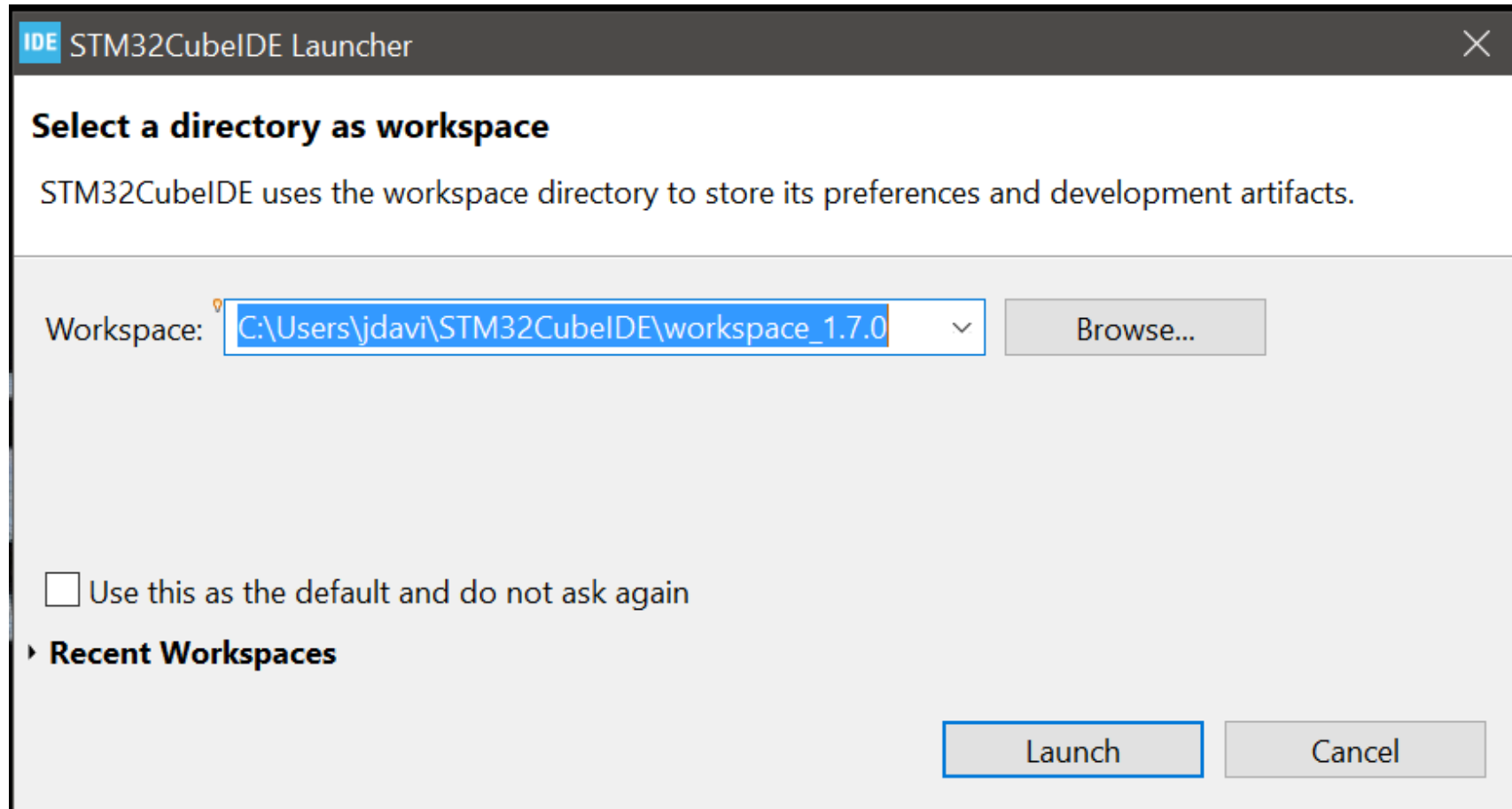# UCSD Embedded C Final Assignment
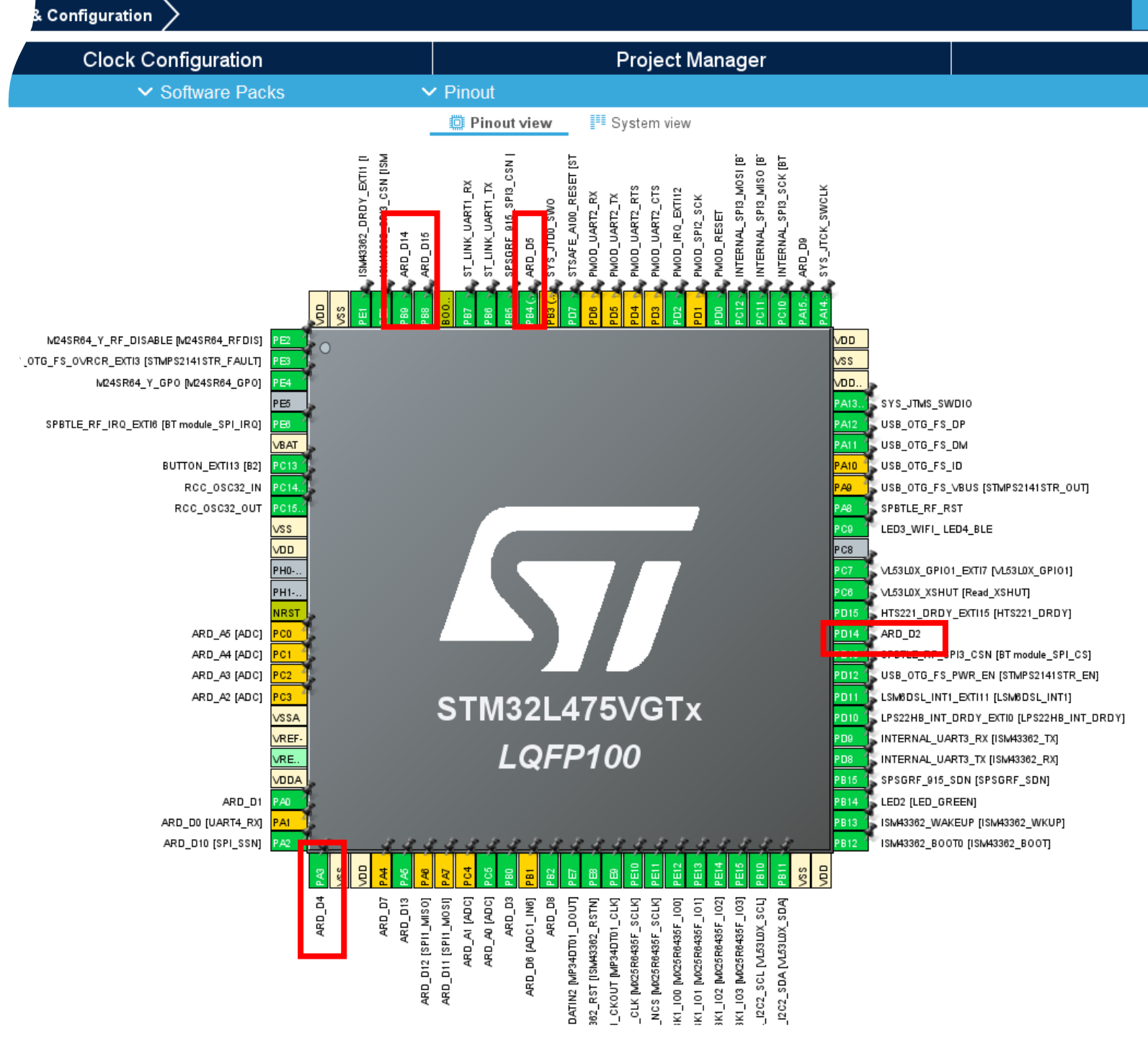
Jonathan Francisco

jdavidfrancisco@gmail.com

Startup STM32CubeMX

Observe Pinout View

- Here we enable the Arduino connections for inputs / outputs.
- I have connected 7 external LEDs
  - 1 Energy saver Blue LED
  - 1 Strong brew Yellow LED
  - 4 Cup size Green LED
  - 1 Low water level Red LED
- And 3 external buttons to control the energy saver, strong brew, and cup size settings.

Observe Project Manager

- The Project name and Linker settings are here.

- Also, the code will be generated and exported into the STM32CubeIDE.

Verify Project Components

- IDE Coffee-Maker
  - > Binaries
  - > Includes
  - ✓ Core
    - > Inc
    - ✓ Src
      - > main.c
      - > stm32l4xx_hal_msp.c
      - > stm32l4xx_it.c
      - > syscalls.c
      - > sysmem.c
      - > system_stm32l4xx.c
    - > Startup
  - ✓ Drivers
    - ✓ BSP
      - > B-L475E-IOT01
      - > Components
    - > CMSIS
    - > STM32L4xx_HAL_Driver
  - > Debug
  - MX Coffee-Maker.ioc
  - Coffee-Maker.launch
  - STM32L475VGTX_FLASH.ld
  - STM32L475VGTX_RAM.ld
- IDE Embedded-C-Assignment-2
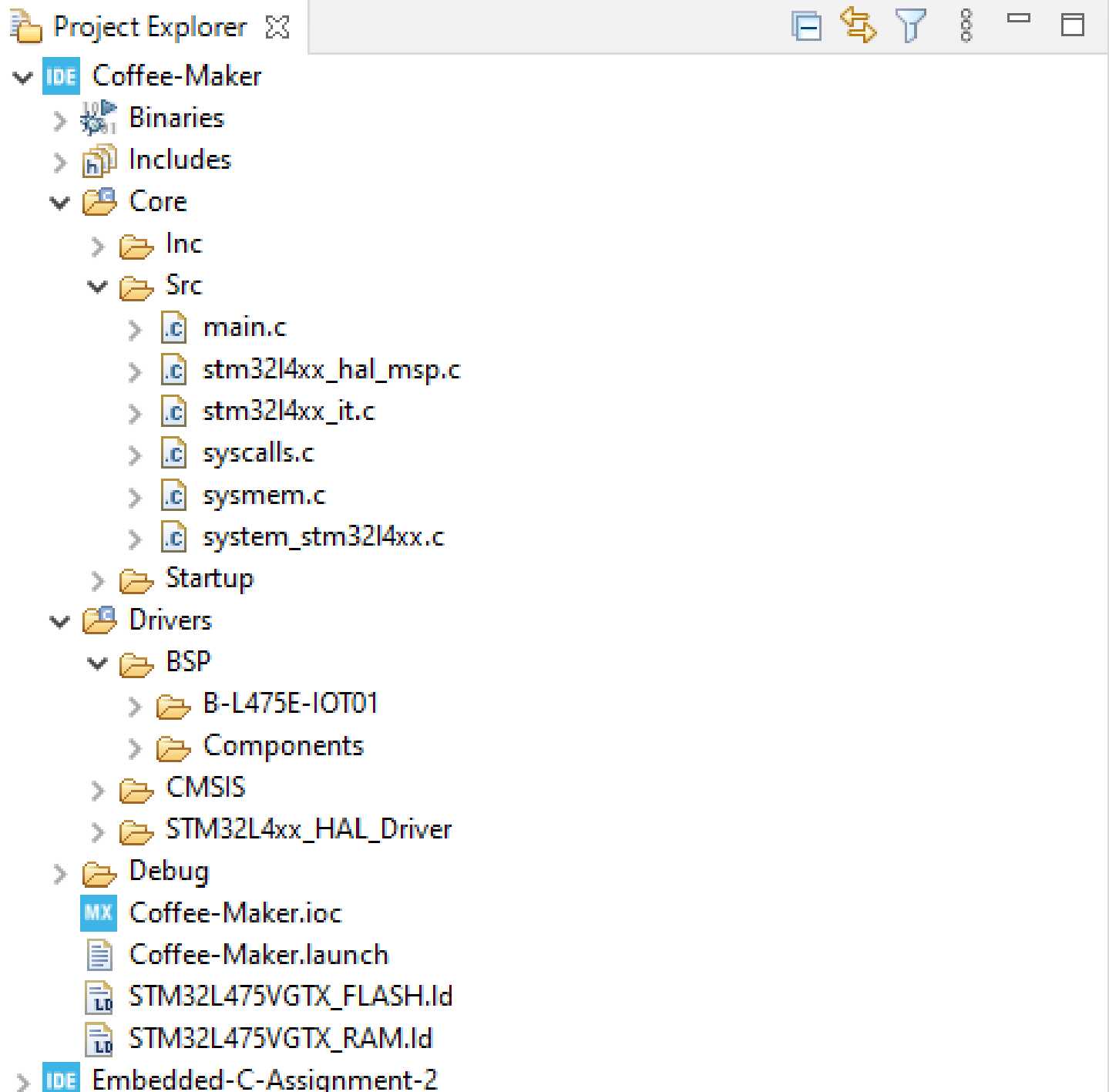
Private header included for printf and temperature sensor in main.c

```
23⊖ /* Private includes ------------------------------------------------------------*/
24   /* USER CODE BEGIN Includes */
25   #include <stdio.h>
26   #include "stm32l475e_iot01_tsensor.h"
27
28   /* USER CODE END Includes */
```

Private variables to hold different values in main.c

```
39  /* Private macro -------------------------------------------------------------*/
40  /* USER CODE BEGIN PM */
41
42  int energy_save;
43  int strong_brew;
44  int water_level;
45  int cup_size;
46  int cup_choice;
47  float water_temp;
48
49  /* USER CODE END PM */
```

Function to print to console using printf through UART1 in main.c

```
78⊝ /* Private user code ------------------------------------------*/
79  /* USER CODE BEGIN 0 */
80
81⊝ int __io_putchar(int ch)
82  {
83      HAL_UART_Transmit(&huart1, (uint8_t *)&ch, 1, 1000);
84      return 1;
85  }
86
87  /* USER CODE END 0 */
```

Code in main.c Part 1

-Initialize various settings to start a brew on power on

-The temperature sensor is taken through the BSP read temp function and shown here

```c
119
120  /* Configure the peripherals common clocks */
121    PeriphCommonClock_Config();
122
123    /* USER CODE BEGIN SysInit */
124
125    /* USER CODE END SysInit */
126
127    /* Initialize all configured peripherals */
128    MX_GPIO_Init();
129    MX_DFSDM1_Init();
130    MX_I2C2_Init();
131    MX_QUADSPI_Init();
132    MX_SPI3_Init();
133    MX_USART1_UART_Init();
134    MX_USART3_UART_Init();
135    MX_USB_OTG_FS_PCD_Init();
136    MX_ADC1_Init();
137    /* USER CODE BEGIN 2 */
138
139    // Automatically turn on energy saver mode
140    energy_save = 1;
141    HAL_GPIO_WritePin(GPIOB, ARD_D8_Pin, GPIO_PIN_SET);
142
143    //Set strong Brew variable and turn on LED
144    strong_brew = 1;
145    HAL_GPIO_WritePin(GPIOB, ARD_D5_Pin, GPIO_PIN_SET);
146
147    //Cup size selector
148    cup_size = 1;
149    cup_choice = 12;
150    //Start with 12oz cup selected
151    HAL_GPIO_WritePin(GPIOA, ARD_D1_Pin, GPIO_PIN_SET);
152
153    //Water temperature sensor
154    BSP_TSENSOR_Init();
155    water_temp = BSP_TSENSOR_ReadTemp();
156
157    // Water level variable (Max water == 100 | Min Water = 0;
158    water_level = 0;
159
160    //Time tracker in seconds
161    int time_sec = 0;
162
163    //Turn On Power LED2 at Start
164    HAL_GPIO_WritePin(LED2_GPIO_Port, LED2_Pin, GPIO_PIN_SET);
165
166    printf("Power On\r\n\n");
167
168    printf("---Coffee-Maker V1.0---\r\n");
169    printf("Brewing Start\r\n");
170
171    /* USER CODE END 2 */
```

Code in main.c Part 2

- Continue sequence in while loop, until power off is given by the energy_save variable or user turns power off.
- Once power off is initiated, the various settings are printed to the console

- There are two analog inputs:
    - Water level given by an ADC Poll
    - Water Temp given by BSP temperature sensor

```c
{
    /* USER CODE BEGIN 3 */

    //Check water level
    //If water level is below 25% capacity, turn on Add Water LED
    HAL_ADC_Start(&hadc1);
    HAL_ADC_PollForConversion(&hadc1, 100);
    water_level = HAL_ADC_GetValue(&hadc1) / 100;
    HAL_ADC_Stop(&hadc1);
    if (water_level == 0 || water_level > 25)
    {
        // 0 level will mock that the water level is full
        if (water_level == 0)
        {
            water_level = 100;
        }
        HAL_GPIO_WritePin(GPIOA, ARD_D9_Pin, GPIO_PIN_RESET);
    }
    else if (water_level <= 25)
    {
        HAL_GPIO_WritePin(GPIOA, ARD_D9_Pin, GPIO_PIN_SET);
    }

    //if LED2 on after 25 seconds
    if((time_sec != 0) && (time_sec % 25 == 0) && energy_save && HAL_GPIO_ReadPin(LED2_GPIO_Port, LED2_Pin))
    {
        //Turn off Power LED
        HAL_GPIO_WritePin(LED2_GPIO_Port, LED2_Pin, GPIO_PIN_RESET);

        printf("Brew Status: Finished\r\n");
        if(strong_brew)
        {
            printf("Strong Brew: Enabled\r\n");
        }
        else
        {
            printf("Strong Brew: Disabled\r\n");
        }
        printf("Cup size: %doz\r\n\n", cup_choice);

        printf("Water Temp: %f C\r\n", water_temp);
        printf("Water Level: %d\r\n", water_level);

        printf("Brewing Stop\r\n");
        printf("---Have a great day!---\r\n");
        printf("---Fran Clan Roasts---\r\n\n");
        printf("Power off\r\n\n\n");
    }

    //Increment timer
    time_sec++;
    HAL_Delay(1000);

    /* USER CODE END 3 */
```

Code in main.c Part 3

- The HAL interrupt is shown here. There are three external GPIO buttons added to toggle the energy saver mode, the strong brew setting, and to toggle between coffee sizes.

- There are LEDs that correspond to the settings and are toggled by the external buttons.

- Here the first two settings are shown.


```c
void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
{
    // Prevent unused argument(s) compilation warning
    UNUSED(GPIO_Pin);

    // Toggle Auto Turn off
    if(GPIO_Pin == ARD_D13_Pin)
    {
        energy_save = !energy_save;
        HAL_GPIO_TogglePin(GPIOB, ARD_D8_Pin);
    }
    //Toggle Strong Brew
    else if(GPIO_Pin == ARD_D14_Pin)
    {
        strong_brew = !strong_brew;
        HAL_GPIO_TogglePin(GPIOB, ARD_D5_Pin);
    }
```

Code in main.c Part 4

- Shown here is the toggle between the various cup sizes using an external button interrupt.


```c
//Toggle through cup sizes
else if(GPIO_Pin == ARD_D15_Pin)
{
    //Toggle between cup sizes
    cup_size++;

    if (cup_size == 1)
    {
        cup_choice = 12;

        HAL_GPIO_WritePin(GPIOA, ARD_D4_Pin, GPIO_PIN_RESET);
        HAL_GPIO_WritePin(GPIOA, ARD_D1_Pin, GPIO_PIN_SET);
    }
    else if (cup_size == 2)
    {
        cup_choice = 10;

        HAL_GPIO_WritePin(GPIOA, ARD_D1_Pin, GPIO_PIN_RESET);
        HAL_GPIO_WritePin(GPIOD, ARD_D2_Pin, GPIO_PIN_SET);
    }
    else if (cup_size == 3)
    {
        cup_choice = 8;

        HAL_GPIO_WritePin(GPIOD, ARD_D2_Pin, GPIO_PIN_RESET);
        HAL_GPIO_WritePin(GPIOB, ARD_D3_Pin, GPIO_PIN_SET);
    }
    else if (cup_size == 4)
    {
        cup_choice = 6;

        HAL_GPIO_WritePin(GPIOB, ARD_D3_Pin, GPIO_PIN_RESET);
        HAL_GPIO_WritePin(GPIOA, ARD_D4_Pin, GPIO_PIN_SET);

        cup_size = 0;
    }
}
```

Code in main.c Part 5

- Lastly, the Power On/Off interrupt is shown here. If the user stops the brewing sequence, the brew is finished, and the status is output to the console.

- Power On/Off is controlled by the blue button inherent to the STM32 board.

```
293         //Power on/off
294         else
295         {
296             if (HAL_GPIO_ReadPin(LED2_GPIO_Port, LED2_Pin))
297             {
298                 printf("Brew Status: Paused\r\n");
299                 if(strong_brew)
300                 {
301                     printf("Strong Brew: Enabled\r\n");
302                 }
303                 else
304                 {
305                     printf("Strong Brew: Disabled\r\n");
306                 }
307                 printf("Cup size: %doz\r\n\n", cup_choice);
308
309                 printf("Water Temp: %f C\r\n", water_temp);
310                 printf("Water Level: %d %%\r\n", water_level);
311
312                 printf("Brewing Stop\r\n");
313                 printf("---Have a great day!---\r\n");
314                 printf("---Fran Clan Roasts---\r\n\n");
315                 printf("Power off\r\n\n");
316             }
317             else
318             {
319                 printf("Power on\r\n\n");
320                 printf("---Coffee-Maker V1.0---\r\n");
321                 printf("Brewing Start\r\n");
322             }
323
324             HAL_GPIO_TogglePin(LED2_GPIO_Port, LED2_Pin);
325         }
326 }
327
328  /**
```

Build Results

Debug Results

At start of Debug, a breakpoint is hit in main.c
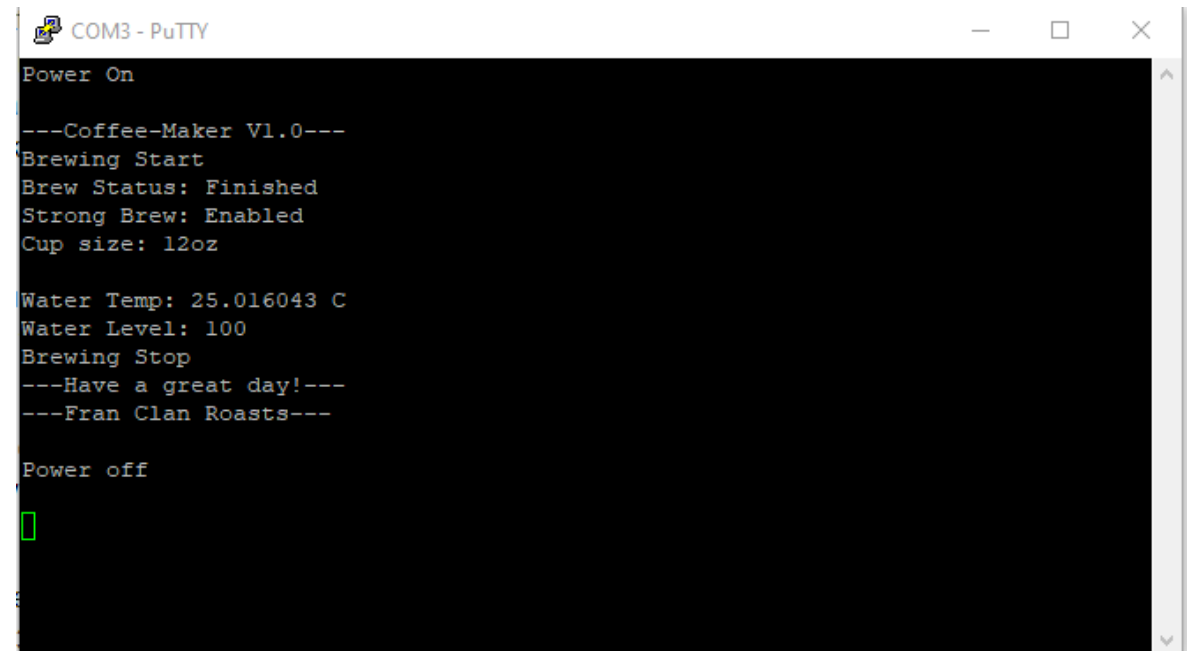
Open Putty
Terminal

Output Results

- Shown is the full brew sequence at startup.

- The brew status shows finished, as the energy saver option is turned on and turned the power off without user input

- The Strong brew setting is enabled.

- The cup size was set to 12 oz

- The temperature was taken from the board's temperature sensor.

- Lastly, the water level was taken through the ADC Polling. In this case the ADC is connected to the ground from A0 setting the charge to 0, which in our case mocks that the water level is full.


```
COM3 - PuTTY                          —   □   ×
Power On


---Coffee-Maker V1.0---
Brewing Start
Brew Status: Finished
Strong Brew: Enabled
Cup size: 12oz

Water Temp: 25.016043 C
Water Level: 100
Brewing Stop
---Have a great day!---
---Fran Clan Roasts---

Power off

█
```
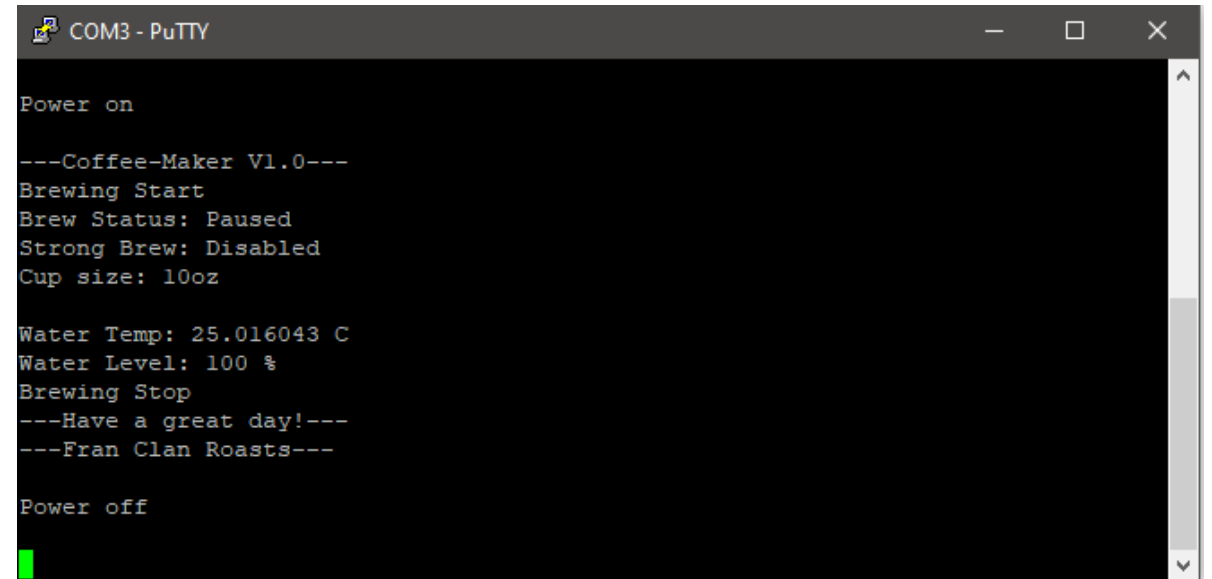
Board Results

- Shown is the full brew sequence at startup.

- The brew status shows finished, as the energy saver option is turned on and turned the power off without user input

- The Strong brew setting is enabled.

- The cup size was set to 12 oz

- The temperature was taken from the board's temperature sensor.

- Lastly, the water level was taken through the ADC Polling. In this case the ADC is connected to the ground from A0 setting the charge to 0, which in our case mocks that the water level is full.
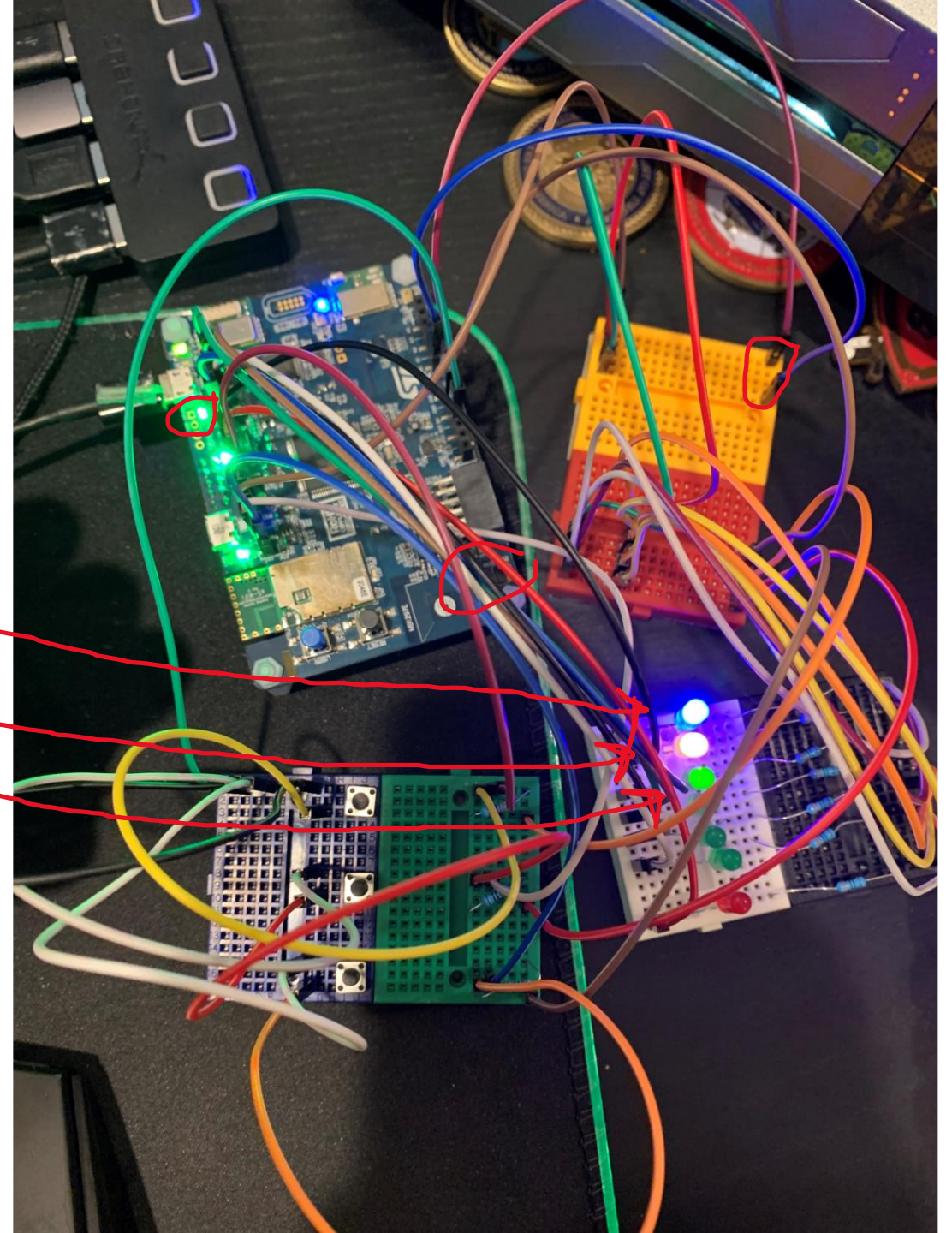
Output Results 2

- Shown is the full brew sequence that the user stopped.

- The settings shown have changed to show that the brew status paused from user input, and the strong brew was disabled.

- The cup size also changed from 12oz to 10oz
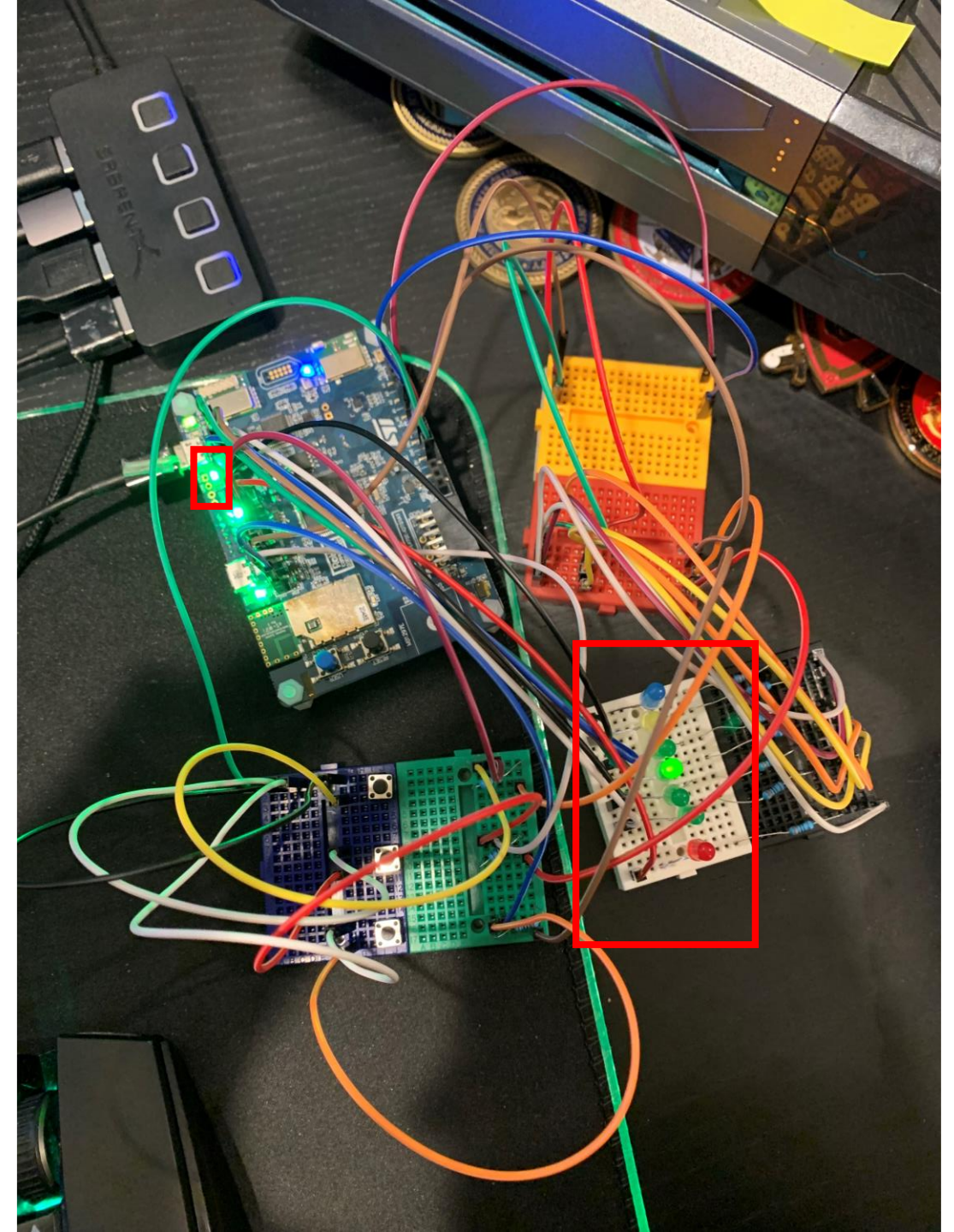


```
COM3 - PuTTY

Power on

---Coffee-Maker V1.0---
Brewing Start
Brew Status: Paused
Strong Brew: Disabled
Cup size: 10oz

Water Temp: 25.016043 C
Water Level: 100 %
Brewing Stop
---Have a great day!---
---Fran Clan Roasts---

Power off
```
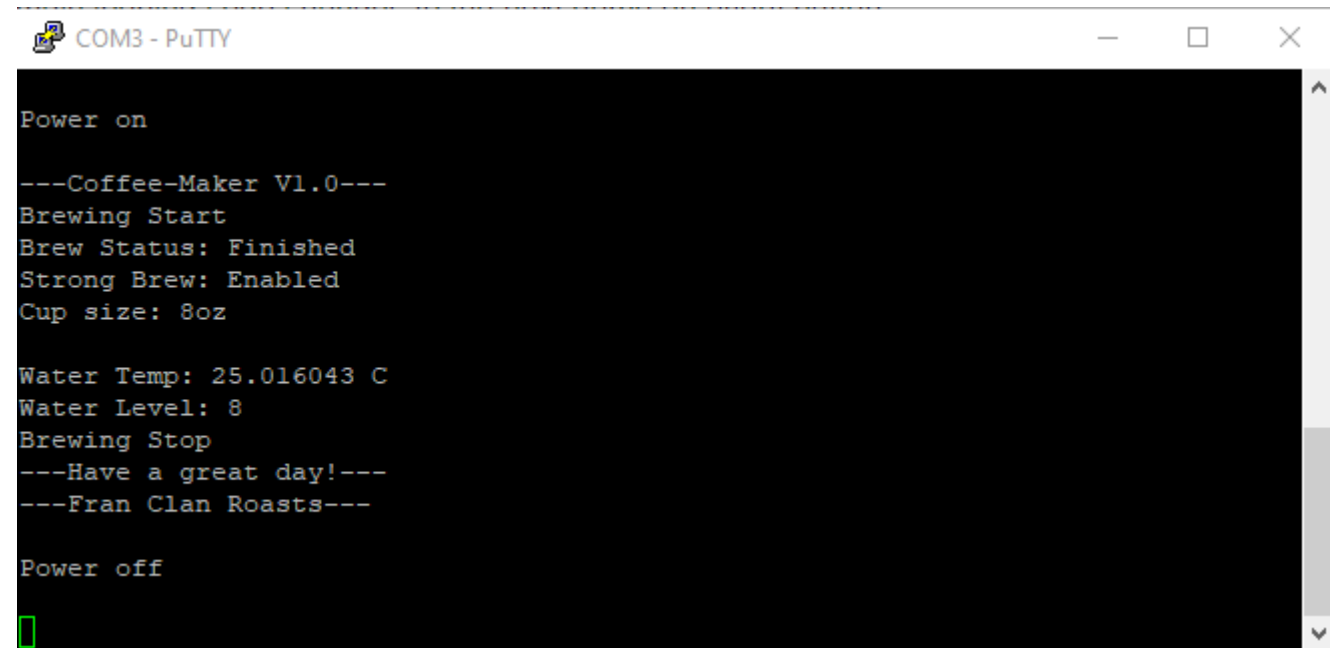
Board Results 2

- Shown is the full brew sequence that the user stopped.

- The settings shown have changed to show that the brew status paused from user input, and the strong brew was disabled.

- The cup size also changed from 12oz to 10oz

Output Results 3

- For the final test, the brew sequence ran through and turned off from the power saver mode.

- Strong brew was enabled, and the cup size changed to 8oz

- The water level did change, since the ADC is no longer connected to ground. On the board you will see the water low light on.


```
COM3 - PuTTY                              —    □    ×

Power on

---Coffee-Maker V1.0---
Brewing Start
Brew Status: Finished
Strong Brew: Enabled
Cup size: 8oz

Water Temp: 25.016043 C
Water Level: 8
Brewing Stop
---Have a great day!---
---Fran Clan Roasts---

Power off
```

Board Results 3

- For the final test the brew sequence ran through and turned off from the power saver mode.

- Strong brew was enabled, and the cup size changed to 8oz

- The water level did change, since the ADC is no longer connected to ground. On the board you will see the water low light on.