

SAFE VIRTUAL MACHINE FOR C

The ultra small target independent Virtual Machine

Dynamic code in C for deeply embedded and distributed systems

Altreonic is breaking again new grounds by complementing its OpenComRTOS with the capability to run processor independent code. The code is binary compiled and linked C code, resulting in only 3.8 Kbytes for the safety enhanced Virtual Machine task running the code. As the native OpenComRTOS also only requires Kbytes, this capability opens new possibilities for memory and power constrained embedded systems.

Capabilities:

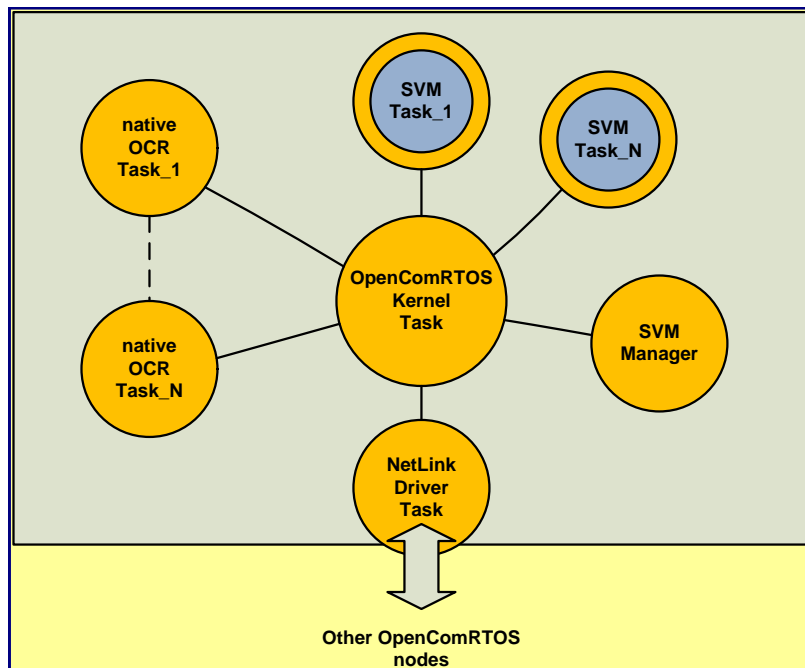
- Loading and running code on a remote node without halting it
- Dynamic code loading at runtime
- Dynamic task migration at runtime
- Transparent communication with native OpenComRTOS tasks on the same or on remote nodes
- Safe execution: SafeVM tasks cannot violate memory boundaries.

Future releases will also support calling native functions for maximum performance and direct execution of the binary images on the target processor (ARM only).

Applications:

- Remote diagnostics
- Fail safe and fault tolerant control
- Processor independent programming

Thanks to the use of OpenComRTOS, SafeVM tasks can operate system wide across all nodes in the network. The user can also put several SafeVM tasks on the same node.



How it works:

The SafeVM capability is provided by a SafeVM task manager that can start and stop the SVM task, monitor their status and interact with the SafeVM host task. The latter is composed of a safe VM execution module and the actual SafeVM task. The SafeVM task itself interacts with the rest of the node and the network through pre-compiled services, including the standard OpenComRTOS services.

From Deep Space to Deep Sea

WWW. ALTREONIC.COM

Push button high reliability



Available SafeVM task services:

SVM_LoadTask
// loads the SVM task in the SVM workspace

SVM_StartTask
// starts the SVM task

SVM_StopTask
// Stops the SVM task

SVM_ClearTask
// Clears the workspace memory of the SVM
// Equivalent to unloading the SVM task

SVM_GetErrorInfo
// Returns the last error code

SVM_GetState
// Returns the current state of the SVM

OpenComRTOS services

OpenComRTOS services operate transparently across a networked or multicore system.

Services include: preemptive priority based task scheduling, events, counting semaphores, fifos, resource locks with support for priority inheritance, packet pools, memory pools.

See the OpenComRTOS datasheets and manuals for more details.

The SVM task can be written and compiled using all features of ANSI-C, including libraries.

Relative performance

Measured on ARM Cortex M3:

Program code for SVM
(manager + execution module): 3.8 KBytes

Relative RTOS performance:
Example semaphore loop with one task being a SVM_Task: about 7x slower.

Computational performance:
depends on application code.

Example program:

This example first loads a first SVM Task on the node, waits until it stops, and then uploads a new SVM Task to the node and starts it.

```
#include <SVM_HostService/  
        SVM_HostClient.h>  
  
void Task1_entry(L1_TaskArguments Args)  
{  
    FILE *fo;  
    SVM_STATE state;  
  
    // TASK1 - binary file with image  
    fo = fopen(SVM_TASK1,"r");  
    assert(fo!=NULL);  
  
    // buffer for storing program  
    L1_UINT32 buffer[128];  
    L1_UINT32 n = fread(buffer,  
                        sizeof(L1_UINT32),  
                        128,  
                        fo);  
    assert(n!=0);  
  
    // send task to SVM task  
    SVM_LoadTask(SVM_Manager,  
                buffer,  
                n*4,  
                5000);  
  
    // start task on the node  
    SVM_StartTask(SVM_Manager);  
    // wait until first task stops  
    do  
    {  
        SVM_GetState(SVM,&state);  
        if(state==VSP_VM_ERROR)  
            exit(1);  
    }  
    while(state!=SVM_VM_STOPPED);  
  
    // Load second task  
    // SVM_TASK2 - binary file with image  
    fo = fopen(SVM_TASK2,"r");  
    assert(fo!=NULL);  
  
    n = fread(buffer,  
            sizeof(L1_UINT32),  
            128,  
            fo);  
    assert(n!=0);  
    SVM_LoadTask(SVM_Manager,  
                buffer,  
                n*4,  
                5000);
```

From Deep Space to Deep Sea
WWW. ALTREONIC.COM

Push button high reliability



Contact: **Altreonic NV**
Gemeentestraat 61A b1
B3210 Linden—Belgium
Tel.:+32 16 202059
info.request @ altreonic.com