



DesignNews

C++ Primer for Embedded Systems

Day 4:

Embedded C++: Interfacing with the STM32 HAL

Sponsored by

DigiKey



Webinar Logistics

- Turn on your system sound to hear the streaming presentation.
- If you have technical problems, click “Help” or submit a question asking for assistance.
- Participate in ‘Attendee Chat’ by maximizing the chat widget in your dock.

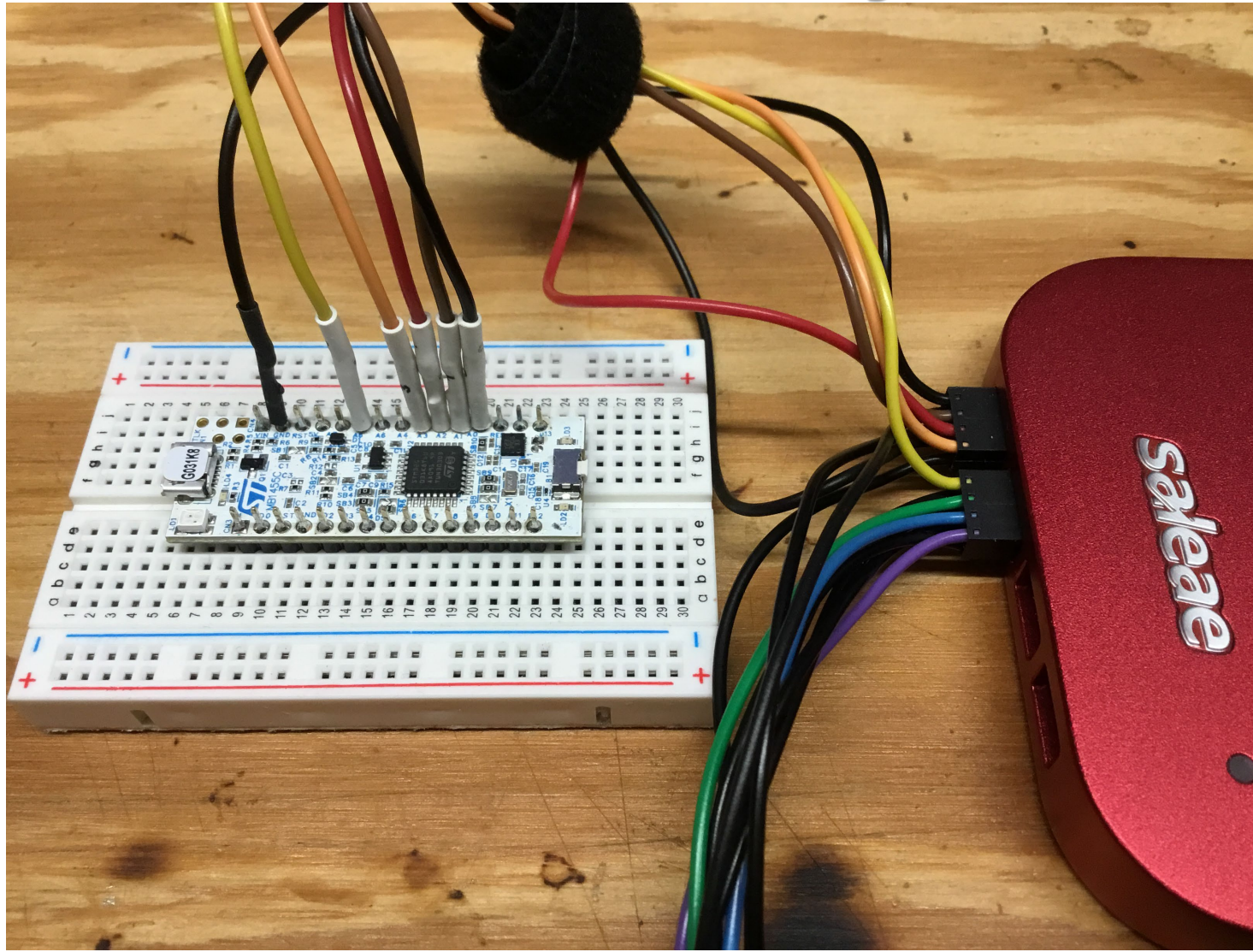


Fred Eady

Visit 'Lecturer Profile' in your console for more details.

AGENDA

- **Create an STMCubeIDE C++ GPIO Project**



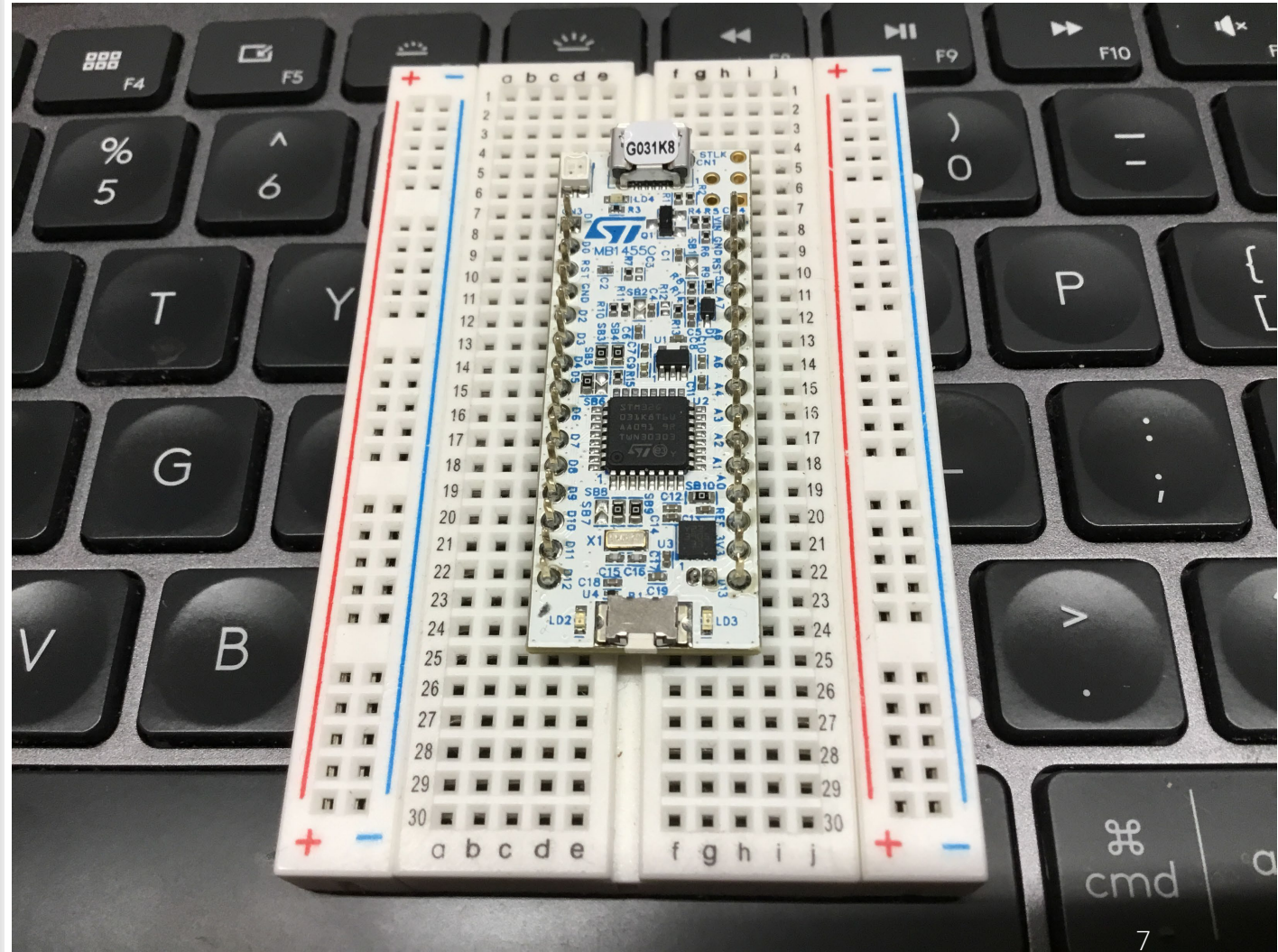
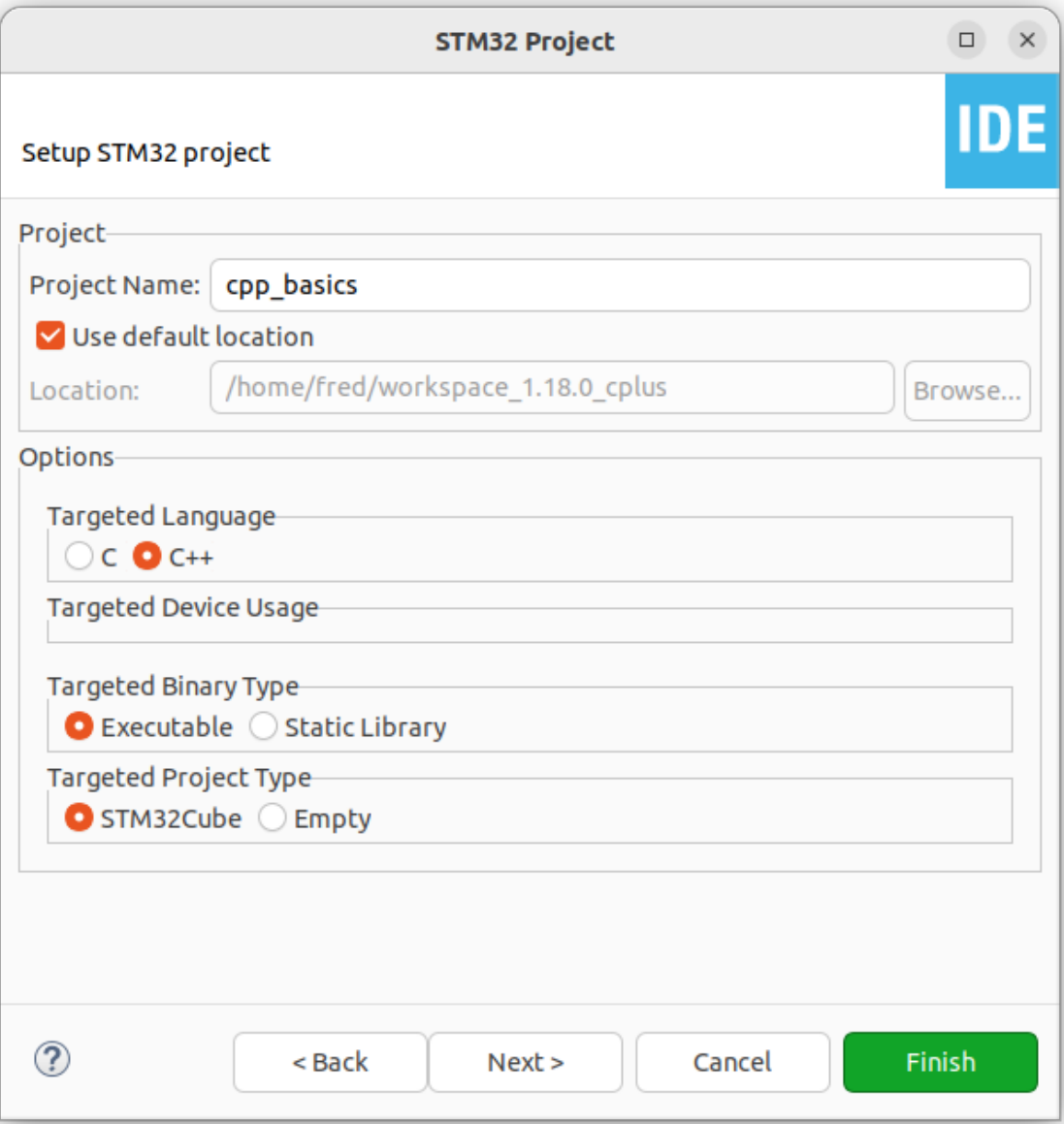
STM32Gpio Class Components

```
14 class STM32Gpio // Class Keyword - Class Name
15 {
16 public: // Access specifier - Members (attributes and methods) accessible outside of the class
17     STM32Gpio(GPIO_TypeDef* port,uint16_t pin); // Constructor
18
19     void SetPin(); // Methods (Functions)
20     void ClrPin();
21     void TogPin();
22     bool ReadPin();
23
24 private: // Access specifier - Members not accessible outside of the class
25     GPIO_TypeDef* stm32port; // Attributes
26     uint16_t stm32pin;
27 };
```

STM32Gpio Class Components

```
9 #include "STM32Gpio.hpp"
10
11 STM32Gpio::STM32Gpio(GPIO_TypeDef* port, uint16_t pin) // Constructor parameter list
12 : // Parameter list/Member Initializer Delimiter
13   stm32port(port), // Member Initializer list
14   stm32pin(pin)
15 {}
16
17 // The :: in each function declaration is known as the scope resolution operator
18
19 void STM32Gpio::SetPin()
20 {
21     return HAL_GPIO_WritePin(stm32port, stm32pin, GPIO_PIN_SET);
22 }
23
24 void STM32Gpio::ClrPin()
25 {
26     return HAL_GPIO_WritePin(stm32port, stm32pin, GPIO_PIN_RESET);
27 }
28
29 void STM32Gpio::TogPin()
30 {
31     return HAL_GPIO_TogglePin(stm32port, stm32pin);
32 }
33
34 bool STM32Gpio::ReadPin()
35 {
36     return HAL_GPIO_ReadPin(stm32port, stm32pin);
37 }
```

Create a C++ Project Named cpp_basics



Configure the C++ Project for .c and .h File Generation

The screenshot shows the STM32CubeIDE Project Manager interface. The left sidebar displays a project tree with the following structure:

- blinkit_main_rename
- IDE cpp_basics
 - Includes
 - Core
 - Drivers
 - MX cpp_basics.ioc
 - STM32G031K8TX_FL

The main area is titled "cpp_basics.ioc - Project Manager" and is divided into two tabs: "Pinout & Configuration" and "Clock Configuration". The "Pinout & Configuration" tab is active and shows the following settings:

Project

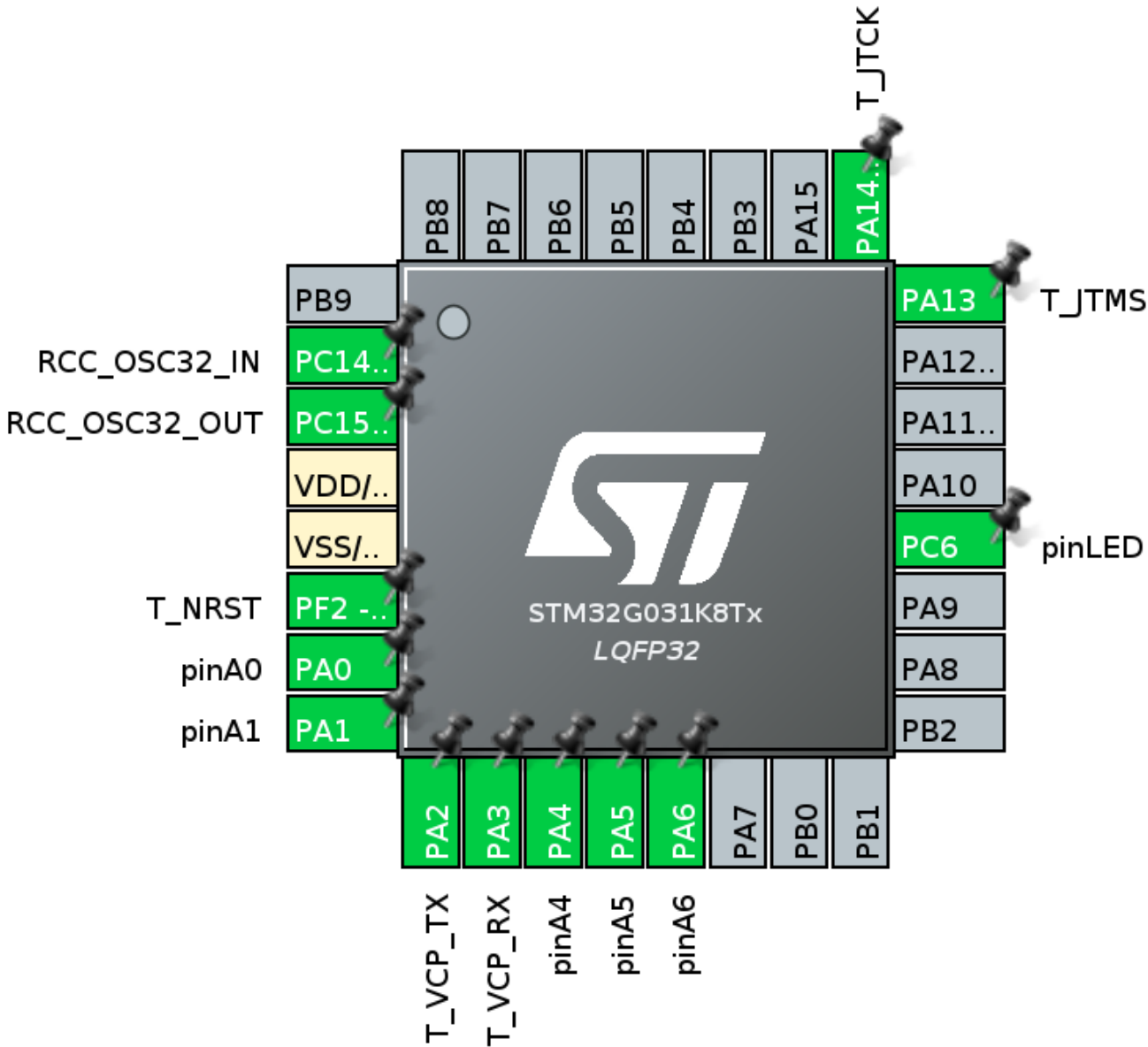
- STM32Cube MCU packages and embedded software packs
 - Copy all used libraries into the project folder
 - Copy only the necessary library files
 - Add necessary library files as reference in the toolchain project configuration file

Code Generator

Generated files

- Generate peripheral initialization as a pair of '.c/.h' files per peripheral
- Backup previously generated files when re-generating
- Keep User Code when re-generating
- Delete previously generated files when not re-generated

Configure the NUCLEO-G031K8 Hardware



```

59  /* Private defines -----
60  #define T_NRST_Pin GPIO_PIN_2
61  #define T_NRST_GPIO_Port GPIOF
62  #define pinA0_Pin GPIO_PIN_0
63  #define pinA0_GPIO_Port GPIOA
64  #define pinA1_Pin GPIO_PIN_1
65  #define pinA1_GPIO_Port GPIOA
66  #define T_VCP_TX_Pin GPIO_PIN_2
67  #define T_VCP_TX_GPIO_Port GPIOA
68  #define T_VCP_RX_Pin GPIO_PIN_3
69  #define T_VCP_RX_GPIO_Port GPIOA
70  #define pinA4_Pin GPIO_PIN_4
71  #define pinA4_GPIO_Port GPIOA
72  #define pinA5_Pin GPIO_PIN_5
73  #define pinA5_GPIO_Port GPIOA
74  #define pinA6_Pin GPIO_PIN_6
75  #define pinA6_GPIO_Port GPIOA
76  #define pinLED_Pin GPIO_PIN_6
77  #define pinLED_GPIO_Port GPIOC
78  #define T_JTMS_Pin GPIO_PIN_13
79  #define T_JTMS_GPIO_Port GPIOA
80  #define T_JTCK_Pin GPIO_PIN_14
81  #define T_JTCK_GPIO_Port GPIOA

```

Create C++ Header File - c_to_cpp_bridge.hpp

The screenshot shows the STM32CubeIDE interface. The 'New Header File' dialog is the central focus, with the following fields:

- Header File: Create a new header file.
- Source folder:
- Header file:
- Template:

At the bottom of the dialog are and buttons.

In the background, the Project Explorer shows a tree view with 'Inc' selected. A context menu is open over 'Inc', with 'Header File' highlighted. A tooltip for 'Header File' reads: 'Header File: Create a new header file Source Folder'.

Create C++ Source File - c_to_cpp_bridge.cpp

The screenshot shows the STM32CubeIDE interface. On the left, the Project Explorer shows a project named 'blinky_main_rename' with a sub-project 'cpp_basics'. Under 'cpp_basics', there is a 'Src' folder containing 'c_to_cpp_bridge.cpp'. The main editor window shows the code for 'c_to_cpp_bridge.cpp' with the following content:

```
1 /*
2  * c_to_cpp_bridge.hpp
3  *
4  * Created on: Apr 24, 2025
5  * Author: fred
6  */
7
8 #ifndef INC_C_TO_CPP_BRIDGE_HPP
9 #define INC_C_TO_CPP_BRIDGE_HPP
10
11 // C++ Content Start
12 void runCpp(void);
13 // C++ Content End
14
15 #ifdef __cplusplus
16 extern "C"
17 {
```

The 'New Source File' dialog is open, showing the following fields:

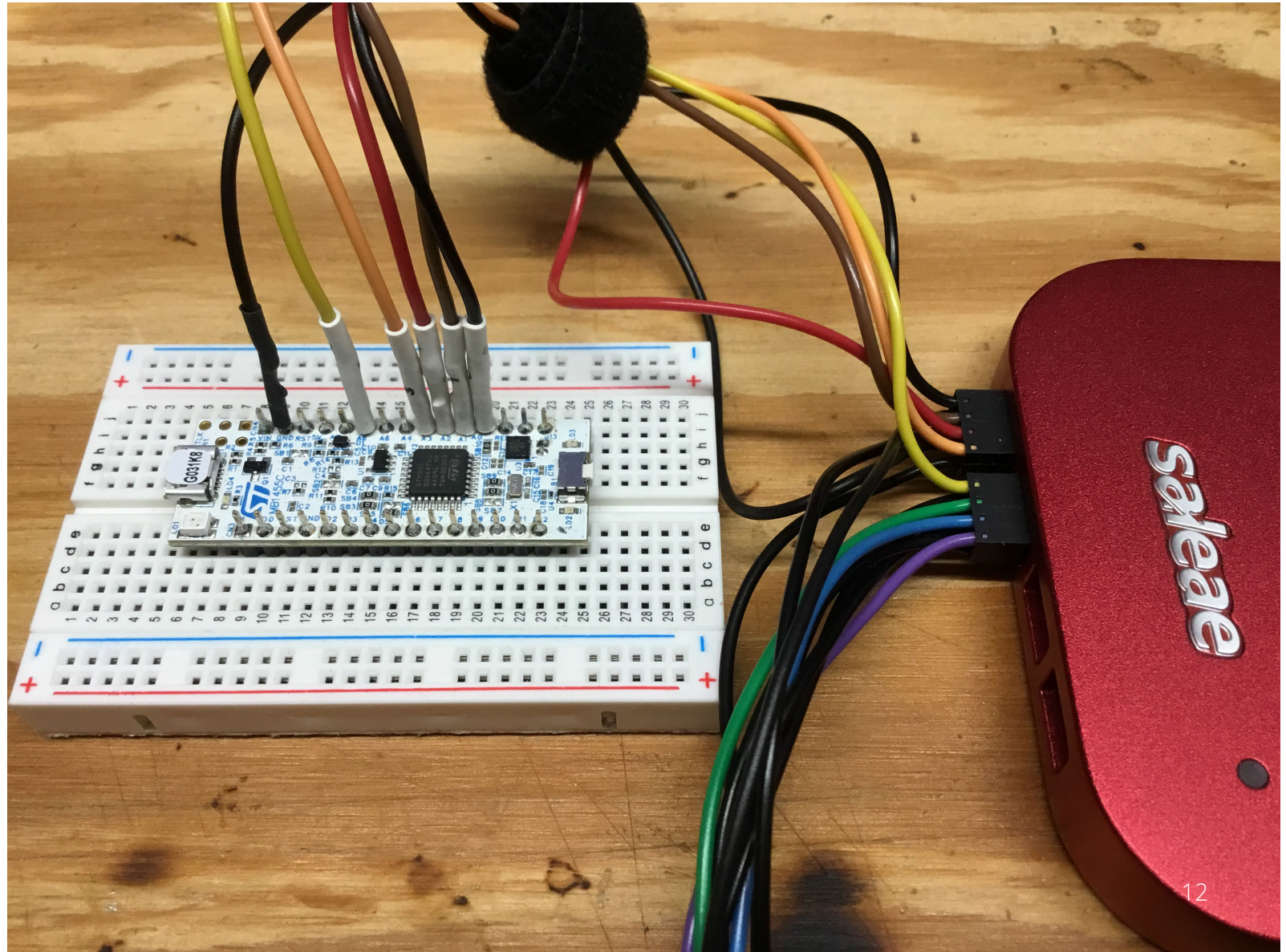
- Source folder: `cpp_basics/Core/Src` (with a 'Browse...' button)
- Source file: `c_to_cpp_bridge.cpp`
- Template: `Default C++ source template` (with a 'Configure...' button)

At the bottom of the dialog are 'Cancel' and 'Finish' buttons. A context menu is also visible over the 'Src' folder, with 'New' selected, and a sub-menu showing 'Source File' as the chosen option. A tooltip over 'Source File' reads 'Create a new source file'. The keyboard shortcut 'Ctrl+N' is shown at the bottom right of the dialog.

Populate C++ Header File - `c_to_cpp_bridge.hpp`

c_to_cpp_bridge.hpp X

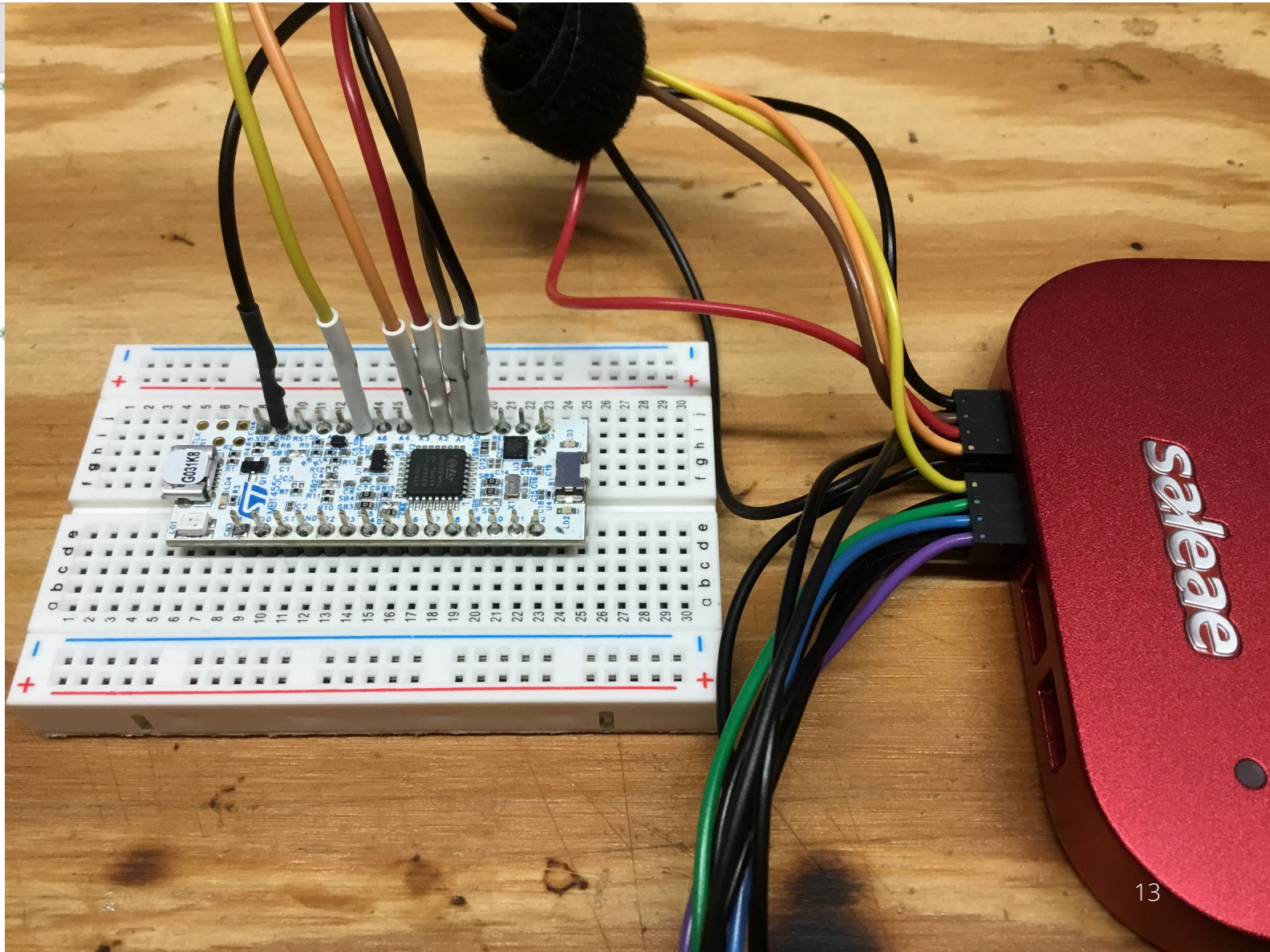
```
1 //*****
2 /** C++ Primer for Embedded Systems
3 /** c_to_cpp_bridge.hpp
4 /** Last Update: Apr 26, 2025
5 /** Author: Fred Eady
6 /** Notes:
7 //*****
8
9 #ifndef INC_C_TO_CPP_BRIDGE_HPP_
10 #define INC_C_TO_CPP_BRIDGE_HPP_
11
12 // C++ Content Start
13 void runCpp(void);
14 // C++ Content End
15
16 // C Content Start
17 #ifdef __cplusplus
18 extern "C"
19 {
20 #endif
21
22 void call_runCpp(void);
23
24 #ifdef __cplusplus
25 }
26 #endif
27 // C Content End
28
29
30
31 #endif /* INC_C_TO_CPP_BRIDGE_HPP_ */
```



Populate C++ Source File - `c_to_cpp_bridge.cpp`

c_to_cpp_bridge.cpp ×

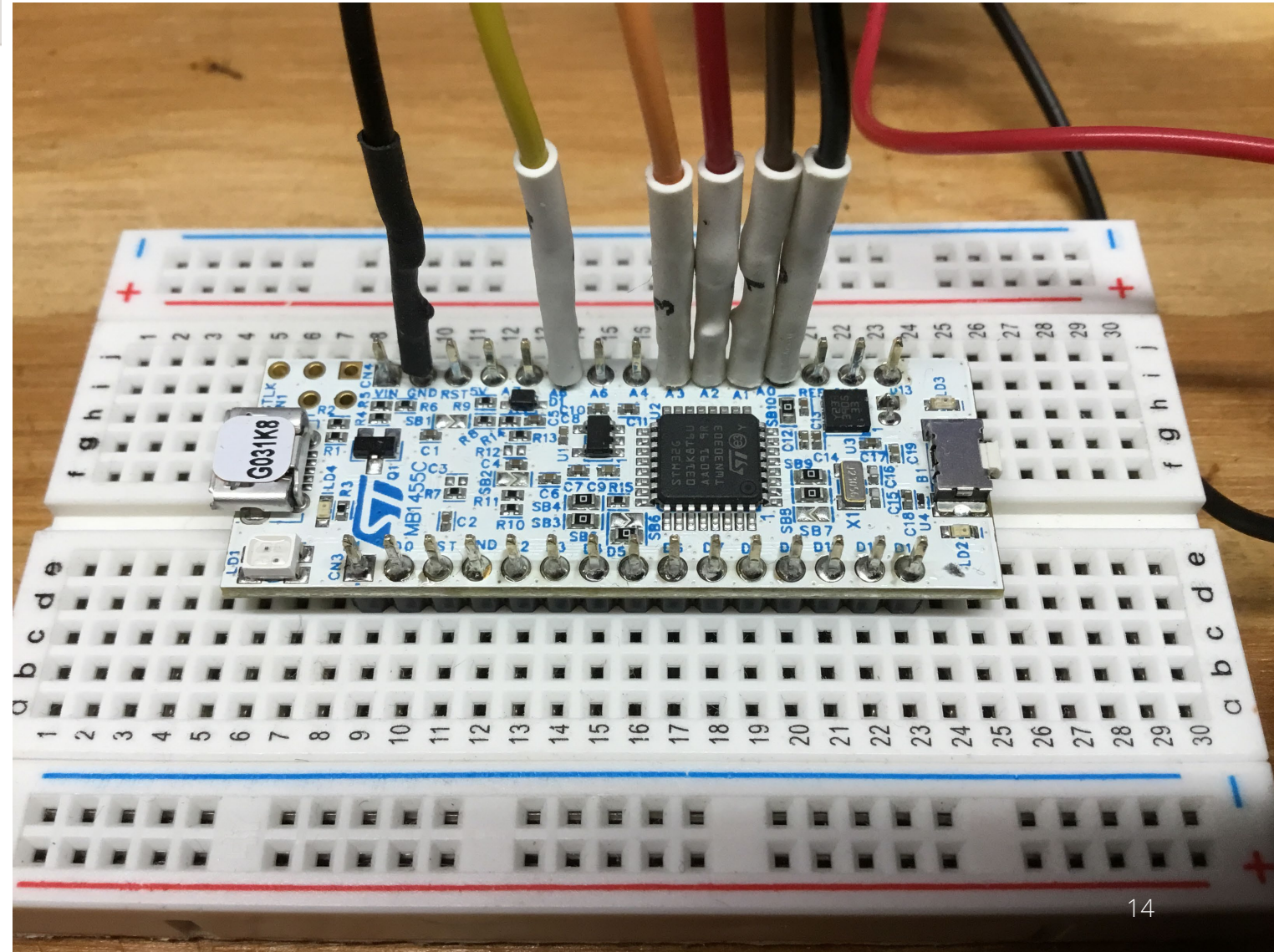
```
1 //*****
2 /** C++ Primer for Embedded Systems
3 /** c_to_cpp_bridge.cpp
4 /** Last Update: Apr 26, 2025
5 /** Author: Fred Eady
6 /** Notes:
7 /** Include .hpp file for all C++ classes
8 //*****
9
10 #include "main.h"
11
12 // Run C++ application
13 void runCpp()
14 {
15     // C++ App Code Goes Here
16 }
17
18 // Function calls from main.c
19 extern "C"
20 {
21 void call_runCpp()
22 {
23     runCpp();
24 }
25 }
```



Create the STM32Gpio Class

STM32Gpio.hpp

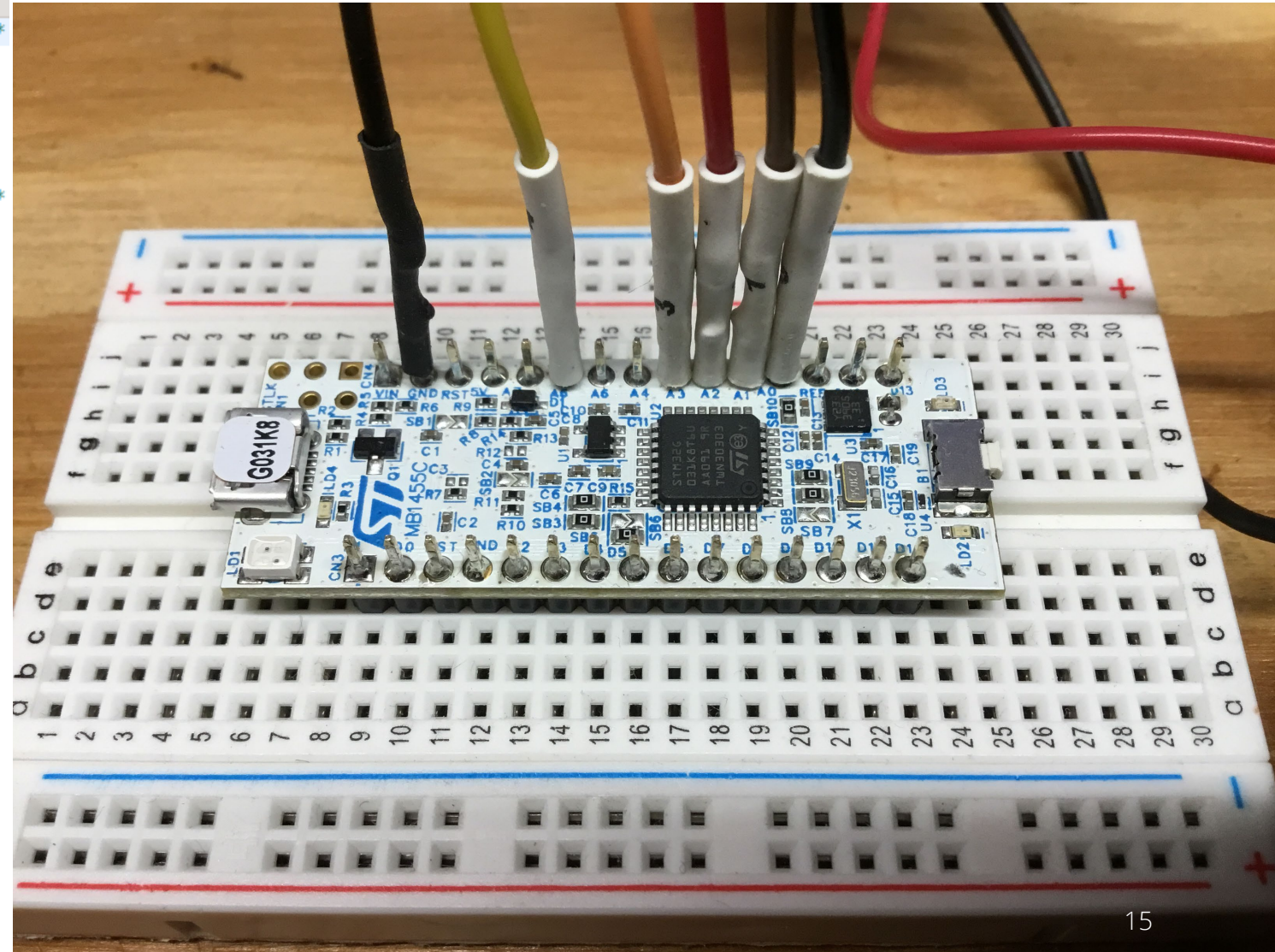
```
1 //*****
2 /* C++ Primer for Embedded Systems
3 /* c_to_cpp_bridge.hpp
4 /* Last Update: Apr 27, 2025
5 /* Author: Fred Eady
6 /* Notes:
7 //*****
8
9 #ifndef INC_STM32GPIO_HPP
10 #define INC_STM32GPIO_HPP
11
12 #include "stm32g0xx_hal.h"
13
14 class STM32Gpio
15 {
16 public:
17     STM32Gpio(GPIO_TypeDef* port, uint16_t pin);
18
19     void SetPin();
20     void ClrPin();
21     void TogPin();
22     bool ReadPin();
23
24 private:
25     GPIO_TypeDef* stm32port;
26     uint16_t stm32pin;
27 };
```



Create the STM32Gpio Class

STM32Gpio.cpp x

```
1 //*****
2  /** C++ Primer for Embedded Systems
3  /** STM32Gpio.cpp
4  /** Last Update: Apr 27, 2025
5  /** Author: Fred Eady
6  /** Notes:
7  /*******
8
9  #include "STM32Gpio.hpp"
10
11 STM32Gpio::STM32Gpio(GPIO_TypeDef* port,uint16_t pin)
12 :
13   stm32port(port),
14   stm32pin(pin)
15 {}
16
17 void STM32Gpio::SetPin()
18 {
19   return HAL_GPIO_WritePin(stm32port,stm32pin,GPIO_PIN_SET);
20 }
21
22 void STM32Gpio::ClrPin()
23 {
24   return HAL_GPIO_WritePin(stm32port,stm32pin,GPIO_PIN_RESET);
25 }
26
27 void STM32Gpio::TogPin()
28 {
29   return HAL_GPIO_TogglePin(stm32port,stm32pin);
30 }
31
32 bool STM32Gpio::ReadPin()
33 {
34   return HAL_GPIO_ReadPin(stm32port,stm32pin);
35 }
```



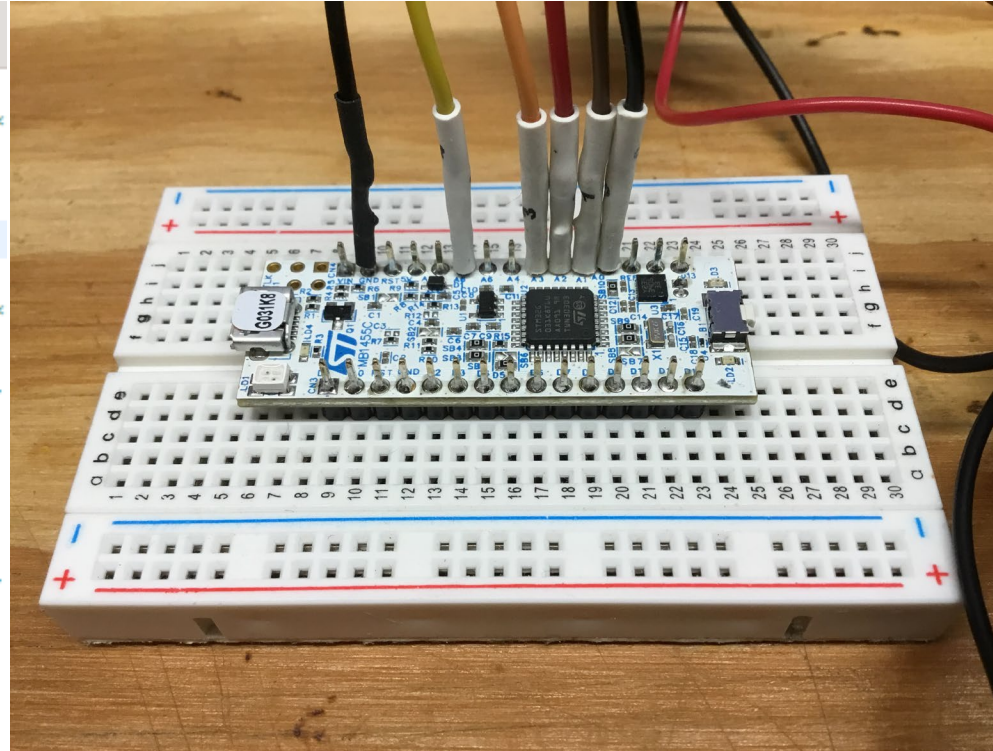
Modify main.c

main.c ×

```

1  /* USER CODE BEGIN Header */
2  /**
3   ** C++ Primer for Embedded Systems
4   ** Author: Fred Eady
5   ** Last Update: 04-28-2025
6   ** Notes:
7   **
8   ** USER CODE END Header */
9   /* Includes -----
10  #include "main.h"
11  #include "usart.h"
12  #include "gpio.h"
13
14  /* Private includes -----
15  /* USER CODE BEGIN Includes */
16  #include "c_to_cpp_bridge.hpp"
17  /* USER CODE END Includes */
18

```



main.c ×

```

76  /* USER CODE END SysInit */
77
78  /* Initialize all configured perip
79  MX_GPIO_Init();
80  MX_USART2_UART_Init();
81  /* USER CODE BEGIN 2 */
82  call_runCpp();
83  /* USER CODE END 2 */
84
85  /* Infinite loop */
86  /* USER CODE BEGIN WHILE */
87  while (1)
88  {
89      //HAL_GPIO_TogglePin(pinLED_GP
90      //HAL_Delay(100);
91      /* USER CODE END WHILE */
92
93      /* USER CODE BEGIN 3 */
94

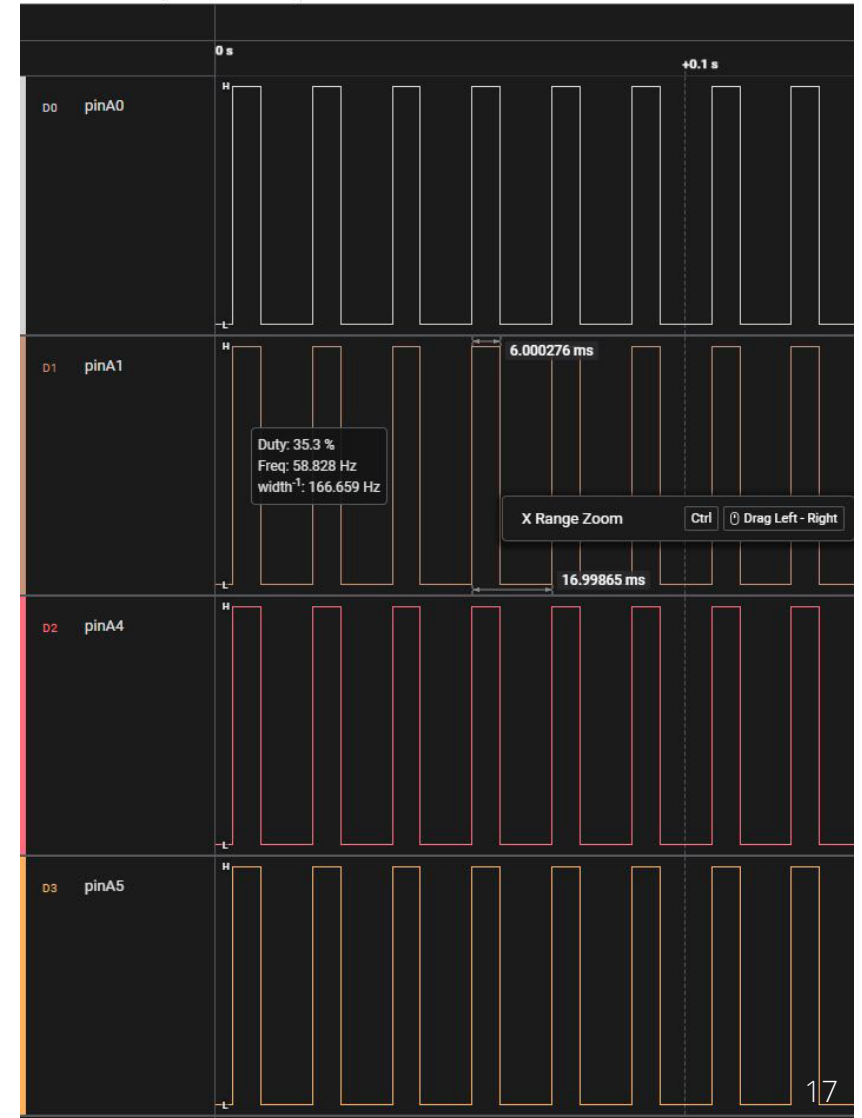
```

Run It

c_to_cpp_bridge.cpp x

```
1 //*****
2 /* C++ Primer for Embedded Systems
3 /* c_to_cpp_bridge.cpp
4 /* Last Update: Apr 26, 2025
5 /* Author: Fred Eady
6 /* Notes:
7 /* Include .hpp file for all C++ classes used in the c_to_cpp_bridge.cpp file
8 //*****
9
10 #include "main.h"
11 #include "STM32Gpio.hpp"
12
13 // Run C++ application
14 void runCpp()
15 {
16     STM32Gpio iopins(GPIOA, pinA0_Pin|pinA1_Pin|pinA4_Pin|pinA5_Pin);
17     STM32Gpio led(pinLED_GPIO_Port, pinLED_Pin);
18     while(1)
19     {
20         iopins.SetPin();
21         HAL_Delay(5);
22         iopins.ClrPin();
23         HAL_Delay(10);
24     }
25 }
26
27 // Function calls from main.c
28 extern "C"
29 {
30     void call_runCpp()
31     {
32         runCpp();
33     }
34 }
```

File Edit View Capture Measure Help



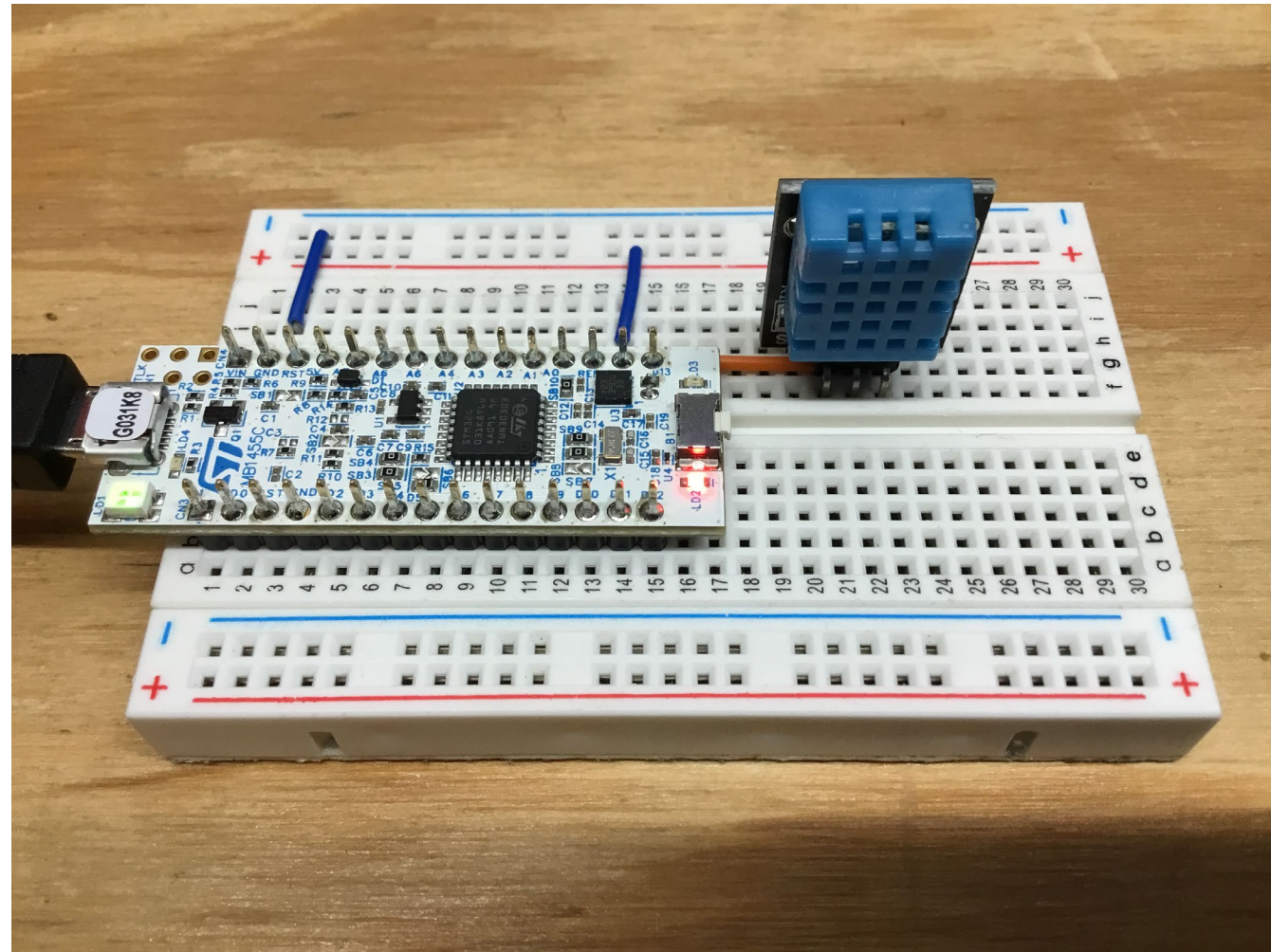
Next Time...

MORE TO COME..

Thank you for attending!!!

Please consider the resources below:

- [Today's Download Package](#)





DesignNews

Thank You

Sponsored by

DigiKey

