```
 1 /*
 2  * CMPE 146: I2C Lab Main_Slave
 3  */
 4
 5 /**
 6  * @file
 7  * @brief This is the application entry point.
 8  */
 9
10 #include <stdio.h>
11 #include "utilities.h"
12 #include "io.hpp"
13 #include <i2c2.hpp>
14 #include <tasks.hpp>
15 #include <GPIO/GPIOInterrupt.hpp>
16 #include <printf_lib.h>
17 #include <uart0_min.h>
18
19 volatile uint8_t buffer[256] = { 0 };
20
21 typedef enum {
22     addition,
23     subtraction,
24     multiplication
25 } operation;
26
27 uint8_t operand_1 = 0, operand_2 = 0;
28
29 uint8_t result = 0;
30
31
32
33
34 void vReadBuffer(void *pvParameters){
35     while(1){
36         for (uint8_t i = 0; i < 10; i++){
37             printf("Buffer %u: %X\n", i, buffer[i]);
38         }
39         puts("\n");
40         vTaskDelay(1000);
41     }
42 }
43
44
45 void vCalculate(void *pvParameters){
46     while (1){
47         //...do stuff
```

```
48          operand_1 = buffer[1];
49          operand_2 = buffer[2];
50
51          switch (buffer[3]){
52              case addition:
53                  result = operand_1 + operand_2;
54                  break;
55              case subtraction:
56                  if (operand_1 >= operand_2){
57                      result = operand_1 - operand_2;
58                  }
59                  else result = 0;
60                  break;
61              case multiplication:
62                  if ((operand_1 < 16) && (operand_2 < 16)){
63                      result = operand_1 * operand_2;
64                  }
65                  else result = 0;
66                  break;
67              default:
68                  result = 0;
69                  break;
70          }
71
72
73          buffer[4] = result;
74
75          //printf("op1: %u\nop2: %u\noperation: %u\nResult:
  %u\n\n\n",operand_1, operand_2, buffer[3], result);
76
77          vTaskDelay(10);
78      }
79  }
80
81
82
83  int main(void)
84  {
85
86
87      I2C2& i2c = I2C2::getInstance();
88      const uint8_t slaveAddr = 0xC0;
89
90      i2c.initSlave(slaveAddr, &buffer[0], (size_t)sizeof(buffer));
91
92
93      //xTaskCreate(vReadBuffer, "ReadBuf", 1024, NULL, PRIORITY_LOW, NULL);
```

```
 94     xTaskCreate(vCalculate, "Calc", 1024, NULL, PRIORITY_LOW, NULL);
 95
 96     scheduler_add_task(new terminalTask(PRIORITY_HIGH));
 97
 98     scheduler_start();
 99
100     return -1;
101 }
```