NAME	M. Arslan Ahmad, M. Basit, Abdul Mateen
REG #	2020-EE-409, 401, 399

EXPERIMENT NO: 01

Objective:

- Introduction to the Micro – Controller and their understanding.
- Understanding of the software code for the blinking of LED.

Introduction:

A microcontroller is a compact integrated circuit designed to govern a specific operation in an embedded system. A typical microcontroller includes a processor, memory and input/output (I/O) peripherals on a single chip.

Sometimes referred to as an embedded controller or microcontroller unit (MCU), microcontrollers are found in vehicles, robots, office machines, medical devices, mobile radio transceivers, vending machines and home appliances among other devices.

Even at a time when Intel presented the first microprocessor with the 4004 there was already a demand for microcontrollers: The contemporary TMS1802 from Texas Instruments, designed for usage in calculators, was by the end of 1971 advertised for applications in cash registers, watches and measuring instruments. The TMS 1000, which was introduced in 1974, already included RAM, ROM, and I/O on-chip and can be seen as one of the first microcontrollers, even though it was called a microcomputer. The first controllers to gain really widespread use were the Intel 8048, which was integrated into PC keyboards, and its successor, the Intel 8051, as well as the 68HCxx series of microcontrollers from Motorola.

Microcontroller Features:

A microcontroller's processor will vary by application. Options range from the simple 4-bit, 8-bit or 16-bit processors to more complex 32-bit or 64-bit processors. In terms of memory, microcontrollers can use random access memory (RAM), flash memory, and EEPROM. Generally, microcontrollers are designed to be readily usable without additional computing components because they are designed with sufficient onboard memory as well as offering pins for general I/O operations, so they can directly interface with sensors and other components.

Some other Features are:

- Digital I/O Pins
- Analog I/O Pins
- Interrupts
- Timers
- Power Consumption and Sleep

Microcontroller Applications:

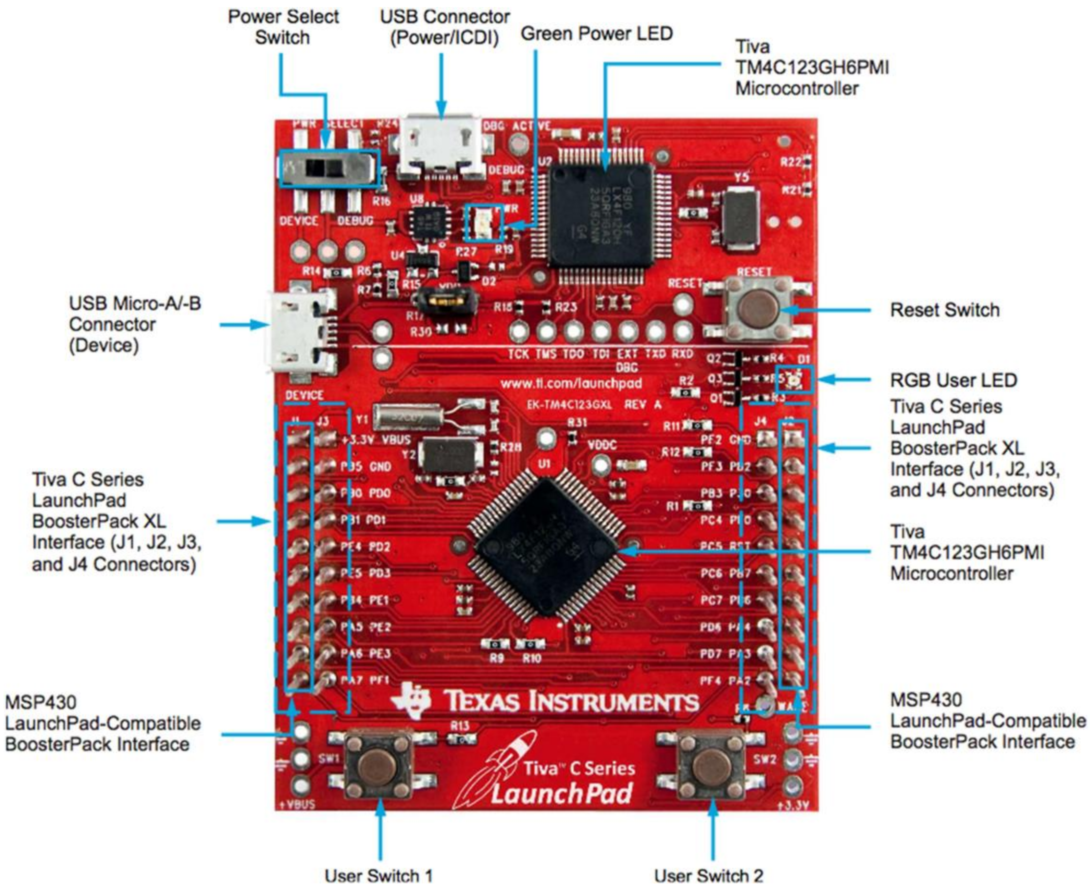
Today, microcontroller production counts are in the billions per year, and the controllers are integrated into many appliances we have grown used to, like

- household appliances (microwave, washing machine, coffee machine, . . .)
- telecommunication (mobile phones)
- automotive industry (fuel injection, ABS, . . .)
- aerospace industry
- industrial automation
- . . .

The simplest microcontrollers facilitate the operation of electromechanical systems found in everyday convenience items, such as ovens, refrigerators, toasters, mobile devices, key fobs, video games, televisions and lawn-watering systems. They are also common in office machines such as photocopiers, scanners, fax machines and printers, as well as smart meters, ATMs and security systems.

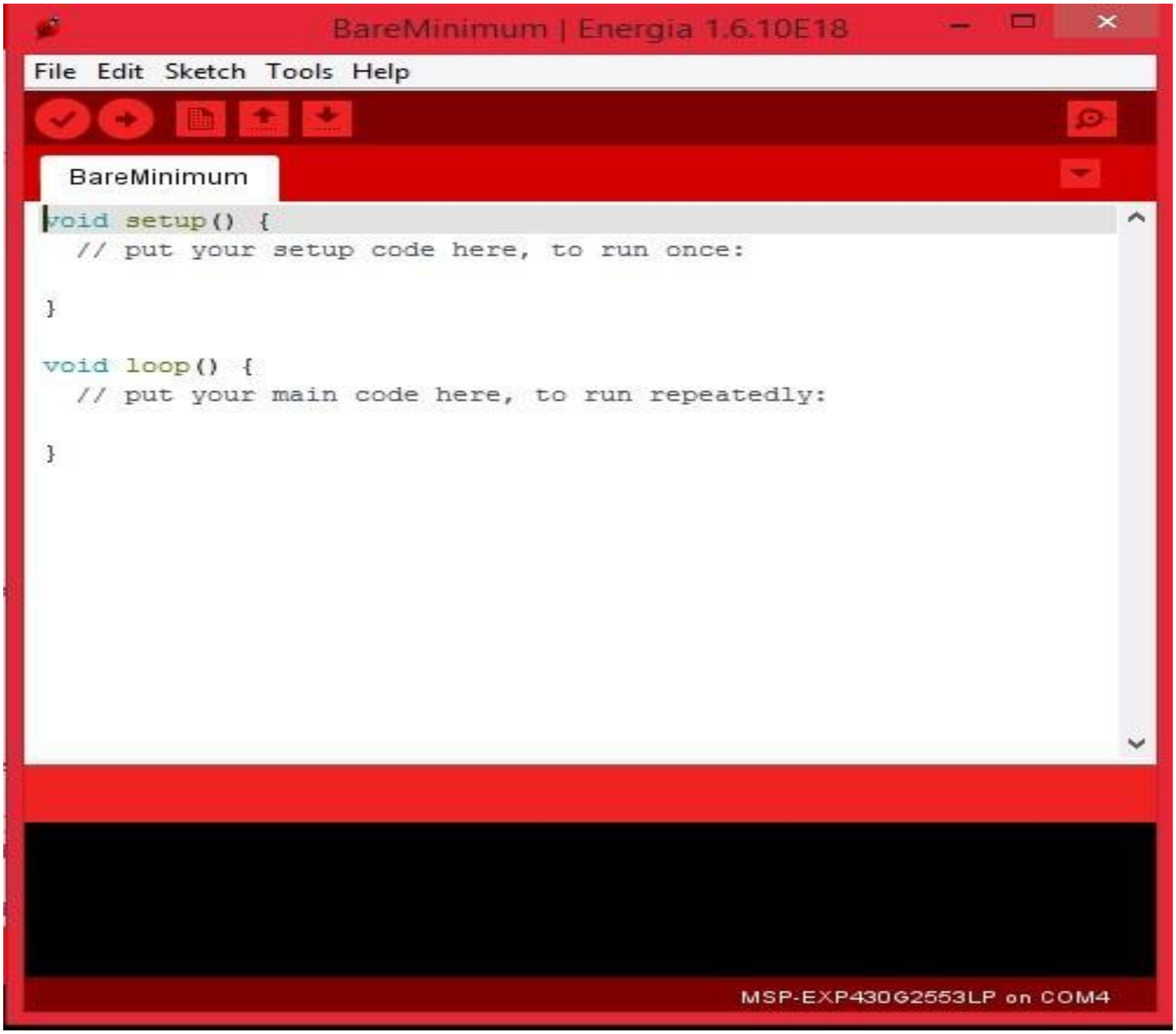
TM4C123GH6PM (ACTIVE):

High performance 32-bit ARM® Cortex®-M4F based MCU.



The TM4C123GH6PM microcontroller is targeted for industrial applications, including remote monitoring, electronic point-of-sale machines, test and measurement equipment, network appliances and switches, factory automation, HVAC and building control, gaming equipment, motion control, transportation, and fire and security.

Now we are going to understand how to code the TM4C123GH6PM using the Energia software. This software is same interface as Arduino's IDE. Here we use the same setup function for initialization of the pins and loop function for the repeated program. The following fig shows the coding interface of Energia.



Interface of Coding Environment for TIVA Microcontroller

Task 1: Energeia Code for LED Blinking:

```
// Task-1
// most launchpads have a red LED
#define LED RED_LED // Red led
//see pins_energia.h for more LED definitions
//#define LED GREEN_LED
// the setup routine runs once when you press reset:
void setup() {
  // initialize the digital pin as an output.
  pinMode(LED, OUTPUT); // Red led as output
}

// the loop routine runs over and over again forever:
void loop() {
  digitalWrite(LED, HIGH);   // turn the LED on (HIGH is the voltage
level)
  delay(1000);               // wait for a second
  digitalWrite(LED, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);               // wait for a second
}
```

Task 2: All Three LED’s

```
// Task-2
// Defining led's
#define LED1 RED_LED
#define LED2 BLUE_LED
#define LED3 GREEN_LED
void setup() {
  // initialize the digital pin as an output
  pinMode(LED1 , OUTPUT); // LED1 as o/p
  pinMode(LED2 , OUTPUT); // LED2 as o/p
  pinMode(LED3 , OUTPUT); // LED3 as o/p
}

// the loop routine runs over and over again forever:
void loop() {
  digitalWrite(LED1, HIGH);          // First red led will be high
  delay(1000);                       // Then wait a sec
  digitalWrite(LED1, LOW);          // Then red led will be low
  delay(1000);                     // Then wait a sec
  digitalWrite(LED2 , HIGH);      // Blue led high
  delay(1000);                   // Then wait a sec
  digitalWrite(LED2 , LOW);     // Then Blue led will be low
  delay(1000);                 // Then wait a sec
  digitalWrite(LED3, HIGH);   //Green led will be high
  delay(1000);               // Then wait a sec
  digitalWrite(LED3, LOW); //  Green led will be low
  delay(1000);             // Then wait a sec
}
```

Conclusion: