Assignment 1

1. **What is Prototype? Open-Source and Closed-Source Prototype Platforms:**

   A prototype is an initial model or sample of a product, used to test and refine its design and functionality.

   - **Open-source:** These platforms provide open & free access to the design, software, and hardware files, allowing users to modify, share, and enhance them.

     E.g. **Arduino, Raspberry Pi.**

   - **Closed-source:** These are proprietary platforms where the design and software source code are not publicly available. Users are restricted from modifying or distributing the underlying code or design, and modifications are typically limited to authorized developers.

     E.g. **Google Earth, Skype, WinRAR, Microsoft Windows, Mac OS**

2. **What is Arduino?**

   Arduino is an open-source electronics platform based on easy-to-use hardware and software. It enables users to create interactive projects by connecting components like sensors, LEDs, and motors to microcontrollers, which can be programmed to read inputs and control outputs. Arduino is popular among hobbyists, educators, and professionals for prototyping and building various electronic projects due to its simplicity and versatility.

   It has two parts:

   - a Circuit Board
   - a program that lets people tell the circuit board what to do.

3. **What are the Arduino Uno R3 Key Specifications?**

   - Main Processor
     - **ATmega328P**, a modified Harvard architecture 8-bit RISC* processor core. **Reduced Instruction Set Computer*
   - Memory (SRAM, FLASH MEMORY, EEPROM)
     - **SRAM:** Static Random Access Memory a type of RAM which uses a flip-flop to store 1-bit of data.

        - The system's temporary data or run-time data is stored in the SRAM; with a size of 2KB.

- **FLASH MEMORY:** In Arduino, the Flash stores the application code to be run.
    - The Size of Flash Memory is 32KB.
- **EEPROM:** An Electrically Erasable Programmable Read-Only Memory. It is a form of non-volatile memory that can remember things with the power being turned off, or after resetting the Arduino.
    - The Size of EEPROM is 1KB.

➢ I/O Pins

- An Arduino has **14 digital input/output pins** (of which 6 can be used as PWM* outputs), 6 analog inputs.

Assignment 2

1. **What is an Encoding format? List down encoding formats for various types of data (Text, Number, Photo, Audio, Video).**

   Encoding Formats: A Brief Overview ~

   Encoding format is essentially a standardized method to translate data into a format that computers can understand and process efficiently. It's like converting human language into a language computer can comprehend.

   ➢ Different Encoding Formats:

   - Text Encoding

     Text data is represented by characters, which are assigned specific numerical values. These values are then stored and transmitted in a specific format.

     - **ASCII (American Standard Code for Information Interchange):** Represents 128 characters, including uppercase and lowercase letters, numbers, punctuation, and control characters.
     - **Unicode**: A more comprehensive character encoding standard, capable of representing text in almost all languages. It includes ASCII as a subset.
     - **UTF-8**: A variable-length encoding scheme compatible with ASCII, widely used for web pages and email.

   - Number Encoding

     Numbers are represented digitally using different numerical systems.

     - **Binary**: Uses only 0s and 1s.
     - **Decimal:** The base-10 system we commonly use.
     - **Hexadecimal**: Uses 16 digits (0-9, A-F).
     - **Floating-point**: Represents real numbers with a decimal point.

- Image Encoding

  Images are represented by pixels, each with color information.

  - **JPEG (Joint Photographic Experts Group)**: Lossy compression, suitable for photographs.
  - **PNG (Portable Network Graphics)**: Lossless compression, suitable for images with sharp edges and text.
  - **GIF (Graphics Interchange Format)**: Supports animation and transparency, often used for simple images and logos.
  - **BMP (Bitmap)**: Uncompressed format, large file size.
  - **TIFF (Tagged Image File Format)**: Lossless compression, supports various image depths.

- Audio Encoding

  Audio data is represented by digital samples of sound waves.

  - **MP3**: Lossy compression, widely used for music.
  - **AAC (Advanced Audio Coding)**: Lossy compression, often used in iTunes
  - and digital broadcasting.
  - **WAV (Waveform Audio File)**: Lossless compression, high-quality audio format.
  - **FLAC (Free Lossless Audio Codec)**: Lossless compression, maintains audio quality.

- Video Encoding

  Video data combines image and audio data.

  - **MP4 (MPEG-4 Part 14)**: Commonly used for video storage and distribution.
  - **AVI (Audio Video Interleave)**: Container format supporting various codecs.
  - **MOV (QuickTime Movie)**: Apple's video format.
  - **WMV (Windows Media Video)**: Microsoft's video format.

Assignment 3

1. **Explain Basic Structure of an Arduino Program.**
   - The basic structure of an Arduino program, also called a *sketch*, consists of two main functions:

     setup() is the preparation, **loop()** is the execution.
   - Both functions are required for the program to work.

```
void setup() {
        //statements;
}
void loop() {
        //statements;
}
```

The setup() function is called once, when the Arduino board is first turned on or reset. It is used to initialize the board and set up the hardware.

The setup function should follow the declaration of any variables at the very beginning of the program. It is **the first function to run in the program**, is run only once, and is used to set **pinMode** or initialize serial communication.

Setting pinMode: This tells the Arduino **whether a specific pin is going to be used for input** (reading data) or **output** (sending data).

For example, if you have an LED connected to pin 13, you would specify in the setup() function that pin 13 is an output pin.

Initializing Serial Communication: This is like **opening a communication channel** between your Arduino and your computer or another device. This is useful for **sending data back and forth**.

The loop() function is called repeatedly, until the Arduino board is turned off or reset. It is where the Arduino program does most of its work.

The loop function follows next and includes the code to be executed continuously – reading inputs, triggering outputs, etc.

Assignment 4

How do UART, I²C, SPI, CAN, and USB communication protocols differ in terms of data transmission speed, complexity, pin usage, and device-to-device communication? What are the key features that make each protocol suitable for specific applications, and in what types of embedded systems would each be most commonly used?

Following is the comparison of the UART, I²C, SPI, CAN, and USB communication protocols in terms of data transmission speed, complexity, pin usage, and device-to-device communication, as well as their applications:

	Protocol	**Data Transmission Speed**	**Complexity**	**Pin Usage**	**Device-to-Device Communication**	
	UART	Up to ~1 Mbps	Simple	2 pins (TX, RX)	Only supports 1-to-1	
	I2C	Up to 3.4 Mbps	Moderate	2 pins (SDA, SCL)	Supports 1 master, multiple slaves or multi-master	
	SPI	Up to ~50 Mbps	Moderate to high	4 pins (MOSI, MISO, SCLK, SS)	1 master, multiple slaves	
	CAN	Up to 1 Mbps (standard), 5 Mbps (CAN FD)	Moderate to high	2 pins (CAN_H, CAN_L)	Multi-master, multi-slave	
	USB	Up to 20 Gbps (USB 3.2)	High	Varies (4-9 pins)	1 host, multiple devices	

Key Features and Application Suitability

1. **UART (Universal Asynchronous Receiver-Transmitter):**

   Key Features: Simple, requires only two pins, reliable for point-to-point communication.

Application Suitability: Suitable for low-speed communication and where simplicity and minimal wiring are required.

Common Uses: Embedded systems requiring simple, low-speed data transfer between two devices, e.g., microcontroller and a peripheral.

2. **I²C (Inter-Integrated Circuit):**

   Key Features: Two-wire bus, multiple device support with addressing, synchronous.

   Application Suitability: Ideal for connecting multiple low-speed devices like sensors to a microcontroller.

   Common Uses: Embedded systems with multiple peripherals (e.g., temperature, humidity, pressure sensors) due to its simplicity in expanding device connections on a single bus.

3. **SPI (Serial Peripheral Interface):**

   Key Features: High-speed, full-duplex communication, requires more pins, flexible but requires additional pins per slave.

   Application Suitability: Used in systems needing high data transfer rates and low latency, such as with displays or high-speed memory.

   Common Uses: Applications requiring fast data exchange, such as SD card interfacing, displays, and flash memory in data-intensive embedded systems.

4. **CAN (Controller Area Network):**

   Key Features: Robust, supports differential signaling, resistant to noise, multi-master, suitable for real-time communication.

   Application Suitability: Built for high-reliability environments where fault tolerance and robustness are crucial.

   Common Uses: Automotive systems (engine control units, body control modules), industrial automation, and robotics, where environmental noise is a concern.

5. **USB (Universal Serial Bus):**

   Key Features: High-speed, complex protocol with multiple layers, plug-and-play support, power delivery.

   Application Suitability: Designed for connecting computers to peripherals and suitable for applications that require fast data transfer and easy connection to computers.

Common Uses: Peripherals like mice, keyboards, storage devices, mobile phones, and industrial devices for fast and easy interfacing with computers.

Each protocol is chosen for specific applications based on its speed, complexity, robustness, and ease of use, which help optimize communication in various embedded systems.