Basics: Functions

Functions allow us to simplify the logical flow of a program and reduce redundancy in code.

Suggested Prerequisites

- Programming Intro
- Conditional Statements

Setup and Preparation

List of Materials

- TI MSP430G2 LaunchPad
- USB cable

Program 1: Blink by function

Create the following program (Figure 1), and upload it to the LaunchPad.

```c
void setup() {
    pinMode(RED_LED, OUTPUT);
}

void loop() {
    blinkLED();
}

void blinkLED() {
    digitalWrite(RED_LED, HIGH);
    delay(1000);
    digitalWrite(RED_LED, LOW);
    delay(1000);
}
```

Figure 1: Blink by function

Things to consider

- You can try to explain what is happening in the program using a block diagram.
  - Consider `blinkLED()` as a big block that contains another block diagram within it
Program 2: Blink by function with argument

Create the following program (Figure 2), and upload it to the LaunchPad.

```cpp
void setup() {
  pinMode(RED_LED, OUTPUT);
  pinMode(GREEN_LED, OUTPUT);
}

void loop() {
  blinkLED(RED_LED);
  blinkLED(GREEN_LED);
}

void blinkLED(int pinNumber) {
  digitalWrite(pinNumber, HIGH);
  delay(1000);
  digitalWrite(pinNumber, LOW);
  delay(1000);
}
```

Figure 2: Blink by function with arguments

Things to consider

- You can try to explain what is happening in this program using a block diagram.
- You can observe the value of pinNumber within the blinkLED() function using the Serial Monitor.
  - You will need to modify the program to enable this.
Program 3: Returning values from functions

Create and upload the following program (Figure 3) to the LaunchPad.

```cpp
void setup() {
  pinMode(RED_LED, OUTPUT);
  pinMode(PUSH2, INPUT_PULLUP);
}

void loop() {
  boolean buttonIsPressed = getButtonState();
  if (buttonIsPressed) {
    digitalWrite(RED_LED, HIGH);
  }
  else {
    digitalWrite(RED_LED, LOW);
  }
}

boolean getButtonState() {
  boolean buttonState = !digitalRead(PUSH2);
  return buttonState;
}
```

Figure 3: Returning values from functions

Things to consider

- You can try to explain what is happening in this program using a block diagram.
- Every function declaration we have seen so far used a `void` return type. Consider why `getButtonState()` does not have the return type `void`. 
Create and upload the following program (Figure 4) to the LaunchPad.

```c
void setup() {
    Serial.begin(9600);

    int n1 = 12;
    int n2 = 7;
    int result = add(n1, n2);
    Serial.println();
    Serial.println(result);
}

void loop() {
}

int add(int n1, int n2) {
    int result = n1 + n2;
    return result;
}
```

**Figure 4: Mathematical functions**

**Things to consider**

- You can use the *Serial Monitor* to observe what is happening.
- You can press the RESET button to replay the program.
- You can modify the numbers n1 and n2 to better characterize the system.
- You can create other mathematical functions in a similar way, such as `multiply()`.
Review

After this lab, you should have a good understanding of the following topics. If you’re not sure about some of them, go back through the lab and try to find a good place to explore the topic.

You should be able to do the following

- Create and use a function that can take parameters and/or return data.
- Create and use a function to perform some computation or evaluation, and return the result.

Application

Try applying the skills you learned from this lab. You can come up with your own project idea or try one from the flowchart.