JBOSN RTOS

LIFE WITH JBOSN RTOS

"Ensuring Embedded Real-Time System Success"

Create a new value through
Embedded Real-Time Operating System

jBOSN RTOS

General Description

The JBOSN Operating System is a multi-layered, modular, high-performance real-time operating system designed specifically for embedded microprocessors. It provides a complete multitasking environment. It is built around the JBOSN real-time multi-tasking NANO-KERNEL and a collection of companion software components and service servers. Every servers implement a logical collection of system services. To the application developer, the system service requests appear as re-entrant C functions callable from an application. Any combination of server and library components can be incorporated into a system to match your real-time design requirements.

Design Principles

- **Kernel**
  1. Multi-Layer kernel structure : modularity, portablibity, scalability
  2. Multi-Tasking/Thread and flexible IPC
  3. Precise timer and Priority based real-time scheduler
  4. Scalable hard-real-time
- **I/O manager**
  1. Constant device management and efficient I/O system
- **Resource Management**
  1. Cost efficient use of memory
  2. Reliable and Robust system services
- **User**
  1. Easy to use
  2. Low latency

Advantages

- **Modularity**

Nano-kernel	Minimum real-time operating system and library(8KB)
Micro-kernel	Principal RTOS servers (time server, system server, sync server)
Macro-kernel	Expanded RTOS servers (device server, filesystem server, window server, network server)
Library/Driver/HAL	Consisted of the standard library, middle-ware, device driver and HAL
Applications	Applications by user

- **Scalability**

Server concept	Main functions are designed by server concept and added to JBOSN RTOS. User-required server can be developed by server concept.
User developed library	The in-house library can be added and applied to server-development
Mutual exclusion	The resources is mutually exclusive between servers, then the expansion of required function is very easy

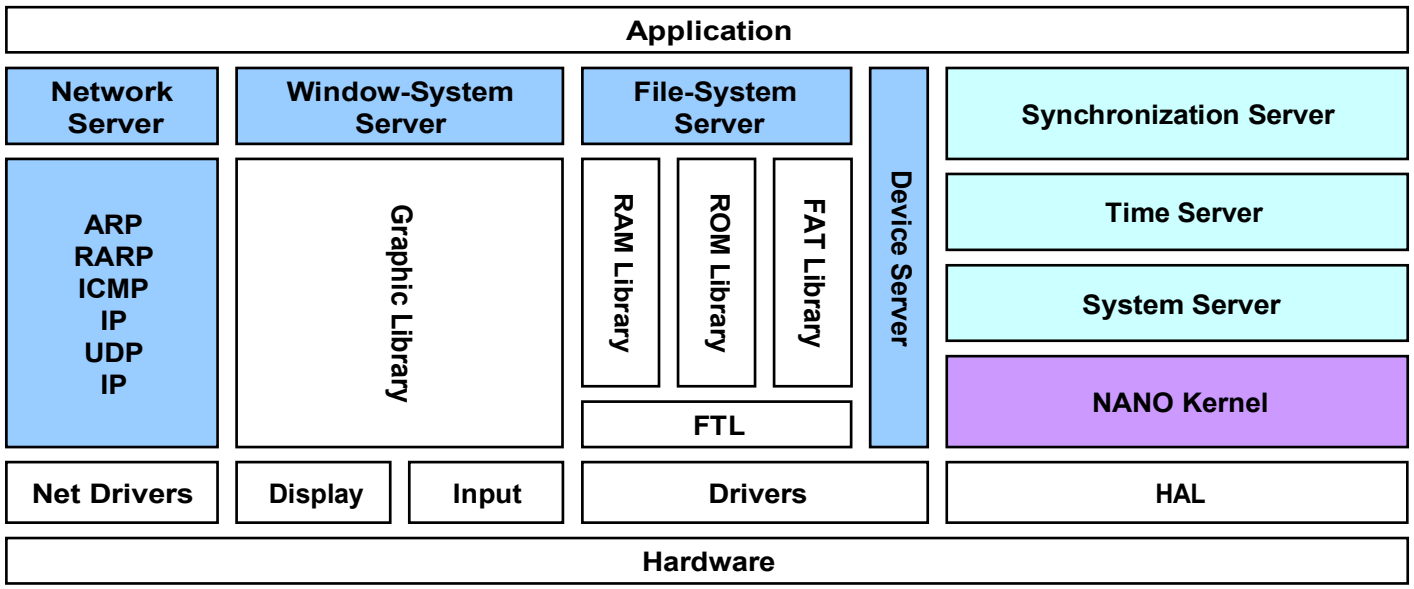
- **Stability**

Independency	The independency between modules increase the stability and make easy to debug
modularity	The module are seperated in phsyical area
Error propagation	The problem of one module can not propagate to the others

- **Memory**

Small memroy requirement	The memory requirement of modules are very small. For example NANO-KERNEL size is 8KB
Efficient relocation	All modules can be generated by seperated binary images. The relocation is very easy and efficient

Block Diagram



jbosn@jbosn.com

www.jbosn.com

Tel. 02-2025-2955
Fax. 02-2025-2957

iBOSN SYSTEMS