FreeMiNT Operating System

éidhne kennedy

December 2021

1 Introduction

FreeMiNT is an alternative operating system kernel for the Atari ST system, written in C, released first by Eric Smith in 1990, under the name MiNT (*MiNT is Not TOS*), as a method to port the GNU library and similar utilities to to Atari ST TOS. Atari later hired developer Eric Smith, and MiNT was adopted as an official alternative kernel with the alongside the release of the Atari Falcon. After Atari left the computer market, the development of MiNT was maintained by a team of volunteers, mainly with the aim of emulation on virtual systems like ARAnyM, or on emulators like Hatari. The latest release was version 1.18, in January 2017, and the FreeMiNT Github page is still maintained to this day.

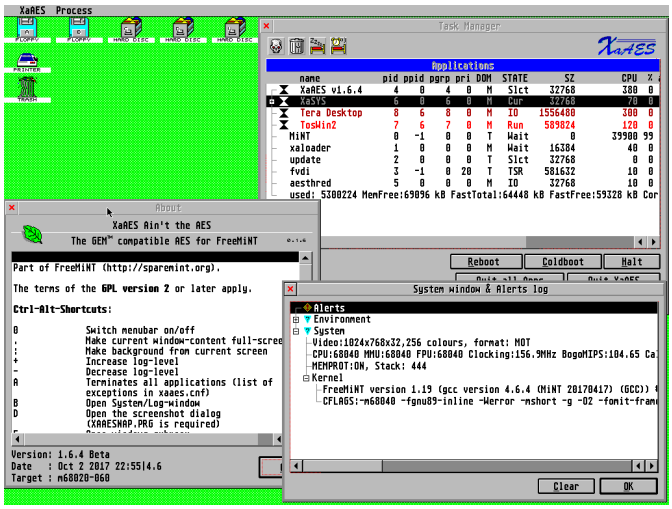

Figure 1: [3]

The Atari TOS (The Operating System) debuted in 1985 on the Atari 520ST, combining elements of the DigitalResearch GEM GUI running on top of GEM-DOS. TOS usually ran in a ROM chip contained in the computer, therefore,

before local hard drives were available in home computers, it was an almost instant-running OS. [6] [2]

TOS consisted of:

- a desktop interface
- GEM - the Graphics Environment Manager
- GEMDOS - GEM Disk Operating System
- a BIOS - Basic Input/Output System
- XBIOS - eXtended BIOS

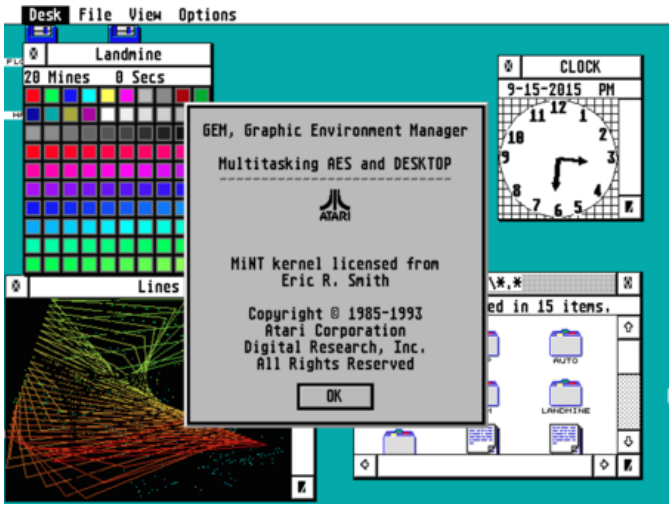

Figure 2: [5]

One major downside of TOS was that it did not directly support multitasking, at first. This is where FreeMiNT comes in.

FreeMiNT provides only a kernel, and while it can use the graphical interface that TOS did, GEM, a better choice would be an AES (Application Environment Service) that can make use of its multitasking facilities. The default one today is XaAES.

FreeMiNT is a monolithic kernel, meaning that the entire operating system is working in kernel space, meaning it defines a divide between the area in virtual memory where the operating system runs, and where other application software runs, known as user space. [7]

2 Process Management

Process management is the creation, scheduling, and management of processes. A process is the act of executing a program, or a static set of instructions. The

kernel manages this by allocating and deallocating the CPU to each competing process, maintaining a record of all of them through its internal process table.

FreeMiNT is a multitasking system, or a timesharing system. This means that the execution of each task is divided up into a fixed time interval, known as the *time quantum*, and the execution of every task is limited to this interval. Each task is divided into segments the same length as the time quantum, and then added to a queue of tasks to be completed. The CPU then executes these tasks in this order, one at a time, until all tasks are completed.

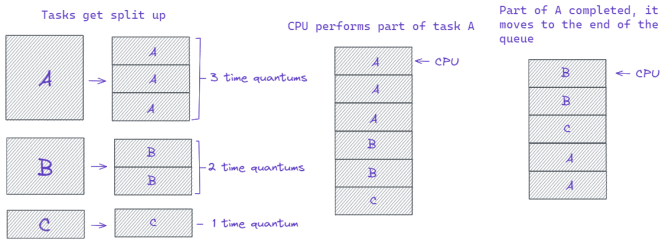

Figure 3: created using excalidraw.com

3 Memory Management

Memory management is function that the OS carries out, which is responsible for managing and allocating the computer's primary memory, and avoiding deadlock - when processes block each other from continuing due to competition over non-shareable resources.
The memory descriptor in TOS is as follows:

```
typedef struct md
{
    struct md *m_link;      /* pointer to next MD         */
    int32_t     m_start;    /* start address of the block */
    int32_t     m_length;   /* length of the block        */
    BASEPAGE  *m_own;       /* pointer to the basepage    */
}
```

[4] The component *m_own* is a pointer to the basepage of the process to which the memory block belongs.

These structs are administered in the internal TOS memory management, and hang in three memory lists, for free blocks of memory, occupied blocks, and for the next block to be occupied. What this attempts to achieve is to that consecutive *Malloc* calls reserve consecutive memory if possible, intended to reduce memory segmentation. A known advantage of this approach is that it is not vulnerable to processes that attempt to interfere with others, as the memory descriptors lie in system memory (TOS being a monolithic kernel) and far away from system memory.

There are significant differences between this and FreeMiNT, as a new feature of FreeMiNT is memory protection, and the concept of protecting a program's allocated memory space from unauthorised "intruders". An intruder in this case is any other program that attempts to access another's memory, either by design or due to a bug. This allows for increased stability in case a program goes berserk. Memory protection works on a per-process basis.

```
/** @struct memregion
 * Description of a memory region.
 */
struct memregion
{
        unsigned long        loc;               ///< Base of memory region.
        unsigned long        len;               ///< Length of memory region.
        long                links;               ///< Number of users of region.
        unsigned short       mflags;                ///< E.g. which map this came from.
        short               reservd;
        MEMREGION *save;        ///< Used to save inactive shadows.
        MEMREGION *shadow;        ///< Ring of shadows or 0.
        MEMREGION *next;        ///< Next region in memory map.
};
```

Taken from /sys/mem.c from the FreeMiNT 1.18 release source code.

4 File Management

File management is the process of creating and maintaining an organised, hierarchical structure for storing information.

As mentioned above, one of Eric Smith's main aims when creating MiNT was to be able to port the GNU library to the Atari ST, and so the unified filesystem that FreeMiNT uses is very similar to the root directory found in all versions of unix. The filesystem is assigned the letter U, similar to windows' C, and all drives are available under this "drive" as directories, for example "u:\k\".

From the FreeMiNT github, in a document written by Eric Smith on the creation of MiNT:

> MiNT provides a fake "disk drive" called U: (for "unified"). There are "directories" in the root of this drive which contain "files" special to MiNT. These "files" are not real files, but may represent other objects such as devices or executing programs. Ordinary file-access calls are used to deal with them; for instance, writing to U:\DEV\CON is like writing to the console (as distinct from stdout).

[1] There are several directories that are defined by default under u:\:

- **proc** - every process has its own file in this directory, and the suffix is its id. deleting a process' entry in u:\proc\will kill the process.

- **kern** - a system compatible to linux' procfs, to ease porting of unix programs.
- **dev** - used to access devices. most device drivers (*.xdd files) are also located in this directory.
- **pipe** - directory where pipes are created. Pipes are special files that are used to communicate between processes. The data in a pipe is always in memory, so using a pipe instead of a temporary file is usually faster; it also doesn't consume disk space.
- **shm** - directory for shared memory.
- **ram** - the internal R.A.M. disk.

5 Device Management

Device management is the management of the operation, maintenance, and implementation of a physical device that is connected to the computer, or a virtual one. Devices are managed by device drivers, which are kernel modules that give the OS an interface through which it can control the device.

As mentioned above, there is a false filesystem sitting on top of the FreeMiNT filesystem, and so every device has its own entry in the \dev "directory".

FreeMiNT uses file handles to refer to devices, which are all as follows:

- -1 points to the current control terminal
- -2 points to the RS232 port, located under U:\DEV\MODEM1
- -3 points to the printer port, U:\DEV\CENTR
- -4 points to the MIDI input device, and
- -5 point to the MIDI output device.

References

[1] Eric Smith. *freemint/mint.txt at master · freemint/freemint*. Jan. 2020. URL: https://github.com/freemint/freemint/blob/master/doc/programmer/mint.txt.

[2] AltF4. URL: http://freemint.de/en/.

[3] *freemint screenshot*. URL: https://sourceforge.net/projects/freemint/.

[4] Rolf Kotzian. *tos.hyp - documentation of all functions of TOS*. Ed. by Peter A.Translator West. URL: https://freemint.github.io/tos.hyp/.

[5] *multitos screenshot*. URL: https://upload.wikimedia.org/wikipedia/commons/thumb/4/43/MultiTOS.png/450px-MultiTOS.png.

[6] John Townsend. *TOS$_R$evisions*. URL: https://www.atari.st/content.php?type=t&file=toslist.

[7] various. *User manual · freemint/freemint Wiki*. URL: https://github.com/freemint/freemint/wiki/User_manual.