AN1153: Developing Connect vNCP Applications with Micrium OS

The Silicon Labs Flex SDK provides support for running the Connect stack on top of the Micrium OS kernel, a full-featured RTOS for embedded systems. Support for Micrium is integrated into the AppBuilder utility in Simplicity Studio. The Connect stack runs within a Micrium Task while the application framework code runs within a separate Micrium Task.

This two-task model is also known as the “virtual Network Co-Processor” architecture, or “vNCP” for short. This is because the processing and communication between the two tasks is the same as for the host/NCP architecture. The difference is that instead of running on separate processors and passing messages to each other through the serial port, the application and NCP run on the same processor but in different Micrium tasks. Message passing is handled using Micrium message queues or protected global data structures, and is transparent to the application.

For documentation on Micrium OS itself, please see http://doc.micrium.com.

KEY POINTS

- Micrium OS support for Connect is available in Simplicity Studio in AppBuilder.
- The app task and Connect stack task run as separate threads and communicate via an IPC protocol (known as vNCP)

1. Getting Started

Any Connect application can be easily turned into an application running on Micrium OS by simply enabling the **Micrium RTOS** plugin.

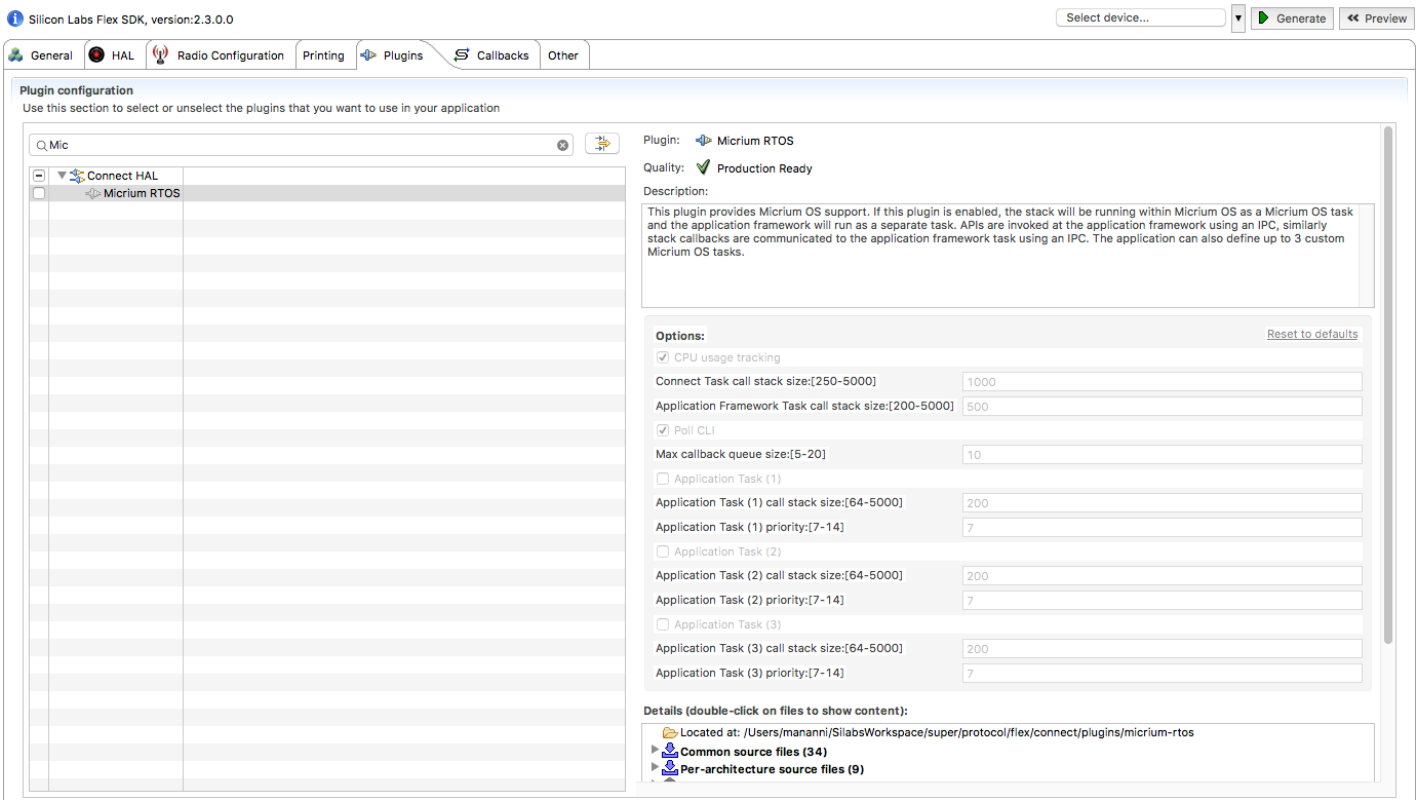


Figure 1.1. Micrium RTOS Plugin

The **Micrium RTOS** plugin has a few plugin options that allow the user to customize certain settings. In particular:

- **CPU usage tracking**: If enabled, the application framework compiles in some additional code that allows certain debug tools to have extended debug information from the OS.
- **Connect Task call stack size**: The size in bytes of the call stack used by the stack task.
- **Application Framework Task call stack size**: The size in bytes of the call stack used by the application framework task.
- **Poll CLI**: If enabled, the application framework task (which is responsible for processing the incoming serial CLI commands) is allowed to yield only for a small amount of time, resulting in a responsive CLI user experience.
- **Max callback queue size:** The maximum number of simultenous callback messages from the stack task to the application tasks.
- **Application Task (n)**: The user can also optionally enable up to three custom application tasks. For each of these tasks, the user can specify the call stack size and the priority. Notice that the allowed priority range (7-14) is always lower priority than both the stack task and the application framework task.

2. Virtual NCP (vNCP)—Architecture Details

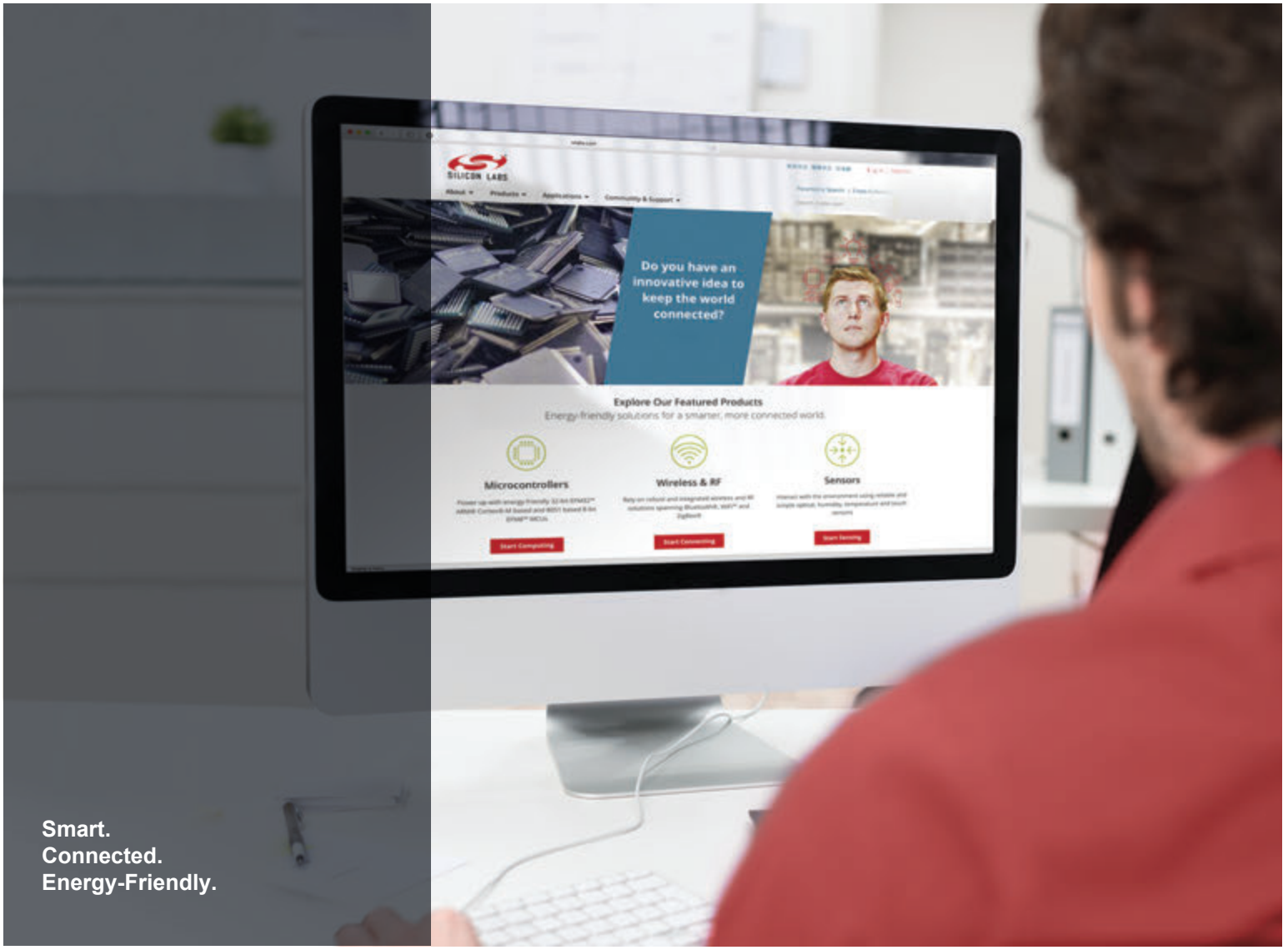
By enabling the Micrium RTOS plugin, the Micrium kernel code is added to the project. The plugin code automatically starts all the OS tasks as per use configuration. At minimum, two tasks are created:

- A **stack task** (highest priority task) responsible for running the Connect stack. In particular, this task is responsible for the following operations:
  - Periodically tick the Connect stack by invoking the `emberTick()` API.
  - Process incoming IPC (inter-process communication) commands from application tasks (if any) and send out a response.
  - Send callback IPC commands (if any) to the application tasks.
  - Attempt whenever possible to suspend itself to allow lower priority application tasks to run.
- An **application framework task** responsible for running the application framework code. In particular, this task is responsible for the following operations:
  - Periodically tick the application framework plugins (for those that implement a tick callback). This includes, for instance, the processing of incoming CLI commands.
  - Run application events.
  - Process incoming callback IPC commands (if any) from the stack task.
  - Attempt whenever possible to suspend itself to allow lower priority custom application tasks to run.

The Micrium RTOS plugin also allows to enable up to three custom application tasks. The custom application tasks must have a lower priority than both the stack task and the application framework task. Each of these custom application tasks is started automatically and should implement the following two callbacks:

- `emberAfPluginMicriumRtosAppTaskNInitCallback(void)` – This callback is invoked at initialization time to allow the N-th custom application task to execute some initialization code.
- `emberAfPluginMicriumRtosAppTaskNMainLoopCallback(void *p_arg)` – This callback shall contain the N-th task main loop.

Custom application tasks can call stack APIs freely since the underlying IPC protocol ensures that all stack APIs are thread-safe.



Products
www.silabs.com/products

Quality
www.silabs.com/quality

Support and Community
community.silabs.com

SILICON LABS®

Silicon Laboratories Inc.
400 West Cesar Chavez
Austin, TX 78701
USA

http://www.silabs.com