



ZKA Technologies  
Extended Mobile specification.  
07/03/2024 Rev A.  
Amlal El Mahrouss

## Table of contents:

1. The Zeta Kernel Architecture (NewOS)
  - a. Microkernel.
  - b. Loader.
  - c. Why this matters.
2. The New OS's Core\* APIs (CoreFoundation, CoreGraphics...)
  - a. CoreFoundation.
  - b. CoreGraphics.
  - c. CoreMT. (Multi touch support or emulation)
  - d. CoreWidgets. (UI/Java loader using XML)
  - e. MultiTouch drivers, or the need of emulating a multi-touch.
  - f. The 'Hello' protocol, a way of sending data like Bluetooth.
3. The New OS System Call Interface and DDK.
  - a. The DDK.
  - b. The SCI.
  - c. How those relates.
4. The New FS file system internals and its compression.
  - a. Forks and catalogs.
  - b. EPM support.
  - c. Long filenames.
  - d. File system versioning.
5. The New OS compression engine.
  - a. The Compression Kit.
  - b. A practical example: IPCEP.
6. The CoreBoot's Explicit Partition Map.
  - a. EPM's BootBlock data structure.
  - b. Globally Unique Identifier.
7. The Core Boot firmware (PC only).
  - a. Explicit Partition Map
  - b. Core Boot LX header.
  - c. EEFI (EPM EFI) open source project.
8. The Zeta Hardware Platform.
  - a. MBCI.
  - b. Base Processor (or Bridge Processor)
  - c. Central Processor (or Main CPU which runs the OS)
  - d. USB-PD (Power Delivery)
  - e. ARMv8.
  - f. Double Data Rate RAM.
9. The Zeta Developer SDK.
  - a. cplusplus.exe
  - b. armasm.exe
  - c. link.exe
  - d. The Application Package (.pkg) and PEF format.

# I - The ZKA Kernel Architecture (ZKA)

ZKA's architecture is microkernel based, because of the target product being an embedded device, there is a need to embed the kernel and bootloader as one. Although two things are done:

- They are kept separated, for obvious reasons.
- A protocol (Handover) is established, to communicate between the bootloader and kernel.

This originates because newOS also targets personal computers. The support for mobile phones is quite recent (ARM64 HAL)

## 1. MicroKernel:

It (newoskrnl.dll) consists of a preemptive multithreaded scheduler, MP manager, file manager, device manager, drive manager, networking, System Call Interface, Driver Call Interface, Dynamic symbols and IPC. This is a little bit more than a MK, but due to practical reasons (context switching and power consumption) We have to move some code to hypervisor/ring-0 mode.

Everything that touches the hardware is abstracted in the HAL. which is a layer within the kernel used to communicate with HW (ports, DMA or BP to MCPU configurations).

## 2. Loader:

It (newosldr.exe) Takes care of loading a pre embed or newer kernel to the device. The first option is done on mobile platforms, to avoid too much memory from being used. newosldr supports QR code error generation, something needed in order to provide excellent 'production error handling'

## 3. Why this matters:

- A microkernel makes the design simpler and more understandable (although an effort is still needed.)
- The loader is here to load patched versions of the kernel. To provide much simpler and safer OTA updates.
- A simple system means a complex architecture, as everything that ranges from multitouch for example requires a framework -> Driver -> KCI -> Kernel -> HW (simplified stack)
- There is also a possibility for drivers to add syscalls to the system. (Called 'hooks')

- ZKA was made with workstation computing in mind, so you get a quality OS within a phone/tablet.