A DIGITAL UNIX Realtime Functional Summary

This appendix summarizes the functions that are of particular interest to realtime application developers. The source of these functions ranges from System V to POSIX 1003.1 and POSIX 1003.1b. The tables given in this appendix serve as a guide in application development, but you may need to consult the online reference pages for additional information or pointers to additional functions and commands.

The function tables are arranged according to the following categories:

- Process Control Functions, Table A-1
- P1003.1b Priority Scheduling Functions, Table A-2
- P1003.1b Clock Functions, Table A-3
- Date and Time Conversion Functions, Table A-4
- P1003.1b Timer Functions, Table A-5
- BSD Clock and Timer Functions, Table A-6
- P1003.1b Memory Locking Functions, Table A-7
- System V Memory Locking Function, Table A-8
- P1003.1b Asynchronous I/O Functions, Table A-9
- POSIX Synchronized I/O Functions, Table A-10
- BSD Synchronized I/O Function, Table A-11
- P1003.1b Message Functions, Table A-12
- P1003.1b Shared Memory Functions, Table A-13
- P1003.1b Semaphore Functions, Table A-14
- P1003.1b Realtime Signal Functions, Table A-15
- Signal Control and Other Signal Functions, Table A-16
- `sigsetops` Primitives, Table A-17
- Process Ownership Functions, Table A-18
- Input and Output Functions, Table A-19
- Device Control Functions, Table A-20
- System Database Functions, Table A-21

Table A-1: Process Control Functions

Function	Purpose
`alarm`	Sends the calling process a SIGALRM signal after a specified number of seconds
`exit`	Terminates the calling process
`exec`	Runs a new image, replacing the current running image
`fork`	Creates a new process
`getenv`	Reads an environment list
`isatty`	Verifies whether a file descriptor is associated with a terminal
`kill`	Sends a signal to a process or a group of processes
`malloc`	Allocates memory
`pause`	Suspends the calling process until a signal of a certain type is delivered
`sleep`	Suspends the current process either for a specified period or until a signal of a certain class is delivered
`sysconf`	Gets the current value of a configurable system limit or option
`uname`	Returns information about the current state of the operating system
`wait`	Lets a parent process get status information for a child that has stopped, and delays the parent process until a signal arrives
`waitpid`	Lets a parent process get status information for a specific child that has stopped and delays the parent process until a signal arrives from that child or that child terminates

Table A-2: P1003.1b Priority Scheduling Functions

Function	Purpose
`sched_getscheduler`	Returns the scheduling policy of a specified process
`sched_getparam`	Returns the scheduling priority of a specified process
`sched_get_priority_max`	Returns the maximum priority allowed for a scheduling policy
`sched_get_priority_min`	Returns the minimum priority allowed for a scheduling policy
`sched_rr_get_interval`	Returns the current quantum for the round-robin scheduling policy
`sched_setscheduler`	Sets the scheduling policy and priority of a specified process
`sched_setparam`	Sets the scheduling priority of a specified process
`sched_yield`	Yields execution to another process

Table A-3: P1003.1b Clock Functions

Function	Purpose
`clock_getres`	Returns the resolution of the specified clock
`clock_gettime`	Returns the current value for the specified clock
`clock_settime`	Sets the specified clock to the specified value

Table A-4: Date and Time Conversion Functions

Function	Purpose
`asctime`	Converts time units (hours, minutes, and seconds) into a 26-character string
`ctime`	Converts a time in seconds since the Epoch to an ASCII string in the form generated by `asctime`
`difftime`	Computes the difference between two calendar times (time1-time0) and returns the difference expressed in seconds
`gmtime`	Converts a calendar time into time units, expressed as GMT
`localtime`	Converts a time in seconds since the Epoch into time units
`mktime`	Converts the time units in the `tm` structure pointed to by *timeptr* into a calendar time value with the same encoding as that of the values returned by `time`
`tzset`	Sets the external variable *tzname*, which contains current time zone names

Table A-5: P1003.1b Timer Functions

Function	Purpose
`nanosleep`	Causes the calling process to suspend execution for a specified period of time
`timer_create`	Returns a unique timer ID used in subsequent calls to identify a timer based on the systemwide clock
`timer_delete`	Removes a previously allocated, specified timer
`timer_getoverrun`	Returns the timer expiration overrun count for the specified timer
`timer_gettime`	Returns the amount of time before the specified timer is due to expire and the repetition value
`timer_settime`	Sets the value of the specified timer to either an offset from the current clock setting or an absolute value

Table A-6: BSD Clock and Timer Functions

Function	Purpose
`getitimer`	Returns the amount of time before the timer expires and the repetition value
`gettimeofday`	Gets the time of day
`setitimer`	Sets the value of the specified timer
`settimeofday`	Sets the time of day

Table A-7: P1003.1b Memory Locking Functions

Function	Purpose
`mlock`	Locks a specified region of a process's address space
`mlockall`	Locks all of a process's address space
`munlock`	Unlocks a specified region of a process's address space
`munlockall`	Unlocks all of a process's address space

Table A-8: System V Memory Locking Function

Function	Purpose
`plock`	Locks and unlocks a process, text, or data in memory

Table A-9: P1003.1b Asynchronous I/O Functions

Function	Purpose
`aio_cancel`	Cancels one or more requests pending against the file descriptor
`aio_error`	Returns the error status of a specified operation
`aio_fsync`	Asynchronously writes system buffers containing a file's modified data to permanent storage
`aio_read`	Initiates a read request on the specified file descriptor
`aio_return`	Returns the status of a completed operation
`aio_suspend`	Suspends the calling process until at least one of the specified requests has completed
`aio_write`	Initiates a write request to the specified file descriptor
`lio_listio`	Initiates a list of requests

Table A-10: POSIX Synchronized I/O Functions

Function	Purpose
`fcntl`	Controls operations on files and memory objects
`fdatasync`	Flushes modified data only from the buffer cache, providing operation completion with data integrity
`fsync`	Flushes modified data and file control information from the buffer cache, providing operation completion with file integrity

Table A-11: BSD Synchronized I/O Function

Function	Purpose
`sync`	Updates all file systems -- all information in memory that should be on disk is written out

Table A-12: P1003.1b Message Functions

Function	Purpose
`mq_close`	Closes a message queue
`mq_getattr`	Retrieves the attributes of a message queue
`mq_notify`	Requests that a process be notified when a message is available on a queue
`mq_open`	Opens a message queue
`mq_receive`	Receives a message from the queue
`mq_send`	Sends a message to a queue
`mq_setattr`	Sets the attributes of a message queue
`mq_unlink`	Removes a message queue

Table A-13: P1003.1b Shared Memory Functions

Function	Purpose
`shm_open`	Opens a shared-memory object, returning a file descriptor
`shm_unlink`	Removes the name of the shared-memory object

Table A-14: P1003.1b Semaphore Functions

Function	Purpose
`sem_close`	Deallocates the specified named semaphore
`sem_destroy`	Destroys an unnamed semaphore

Function	Purpose
sem_getvalue	Gets the value of a specified semaphore
sem_init	Initializes an unnamed semaphore
sem_open	Opens/creates a named semaphore for use by a process
sem_post	Unlocks a locked semaphore
sem_trywait	Performs a semaphore lock on a semaphore only if it can lock the semaphore without waiting for another process to unlock it
sem_unlink	Removes a specified named semaphore
sem_wait	Performs a semaphore lock on a semaphore

Table A-15: POSIX 1003.1b Realtime Signal Functions

Function	Purpose
sigaction	Specifies the action a process takes when a particular signal is delivered
sigqueue	Sends a signal, plus identifying information, to a process
sigtimedwait	Waits for a signal for the specified amount of time and, if the signal is delivered within that time, returns the signal number and any identifying information the signaling process provided
sigwaitinfo	Waits for a signal and, upon its delivery, returns the signal number and any identifying information the signaling process provided

Table A-16: Signal Control and Other Signal Functions

Function	Purpose
signal	Changes the action of a signal
sigpending	Returns a signal set that represents those signals that are blocked from delivery to the process but are pending
sigprocmask	Sets the process's current blocked signal mask
sigsetops	Manipulates signal sets
sigsuspend	Replaces the process's current blocked signal mask, waits for a signal, and, upon its delivery, calls the handler established for the signal and returns
sigwait	Suspends a calling thread until a signal arrives

Table A-17: sigsetops Primitives

Function	Purpose
sigaddset	Adds a signal to the signal set

Function	Purpose
`sigdelset`	Removes a signal from the signal set
`sigemptyset`	Initializes a signal set such that all signals are excluded
`sigfillset`	Initializes a signal set such that all signals are included
`sigismember`	Tests whether a signal is a member of the signal set

Table A-18: Process Ownership Functions

Function	Purpose
`geteuid`	Returns the effective user ID of the calling process
`getegid`	Returns the effective group ID of the calling process
`getgid`	Returns the real group ID of the calling process
`getpgrp`	Returns the process group ID of the calling process
`getpid`	Returns the process ID of the calling process
`getppid`	Returns the process ID of the parent of the calling process
`getuid`	Returns the real user ID of the calling process
`setgid`	Sets the group ID of the calling process
`setsid`	Creates a new session, for which the calling process is the session leader
`setuid`	Sets the user ID of the calling process

Table A-19: Input and Output Functions

Function	Purpose
`close`	Closes a file
`dup`	Duplicates a file descriptor
`dup2`	Duplicates a file descriptor
`fileno`	Retrieves a file descriptor
`lseek`	Moves a pointer to a record within a file
`mkfifo`	Creates fifo special files
`open`	Opens a file
`pipe`	Creates an interprocess channel
`read`	Reads the specified number of bytes from a file
`write`	Writes the specified number of bytes to a file

Table A-20: Device Control Functions

Function	Purpose
`cfgetispeed`	Retrieves the input baud rate for a terminal
`cfgetospeed`	Retrieves the output baud rate for a terminal
`cfsetispeed`	Sets the input baud rate for a terminal
`cfsetospeed`	Sets the output baud rate for a terminal
`isatty`	Verifies whether a file descriptor is associated with a terminal
`tcdrain`	Causes a process to wait until all output has been transmitted
`tcflow`	Suspends or restarts the transmission or reception of data
`tcflush`	Discards data that is waiting to be transmitted
`tcgetattr`	Retrieves information on the state of a terminal
`tcsendbreak`	Sends a break character for a specified amount of time
`tcsetattr`	Applies a set of attributes to a terminal

Table A-21: System Database Functions

Function	Purpose
`getgrgid`	Returns group information when passed a group ID
`getgrnam`	Returns group information when passed a group name
`getpwnam`	Returns user information when passed a user name
`getpwuid`	Returns user information when passed a user ID