

# **Project Configurator & Gecko Platform - Lab**

This lab procedure walks through the steps to create a new project utilizing Simplicity Studio 5's Gecko Platform and Project Configurator Wizard. The lab highlights the new process action programming model, a few of the software components available within Project Configurator, and goes into detail about integrating application code available through our peripheral examples repository on GitHub.

There are four project stages throughout the lab signifying a different software addition to the demostration, with each stage building upon the last.

### KEY POINTS

- Gecko Platform and Project Configurator Wizard
- Software component installation and configuration
- Addition of application code (ADC example)
- RTOS and Bluetooth project integration

The first part shows how to build a new project which uses the CLI interface and onboard sensor of the Thunderboard Sense 2, starting with an Empty C project generated by the Project Configurator Wizard. The second part demostrates how to integrate the ADC peripheral with the use of example code provided in our GitHub repository. The third part outlines the addition of a real-time operating system and the changes in project model with this addition. The fourth and final portion of this lab adds Bluetooth to the project.

# **1** Prerequisites

For this lab you will need the following:

- EFR32MG12 Thunderboard Sense 2 (SLTB004A)
- Micro-USB to USB Type-A cable (Not included with Thunderboard)
- Simplicity Studio v5
  - o Bluetooth SDK 3.1.0.0 or later
  - o Gecko SDK Suite 3.1.0 or later
- <u>BLE Scanner</u> mobile app (or other generic Bluetooth scanning application for Bluetooth demo)
- Accompanying lab source code provided in compressed zip <u>https://www.dropbox.com/s/4xbz4uzxq9l63nc/v5\_ProjectConfigu-</u> rator\_Lab.zip?dl=0



# 2 Creating a bare-metal project

# 2.1 Create an empty project using Silicon Labs Project Wizard

Run the installation of Simplicity Studio v5 and navigate to the Simplicity IDE perspective.

From the project explorer window, click on the link "Create a project"

C+ 4 4 8 + 5 + 4 + 9. +	Bun Window Help	🕆 患 🏫 Welcome \ominus Recent 田 Tools 古 Install 🌣 Preferences	B 🕫 Launcher 🚯 Simplicity IDE
Droject Explorer	8878 *		
There are no projects in your workspace	£.		
To add a project			
El Create a project			
às Import projects			
🖬 Debug Adapters 💷 📴 Outline			
	3××\$•		
	815)		
		🐮 Problems 🛷 Search 🔅 Call Hierarchy 🖾 Console 💠	🗟 🖬 🗷 🗮 🕶 🗂
		Adapter Pack Console	
		<pre>boardRevision(0)=000 boardSerial(0)=184606271</pre>	
		boardDate[0]-2018/11/15	
		inferPart[0]-yes	
		The sector - yes	
		left: 1366:0105@015:001[000440137971]	

A "New Project" window will pop up. Select the Silicon Labs Project Wizard and click "Finish".

A "New Project Wizard" window will now display. In the "Target Boards" text entry field, enter the board used for this demo – BRD4166A (If the Thunderboard Sense 2 is already connected to the PC and is the only development kit detected by Simplicity Studio, this field will auto-populate).

Verify that the latest Gecko SDK 3.1.0 or later is selected in the SDK drop-down menu.

Click "Next" to move on to the next configuration window.

	New Project Wizard – 🗆 X
S New Project — — X	Target, SDK, and Toolchain Selection
Select a wizard Create a C/C++/Assembly project targeting Silicon Labs MCUs.	Select the target board, device, SDK, and IDE/toolchain to use for the project.    Target, SDK  Configuration Configuration
Wizards: type filter text ×	Target Boards: Search or Select  Thunderboard Sense 2 (BRD4166A)
<ul> <li>Silicon Labs Project Wizard</li> <li>General</li> <li>C/C++</li> <li>Simplicity Studio</li> </ul>	Target Device: Search or Select
Show All Wizards.	IDE / Toolchain: Select BE / Toolchain Simplicity IDE / GNU ARM v7.2.1
Image: Second	
	CANCEL BACK NEXT FINISH

Under "Example Project Selection", click "Empty C Project". This example project is typically located at the top of the list; filtering by the "Platform" Technology Type will narrow the selection options. Depending on the various packages installed with Simplicity Studio, there may be a different list of available examples than captured in the provided screenshot.

In the "Project Configuration" window:

- Enter a project name for the purposes of this lab, "demo" is used as the project name
- Use the default project location, or specify a new location
- Make sure the "Link SDK and copy project sources" radio button is selected
- Click "Finish"

The Simplicity Studio Project Wizard will begin to auto-generate the necessary project structure and files

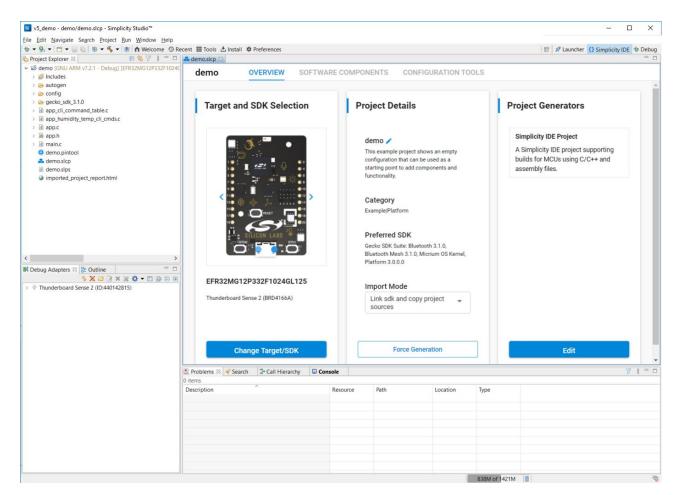
New Project Wizard		- <b>D</b> X	
Example Project Selection			
Select the project template to open in Simp	plicity IDE.		
Contract SDK	Examples	Configuration	
	58 resources found		
Filter on keywords	Empty C Project		
Technology Type O Clear Filter	This example project shows an empty configuration that can be used as a starting point to add components and functionality.	a	
Bluetooth (11)			
Bootloader (9)		New Project Wizard	- 🗆 X
Platform (6)	sleepy-demo-mtd This is a sample application to demonstrate Sleepy End Device whaviour	Project Configuration	
Proprietary (10)	on an MTD using the EFR32's low power EM2 mode.	Select the project name and location.	
Thread (8)		C Target, SDK	Configuration
Zigbee (14)	sleepy-demo-ftd		•
	This is a sample application to start and form a Thread network on an FT	•	
	for the sleepy-demo.	Project name: demo	
	ot-rcp This is a simple OpenThread RCP application.	✓ Use default location	
		Location: C:\Users\augoodwi\SimplicityStudio\v5_demo\demo	BROWSE
	ot-ncp-mtd		
	This is a simple OpenThread Minimal Thread Device NCP application.	With project files:	
		C Link to sources	
CANCEL	BACK	Link sdk and copy project sources	
		Copy contents	
			l l
		CANCEL BACK	NEXT FINISH

#### 2.2 Explore the new project structure

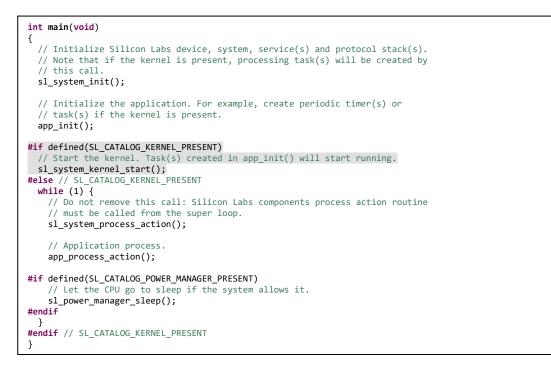
After the Simplicity Studio Project Wizard completes, the new project directory structure and source will display in the "Project Explorer" window located on the left side in the "Simplicity IDE" perspective.

The \*.slcp file will automatically open in the main window; three tabs are now available when viewing the project \*.slcp file:

- Overview Displays information about the target device, SDK selection, and some project details
- Software Components Project Configurator for simple integration of software tools/drivers for easy intergration of peripherals
- Configuration Tools Quick access to additional configuration tools, depending on installed components (Pin Tool, e.g.)



Exploring main.c gives a glimse of the new high-level project model utilized with Simplicity Studio v5's Gecko Platform and Project Configurator.



The main() function is simple: top-level initialization routines and a main while loop or "super loop" which utilizes process action routines for a bare-metal application. With the presence of a real-time operating system, the kernel is started and a task based action model is used.

In the high-level initialization there are two initialization function calls:

- sl\_system\_init() within this function, Silicon Labs software components are initialized in a particular order (discussed in training)
  - Silicon Labs (sl\_\*) initialization functions are located in sl\_event\_handler.c, and include errata fixes, default pin states, initializing interrupt vector, board specific settings, DCDC/energy mode settings, and clock settings, etc.
  - sl\_internal\_app\_init() will also hold initialization code for software components. Currently blank, as we have not yet installed any software components
- app\_init() located in app.c this is where the user can add initialization routines specific to their particular project and application. This function will be modified periodically throughout the demonstration.

In the main while loop (super loop), two process action routines are called:

- sl\_system\_process\_action() Silicon Labs components process action routine
  - sl\_internal\_app\_process\_action() located in sl\_event\_handler.c; again, currently blank but will hold software component specific process actions
- app\_process\_action() located in app.c; application specific process action routines

Because this is an empty project, most initialization routines are currently empty. Adding certain components using Project Configurator will autogenerate new code in the appropriate inits and process action routines.

### 2.3 Adding Software Components

For this next step, the Command Line Interface (CLI) will be added, which in this case will also use the USART and Virtual COM port onboard the Thunderboard Sense 2. The Power Manager software component is also added to manage energy modes for the application.

Go back to viewing "demo.slcp" in the main window, or re-open this file from the "Project Explorer" window if it was closed.

In the main window, navigate to the "Software Components" tab.

From here, install the following Software Components:

- Services > IO Stream > IO Stream: USART
  - Create Component Instance name: vcom
- Services > CLI > CLI: Command Line Interface
  - Create Component Instance name: vcom
- Services > Power Manager

🗧 v5_demo - demo/demo.slcp - Simplicity Studio™				– o ×		
Eile Edit Navigate Search Project Bun Window Help						
🕸 🕶 💁 🕶 🛅 🕶 🔛 🔞 👘 🐨 🐇 🖛 🔛 🐨 🕲 Recent 🚦		rences		😰 🖉 Launcher 🚯 Simplicity IDE		
Project Explorer 😫 📃 🗎 🖶 🗖	🚢 demo.slcp 🖾			- 0		
	demo	demo OVERVIEW SOFTWARE COMPONENTS CONFIGURATION TOOLS				
> config > c	<b>Y</b> Filter : Configur	rable Components 🔲 Installed Componen	Components Installed by You	Q. Search keywords, component's name		
App.h     B main.c     demo.pintool     demo.stop	IO Stream		IO Stream: USART	Install		
demo.slps		: LEUART Core		Add component to project		
	IO Stream		Description 10 Stream over Universal Synchronous Asynch	ironous Receiver Transceiver (USART)		
		: Retarget STDIO : STDLIB Configuration	communication protocol.			
Cebug Adapters 🛛 📴 Outline 👘 🗖	IO Stream		Quality			
S 🗶 😂 🖉 ¥ 🗶 🎝 ▾ 🗇 📼 🖻 🖻 ♦ Thunderboard Sense 2 (ID:440142815)	IO Stream: USART		Open in Browser			
		SART Core		•		
	Close 🗙			View Dependencies		
		Call Hierarchy 🕮 📮 Console		0 0 X D C I - 0 C I - 0		
Create A Component Ins This component allows multiple instances. this instance in the field below. The name construct #defines in the source files of	Create a name for will be used to	tct a function or method and select the 'Open Call Hier	archy' menu option.			
INSTANCE NAME			509M of 1-	410M		
vcom						
Names must adhere to both C and POSIX filename rules						
Recommended Instance Names						
Cancel Done	←					

While still viewing "demo.slcp" in the main menu, navigate again to the "Software Components" tab and to the IO Stream: USART (Services > IO Stream > IO Stream: USART).

Click on the "Configure" button in the right section of the main window, or the gear icon in the Software Components tree view

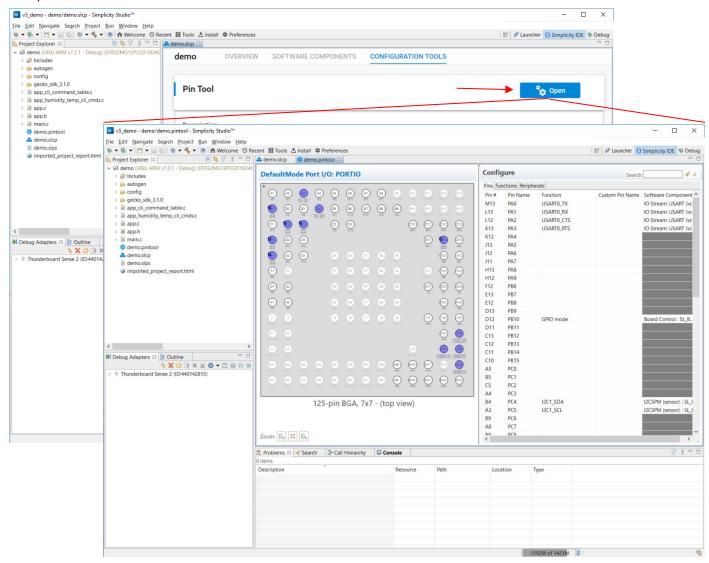
Be Edit Navigate Search Project Bun Window Help ✿ ♥ 💁 ♥ 📑 ♥ 🔜 🐑   🕲 ♥ % ♥ 😻 🛉 Welcome ⊙ Recent III Tools	📩 Install 🕸 Preferences			😰 🖉 Launcher 🚺 Simp			
	demosicp 🗈						
<ul> <li>✓ demo (GNU ARM v7.2.1 - Debug) [EFR32MG12P332F1024GL125 - Gecko SI</li> <li>&gt; includes</li> <li>&gt; in autogen</li> <li>&gt; includes</li> </ul>	demo overview SOPTWAR						
> 😂 gecko_sdk_3.1.0 > 👔 app.c > 🗟 app.h		installed components compo		Q Search keywords, component's nam			
Imain.c     demo.pintool     demo.sicp	► Micrium OS	1	0 Stream: USART	Configure			
demo.sips	► Platform						
	► RTOS		Description IO Stream over Universal Synchronous Asynchron	ous Receiver Transceiver (USART) communication protocol.			
	▼ Services		Quality				
	CLI     File System						
	▶ 10		Open in Browser				
Debug Adapters II      De Outline     S      S      S      S      S      C     C			IO Stream USART				
	⊘ IO Stream	-	Description				
			Description				
				- 0			
	to Stream: USARI (vcom) ∷ IO Stream: USART (vcom)			View Source X			
	B 0 Stream: USARI (voon) :: IO Stream: USART (vcom)						
	10 Stream: USART (vcom) ::	Parity mode to use	Number of stop bits to use.				
	IB to Stream: USART (scon) :: IO Stream: USART (vcom)	Parity mode to use No Parity *	Number of stop bits to use. 1 stop bits	View Source X			
	IO Stream: USART (vcom) USART settings Baud rate 115200 Receive buffer size			Hardware flow control			
	ID Stream: USART (vcom)	No Parity 👻	1 stop bits 💌	Hardware flow control			
	IO Stream: USART (vcom) USART settings Baud rate 115200 Receive buffer size	No Parity +	1 stop bits 💌	Hardware flow control			
	B No Stream: USART (score)  IO Stream: USART (vcore) USART settings Baud rate      115200 Receive buffer size     2     SL_IOSTREAM_USART_VCOM	No Parity  Convert In to VIn	1 stop bits   Restrict the energy mode to allow the re	Hardware flow control			
	Bit Stream: USART (scom)      IO Stream: USART (vcom)      USART settings     Baud rate     115200     Receive buffer size     2     32	No Parity  Convert In to VIn	1 stop bits 💌	Hardware flow control			
		Convert Into I/Vn	1 stop bits   Restrict the energy mode to allow the re  CS	Hardware flow control None •			

Within the USART configuration window USART settings and pin configurations can be modified. The pin configurations defaut to the pins specified by the hardware selected when createing the new project (refer to the schematic for Thunderboard Sense 2).

For custom configurations, users will want to modify these settings based on their custom PCB to reflect specific pin routing.

From the USART configuration window, the generated source code can also be viewed by clicking the "View Source" button in the upper right corner.

Changes to pin allocation/routing can also be made in the Pin Tool, available under the "Configuration Tools" tab when viewing "demo.slcp".



Changes made in the USART configuration GUI, source code, or Pin Tool propagate to the other tools when saved. Thus all three configuration tools are linked. Similar configuration tools/options are available for many of the other software components. The same GUI/source/pin tool relationship applies, and configuration changes can be made to any of the three tools.

No changes need to be made in this example, as the default settings apply.

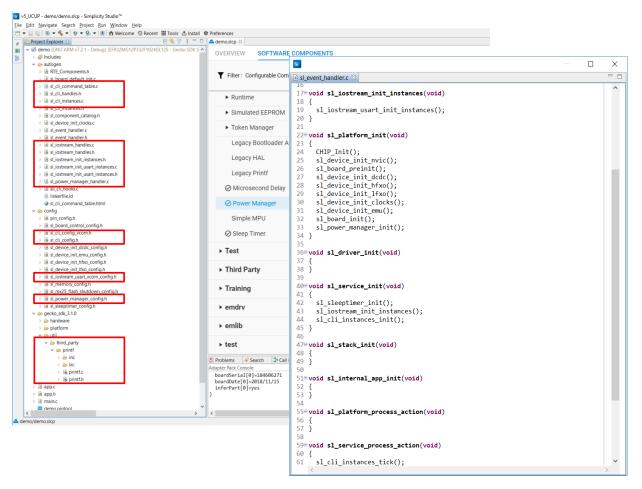
### 2.3.1 Explore project additions

In the "Project Explorer" window, expand the autogen and config folders. New source files have been added to the project which correspond to the additional software components added through Project Configurator.

Additional initializations are also added within sl\_event\_handler.c.

Third party *printf* functions have also been added (a dependency for the CLI interface).

There is also a new process in sl\_service\_process\_action for the CLI interface.



Power Manager has now been added to the project, and the source file sl\_power\_manager\_handler.c is now added to the project under the autogen directory.

sl\_power\_manager\_is\_ok\_to\_sleep() and sl\_power\_manager\_sleep\_on\_isr\_exit() callback functions are implemented which allow software to cancel going to sleep in case of a last-minute event and validate whether the MCU can return to sleep after processing an interrupt when the system was sleeping. The CLI interface and IO Stream software instances have hooks into these functions to properly manage the lowest possible energy mode.

Application code can also be used to overwrite weak functions app\_is\_ok\_to\_sleep() and app\_sleep\_on\_isr\_exit() so that application code can also perform these tasks.

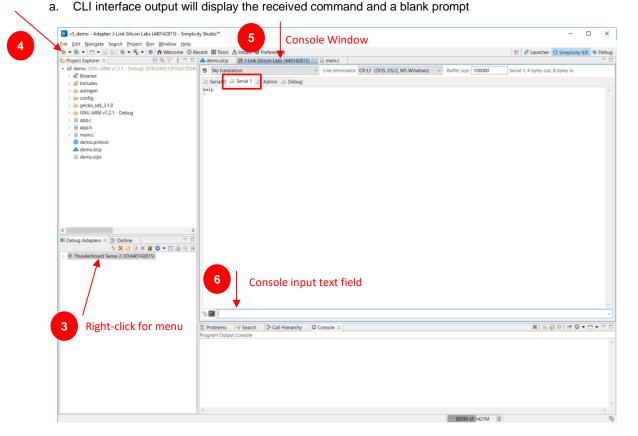
Power manager manages energy modes for the application:

- Ensures that when the system enters sleep, it does so at the lowest energy mode possible. (For this example, because we are using the UART, the lowest energy mode is set to EM1)
- Lowest energy mode allowed is selected based on requirements set by different modules (drivers, stacks, application, etc.)
- Also applies some control on the HFXO (SYXO)
- No support for EM4.

## 2.4 Empty project with CLI interface

At this point, the project can be built and run, but won't really do anything besides show the default CLI interface.

- 1. Connect the Thunderboard Sense 2 to your PC using a micro-USB cable (if you haven't done so already)
- 2. When the device is connected to your PC, you should see it listed in the Debug Adapters window in Simplicity Studio
- 3. In the Debug Adapters window, right-click on the device to display the options window and select "Launch Console..."
- Build the project and flash the project to connected Thunderboard Sense 2 using the "Debug" button in the top menu.
   a. Simplicity Studio will automatically shift views to the Debug perspective and halt at the MCU at the start of main()
  - b. Press F8 or click on the "Resume" button where to resume program execution
- 5. In the main window, navigate to the console window and select the "Serial 1" tab; this corresponds to the VCOM port
- 6. In the console input text field type "help" and press Enter



Build the project and flash to connected Thunderboard Sense 2 to display the bare CLI interface (help is the only command currently available).

#### 2.5 Add Sensor Components

In this step the onboard relative humidity and temperature sensor is added to the project. The onboard sensor uses I2C for communication, therefore the addition of an I2C driver is also required. Disconnect from the Thunderboard Sense 2 and return to the Simplicity IDE perspective. In the main window, navigate back to the "demo.slcp" or re-open the file by double-clicking in Project Explorer

Again, navigate to the "Software Components" tab.

From here, install the following Software Components:

- Platform > Driver > I2CSPM
  - Create Component Instance name: sensor
- Bluetooth > Sensor > Relative Humidity and Temperature sensor

NOTE: occasionally the **Assert** software component does not get pulled in, though it is a dependency of this component; Navigate to Bluetooth > Utility > Assert in the Software Component tree view and install if this is not installed with the last step.

• 9. • 1 1 • 1 1 1 1 1 • • • • • • • • • •	Weicome ③ Recent III Tools & Install     B % 7   ** □				8 🕫 Launcher () Simplicity
	EFR32MG12P332F1024GL125 - Gecko SDK Suite:	demo OVERVIEW SOFTWARE COMPO		TION TOOLS	
<ul> <li>gil Includes</li> <li>autogen</li> </ul>		dento overview software compo	CONFIGURA	10010005	
config gecko_sdk_3.1.0 g app.c		Filter : Configurable Components         Installed C	Components Co	omponents Installed by You	Q. Search keywords, component's name
<ul> <li>Bit pañ</li> <li>Bit paña</li> <li>O demo partod</li> <li>A demo stop</li> <li>I demo stops</li> </ul>		GLIB Graphics Library     GLIB driver for SHARP Memory LCD     GPIOINT		I2CSPM     Description	
		Generic Button API Generic LED API		This is an I2C Simple Polled Master driver the blocks while waiting for the transfer to comp	at supports master mode with a single bus-master only. It plete, polling for completion in ENG.
		12CSPM	0		several named instances can be created. For each instance a
		I2CSPM Core			tialization struct set according to this configuration. These in a generated header file, sl_i2cspm_instances.h.
		PWM	•		m Init component is included in a project, the driver using the instance configurations, during the sl_system_init()
ebug Adagters (H ) 📴 Ouffine P Trunderboard Sense 2 (ID4 <sup>2</sup>	This component allow: this instance in the construct #defines INSTANCE NAMI sensor	_	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	call in main.c.	= 122590 Core component, which is the implementation of the $\label{eq:view} View Dependencies $$ \psi \in [N_i] [S_i < T_i] + \phi < 1$$ $$
	Names must adhere to both				509M of 1410M
	<ul> <li>Recommend</li> </ul>	ed Instance Names			
	sensor_env				
	sensor_hall				
	sensor_gas				
	sensor				
	с	ancel Done			

#### 2.6 Enable onboard sensor

An additional step is required to toggle the enable pin for the onboard sensor.

Again, from the "Software Components" tab, navigate to Platfor > Board > Board Control and click the gear to open the Board Control configuration window.

In the "General" section, click the slider below "Enable Relative Humidity and Temperature sensor".

The slider will change color to blue and a notification will pop up (green) that the changes have been saved.

■ v5_UCUP - demo/.uceditor/.demo.board_control.cedit - Simplicity Studio™ Elie Edit Navigate Search Project Bun Window Help					- 0 ×
🗂 🕶 🗔 🐚 🕷 🕶 🐐 🗣 🗣 🗣 👘 🏦 Welcome 💿 Recent 🏢 Tools 📩 Install 🗄	Preferences			🖹 🖈 Launcher 🚺 Simplicit	ty IDE 🌵 Debug 🤳 Network Analyzer
v G demo [GNU ARM v7.2.1 - Debug] [EFR32MG12P332F1024GL125 - Gecko SDK Suite:     S & Binaries     S Includes	a demo.slcp Ø J-Link Silicon Labs (440142815) @ main.c ▲ Board Control		View Source		
> 😂 autogen 🔰	»		✓ Saved Changes		×
a)         a)         b)           b)         b)         b)         b)         b)           b)         b)         b)         b)         b)           b)         b)         b)         b)         b)           b)         b)         b)         b)         b)         b)           b)         b)         b)         b)         b)         b)         b)           b)	General Enable Relative Humidity and Temperature sensor	Enable Hall Effect sensor	Enable Barometric Pressure sensor	Enable Light sensor	
	Enable Air Quality sensor	Enable Inertial Measurement Unit	Enable Microphone	Disable SPI Flash	
C Second Adaptes :: Source	SL_BOARD_ENABLE_SENSOR_GA Selected Module PF14	45			
	Problems Search It Call Hierarchy Console 8				x 🛼 🖬 🖻 🚽 🛛 🕶 😁 🗆
	rogram Output Console				
				626M of 1633M	>

The default GPIO configuration associated with the Thunderboard Sense 2 hardware selection will route the appropriate pin for enabling the sensor.

# 2.7 Adding Application Specific Code

Download the accompanying lab source code provided in compressed zip – <u>https://www.dropbox.com/s/4xbz4uzxq9l63nc/v5 Pro-jectConfigurator Lab.zip?dl=0</u>

Extract the files in the compressed zip to a known location and open the "demo1" folder. Take note of the extraction location. The remaining files will be used through the lab.

Drag-and-drop the files within this folder into the demo project's top directory in Simplicity Studio's "Project Explorer" window.

CUP - demo/demo.slcp - Simplicity Studio™ it Navigate Search Project Bun Window Help					- 0
🔟 💿 • 🍕 • 💁 • 🐏 • 💼 🏠 Welcome 🛈 Recent 🏢 Tools 🛓 In	tall 🌣 Preferences			🔡 🖉 Launcher 🚺 Simplicit	VIDE Debug J. Network
	a demo.sicp 💠 😽 J-Link Silicon Labs (440142815) 📄 main.c				
emo (GNU ARM v7.2.1 - Debug) [EFR32MG12P332F1024GL125 - Gecko SDK Sur Binaries	OVERVIEW SOFTWARE COMPONENTS				
Includes autogen config	Y Filter : Configurable Components Installed Components	Components Installed by You		Q. Search key	words, component's nam
gecko_sdk_3.1.0 GNU ARM v7.2.1 - Debug	📕   🖓 📕 🖛   demo1			- 0	×
app.c app.h main.c	File Home Share View				~ () Configure
demo.pintool demo.sicp demo.sips	$\leftarrow$ $\rightarrow$ $\checkmark$ $\uparrow$ $\blacksquare$ $\rightarrow$ This PC $\rightarrow$ Documents $\rightarrow$ v5_F	rojectConfigurator_Lab → de	emo1	✓ Ö Search de	A.
	✓ Name	Date modified	Туре	Size	aly. It
drag-and-drop	🖂 🖻 app.c	11/17/2020 1:21 PM	C Source	5 KB	
<b>3</b> .	🗹 🛍 app.h	11/17/2020 1:20 PM	C/C++ Header	2 KB	instance a tion. These
	app_cli_command_table.c	11/17/2020 1:22 PM	C Source	4 KB	×.
	app_humidity_temp_cli_cmds.c	11/17/2020 1:20 PM	C Source	2 KB	ver
					ystem_init(
g Adapters III 📴 Outline 🔰 🛸 😂 🕢 🕷 💥 🗢 🗂 🔯 🗁 🐨 hunderboard Sense 2 (ID:440142815)	4 items 4 items selected 10.7 KB			1420	ion of the
S File Operation	×	Quality PRODUCTION	f.		
Select how files should be imported in	to the project:	Si Question			>
<u>Copy files</u> Uink to files		demo/app.c ex	ists. Do you wish to overwrite?		
			↓		
Create link locations relative	e to: PROJECT_LOC ~		Overwrite Overwrite A	II Do <u>n</u> 't Overwrite	Cancel
Configure Drag and Drop Settings					
(?)	OK Cancel				
~					

In the "File Operation" window, select the "Copy Files" radio button and press OK.

When prompted about overwriting app.c/app.h, select "Overwrite All".

The following files should be added to the project:

- Updated app.c/app.h
  - Initialization
    - Initializes RHT sensor
    - Configures sleeptimer to wake every 2 seconds; ISR sets measurement flag
    - Adds CLI commands to CLI instance
  - On wake, main()'s super loop calls app\_process\_action():
    - Checks for measurement flag
    - Reads relative humidity and temperature
    - Stores values in global variable
- NEW app\_cli\_command\_table.c
  - CLI command structs for sensor readout over VCOM
- NEW app\_humidity\_temp\_cli\_cmds.c
  - Function calls by the CLI interface to print command responses

The provided app.c file utilizes **#if !defined(SL\_CATALOG\_KERNEL\_PRESENT)** in order to include different portions of the code when running with/without an RTOS kernel.

For this first portion, the code is utilizing the programming model with process actions. For the third part where software component FreeRTOS is added to the project, code will utilize the RTOS kernel and task actions.

#### 2.8 Build and test

The project can now be built and run to demonstrate the relative humidity and temperature sensor readout. Additional commands are now available through the CLI interface.

- 1. Connect the Thunderboard Sense 2 to your PC using a micro-USB cable (if you haven't done so already)
- 2. When the device is connected to your PC, you should see it listed in the Debug Adapters window in Simplicity Studio
- 3. In the Debug Adapters window, right-click on the device to display the options window and select "Launch Console..."
- 4. Build the project and flash the project to connected Thunderboard Sense 2 using the "Debug" button in the top menu. a. Simplicity Studio will automatically shift views to the Debug perspective and halt at the MCU at the start of main()
  - b. Press F8 or click on the "Resume" button 🏴 to resume program execution
- 5. In the main window, navigate to the console window and select the "Serial 1" tab; this corresponds to the VCOM port
- 6. The following commands are now availabe in the console window:
  - a. help general help; returns other command syntax
    - b. **help sensor** help using the sensor command
  - c. sensor temp [unit] (C or F) returns temperature reading
  - d. sensor humidity returns humidity reading
  - e. sensor all [unit] (C or F) returns temperature and humidity
- 7. In the console input text field type "help" and press Enter
  - a. CLI interface output will display the received command and "sensor"
  - b. Type "help sensor" to display CLI command information pertaining to the onboard sensor readout
  - c. Type "sensor all F" to display the temperature in degrees Farenheit and humidity as a percentage
  - d. Try any of the other commands

~

🎄 ▼ 🥵 ▼ 📑 ▼ 📾 🖏   🕹 ▼ 🗞 ▼		emo.slcp & J-Link Silicon Labs (440142815) 🔅 🕜 main.c	🗃 🖈 Launcher 🚯 Simplicity
<ul> <li>Gene (SNU AMM v7.2.1 - Debug)</li> <li>G demo (SNU AMM v7.2.1 - Debug)</li> <li>G indudes</li> <li>autogen</li> <li>activation</li> <li>activ</li></ul>	(EFR32MG12P332F1024C	No transition         ✓         Line terminator:         CR-LF (DOS, OS/2, MS Windows)         ✓         Buffer size:         10000           Seria         0         ≦ Serial 1         _         Admin         ≦ Debug	9 Serial 1, 35 bytes out, 507 bytes in
< Bi Debug Adapters 32 Bi Dutline S X I III IIII R Thunderboard Sense 2 (ID:4014)	> 2 C = C = C 815) (Simplicity Debug;	6 Console input text field	
/			
3 Right-click f	or menu	oblems 🔗 Search 🔄 Call Hierarchy 🖸 Console 🕮	× 🛼 🖬 🕬 🖻 🛛

Disconnect from the Thunderboard Sense 2 debug connection and return to the Simplicity IDE perspective before moving on to the next part of the lab.

# 3 Adding the ADC peripheral and application code

For this portion of the lab we will be adding the ADC peripheral and using example code provided on our GitHub repository to populate the necessary functions within our application to take a software triggered measurement of the DCDC output voltage.

# 3.1 Add the ADC software component via Project Configurator

In Simplicity Studio, back in the Simplicity IDE perspective, navigate to the demo.slcp in the main viewing window and open the "Software Components" tab.

From here, install the following Software Components:

• Platform > Peripheral > ADC

This will add the necessary source files from the emlib library which is required for running code provided through our Peripheral Examples repository on GitHub.

## 3.2 Adding application specific code

From the previously extracted compressed zip in step 2.7, open the "demo2" folder.

Drag-and-drop the files within this folder into the demo project's top directory in Simplicity Studio's "Project Explorer" window.

u5_UCUP - demo/demo.slcp - Simplicity Studio**					-	ø ×
Eile Edit Navigate Search Project Bun Window Help				A CALL NAME OF COMPANY OF CALL		
□         •         •         •         •         ●         •         ●	tall Ø Preferences           a demo.sicp 12         Ø J-Link Silicon Labs (440142815) <ul></ul>			🔝 🖉 Launcher 🚺 Simplicity	IDE P Debug J. Net	twork Analyzer
✓ demo (GNU ARM v7.2.1 - Debug) [EFR32MG12P332F1024GL125 - Gecko SDK Sur > c Binaries						
> gi Includes > ge autogen > ge config	Filter : Configurable Components         Installed Components	Components Installed by You		Q Search key	words, component's	name
> 20 gecko sdl(3.1.0 > 20 GNU ARM V7.2.1 - Debug > (8 app.c	📕   🛃 📕 🚽   demo2			-		
> 🗟 app.h > 🗟 main.c	File Home Share View				~ 🕐	jure
demo.pintool     demo.sicp     demo.sips	$\leftarrow$ $\rightarrow$ $\checkmark$ $\uparrow$ $\blacksquare$ $\rightarrow$ This PC $\rightarrow$ Documents $\rightarrow$ v5	ProjectConfigurator_Lab > der	no2	✓ <sup>™</sup> Searce	ch de 🔎	
	□ Name	Date modified	Туре	Size		
drag-and-drop	app_adc_cli_cmds.c	11/12/2020 2:19 PM	C Source	2 KE		
	🗹 🖻 app_adc_demo.c	11/12/2020 4:31 PM	C Source	4 KE	4	
	✓ bi app_adc_demo.h	11/12/2020 4:28 PM	C/C++ Header	3 KE	1	
	app_cli_command_table.c	11/12/2020 4:33 PM	C Source	5 KE		(t ()
C Debug Adapters III (2) Outline	4 items 4 items selected 11.9 KB					58
S File Operation	×	Quality PRODUCTION				
Select how files should be imported in	to the project:	Si Question				×
© Copy files ○ Link to files		demo/app_cli_comm	and_table.c exists. Do you w	ish to overwrite?		
Create link locations relative	to: PROJECT_LOC ~	Qve	erwrite Overwrite <u>A</u> ll	Do <u>n</u> 't Overwrite	Cancel	
Configure Drag and Drop Settings		_				1
?	OK Cancel					

In the "File Operation" window, select the "Copy Files" radio button and press OK.

When prompted about overwriting app\_cli\_command\_table.c select "Overwrite".

The following files should be added to the project:

- UPDATED app\_cli\_command\_table.c
  - Adds CLI command structs for ADC readout over VCOM
- app\_adc\_demo.c/.h
  - Skeleton functions for ADC -> will be utilizing peripheral example code from GitHub to populate routines

- NEW app\_adc\_cli\_cmds.c
  - Function calls by the CLI interface to print command responses

# 3.3 Implement source for ADC functions

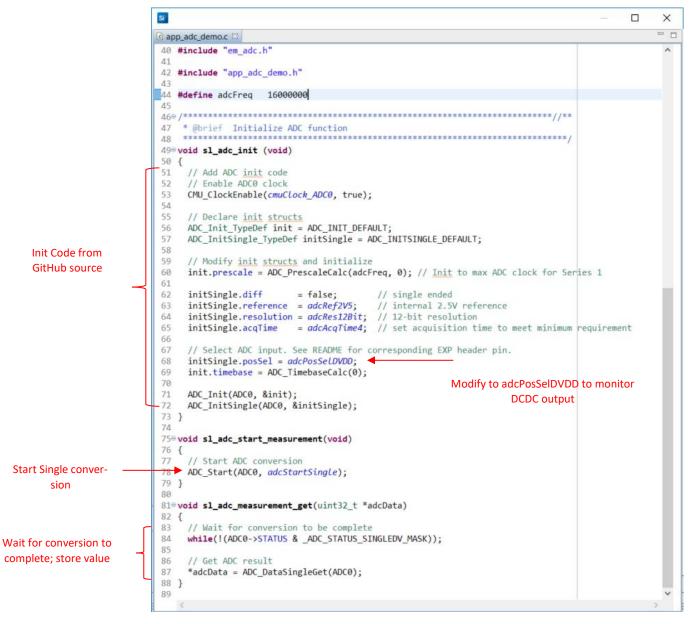
In the "Project Explorer" window, open app\_adc\_demo.c. There are three function definitions: sl\_adc\_int(), sl\_adc\_start\_measure(), and sl\_adc\_measurement\_get().

The source from GitHub ADC Single Polled example is used to populate the three function definitions, with some slight modification. This source is located at <a href="https://github.com/SiliconLabs/peripheral\_examples/blob/master/series1/adc/adc\_single\_polled/src/main\_s1.c">https://github.com/SiliconLabs/peripheral\_examples/blob/master/series1/adc/adc\_single\_polled/src/main\_s1.c</a> and is also provided in the previously downloaded zip file under the folder "GitHub ADC Single Polled Src".

Copy the initialization code from the GitHub source's initADC() function into sl\_adc\_init() in app\_adc\_demo.c. Modify the positive input selection (initSingle.posSel) from APORT 2 channel 9 to DVDD (adcPosSelDVDD) to measure the DCDC output, which is regulating at 1.8V.

Copy or add the ADC single conversion start function to sl\_adc\_start\_measurement().

Copy or add the code polling the ADC status and then retrieving the single conversion to sl\_adc\_measurement\_get().



The ADC measurement is split in this example into two functions, one to kick off a conversion, and another to retrieve the conversion data, however the two could be combined. The benefit of splitting this up is while the ADC conversion is taking place, other code can be executing (relative humidity and temperature sensor measurement over I2c, e.g.).

# 3.4 Modifiy app.c init and process action calls

The ADC measurement and implementation need to be added to the application. In the "Project Explorer" window, open app.c.

At the top of the file, include the new app\_adc\_demo.h header file - suggested insertion on line 33 -> #include "app\_adc\_demo.h"

Add global uint32 variable "adcData" to store ADC result - suggested insertion on line 62 -> uint32\_t adcData = 0;

In app\_init(), add the ADC initialization function sl\_adc\_init() - suggested insertion on line 83.

Add ADC measurement calls to the process action

- sl\_adc\_start\_measurement() prior to RHT sensor readout to start ADC conversion insertion on line 122
- sl\_adc\_measurement\_get(&adcData) after RHT sensor readout to retrieve ADC conversion insertion on line 124

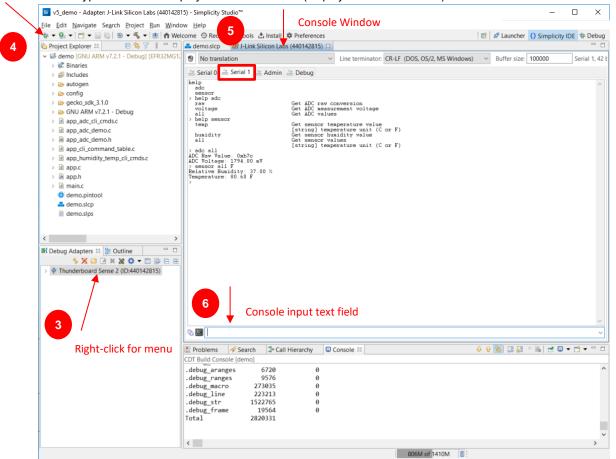
Solution v5_demo - demo/app.c - Simplicity Studio™			- 🗆 X
Eile Edit Source Refactor Navigate Search Project Bun	Window Help		
to • 9. • 1 • 1 • 1 • 1 • 1 • • • • • • • • •	ecent 🏢 Tools 📩 Install 🌞 Preferences	😰 🖉 Launcher 🚺	Simplicity IDE * Debug
Project Explorer 🛛 📄 🛱 🍞 🛔 🗂 🗖	demosicp C app.c B		- 0
demo [GNU ARM v7.2.1 - Debug] [EFR32MG12P332F10240]	769 /************************************		^
> 🖑 Binaries	77 * Initialize application.		
> 🗊 Includes	78 ************************************		
> 😓 autogen	79® void app_init(void)		
> 🤄 config	80 { 81 sl sensor rht init():		
> 🧁 gecko_sdk_3.1.0	<pre>81 sl_sensor_rht_init(); 82</pre>		
> 🗁 GNU ARM v7.2.1 - Debug	<pre>83 sl_adc_init();</pre>		
> app_adc_cli_cmds.c	84 #if !defined(SL_CATALOG_KERNEL_PRESENT)		
> app_adc_demo.c	85 /* Set up periodic measurement timer. */		
> 🖻 app_adc_demo.h	<pre>86 sl_sleeptimer_start_periodic_timer_ms(&amp;measurement_timer, MEASUREMENT_INTERVAL_MS, meas 87 #else</pre>	<pre>surement_callback, NULL, 0, 0);</pre>	
> app_cli_command_table.c	87 #else 88 osThreadAttr_t attr;		
app_humidity_temp_cli_cmds.c	89		
> 🗟 app.c	<pre>90 attr.name = "si7013 app task";</pre>		
> 🖻 app.h	91 attr.priority = 25;		
> 🗟 main.c	<pre>92 attr.stack_mem = task_stack;</pre>		
o demo.pintool	<pre>93 attr.stack_size = STACK_SIZE; 94 attr.cb mem = thread cb;</pre>		
demo.slcp	95 attr.cb_size = osThreadCbSize;		
🗟 demo.slps	96 attr.attr_bits = 0u;		
	<pre>97 attr.tz_module = 0u;</pre>		
	98		
	<pre>99 thread_id = osThreadNew(&amp;task, NULL, &amp;attr);</pre>		
	<pre>100 EFM_ASSERT(thread_id != NULL); 101 #endif</pre>		
	102		
< >	103 // Remove unused autogen CLI command group		
	<pre>104 sl_cli_command_remove_command_group(sl_cli_vcom_handle, &amp;sl_cli_vcom_command_group);</pre>		
ar beographers of D. Satisfie	105		
\$ X ≅ 2 × X \$ - □ ■ C B	<pre>106 // Add application CLI commands 107 sl_cli_command_add_command_group(sl_cli_vcom_handle, &amp;sl_cli_app_command_group);</pre>		
Thunderboard Sense 2 (ID:440142815)	107 Si_cii_command_auu_command_group(si_cii_vcom_nanule, &si_cii_app_command_group), 108 }		
	109		
	110 #if !defined(SL_CATALOG_KERNEL_PRESENT)		
	1110/**********************************		
	112 * App ticking function.		
	113 114@void app_process_action(void)		
	115 {		
	116 CORE_DECLARE_IRQ_STATE;		
	117		
	118 CORE_ENTER_CRITICAL();		
	<pre>119 if (measurement_flag) { 120 measurement flag = false;</pre>		
	120 measurement_riag = raise, 121 CORE_EXIT_CRITICAL();		
· · · · · · · · · · · · · · · · · · ·	<pre>122 sl_adc_start_measurement();</pre>		
	<pre>123 sl_sensor_rht_get(&amp;rhData, &amp;tempData);</pre>		
<b>→</b>	<pre>124 sl_adc_measurement_get(&amp;adcData);</pre>		
	125 } else {		
	126 CORE EXIT CRITICAL();		
	Deltar Scatter Scattered	w110.00	i d 🛛 🕶 👌 🕶 🗖
	Problems Search Call Hierarchy Console 2	A   12 🔠	
	Program Output Console		123
			0
	Writable Smart Insert 33 : 1 [25] 690M (	of 1421M	

While adding the ADC measurement calls, this is an opportunity to also add the calls to the kernel task, which currently is not compiled with the code, but will be necessary in the next demo. Use the same steps to straddle the RHT sensor readout function in the kernel task with the two ADC measurement calls (insertions on lines 144 and 146).

## 3.5 Build and test

The project can now be built and run to demonstrate the relative humidity and temperature sensor readout along with DCDC output voltage readout via the ADC peripheral. Additional commands are now available through the CLI interface.

- 1. Connect the Thunderboard Sense 2 to your PC using a micro-USB cable (the Thunderboard Sense 2 may require the USB cable to be unplugged and reconnected...sometimes when the USB bus puts the device to sleep it does not wake the VCOM port)
- 2. When the device is connected to your PC, you should see it listed in the Debug Adapters window in Simplicity Studio
- 3. In the Debug Adapters window, right-click on the device to display the options window and select "Launch Console..."
- Build the project and flash the project to connected Thunderboard Sense 2 using the "Debug" button in the top menu.
   a. Simplicity Studio will automatically shift views to the Debug perspective and halt at the MCU at the start of main()
  - b. Press F8 or click on the "Resume" button 🏴 to resume program execution
- 5. In the main window, navigate to the console window and select the "Serial 1" tab; this corresponds to the VCOM port
- 6. The following commands are now available in the console window:
  - a. help general help; returns other command syntax
  - b. help sensor / help adc help using the group command
  - c. sensor temp [unit] (C or F) returns temperature reading
  - d. **sensor humidity** returns humidity reading
  - e. sensor all [unit] (C or F) returns temperature and humidity
  - f. **adc raw** returns temperature and humidity
  - g. adc voltage returns temperature and humidity
  - h. adc all returns temperature and humidity
- 7. In the console input text field type "help" and press Enter
  - a. CLI interface output will display the received command along with "adc" and "sensor"
  - b. Type "help sensor" or "help adc" to display CLI command information for each command group
  - c. Type "sensor all F" to display the temperature in degrees Farenheit and humidity as a percentage
  - d. Type "adc all" to display the raw ADC data (displayed in hexadecimal) and conversion to mV.



Disconnect from the Thunderboard Sense 2 debug connection and return to the Simplicity IDE perspective before moving on to the next part of the lab.

# 4 Adding RTOS

With Project Configurator and the application code already provided, adding a real-time operating system is straight forward.

#### 4.1 Add the RTOS Kernel software component via Project Configurator

From the Simplicity IDE perspective and viewing demo.slcp in the main window, navigate to the "Software Components" tab.

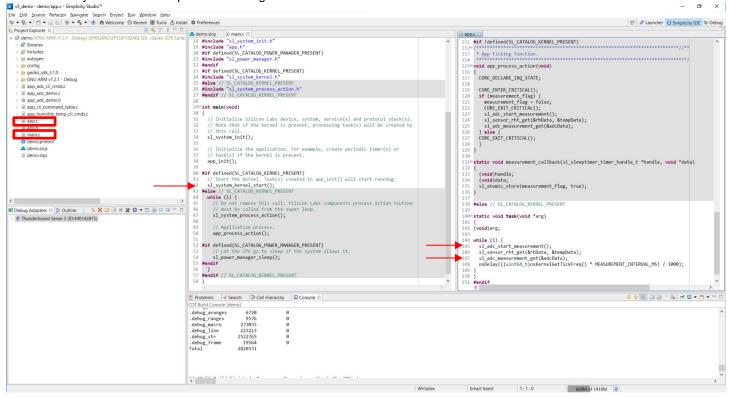
From here, install the following Software Components:

RTOS > FreeRTOS > FreeRTOS

In sl\_event\_handler.c, the process action has been replaced with function sl\_kernel\_start().

In the project folder, CMSIS>RTOS2 is included automatically

The "super loop" in main() has been replaced by the RTOS kernel start, and app\_init() now creates a kernel task to handle timed measurements every 2 seconds. Sleeptimer is no longer utilized.



If the ADC functions were not added to the kernel task as suggested in step 3.4, the ADC functions sl\_adc\_start\_measurement() and sl\_adc\_measurement\_get(&adcData) must be added surrounding sl\_sensor\_rht\_get() in app.c

NOTE: For this demo, we use FreeRTOS, but either the FreeRTOS or Micrium Kernel will work with the provided application code. To switch to the Micrium OS, first uninstall FreeRTOS in the Software Components, then install RTOS > Micrium OS > Kernel > Micrium OS Kernel.

## 4.2 Build and test

The project can now be built using an RTOS kernel and tasked based programming model. The CLI interface will have the same command list and appear no different functionally from the CLI console. Build and run using the same steps outlined in 3.5.

Again, be sure to disconnect from the Thunderboard Sense 2 debug connection and return to the Simplicity IDE perspective before progressing to the final part of the lab.

# 5 Adding Bluetooth to the project

## 5.1 Add the Bluetooth software component via Project Configurator

In Simplicity Studio, back in the Simplicity IDE perspective, navigate to the demo.slcp in the main viewing window and open the "Software Components" tab.

From here, install the following Software Components:

- Bluetooth > Bluetooth Default Features
- RTOS > FreeRTOS > FreeRTOS Heap 4
- Bluetooth > Service > Simple timer service for FreeRTOS

vi_demo-demo/demo.kcp-SinglicityStudio** Sie Edit Banigate Search Broject Bun Window Help			- 0 ×
to • • • • • • • • • • • • • • • • • • •	Directorences		If Suncher Simplicity IDE To Debug
Project Explorer 11 E to 7 8 ** 0			
## Ame (DV) AMM /21 - Denug (DHEAMOL7930740040125 - Gene SDC Suite         > @ Denus           > @ Denus         > @ Denus	demo         OVERVIEW         SOFTWARE COMPONENTS         CON           ▼ Fiter : Configurable Components         Installed Components         Installed Components	Components Installed by You	Q. Search keywords, component's name
	Advanced Configurators	Bluetooth Default Features	Install
) is app_adc_demo.t ) is app_adc_demo.t ) is app_di_command_table.c ) is app_dii_command_table.c	▼ Bluetooth		· Add component to project
> i appc > i apph > i mainc	Application     GATT	Description Includes Bluetooth stack and default selection of features.	
<ul> <li>demo.pintool</li> <li>demo.skp</li> </ul>	► NCP	Configuration for Bluetooth stack is provided under Bluetooth core of	omponent.
🖹 demoslps	► NVM	Quality PRODUCTION	
	OTA     Sensor		
K Debug Adapters IX 😒 Outline 🔰 🛠 🐸 🖉 🕷 💥 🖉 🕶 🗔 📾 🕀 🖷 🖸	Service		
Proof Magness of a Counter      Proof Magness of a Counter     Proof Magness of a Counter      Proof Magness of a Counter	▶ Stack	1	
	▶ Utility		
	Bluetooth Default Features		
	► LwIP		View Dependencies
	Problems      Process Call Hierarchy Console      Program Output Console		¥   16 Q IF   🖻 🛡 😁 😁 🖓
		813M (d)1	

# 5.2 Update Device Name via GATT configurator

While still viewing demo.slcp in the main window, navigate to the "Configuration Tools" tab and open the Bluetooth GATT Configurator tool.

v5_demo - demo/demo.slcp - Simplicity Studio <sup>™</sup>		- 🗆 X					
Eile Edit Navigate Search Project Bun Window Help							
🎋 🕶 💁 🖝 🔝 🐨 🐘 🖏 💌 🐐 👘 👘 Welcome 💿 Reci		😰 🖉 Launcher 🚯 Simplicity IDE 🕸 Debug					
🔁 Project Explorer 😫 📃 😫 🖗 🍸 🧯 🖓 🔲	demosicp []	° 0					
	demo OVERVIEW SOFTWARE COMPONENTS CON Bluetooth GATT Configurator	VFIGURATION TOOLS					
Bondy John V.C Leong     Bondy John V.C Leong John V.C L	remote devices. The GATT Configurator is a simple-to-use tool to	s/services/characteristics defined by the Bluetooth SIG as well as					
	Pin Tool	ିକୁ Open					
< >> ® Debug Adapters :: S 20 of the " S 20 of the Company o	Description Graphical tool for configuring plas on your hardware.						
	E Problems 🛷 Search 🖫 Call Hierarchy 🛛 Console 11						
	COT Dud Consel General 1027 Buil Consel General 1027 Buil Consents Build of configuration GBU ABM v7.2.1 - Debug for project demo **** make .9 Build make .9 Build get to be done for "all". 10:20:12 Build Finished. 0 errors, 0 warnings. (took Bs.45es)						
	¢	2414-0 1421M					

Within the Bluetooth GATT Configurator tool window, select from the GATT tree Custom BLE GATT > Generic Access > Device Name.

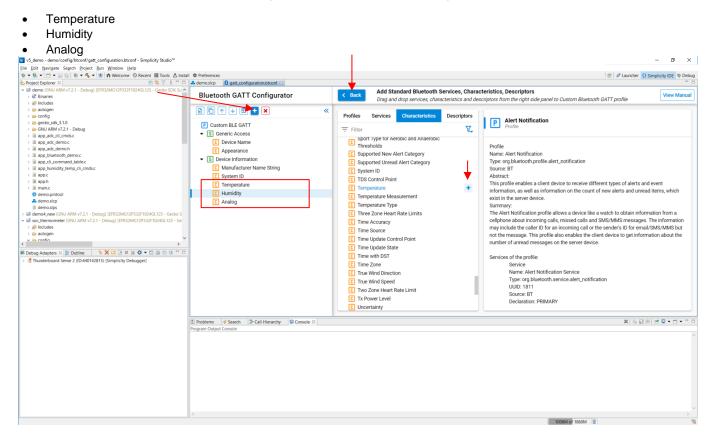
In the configuration window to the right, modify the **Device Name** characteristic from "Silabs Example" to "FAE\_TRAINING" and uncheck the overwrite auto calculated length checkbox. Also consider adding a unique numeral or any unique name in order to identify with mobile device.

Ø gatt_configuration.btconf ≅		- 0
Bluetooth GATT Configurator		View Manual
<ul> <li>▶ □ ↑ ↓ ⋺ + ×</li> <li>P Custom BLE GATT</li> <li>S Generic Access</li> <li>Device Name</li> <li>Appearance</li> </ul>	C Device Name Characteristic Name Device Name	(?) UUID 2A00
<ul> <li>5 Device Information</li> <li>C Manufacturer Name String</li> </ul>	device_name	SIG type org.bluetooth.characteristic.gap.c
C System ID	Value settings Constant	USER HEX UTF-8 Variable length
	Initial value FAE_TRAINING	Value length 12 byte
	Characteristic capabilities -	User description

## 5.3 Add new GATT characteristics to Device Info service

Select the "Device Information" Service in the GATT tree view and then click on the "+" icon to add some standard GATT items to display the sensor information

Select "Characteristics" tab in the main GATT configuration window; add the following characteristics:



# Adding Bluetooth to the project

Click "Back" to view the new characteristics in the main view

For the Temperature Characteristic:

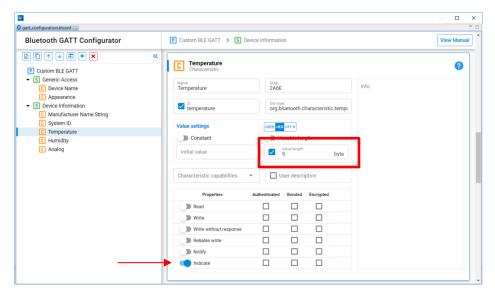
- change the value length value to 5 bytes
- turn on the "indicate" property

For the Humidity Characteristic:

- change the value length value to 4 bytes
- turn on the "indicate" property

For the Analog Characteristic:

- change the value length value to 7 bytes
- turn on the "indicate" property



## 5.4 Add application specific code

From the previously extracted compressed zip in step 2.7, open the "demo4" folder.

Drag-and-drop the file app\_bluetooth\_demo.c within this folder into the demo project's top directory in Simplicity Studio's "Project Explorer" window.

	all 🌣 Preferences				E A Launch	ther () Simplicity IDE P De	bug 🙏 Network A
	demo.slcp 💠 😽 J-Link Silicon Labs	(440142815) 🔄 main.c					
demo (GNU ARM v7.2.1 - Debug) [EFR32MG12P332F1024GL125 - Gecko SDK Suite	OVERVIEW SOFTWARE	COMPONENTS					
includes autopen is config pecko, sdk, 3.1.0	▼ Filter : Configurable Com	ponents Installed Component	Components Installed by You		C	Q Search keywords, co	mponent's name
GNU ARM v7.2.1 - Debug	UMAUKV		· · ·				
an app.c ∰ app.h @ main.c	📕   🗹 📕 🖛	demo4			-		onfigure
e demo pintool demo sicp demo sios	File Home	Share View				~	0
a anno ann	$\epsilon \rightarrow \cdot \uparrow$		> v5_ProjectConfigurator_Lab > d	lemo4	~ Ū	Search de 🔎	It
drag-and-drop	Name	^	Date modified	Туре	Size		
	🗹 🖻 app_blueto	ooth_demo.c	11/20/2020 1:14 PM	C Source		14 KB	ance a
							n_init()
	1	elected 13.0 KB					of the
	1	elected 13.0 KB	Quality PRODUCTION				of the
	1 item 1 item s	elected 13.0 KB	~		X		of the
g Adapters III (2) Outline S X G R K (2) O • C (2) O (2) (2) (2) (2) (2) (2) (2) (2) (2) (2)	1 item 1 item s RTC driver		~		×	(BEE)	of the
	1 item 1 item s RTC driver SPIDRV	Si File Operation	PRODUCTION		×	(BEE)	of the
	1 item 1 item s RTC driver SPIDRV SPIDRV Core	S File Operation	~		×		of the
	1 item 1 item s RTC driver SPIDRV SPIDRV Core Simple Button	Si File Operation	PRODUCTION		×		of the
	1 item 1 item s RTC driver SPIDRV SPIDRV Core Simple Button Simple Button Core	Select how files should	PRODUCTION		×	Vie	of the
	1 item 1 item s RTC driver SPIDRV SPIDRV Core Simple Button Simple Button Simple Button Simple Button	Select how files should	PRODUCTION	c	×	Vie	of the
	1 item 1 item s RTC driver SPIDRV SPIDRV Core Simple Button Simple Button Simple Button Simple Button	Select how files should	PRESOUTHON  be imported into the project: cations relative to: PROJECT_LOG	c	×	Vie	of the

In the "File Operation" window, select the "Copy Files" radio button and press OK.

#### 5.5 Build and flash to device; Use BLE Scanner to locate Thunderboard Sense 2 and readout sensor data

- 1. Build and flash the latest project configuration using the same methods as used previously in steps 2.8
- 2. Open BLE Scanner on your mobile device.
- 3. Locate your demo device by device name (FAE\_TRAINING) programmed in step 5.2 and click connect
- 4. Expand the "Device Information" service and navigate to the "Temperature" characteristic
- 5. Click the "Indicator" button next to the characteristic to enable indications and observe the sensor value
- 6. Follow step 5 above for "Humidity" and "Analog" to view these sensor value

