9.0 QtSpim System Service Calls

The operating system must provide some basic services for functions that a user program can not easily perform on its own. Some key examples include input and output operations. These functions are typically referred to as *system services*. The QtSpim simulator provides a series of operating system like services by using a **syscall** instruction.

To request a specific service from the QtSpim simulator, the 'call code' is loaded in the **$v0** register. Based on the specific system service being requested, additional information may be needed which is loaded in the argument registers (as noted in the Procedures/Functions section).

9.1 Supported QtSpim System Services

A list of the supported system services is listed in the below table. A series of examples are provided in the following sections.

Service Name	**Call Code**	**Input**	**Output**
Print Integer (32-bit)	1	**$a0** → integer to be printed	
Print Float (32-bit)	2	**$f12** → 32-bit floating-point value to be printed	
Print Double (64-bit)	3	**$f12** → 64-bit floating-point value to be printed	
Print String	4	**$a0** → starting address of NULL terminated string to be printed	
Read Integer (32-bit)	5		**$v0** → 32-bit integer entered by user
Read Float (32-bit)	6		**$f0** → 32-bit floating-point value entered by user

Read Double (64-bit)	7		**$f0** → 64-bit floating-point value entered by user
Read String	8	**$a0** → starting address of buffer (of where to store character entered by user) **$a1** → length of buffer	
Allocate Memory	9	**$a0** → number of bytes to allocate	**$v0** → starting address of allocated memory
Terminate	10		
Print Character	11	**$a0** → character to be printed	
Read Character	12		**$v0** → character entered by user
File Open	13	**$a0** → file name string, NULL terminated **$a1** → access flags **$a2** → file mode, (UNIX style)	**$v0** → file descriptor
File Read	14	**$a0** → file descriptor **$a1** → buffer starting address **$a2** → number of bytes to read	**$v0** → number of bytes actually read from file (-1 = error, 0 = end of file)
File Write	15	**$a0** → file descriptor **$a1** → buffer starting address **$a2** → number of bytes to read	**$v0** → number of bytes actually written to file (-1 = error, 0 = end of file)
File Close	16	**$a0** → file descriptor	

The file open access flags are defined as follows:

```
Read = 0x0, Write = 0x1, Read/Write = 0x2
OR Create = 0x100, Truncate = 0x200, Append = 0x8
OR Text = 0x4000, Binary = 0x8000
```

For example, for a file read operation the 0x0 would be selected. For a file write operation, the 0x1 would be selected.