Oğuzhan Kürşat Karayılan
150140011

Comparison of RTOS and GPOS

Outline

Similarities:

- Multitasking
- Resource Management
- Hardware Abstraction

Differences:

- Latency
- Determinism
- Task Priority
- Scheduling
- Hardware
- Mathematical Equation
- Memory Model

Operating systems are primary kind of software on a machine that interface with various hardware devices. The main missions of an operating system are scheduling tasks, controlling peripheral, provide services to programs, and allocate resources among applications. Operating systems are commonly used in wide variety of purposes on our daily life such as when playing games on a game console or when withdrawing money from an ATM machine. Every operating system is designed for different purpose. While we use general purpose operating systems on our personal computers to check emails, play games or watch video on the internet, there are also operating systems that are called real time operating systems which are designed to perform more specific tasks such as ensuring to open airbags on right time. Even though there are a number of differences between real time operating systems(RTOS) and general purpose operating systems(GPOS), there are striking similarities as well.

GPOS and RTOS both are kind of operating systems which performs in a similar way and have similar goal that is providing ground for tasks to run. Each task on a system under control of the operating system. GPOS and RTOS both do some level of multitasking when running tasks. On a classic processor, a single task can be executed at a time. However, a multitasking operating system can make it look as if each task is executing simultaneously by quickly shifting between tasks.

One of tasks of operating system is resource management. It manages resource allocations to run system correctly. Since we have components which comes with limited availability, operating system needs to use them efficiently by distributing resources between processes. GPOS and RTOS does resource management to ensure that every process get what it needs in order to run correctly.

Providing an abstract layer between software and hardware is another similarity between GPOS and RTOS. Without hardware abstraction, a program should have information about how certain hardware device works and communicates with each other. This does not make just making program for an operating system hard but also makes a program more complex. By providing hardware abstraction, the program does not directly communicate with the hardware. It sends calls to the operating system to communicate with device. This makes everything easier and less complex.

Above notions are common between GPOS and RTOS since they are both an operating system. But they also differ according to their purpose of work. GPOS are used in our personal computers to perform daily tasks and run variety of programs. On the other hand, RTOS are used on car control systems, nuclear facilities and many medical related fields. RTOS need to respond in a certain time called deadline when they performing tasks and missing a deadline can cause affects ranging from undesired to catastrophic.

As stated above, the prime difference between GPOS and RTOS is importance of a task's respond time. When working on RTOS, result of the time delay can be a lot costly or hazardous than GPOS. While waiting five minutes to load a program on personal computer that uses GPOS does not lead issues, on a real time system even seconds are very important. RTOS are specially designed to run programs with very low latency and a high degree of reliability. Particularly this is significant in measurement and automation systems where downtime is costly or a program delay could cause a safety hazard. To count an operating system as real time, maximum time for each process take to finish should be known beforehand. RTOS can guarantee that a program will run with very consistent timing whereas GPOS do not guarantee that a task will end on certain time.

Since there is deadline for RTOS's tasks it can be said that RTOS are deterministic naturally. An application that runs on a real-time operating system is called as deterministic if its timing can be guaranteed within a certain margin of error. An RTOS differs in that it typically provides a hard real time response, providing a fast, highly deterministic reaction to external events. On the other hand, GPOS are non-deterministic which is do not have the ability to complete all tasks in a pre-determined time window. By contrast, the real-time operating system (RTOS) executes commands within a pre-determined time window.

In the case of a GPOS, task scheduling is not based on priority always. GPOS are designed to handle scheduling in such a way that it manages to achieve high output. In order to maximize the total number of processes that complete their execution per unit time GPOS schedule their tasks differently. For example, to produce high number of output 3 or 4 low priority based tasks can be run rather than high priority task. High throughput is achieved by serving 5 low priority tasks than by serving a single high priority one. Where as in an RTOS, scheduling is always priority based. Most RTOS uses preemptive task scheduling method based on priority levels. An RTOS must also include a system of priority inheritance, predictably sustain thread synchronization, and have a mechanism to avoid priority inversion. Priority inversion is a situation where a low priority thread uses a resource and delays the execution of a high priority thread when both are competing for the same resources. Additionally, one of the primary provisions of an RTOS is that interrupt latency is predictable.

The real time kernel runs on preemptive scheduling rules whereas GPOS follow non preemptive scheduling technique. In preemptive scheduling the resources are allocated to the process for a limited amount of time. When a process uses it's determined time on processor, it is placed back to the ready process queue. The process stays in ready queue till it gets next chance to execute. If a process with high priority arrives in the ready queue, it does not have to wait for the current process to complete its time. Instead, the current process is interrupted in the middle of execution and is placed in the ready queue till the process with high priority is utilizing the CPU cycles. However, in non-preemptive scheduling, once the resources are allocated to a process, the process holds the CPU till it gets terminated or it reaches a waiting state. Unlike preemptive scheduling, non-preemptive scheduling does not interrupt a process running CPU in middle of the execution. Instead, it waits for the process to complete its CPU burst time and then it can allocate the CPU to another process.

An RTOS is usually designed for a low end, stand alone device like an ATM, Vending machines, Kiosks etc. RTOS is light weight and small in size compared to a GPOS. A GPOS is made for high end, general purpose systems like a personal computer, a work station, a server system etc. The basic difference between a low end system and high end system is in it's hardware configuration. Nowadays a personal computer or even a smart phone comes with high speed processors (in the range of many Gigahertz), large RAM's (in the range 2 or 3 GB's and even higher) etc. But an embedded system works on low hardware configurations usually, speed in the range of Megahertz and RAM in the range of Megabytes. A GPOS being too heavy demands very high end hardware configurations. It is economical to port an RTOS to an embedded system of limited expectations and functionalities (Example: An ATM is supposed to do only certain functions like Money transfer, Withdrawal, Balance check etc.). So it is more logical to use an RTOS inside the ATM with its limited hardware. It is not economical to improve the hardware of an ATM just to port a GPOS as it's user interface.

Another difference between RTOS and GPOS is that for a real time OS mathematical equation can be defined. In contrast, the mathematical relation can not be defined for GPOS. Because of the deterministic nature of RTOS, we can gather a mathematical equation about how the system works and performs scheduling. For example, rate monotonic algorithm is a mathematical technique to select the relative priorities of tasks, and it is used in conjunction with an RTOS or some kind of fixed priority preemptive scheduler.

GPOS can use virtual memory concept. Also, it works on protected memory model. Memory protection is a way to control memory access rights on a computer, and is a part of most modern instruction set architectures and operating systems. The main purpose of memory protection is to prevent a process from accessing memory that has not been allocated to it. However, RTOS use flat memory model. Flat memory model or linear memory model refers to a memory addressing paradigm in which memory appears to the program as a single contiguous address space. The CPU can directly (and linearly) address all of the available memory locations without having to resort to any sort of memory segmentation or paging schemes.

To sum up, there are both similar and different aspects of RTOS and GPOS. They are both an operation system but used at fairly different fields. As the name indicates GPOS are designed for general purpose usages. However, RTOS are designed for more specific tasks and response time is playing important role.