# A Report on Real Time Operating Systems

# Yunus Güngör

Student No: 150150701

This report written in context of BLG450E Real Time System Software class by Deniz Turgay Altılar in Istanbul Technical University. Real time operating systems are operating systems designed for delivering fully or partly predictive schedule for given tasks. Since all real time operating systems also an operating system, those systems also have tasks like, resource distribution, communication, I/O management. But most important aspect of real time operating systems is the consistency, predictability and timing constraints. However recent evolution of real time operating systems, created a shift from completely predictable applications to soft real time applications with the concept called quality of service (QoS) which often used in multimedia applications and complex distributed real time systems (Stankovic, Rajkumar, 2004).

Real time applications demand very difficult constrains such as on time delivery, minimum task shift time, robust and error free programs, stability and so on. Real time operating systems must handle with these demands. To handle with all constrains, these operating systems has several features. The IEEE Portable Operating System Interface for Computer Environments, POSIX 1003.1b, defines a feature list for a basic real time operating system:

* Asynchronous I/O
* Synchronous I/O
* Memory Locking
* Semaphores
* Shared Memory
* Execution Scheduling
* Timers
* Interprocess communication
* Real Time Files
* Real Time Threads

Most of defined features by IEEE exists in modern operating systems except real time files and real time threads. Those two features separate real time operating systems from other types of operating systems. Other features than real time files and real time threads is out of scope for this report.

Real time files is the ability to process a filesystem with deterministic speeds. Unlike common operating systems process must be deterministic.

Real time threads is the ability to create tasks that belongs to a real time application with specifications like schedule constrains.

Other features might be included in a real time operating system are:

* Low context switch times
* Preemptive
* Deterministic synchronization
* Priority levels
* Predefined runtimes for API’s in OS
* Maximum consumption which is consuming all resources to produce results in time
* Small size for embedded systems
* Scalability for distributed communication or distributed computing systems

Those features applied and maintained during design and development of the operating system. Every code written for a real time operating system must be written with those constraints in mind. Depending on the application some of constrains required by real time application can be loose than others. For example controlling an airplane has more strict deadlines that controlling an IoT network, but controlling an IoT network requires scalability more than controlling an airplane. Therefore, real time operating systems can be developed application specific to lower costs.

Main difference between a real time operating system and a modern operating system is to ability to provide determinism and stability. Real time operating systems must predefine API run times for usability in real time applications. However modern operating systems does not have that type of requirement. Modern OS can just pause a process and let it continue whenever, in this case until API call completed. However, a real time OS must consider deadline while running the API call. Another difference between a real time OS and modern OS is the context switch time. Real time systems must provide shorter context switch time to run scheduling algorithms as accurate as possible. Depending on application real time operating systems might be required to run stable for a very long time. This requirement can be accomplished with error free code (memory leaks and other performance related issues) and basic structure. However modern operating system does not have that type constrain, because average user usually reboots or turn of their operating systems daily. However some server grade operating systems expected to run for long periods.

According to application field real time operating systems has different features and different specifications. RTOSs can be categorized by kernel type, by footprint and by real time constraints applied.

A real time OS can use different architectures to achieve its goals. Some OS structures used in RTOSs include, small and fast kernels, real time extensions to widely used operating systems, component based kernels, research kernels. Small and fast kernel mostly used in embedded systems because they have very small footprint and consume less resources (for example energy). Real time extensions to widely used operating systems can be useful for developing quick, portable and familiar software, but this type of systems generally became less predictable. Component based kernels developed component by component and compiled according to needs of the application. And research kernels developed simply to show proof or solve other problems in real time systems. Research kernels might not include widely used specifications or APIs.

Another aspect of RTOSs is footprint. Since most embedded systems are part of or it is a real time system, using a real time operating system is very useful. However due to nature of embedded systems like IoT systems or sensor networks, there is limited resources operating system can use. Footprint is the combination of resources used. This resources include, size operating system needs, memory, CPU usage and energy consumption. Real time operating systems with small footprints used in embedded systems and have small or lite kernels that designed to perform specific set of tasks rather than providing APIs and features. RTOSs with big footprint allows more functionality but consumes much more resources. This types of RTOS used for bigger and complex systems such as air traffic control system.

Depending on application real time systems might have different time constraints. Real time operating systems has three different types by time constraints: hard real time operating systems, soft real time operating systems and mixed operating systems. Hard real time systems requires that a task to be completed before deadline. After deadline completing that task has no value. Soft real time systems allows a task to be completed after deadline. But completing that task after deadline has less value than completing before deadline. Later a task done, less value it has. Mixed real time systems combines both hard real time and soft real time tasks.

At the time writing that report there is 186 real time operating systems in Wikipedia page of “Comparison of real-time operating systems” which is derived form DMOZ (*directory.mozilla.org)* which is an open content directory and still exists on *https://dmoztools.net.* 136 of those RTOSs are still active. Since its impossible to mention all RTOSs in this report only most significant ones will be mentioned. According to Hambarde, Varma and Jha most significant RTOSs chosen from different categories are VxWorks which is most compatible, Windows CE which is commercially successful, QNX Neutrino which is commonly used for multiple node systems and RTAI which is very popularly used. VxWorks used in many of NASA’s projects, Windows CE used in many end user devices, QNX Neutrino used in many different cars and mobile phones, RTAI is a real time extension for linux which is used for controlling robots and other complicated systems (2014).

There is different application fields for RTOSs. Most of the communication based systems make use of RTOSs. Even though they are not hard real time systems, they make use of real time operating systems to benefit from time constraints. Using time constraints and a concept called quality of service (QoS) communication systems developed to deliver package as fast as possible. However, there is other hard real time systems like, air traffic control, power plant management, automated manufacturing management, sensor networks, radars, Industrial process control systems (controlling temperature, pressure and other variables to ensure everything is stable) and energy distribution systems.

Real time operating systems are a key essential to create a stable and predictable environment for developing real time applications. This type of operating systems performs standard operating system’s tasks but also provide scheduling with time constraints and predictable run times. These types of operating systems has different features and types depending on application they are used, and they are used in time sensitive systems, complicated systems and in communication systems.

Resources:

“Real-Time Operating Systems”, John A. Stankovic, R. Rajkumar, 2004, Real-Time Sytems, 28, pp. 237-253, doi: 10.1023/B:TIME.0000045319.20260.73

“An Overview of Real-Time Operating Systems”, Walter Ceden, Phillip A. Laplante, 2007, JALA: Journal of the Association for Laboratory Automation Vol 12, Issue 1, pp. 40 - 45 , doi: 10.1016/j.jala.2006.10.016

"The Survey of Real Time Operating System: RTOS", P. Hambarde, R. Varma and S. Jha, 2014 International Conference on Electronic Systems, Signal Processing and Computing Technologies, Nagpur, 2014, pp. 34-39. doi: 10.1109/ICESC.2014.15