CEC Continuing Education Center

DesignNews

Bare Metal C Programming for STM32 Devices

Day 3:

Bare Metal NUCLEO-F207ZG :

Mongoose Network Library API Primer

Sponsored by

DigiKey

informa markets

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

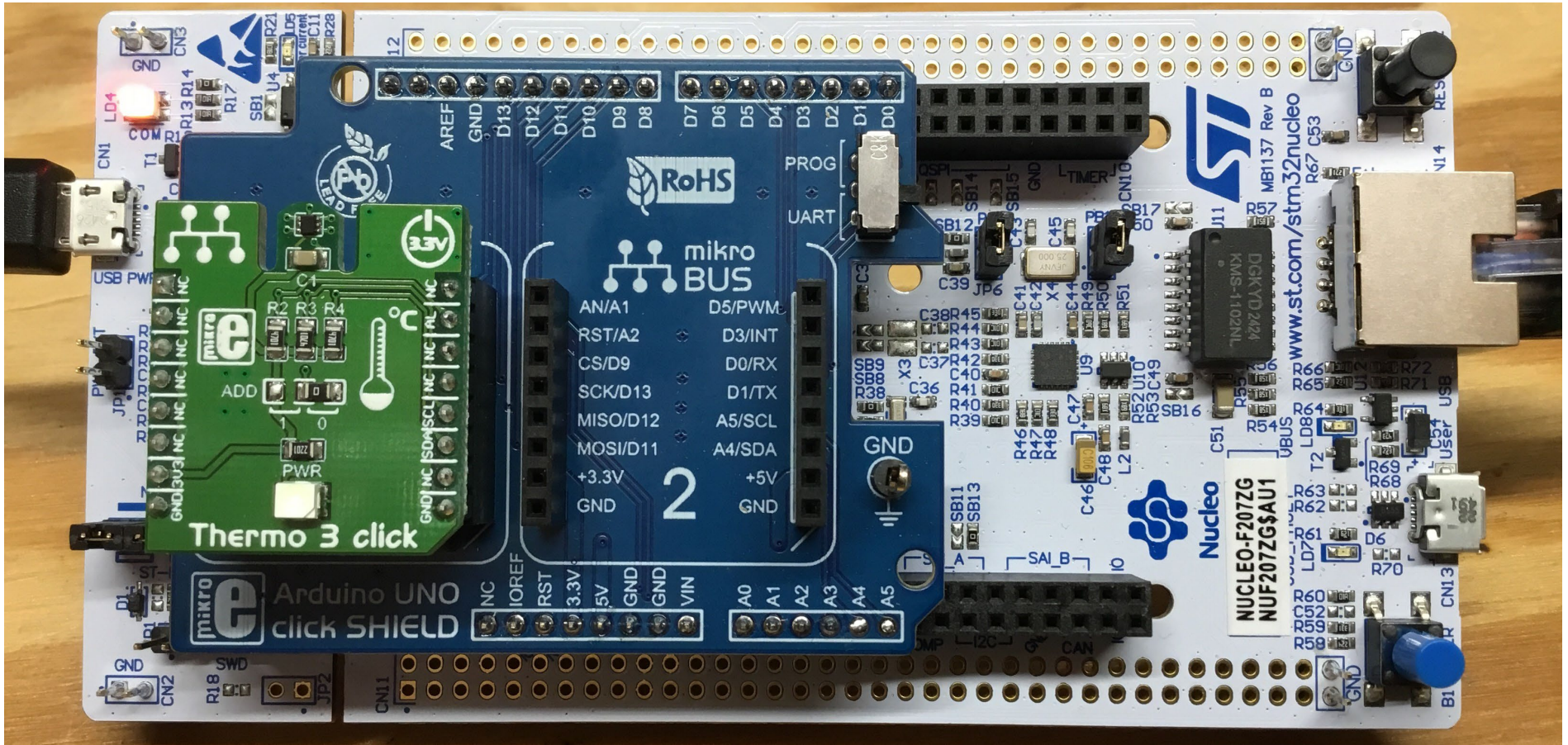
Visit ‘Lecturer Profile’ in your console for more details.

CEC Continuing Education Center

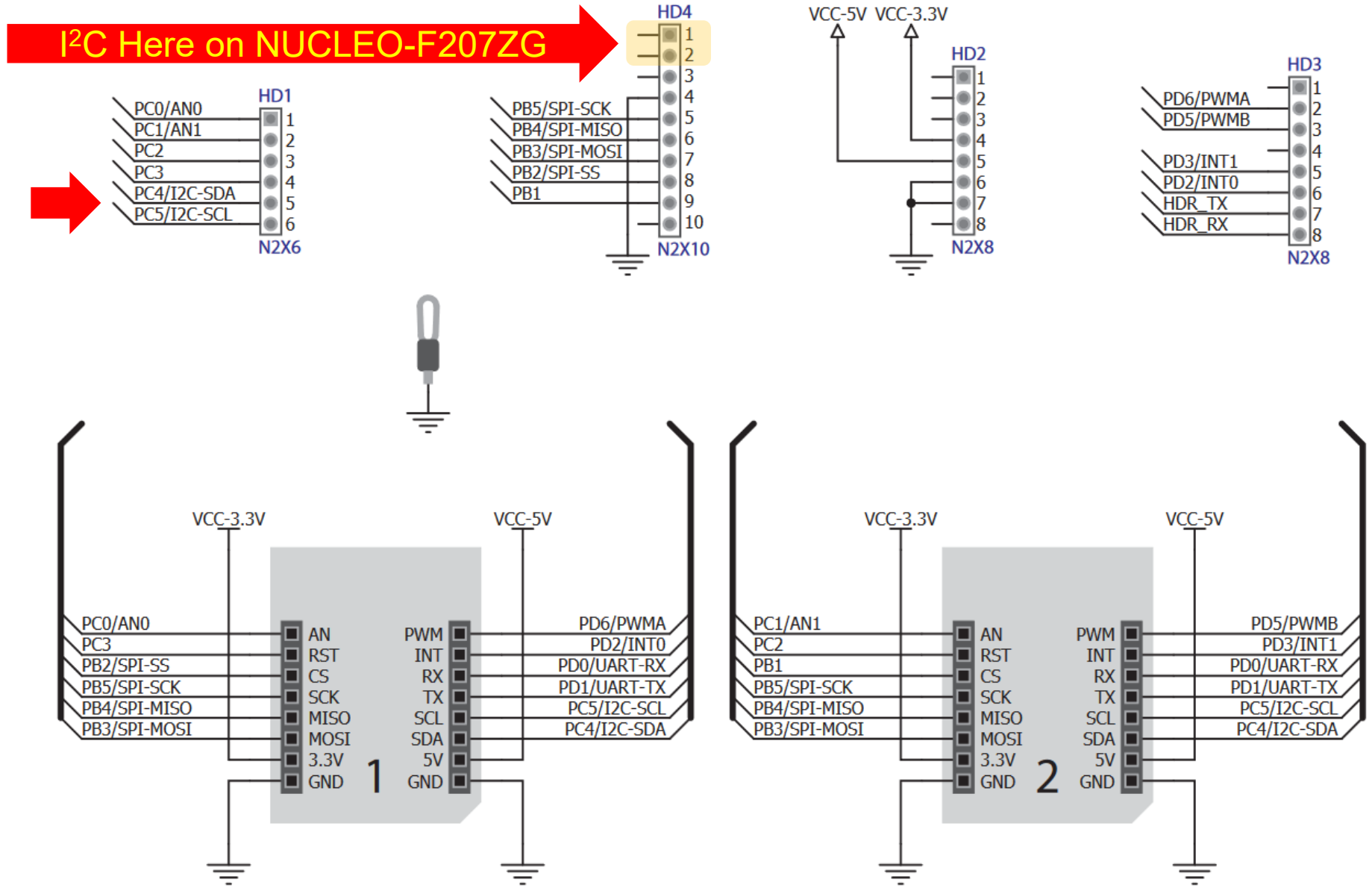
AGENDA

DigiKey

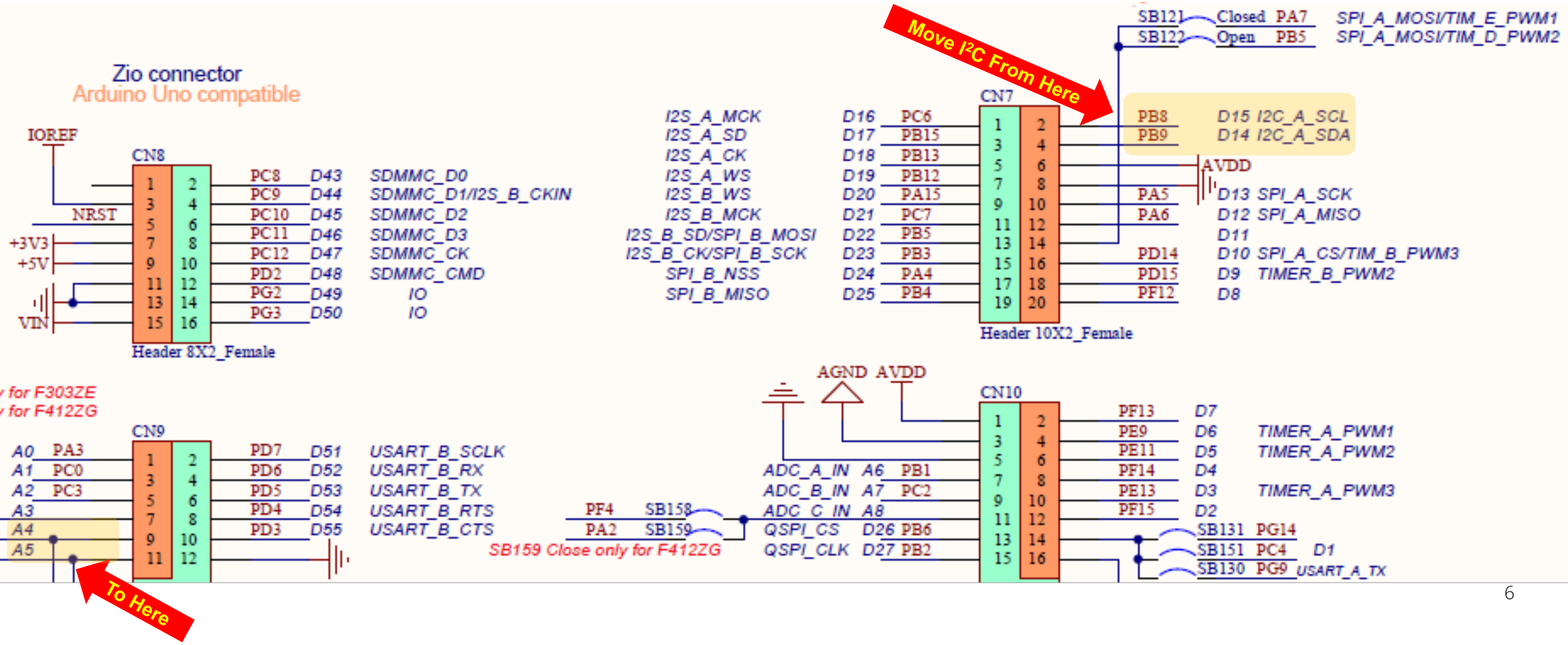
- Modify the Arduino UNO click SHIELD
- Thermo 3 click Driver
- Integrate Mongoose Using the STM32 Mongoose Package



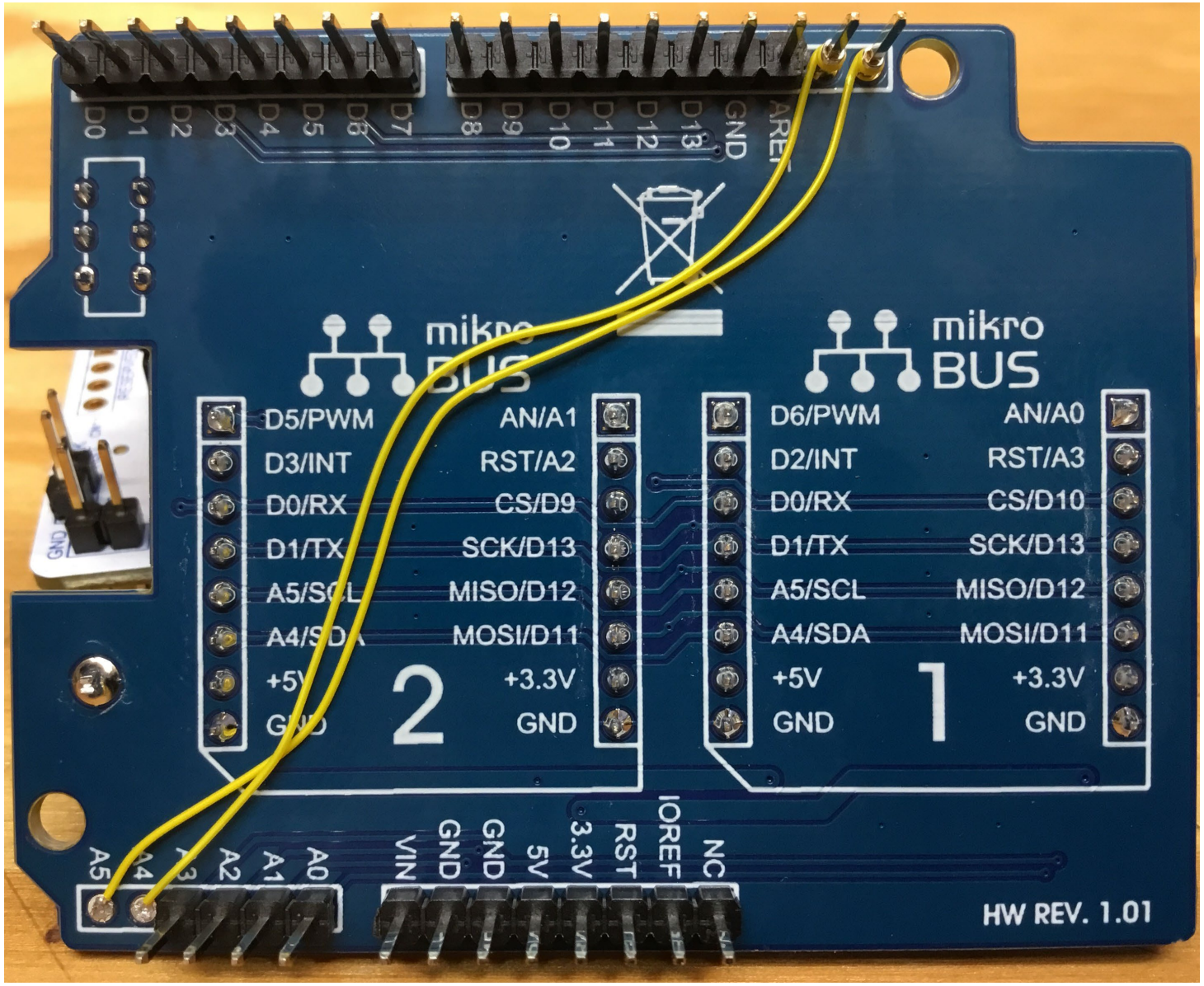
Arduino UNO click SHIELD I²C Hardware Modification



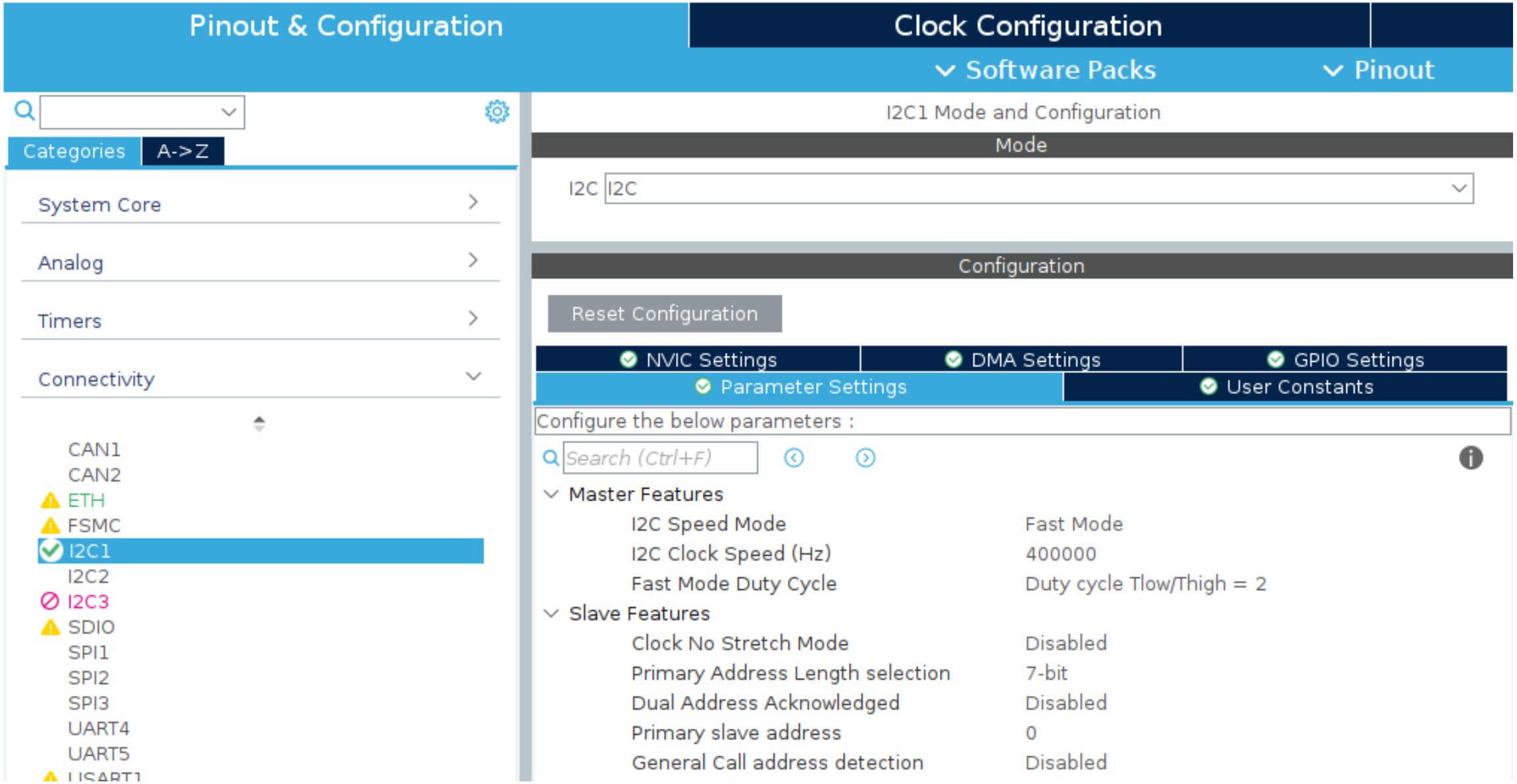
Arduino UNO click SHIELD I²C Hardware Modification



Arduino UNO click SHIELD I²C Hardware Modification



Configure and Activate I2C1



Configure and Activate I2C1

Pinout & Configuration | Clock Configuration | Project Manager

Software Packs

Categories | A->Z

System Core

- DMA
- GPIO
- IWDG
- NVIC
- RCC
- SYS
- WWDG

Analog

Timers

Connectivity

Multimedia

GPIO Mode ar

Config

Group By Peripherals

RCC | SYS | US | NVIC

GPIO | Single Mapped S | I2C

Search Signals

Search (Ctrl+F)

Show only Modified Pins

Pin Na...	Signal on ...	GPIO outp...	GPIO mode	GPIO Pull-u...	Maximum ...	User Label	Modified
PB8	I2C1_SCL	n/a	Alternate ...	No pull-up ...	High	A5_SCL	☑
PB9	I2C1_SDA	n/a	Alternate ...	No pull-up ...	High	A4_SDA	☑

Assemble tmp102.c

tmp102.c

Core > Src > C tmp102.c > ...

```
20  //*****************************************************************************
21  //* INCLUDES
22  //*****************************************************************************
23  #include "tmp102.h"
24  //*****************************************************************************
25  //* TYPEDEFS
26  //*****************************************************************************
27  extern I2C_HandleTypeDef hi2c1;
28  //*****************************************************************************
29  //* TMP102 VARIABLES
30  //*****************************************************************************
31  uint8_t txBuf[8];       // I2C transmit buffer
32  uint8_t rxBuf[8];       // I2C receive buffer
33  uint16_t tempBuf;       // TMP102 temperature buffer
34  float tempC;            // temperature in C
35  char registerName[18];  // holding array for TMP102 register name
36  //*****************************************************************************
37  //* TMP102 and I2C CONSTANTS
38  //* 7-bit address = 0x48
39  //* 8-bit address = 0x90 write
40  //* 8-bit address = 0x91 read
41  //*****************************************************************************
42  const unsigned char tmp102_addr = 0x90;     // TMP102 I2C 7-bit address = 0x48
```

Assemble tmp102.c

tmp102.c

Core > Src > tmp102.c > ...

```c
//*****************************************************************************
// Select TMP102 Register
//*****************************************************************************
void tmp102_select_reg(uint8_t reg)
{
    //*************************************************************************
    // Write Pointer Register with location of the temperature register
    // P7 P6 P5 P4 P3 P2 P1 P0
    // 0  0  0  0  0  0  0  0   Temperature Register
    // 0  0  0  0  0  0  0  1   Configuration Register
    // 0  0  0  0  0  0  1  0   Temperature Low Register
    // 0  0  0  0  0  0  1  1   Temperature High Register
    //*************************************************************************
    txBuf[0] = reg;
    HAL_I2C_Master_Transmit(hi2c: &hi2c1, DevAddress: tmp102_addr, pData: txBuf, Size: 0x01, Timeout: HAL_MAX_DELAY);
    switch (txBuf[0])
    {
    case 0x00:
        sprintf(registerName,"Temperature");
    break;
    case 0x01:
        sprintf(registerName,"Configuration");
    break;
    case 0x02:
        sprintf(registerName,"Temperature Low");
    break;
    case 0x03:
        sprintf(registerName,"Temperature High");
    break;
    }
}
```

Assemble tmp102.c

C tmp102.c

Core > Src > C tmp102.c > tmp102_select_reg

```
76  //*****************************************************************************
77  // Read the temperature
78  //*****************************************************************************
79  float tmp102_read_temperature(void)
80  {
81      HAL_I2C_Master_Receive(hi2c: &hi2c1, DevAddress: tmp102_addr, pData: rxBuf, Size: 2, Timeout: HAL_MAX_DELAY);
82      MG_INFO((fmt: "INFO: 2 bytes were read-> rxBuf[0] 0x%02X -- rxBuf[1] 0x%02X",rxBuf[0],rxBuf[1]));
83      //*************************************************************************
84      // Convert temperature bytes to 16-bit value
85      //*************************************************************************
86      tempBuf = (rxBuf[0] << 4) | (rxBuf[1] >> 4);
87      MG_INFO((fmt: "INFO: 16-bit value = 0x%04X",tempBuf));
88      //*************************************************************************
89      // Compute and print the temperature
90      //*************************************************************************
91      if(tempBuf & (1 << 11))
92      {
93          tempBuf |= 0xF800;
94      }
95      tempC = (float)tempBuf * 0.0625;
96      return tempC;
97  }
```

Assemble tmp102.h

tmp102.h

Core > Inc > tmp102.h > ...

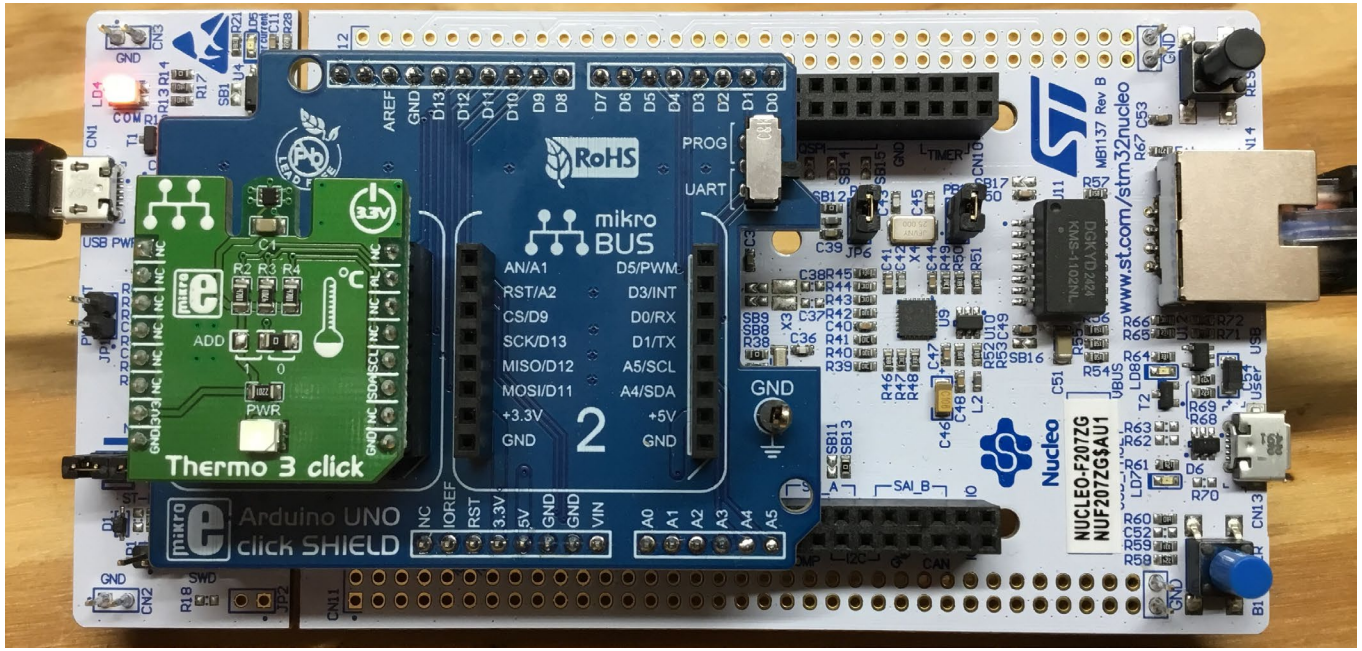
```
17  #include "main.h"
18  #include "i2c.h"
19  #include "stm32f2xx_hal_def.h"
20  #include "stm32f2xx_hal_i2c.h"
21  #include "usart.h"
22  #include "gpio.h"
23  #include <stdio.h>
24  #include <stdint.h>
25  #include <string.h>
26  #include "mongoose.h"
27
28  void tmp102_select_reg(uint8_t reg);
29  float tmp102_read_temperature(void);
30
31  //*****************************************************************************
32  // Write Pointer Register with location of the temperature register
33  // P7 P6 P5 P4 P3 P2 P1 P0
34  // 0  0  0  0  0  0  0  0   Temperature Register
35  // 0  0  0  0  0  0  0  1   Configuration Register
36  // 0  0  0  0  0  0  1  0   Temperature Low Register
37  // 0  0  0  0  0  0  1  1   Temperature High Register
38  //*****************************************************************************
39  #define tempreg     0x00
40  #define confreg     0x01
41  #define tempreglo   0x02
42  #define tempreghi   0x03
```

Assemble main.c – Main Loop

main.c 9+

Core > Src > main.c > main

```
213   /* Initialize all configured peripherals */
214   MX_GPIO_Init();
215   MX_ETH_Init();
216   MX_USART3_UART_Init();
217   MX_USB_OTG_FS_PCD_Init();
218   MX_I2C1_Init();
219   /* USER CODE BEGIN 2 */
220   NVIC_EnableIRQ(IRQn: ETH_IRQn);
221   struct mg_mgr mgr;          // Mongoose event manager
222   mg_log_set(MG_LL_DEBUG);  // Set log level to debug
223   mg_mgr_init(&mgr);          // Initialise event manager
224   //* Kick off the server
225   mg_listen(&mgr, url: s_lsn,fn: server_fn,fn_data: NULL);
226  /* USER CODE END 2 */
227
228   /* Infinite loop */
229   /* USER CODE BEGIN WHILE */
230   while (1)
231   {
232     mg_mgr_poll(&mgr,ms: 0);
233     /* USER CODE END WHILE */
234
235     /* USER CODE BEGIN 3 */
236   }
237   /* USER CODE END 3 */
238 }
```

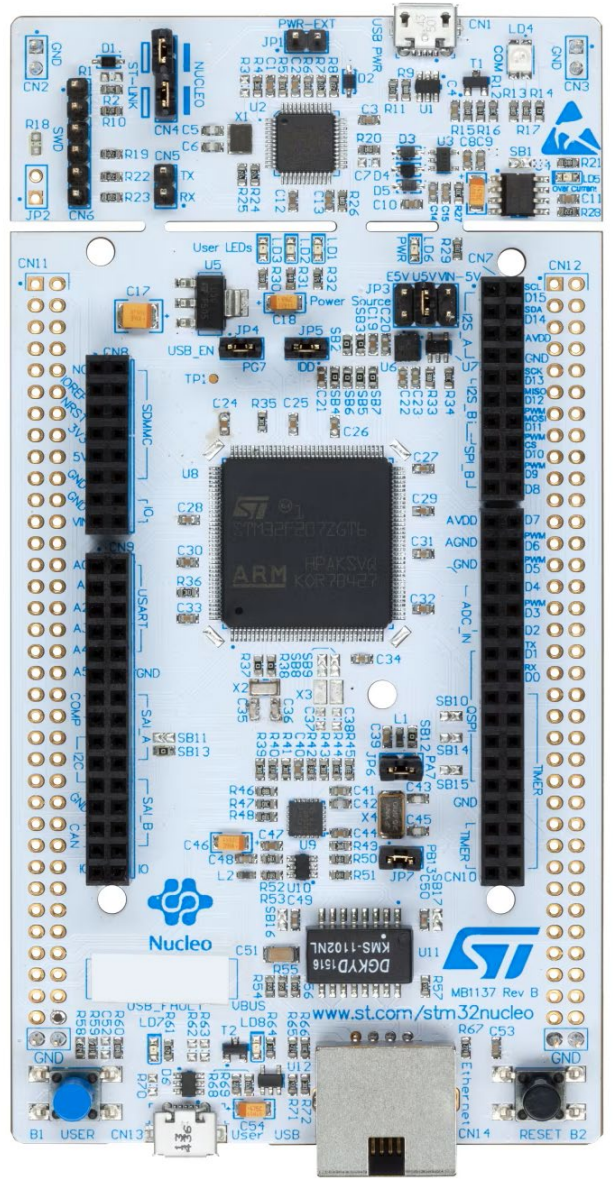


Assemble main.c – Redirect printf

main.c 9+

Core > Src > main.c > server_fn

```
50  /* USER CODE BEGIN PV */
51  float temptosend;
52  //* The tcp://0.0.0.0:8088 indicates that the server listens for connections
53  //* on all available network interfaces. 8088 is the destination port.
54  static const char *s_lsn = "tcp://0.0.0.0:8088";
55  /* USER CODE END PV */
56
57  /* Private function prototypes -----------------------------------------------*/
58  void SystemClock_Config(void);
59  /* USER CODE BEGIN PFP */
60  /* USER CODE END PFP */
61
62  /* Private user code ---------------------------------------------------------*/
63  /* USER CODE BEGIN 0 */
64  //*****************************************************************************
65  //* REDIRECT printf TO USART3
66  //*****************************************************************************
67  #ifdef __GNUC__
68  #define PUTCHAR_PROTOTYPE int __io_putchar(int ch)
69  #else
70  #define PUTCHAR_PROTOTYPE int fputc(int ch, FILE *f)
71  #endif
72  PUTCHAR_PROTOTYPE
73  {
74    // Place your implementation of fputc here
75    // e.g., write a character to the USART3 and loop until the end of transmission
76    HAL_UART_Transmit(huart: &huart3, pData: (uint8_t *)&ch, Size: 1, Timeout: HAL_MAX_DELAY);
77    return ch;
78  }
```

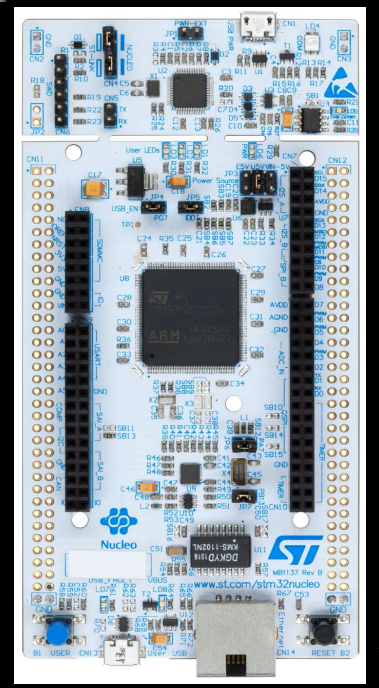


Assemble main.c – Build the Server Event Handler

main.c 9+

Core > Src > main.c > server_fn

```
80  //*****************************************************************************
81  //* SERVER EVENT HANDLER
82  //*****************************************************************************
83  static void server_fn(struct mg_connection *c, int ev, void *ev_data)
84  {
85    if (ev == MG_EV_OPEN && c->is_listening == 1)
86    {
87      MG_INFO((fmt: "INFO: SERVER IS LISTENING"));
88    }
89    else if (ev == MG_EV_ACCEPT)
90    {
91      MG_INFO((fmt: "INFO: SERVER ACCEPTED A CONNECTION"));
92    }
93    else if (ev == MG_EV_READ)
94    {
95    MG_INFO((fmt: "INFO: SERVER RECEIVED DATA: %.*s", c->recv.len, c->recv.buf));
```

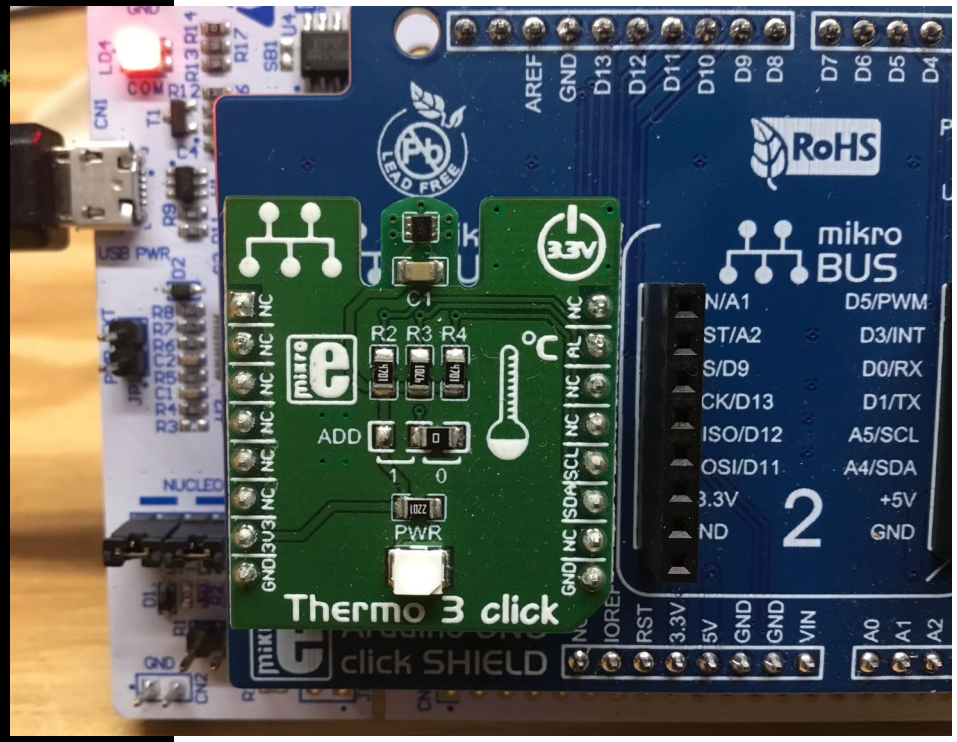


Assemble main.c – Build the Server Event Handler

main.c 9+

Core > Src > main.c > server_fn

```
96  //*****************************************************************************
97  //* SERVER EVENT HANDLER - SELECT TMP102 WORKING REGISTER
98  //*****************************************************************************
99      if(strncmp(c->recv.buf,"SELECT REGISTER TEMP",21) == 0)
100     {
101       tmp102_select_reg(reg: 0);
102       MG_INFO((fmt: "INFO: TMP102 TEMPERATURE REGISTER SELECTED!"));
103     }
104     else if(strncmp(c->recv.buf,"SELECT REGISTER CONFIG",23) == 0)
105     {
106       tmp102_select_reg(reg: 1);
107       MG_INFO((fmt: "INFO: TMP102 CONFIGURATION REGISTER SELECTED!"));
108     }
109     else if(strncmp(c->recv.buf,"SELECT REGISTER TEMPLO",23) == 0)
110     {
111       tmp102_select_reg(reg: 2);
112       MG_INFO((fmt: "INFO: TMP102 TEMPERATURE LO REGISTER SELECTED!"));
113     }
114     else if(strncmp(c->recv.buf,"SELECT REGISTER TEMPHI",23) == 0)
115     {
116       tmp102_select_reg(reg: 3);
117       MG_INFO((fmt: "INFO: TMP102 TEMPERATURE HI REGISTER SELECTED!"));
118     }
```

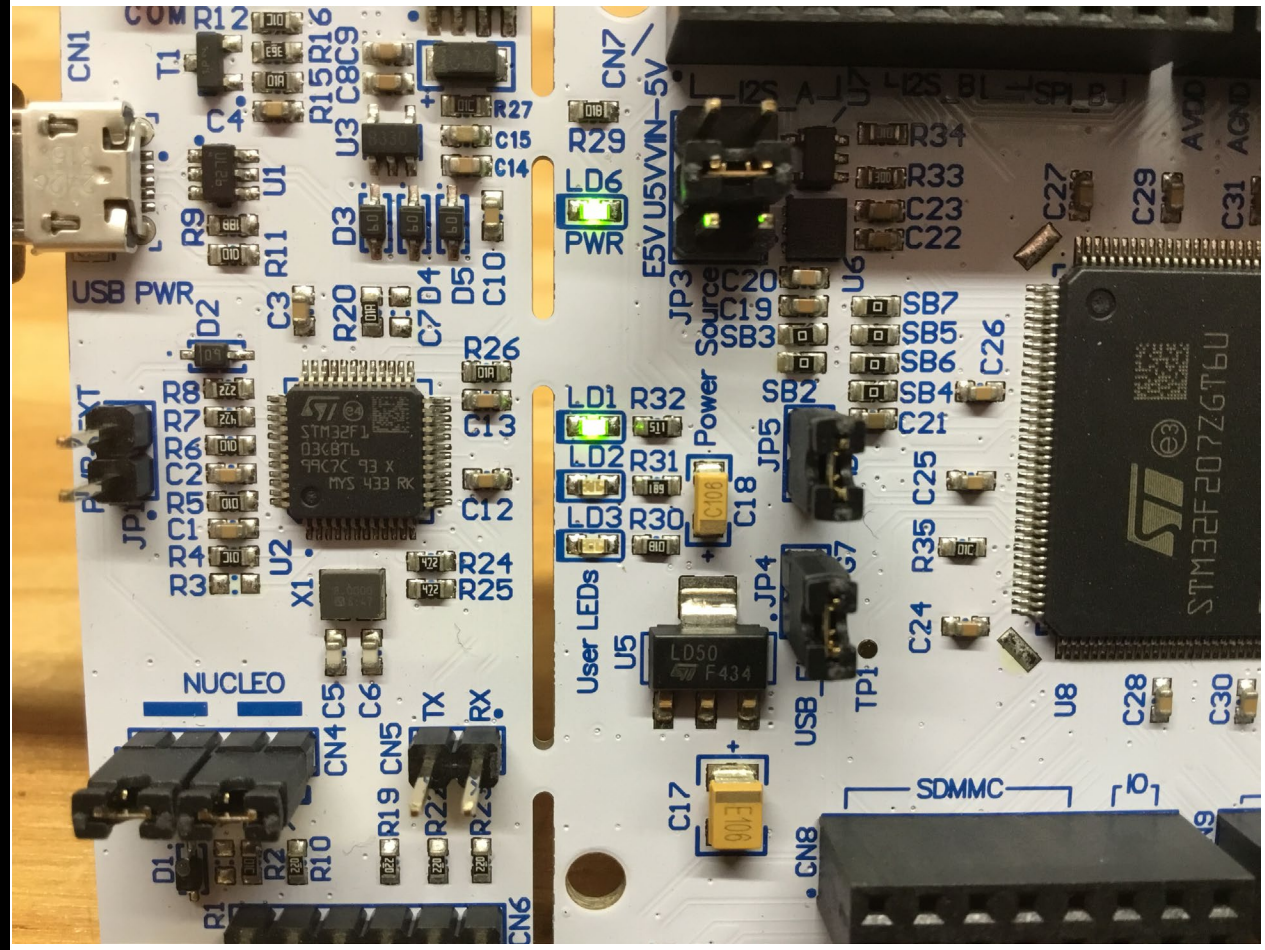


Assemble main.c – Build the Server Event Handler

main.c 9+

Core > Src > main.c > server_fn

```
119  //*****************************************************************************
120  //* SERVER EVENT HANDLER - RED LED CONTROL
121  //*****************************************************************************
122      if(strncmp(c->recv.buf,"RED LED ON",11) == 0)
123      {
124      HAL_GPIO_WritePin(GPIOx: LED3R_GPIO_Port, GPIO_Pin: LED3R_Pin, PinState: GPIO_PIN_SET);
125      MG_INFO((fmt: "INFO: RED LED IS ON!"));
126      }
127      else if(strncmp(c->recv.buf,"RED LED OFF",12) == 0)
128      {
129      HAL_GPIO_WritePin(GPIOx: LED3R_GPIO_Port, GPIO_Pin: LED3R_Pin, PinState: GPIO_PIN_RESET);
130      MG_INFO((fmt: "INFO: RED LED IS OFF!"));
131      }
132  //*****************************************************************************
133  //* SERVER EVENT HANDLER - GREEN LED CONTROL
134  //*****************************************************************************
135    if(strncmp(c->recv.buf,"GREEN LED ON",13) == 0)
136      {
137      HAL_GPIO_WritePin(GPIOx: LED1G_GPIO_Port, GPIO_Pin: LED1G_Pin, PinState: GPIO_PIN_SET);
138      MG_INFO((fmt: "INFO: GREEN LED IS ON!"));
139      }
140      else if(strncmp(c->recv.buf,"GREEN LED OFF",14) == 0)
141      {
142      HAL_GPIO_WritePin(GPIOx: LED1G_GPIO_Port, GPIO_Pin: LED1G_Pin, PinState: GPIO_PIN_RESET);
143      MG_INFO((fmt: "INFO: GREEN LED IS OFF!"));
144      }
145  //*****************************************************************************
146  //* SERVER EVENT HANDLER - BLUE LED CONTROL
147  //*****************************************************************************
148      if(strncmp(c->recv.buf,"BLUE LED ON",12) == 0)
149      {
150      HAL_GPIO_WritePin(GPIOx: LED2B_GPIO_Port, GPIO_Pin: LED2B_Pin, PinState: GPIO_PIN_SET);
151      MG_INFO((fmt: "INFO: BLUE LED IS ON!"));
152      }
153      else if(strncmp(c->recv.buf,"BLUE LED OFF",13) == 0)
154      {
155      HAL_GPIO_WritePin(GPIOx: LED2B_GPIO_Port, GPIO_Pin: LED2B_Pin, PinState: GPIO_PIN_RESET);
156      MG_INFO((fmt: "INFO: BLUE LED IS OFF!"));
157      }
```

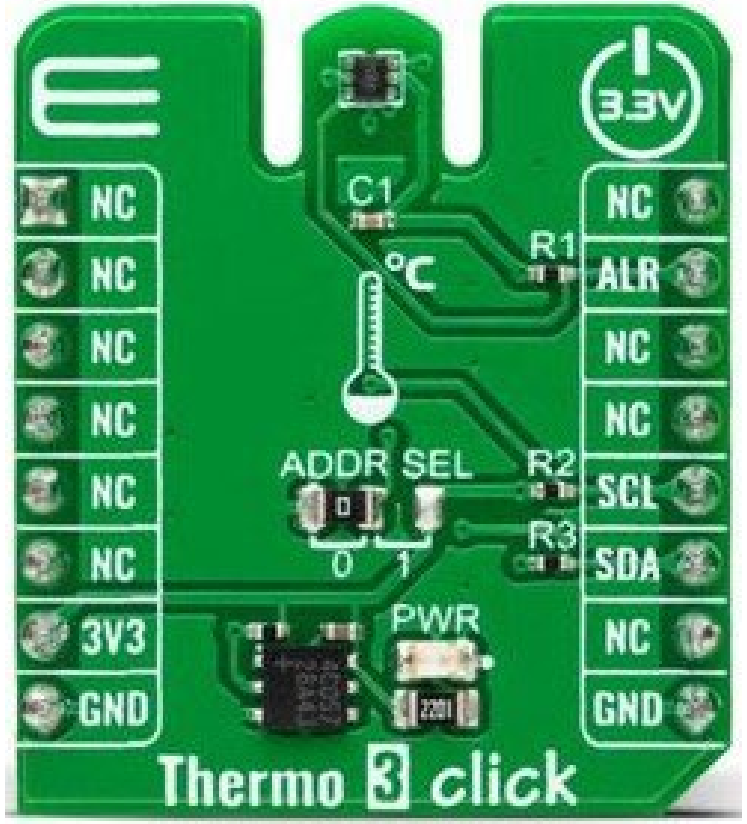


Assemble main.c – Build the Server Event Handler

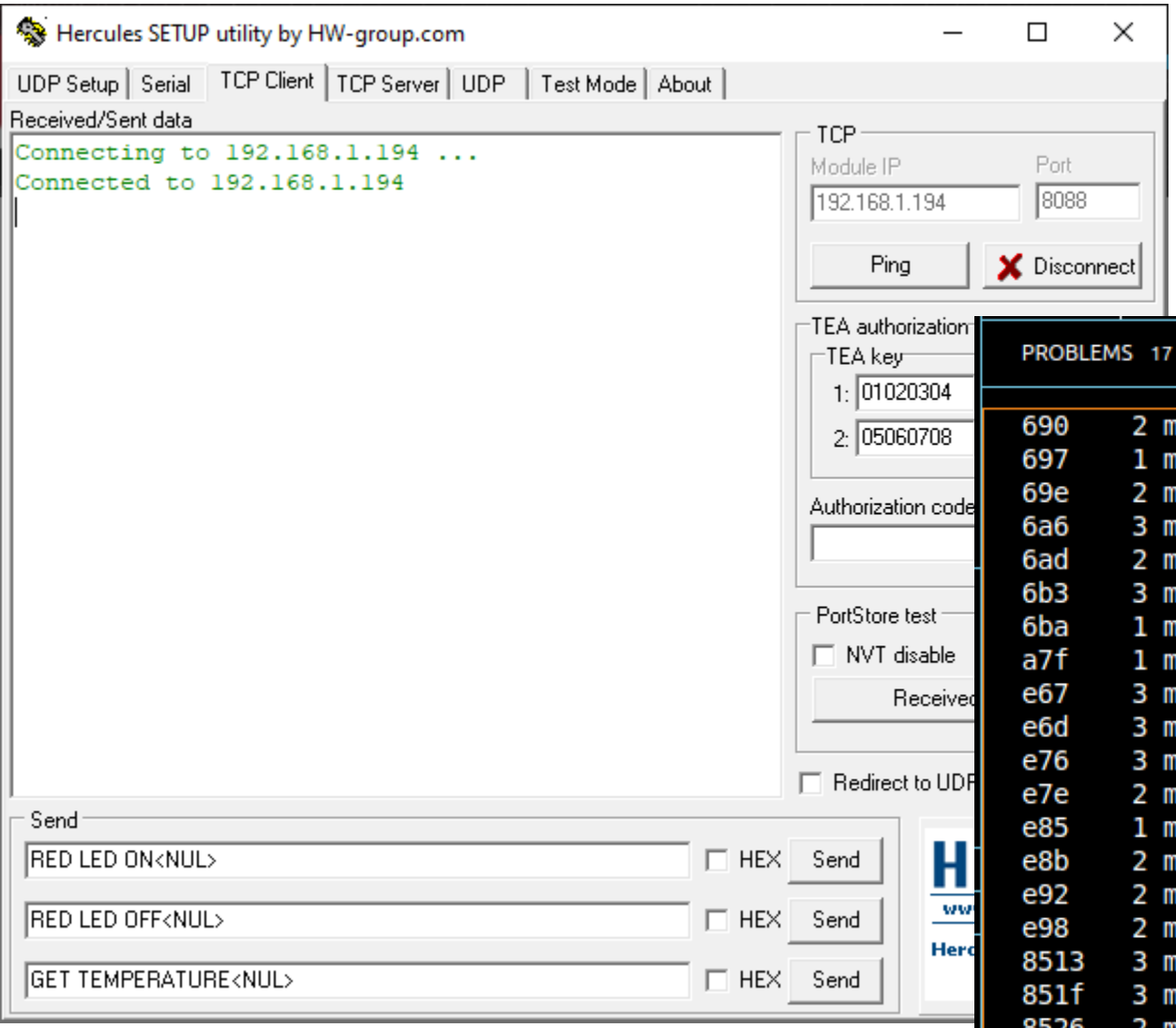
main.c 9+

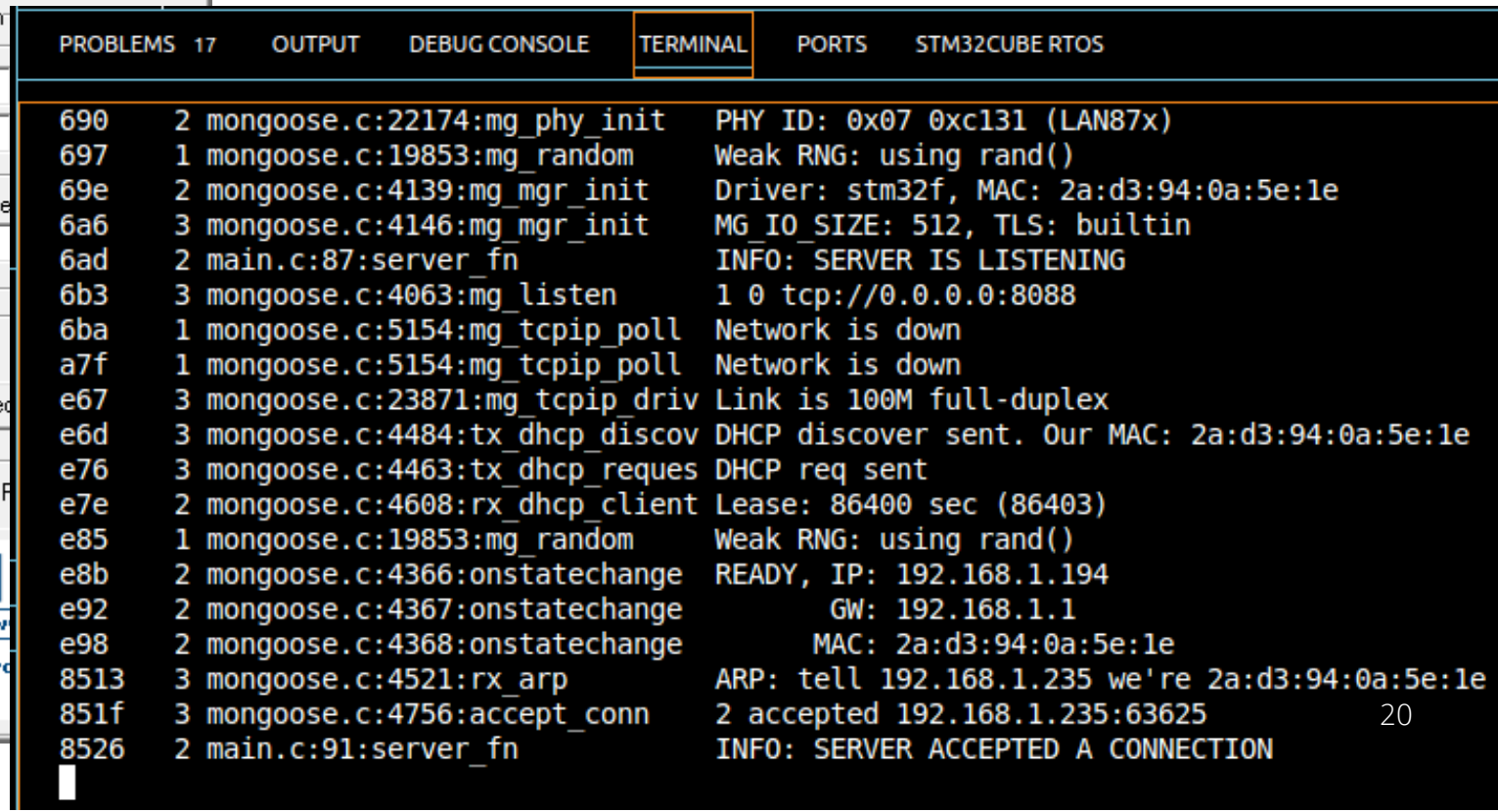
Core > Src > main.c > server_fn

```
158  //****************************************************************************
159  //* SERVER EVENT HANDLER - GET TEMPERATURE IN CELCIUS
160  //****************************************************************************
161      else if(strncmp(c->recv.buf,"GET TEMPERATURE",16) == 0)
162      {
163        HAL_GPIO_TogglePin(GPIOx: LED2B_GPIO_Port, GPIO_Pin: LED2B_Pin);
164        temptosend = tmp102_read_temperature();
165        sprintf(c->recv.buf, "%.2fC", temptosend);
166        c->recv.len = strlen(c->recv.buf);
167        MG_INFO((fmt: "INFO: Current Temperature -> %.2f deg C\n\n",temptosend));
168      }
169      mg_send(c, c->recv.buf, c->recv.len);
170      c->recv.len = 0;                          // Tell Mongoose we've consumed data
171    }
172    else if (ev == MG_EV_CLOSE)
173    {
174      MG_INFO((fmt: "SERVER DISCONNECTED!"));
175    }
176    else if (ev == MG_EV_ERROR)
177    {
178      MG_INFO((fmt: "SERVER ERROR: %s", (char *) ev_data));
179    }
180  }
```

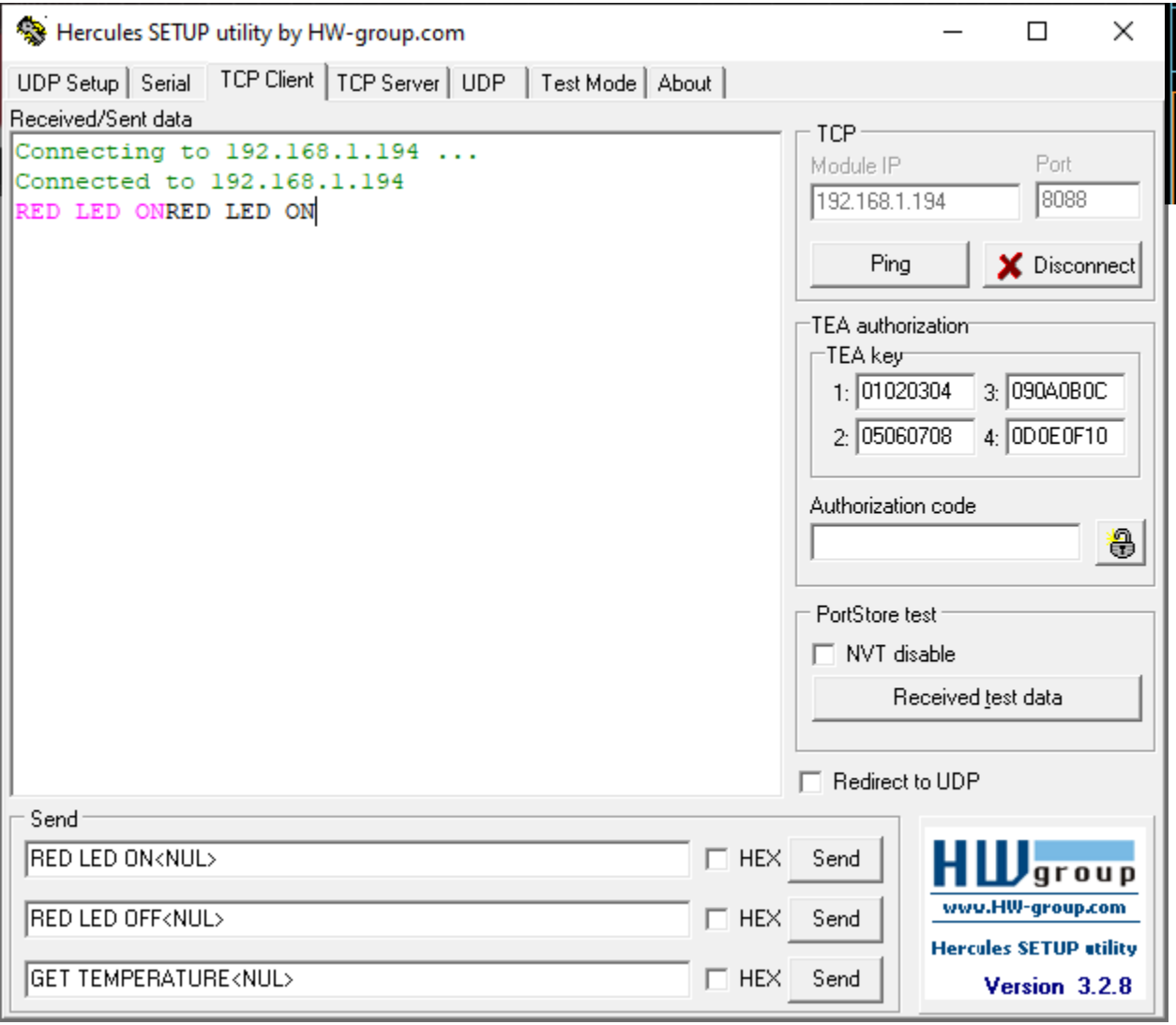


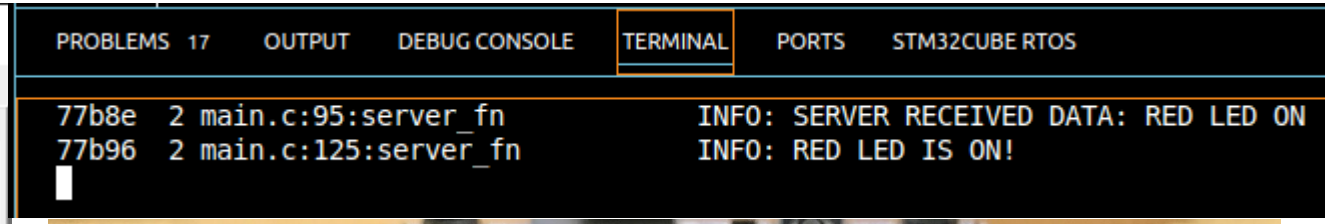
Let's Put MAX to Work – Connect to the Server

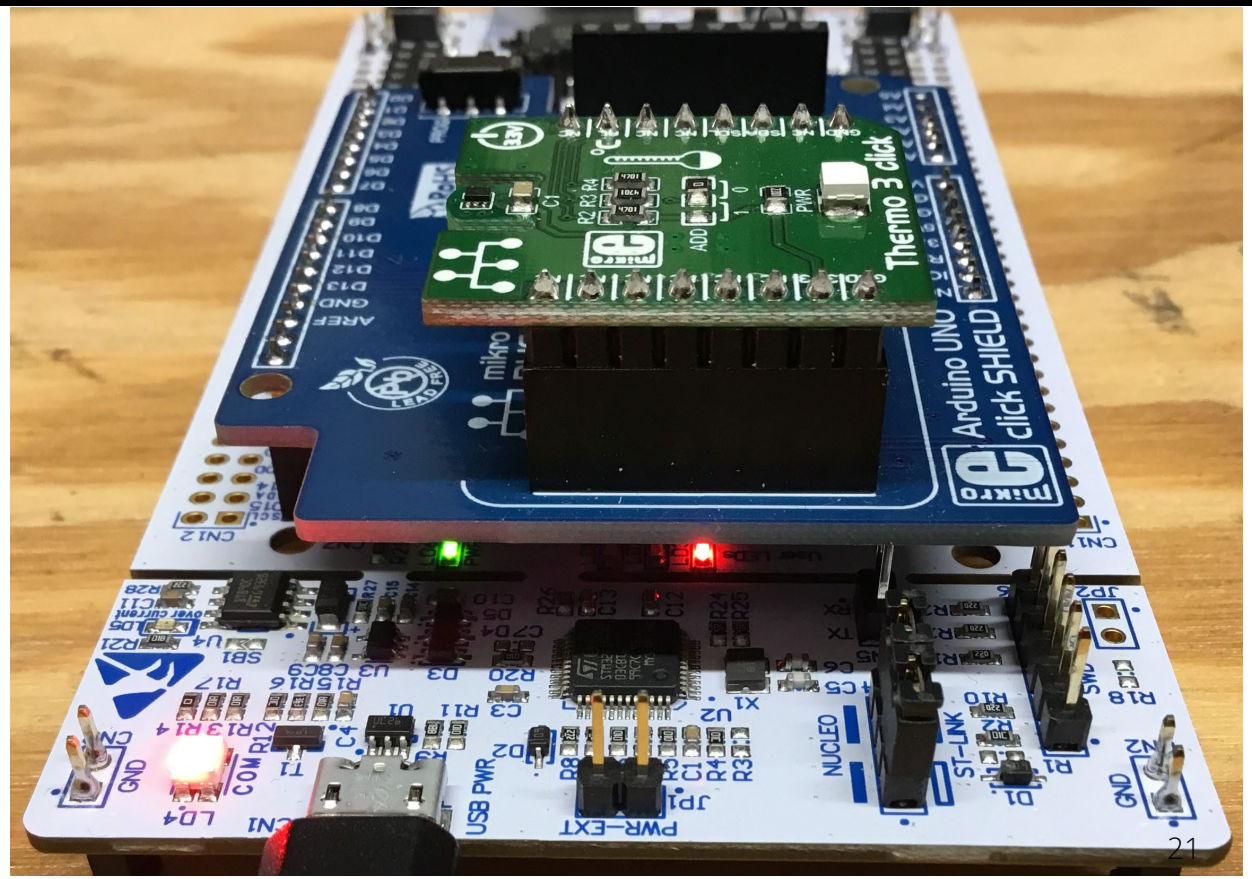




Let's Put MAX to Work – Control the RED LED







Let's Put MAX to Work – Get the Temperature

Hercules SETUP utility by HW-group.com

UDP Setup | Serial | TCP Client | TCP Server | UDP | Test Mode | About

Received/Sent data

```
Connecting to 192.168.1.194 ...
Connected to 192.168.1.194
RED LED ONRED LED ONGET TEMPERATURE26.31C
```

TCP
Module IP: 192.168.1
Ping | Disconnect

TEA authorization
TEA key
1: 01020304 3: 090A0B0C
2: 05060708 4: 0D0E0F10

Authorization code

PortStore test
NVT disable
Received test data

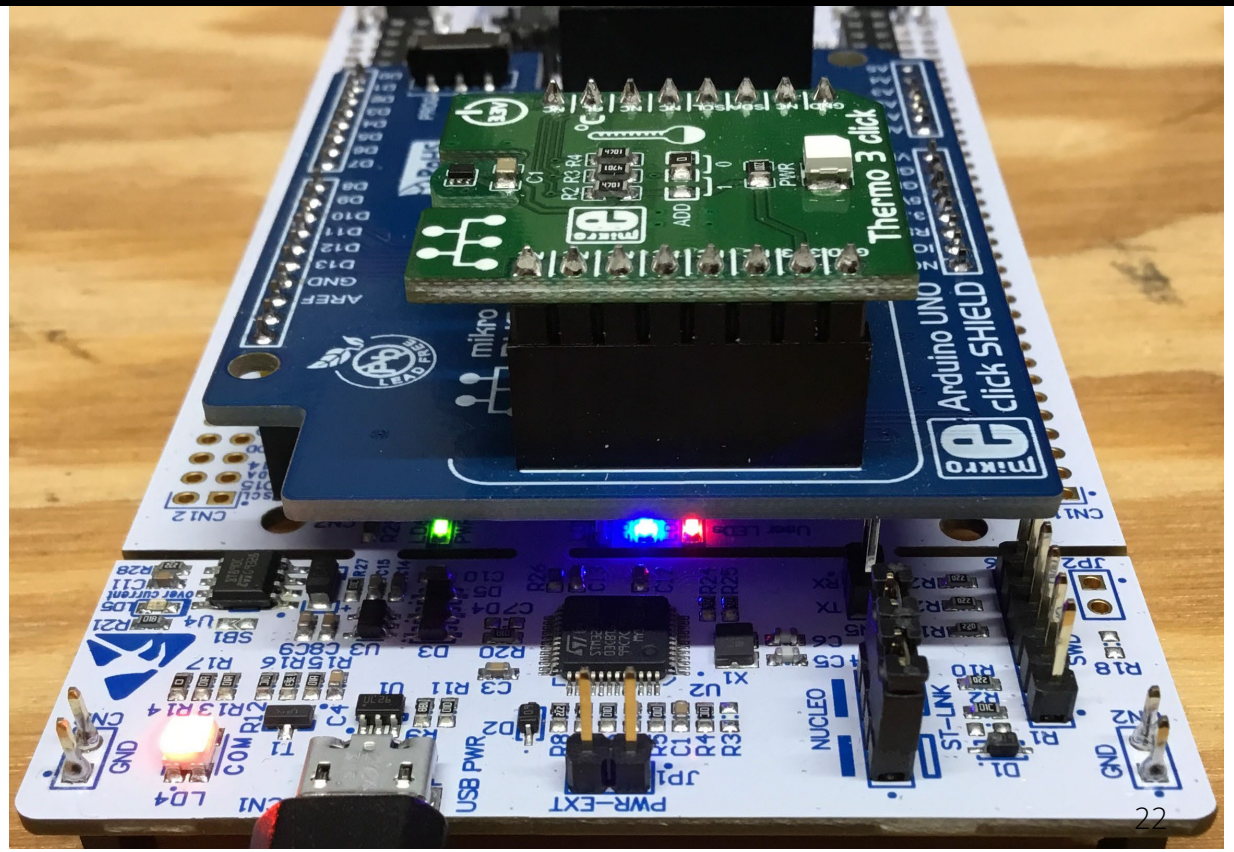
Redirect to UDP

Send
RED LED ON<NUL> HEX Send
RED LED OFF<NUL> HEX Send
GET TEMPERATURE<NUL> HEX Send

HW group
www.HW-group.com
Hercules SETUP utility
Version 3.2.8

PROBLEMS 17 | OUTPUT | DEBUG CONSOLE | TERMINAL | PORTS | STM32CUBE RTOS

```
bc466  2 main.c:95:server_fn            INFO: SERVER RECEIVED DATA: GET TEMPERATURE
bc46e  2 tmp102.c:82:tmp102_read_temper INFO: 2 bytes were read-> rxBuf[0] 0x1a -- rxBuf[1] 0x50
bc478  2 tmp102.c:87:tmp102_read_temper INFO: 16-bit value = 0x01a5
bc47f  2 main.c:167:server_fn           INFO: Current Temperature -> 26.31 deg C
```

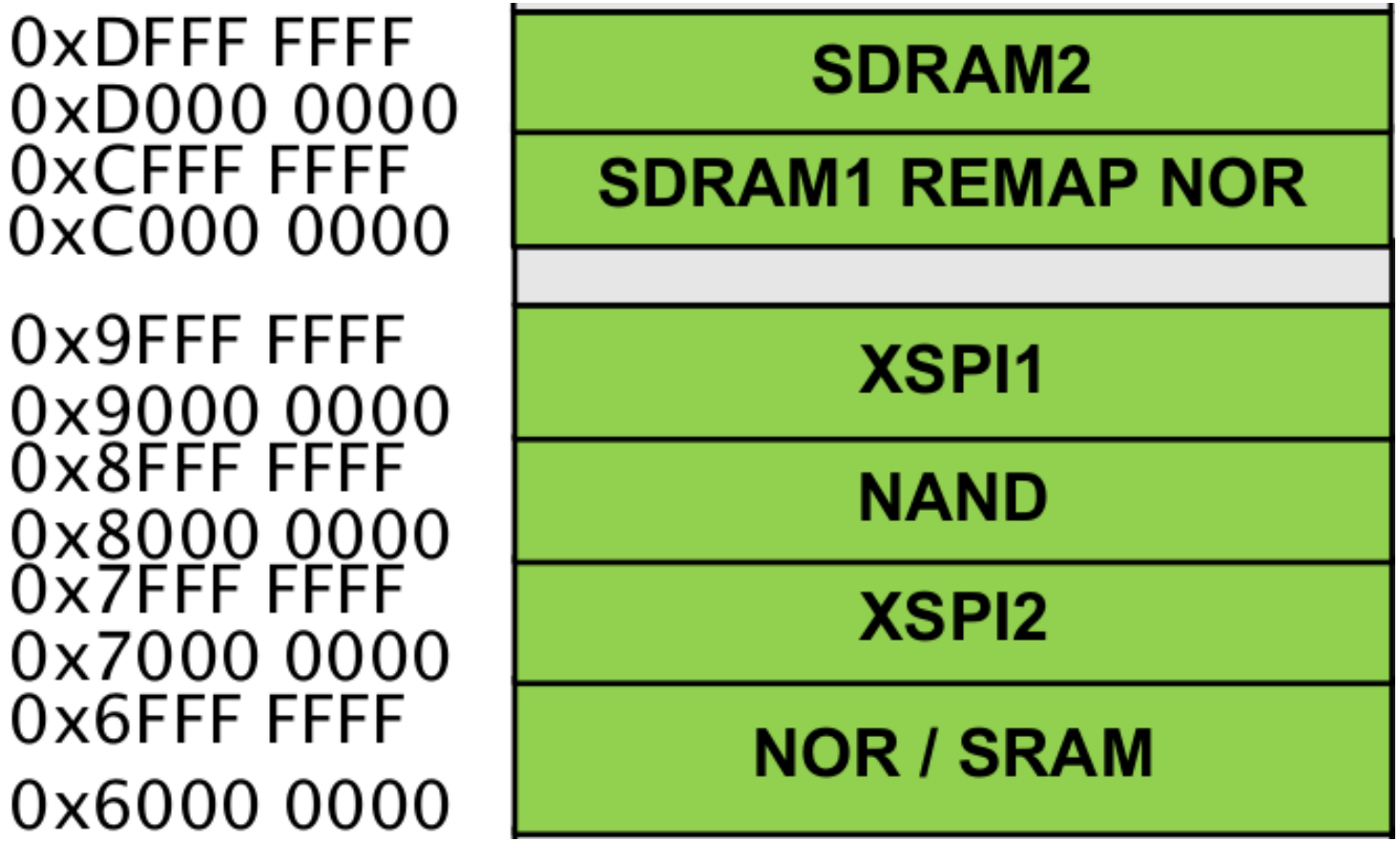


Thank you for attending!!!

Please consider the resources below:

- Today's Download Package
- www.st.com
- digikey.com

MORE TO COME..



CEC Continuing Education Center

DesignNews

Thank You

Sponsored by

DigiKey

informa markets