

Name : Tazmeen Afroz
Roll No : 22p-9252
COAL-LAB-TASK-10

This manual will guide you through the basics of microprocessors and programming them using Assembly Language. We will also explore how microprocessors are used in Arduino board.

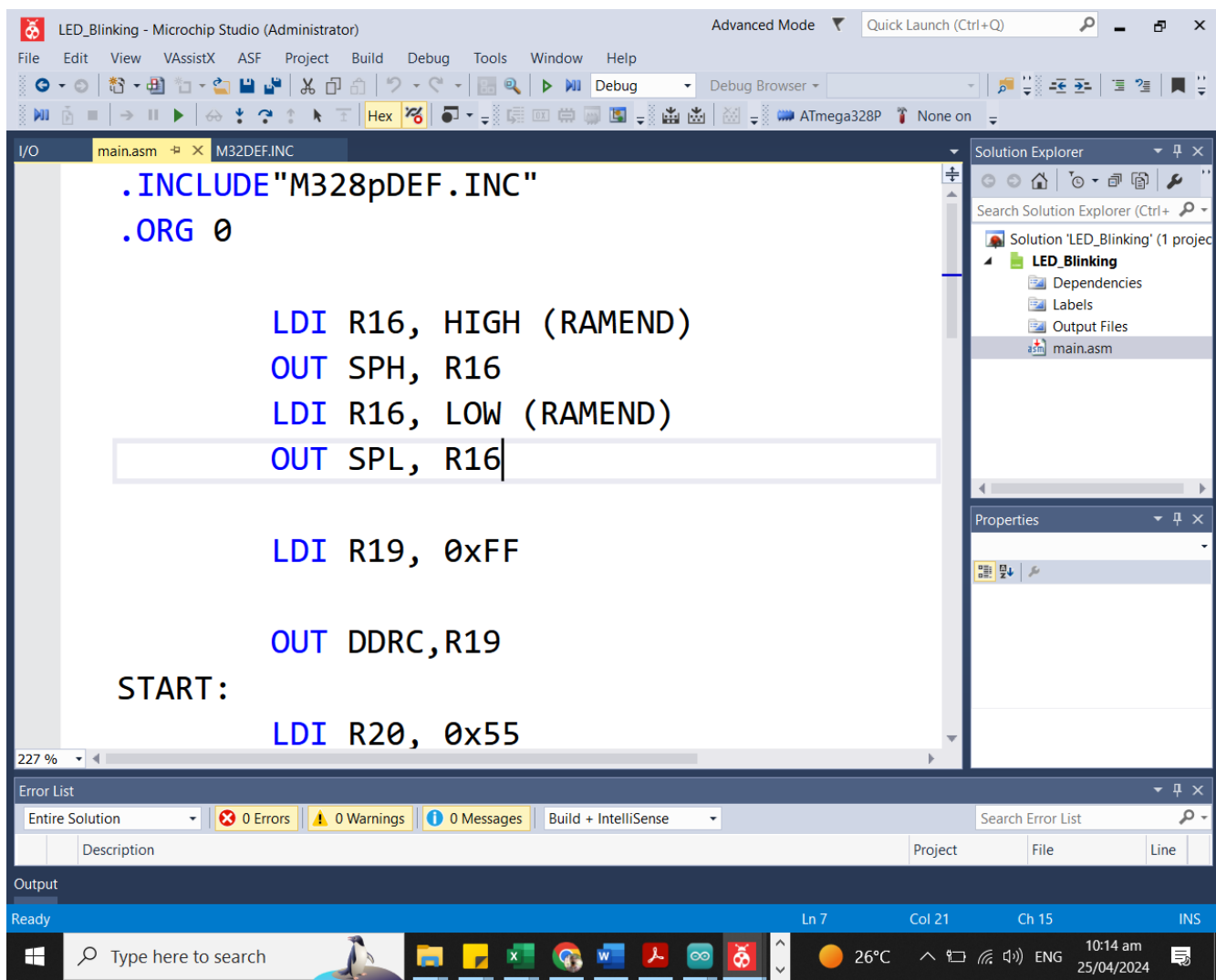
Microprocessor:

A microprocessor is a central processing unit (CPU) that contains the entire processing system of a computer on a single integrated circuit (IC). It executes software instructions and performs calculations.

Programming Microprocessor:

To program a microprocessor, we write code in Assembly Language, which is specific to the processor's architecture. The code is then assembled into machine code that the processor can execute directly.

Example: Writing a simple “COAL” on LED program in Assembly Language for the Microprocessor:



The screenshot displays the Microchip Studio IDE in Advanced Mode. The main editor window shows the assembly file 'main.asm' with the following code:

```
.INCLUDE "M328pDEF.INC"
.ORG 0

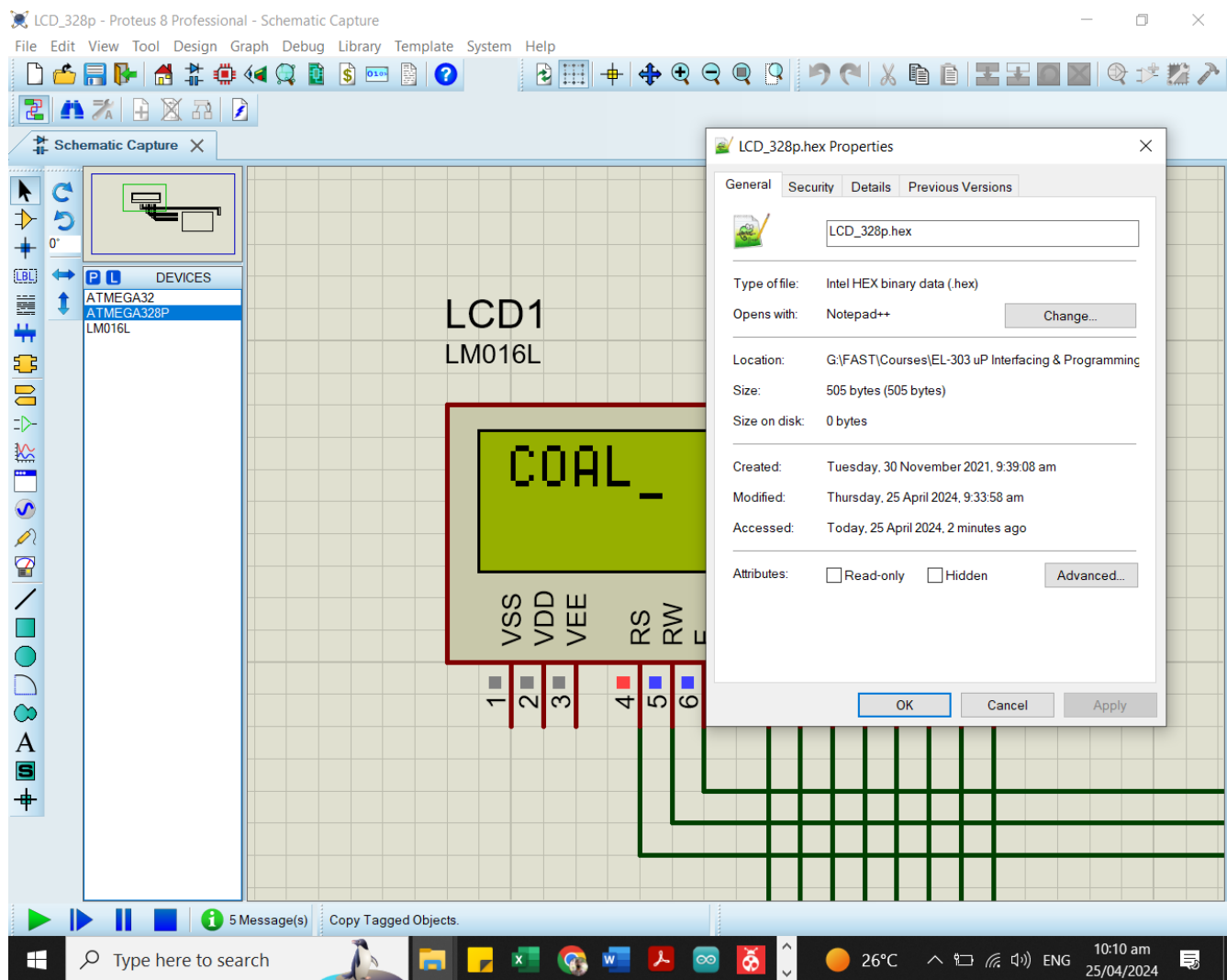
    LDI R16, HIGH (RAMEND)
    OUT SPH, R16
    LDI R16, LOW (RAMEND)
    OUT SPL, R16

    LDI R19, 0xFF
    OUT DDRC, R19

START:
    LDI R20, 0x55
```

The right-hand side of the IDE features a Solution Explorer showing the project 'LED_Blinking' with files 'Dependencies', 'Labels', 'Output Files', and 'main.asm'. Below it, the Properties window is visible. At the bottom, the Error List shows '0 Errors', '0 Warnings', and '0 Messages'. The status bar at the very bottom indicates 'Ready', 'Ln 7', 'Col 21', 'Ch 15', and 'INS'.

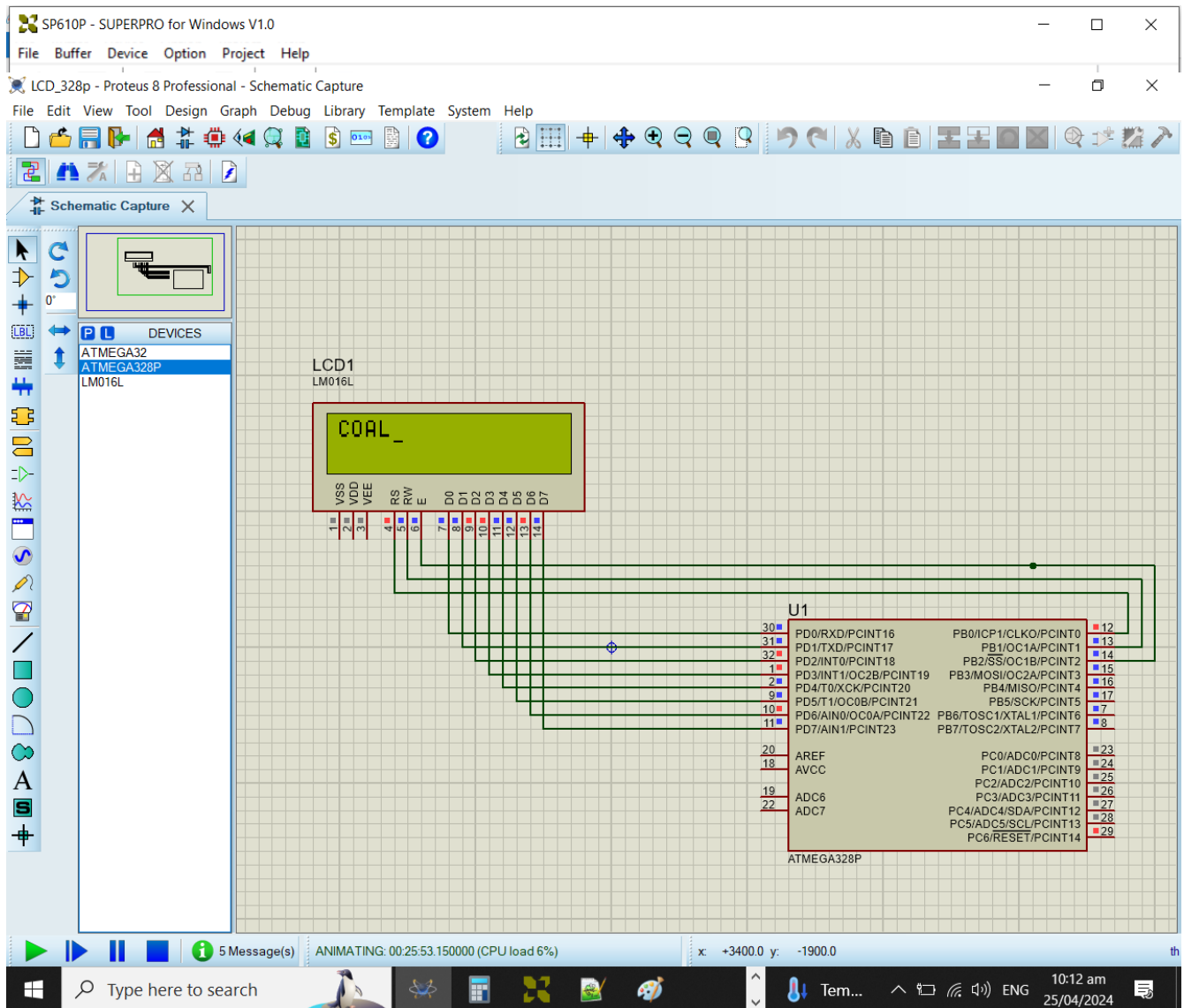
This code takes 505 bytes of memory.



Burning Code into Microprocessor:

To display the code on an LED using a microprocessor, we need to burn the code into the microprocessor using a programmer device. Here are the steps:

1. Write the Assembly Language code for the microprocessor.
2. Assemble the code into machine code using an assembler.
3. Connect the microprocessor to a programmer device (e.g., Intel 8085 Programmer).
4. Burn the machine code into the microprocessor using the programmer device.
5. Connect the microprocessor to an LED display circuit.
6. Power on the microprocessor and the LED display will show the output.



Microprocessor in Arduino:

Arduino boards use microprocessors like the ATmega328P to execute code. The microprocessor is programmed using the Arduino IDE, which compiles and uploads the code to the microprocessor.

To program a microcontroller like Arduino, we write code in a high-level language like C++.

Example: Writing a simple "COAL" on LED program in C++ for Arduino:

The screenshot shows the Arduino IDE 2.3.2 interface. The top menu bar includes File, Edit, Sketch, Tools, and Help. Below the menu is a toolbar with icons for checking, running, and uploading code, along with a dropdown menu for the board type, currently set to 'Arduino Uno'. The main editor window displays the code for 'HelloWorld.ino'. The code includes comments for initializing the LCD library and setting up the pins. The `setup()` function initializes the LCD and prints 'COAL' to the screen. The `loop()` function sets the cursor to the start of the first line. The output window at the bottom shows the compilation path and the resulting hex file.

```
45
46 // initialize the library by associating any needed LCD interface pin
47 // with the arduino pin number it is connected to
48 const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;
49 LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
50
51 void setup() {
52   // set up the LCD's number of columns and rows:
53   lcd.begin(16, 2);
54   // Print a message to the LCD.
55   lcd.setCursor(0, 0);
56   lcd.print("COAL");
57   //lcd.setCursor(4, 1);
58   //lcd.print("COAL Lab");
59 }
60
61 void loop() {
62   // set the cursor to column 0, line 1
63   // (note: line 1 is the second row, since counting begins with 0):
64   //lcd.setCursor(0, 1);
```

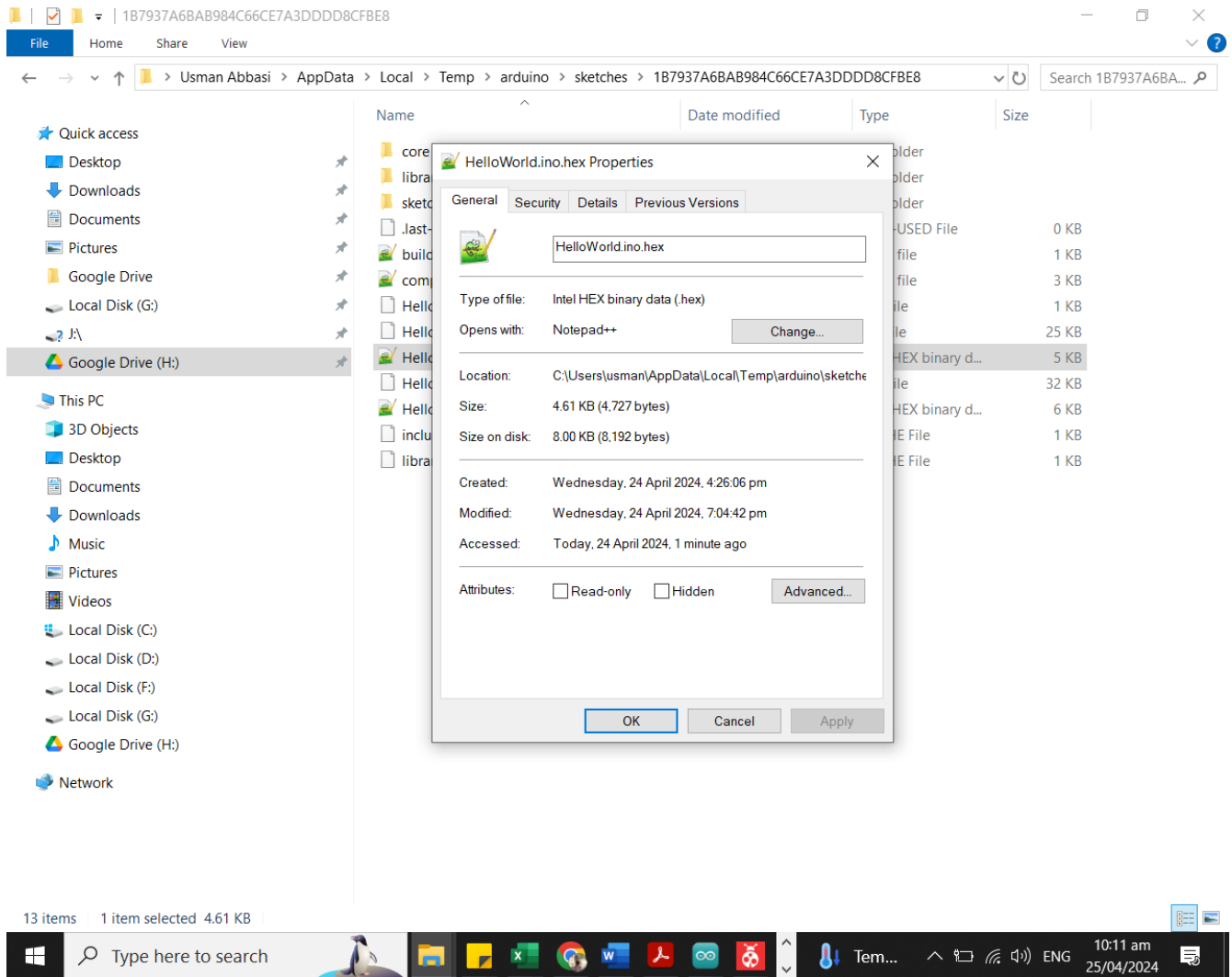
Output

```
\1B7937A6BAB984C66CE7A3DDDD8CFBE8/HelloWorld.ino.elf" "C:\Users\usman\AppData\Local\Temp\arduino\sketches\1B7937A6BAB984C66CE7A3DDDD8CFBE8/HelloWorld.ino.elf" "C:\Users\usman\AppData\Local\Temp\arduino\sketches\1B7937A6BAB984C66CE7A3DDDD8CFBE8/HelloWorld.ino.hex"
```

Ln 58, Col 27 Arduino Uno on COM33 [not connected] 2

Type here to search 26°C 10:15 am 25/04/2024

This code takes 8.00 KB (8,192 bytes) of memory.



Comparison:

As you can see, the Assembly Language code for the microprocessor takes much less memory (505 bytes) compared to the C++ code for the microcontroller (8.00 KB). This is because Assembly Language is specific to the processor's architecture and requires manual management of memory and resources, whereas C++ is a high-level language that abstracts away many details, resulting in larger code size.