yhOS documentation

aceinetx (c) 2022-2025

System calls

id (%eax)	%ebx	%ecx	%edx	description
SYS_WRITEC - 0	character	-	-	Prints a character to the screen
SYS_WRITE - 1	buffer	-	-	Prints a null terminated string to the screen
SYS_GETS - 2	buffer	length	-	Requests a string from the user
SYS_VFSWRITE - 3	filename (char*)	buffer (char*)	buffer_size (dword)	Writes a file to the virtual file system. If no file is found it will create it (if there is enough space). Returns 0 in eax if fails. Returns a content address if doesn't fail
SYS_VFSREAD - 4	filename (char*)	buffer (char*)	buffer_size (dword)	Reads a file from the virtual file system. Returns 0 in eax if fails. Returns a content address if doesn't fail
SYS_VFSQUERY - 5	filename (char*)	-	-	Queries file size from the virtual file system. Returns -1 in eax if fails
SYS_ALLOC - 6	size	-	-	Allocates dynamic memory. (Exposes yalloc to raw assembly programs). Returns allocated address in eax
SYS_FREE - 7	addr	-	-	Free's dynamic memory. (Exposes yfree to raw assembly programs)
SYS_VFSBASE - 8	-	-	-	Returns a pointer to the beggining of virtual file system. (The returned pointer can become unusable if the filesystem size changes)
SYS_EXEAR	-	-	-	Get next argument

G - 9				passed to the executable from the sheel. Returns a pointer to the argument. Need to free
SYS_VFSHANDLE - 10	filename (char*)	-	-	Returns a pointer to the start address of a file. (The returned pointer can become unusable if the filesystem size changes)
SYS_ITOA - 11	number	buffer (char*)	buffer size	Converts a number into a base-10 string
SYS_ITOA16 - 12	number	buffer (char*)	buffer size	Similar to SYS_ITOA. Converts a number into a base-16 string
SYS_GETCWD - 13	-	-	-	Returns a pointer to a string containing current working directory

A writec system call example in flat assembler:

```
mov eax, SYS_WRITEC
mov ebx, 0x45
int 0x80
```

And in C:

```
syscall(SYS_WRITEC, 'E');
```

Virtual file system

yhOS has a mechanism called "Virtual file system". You can think of it like a virtual hard drive, it stores every file in RAM which unloads on restart.
Virtual file system is structured as a pointer to array of structures called vfs_file. This structure contains these variables: a char* filename, char* content, and dword size. Virtual file system is resizable at runtime via the dynamic memory allocator.
System calls that start with SYS_VFS are not for I/O but for the virtual file system

yhOS Static Executable Format (yhSE)

This is the default format yhOS uses for executables. This is the structure of yhSE header: (YHSE_IDENT is usually 5)

```
typedef struct {
  char ident[YHSE_IDENT];
  dword entry;
} yhse_hdr;
```

yhOS provides a toolchain to easily create these executables, the toolchain is located in /yhse directory and has these tools:

- i386-yhse-gcc
- i386-yhse-g++
- i386-yhse-ld

Remember: this is not actually a toolchain, but rather a collection helper scripts, linker scripts and header files to actually build the executable. So you would still need i386-elf toolchain installed on your machine

Magic addresses

yhOS has a few magic addresses that you can use in your programs. Here's the table of them:

Address	Description
0x140000 (constant)	Saves a empty vga buffer
0x200000	Dynamic memory allocator start
0x40000	Default executable load address