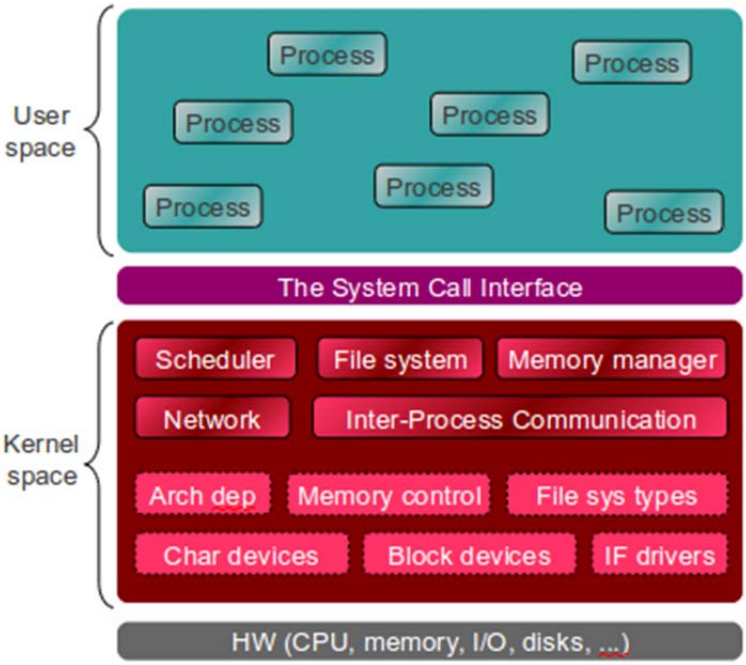
Linux system overview

By Anders Törnqvist

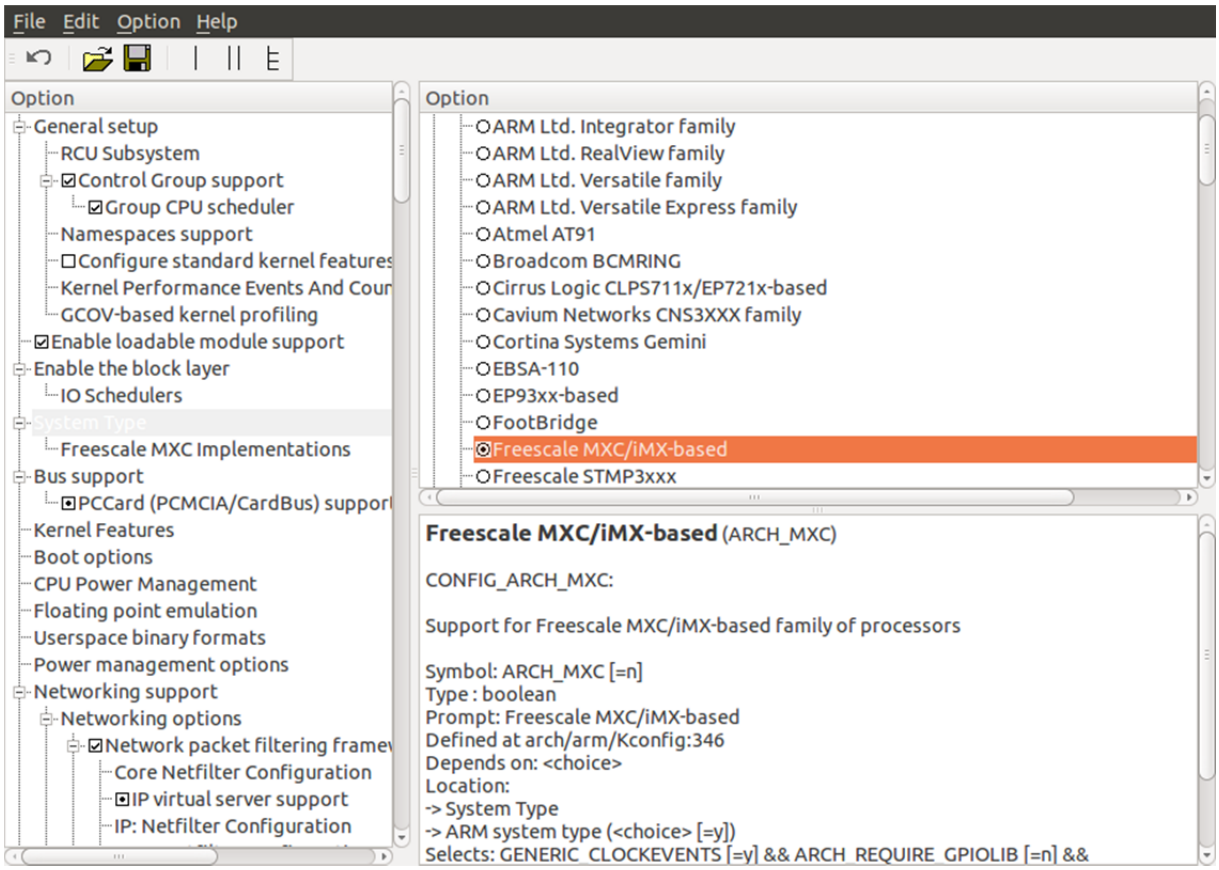
A Linux system comes in many flavors depending on where you get it from. It can be pre-installed in your embedded system, downloadable as some kind of development kit, bought on a DVD for installation on a PC or just simply as plain source code.

Linux systems are extremely configurable both at build time and at runtime. This means that, in detail, two separate systems can differ a lot. In general though all Linux systems are very alike. The similarity is based on the overall structure and architecture that all Linux systems share. This is developed over long time and ensures the stability and robustness of Linux.



Most of the source code that Linux is based on is very hardware platform independent. Mainly smaller parts of the Linux kernel contains code that is specific for various hardware. Even the kernel itself contains mostly hardware independent code which means that the code can be used directly on any hardware such as in a large file server or your tiny embedded system.

The difference is how the source code is configured when it is selected and built into a runnable system. With a specific configuration tool you select only the parts that you need for your system. This configuration is done by the developer or integrator of the system. Users do not do this.



A basic part of the structure in Linux systems is the separation of the execution environment into two areas - kernel space and user space.

Kernel space runs with higher priority and contains the main system functionality like scheduling, memory management, networking, file-systems and device drivers. All hardware resources are controlled by kernel space. Management and decisions about the resources are made by different kernel subsystems.

User space is the part of the system where all the user processes (programs) run. The execution of all these processes are controlled by the scheduler in kernel space to gain a fair amount of time and performance.

All user space processes are totally isolated from each other and kernel space by a well defined interface. No process can get access to resources such as memory or files owned by other processes. Every process gets its own virtual memory space where approved resources are mapped in by the kernel memory manager. User space processes have to request access to resources.

Processes also executes associated to specific user and group privileges defined in the systems.

Based on this structure and functionality it is possible to also used shared resources between parts of the system, such as between specific processes, in a controlled way without jeopardizing the stability of the entire system.

The separation and isolation of processes are basic reasons to Linux robustness and stability. Many years of development of core kernel functionality also ensures for high performance and throughput in a Linux system.

Contact Anders at

Anders.Tornqvist@fosscentral.se
http://www.fosscentral.se