ne u previo
Next: Description of the MIPS **Up:** SPIM **Previous:** Assembler Syntax

System Calls

Table: System services.

Service	System Call Code	Arguments	Result
print_int	1	$a0 = integer	
print_float	2	$f12 = float	
print_double	3	$f12 = double	
print_string	4	$a0 = string	
read_int	5		integer (in $v0)
read_float	6		float (in $f0)
read_double	7		double (in $f0)
read_string	8	$a0 = buffer, $a1 = length	
sbrk	9	$a0 = amount	address (in $v0)
exit	10		
print_character	11	$a0 = character	
read_character	12		character (in $v0)
open	13	$a0 = filename,	file descriptor (in $v0)
		$a1 = flags, $a2 = mode	
read	14	$a0 = file descriptor,	bytes read (in $v0)
		$a1 = buffer, $a2 = count	
write	15	$a0 = file descriptor,	bytes written (in $v0)
		$a1 = buffer, $a2 = count	
close	16	$a0 = file descriptor	0 (in $v0)
exit2	17	$a0 = value	

SPIM provides a small set of operating-system-like services through the system call (syscall) instruction.

To request a service, a program loads the system call code (see Table [*]) into register $v0 and the arguments into registers $a0 . . . $a3 (or $f12 for floating point values). System calls that return values put their result in register $v0 (or $f0 for floating point results). For example, to print ``the answer = 5'', use the commands:

```
      .data
 str:  .asciiz "the answer = "
      .text
      li $v0, 4       # system call code for print_str
```

```
    la $a0, str     # address of string to print
    syscall         # print the string

    li $v0, 1       # system call code for print_int
    li $a0, 5       # integer to print
    syscall         # print it
```

print_int is passed an integer and prints it on the console. print_float prints a single floating point number. print_double prints a double precision number. print_string is passed a pointer to a null-terminated string, which it writes to the console. print_character prints a single ASCII character.

read_int, read_float, and read_double read an entire line of input up to and including the newline. Characters following the number are ignored. read_string has the same semantics as the Unix library routine fgets. It reads up to $n-1$ characters into a buffer and terminates the string with a null byte. If there are fewer characters on the current line, it reads through the newline and again null-terminates the string. read_character reads a single ASCII character. **Warning:** programs that use these syscalls to read from the terminal should not use memory-mapped IO (see Section [*]).

sbrk returns a pointer to a block of memory containing n additional bytes. This pointer is word aligned. exit stops a program from running. exit2 stops the program from running and takes an argument, which is the value that spim or xspim uses in its call on exit.

open, read, write and close behave the same as the Unix system calls of the same name. They all return -1 on failure.

---

ne u previo

Next: Description of the MIPS **Up:** SPIM **Previous:** Assembler Syntax
Ian Moor 2009-03-11