UnixAOS Implementation Notes

G. Feldmann 24.11.2011

UnixAos is a 32-bit X-Windows application which emulates the Aos/A2 system in the Solaris, Linux and Darwin (MacOS X) environments.

The most important difference to native Aos is the missing support for sound in UnixAos.

Furthermore UnixAos has no explicit support for hardware interfaces like RS 232 or USB. They aren't really needed as all devices are accessible through the Unix filesystem. To access the serial line interface just open it as an old file:

```
f := Files.Old( “/dev/ttyS1” )
```

Port specific modules:

Most of the system sources are the original native A2 sources. Only some low level modules got replaced by port specific versions.

a) Statically linked modules in {Solaris,Linux,Darwin}AosCore:

Unix.Glue.Mod

Interface to the dynamic linker of Unix and console output

{Solaris,Linux,Darwin}.Unix.Mod

Interface to the Unix system calls and host system specific data structures and constants.

Unix.I386.Machine.Mod

Interface to the configuration parameters (AOSCONFIG)
Spin locks based on Unix mutexes,
Dynamically extending and shrinking of the Aos heap,
Garbage collection loop.

Unix.Heaps.Mod

Unix.Objects.Mod

Creation and synchronisation of active objects based on Unix threads

Unix.Kernel.Mod

Port specific invocation of the garbage collector

Unix.Traps.Mod

Unix.UnixFiles.Mod

Filesystem plugin for the Unix filesystem

Unix.BootConsole.Mod

Starts the Aos system or the procedure specified by the '-x' Argument in

aos -x M.P

and thereafter starts the garbage collection loop in module Machine.

b) Dynamically linked low level modules:

Unix.Clock.Mod

Display output, keyboard and mouse input:

Unix.X11.Mod, Unix.X11Api.Mod

Interface to the X11 library

Unix.Beep.Mod

system beep -> X11.Bell

Unix.XDisplay.Mod

Dispaly plugin for the X11 display

Unix.KbdMouse.Mod

Replacement for the keyboard and mouse drivers based on X11

Network:

Unix.IP.Mod

Port specific IP module without the interfaces layer of A2.
The object 'Interface' has no mission in UnixAos but is included (only partially) as applications exist which obtain the local IP address from it.

Unix.Sockets.Mod

Oberon interface to the Unix sockets

Unix.UDP.Mod, Unix.TCP.Mod

UDP and TCP based on Unix.Sockets.Mod

Unix.DNS.Mod

Unix.ProcessInfo0.Mod

Unix.DisplayRefresher.Mod

Handles the exposure events of the X11 display

Unix.BootLinker.Mod

Links the core modules statically

c) Loader for the statically linked core modules:

aos.c, Threads.h, {Solaris,Linux,Darwin}Threads.c

Get compiled and linked into **aos.solaris, aos.linux** or **aos.darwin** and contain:

- the loader for the prelinked core modules
- a unified interface for the the POSIX threads (Linux, Darwin) and Solaris threads (Solaris)
- 32-bit C-wrappers for the system calls for memory allocation. These wrappers are needed for running UnixAos in 64-bit Linux systems.
- signal handling interface for the Aos trap

Filename specification

If in UnixAos a file gets opened for reading without path specification, the system looks for it in the directories contained in the environment variable AOSPATH. The current working directory '.' should always be the first entry.

If a file gets opened for writing without path specification, it will be created in the current working directory.

Files which get opened with path specification follow the Unix rules for absolute or relative pathnames.

The current working directory (the directory in which Aos gets started) must grant write permission to the user!

Configuration

UnixAos gets configured in the start script 'aos' via the environment variables AOSPATH and AOSCONFIG.

The most impotant parameters in AOSCONFIG are the 'DisplaySize' (default is the whole X11 display) and the 'StackSize' for the active objects. The 'StackSize' prameter does not effect the main thread which has always a large stack needed for the garbage collector.

Darwin port specific notes

The Darwin OS, like Solaris and Linux, uses the SYSV ABI but with some additional constraints concerning the stack alignments. These constaints are partially implemented for extenal C-procedures in the Fox compiler and get activated by the option '--darwinHost'. As the Fox implementation is not comprehensive the following restrictions apply to external C-procedures:

- do not use them without parameters. Because of this some parameter less procedures of the threads library got a wrapper with a dummy parameter.
  e.g:

  ```
  int pthreadThis();
  ```

  gets called by the following C-wrapper

  ```
  int thrThis(int dummy);
  ```

- if they are functions use them only in assignments. The following instruction

  ```
  IF thrThis(0) = y THEN ...
  ```

  will cause a segmentation violation error at runtime and should be replaced by

  ```
  x := thrThis(0);
  IF x = y THEN ...
  ```