DEPARTMENT OF:- ***Software Engineering***
COURSE TITLE:- ***Real Time And Embedded Systems***
COURSE CODE:- ***SEng 4031***

Individual Assignment

Name: Mikiyas Bayle Id No: DBUR/0262/13

Submitted Date . 03/06/2024
Submitted to . Yasregdal.

1, Read and write short note about the differences between 8051, ARM, AVR, And PIC Microcontrollers and their unique features.

What is a Microcontroller?

A micro-controller can be comparable to a little stand alone computer; it is an extremely powerful device, which is able of executing a series of pre-programmed tasks and interacting with extra hardware devices. Being packed in a tiny integrated circuit (IC) whose size and weight is regularly negligible, it is becoming the perfect controller for as robots or any machines required some type of intelligent automation. A single microcontroller can be enough to manage a small mobile robot, an automatic washer machine or a security system. Several microcontrollers contains a memory to store the program to be executed, and a lot of input/output lines that can be a used to act jointly with other devices, like reading the state of a sensor or controlling a motor.

1,, 8051 Microcontroller

8051 microcontroller is an 8-bit family of microcontroller is developed by the Intel in the year 1981. This is one of the popular families of microcontroller are being used all across the world. This microcontroller was moreover referred as "system on a chip" since it has 128 bytes of RAM, 4Kbytes of a ROM, 2 Timers, 1 Serial port, and 4 ports on a single chip. The CPU can also work for 8bits of data at a time since 8051 is an 8-bit processor. In case the data is bigger than 8 bits, then it has to be broken into parts so that the CPU can process easily. Most manufacturers contain put 4Kbytes of ROM even though the number of ROM can be exceeded up to 64 K bytes.

8051 Microcontroller

The 8051 has been in utilized in a wide number of devices, mostly because it is easy to integrate into a project or make a device approximately. The following are the major areas of focus:

a) **Energy Management:** Efficient metering systems facilitate in controlling energy usage in homes and manufacturing applications. These metering systems are prepared capable by incorporating microcontrollers.
b) **Touch screens:** A high number of microcontroller providers incorporate touch-sensing capabilities in their designs. Portable electronics such as cell phones, media players and gaming devices are examples of microcontroller-based touch screens.
c) **Automobiles:** The 8051 finds wide taking in providing automobile solutions. They are broadly used in hybrid vehicles to handle engine variants. Furthermore, functions such as cruise control and anti-brake system have been prepared more capable with the use of microcontrollers.

d) **Medical Devices:** Moveable medical devices such as blood pressure and glucose monitors use microcontrollers will to show data, thus provided that higher reliability in providing medical results.

2, PIC Microcontroller

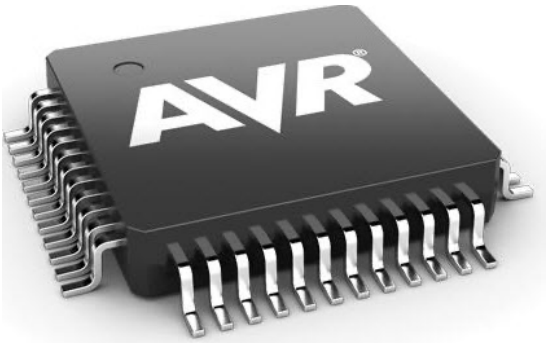
Peripheral Interface Controller (PIC) is microcontroller developed by a Microchip, PIC microcontroller is fast and simple to implement program when we contrast other microcontrollers like 8051. The ease of programming and simple to interfacing with other peripherals PIC become successful microcontroller.

PIC Microcontroller

We know that microcontroller is an integrated chip which is consists of RAM, ROM, CPU, TIMER and COUNTERS. The PIC is a microcontroller which as well consists of RAM, ROM, CPU, timer, counter, ADC (analog to digital converters), DAC (digital to analog converter). PIC Microcontroller also support the protocols like CAN, SPI, UART for an interfacing with additional peripherals. PIC mostly used to modify Harvard architecture and also supports RISC (Reduced Instruction Set Computer) by the above requirement RISC and Harvard we can simply that PIC is faster than the 8051 based controllers which is prepared up of Von-Newman architecture.

3, AVR Microcontroller

AVR microcontroller was developed in the year of 1996 by Atmel Corporation. The structural design of AVR was developed by the Alf-Egil Bogen and Vegard Wollan. AVR derives its name from its developers and stands for Alf-Egil Bogen Vegard Wollan RISC microcontroller, also known as Advanced Virtual RISC. The AT90S8515 was the initial microcontroller which was based on the AVR architecture, though the first microcontroller to hit the commercial market was AT90S1200 in the year 1997.



AVR Microocntroller

AVR Microcontrollers are Available in three Categories

a) **TinyAVR:-** Less memory, small size, appropriate just for simpler applications
b) **MegaAVR:-** These are the mainly popular ones having a good quantity of memory (up to 256 KB), higher number of inbuilt peripherals and appropriate for modest to complex applications.
c) **XmegaAVR:-** Used in commercial for complex applications, which need large program memory and high speed

4, ARM Processor

An ARM processor is also one of a family of CPUs based on the RISC (reduced instruction set computer) architecture developed by Advanced RISC Machines (ARM).

ARM Microcontroller

An ARM makes at 32-bit and 64-bit RISC multi-core processors. RISC processors are designed to perform a smaller number of types of computer instructions so that they can operate at a higher speed, performing extra millions of instructions per second (MIPS). By stripping out unnecessary instructions and optimizing pathways, RISC processors give outstanding performance at a part of the power demand of CISC (complex instruction set computing) procedure.

ARM processors are widely used in customer electronic devices such as smart phones, tablets, multimedia players and other mobile devices, such as wearables. Because of their reduced to instruction set, they need fewer transistors, which enable a smaller die size of the integrated circuitry (IC). The ARM processors, smaller size reduced difficulty and lower power expenditure makes them suitable for increasingly miniaturized devices.

Main Difference between AVR, ARM, 8051 and PIC Microcontrollers

	8051	PIC	AVR	ARM
Bus width	8-bit for standard core	8/16/32-bit	8/32-bit	32-bit mostly also available in 64-bit
Communication Protocols	UART, USART,SPI,I2C	PIC, UART, USART, LIN, CAN, Ethernet, SPI, I2S	UART, USART, SPI, I2C, (special purpose AVR support CAN, USB, Ethernet)	UART, USART, LIN, I2C, SPI, CAN, USB, Ethernet, I2S, DSP, SAI (serial audio interface), IrDA
Speed	12 Clock/instruction cycle	4 Clock/instruction cycle	1 clock/ instruction cycle	1 clock/ instruction cycle
Memory	ROM, SRAM, FLASH	SRAM, FLASH	Flash, SRAM, EEPROM	Flash, SDRAM, EEPROM
ISA	CLSC	Some feature of RISC	RISC	RISC
Memory Architecture	Harvard architecture	Von Neumann architecture	Modified	Modified Harvard architecture
Power Consumption	Average	Low	Low	Low
Families	8051 variants	PIC16,PIC17, PIC18, PIC24, PIC32	Tiny, Atmega, Xmega, special purpose AVR	ARMv4,5,6,7 and series
Community	Vast	Very Good	Very Good	Vast
Manufacturer	NXP, Atmel, Silicon Labs, Dallas, Cyprus, Infineon, etc.	Microchip Average	Atmel	Apple, Nvidia, Qualcomm, Samsung Electronics, and TI etc.
Cost (as compared to features provide)	Very Low	Average	Average	Low
Other Feature	Known for its Standard	Cheap	Cheap, effective	High speed operation Vast

2, What is the differenco between Reduced Instruction Set Architecture (RISC) and Complex Instruction Set Architecture (CISC)

What is RISC?

In the **RISC architecture**, the instruction set of the computer system is simplified to reduce the execution time. RISC architecture has a small set of instructions that generally includes register-to-register operations.

The RISC architecture uses comparatively a simple instruction format that is easy to decode. The instruction length can be fixed and aligned to word boundaries. RISC processors can execute only one instruction per clock cycle.

The following are some important **characteristics** of a RISC Processor −

- A RISC processor has a few instructions.
- RISC processor has a few addressing modes.
- In the RISC processor, all operations are performed within the registers of the CPU.
- RISC processor can be of fixed-length.
- RISC can be hardwired rather than micro-programmed control.
- RISC is used for single-cycle instruction execution.
- RISC processor has easily decodable instruction format.

RISC architectures are characterized by a small, simple instruction set and a highly efficient execution pipeline. This allows RISC processors to execute instructions quickly, but it also means that they can only perform a limited number of tasks.

What is CISC?

The **CISC architecture** comprises a complex instruction set. A CISC processor has a variable-length instruction format. In this processor architecture, the instructions that require register operands can take only two bytes.

In a CISC processor architecture, the instructions which require two memory addresses can take five bytes to comprise the complete instruction code. Therefore, in a CISC processor, the execution of instructions may take a varying number of clock cycles. The CISC processor also provides direct manipulation of operands that are stored in the memory.

The primary objective of the CISC processor architecture is to support a single machine instruction for each statement that is written in a high-level programming language.

The following are the important **characteristics** of a CISC processor architecture −

- CISC can have variable-length instruction formats.
- It supports a set of a large number of instructions, typically from 100 to 250 instructions.
- It has a large variety of addressing modes, typically from 5 to 20 different modes.
- CISC has some instructions which perform specialized tasks and are used infrequently.

CISC architectures have a large, complex instruction set and a less efficient execution pipeline. This allows CISC processors to perform a wider range of tasks, but they are not as fast as RISC processors when executing instructions.

Difference between RISC and CISC

The following table highlights all the important differences between RISC and CISC architectures

S.No.	RISC	CISC
1.	It stands for Reduced Instruction Set Computer.	It stands for Complex Instruction Set Computer.
2.	It is a microprocessor architecture that uses small instruction set of uniform length.	This offers hundreds of instructions of different sizes to the users.
3.	These simple instructions are executed in one clock cycle.	This architecture has a set of special purpose circuits which help execute the instructions at a high speed.
4.	These chips are relatively simple to design.	These chips are complex to design.
5.	They are inexpensive.	They are relatively expensive.
6.	Examples of RISC chips include SPARC, POWER PC.	Examples of CISC include Intel architecture, AMD.
7.	It has less number of instructions.	It has more number of instructions.
8.	It has fixed-length encodings for instructions.	It has variable-length encodings of instructions.
9.	Simple addressing formats are supported.	The instructions interact with memory using complex addressing modes.
10.	It doesn't support arrays.	It has a large number of instructions. It supports arrays.
11.	It doesn't use condition codes.	Condition codes are used.
12.	Registers are used for procedure arguments and return addresses.	The stack is used for procedure arguments and return addresses.

3.READ ABOUT analog write and pulse width modulation and prepare some note with programming code example ?

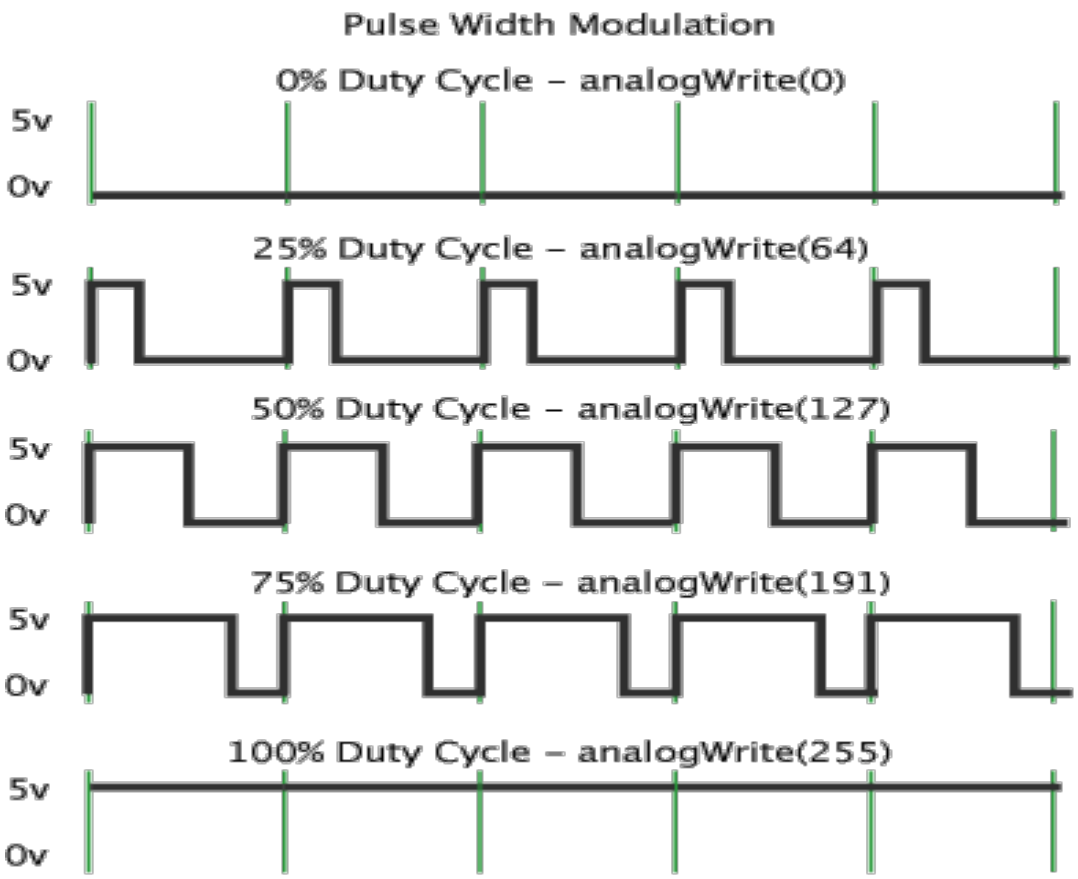
Analog Write and Pulse Width Modulation (PWM) in Arduino

Analog Write and Pulse Width Modulation (PWM) are crucial concepts in microcontroller programming, particularly with Arduino. These techniques allow digital devices to simulate an analog output, essential for controlling devices like LEDs, motors, and other actuators.

Pulse Width Modulation (PWM) is a technique used to create a pseudo-analog output by varying the width of digital pulses. PWM can control the amount of power delivered to a load without generating significant heat, making it efficient for various applications.

PWM operates by switching the output on and off at a high frequency. The "duty cycle" of a PWM signal determines the proportion of time the signal is high (on) versus low (off). For example, a 50% duty cycle means the signal is on for half the time and off for the other half.

Pulse Width Modulation, or PWM, is a technique for getting analog results with digital means. Digital control is used to create a square wave, a signal switched between on and off. This on-off pattern can simulate voltages in between the full Vcc of the board (e.g., 5 V on UNO, 3.3 V on a MKR board) and off (0 Volts) by changing the portion of the time the signal spends on versus the time that the signal spends off. The duration of "on time" is called the pulse width. To get varying analog values, you change, or modulate, that pulse width. If you repeat this on-off pattern fast enough with an LED for example, the result is as if the signal is a steady voltage between 0 and Vcc controlling the brightness of the LED.



Applications of Pulse Width Modulation (PWM) in Arduino:

Pulse Width Modulation (PWM) is a powerful technique used in various applications involving Arduino to control analog devices with digital signals. By adjusting the duty cycle of a PWM signal, you can simulate a range of analog outputs from a digital pin. Here are some common applications of PWM in Arduino projects:

1. **LED Brightness Control**

Dimming LEDs: PWM can control the brightness of LEDs by adjusting the duty cycle. A higher duty cycle results in a brighter LED, while a lower duty cycle makes it dimmer.
Fading Effects: By gradually increasing and decreasing the duty cycle, you can create smooth fading effects for LEDs, useful in decorative lighting and indicators.

```
// LED Brightness Control
int ledPin = 9; // LED connected to digital pin 9

void setup() {
  pinMode(ledPin, OUTPUT); // Set pin as an output
}

void loop() {
  // Fade in from 0 to 255
  for (int brightness = 0; brightness <= 255; brightness++) {
    analogWrite(ledPin, brightness); // Set the brightness
    delay(10); // Wait for 10 milliseconds
  }

  // Fade out from 255 to 0
  for (int brightness = 255; brightness >= 0; brightness--) {
    analogWrite(ledPin, brightness); // Set the brightness
    delay(10); // Wait for 10 milliseconds
  }
}
```

2. **Motor Speed Control**

DC Motors: PWM is commonly used to control the speed of DC motors. By varying the duty cycle, you can adjust the power delivered to the motor, thus controlling its speed.
Servo Motors: Although servos are typically controlled via the Servo library, understanding PWM is crucial for fine-tuning their position and movement.

```
int motorPin = 3; // Motor connected to digital pin 3

void setup() {
  pinMode(motorPin, OUTPUT); // Set pin as an output
}

void loop() {
  // Motor speeds up
  for (int speed = 0; speed <= 255; speed++) {
    analogWrite(motorPin, speed); // Set the motor speed
    delay(20); // Wait for 20 milliseconds
  }

  // Motor slows down
  for (int speed = 255; speed >= 0; speed--) {
    analogWrite(motorPin, speed); // Set the motor speed
    delay(20); // Wait for 20 milliseconds
  }
}
```

3. Communication

IR Transmitters: PWM is used in Infrared (IR) communication to encode data for remote controls and other IR-based communication devices.

4. Voltage Regulation
DC-DC Converters: PWM can be used in DC-DC converters to regulate the output voltage. By adjusting the duty cycle, the output voltage can be controlled efficiently.

5. Light Control
RGB LEDs: By using PWM on the red, green, and blue channels of an RGB LED, you can create a wide range of colors by mixing different brightness levels of the primary colors.

Reference:

1, https://www.elprocus.com/difference-between-avr-arm-8051-and-pic-microcontroller/
2, https://www.tutorialspoint.com/difference-between-risc-and-cisc
3, https://www.geeksforgeeks.org/computer-organization-risc-and-cisc/
4, https://cs.stanford.edu/people/eroberts/courses/soco/projects/risc/risccisc/