**INTERNET OF THINGS ASSIGNMENT RECORD**

**Subject code: BTCS-AMDS-009T**

|  |  |
| --- | --- |
| Name: | Subham Pratik Das |
| Registration Number: | FET-BAML-2022-2026-013 |
| Course: | B.Tech CSE-AIML |
| Semester: | 5th |
| Faculty: | Mr. Biswajeeban Mishra &  Mr. Pritam Nanda |

|  |  |
| --- | --- |
| Remarks |  |
| Signature |  |



SRI SRI UNIVERSITY

Bidyadharpur, Cuttack, Odisha.

Index

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sl. No.** | **Date** | **Experiment/Case Study** | **Page No.** | **Remark** |
| **1** | 9-8-2024 | Prototyping and Arduino UNO R3. | 1 |  |
| **2** | 9-8-2024 | Encoding formats. | 3 |  |
| **3** | 9-8-2024 | Basic Structure of an Arduino Program. | 5 |  |
| **4** | 7-10-2024 | The architecture of modern computers, microcontrollers, and memory location. | 7 |  |
| **5** | 22-10-2024 | UART, I2C, SPI, CAN, and USB communication protocols. | 11 |  |
| **6** |  |  |  |  |
| **7** |  |  |  |  |
| **8** |  |  |  |  |
| **9** |  |  |  |  |
| **10** |  |  |  |  |
| **11** |  |  |  |  |
| **12** |  |  |  |  |
| **13** |  |  |  |  |
| **14** |  |  |  |  |
| **15** |  |  |  |  |
| **16** |  |  |  |  |

***Assignment No.:1 Date: 9-8-2024***

**Q1. *What is a Prototype?***

***What are Open source and closed source prototype platforms?***

**A:**

* **Prototype**

A prototype is a preliminary model, sample, or version of a product or system used to test a concept or process. It's a tangible representation of an idea, allowing for evaluation, feedback, and refinement before full-scale development. Prototypes can be physical, digital, or a combination of both.

* **Open Source and Closed Source Prototype Platforms**
* Open Source Prototype Platforms: These platforms provide the underlying code and design freely accessible to the public. Users can modify, distribute, and build upon the platform. Examples include Arduino, Raspberry Pi, and open source CAD software like FreeCAD.
* Closed Source Prototype Platforms: These platforms keep the source code proprietary, restricting access and modification. Users typically pay for licenses to use the platform. Examples include many commercial 3D printing software, electronic design automation (EDA) tools, and rapid prototyping machines.

**Q2. *What is Arduino?***

**A: Arduino** is an open-source electronics platform based on easy-to-use hardware and software. It's designed for artists, designers, hobbyists, and anyone interested in creating interactive objects or environments. Arduino boards are microcontroller-based, meaning they have a small computer on board that can be programmed to control various electronic components.

**Q3. *Write down the Arduino Uno R3 Key Specifications.***

**A3.**

* **Main Processor :**

ATmega328P

* **Memory :**

SRAM: 2 KB

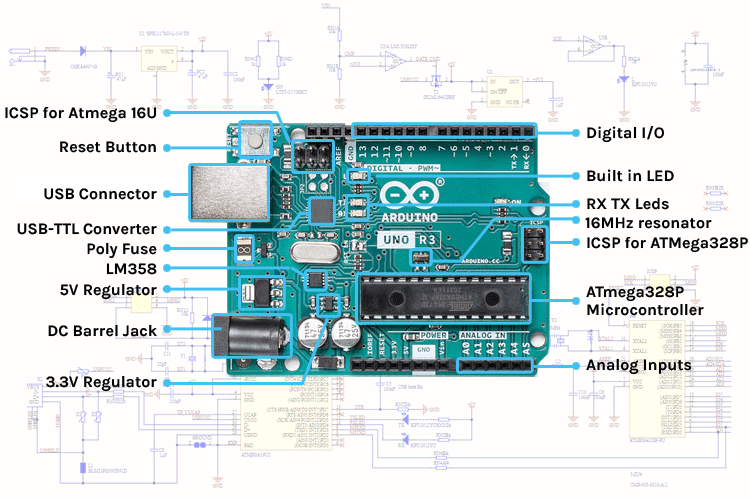
FLASH MEMORY: 32 KB (0.5 KB used by bootloader)

EEPROM: 1 KB

* **I/O Pins :**

Digital I/O pins: 14 (of which 6 can be used as PWM outputs)

Analog input pins: 6



***Assignment No.:2 Date: 9-8-2024***

**Q1. *What is an Encoding format? List down encoding formats for various types of data  ( Text, Number, Photo, Audio, Video).***

**A:** An encoding format is a standardized method for converting data into a specific digital format for efficient storage and transmission. It translates information from human-readable form into a format that computers can interpret. Here’s how encoding formats are used in different contexts.

**Text:**

* [**ASCII (American Standard Code for Information Interchange)**](https://en.wikipedia.org/wiki/ASCII)
* [**UTF-8 (Unicode Transformation Format - 8-bit)**](https://en.wikipedia.org/wiki/UTF-8)
* [**UTF-16 (Unicode Transformation Format - 16-bit)**](https://en.wikipedia.org/wiki/UTF-16)
* [**UTF-32 (Unicode Transformation Format - 32-bit)**](https://en.wikipedia.org/wiki/UTF-32)
* [**ISO 8859-1 (Latin-1)**](https://en.wikipedia.org/wiki/ISO/IEC_8859-1)

**Numbers:**

* [**Binary (Base-2)**](https://en.wikipedia.org/wiki/Binary_number)
* [**Decimal (Base-10)**](https://en.wikipedia.org/wiki/Decimal)
* [**Hexadecimal (Base-16)**](https://en.wikipedia.org/wiki/Hexadecimal)
* [**IEEE 754**](https://en.wikipedia.org/wiki/IEEE_754)
* [**BCD (Binary-Coded Decimal)**](https://en.wikipedia.org/wiki/Binary-coded_decimal)

**Photos or Images:**

* [**JPEG (Joint Photographic Experts Group)**](https://en.wikipedia.org/wiki/JPEG)
* [**PNG (Portable Network Graphics)**](https://en.wikipedia.org/wiki/Portable_Network_Graphics)
* [**GIF (Graphics Interchange Format)**](https://en.wikipedia.org/wiki/GIF)
* [**BMP (Bitmap)**](https://en.wikipedia.org/wiki/BMP_file_format)
* [**TIFF (Tagged Image File Format)**](https://en.wikipedia.org/wiki/TIFF)

**Audio:**

* [**MP3 (MPEG-1 Audio Layer III)**](https://en.wikipedia.org/wiki/MP3)
* [**WAV (Waveform Audio File Format)**](https://en.wikipedia.org/wiki/WAV)
* [**AAC (Advanced Audio Coding)**](https://en.wikipedia.org/wiki/Advanced_Audio_Coding)
* [**FLAC (Free Lossless Audio Codec)**](https://en.wikipedia.org/wiki/FLAC)
* [**OGG (Ogg Vorbis)**](https://en.wikipedia.org/wiki/Vorbis)

**Video:**

* [**MP4 (MPEG-4 Part 14)**](https://en.wikipedia.org/wiki/MPEG-4_Part_14)
* [**AVI (Audio Video Interleave)**](https://en.wikipedia.org/wiki/Audio_Video_Interleave)
* [**MKV (Matroska Video)**](https://en.wikipedia.org/wiki/Matroska)
* [**MOV (QuickTime Movie)**](https://en.wikipedia.org/wiki/QuickTime_File_Format)
* [**WMV (Windows Media Video)**](https://en.wikipedia.org/wiki/Windows_Media_Video)

***Assignment No.:3 Date: 9-8-2024***

***Q1. Explain Basic Structure of an Arduino Program.***

***A1.*** An Arduino program is also called a sketch.

**Basic Structure:**

void setup()

{

statements;

}

void loop()

{

statements;

}

It consists of two main functions:

*1) setup() function*

* Runs only once when the Arduino board starts or resets.
* Used to initialize hardware components, set pin modes (input or output), and define initial values.

*2) loop() function*

* Runs repeatedly after the setup() function completes.
* Contains the core logic of the program.
* This is where the Arduino performs actions and interacts with the environment.

Both functions are enclosed in curly braces {}

Every statement inside these functions is ended with a semicolon ‘ ; ’

**Basic Sketch:**

This simple sketch blinks an LED connected to pin 13.

void setup() {

pinMode(13, OUTPUT); // Set pin 13 as an output

}

void loop() {

digitalWrite(13, HIGH); // Turn LED on

delay(1000); // Wait for 1 second

digitalWrite(13, LOW); // Turn LED off

delay(1000); // Wait for 1 second

}

***Assignment No.:4 Date: 9-8-2024***

***Q. 1. The Architecture of Modern Computers:***

* ***How does the architecture of modern CPUs (e.g., x86-64 architecture) handle parallelism, and what are the implications for software design and performance?***
* ***What are the key differences between RISC (Reduced Instruction Set Computing) and CISC (Complex Instruction Set Computing) architectures, and how do these differences influence the design of operating systems?***
* ***Explain the differences between the CISC (Complex Instruction Set Computing) and RISC (Reduced Instruction Set Computing) architectures. Additionally, compare the John von Neumann architecture with the Harvard architecture, focusing on their memory organization and instruction processing. How do these differences impact the performance and design of modern processors?***

***2. Microcontrollers (e.g., Arduino Uno R3):***

* ***How does the AVR architecture used in the Arduino Uno R3 differ from the architecture used in more advanced microcontrollers like ARM Cortex-M?***
* ***What is the significance of using an 8-bit microcontroller (like the ATmega328P in Arduino Uno) compared to 16-bit or 32-bit microcontrollers in terms of memory addressing, processing power, and application suitability?***

***3. Memory Segmentation:***

* ***How does memory segmentation in x86 architecture support backward compatibility, and what are the advantages and disadvantages compared to flat memory models used in modern 64-bit systems?***
* ***What is the difference between User Space and Kernel Space in the virtual memory layout of modern computers, and why is this separation important? Additionally, how does memory segmentation work, and what role does it play in managing memory in older vs. modern computing systems?***

***4. Memory Location and Byte Addressability:***

* ***Why is each memory location generally 1 byte in modern computer systems, and how does this design choice impact the way data structures are stored and accessed in memory?***
* ***How does the concept of endianness (big-endian vs. little-endian) relate to the byte-wise organization of memory, and what are the challenges developers face when working with systems of different endianness?***

***A1.* 1. The Architecture of Modern Computers**

**Parallelism in Modern CPUs**

Modern CPUs employ various techniques to achieve parallelism, including:

* **Pipeline:** Instructions are broken down into stages, and multiple instructions are processed simultaneously in different stages.
* **Superscalar:** Multiple instructions are executed simultaneously in a single cycle.
* **Multi-core:** A single CPU contains multiple cores, each capable of executing instructions independently.
* **SIMD (Single Instruction, Multiple Data):** A single instruction operates on multiple data elements simultaneously.

These techniques significantly improve performance but require careful software design to maximize their benefits. For example, programmers must ensure that instructions are not dependent on each other to avoid pipeline stalls.

**RISC vs. CISC Architectures**

* **RISC (Reduced Instruction Set Computing):** Uses a small set of simple instructions that can be executed in a single cycle. This leads to simpler and faster hardware but requires more complex software.
* **CISC (Complex Instruction Set Computing):** Uses a large set of complex instructions that can perform multiple operations in a single cycle. This simplifies software but requires more complex and slower hardware.

RISC architectures are generally preferred in modern systems due to their simplicity and efficiency. However, CISC architectures are still used in some legacy systems. The choice of architecture influences the design of operating systems, as the OS must be tailored to the specific instruction set.

**CISC vs. RISC and Von Neumann vs. Harvard Architectures**

* **CISC vs. RISC:** As discussed above, CISC has complex instructions while RISC has simpler ones.
* **Von Neumann Architecture:** Uses a single memory for both instructions and data. This is the most common architecture in modern computers.
* **Harvard Architecture:** Uses separate memories for instructions and data. This can improve performance but is more complex to implement.

The Von Neumann architecture is more common due to its simplicity and cost-effectiveness. However, the Harvard architecture can be advantageous in certain applications, such as digital signal processing, where there is a high demand for both instruction and data fetches.

**2. MicroControllers (e.g., Arduino Uno R3)**

**AVR vs. ARM Architecture**

* **AVR (Atmel RISC Architecture):** Used in the Arduino Uno R3. It is a 8-bit RISC architecture with a simple instruction set and low power consumption.
* **ARM Cortex-M:** A family of 32-bit RISC architectures used in more advanced microcontrollers. They offer higher performance, larger address space, and more features than AVR.

**8-bit vs. 16/32-bit Microcontrollers**

* **Memory Addressing:** 8-bit microcontrollers have a smaller address space, limiting the amount of memory they can access.
* **Processing Power:** 16-bit and 32-bit microcontrollers have higher processing power, making them suitable for more demanding applications.
* **Application Suitability:** 8-bit microcontrollers are well-suited for simple applications with limited memory and processing requirements, while 16-bit and 32-bit microcontrollers are better for more complex applications.

**3. Memory Segmentation**

**Memory Segmentation and Backward Compatibility**

Memory segmentation in x86 architecture divides the address space into segments, which can be used to isolate different processes or parts of a program. This allows for backward compatibility with older 16-bit software, which used segmented memory. However, segmentation can be inefficient and complex to manage.

**User Space vs. Kernel Space**

* **User Space:** The area of memory accessible to user applications.
* **Kernel Space:** The area of memory reserved for the operating system kernel.

This separation is important for security and isolation. It prevents user applications from accessing or modifying kernel code, which could lead to system instability or security breaches.

**Memory Segmentation in Older vs. Modern Systems**

* **Older Systems:** Memory segmentation was used extensively to manage memory and protect processes.
* **Modern Systems:** Segmentation is still used in some cases, but flat memory models are more common. Flat memory models provide a simpler and more efficient way to manage memory, especially in 64-bit systems.

**4. Memory Location and Byte Addressability**

**1-Byte Memory Locations**

Each memory location is generally 1 byte in modern computer systems because it is a convenient unit of data. Bytes can represent a wide range of values, from characters to small integers. This design choice simplifies memory addressing and data manipulation.

**Endianness**

Endianness refers to the order in which bytes are stored in memory.

* **Big-Endian:** The most significant byte is stored first.
* **Little-Endian:** The least significant byte is stored first.

Developers must be aware of the endianness of the system they are working with to ensure correct data interpretation. Misunderstanding endianness can lead to data corruption and errors.

***Assignment No.:5 Date: 9-8-2024***

***Q. 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?***

***A. 1. UART (Universal Asynchronous Receiver/Transmitter)***

* *Data Transmission Speed: Up to 1 Mbps (depends on the specific microcontroller and distance).*
* *Complexity: Simple to implement as it requires no clock signal, making it asynchronous.*
* *Pin Usage: Requires only two pins – one for transmitting data (TX) and one for receiving data (RX).*
* *Device-to-Device Communication: Primarily supports two devices, with the possibility of adding more devices by connecting RX and TX lines, though this can become unreliable for multiple devices.*

***Key Features***

* *UART uses start and stop bits to define data frames, making synchronization easier without a clock signal.*
* *It includes error-checking mechanisms like parity bits.*

***Applications****Due to its simplicity, UART is widely used in low-speed, point-to-point communication, such as serial console interfaces, GPS modules, and basic microcontroller-to-microcontroller communication. It’s ideal for systems where only two devices need to communicate directly.*

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

* *Data Transmission Speed: Typical speeds are 100 kHz (standard), 400 kHz (fast mode), up to 3.4 MHz (high-speed mode).*
* *Complexity: Moderately complex, with a master-slave configuration that manages multiple devices.*
* *Pin Usage: Requires only two wires – Serial Data Line (SDA) and Serial Clock Line (SCL).*
* *Device-to-Device Communication: Supports communication between multiple devices on the same bus. Each device has a unique address, allowing a single master to communicate with multiple slaves.*

***Key Features***

* *I²C is a synchronous protocol, with a clock signal provided by the master device.*
* *The protocol uses acknowledgments to confirm data reception, improving reliability.*
* *Pull-up resistors are required for SDA and SCL lines to operate correctly.*

***Applications****I²C is widely used for connecting peripherals like sensors, EEPROMs, real-time clocks, and LCDs in applications where moderate speed and multi-device communication are needed. It’s suitable for systems with limited pin availability, such as home automation, portable devices, and sensor networks.*

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

* *Data Transmission Speed: Speeds vary but can reach up to 50 Mbps, depending on the hardware.*
* *Complexity: Slightly complex, as it involves four connections and requires careful timing for multiple devices.*
* *Pin Usage: Uses four lines – MISO (Master In Slave Out), MOSI (Master Out Slave In), SCK (Serial Clock), and SS (Slave Select).*
* *Device-to-Device Communication: Allows one master to communicate with multiple slaves, though each slave needs a dedicated SS pin from the master.*

***Key Features***

* *SPI is a full-duplex, synchronous protocol, meaning data can be sent and received simultaneously.*
* *It is fast and has low latency due to the dedicated clock line.*
* *No addressing scheme is needed, so communication setup is faster but requires more pins for each device.*

***Applications****SPI is commonly used in high-speed applications where data transfer rates are crucial, such as displays, SD cards, and communication with high-speed peripherals. It’s popular in industrial controls, consumer electronics, and applications where high data rates and reliable, low-latency communication are needed.*

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

* *Data Transmission Speed: Up to 1 Mbps, suitable for automotive applications where reliability is more critical than speed.*
* *Complexity: More complex due to its differential signaling and error detection/correction mechanisms.*
* *Pin Usage: Requires two wires (CAN\_H and CAN\_L) for differential signaling, making it resistant to noise.*
* *Device-to-Device Communication: Multi-master, meaning multiple devices can transmit data on the same bus. Devices identify each other by unique message IDs instead of addresses.*

***Key Features***

* *CAN protocol has excellent error-checking and arbitration mechanisms, making it highly reliable.*
* *Data packets have a priority-based identifier system, allowing critical data to transmit first.*
* *Supports message acknowledgment and error detection, which is essential for safety-critical applications.*

***Applications****CAN is predominantly used in automotive and industrial systems, where reliability and fault tolerance are critical. It’s used for in-vehicle communication in cars, such as between the engine control unit and sensors. CAN is also applied in robotics and factory automation.*

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

* *Data Transmission Speed: Supports various speeds, including Low-Speed (1.5 Mbps), Full-Speed (12 Mbps), High-Speed (480 Mbps), and SuperSpeed (up to 5 Gbps).*
* *Complexity: More complex than other protocols due to extensive protocol layers and required drivers.*
* *Pin Usage: Typically uses four wires – VCC, GND, D+, and D- (for differential signaling).*
* *Device-to-Device Communication: Uses a host-controller model, where one device (the host) initiates communication, allowing multiple peripheral devices to be connected via hubs.*

***Key Features***

* *USB is plug-and-play, allowing easy connection and disconnection of devices.*
* *It provides power and data over a single cable, making it suitable for portable devices.*
* *The protocol has built-in error-checking and handshaking mechanisms, ensuring reliable data transfer.*

***Applications****USB is ideal for applications requiring high data transfer rates and flexibility, such as computers, external storage, and peripherals (like printers, keyboards, and mice). It’s commonly found in consumer electronics, personal computers, and portable devices.*