorithms address some specific parameter. Some algorithms are good in terms of latency and some are good in response time. But for best results it needs mechanism which gives the efficient output with every parameter.

IV.Conclusion

IoT devices are widely used and also have impacted our lives providing assistance to many applications such as transportation, health services, agriculture, gadgets, manufacturing and many more. General purpose operating systems are made for gen- eral use while IoT operating systems are basically designed for dedicated purpose. IoT OS has its components such as Memory Footprint, Power Management and proc- essing capabilities. This paper elaborated and discussed resources of the general OS, different IoT-OS resource management to analyze the selected types of IoT-OS. This paper also analyses scheduling mechanism of IoT OS with its properties, limitations and advantages. The study of IoT OS is useful in designing a new model as per advanced requirements.

V.References

1. N. Windpassinger, “Internet of Things - the complete online guide to the IoT,” i- SCOOP. [Online]. Available: https://www.i-scoop.eu/internet of-things-guide/. [Accessed: 02-Apr-2017].
2. M. Weiser, the computer for the 21st century, ACM SIGMOBILE Comput. Commun. Rev., vol.3, no. 3, pp. 3–11, Jul. 1999.
3. A. Gubbi, R. Buyya, “Internet of Things (IoT): A vision, architectural elements, and future directions,” Future Generat. Comput. Syst., vol. 29, no. 7, pp. 1645– 1660, Sep. 2013.
4. Challouf Sabri, Kriaa Lobna, Saidane Leila Azzouz, “Comparison of IoT con- strained devices operating systems: A Survey” IEEE/ACS 14th International Conference on Computer Systems and Applications, 2017.
5. O. Hahm, E. Baccelli, H. Petersen, and N. Tsiftes, “Operating systems for low- end devices in the Internet of Things: A survey,” IEEE Internet Things J., vol. 3,no. 5, pp. 720–734, Oct. 2016.
6. E. Upton and G. Halfacree, Meet the Raspberry Pi. John Wiley & Sons, 2012.
7. J. Hill and D. Culler, “A wireless embedded sensor architecture for system-level optimization” UC Berkeley, Berkeley, CA, USA, Tech.Rep., 2002.[Online].

ISSN: 2278-4632
Vol-13, Issue-10, No.03, October: 2023

8. Lytras, M.D.; Raghavan, V.; Damiani, E, "Big data and data analytics research: From metaphors to values pace for collective wisdom in human decision making and smart machines." Int. J. Semant. Web Inf. Syst. 2017, 13, 1–10. [CrossRef].
9. Chen, C.Y.; Hasan, M.; Mohan, S. Securing real-time internet-of-things. Sensors 2018, 18, 4356. [CrossRef] [PubMed].
10. Sehrish Malik , Shabir Ahmad, Israr Ullah, Dong Hwan Park and DoHyeun Kim ,"An Adaptive Emergency First Intelligent Scheduling Algorithm for Efficient Task Management and Scheduling in Hybrid of Hard Real-Time and Soft Real- Time Embedded IoT Systems" www.mdpi.com/journal/sustainability 2019.
11. R. Lajara, "Power consumption analysis of operating systems for wireless sensor networks" Sensors, vol. 10, no.6, pp. 5809–5826, Jun. 2010.
12. C. L. Liu and J. Layland, "Scheduling algorithms for multi programming in a hard real-time environment," J. Amer. Compt. Mach., vol. 20, no. 1, pp. 4M1, 1973.
13. G. D. Carlow, "Architecture of the Space Shuttle primary avionics software system," Commun. ACM, vol. 27, no. 9, Sept. 1984.
14. E. L. Lawler and C. U. Martel, "Scheduling periodically occurring tasks on multiple processors," Information Processing Lett., vol. 12, no. 1, Feb. 1981.
15. H. Dai, M. Neufeld, and R. Han. "ELF: An Efficient Log-Structured Flash File System for Micro Sensor Nodes". In Proceedings of the 2nd international confer- ence on Embedded networked sensor systems - SenSys 04, page 176, New York, New York, USA, Nov. 2004. ACM Press.
16. E.Baccelli, O.Hahm, M.G¨unes, M.W¨ahlisch, and T.C.Schmidt, "RIOT OS: Towards an OS for the Internet of Things," in Proc. of the 32nd IEEE Infocom. Poster. Piscataway, NJ, USA: IEEE Press, 2013.
17. Yan Zhao, Qianping Wang, Wei Wang, Dong Jiang, Yiwen LIU , "Research on the Priority-based Soft Real-time Task Scheduling in TinyOS" in International Conference on Information Technology and Computer Science, 2009.
18. Lin Kai, Zhao Hai, YIN Zhen-yu, BI Yuan-guo, "An adaptive double ring scheduling strategy based on TinyOS," Journal of Northeastern University(Natural Science), 7th ed., vol. 28., 2007, pp. 985-988.
19. Min Yu, SiJi Xiahou, XinYu LI, A Survey of Studying on Task Scheduling Mechanism for TinyOS, IEEE 2008.
20. Z. Deng and J. W.-S. Liu. "Scheduling real-time applications in an open environment. InProc.18th IEEE Real-Time Systems Symposium (RTSS)", 1997.
21. Moris Behnam, Thomas Nolte, Insik Shin, Mikael Asberg, and Reinder J. Bril. Towards hierarchical scheduling on top of vxworks. In Proceedings of the Fourth International Workshop on Operating Systems Platforms for Embedded Real- Time Applications (OSPERT"08), pages 63–72, July 2008.
22. Rathna. R And Sivasubramanian. "Improving energy efficiency in wireless Sensor networks through scheduling and Routing"; International Journal of Advanced Smart Sensor Network Systems (IJASSN), Vol 2, No.1, January 2012.
23. Saleh, Maen and LiangDong. "Real-time scheduling with security awareness for packet switched networks." In Radio and Wireless Symposium (RWS), 2012 IEEE, pp. 391-394. IEEE, 2012.
24. Tan, Ming, and Zhen Wei. "Schedulability analysis for real-time messages over switched Ethernet with EDF scheduling." In Information Science and Engineering (ICISE), 2010 2nd International Conference on, pp. 2362-2366. IEEE, 2010.
25. Saima Abdullah, Kun Yang, "An Energy-efficient Message Scheduling Algorithm in Internet of Things Environment", IEEE 2013.