



VeriWiz Code Base

The code base includes the following:

OS

A real-time, prioritized preemptive task scheduler capable of 100,000 task switches per second with 60 MHz clock. Tasks are easily declared. Tasks communicate with the system via *Service Requests*, such as *wait until this byte takes value xx*, or *wait for this int to take value xx*, or *until y seconds have expired*. Adding a service request type is easy if required, but there is a request to wait until a given function returns a specific value, which can be used as a catch all request.

Tasks can invoke and suspend others. For example a server task can invoke and suspend response tasks as necessary. Suspended task don't consume scheduling time.

Timer Tick Function Calls

OS *timer call backs* to small application functions at specific timer ticks, i.e. at 65,536 times a second, or other powers-of-two divisors of the system tick clock.

Timer call backs are generally used to tightly couple a task to the clock. The called function may increment a counter and/or set a variable to a given value. The task waiting for the counter or variable will then be scheduled at the next opportunity based on task priority.

Task Mail

OS managed task communications. Tasks can wait on mail reception or a bounding time or function. Tasks can send mail non-blocking, or block until mail is received and/or consumed.

General mail can be handled through the use of a structure that has provisions for byte length, the amount remaining for the consumer, and a slot to pass arguments back to the sender/

Task to task communications can also be done by thread safe linked list, or using protected signal variables or functions.

Resource Locks, (Mutexes)

Resource locks are used to prevent concurrent access to a common data structure, such as malloc/free, linked lists, queues common variables, and functions. Two functions, `get_lock` and `release_lock` are placed around the code to be protected. These functions are written in assembly language.

Malloc and Free

A thread-safe, small and very fast memory manager. Using a first fit algorithm, which has been shown to be the fastest method, and with low fragmentation, this version has been pounded on and stands up to it.

Printf and Scanf

Thread-safe integer version with a low memory imprint. Mostly compatible with the Linux version, additions have been made to allow format strings with decimal point specifiers to be used with integers where fixed point is used.

For example an integer value of 3456 can be displayed as 34.56 using a decimal point format string.

Double Linked Lists

Thread-safe double linked lists can be used as LIFO, FIFO, or for node insertion/deletions. Opening a list returns a non-moving head that is used thereafter to insert or delete items. The head also has a pointer to the list lock, and the functions to use for malloc and free. The latter functions can be the system malloc and free, or any other buffer management functions.

Lists can use an insert-sort function to order a list. A list traverse function is provided that calls back a function for each list element. List accesses are protected by locks to allow usage between pre-emptable tasks.

Simple Queues

Queues are implemented for bytes and pointers. The number of elements in queues can be queried. Queue length is set when the queue is opened.

Stdio Lib

The stdio library contains the usual suspects such as `fgetc`, `fputc`, etc.

String Lib

Again, the usual string functions but written in-house to avoid bugs, buffer overflows, and operate in ROM as well as RAM. That is, functions do not write into a string passed into it.

Math Lib

As noted earlier, this is a hack of the Linux newlib to permit some limited floating point operations.