works with
BY SILICON LABS
VIRTUAL CONFERENCE

SILICON LABS

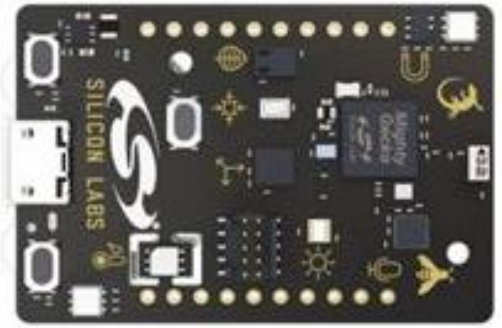
PiP Placeholder Window
(Remove Before Distribution)

Exploring RTOS Options for Wireless IoT Projects

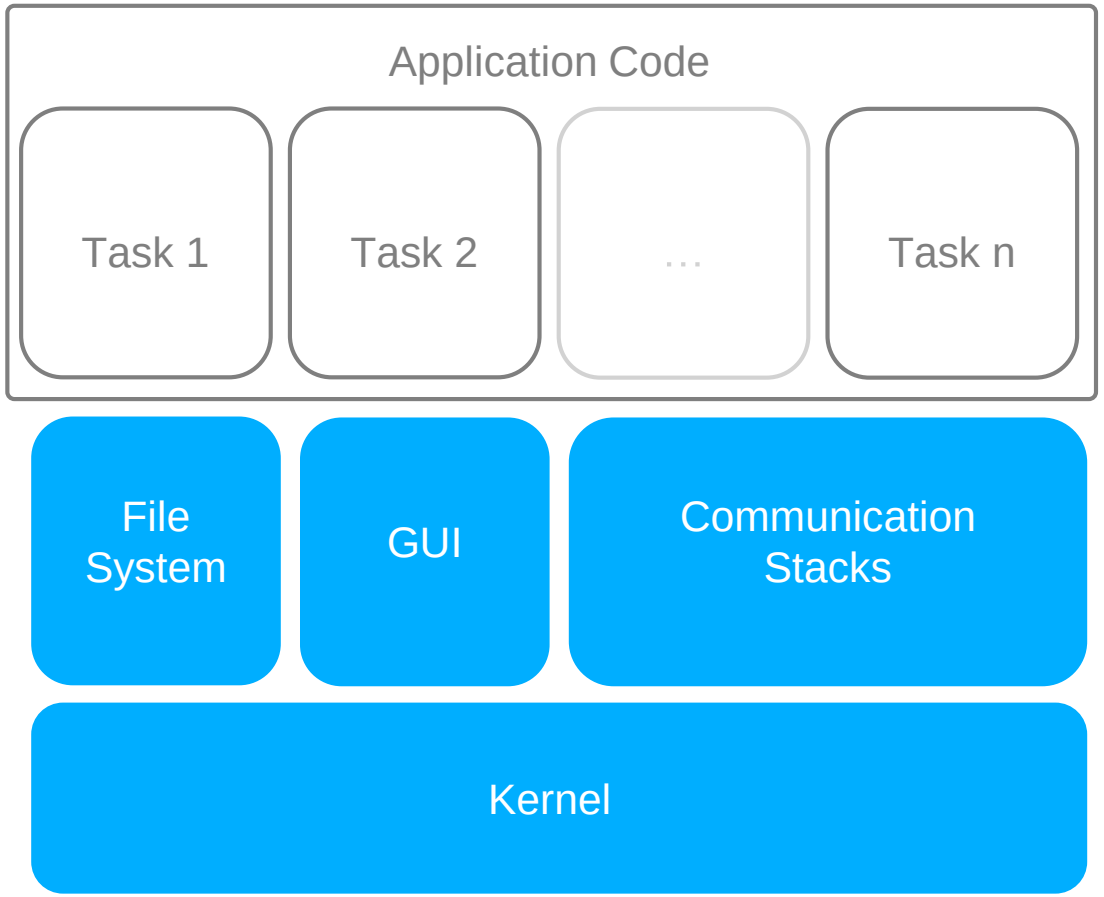
Matt Gordon, Sr. Product Manager IoT OS

Objectives

- Provide a high-level overview of RTOS functionality
  - This session *is not* intended to be a deep-dive on RTOS theory
- Discuss the rationale for using an RTOS in a wireless IOT project
  - The focus here is on the benefits of kernel vs bare-metal code
- Introduce the different RTOSes available to developers on EFR32 wireless devices
  - Multiple RTOSes, each with strong technical specifications, are supported
- Lab: Walk through the process for getting started with a couple of different RTOSes on EFR32
  - **An opportunity for hands-on RTOS development using the Simplicity Studio IDE and the Thunderboard Sense 2**



A High-Level RTOS Introduction



Real-Time Operating System (RTOS)

- A framework for writing multi-task application code
  - Alternative to bare-metal, or super-loop, applications
- Embedded RTOSes tend to be relatively lightweight
  - Goal is efficient operation on resource-constrained devices
- Based on a kernel that provides task scheduling services
  - Kernel is often 15 kBytes of code or less
- “RTOS” label may be applied to a broad collection of SW
  - File system, GUI, protocol stacks, drivers, etc.
- The lab portion of this session will focus on the kernel
  - Discussion beforehand will touch on other components

Do I Need an RTOS?

- The fundamental decision is kernel vs. bare-metal
  - Two different approaches to structuring application code
- Any application *could* be written without a kernel
  - Silicon Labs requires a kernel for DMP, Wi-SUN, and Z-Wave
  - Kernel is optional for BLE, proprietary wireless, and Zigbee
- Decision on using a kernel should involve multiple criteria
  - Complexity of code (including stacks)
  - Future plans for expanding the application w/new features
  - Development team size and background
  - Available Flash and RAM on the HW platform
- Lab highlights two applications in which kernel can be helpful
  - BLE and proprietary wireless

Bare-Metal

```
int main (void)
{
  while (1) {
    ADC_read();
    UART_handler();
    …
  }
}
```

Functions called from `main()` loop

Kernel

```
void ADC_Task (void)
{
  while (1) {
    ADC_read();
    //1 ms sleep
    OSTimeDly(1);
  }
}
```

```
void UART_Task (void)
{
  while (1) {
    UART_Handler();
    //Wait for data
    OSSemPend(&MySem);
  }
}
```

Functions called from tasks managed and scheduled by the kernel

RTOS Support in Simplicity Studio

- Simplicity Studio is Silicon Labs' IDE for EFR32 devices
  - Eclipse-based, with a number of helpful plugins and extensions
- The IDE makes it easy to get started with an RTOS
  - RTOS-based examples are provided as references
  - Configuration tools automate addition of RTOS code to new projects
- Currently, there are two RTOS options in Studio
  - **FreeRTOS:** Popular kernel used across the embedded space
  - **Micrium OS:** Full software suite from longtime commercial OS vendor
- FreeRTOS and Micrium OS are part of GSDK Suite
  - Full-featured, integrated software platform
  - Includes wireless stacks, most of which are compatible with either OS
  - Also includes *Amazon* FreeRTOS libraries that supplement the kernel

3rd-Party RTOS Repositories

- As the IoT has evolved, so has the RTOS world
- Connectvity, security, etc. are increasingly important
- Role of cloud providers and open-source communities is growing
  - Some of these organizations serve as RTOS developers and integrators
  - They offer tools, docs, etc. for building RTOS-based projects
  - In some cases, their solutions target connectivity to a particular cloud
- Silicon Labs works closely with a number of RTOS providers
- EFR32 examples are being developed for Azure RTOS and Zephyr
  - Projects will be delivered via GitHub
  - Tools and build environment mosty established by the RTOS provider
  - Community-contributed projects are avaialble now for Zephyr

Choosing the Right RTOS for Your Project

- All of the RTOSes that Silicon Labs supports have strong technical specifications
- The recommended RTOS for you depends on your background

Simplicity Studio and GSDK Suite

Recommended for…

- Developers who are already familiar with FreeRTOS or Micrium
- Developers who are completely new to RTOSes
- Developers with multi-protocol wireless projects

3rd-Party OS Repositories

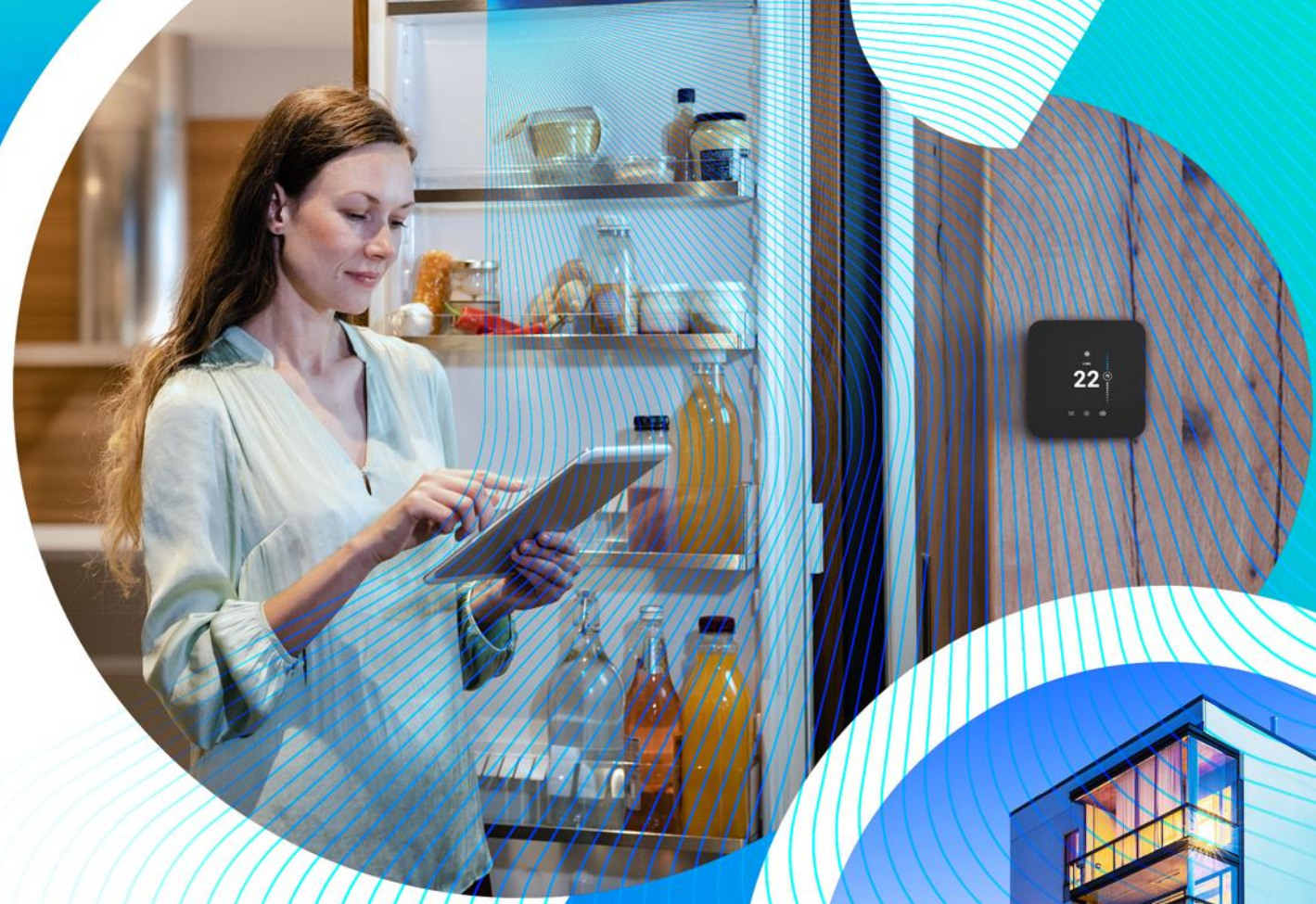
GitHub

Recommended for…

- Developers who are already familiar with Azure or Zephyr
- Developers who are connecting to the Azure cloud

works with
BY SILICON LABS

VIRTUAL CONFERENCE



SILICON LABS