Designed by:

CONNECTED DEVELOPMENT

Components provided by:

tti Symmetry ELECTRONICS MOUSER ELECTRONICS

Contactless Temperature Sensor

Reference Design

By: Dave Hoover | www.CovidTempSensor.com

Build Your Own Contactless Temperature Sensor
Open-Source Reference Design

By: Dave Hoover, VP Advanced Technology - Connected Development

The COVID-19 pandemic has presented hardships for businesses and consumers across the globe. Even businesses that have been categorized as essential have had to seek solutions to ensure they are operating safely and efficiently. Unfortunately, enterprises such as hospitals and warehouses that are prone to large gatherings run a heightened risk of contracting the virus. The solution? An economic contactless wrist temperature sensor that ensures essential workers can safely enter the workplace, without endangering their co-workers or any manual temperature gauge operators.

Members of the TTI, Inc. Family of Specialists have combined their resources to develop an open-source, contactless wrist temperature sensor that is cost-effective and efficient. The temperature sensor was designed and prototyped by Exponential Technology Group company, Connected Development (CD), which specializes in hardware and software design services. CD tapped into the expansive electronic component portfolios of TTI, Inc., Symmetry Electronics, and Mouser Electronics, to develop this open-source design. CD's temperature sensor design centers around a Silicon Labs microcontroller starter kit, a TE thermopile for temperature measurements, a TDK piezoelectric buzzer for audio feedback, and Cree LEDs for a visual cue.

TTI's employees are currently using the temperature sensor at their warehouses. Employees simply hold up their wrist to the sensor to receive a reading of their temperature accompanied by a beeping noise and a red or green light that indicates whether they are at or above acceptable temperature range.

The contactless wrist temperature sensor could benefit several use cases, including hospitals, warehouses, retail, and offices. Although the sensor is not available for sale, the design has been made open source to allow engineers to develop the product on their own. The following is a detailed reference guide to walk intermediate-level engineers through the steps it takes to build this technology.

Project Materials and Resources

The following information pertains to the various components used in the contactless wrist temperature sensor project. The links can be accessed to order relevant parts and download helpful datasheets, and technical information or complete project bill of materials link here.

Bill of Materials (BOM)

- SLSTK3301A–Silicon Labs EFM32 Tiny Gecko TG11 Starter Kit
- TSD305-1C55–TE TSD305 Digital Thermopile Sensor
- PS1240P02CT3–TDK Piezoelectric Buzzer
- CLX6F-FKC-CKNNQDGBB7A363–Cree PLCC6 3 in 1 SMD LED

- B07GD1W1VL–Jumper Cables (To connect starter kit to sensor, buzzer, and LEDs. Note: Gender and length can vary based on mechanical design. The current design uses nine separate jumper cables. (See 'Install Electrical' section below.)
- 3318–USB Cable (For powering the device. Note: Length and brand selection is not imperative–any USB mini cable will work.)

Software

- Simplicity Studio–Getting Started with EFM32 Tiny Gecko 11 Instructions
- Contactless Temperature Sensor GitHub Repository

Useful Links and Resources

- Silicon Labs EFM32 Tiny Gecko TG11 Starter Kit User's Guide
- TE TSD305 Digital Thermopile Sensor Data Sheet
- TDK Piezoelectric Buzzer Data Sheet
- Cree PLCC6 3 in 1 SMD LED Data Sheet

Mechanical

The mechanical design of the temperature sensor is beyond the scope of this article. Care should be taken to provide a design that is ergonomically sound and capable of rapidly taking temperatures for individuals of various heights. The display and LED should be easy to see while holding your wrist within close proximity to the temperature sensor. The design should not require the user to have to keep the wrist at an uncomfortable angle during measurements, nor should it require additional personnel to operate.

Project Technology Overview

The contactless temperature sensor utilizes the TE TSD305 Digital Thermopile Sensor to measure object temperature. The Silicon Labs EFM32 Tiny Gecko TG11 Starter Kit is used to collect temperature sensor information from the thermopile and to notify the end-user of results.

The thermopile is also used to detect whether a human body is within close proximity to the measurement station. The Silicon Labs starter kit microcontroller constantly regulates the thermopile for temperature fluctuations. If the temperature rises above 90˚ Fahrenheit, it is assumed that a human body is close by, prompting a measurement cycle to begin. The microcontroller then awaits a consistent measurement, which is achieved by capturing three readings in a row that are within one degree of each other. Once a stable measurement is captured, the temperature is displayed on the Silicon Labs starter kit LCD. The TDK Piezoelectric Buzzer sounds a short beep accompanied by a green light from the Cree PLCC6 3 in 1 SMD LED (if the temperature is below 100.4˚ Fahrenheit), or a long beep with a red LED light (if the temperature is greater than 100.4˚ Fahrenheit).

(Note: All temperature thresholds mentioned above can easily be modified by changing a setting within the code.)

Operational Flowchart

The following flowchart (**Figure 1**) demonstrates the high-level operation of the temperature sensor.

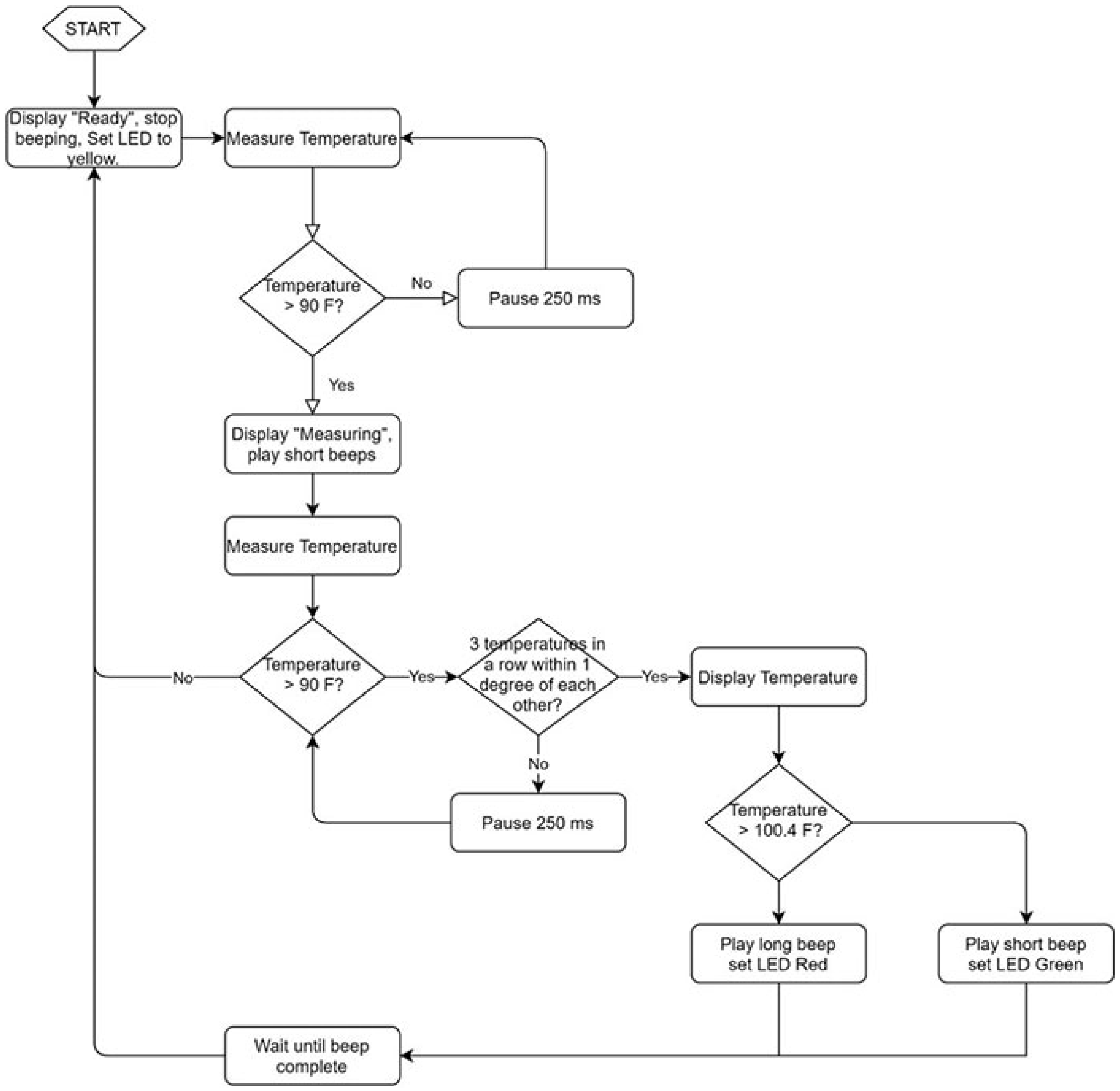

START
Display "Ready", stop beeping, Set LED to yellow.
Measure Temperature
Temperature > 90 F?
No
Pause 250 ms
Yes
Display "Measuring", play short beeps
Measure Temperature
No
Temperature > 90 F?
Yes
3 temperatures in a row within 1 degree of each other?
Yes
Display Temperature
No
Pause 250 ms
Temperature > 100.4 F?
Play long beep set LED Red
Play short beep set LED Green
Wait until beep complete

Figure 1: *Flowchart offering a high-level demonstration of how the temperature sensor operates. (Source: Connected Development)*

Silicon Labs EFM32 Tiny Gecko TG11 Starter Kit

The Silicon Labs EFM32 Tiny Gecko TG11 Starter Kit includes the microcontroller for the temperature sensor, all necessary power regulators, a USB connector for power, and an LCD for displaying the temperature and user prompts. All peripherals (thermopile, piezoelectric buzzer, and LED) are connected to the starter kit using jumper cables. (See 'Install Electrical' section below).

TE Digital Thermopile Sensor

The Silicon Labs microcontroller communicates with the TE thermopile via I^{2}C reads and writes. The TE sensor includes an ambient temperature sensor and the thermopile, which is used to calculate calibration offsets. The algorithm for reading measurements and applying calibration offsets to determine the final object temperature can be found in the TE TSD305 Digital Thermopile Sensor Data Sheet.

TDK Piezoelectric Buzzer

A pulse-width modulation (PWM)-capable General Purpose I/O (GPIO) pin on the Silicon Labs microcontroller is used to drive the piezoelectric buzzer.

Cree PLCC6 3 in 1 SMD LED

The temperature sensor only requires a green and red LED for visual cues (the temperature itself is displayed on the Silicon Labs starter kit LCD). The LEDs can be controlled with simple GPIO writes from the Silicon Labs microcontroller.

Simplicity Studio Software

The software development tool used for the project is Silicon Labs' Simplicity Studio Integrated Development Environment (IDE). Instructions related to installing Simplicity Studio are found in the 'Set Up Silicon Labs Tools and Environment' section below.

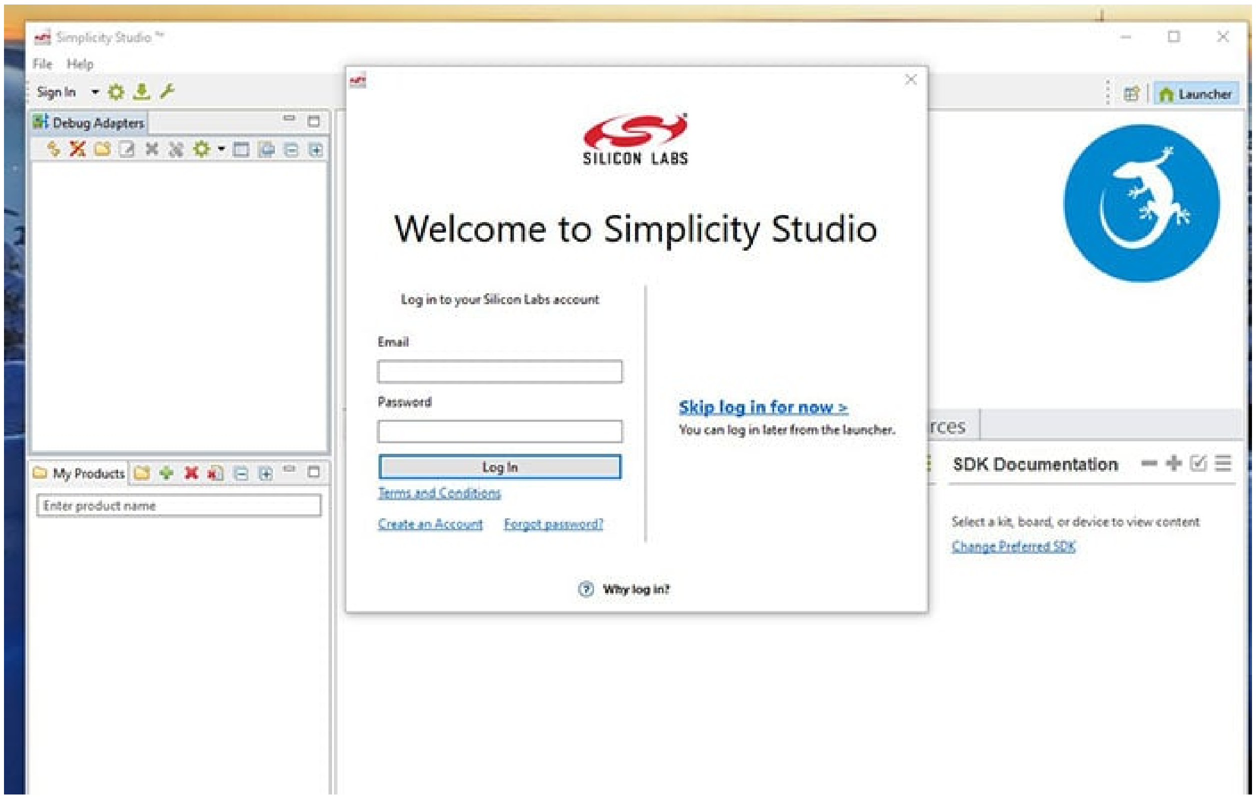
Developing the Temperature Sensor

Perform the following steps to develop the contactless temperature sensor:

1. Set Up Silicon Labs Tools and Environment
2. Retrieve Project Code from Github
3. Import Project Code into Simplicity Studio
4. Install Electrical
5. Set Up Mechanical
6. Calibrate and Tune

Set Up Silicon Labs Tools and Environment

1. Procure a Silicon Labs EFM32 Tiny Gecko TG11 Starter Kit.
2. Register for or Log In to the Silicon Labs site.
3. Follow the Getting Started with EFM32 Tiny Gecko 11 Instructions (Note: Run installer as administrator when installing Simplicity Studio per “Step 2: Download and Install Simplicity Studio”.)
4. Log in to your Silicon Labs account from Simplicity Studio. The login screen will pop up by default (**Figure 2**).
5. Select “Install by Device” when Simplicity Studio Installation Manager window pops up (**Figure 3**).
6. Plug your Silicon Labs starter kit into a USB port on your computer. (Note: The screenshot used in Figure 3 shows a Giant Gecko (instead of Tiny); however, the Tiny Gecko install will look similar.)


Welcome to Simplicity Studio

Log in to your Silicon Labs account

Email

Password

Log In

Terms and Conditions

Create an Account Forgot password?

Skip log in for now >

You can log in later from the launcher.

Why log in?

Figure 2: *Silicon Labs login screen within Simplicity Studio. (Source: Simplicity Studio)*

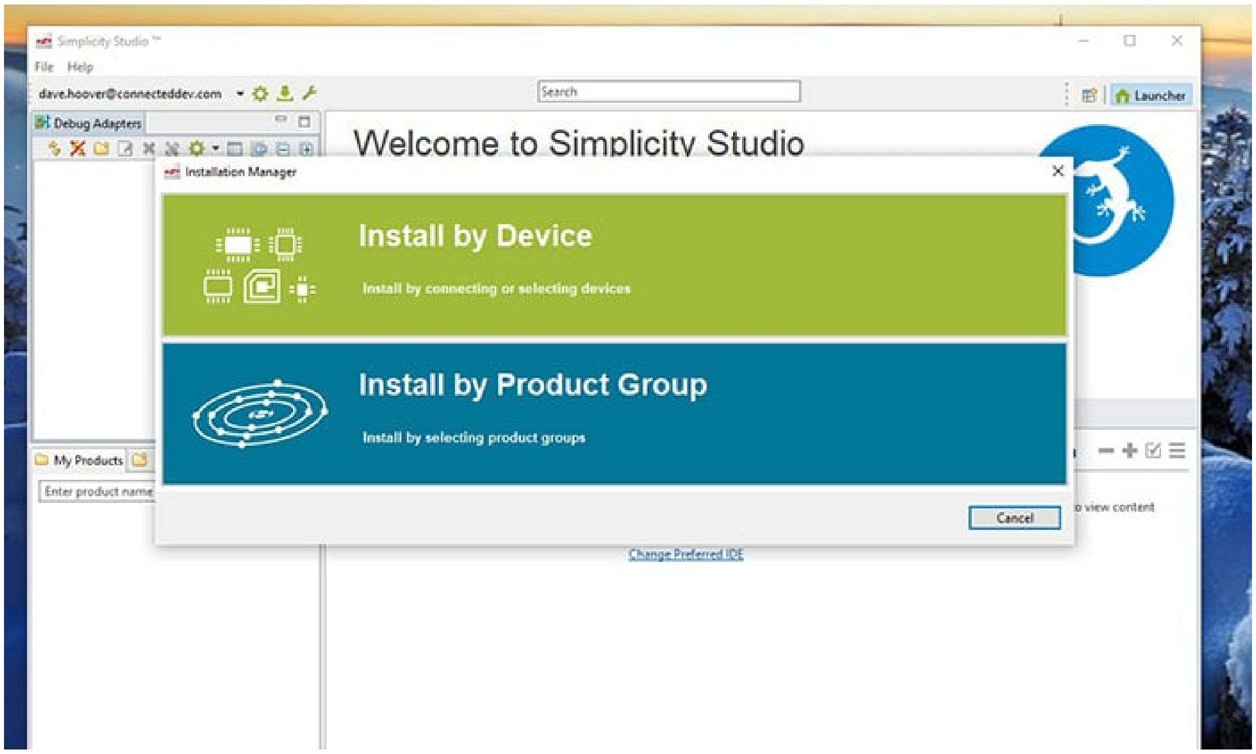
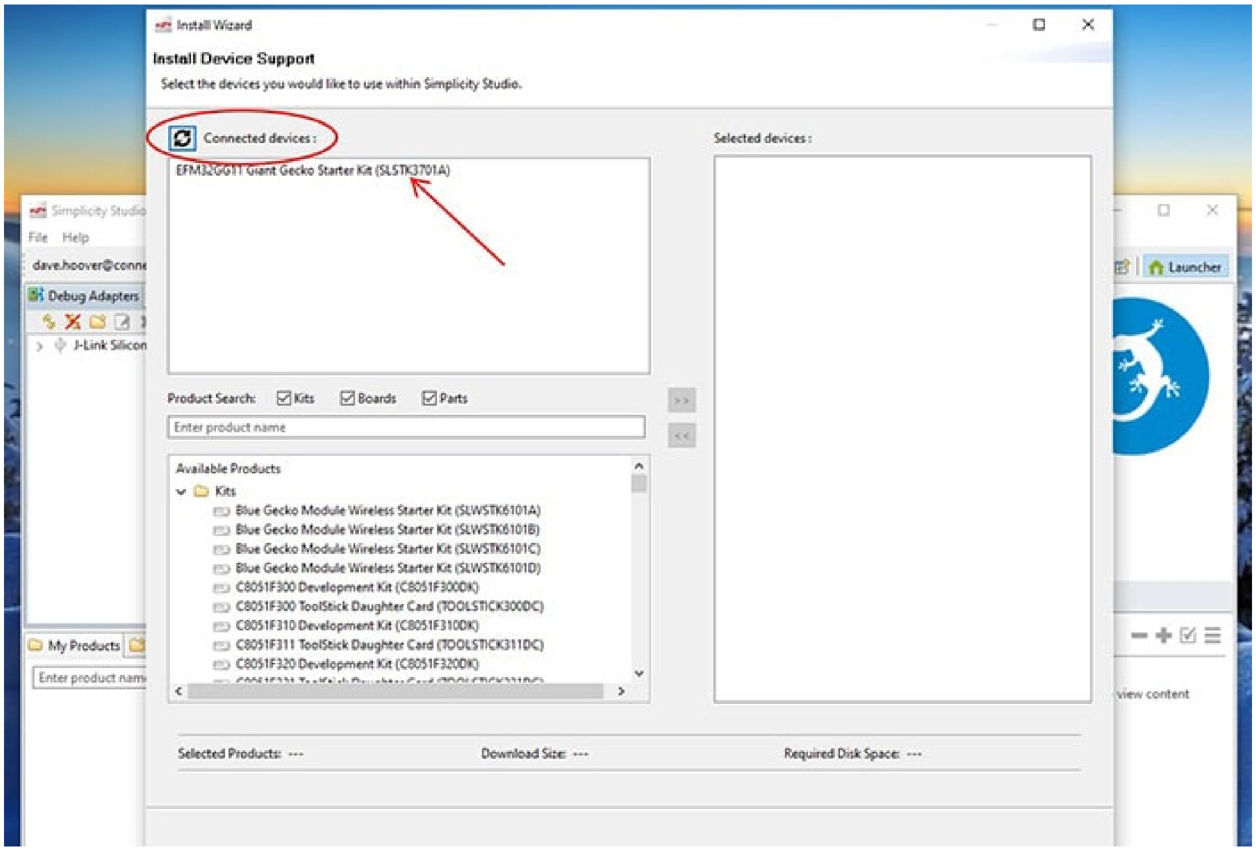


Figure 3: *Install by Device. (Source: Simplicity Studio)*

7. Double click on your starter kit that is now listed in the “Connected devices" pane and click "Next" (**Figure 4**).
8. Continue to click "Next" with default options selected.
9. Accept the MLA (License Agreement), click “Finish”, and wait for installation to complete.
10. Click on the device and then the “Getting Started” tab (**Figure 5**).


Install Wizard

Install Device Support

Select the devices you would like to use within Simplicity Studio.

Connected devices :

EFM32GG11 Giant Gecko Starter Kit (SLSTK3701A)

Selected devices :

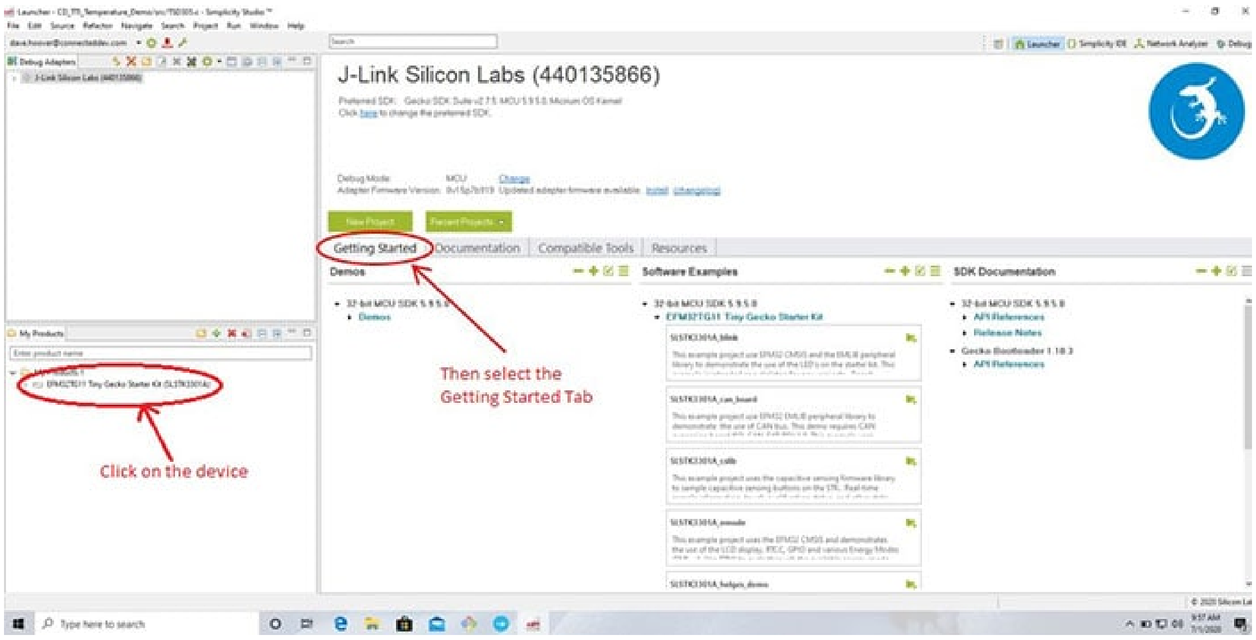
Product Search: [✓ Kits] [✓ Boards] [✓ Parts]

Enter product name

Available Products
- Kits
  - Blue Gecko Module Wireless Starter Kit (SLWSTK6101A)
  - Blue Gecko Module Wireless Starter Kit (SLWSTK6101B)
  - Blue Gecko Module Wireless Starter Kit (SLWSTK6101C)
  - Blue Gecko Module Wireless Starter Kit (SLWSTK6101D)
  - C8051F300 Development Kit (C8051F300DK)
  - C8051F300 ToolStick Daughter Card (TOOLSTICK300DC)
  - C8051F310 Development Kit (C8051F310DK)
  - C8051F311 ToolStick Daughter Card (TOOLSTICK311DC)
  - C8051F320 Development Kit (C8051F320DK)

Selected Products: ---   Download Size: ---   Required Disk Space: ---

Figure 4: *Starter kit listed in the "Connected devices" pane. (Source: Simplicity Studio)*


J-Link Silicon Labs (440135866)

Preferred SDK: Gecko SDK Suite v2.7.5: MCU 5.9.5.0, Micrium OS Kernel

Click here to change the preferred SDK.

Debug Mode: MCU Change

Adapter Firmware Version: 0v15p7b919 Updated adapter firmware available. Install (changelog)

Figure 5: *Getting Started tab. (Source: Simplicity Studio)*

11. Choose a software example (such as “blink”) from the “Software Examples” section (**Figure 6**). (Note: While not the required selection, blink is a simple software that can easily prove that the Simplicity Studio installation, compiling, and microcontroller code download are working properly.
12. After clicking on the “blink” (or preferred software selection - see step 11 above) project, a pop-up dialog appears asking to confirm; create the new project; and switch to the Simplicity IDE perspective. Click “Yes”.

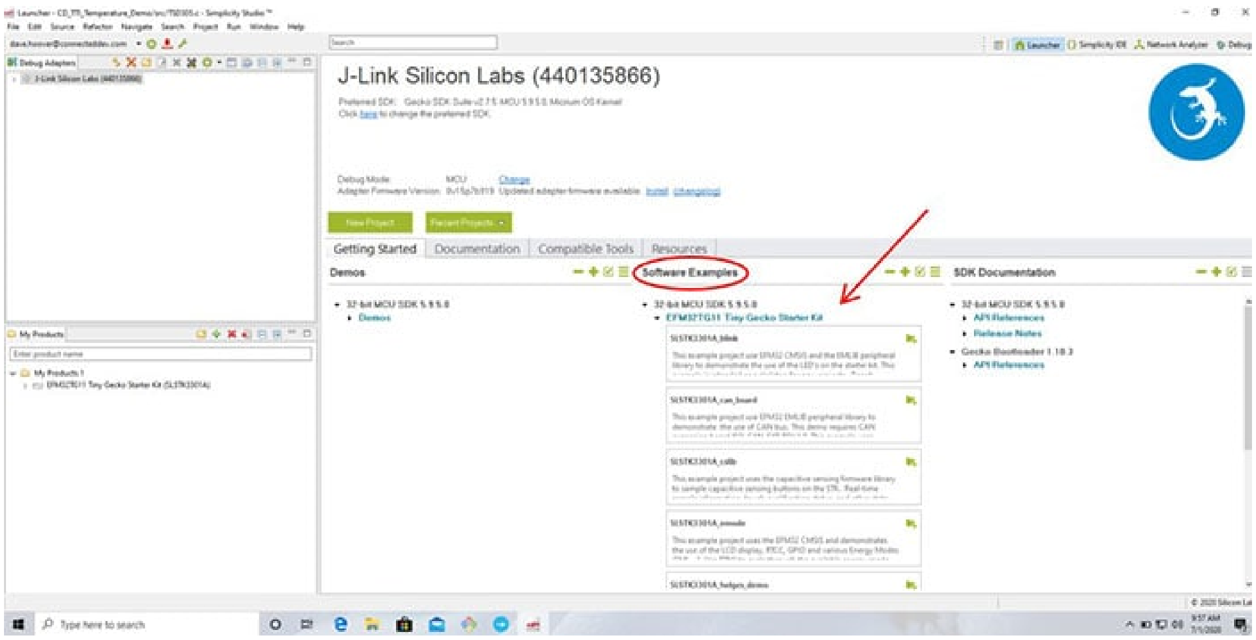
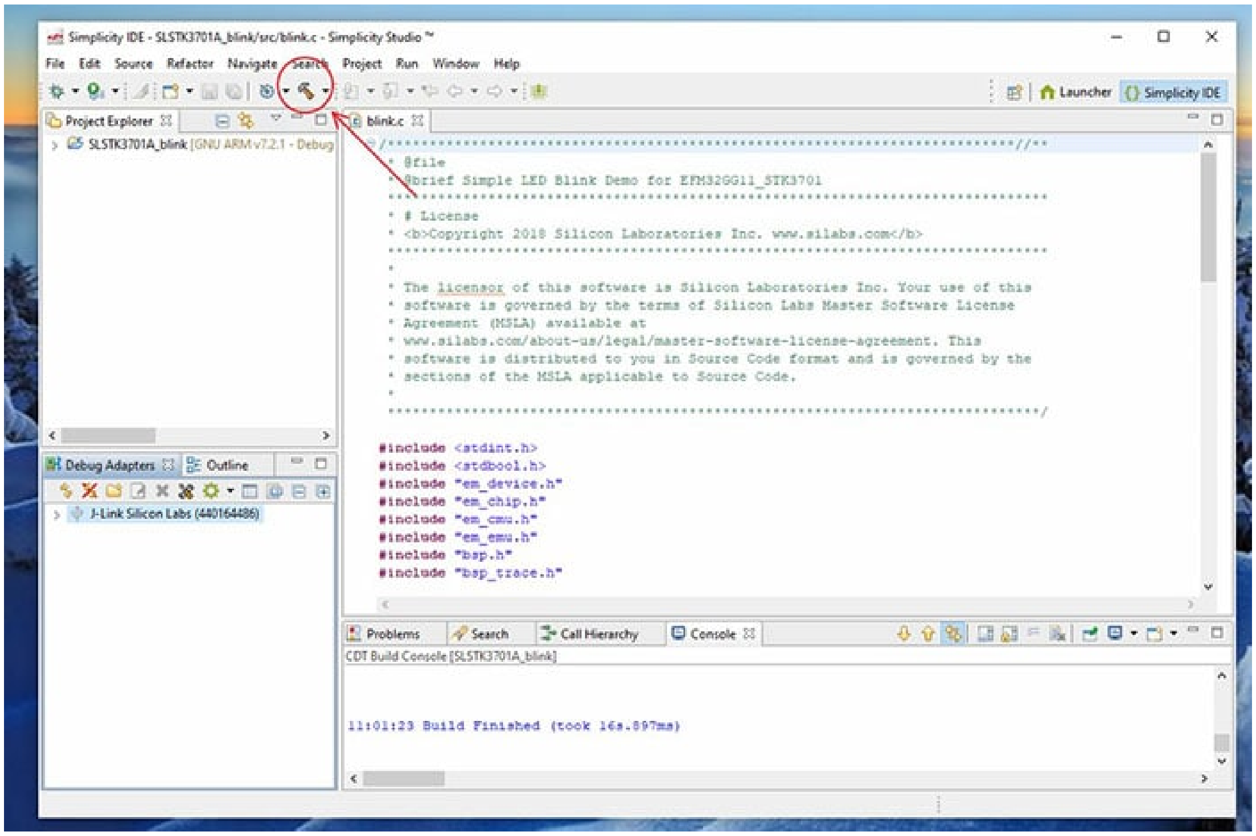


Figure 6: *Software Examples section. (Source: Simplicity Studio)*

13. Compile the project using the “hammer icon” (**Figure 7**).
14. Debug the project using the “bug icon” (**Figure 8**).
15. Run the project once the debugger is launched by clicking on the “run icon”. Breakpoints can be set, single-stepping is enabled, etc. (**Figure 9**).



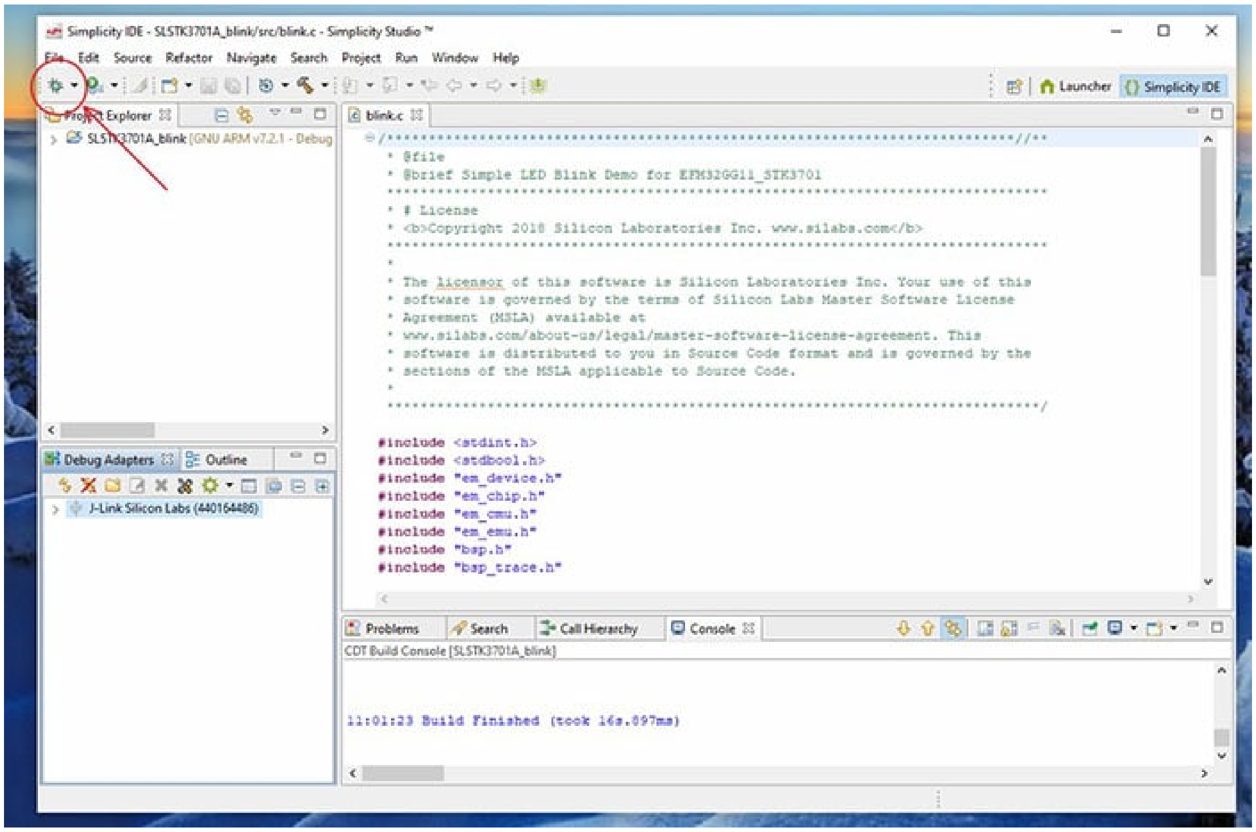
Simplicity IDE - SLSTK3701A_blink/src/blink.c - Simplicity Studio ™
File Edit Source Refactor Navigate Search Project Run Window Help
Launcher
Simplicity IDE
Project Explorer
SLSTK3701A_blink [GNU ARM v7.2.1 - Debug
blink.c
Debug Adapters
Outline
J-Link Silicon Labs (440164486)
Problems
Search
Call Hierarchy
Console

```
/***************************************************************************//**
 * @file
 * @brief Simple LED Blink Demo for EFM32GG11_STK3701
 *******************************************************************************
 * # License
 * <b>Copyright 2018 Silicon Laboratories Inc. www.silabs.com</b>
 *******************************************************************************
 *
 * The licensor of this software is Silicon Laboratories Inc. Your use of this
 * software is governed by the terms of Silicon Labs Master Software License
 * Agreement (MSLA) available at
 * www.silabs.com/about-us/legal/master-software-license-agreement. This
 * software is distributed to you in Source Code format and is governed by the
 * sections of the MSLA applicable to Source Code.
 *
 ******************************************************************************/

#include <stdint.h>
#include <stdbool.h>
#include "em_device.h"
#include "em_chip.h"
#include "em_cmu.h"
#include "em_emu.h"
#include "bsp.h"
#include "bsp_trace.h"
```

CDT Build Console [SLSTK3701A_blink]

```
11:01:23 Build Finished (took 16s.897ms)
```

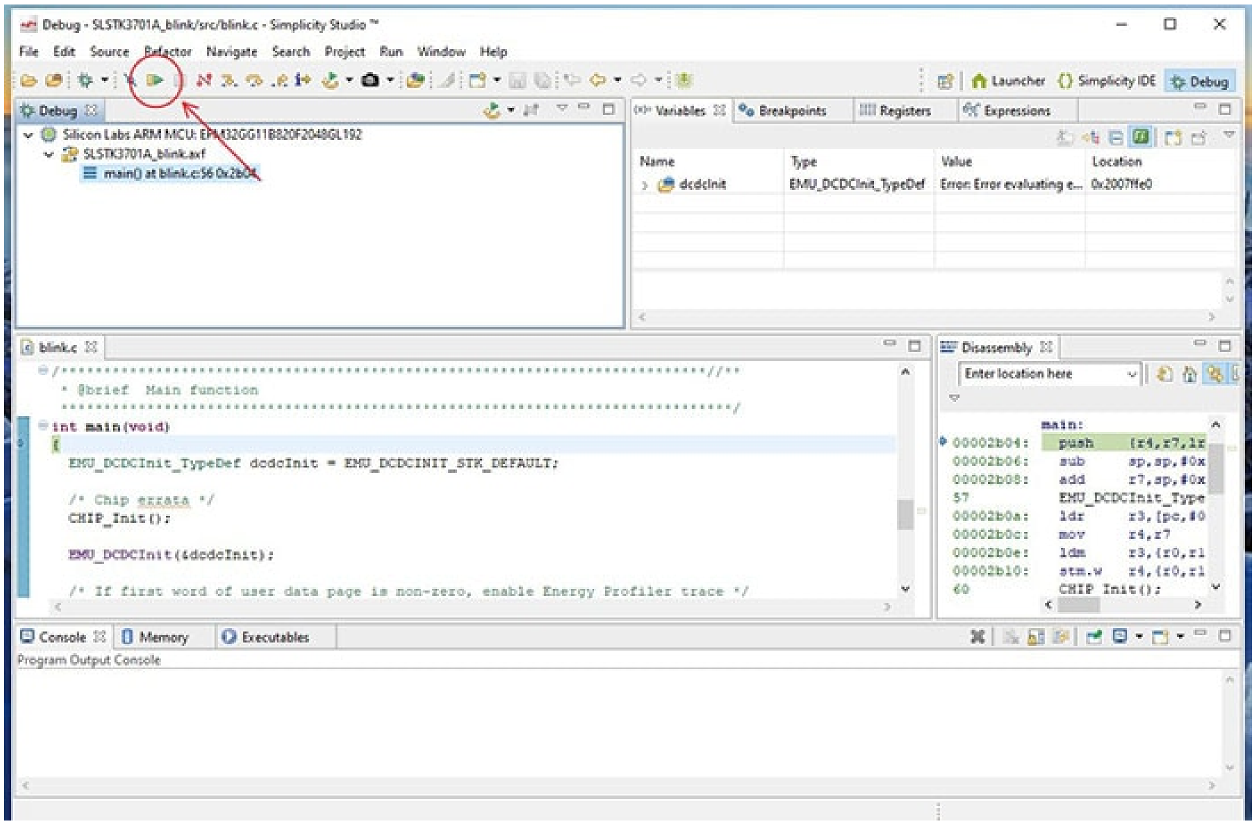
Figure 7: *Compile the project. (Source: Simplicity Studio)*


Simplicity IDE - SLSTK3701A_blink/src/blink.c - Simplicity Studio ™
File Edit Source Refactor Navigate Search Project Run Window Help
Launcher
Simplicity IDE
Project Explorer
SLSTK3701A_blink [GNU ARM v7.2.1 - Debug
blink.c
Debug Adapters
Outline
J-Link Silicon Labs (440164486)
Problems
Search
Call Hierarchy
Console
CDT Build Console [SLSTK3701A_blink]

```
/***************************************************************************//**
 * @file
 * @brief Simple LED Blink Demo for EFM32GG11_STK3701
 *******************************************************************************
 * # License
 * <b>Copyright 2018 Silicon Laboratories Inc. www.silabs.com</b>
 *******************************************************************************
 *
 * The licensor of this software is Silicon Laboratories Inc. Your use of this
 * software is governed by the terms of Silicon Labs Master Software License
 * Agreement (MSLA) available at
 * www.silabs.com/about-us/legal/master-software-license-agreement. This
 * software is distributed to you in Source Code format and is governed by the
 * sections of the MSLA applicable to Source Code.
 *
 ******************************************************************************/

#include <stdint.h>
#include <stdbool.h>
#include "em_device.h"
#include "em_chip.h"
#include "em_cmu.h"
#include "em_emu.h"
#include "bsp.h"
#include "bsp_trace.h"
```

```
11:01:23 Build Finished (took 16s.897ms)
```

Figure 8: *Debug the project. (Source: Simplicity Studio)*



```
Silicon Labs ARM MCU: EFM32GG11B820F2048GL192
  SLSTK3701A_blink.axf
    main() at blink.c:56 0x2b04
```

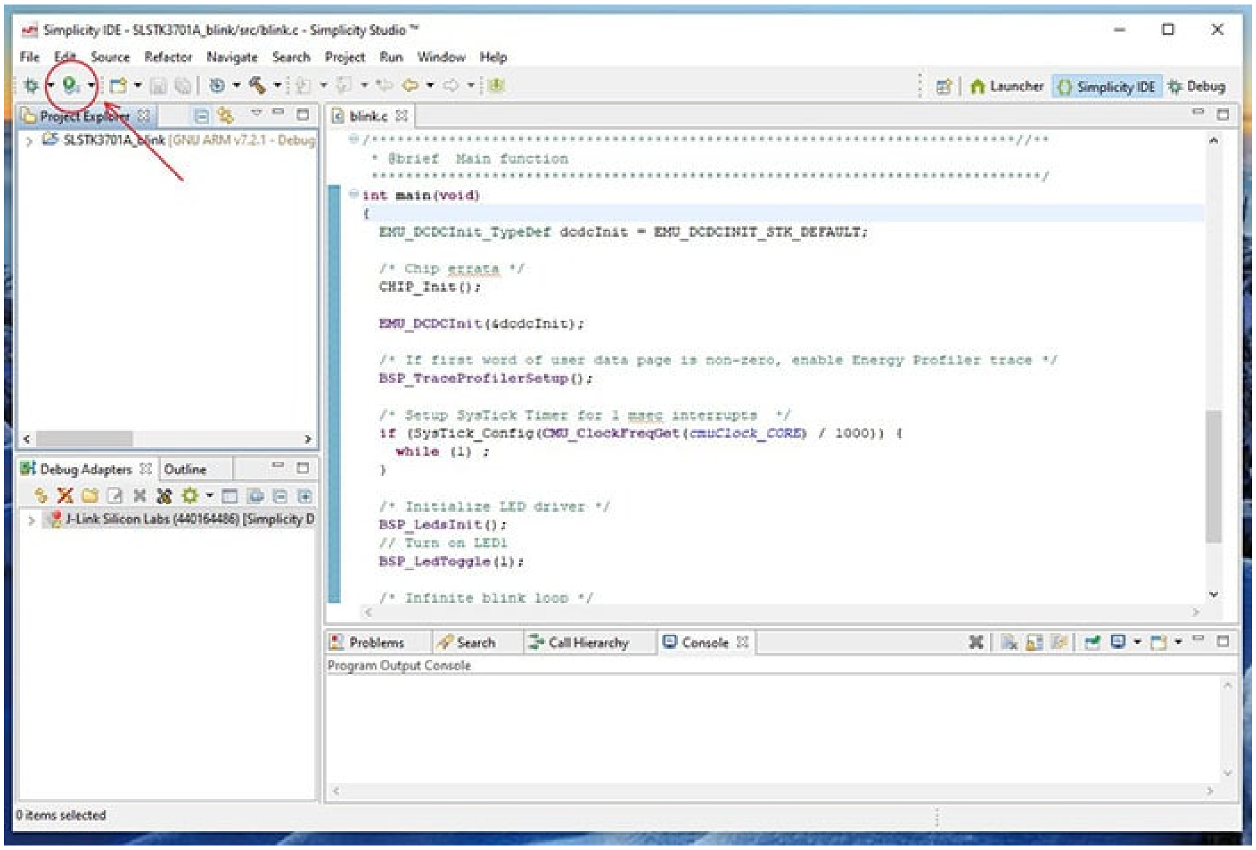
Name	Type	Value	Location
dcdcInit	EMU_DCDCInit_TypeDef	Error: Error evaluating e...	0x2007ffe0

```
/***************************************************************************//**
 * @brief  Main function
 ******************************************************************************/
int main(void)
{
  EMU_DCDCInit_TypeDef dcdcInit = EMU_DCDCINIT_STK_DEFAULT;

  /* Chip errata */
  CHIP_Init();

  EMU_DCDCInit(&dcdcInit);

  /* If first word of user data page is non-zero, enable Energy Profiler trace */
```

```
          main:
00002b04:   push      {r4,r7,lr
00002b06:   sub       sp,sp,#0x
00002b08:   add       r7,sp,#0x
57          EMU_DCDCInit_Type
00002b0a:   ldr       r3,[pc,#0
00002b0c:   mov       r4,r7
00002b0e:   ldm       r3,{r0,r1
00002b10:   stm.w     r4,{r0,r1
60          CHIP_Init();
```

Figure 9: *Run the project. (Source: Simplicity Studio)*

16. Download and run the application by clicking on the "profile icon". A dialog box will appear showing download progress. Once complete, the code will execute on the target (**Figure 10**).
17. Make a change to the blink rate in the code by modifying the src/blink.c file, line 75. The Delay function call indicates the time between each LED blink cycle in milliseconds. Modify this number to a different number of milliseconds and then re-compile and download (see steps 13 - 16 above) and observe the LED blink rate changes that take place on the target.


Simplicity IDE - SLSTK3701A_blink/src/blink.c - Simplicity Studio ™
File Edit Source Refactor Navigate Search Project Run Window Help
Launcher
Simplicity IDE
Debug
Project Explorer
SLSTK3701A_blink [GNU ARM v7.2.1 - Debug
blink.c
Debug Adapters
Outline
J-Link Silicon Labs (440164486) [Simplicity D
Problems
Search
Call Hierarchy
Console
Program Output Console
0 items selected

```
/***************************************************************************//**
 * @brief  Main function
 ******************************************************************************/
int main(void)
{
  EMU_DCDCInit_TypeDef dcdcInit = EMU_DCDCINIT_STK_DEFAULT;

  /* Chip errata */
  CHIP_Init();

  EMU_DCDCInit(&dcdcInit);

  /* If first word of user data page is non-zero, enable Energy Profiler trace */
  BSP_TraceProfilerSetup();

  /* Setup SysTick Timer for 1 msec interrupts  */
  if (SysTick_Config(CMU_ClockFreqGet(cmuClock_CORE) / 1000)) {
    while (1) ;
  }

  /* Initialize LED driver */
  BSP_LedsInit();
  // Turn on LED1
  BSP_LedToggle(1);

  /* Infinite blink loop */
```

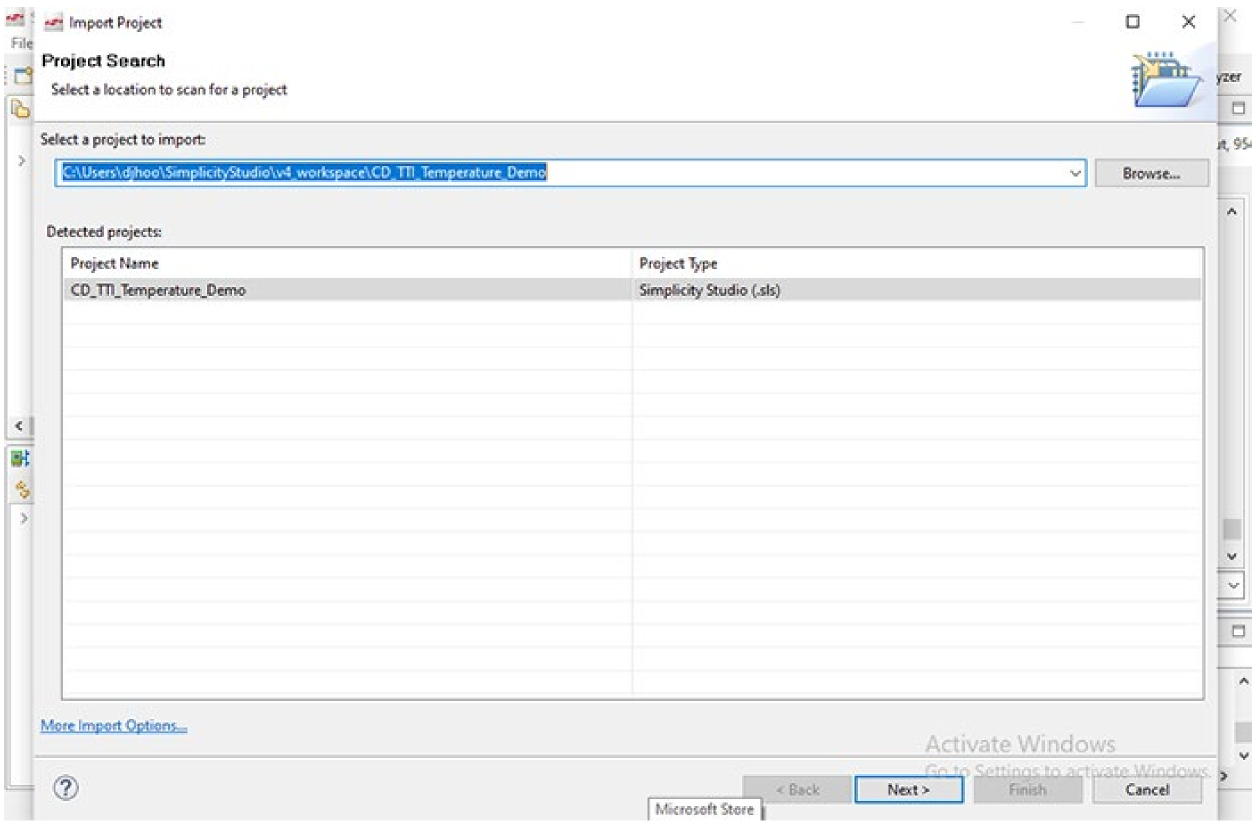
Figure 10: *Download the project. (Source: Simplicity Studio)*

Retrieve Project Code from Github

Contactless Temperature Sensor GitHub Repository

Import Project Code into Simplicity Studio

Navigate to the Simplicity IDE tab in Simplicity Studio (**Figure 11**) and import the Contactless Temperature Sensor code downloaded from Github by selecting “File”, “Import”, Browse to the project code directory, and select the file. Click “Next” and “Finish” to import the project.


Import Project

Project Search

Select a location to scan for a project

Select a project to import:

C:\Users\djhoo\SimplicityStudio\v4_workspace\CD_TTI_Temperature_Demo

Detected projects:

Project Name	Project Type
CD_TTI_Temperature_Demo	Simplicity Studio (.sls)

Figure 11: *Import the Contactless Temperature Sensor project. (Source: Simplicity Studio)*

Install Electrical

Jumper cables can connect the thermopile, buzzer, and LEDs to the Silicon Labs starter kit. Reference the Silicon Labs EFM32 Tiny Gecko TG11 Starter Kit User's Guide for the starter kit pinouts and header definitions. Access the TE TSD305 Digital Thermopile Sensor Data Sheet for the TE thermopile pinout.

The code example provided uses the following starter kit pins:

- **Thermopile VDD:** Exp Header, pin 20 (3V3)
- **Thermopile GND:** Exp Header, pin 1 (GND)
- **Thermopile I2C SDA:** Exp Header, pin 16 (PD6, I2C_SDA)
- **Thermopile I2C SCL:** Exp Header, pin 15 (PD7, I2C_SCL)
- **Piezoelectric Signal:** J101, pin 7 (PC1)
- **Piezoelectric GND:** J101, pin 15 (GND)
- **RGB LED (Red):** J101, pin 6 (PC0)
- **RGB LED (Green):** Exp Header, pin 10 (PC8)
- **RGB GND:** J101, pin 15 (GND)

(Note: Be sure not to exceed the microcontroller's maximum current draw specification. A resistor should be put in series with each LED to limit current. The resistor value can be chosen appropriately based on the desired LED brightness.)

Set Up Mechanical

The project's mechanical design will vary depending on the application and is beyond the scope of this reference design.

Calibrate and Tune

Differences in environment, distance/proximity to the measurement target, and target location of the measurement (forehead, wrist, palm, etc.) can make a difference in the detected temperature. An offset (in degrees) is applied in the code to convert from a typical wrist temperature detected by the thermopile to an internal body temperature. This offset can easily be changed in the compiled code to account for such differences. In addition, the design could be modified mechanically to accommodate different measurement targets such as the forehead.

Using the Device

Once the unit is assembled and coded per the instructions above, the device is ready to use. The temperature sensor will turn on automatically when power is applied. Once powered, the device waits for a body's proximity to be detected per the flowchart in (**Figure 1**). The temperature is then measured per the same flowchart, and results are displayed. The device then waits for the next user/proximity detection.

Expanded Features

The following enhancements could be added to the project in the future:

- **Separate Proximity Sensor:** A separate proximity sensor could inform a user of their distance from the sensor to provide a more accurate and consistent measurement.
- **Enhanced Features**: Adding a larger display, better speaker, and other enhanced features such as voice feedback could further improve the device's usability.

Author Bio

Dave Hoover, VP Advanced Technology
With nine U.S. patents to his name and over 25 years of experience in the wireless industry, Dave Hoover has the development experience to make your project a success. He has successfully brought many M2M and cellular phone projects to market while managing large projects and working with engineering teams across the globe. He can adapt to your tools and processes or implement known best practices to deliver your project on time, on spec and on budget.