(B00828528) **Vineeth Mathangi**.

[*Mathangi-v@ulster.ac.uk*](mailto:Mathangi-v@ulster.ac.uk)*.*

**MSc Internet Of Things F.T**

Jordanstown.



EEE835-Embedded System & Sensors

Course Work -1

**TABLE OF CONTENT :**

**1.INTRODUCTION:……………………………………………………………………………………**

**2.THEORY:……………………………………………………………………………………………….**

**3. OBJECTIVE OF THE LAB:………………………………………………………………………..**

**4.1. WEEK -5 LABORATORY :…………………………………………………………………….**

**4.2. TEST PLAN :……………………………………………………………………………………….**

**4.2. EQUIPMENT REQUIRED:…………………………………………………………………….**

**4.4. EVIDENCE OF CIRCUITS:……………………………………………………………………..**

**4.5. RESULTS:……………………………………………………………………………………………**

**5.1. WEEK -6 LABORATORY:………………………………………………………………….....**

**5.2. TEST PLAN :……………………………………………………………………………………….**

**5.2. EQUIPMENT REQUIRED:…………………………………………………………………….**

**5.4. EVIDENCE OF CIRCUITS:……………………………………………………………………..**

**5.5. RESULTS :…………………………………………………………………………………………..**

**6.1. WEEK -7 LABORATORY:……………………………………………………………………..**

**6.2. TEST PLAN :………………………………………………………………………………………..**

**6.2. EQUIPMENT REQUIRED:……………………………………………………………………..**

**6.4. EVIDENCE OF CIRCUITS:………………………………………………………………………**

**6.5. RESULTS:……………………………………………………………………………………………**

**7.SOURCE OF ERRORS:………………………………………………………………………………**

**1.INTRODUCTION**

**ARDUINO:**

Arduino is an open-source electronics platform that uses simple hardware and software to make it easy to use. Arduino boards can take inputs like a light on a sensor, a finger on a button, or a tweet and turn them into outputs like turning on an LED, triggering a motor, or publishing anything online. By providing a set of instructions to the board's microcontroller, you may tell it what to do. The Arduino programming language (based on Wiring) and the Arduino Software (IDE) (based on Processing) are used to accomplish this. Thousands of projects and applications have made use of Arduino.

Beginners will find the Arduino software simple to use, but advanced users will find it adaptable. It's compatible with Mac, Windows, and Linux. It is used by teachers and students to create low-cost scientific equipment, to demonstrate chemistry and physics principles, and to begin learning programming and robotics. Designers and architects create interactive prototypes, while musicians and artists utilise it to create installations and try out new instruments. Makers, of course, rely on it to construct many of the projects on display at Maker Faire. as an example. Arduino is a valuable tool for learning new skills. Anyone - youngsters, amateurs, artists, and programmers - may get started tinkering by following the step-by-step instructions in a kit or sharing ideas with other Arduino members online. Timeline

Description automatically generated

Figure 1 Arduino Uno (AVR ATMEGA 328P) Architecture

The Arduino Uno pinout. Arduino Uno is based on the [﻿](https://www.circuito.io/blog/atmega328p-bootloader/)ATmega328 by Atmel. The Arduino Uno pinout consists of 14 digital pins, 6 analog inputs, a power jack, USB connection and ICSP header. The versatility of the pinout provides many different options such as driving motors, LEDs, reading sensors and more.

**Serial peripheral interface (SPI)** The serial peripheral interface (SPI) is one of the most extensively used interfaces between microcontrollers and peripheral ICs such sensors, ADCs, DACs, shift registers, SRAM, and others. This article begins with a quick overview of the SPI interface, then moves on to an overview of Analog Devices' SPI enabled switches and muxes, and how they might assist minimise the number of digital GPIOs on system boards. SPI is a full-duplex, synchronous master-slave interface. On the rising or falling clock edge, data from the master or slave is synced. Data can be transmitted by both the master and the slave at the same time. Three-wire or four-wire SPI interfaces are available. The focus of this essay is on the popular culture.

1.MISO (Master In Slave Out) : - A line for sending data to the Master device

2.MOSI (Master Out Slave In) : The Master line for sending data to peripheral devices

3.SCK (Serial Clock) : A clock signal generated by the Master device to synchronize data transmission

4. CS (Chip Select) : the peripheral that it should wake up and receive / send data and is also used when multiple peripherals are present to select the one you'd like to talk to.

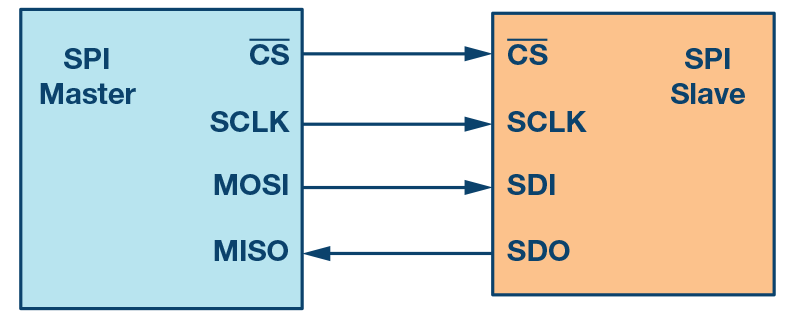


Figure 2 Serial peripheral interface

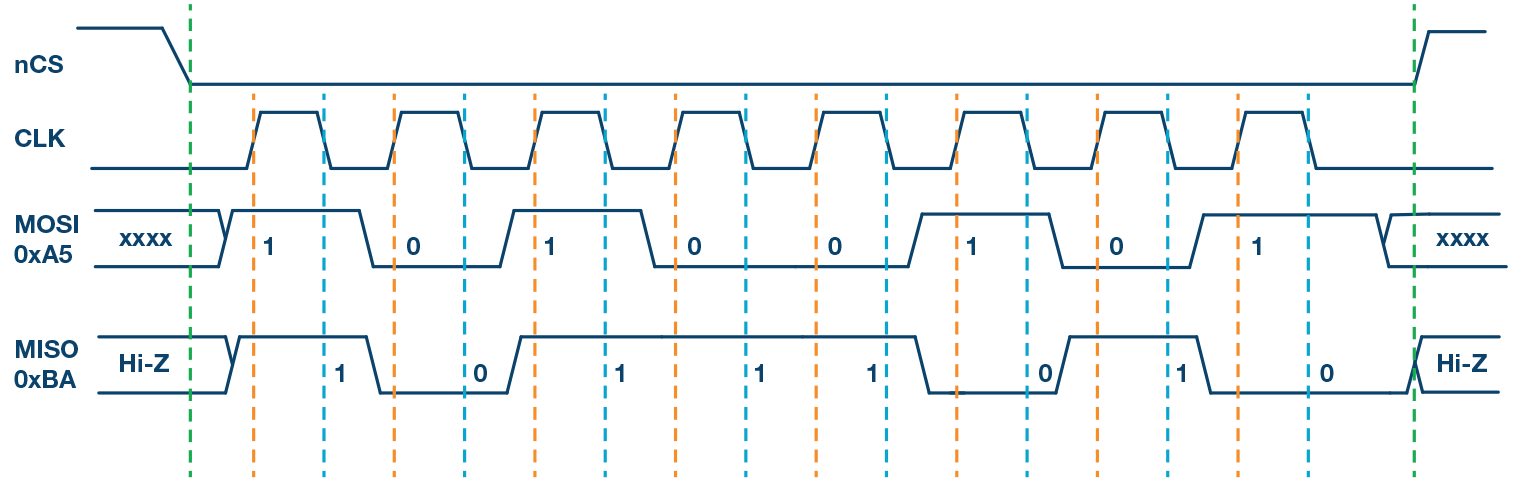


Figure 3 MOSI and MISO Line

 Above figure illustrates the basic idea of SPI’s, data is shown on the MOSI and MISO line. The start and end of transmission is indicated by the dotted green line, the sampling edge is indicated in orange, and the shifting edge is indicated in blue.

**WEEK-5 LABORATORY:**

**INTRODUCTION:**

Assume a person is at home, waiting for his mobile phone to arrive at his mailbox, which he ordered a few days earlier. He looks forward to checking the mailbox every 10 minutes to see if his package has arrived. This is known as polling, but what if he had instructed the mailman to ring the doorbell when he arrived? As a result, he is free to do anything he wants, and when the package arrives at the mailbox, he is told, and the same may be used as an example for interrupts in Arduino.Likewise, exactly how an interrupt causes of processor to act the Arduino program is running and performing some function, however when Interrupt occurs then the main programs stops and run the interrupt code and after finishing this processor it goes back to the main program again and resumes its execution from the point it was stopped.

Interrupts are heavily reliant on the registers. The microchip AVR ATmega328P is the data sheet for the microcontroller switch on the Arduino Uno. The data sheet is 294 pages long and contains all of the information regarding the 328P and its siblings. The CPU core aids in the configuration of the main processor. The MGM arithmetic logic unit is in charge of executing operations such as addition, subtraction, and bitwise logical operations, as well as storing data in 32 general purpose registers, each of which holds 8 bits or one byte. It also has a different list of registered names for ESP.

**THEORY:**

**INTERRUPTS:**

Interrupts cause Arduino's present operation to pause so that other tasks can be completed. In a circuit, the primary programme is running and performing some function. When an interrupt occurs, however, the main programme comes to a halt while another procedure is executed. When these routines are completed, the processor returns to the main procedure.

Interrupts are very useful in Arduino programs as it helps in solving timing problems. A good application of an interrupt is reading a rotary encoder or observing a user input. Generally, an ISR should be as short and fast as possible. If your sketch uses multiple ISRs, only one can run at a time. Other interrupts will be executed after the current one finishes in an order that depends on the priority they have. [1]

**Types of Interrupts:**

There are two types of interrupts:

**Hardware Interrupt:** It happens when an external event occurs like an external interrupt pin changes its state from LOW to HIGH or HIGH to LOW.

**Software Interrupt:** It happens according to the instruction from the software. For example Timer interrupts are software interrupt.

**Interrupts in Arduino:**

Now we will see how to use interrupts in Arduino Board. It has two types of interrupts:

* External Interrupt
* Pin Change Interrupt

**External Interrupt:**

These interrupts are interpreted by hardware and are very fast. These interrupts can be set to trigger on the event of RISING or FALLING or LOW levels.

|  |  |
| --- | --- |
| Arduino Board | External Interrupt pins: |
| UNO , NANO | 2,3 |
| Mega | 2,3,18,19,20,21 |

**Pin Change Interrupts:**

Arduinos can have more interrupt pins enabled by using pin change interrupts. In ATmega168/328 based Arduino boards any pins or all the 20 signal pins can be used as interrupt pins. They can also be triggered using RISING or FALLING edges.

**Using Interrupts in Arduino:**

In order to use interrupts in Arduino the following concepts are need to be understood.

Interrupt Service Routine (ISR)

Interrupt Service Routine or an Interrupt handler is an event that has small set of instructions in it. When an external interrupt occurs, the processor first executes these code that is present in ISR and returns to state where it left the normal execution. [1]

**ISR has following syntax in Arduino:**

**attachInterrupt(digitalPinToInterrupt(pin), ISR, mode);**

digitalPinToInterrupt(pin): In Arduino Uno, NANO the pins used for interrupt are 2,3 & in mega 2,3,18,19,20,21. Specify the input pin that is used for external interrupt here.

ISR: It is a function that is called when an external interrupt is done.

Mode: Type of transition to trigger on, e.g. falling, rising, etc.

* RISING:  To trigger an interrupt when the pin transits from LOW to HIGH.
* FALLING: To trigger an interrupt when the pin transits from HIGH to LOW.
* CHANGE: To trigger an interrupt when the pin transits from LOW to HIGH or HIGH to LOW (i.e., when the pin state changes ).

Some Conditions while using Interrupt

* Interrupt Service Routine function (ISR) must be as short as possible.
* Delay () function doesn’t work inside ISR and should be avoided.

**EQUIPMENT REQUIRED:**

1. INA126P operational amplifier
2. LM324N
3. MCP3208
4. Arduino Uno
5. Breadboard
6. Jumper wires
7. 0.47 µf capacitors 2
8. Gain resistors (16k-1,100K-2)
9. Power supply
10. Signal generator
11. Power supply (12v)
12. Potentiometer

**TEST PLAN 1:**

Using the existing MCP3208\_using\_library.ino code, modify it so that the serial port stops data acquisition upon seeing a change on a digital input pin (say pin 2 which allows for Interrupts)

**EVIDENCE IN LAB:**

**A picture containing text, computer, electronics, computer

Description automatically generated** A picture containing text, electronics, display

Description automatically generated

Figure 4 Circuit Evidence-1.2 for week-5 Lab Figure 5 Circuit Evidence-1.1 for week-5 Lab

Note: The power supply is taken from the Arduino 5v

**Code:** [**Attached in the appendix**](#Testplan1code) **(Appendix)**

**RESULT EVIDENCE :**

**Graphical user interface, text, application

Description automatically generated**

Figure 6 Result evidence-1 for week-5 Lab

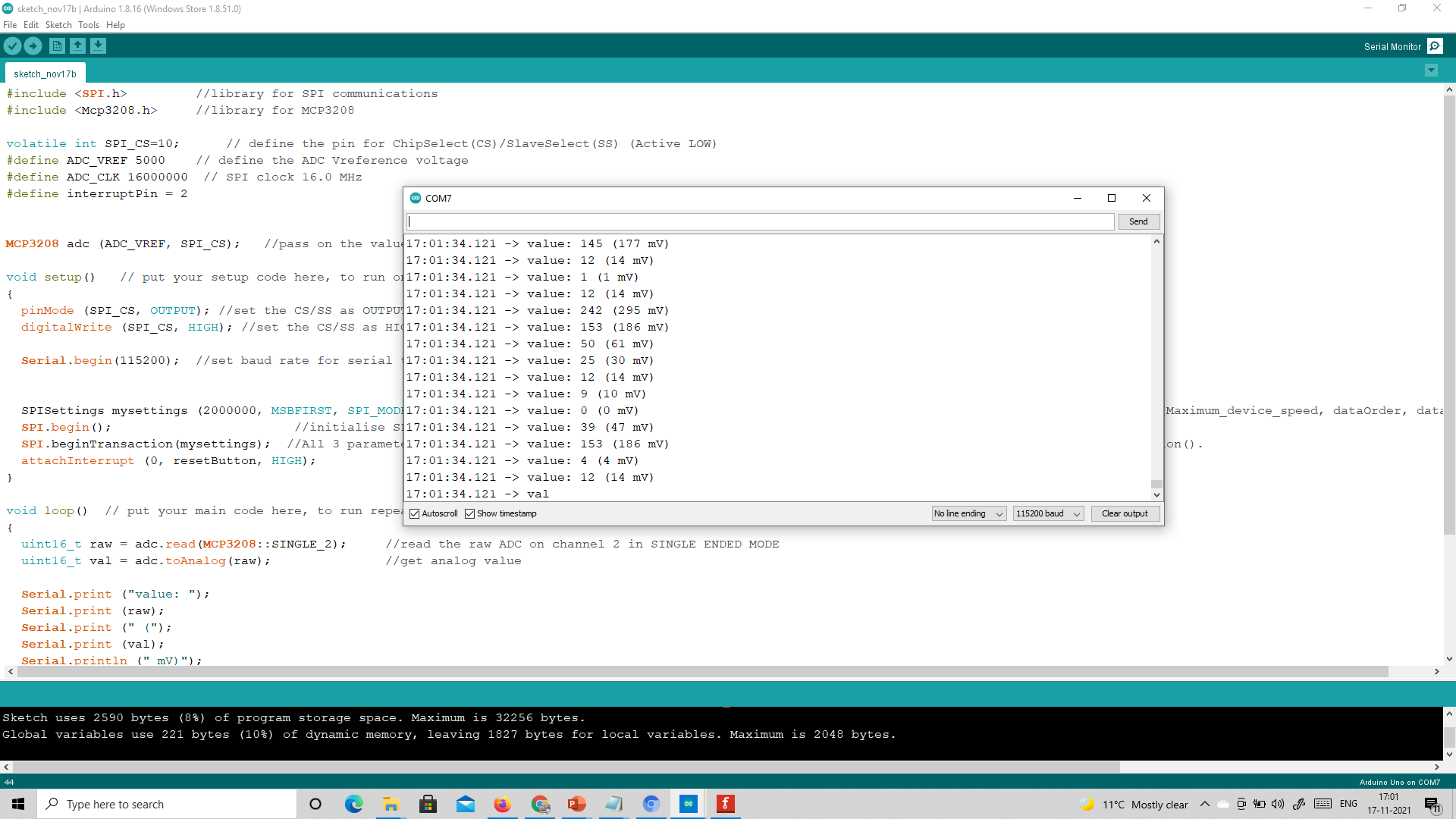


Figure 7 Result evidence-2 for week-5 Lab

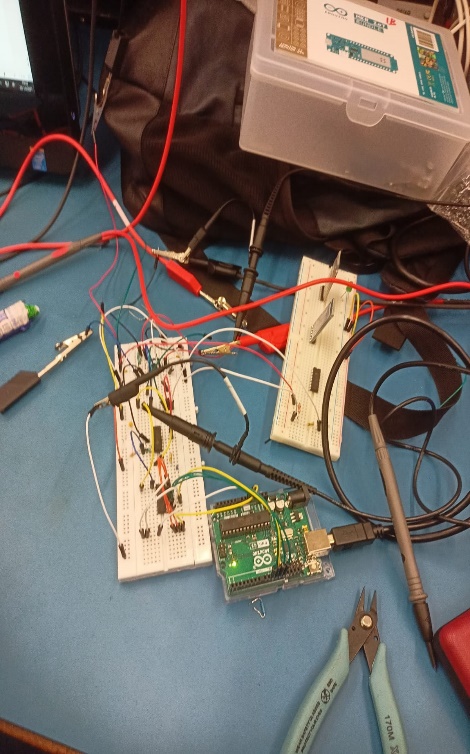
**Discussion on Result:**  The entire setup is taken from the previous week 4, where the Arduino was used to read the filtered signal. In addition to this configuration, we employed external hardware interrupts to reduce timing and power usage.

We set pin 2 to interrupt according to the Atmega328p datasheet References. Pins 2 and 3 are, however, the most important. To transmit data between an ISR and the main programme, global variables are typically employed. Declare variables shared between an ISR and the main programme as volatile[1] to ensure that they are updated correctly. The button is connected to Arduino pin 2 (PD2) for access to the microcontroller's deep memory. Finally, Arduino records the ECG's serial data, but if an interrupt occurs in pin 2, data acquisition is halted, and the execution jumps back to the interrupt function.

**TEST PLAN 2:**

Using the function generator, generate a sine wave of 80Hz and digitize it at 200 samples per second for 30 seconds i.e. take 6000 samples. These samples should only be taken when the Arduino sees a change on its digital input pin (say pin 2 which allows for interrupts). Hint: Using the existing MCP3208\_using\_library.ino code and use TIMER1.

**EVIDENCE IN LAB:**

 Graphical user interface

Description automatically generated

Figure 8 Circuit Evidence-2.1 for week-5 Lab Figure 9 Circuit Evidence-2.2 for week-5 Lab

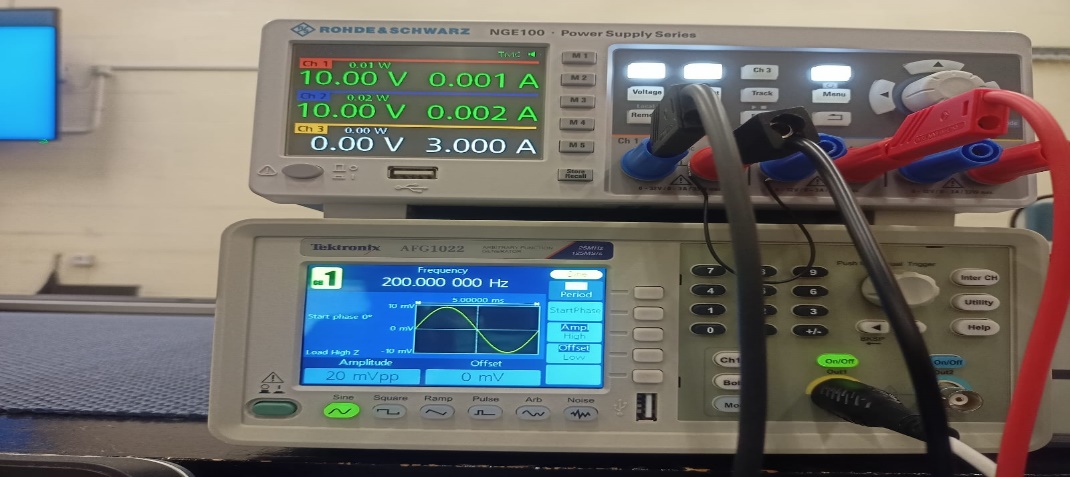


Figure 10 Circuit Evidence-2.3 for week-5 Lab

**Code:** [**Testplan2code**](#Testplan2code) **(Appendix)**

**RESULTS :**

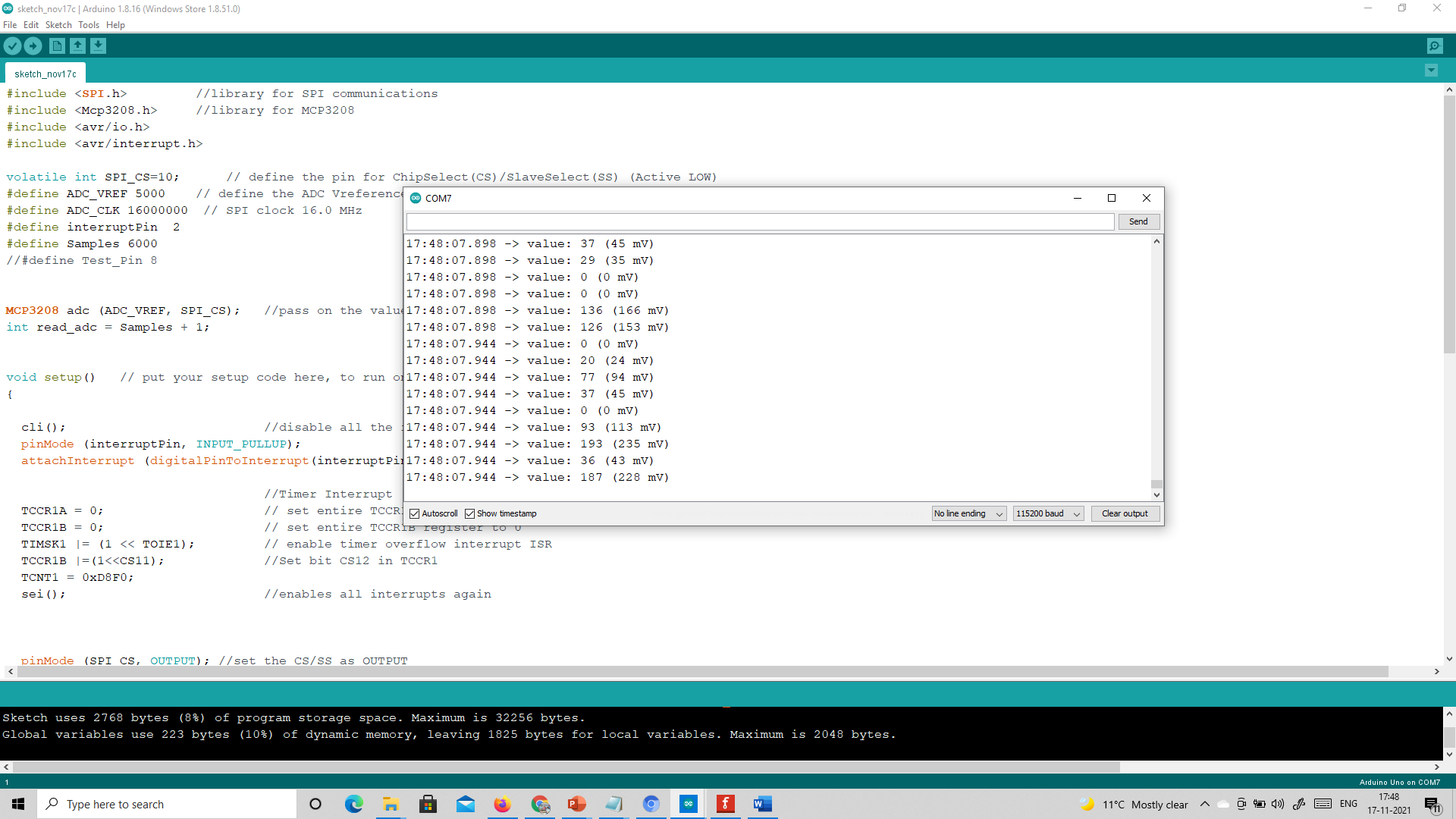
****

Figure 11 Result evidence-3 for week-5 Lab



Figure 12 80hz MATLAB Result evidence-1 for week-5 Lab

Above figure 12 shows that result evidence of samples is up to 6000 with 80Hz

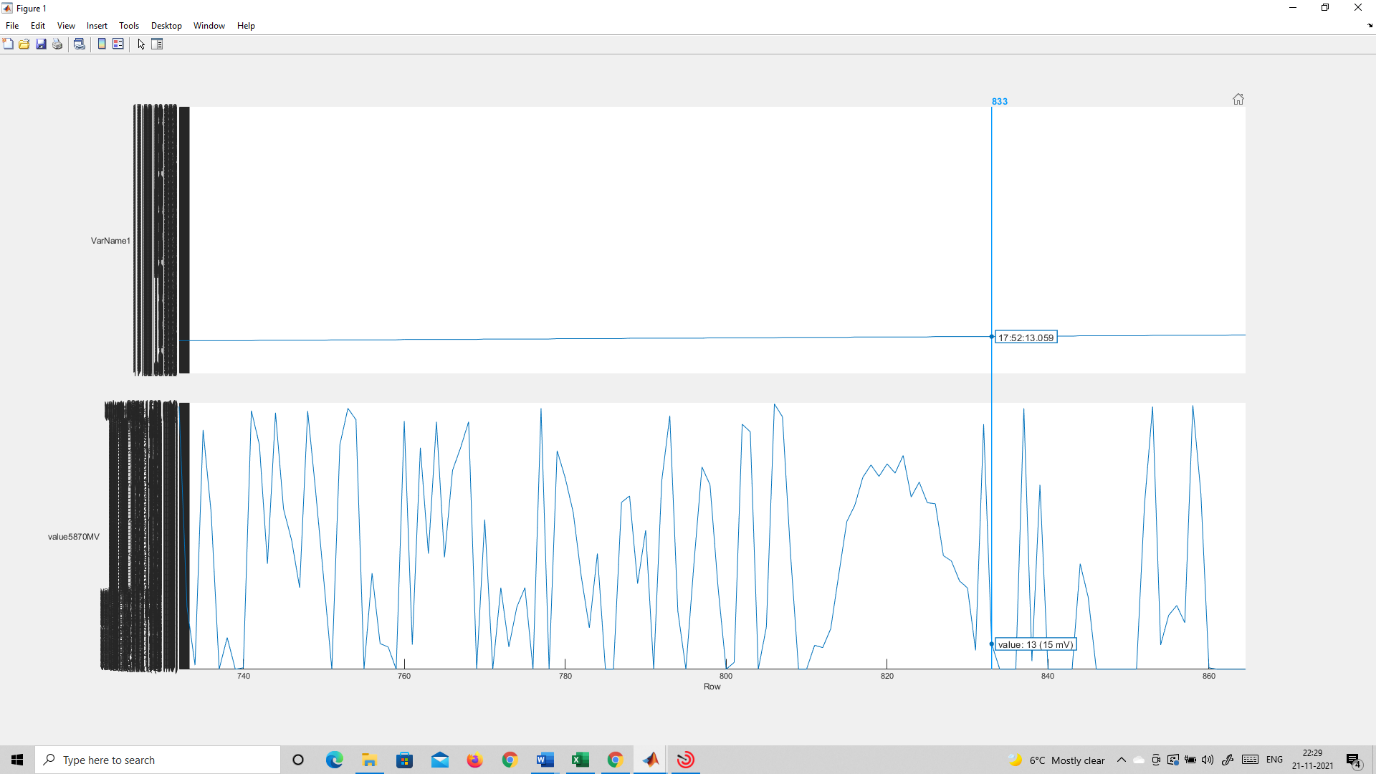


Figure 13 80hz Matlab Result evidence-2 for week-5 Lab

Another example,Above figure 13 shows that result evidence of samples is upto 860 with 80Hz

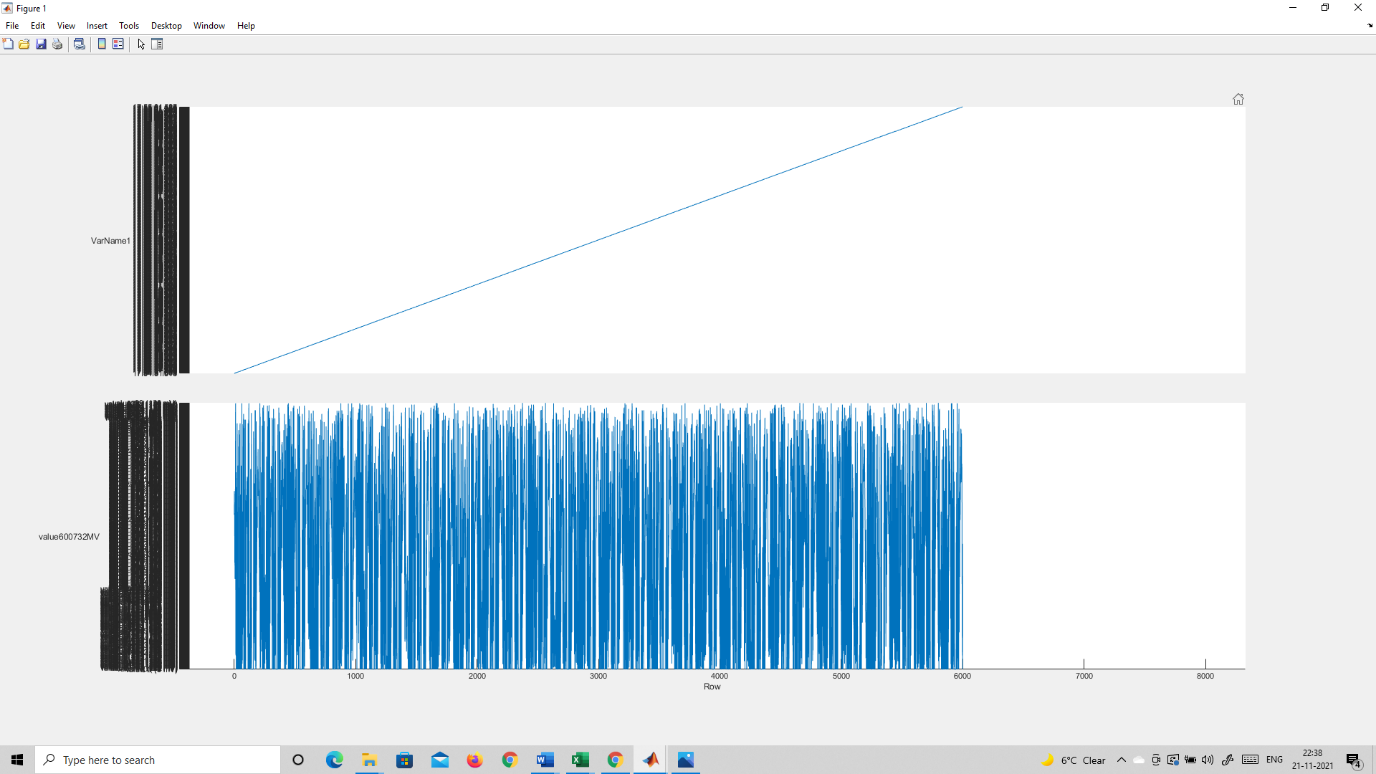


Figure 14 200hz Matlab Result evidence-1 for week-5 Lab

Above figure 12 shows that result evidence of samples is upto 6000 with 200Hz

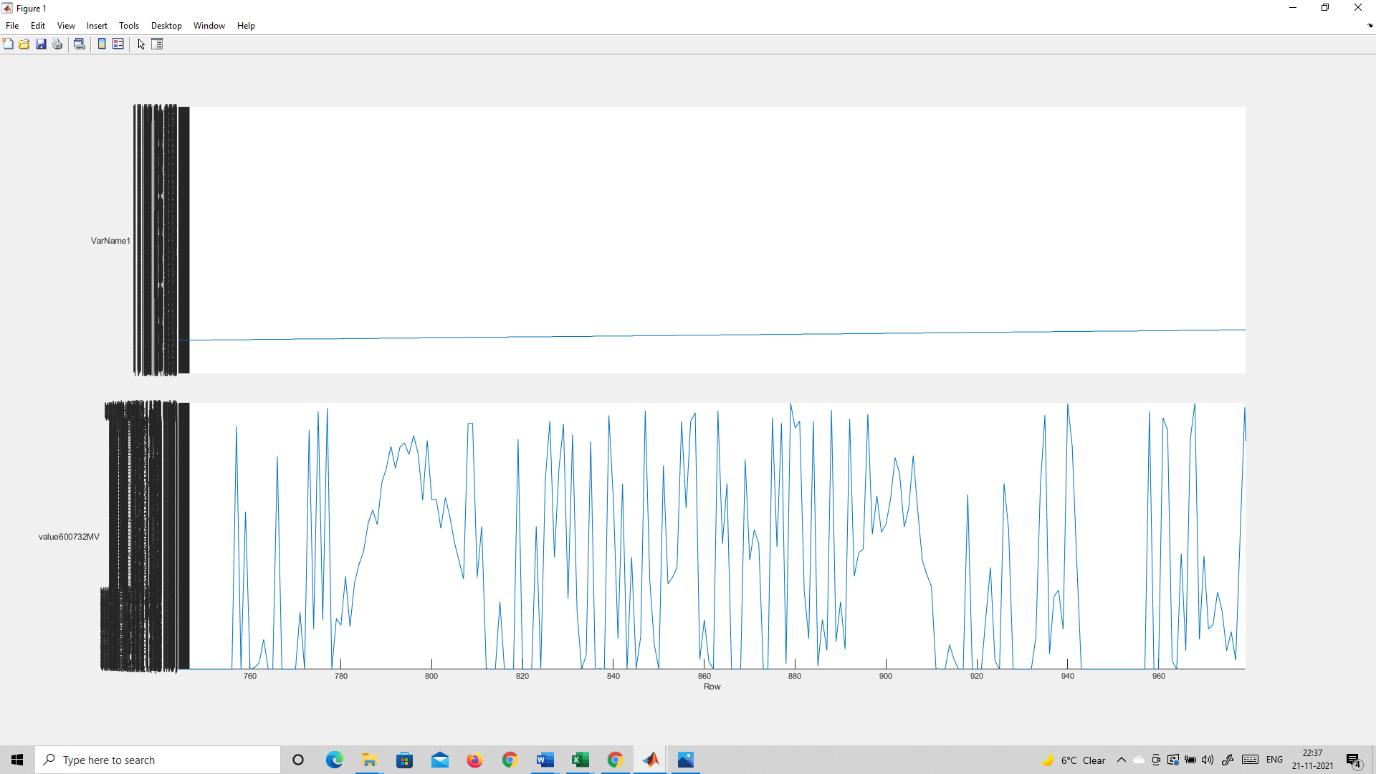


Figure 15 200hz Matlab Result evidence-2 for week-5 Lab

Another example,Above figure 13 shows that result evidence of samples is upto 860 with 200Hz

**Discussion on Result:** : We used our function generator to generate an 80Hz sine wave, but Arduino isn't capturing any data in the meantime. So, we're using an interrupt from pin 2 to cause the Arduino to take 6000 samples of data. Giving 5v electricity to Arduino's digital pin2 creates external interrupts. The data is then put into MATLAB for visualisation in the report. Figure 12: Week 5 Lab 80hz Matlab Result Evidence 1 Figure 13: Week 5 Lab 80Hz Matlab Result evidence-2

The identical operation was carried out with a frequency of 200hz, and the results were visualised using a MATLAB figure. 200hz (Figure 14) Week 5 Lab Matlab Result Evidence-1 Figure 15: Week 5 Lab 200hz Matlab Result Evidence-2

**WEEK-6 LABORATORY:**

**INTRODUCTION:**

A Bluetooth module for use with any microcontroller, the HC-05 4 Pin Wireless Serial Bluetooth Module is a Bluetooth module for use with any microcontroller. It makes it simple to send and receive data wirelessly by employing the UART protocol. For transmitter and receiver, the HC-04 module can be used as both a master and a slave device. This means it can pair with most Bluetooth-enabled phones and laptops, as well as other slave devices like keyboards and other HC-06 modules. A master module would be required to link with other slave devices. [2]

It is used for many applications like wireless headset, game controllers, wireless mouse, wireless keyboard and many more consumer applications. And It has range up to <100m which depends upon transmitter and receiver, atmosphere, geographic & urban conditions. It is IEEE 802.15.1 standardized protocol, through which one can build wireless Personal Area Network ([PAN](https://en.wikipedia.org/wiki/Personal_area_network)). It uses frequency-hopping spread spectrum ([FHSS](https://en.wikipedia.org/wiki/Frequency-hopping_spread_spectrum)) radio technology to send data over air. This module uses serial communication to communicate with devices. It communicates with microcontroller using serial port (USART). [2]

E.g. Send data from a Smartphone terminal to the HC-05 Bluetooth module and view it on a PC serial terminal, for example.

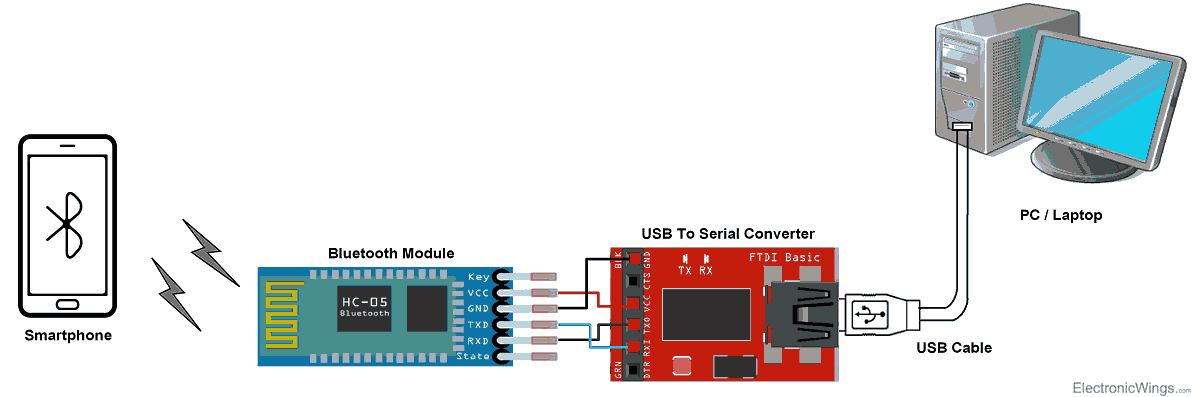
To communicate with the HC-05 Bluetooth module, the smartphone must have a Bluetooth terminal application installed on it for data transmission and reception. Bluetooth terminal programmes for Android and Windows can be found in the relevant app stores.

Figure Bluetooth Module Serial Interface

So, when we want to communicate through smartphone with HC-05 Bluetooth module, connect this HC-05 module to the PC via serial to USB converter. Before establishing communication between two Bluetooth devices, 1st we need to pair HC-05 module to smartphone for communication.

**THEORY:**

|  |  |  |
| --- | --- | --- |
| **SN** | **PIN** | **Description** |
| 1 | RXD | Receive Serial Data. Every serial data given to this pin will be broadcasted via Bluetooth |
| 2 | TXD | Transmits Serial Data.  Everything received via Bluetooth will be given out by this pin as serial data. |
| 3 | VCC | Powers the module. Connect to +5V Supply voltage |
| 4 | GND | Ground pin of module, connect to system ground. |
| 5 | LED | Indicates the status of Module Blink once in 2 sec: Module has entered Command Mode Repeated Blinking: Waiting for connection in Data Mode Blink twice in 1 sec: Connection successful in Data Mode |
| 6 | Button | Used to control the Key/Enable pin to toggle between Data and command Mode |

Table Table 1 HC-05 Bluetooth module pin description

##### **HC-05 Default Settings:**

1. Default Bluetooth Name: “HC-05”
2. Default Password: 1234 or 0000
3. Default Communication: Slave
4. Default Mode: Data Mode
5. Data Mode Baud Rate: 9600, 8, N, 1
6. Command Mode Baud Rate: 38400, 8, N, 1

**EQUIPMENT REQUIRED :**

1. Week-5 whole setup
2. Potentiometer
3. Hc-05 Bluetooth module

**TEST PLAN 1 :**

Modify the code from the previous week MCP3208\_using\_library code using sleep mode commands so that the Arduino Uno is sleeping all the time excepted with interrupted (at PIN 2) to capture readings for 30 seconds from the potentiometer via MCP3208 and then go off to sleep again.

**EVIDENCE :**

A hand holding a circuit board

Description automatically generated with low confidence

Figure Circuit Evidence-1 for week-6 Lab

**Code:** [**week6Testplan1code**](#week6Testplan1code) **(Appendix)**

**RESULTS :**

**Video evidence:** [**http://tiny.cc/k1tluz**](http://tiny.cc/k1tluz)

**Graphical user interface, text, application

Description automatically generated**

Figure Result evidence-2 for week-6 Lab

**DISCUSSION ON RESULTS:**

Except when the potentiometer causes an interrupt, the Arduino remains in sleep mode at all times. The interrupt is called from the sleep enable function in the programme when the value of the potentiometer rises. When an interrupt is called, the state changes from Low to High.

When the patient's heart rate rises from Low to High or High to Low, we may exploit this interrupt as an application to generate an alert or push notification in real time.

**TEST PLAN 2 :**

Create a program to send values via Bluetooth to a mobile phone for display via a mobile app and simultaneously transmit the data to the cloud (Things Speak).

**EVIDENCE:**

**Code :** [**week6Testplan2code**](#week6Testplan2code) **( Appendix )**

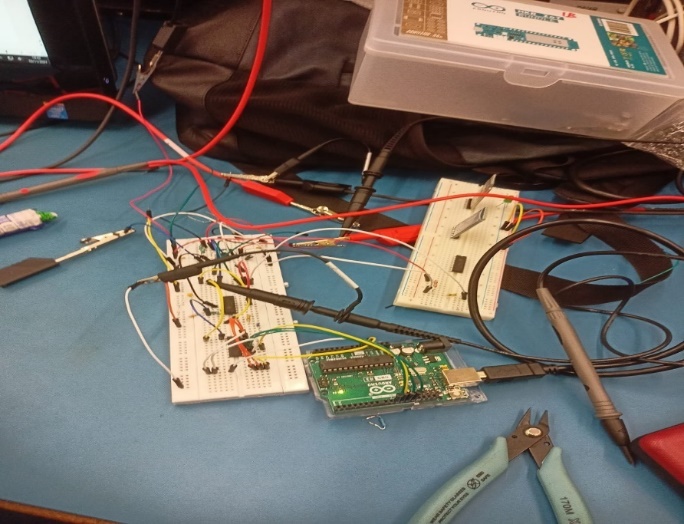
****

Figure Circuit Evidence-2 for week-6 Lab

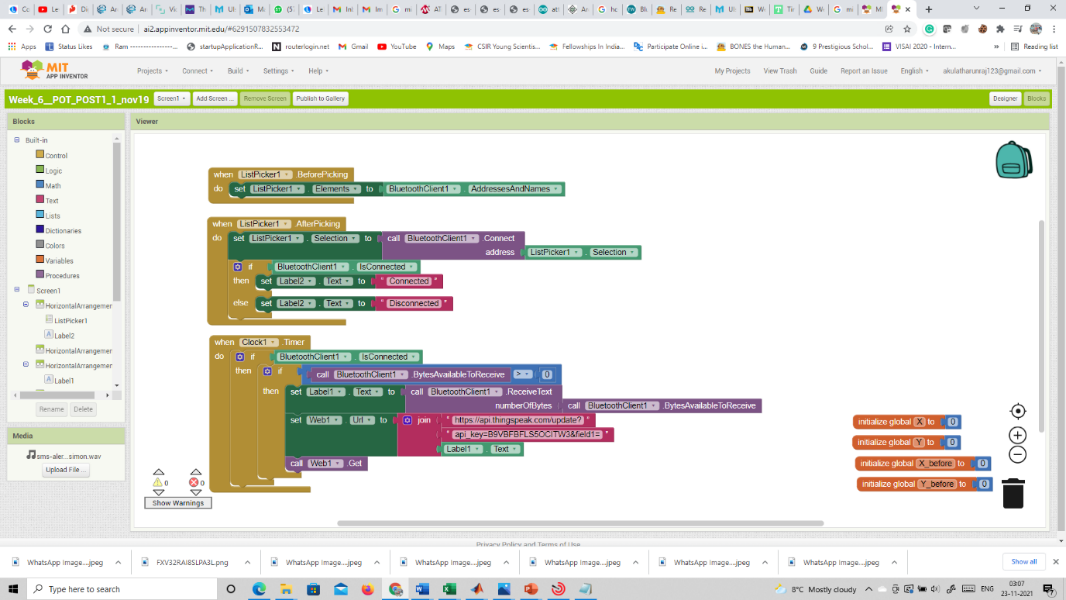
****

Figure MIT APP for reading Arduino Data [3]

**RESULTS :**

****

Figure MIT app Result evidence-3 for week-6 Lab

![Chart, line chart

Description automatically generated](data:image/jpeg;base64,/9j/4AAQSkZJRgABAQEAYABgAAD/4RDcRXhpZgAATU0AKgAAAAgABAE7AAIAAAAGAAAISodpAAQAAAABAAAIUJydAAEAAAAMAAAQyOocAAcAAAgMAAAAPgAAAAAc6gAAAAgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAADkxOTM5AAAFkAMAAgAAABQAABCekAQAAgAAABQAABCykpEAAgAAAAMxOAAAkpIAAgAAAAMxOAAA6hwABwAACAwAAAiSAAAAABzqAAAACAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAMjAyMToxMToyMyAxNTowODoyOQAyMDIxOjExOjIzIDE1OjA4OjI5AAAAOQAxADkAMwA5AAAA/+ELGGh0dHA6Ly9ucy5hZG9iZS5jb20veGFwLzEuMC8APD94cGFja2V0IGJlZ2luPSfvu78nIGlkPSdXNU0wTXBDZWhpSHpyZVN6TlRjemtjOWQnPz4NCjx4OnhtcG1ldGEgeG1sbnM6eD0iYWRvYmU6bnM6bWV0YS8iPjxyZGY6UkRGIHhtbG5zOnJkZj0iaHR0cDovL3d3dy53My5vcmcvMTk5OS8wMi8yMi1yZGYtc3ludGF4LW5zIyI+PHJkZjpEZXNjcmlwdGlvbiByZGY6YWJvdXQ9InV1aWQ6ZmFmNWJkZDUtYmEzZC0xMWRhLWFkMzEtZDMzZDc1MTgyZjFiIiB4bWxuczpkYz0iaHR0cDovL3B1cmwub3JnL2RjL2VsZW1lbnRzLzEuMS8iLz48cmRmOkRlc2NyaXB0aW9uIHJkZjphYm91dD0idXVpZDpmYWY1YmRkNS1iYTNkLTExZGEtYWQzMS1kMzNkNzUxODJmMWIiIHhtbG5zOnhtcD0iaHR0cDovL25zLmFkb2JlLmNvbS94YXAvMS4wLyI+PHhtcDpDcmVhdGVEYXRlPjIwMjEtMTEtMjNUMTU6MDg6MjkuMTc1PC94bXA6Q3JlYXRlRGF0ZT48L3JkZjpEZXNjcmlwdGlvbj48cmRmOkRlc2NyaXB0aW9uIHJkZjphYm91dD0idXVpZDpmYWY1YmRkNS1iYTNkLTExZGEtYWQzMS1kMzNkNzUxODJmMWIiIHhtbG5zOmRjPSJodHRwOi8vcHVybC5vcmcvZGMvZWxlbWVudHMvMS4xLyI+PGRjOmNyZWF0b3I+PHJkZjpTZXEgeG1sbnM6cmRmPSJodHRwOi8vd3d3LnczLm9yZy8xOTk5LzAyLzIyLXJkZi1zeW50YXgtbnMjIj48cmRmOmxpPjkxOTM5PC9yZGY6bGk+PC9yZGY6U2VxPg0KCQkJPC9kYzpjcmVhdG9yPjwvcmRmOkRlc2NyaXB0aW9uPjwvcmRmOlJERj48L3g6eG1wbWV0YT4NCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgPD94cGFja2V0IGVuZD0ndyc/Pv/bAEMABwUFBgUEBwYFBggHBwgKEQsKCQkKFQ8QDBEYFRoZGBUYFxseJyEbHSUdFxgiLiIlKCkrLCsaIC8zLyoyJyorKv/bAEMBBwgICgkKFAsLFCocGBwqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKv/AABEIANEBZgMBIgACEQEDEQH/xAAfAAABBQEBAQEBAQAAAAAAAAAAAQIDBAUGBwgJCgv/xAC1EAACAQMDAgQDBQUEBAAAAX0BAgMABBEFEiExQQYTUWEHInEUMoGRoQgjQrHBFVLR8CQzYnKCCQoWFxgZGiUmJygpKjQ1Njc4OTpDREVGR0hJSlNUVVZXWFlaY2RlZmdoaWpzdHV2d3h5eoOEhYaHiImKkpOUlZaXmJmaoqOkpaanqKmqsrO0tba3uLm6wsPExcbHyMnK0tPU1dbX2Nna4eLj5OXm5+jp6vHy8/T19vf4+fr/xAAfAQADAQEBAQEBAQEBAAAAAAAAAQIDBAUGBwgJCgv/xAC1EQACAQIEBAMEBwUEBAABAncAAQIDEQQFITEGEkFRB2FxEyIygQgUQpGhscEJIzNS8BVictEKFiQ04SXxFxgZGiYnKCkqNTY3ODk6Q0RFRkdISUpTVFVWV1hZWmNkZWZnaGlqc3R1dnd4eXqCg4SFhoeIiYqSk5SVlpeYmZqio6Slpqeoqaqys7S1tre4ubrCw8TFxsfIycrS09TV1tfY2dri4+Tl5ufo6ery8/T19vf4+fr/2gAMAwEAAhEDEQA/APoz7NF/dP8A30aPs0X90/8AfRqWigCL7NF/dP8A30aPs0X90/8AfRqWigCL7NF/dP8A30aPs0X90/8AfRqWigCL7NF/dP8A30aPs0X90/8AfRqWigCL7NF/dP8A30aPs0X90/8AfRqWigCL7NF/dP8A30aPs0X90/8AfRqWigCL7NF/dP8A30aPs0X90/8AfRqWigCL7NF/dP8A30aPs0X90/8AfRqWigCL7NF/dP8A30aPs0X90/8AfRqWigCL7NF/dP8A30aPs0X90/8AfRqWigCL7NF/dP8A30aPs0X90/8AfRqWigCL7NF/dP8A30aPs0X90/8AfRqWigCL7NF/dP8A30aPs0X90/8AfRqWigCL7NF/dP8A30aPs0X90/8AfRqWigCL7NF/dP8A30aPs0X90/8AfRqWigCL7NF/dP8A30aPs0X90/8AfRqWigCL7NF/dP8A30aPs0X90/8AfRqWigCL7NF/dP8A30aPs0X90/8AfRqWigCL7NF/dP8A30aPs0X90/8AfRqWigCL7NF/dP8A30aPs0X90/8AfRqWigCL7NF/dP8A30aPs0X90/8AfRqWigCL7NF/dP8A30aPs0X90/8AfRqWigCL7NF/dP8A30aKlooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACvNV8Q6mvw21RvsuqSTI1yFv1lTC4kYAhjJvAA46duK9KrDTwtbJ4VutCFxN5Fz5u6TjePMYsccY4zVRaW4mmzF1fx2ukXEtuGs/wDQbeGW4W4uNss+8Z2xL1JC8+5IFX/EXiz+wriBjNp/lO0WYJJ8XEiu4Usi+2c/gavTeH2F8bzTdRnsZniSKYoiOsoX7pIYHBGTyKpan4Kh1Ke/c6hcwJqHltMiKhy8eNrAkE4yB8vSrThdXItK2gSa5qt3ea5bWMNrEmmfKJpGYlyYg44HTBNXPCF3qF/4T0+71aSKW4ngRy8YxuBUHJHr9OKnstEhtJ9SlaWSZtRZWmDYAyECcY9QM0/RNK/sXSYdPW5kuYoBsiaRVBVAMBeAM49etRdWt6f8ErW9/X/gHB6Xez3WvpDa3uonUv7cuN4lnl8g2qOdygOdjYBAAUEg46V0+reKTpXiC0snl09kuLiOA24n/wBJG/o+30zjj0qRvB9obF4I7meOb7e9/Dcrt3wyM2Tt4xjBIweoPNQ3HgqKa5mkj1K5hSW8S+8tEQgTLj5iSMkcdDx+VXeLtf8Arb+v+HJtJXt1/wCD/wAD+lrn2vinUtNsfEN/rL288FlqDW8CKdhUnYFBJ4C/NnJ561seGvEn9t3F9bPJaTSWbJ++spfMikVhkYPqCCCPpTj4Vtn/ALUje4ma21KXz3h4Hly4X51bGQcqD6ZrQ06zu7RXF5qUt9nG0yRIhX/vkDNTeNvuHZ3+8u0UUVBYUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRWZ4j1C70rw1qF/ptoby7t4GkigAP7xgOBxXnPgP4wvfWbDx7EulPJLttrv7O8cEvqpY5AI+tK6RahKSuj1mio4J4bmFZraVJonGVeNgysPUEVJTICiiigAooooAKKx9X8W+H9ByNX1iztX/55vKN5/4COT+VcRZ/Hrwtc+Ip9PnW4trSPPl37IWSQj/ZA3AHtx+VK6RapykrpHp9FczY/EfwfqLhLbxDYhjwFmk8on6b8V0kcsc0ayQusiMMqynII+tMlprcdRRRQIKKKKACiiigAooooAKKKKACiiigAooqpqmqWWi6ZNqGqXCW1rAu6SV+iigC3RWPo3i3QvEGnLfaVqdvNAWK5LbCCOoIbBFX/wC0rH/n8t/+/q/40Ds0WaKrf2lY/wDP5b/9/V/xpG1SwRGZ763CqMkmVeB+dAWZaorD8PeM/D/iqW5j0HUo7x7Y4lVQQQPXkDI9xW5QDTTswooooEFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFQ3dnbX9s9ve28VxA4w0cqBlP4GpqKAONf4c2djK0/hPUr3w9KxyY7V98DH3ibK/lioze/EHQzi507T/EluP8AlpaS/Zpse6NlT+BrtqKVi+d9dTiR8UNOtfl17R9b0iUdVuLB3U/Ro92RWTpXx28ManqNzam31CFY/wDUuLdpDMM4+6mSv416ZWTpnhbQ9F1C5vtK0u2tLq6/10sUYDPzn+dGo04W1Rz58b65qnyeGfB2oybvu3OpkWsQ98EliPoM0v8AwiXiPXBnxX4lliibrY6OPITHoZD87fpXa0UWFzW2RgaP4H8NaCQ+m6PapL1MzpvkJ9dzZOauW3hvRrLWJtVtNMtob+YYkuEjAZvXmtOinYnmb6lC/wBB0nVUK6lplpdD/ptArH8yK5W4+H82hM178Pr59MuFO46fNIz2c/8AslT9zPqtdzRSsNSaOd8LeLofEBnsru3bTtZs8C80+U/Mnoyn+JD2YV0Vc74n8L/2w0GpaZKLHXLHLWl4o6+sb/3kboR+NS+F/E8XiC1ljnj+x6pZv5N7ZMfmhcenqp6g9xQNpNXRu0UUUyAooooAKKKKACiiigAoornvGPiR/D+lxpYxC51a+k+z2Ft18yQ9z/sr1J9KBpNuyKvibxhLZ6knh/w1ajUvEE6bhFnEVqn/AD0mbsPQdTWZN8K7fXbSRvGmr32sXsy43rIYooD/ANM4xwOe5ya3vCPhdPDemubiX7Xql4/nX96w+aeQ/wAlHQDsK6Cla+5fNy6ROG0j4O+DNK09baTSkv3BLNPdHc7fljir3/Cr/BX/AELdj/3wf8a6uiiyFzzfU5T/AIVf4K/6Fux/74P+NI3wu8EspU+HLIZGMhT/AI11lFFkLnl3PK3+Hdv8M4pfEvhGW8lktm33lrI4YT2v8aDjqo+YH2r060uob6yhu7VxJBPGskbj+JSMg/lUroskbJIoZWGGBHBFcb8O5DYw6t4akJ3aLetFCGPPkP8APF+QOPwo2Kbc1d7o7OiiimZhRRRQAUUUUAN81PM8veu8jO3POPpTq8cl8I31/wDtHSX1vq0nk2ccV9MuTlFPyiH0w20n6Zr2OkncuUVG1mFFFFMgKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigArxz4j+BNdHjkeONKu99tZrFLNbxErMEjxuC9myATz717HSModSrAMpGCCOopNXLhNwd0Q2N7BqWnwXtnIJbe4jWSNx0ZSMg1PXD+CJG8P+IdX8GzsfKtW+2aaWPW2kJJUf7rZFdxQhSVmFFFFMkKKKKACiiigCG8vINPsZru8kWKCBDJI7HhVAyTXF+DLS58Sa1N431mJo/OQw6RbPn9xb/3yOzP1+lGsv/wnfidvDlu27RNNdZNWlU8TSjlLcH07t+AruVVUUKgCqowABwBS3NPhXmxaKKKZmFFFFABRRRQAVxFz/wASf40WUq/LDr2nPA/vLCdy/wDjpau3riPib/oNnoevjg6TqsMkjekTny3/AD3Cky4b27nb0UUUyAooooAKKKzfEWqLonhnUtTcjFpbSSjPchSQPxOBQNauxzXgD/iZa14p188i71I20Lf9M4Bs/wDQt1dvXNfDzS20j4f6RbS/65rcTSk9S7/Oc+/zV0tJbFT+JhRRRTICiiigAorC8VSeRp8UxvEt0jkLMjXTW5mG0/IGUg59B3rPuL+X+10/f3EFzvtvslo0hHmRtjzNyZ+fA3ZJzjHWhag9DraKRyyoSi7mA4GcZqDzrn/n0/8AIgoAsUVX865/59P/ACIKPOuf+fT/AMiCgCxRVfzrn/n0/wDIgo865/59P/IgoAsUVX865/59P/Igo865/wCfT/yIKALFFV/Ouf8An0/8iCjzrn/n0/8AIgoAsUVX865/59P/ACIKPOuf+fT/AMiCgCxRVfzrn/n0/wDIgo865/59P/IgoA434hxnR9R0TxhACP7LuRDeFe9tKdrZ9gcGu6BDKCpyDyCO9Zer2bazo13pt5Z7oLqFon/eDgEYz/WsT4f6tfXfhGC1vIN97pjtYXP7wA74jtz+Iwc+9LqW9Y+h19FV/Ouf+fT/AMiCjzrn/n0/8iCmQWKKr+dc/wDPp/5EFHnXP/Pp/wCRBQBYrzD4zwarrEOiaD4cvpI9RvLlm+zRSFDJGq8uzDoqn+deh3F9Ja20txcW4jiiQu7tKMKoGSa4zwJHe61f3nja/tCZNSHlafGzgeRaKfl/Fj8x/Ck9dDSHu+92Og8EeHn8L+D7HTLho5LqNN1zKn/LWQnLMSeSfc1v1X865/59P/Igo865/wCfT/yIKZDbbuyxRVfzrn/n0/8AIgo865/59P8AyIKBFiiq/nXP/Pp/5EFHnXP/AD6f+RBQBYoqv51z/wA+n/kQUedc/wDPp/5EFAFisHxxpZ1nwLrNggzJLaOYx/tqNy/+PAVredc/8+n/AJEFIZbh1KtacEYP7wUDTs7mf4Q1Qa14N0nUQ277RaRsT6nbg/qDWzXCfDKa4tNAvtG+z7m0nUp7bG8Dau7eg/75YV2fnXP/AD6f+RBSWw5q0mWKKr+dc/8APp/5EFHnXP8Az6f+RBTJLFcT8Tma+0vS/DsR+fW9Qit3A/55Kd8h/Ja67zrn/n0/8iCuJaWfWvjKpFuWi0DT+V3jAmmPr6hB+tJlw3v2O9VQihUAVVGAB2FLVcTXOebX/wAiCrFMgKKKKACiiigAIz1oxRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFcZZEaF8V72zPy2+vWou4h28+L5XH1K7T+Brs64r4mxS2mj2HiS0Um40G8S6bHUwn5ZR9Npz+FJlw1du52tFMgmjuLeOaFg0cih0YdwRkGn0yAoopskiQxPLKwREUszHoAOpoA4zx/M+rT6d4Os3Kyau+67ZTzHapy59t3Cj6muyghjtreOCBAkUShERRwoAwBXE+AI5Nd1LVPGl2p/wCJk/kaerfwWsZwpH+8ct+VdzSXcuWnuhRRRTICiiigAooooAKKKKACiiigDitC/wCJb8WvEmn/AHY9QtYNQiHqRmNz+e2u1rifE3/Es+J3hTVPupdifTZj67l3xj/vpTXbUkXLowooopkCO6xozuQqqMknsK4j4YK1/p2reJJVIfW9RlnQkc+Sh2Rj8lNX/iPqUun+B7yOzP8Apl8VsrYDu8p2D9CTW3oulw6LoVlplsMRWkCQr77RjNLqXtD1L1FFFMgKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACoL20h1Cwns7lQ8NxG0cinupGD/Op6KAOO+Gt/I3h+fQr1ib7QZ2sZc9WRf9W/4rj8jXY1w2uBfCXxAtfEYATTtXVbDUT0Ecmf3Mp/VT9RXc0kXPe/cK4z4i3k13aWXhXTXK3uuy+S7L1itxzK/t8vH411l9e22m2E97fTJBbQIZJZHOAqjqa5DwVaT61q15411OJo3v0EGnQyDmC1ByDjsXPzH8KH2COnvHX2VnBp9hBZ2iCOC3jWONB2UDAFT0UUyAooooAKKKKACiiigAooooAKKKKAOM+KkbR+DBqkX+s0i8gvlPoEcBj/3yWrsIZVngSWM5WRQyn2IzVTW9NTWNBvtNlAKXVu8RB/2lIrI+Hd++oeANKedt08MX2abPXfGSjfqtLqXvA6WiimyOsUbSSEKigsxPYCmQcXrZ/t34p6LpK/Nb6PC2p3I7eYfkiB/8eP5V21cT8OI31KPVvFdyCJNbui0G7qtvH8kY/EAn8RXbUkXPR27BRRRTICiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAqarpdprWlXGnalCJra4QpIh7g/yPvXlHinxn4s+FVhZaZJbQa3bs5W11CXfvMY/5ZyAfxjIw2eQOmc17FTZIo5QBLGrgHIDLnBpNXLjJLdXR55Bo2v8AxEa1u/GEK6VoI2TR6NFJue4bqDM2B8vQ7R+NeiKoVQqgAAYAA6UtFFhSlcKKKKZIUUUUAFFFFABRRRQAUUUUAFFFFABXEeEGOj+OPE3h2ThJJl1S0Hqkow4H0cfrXb1xfjaP+xNc0fxeit5Vixtb8qM/6NJxuI9FbB+hNJlx1uu52lcl8Qryd9Ht9A01yt/rs32SMr1jjxmWT8Ez+JFdS1xClsbhpUEATeZCw2hcZzn0xXI+Fo38R+Irrxfcxsts0f2XSUkGCIActLjtvbp/sgetDCOmvY6rT7GDTNNt7G0XZBbRLFGvoqjAqxRRTICiiigAooooAKKKKACiiigA6Um4eo/OhlDqVcBgeoIzmo/slv8A8+8X/fAoAk3D1H50bh6j86j+yW//AD7xf98Cj7Jb/wDPvF/3wKegtSTcPUfnRuHqPzqP7Jb/APPvF/3wKPslv/z7xf8AfAo0DUk3D1H50bh6j86j+yW//PvF/wB8Cs1tR09dT+x/YiVEoha4Ea+WshXcE65zgjnGOcZzRoGprbh6j86Nw9R+dYdpqWnajK221NusLg75I02yAkqOhOOezYPtTotV0+RL6VrExwWRYSTMsZDFeoADFvzAo0DU2tw9R+dG4eo/OqGnPb6ha+d/Z/kDdgLII2J9/kZh+uat/ZLf/n3i/wC+BRoBJuHqPzo3D1H51H9kt/8An3i/74FH2S3/AOfeL/vgUaBqSbh6j86Nw9R+dR/ZLf8A594v++BVDUbqz05o0+wG5lkDMIoI03bV+83zEDAyO+eeBRoPU09w9R+dG4eo/OsU6tphv7e2iszKJ0R1lSNNoD528E7j06gEDvila9sra5srKOz+1yXCFleMRgAKQCTuYE9egyaNBamzuHqPzo3D1H51lNdQJrEdg+kSr5m4pPtiKEAZJ4bcB25HWtH7Jb/8+8X/AHwKNA1JNw9R+dG4eo/Oo/slv/z7xf8AfAo+yW//AD7xf98CjQNSTcPUfnRuHqPzqP7Jb/8APvF/3wKhuxZWVnLc3EMaxxKWYiME0aD1LW4eo/OjcPUfnWE+s2EVszy6ZIkyyrE1uyRBwWXcDndtwR33e3Xiny3+nW2nSai1vvRkjk8ny1V1DYAzuIA59TRoLU2tw9R+dG4eo/OsW81Sysba2ln0yTdcAkRKsRZQOSSd2D9FJJ7CtVba2ZQwt48EZ5jFGgEu4eo/OjcPUfnUf2S3/wCfeL/vgUfZLf8A594v++BRoGpJuHqPzo3D1H51H9kt/wDn3i/74FH2S3/594v++BRoGpJuHqPzqO4hgu7aS3uUSWGVSkiOMhlIwQRWWmp6c9+1v9iYJueNZzEux3QZZBznIAPbBwcGm6ff6fqhWVbX7MEYACVE/ebgcYKkjt0PI7gUaBqjzLwf8G9W0/XLuPxLq32jQzgLaQXLhboA/KJF7KB2r2WNY4o1jiCIiAKqqAAoHQAViprOnva3Nx/ZziKCXyQ22I+a+/YFUBiQS397HWr1hJZ38DOtoInjcxyRSIu5GHUHGR+RNJJLYuc5Sepe3D1H50bh6j86j+yW/wDz7xf98Cj7Jb/8+8X/AHwKehGpJuHqPzpai+y246QRf98CpaQwooooAKKKKACiiigAooooAKKKKACiiigArIbQmbUzMt1ttWuBdNB5fzeaFC5D5+7wDjHXv2rXooAw9J8Py6VcSzJdxu7oseRb7d4Bzuf5vmfGRu4+lLP4cF5cX0l1NGFuoTCFgg2YBOctljvbpzx345rbooAzdG0o6XHcb5Y5JLiXzGEMPlRqdoX5VyccDJ55Oa0qKKACiiigArN1TS5b2aKe0uVtriNHj3vF5gKPjIxkc8Ag5/A1pUUAYB8LgahazR3QEVuYygeLdKuwYwsmeFI6jBzk+tWtS0mbUxbLNPAoicO7rbnzDhgRsbd8nTB4OfatWigCrDZeXqNxePIZHlCooIx5ajsPxyatVz+qeIhpPiWCzliubhbi2LRQ28QdiwY5Pr0rQs9d069s0uY7pI0dzGFmPlsHHVSD39qrlaVyeZXNCiqiarYSTXMSXcO+1OJwXA8v61ZjkSWNXiZXRhlWU5BH1qSh1V9Qs01DT5rSUlVlXbuXqPQ/nViigDn7jw3PdQubm9hmuJJA0nmWuYXUIVCmPfnuTnd1P4Vei02eDR1sEuIZQkSxq1zAZASOpYbhkeg4x6mtKigDm7nwkZ9MtrNbxcRI6N5sG9fmIO5BuGxgR8pycDsa6JF2Rqu4ttAGW6mnUUAFFFFABRRRQBijw832pj9sZbYSSzRRomHSSRSpO/PIG5sDHfqaXR9Dl0lpmW5hZpim9Y7cohCjGdu4/Oe7Z7DitmigDFOgO819NLJas11H5YjFqRHjJOXXfl2565FWtG0mPR7JoIyrM8hkdlTapY+gycD8T9a0KKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooA5LW7TUJfHun3mmQwztaWjl45ZTHkMSODg1m6n4P1u6sZEjazeW7klmnGcCN2xtCsVJ2gDnGCTXeeRF9o8/Yvm7Nm/HO3OcfnT619o1a3Qz5E73OGvPCOo3K32yK1ja4kguA4k5YooDRn5ehIJzyPaum8Pae2l6LHavF5RVmbZ5vmYySeuB69hWnRUubasNQSdwoooqCwooooAKKKKACiiigAoorHvNfks7+e3k0y5aOGEztOrx7Sg743Z6jHSplJRV2XCEpu0TYorMTxDprweeLlBBsZmkJxt2kAgjr1YUreIdLAhxdqzTpuiRQSz9eMevBwD1wafMkV7Kp/KzSorIh8S2Bsbe5vXWzFwSI1kdWJA6nKkge/p3qw+uabGsrPdLiKQROQpO1j+H69KOZA6NRO3Ky/RVSz1K3vZZYoyRJE7KyHr8rbc/QkcVA3iHSlTe17GE8wxbsHG4decdPfp70XQvZzbskaVFZv8AwkOkm7+zC9j87zBHs5+8TgD8xj61pUJp6omUJR+JWCiiimSFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAVTutMgvJJnlLgzW5t22n+En+fNXKKTSasxqTi7oy28P2byO5aXL7sjcMclT6f7A/WqNz4albWLee0uWht1fzJhuBZ2DOy8bexc85+oNdFRS5Uaxr1I9Tn5fCiSyKGvrgxv5n2kkRhrjftBDYTAGFwdoB96ZJ4I014J4Ulmijmn89giRdeeMlCSMsTzkg9CK6OilyRLWKrLZmXaaDBY3ktzaXFxHJO26cAqRKc5yRt69RkY4+gxA3he3Notr9tvBCgMaoGTiI8GLO3O3HGfvf7VbdFPlRHtql73MseH7QMGDS5DBvvDqJPM9P736VqUUU4xUVZGcpSluFFFFMkKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooA//9k=)

Figure Things Speak Result evidence-4 for week-6 Lab

**Discussion on Result:**

The Arduino reads the cardiac signal data and sends it to the MIT software through the HC-05 Bluetooth module, where it is then saved on the Things Speak server for MATLAB visualisation.

The code specifies 6000 samples to be captured, and the Bluetooth module's TX and Rx are linked to pins 5 and 6.

Timer0, timer1, and timer2 are the three timers on the Uno. Each timer has a counter that is incremented with every tick of the clock. When the counter reaches a predefined value recorded in the compare match register, CTC timer interruptions are triggered. When a timer counter hits this value, it will clear (reset to zero) on the next tick of the timer's clock, then count up to the comparison match value once more. You may adjust the frequency of timer interrupts by setting the comparison match value and the pace at which the timer advances the counter.  
  
The first is the rate at which the counter is incremented by the timer. The timers may increment their counts at the fastest speed possible thanks to the Arduino clock, which runs at 16MHz. At 16MHz, each counter tick represents 1/16,000,000 of a second (63ns), therefore a counter will take 10/16,000,000 seconds to reach the value of 9, and 100/16,000,000 seconds to reach the value of 99.  
  
Well,we will find that setting the counter speed to 16MHz is too fast.  Timer0 and timer2 are 8 bit timers, meaning they can store a maximum counter value of 255 Timer1 is a 16-bit timer, which means it can hold up to 65535 counter values. When a counter reaches its maximum value, it resets to zero (this is called overflow). This means that at 16MHz, interrupts will occur every 256/16,000,000 seconds (16us) for the 8 bit counters and every 65,536/16,000,000 (4 Ms) seconds for the 16 bit counters, even if the comparison match register is set to the maximum counter value. Obviously, if we just want to interrupt once every second, this isn't very beneficial.

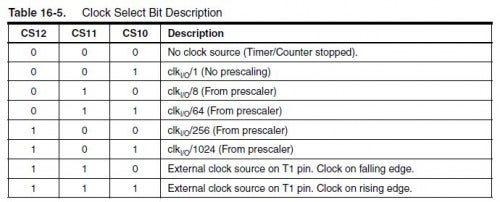
Instead we can control the speed of the timer counter incrementation by using something called a prescaler.  A prescaler dictates the speed of your timer according the the following equation:  
  
**(timer speed (Hz)) = (Arduino clock speed (16MHz)) / prescaler**  
  
So a 1 prescaler will increment the counter at 16MHz, an 8 prescaler will increment it at 2MHz, a 64 prescaler = 250kHz, and so on.  As indicated in the tables above, the prescaler can equal 1, 8, 64, 256, and 1024.  (I'll explain the meaning of CS12, CS11, and CS10 in the next step.)   
  
Now you can calculate the interrupt frequency with the following equation:  
  
**interrupt frequency (Hz) = (Arduino clock speed 16,000,000Hz) / (prescaler \* (compare match register + 1))**  
the +1 is in there because the compare match register is zero indexed  
  


Figure Bit CS12 in TCCR1

When the power is turned off, the sleep mode is triggered, which is similar to the code (set sleep mode). When an interrupt at pin 2 occurs, the Arduino execution exits to read the data. We can save time and electricity by utilising this function.

**WEEK-7 LABORATORY:**

**INTRODUCTION:**

The [AD8232](https://www.sparkfun.com/products/12650) is a neat little chip used to measure the electrical activity of the heart. This electrical activity can be charted as an ECG or Electrocardiogram. [Electrocardiography](http://en.wikipedia.org/wiki/Electrocardiography) is used to help diagnose various heart conditions.

|  |  |  |
| --- | --- | --- |
| Board Label | Pin Function | Arduino Connection |
| GND | Ground | GND |
| 3.3v | 3.3v Power Supply | 3.3v |
| OUTPUT | Output Signal | A0 |
|  |  |  |
| LO- | Leads-off Detect - | 11 |
|  |  |  |
| LO+ | Leads-off Detect + | 10 |
|  |  |  |
| SDN | Shutdown | Not used |
|  |  |  |

Table 2 Table 2 Pinout for AD8232

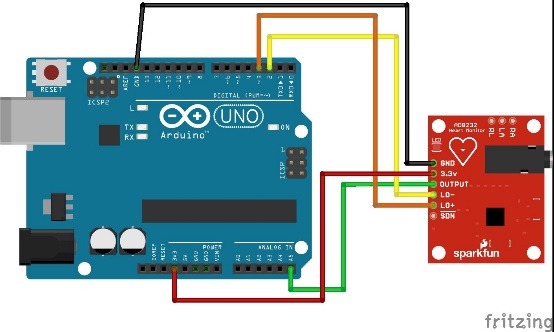


Figure 24 Pinout for AD8232 and Arduino Uno

THEORY :

This weeks laboratory completely deal with the mills and ECG BPM(Beats Rate Per Min) calculation by using simple thresholding , will calculate the events as briefly discussed below.

The Millis() function on the Arduino is a highly handy function that produces a number representing milliseconds since the device was reset. You may learn how to utilise it successfully for event timing and delays, as well as how it works in detail, on this website. In addition, by suing Millis as a timer, Millis() delivers the current time in milliseconds (1/1000th of a second) since the board was turned on (or reset it). It allows you to measure time from within your application, as opposed to the delay() function, which provides no time feedback at all.

To use Millis() for timing, you must first record the time at which an action occurred to begin the timing period, and then check whether the required period has expired at frequent intervals.

A picture containing shape

Description automatically generatedA picture containing schematic

Description automatically generated

For example, every 30 seconds, the water temperature may be measured, or every minute, a servo motor could move left and then right. This basic use of Millis will not suffice for these types of apps. Let’s go back to the beginning of our storey. We still have event 1, and the event is triggered when Millis surpasses this value. However, in addition to activating an action, the event 1 time must be reset to a later point in the timeline.

Text

Description automatically generated

We can update the event time by moving the hash mark (marking some event) along the time line so that every time we reach to that hash mark, we also update the hash mark for the next time we run the code.

**SIMPLE THRESHOLDING:**

This thresholding concept is alternative for Pan-Tompkins Algorithm. where below figure shows the PQRS intervals, R -R interval describes one event in the sense capturing the first pulse time interms of bpm.The upper threshold values can be consider as above 525 and lower considered as below 525.Well by setting the flag high it is ignored that above threshold value which is above 525.

**Diagram

Description automatically generated**

**TEST PLAN :**

1. This week need to Capture ECG data with the AD8232/ADS1292R. (using the simulator)
2. By Creating code to calculate heart rate using either the simplified thresholding approach the
3. Over Bluetooth, send the heart rate to a mobile phone, which is then uploaded to the cloud.
4. Get the last ten heartbeat readings from the IoT server and plot them on a graph on the mobile phone.

**EVIDENCE:**

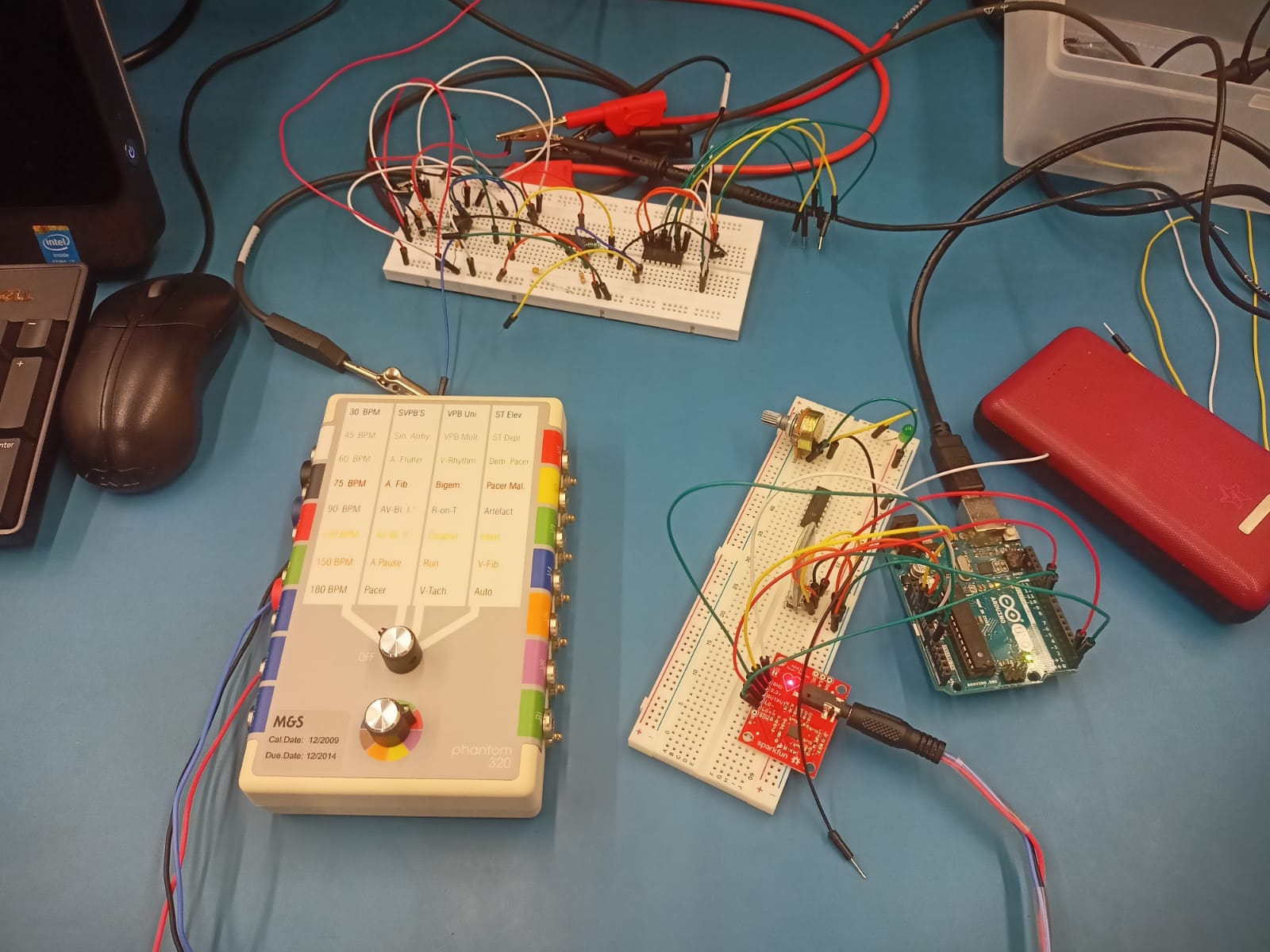
****

Figure Circuit Evidence-2 for week-7 Lab

**Code:** [**week7Testplan1code**](#week7Testplan1code) **(Appendix)**

**RESULTS :**

|  |  |  |
| --- | --- | --- |
| **BPM** | **Input** | **Output** |
| **30** | **A picture containing text  Description automatically generated** |  |
| **45** |  |  |
| **60** |  |  |
| **75** |  |  |
| **90** |  |  |

**![Chart, line chart

Description automatically generated](data:image/jpeg;base64,/9j/4AAQSkZJRgABAQEAYABgAAD/4RDcRXhpZgAATU0AKgAAAAgABAE7AAIAAAAGAAAISodpAAQAAAABAAAIUJydAAEAAAAMAAAQyOocAAcAAAgMAAAAPgAAAAAc6gAAAAgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAADkxOTM5AAAFkAMAAgAAABQAABCekAQAAgAAABQAABCykpEAAgAAAAM5OQAAkpIAAgAAAAM5OQAA6hwABwAACAwAAAiSAAAAABzqAAAACAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAMjAyMToxMToyMyAxNDo1MTo0OAAyMDIxOjExOjIzIDE0OjUxOjQ4AAAAOQAxADkAMwA5AAAA/+ELGGh0dHA6Ly9ucy5hZG9iZS5jb20veGFwLzEuMC8APD94cGFja2V0IGJlZ2luPSfvu78nIGlkPSdXNU0wTXBDZWhpSHpyZVN6TlRjemtjOWQnPz4NCjx4OnhtcG1ldGEgeG1sbnM6eD0iYWRvYmU6bnM6bWV0YS8iPjxyZGY6UkRGIHhtbG5zOnJkZj0iaHR0cDovL3d3dy53My5vcmcvMTk5OS8wMi8yMi1yZGYtc3ludGF4LW5zIyI+PHJkZjpEZXNjcmlwdGlvbiByZGY6YWJvdXQ9InV1aWQ6ZmFmNWJkZDUtYmEzZC0xMWRhLWFkMzEtZDMzZDc1MTgyZjFiIiB4bWxuczpkYz0iaHR0cDovL3B1cmwub3JnL2RjL2VsZW1lbnRzLzEuMS8iLz48cmRmOkRlc2NyaXB0aW9uIHJkZjphYm91dD0idXVpZDpmYWY1YmRkNS1iYTNkLTExZGEtYWQzMS1kMzNkNzUxODJmMWIiIHhtbG5zOnhtcD0iaHR0cDovL25zLmFkb2JlLmNvbS94YXAvMS4wLyI+PHhtcDpDcmVhdGVEYXRlPjIwMjEtMTEtMjNUMTQ6NTE6NDguOTg2PC94bXA6Q3JlYXRlRGF0ZT48L3JkZjpEZXNjcmlwdGlvbj48cmRmOkRlc2NyaXB0aW9uIHJkZjphYm91dD0idXVpZDpmYWY1YmRkNS1iYTNkLTExZGEtYWQzMS1kMzNkNzUxODJmMWIiIHhtbG5zOmRjPSJodHRwOi8vcHVybC5vcmcvZGMvZWxlbWVudHMvMS4xLyI+PGRjOmNyZWF0b3I+PHJkZjpTZXEgeG1sbnM6cmRmPSJodHRwOi8vd3d3LnczLm9yZy8xOTk5LzAyLzIyLXJkZi1zeW50YXgtbnMjIj48cmRmOmxpPjkxOTM5PC9yZGY6bGk+PC9yZGY6U2VxPg0KCQkJPC9kYzpjcmVhdG9yPjwvcmRmOkRlc2NyaXB0aW9uPjwvcmRmOlJERj48L3g6eG1wbWV0YT4NCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgPD94cGFja2V0IGVuZD0ndyc/Pv/bAEMABwUFBgUEBwYFBggHBwgKEQsKCQkKFQ8QDBEYFRoZGBUYFxseJyEbHSUdFxgiLiIlKCkrLCsaIC8zLyoyJyorKv/bAEMBBwgICgkKFAsLFCocGBwqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKv/AABEIAP8BxQMBIgACEQEDEQH/xAAfAAABBQEBAQEBAQAAAAAAAAAAAQIDBAUGBwgJCgv/xAC1EAACAQMDAgQDBQUEBAAAAX0BAgMABBEFEiExQQYTUWEHInEUMoGRoQgjQrHBFVLR8CQzYnKCCQoWFxgZGiUmJygpKjQ1Njc4OTpDREVGR0hJSlNUVVZXWFlaY2RlZmdoaWpzdHV2d3h5eoOEhYaHiImKkpOUlZaXmJmaoqOkpaanqKmqsrO0tba3uLm6wsPExcbHyMnK0tPU1dbX2Nna4eLj5OXm5+jp6vHy8/T19vf4+fr/xAAfAQADAQEBAQEBAQEBAAAAAAAAAQIDBAUGBwgJCgv/xAC1EQACAQIEBAMEBwUEBAABAncAAQIDEQQFITEGEkFRB2FxEyIygQgUQpGhscEJIzNS8BVictEKFiQ04SXxFxgZGiYnKCkqNTY3ODk6Q0RFRkdISUpTVFVWV1hZWmNkZWZnaGlqc3R1dnd4eXqCg4SFhoeIiYqSk5SVlpeYmZqio6Slpqeoqaqys7S1tre4ubrCw8TFxsfIycrS09TV1tfY2dri4+Tl5ufo6ery8/T19vf4+fr/2gAMAwEAAhEDEQA/APpGiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAoorJ8U6rPofhTUtTtFjea1gaVFlBKkj1wQcfjQBrUVheJNcudH8OwX9skTSyXFvERICVxJIqt0I5wxxVjUvE2k6Rc+Rf3RR1UNJsheRYVJwGkZQRGpPdiBwfQ1XK/0J5l+pq0Vnaprtho6br55sbS5EFvJOVUfxMI1JVfc4FR3niXSLH7H9oux/p6M9qI42k84AA/LtBycEYHU54zSs2O6NWiudtfGFvd+LpdDWzvFK28cyzNazDJfdwwKDYAB94nBJI6g1N4t1e80bR4p9O8jz5ruC3BnQuqiSQKTgMpOM+oo5XdLuHMtfI3KK5zRddvm1bWrDW5LJ49LEbm+t1aKPDKWKurM21lxk/MeGHTvoaZ4i0zWJZIrGdzJGiylZYHiJRs4dQ6jcpwfmGR70NMLo06KyrXxLpd5fGzgmlM/ltKge2lQTIuMtGzKBIORypPUVW8LeKofE9vcSRWlxbNBPJERLDIqsFcqDuZFGTjJXqvQ0crDmRvUUUUhhRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAVgeOrae88B6zb2kMk80lq6pFEhZnOOgA5JrfooA4PxP4P0yHw5aS6P4dtI75Lu0bfa2KiVQJU3HKrkDGc+2ai1bSJIfEWvLqV1rUGn6wiEPptktwsiiMRtG+IZHUjqDwCG45Br0Gir5318/xt/kQoJbeX4X/wAzh9Yl1WOeDS7SXVYNM/sxVt5bax82W5mPy7JWdGWLjH3gnLElhiqeg2V403gJpLG7T7BZ3ENyZrd08lxEq4bcBjJBAPQ9s16JRTU7f16/5hybeX/A/wAjl0jls/idcXM1vcG3vtPhhhmjgd0Do7lgzKCE4YctgGk+Ido154ZhjFnJeRrf2zzQxwNMTGJVLfIoJIxnIxXU0VKdrPt/ncdt/P8Aysecz6Nc6tZeI7Dw5bXdlpNxYoltb3ML28f2jJZhFHIAUUjAPAXJz61o6PD5upf2klxr1zq0Ng8UdtqViLeJc4OwukKIx3ADIYjqRkc12tFNy0Fy6nm2mS6vd+KPDWp6kdYnISdLxJdNMMVpJIgwigRhiuRjexZflHzV0PgiOWxt9S067t7iGePULibLwOI3SSVmUq5G1uD0BJHeuoooc79P6vcFC3X+tgoooqCwooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAoorzX4zx+MpNH0//AIQs3uwSv9rFgxE3QbPu/Nt+9nHtmk3ZFRjzOx6VRXl3h34jX3hnSbHTvidp2oafc7Qo1OSPzIpsnjcVztYAgHr0ycV6Pp+p2OrWa3WmXcN3bv8AdkhkDqfxFCdxyg47lqiiimQFFFFABRRTJp4raFpriVIokGWd2Cqo9yaAH0Vxl/8AFjwlaXH2a0vpNVuu1vpsLTs30I+X9a4H/he+rnx7/ZyeHZW08y+ULTyG+3H325xu77cfj3qeZI1jSnLoe40VwZ+LWlW/Oq6H4j0uPvJeaYyqv1wTW5oXjrwx4lkEei6zbXEpGRESUc/8BYA/pTuiXCS1sdBRRRTICiiigAooooAKKKKACiiigAooooAKKKKACiiigAorkfGnxK0HwRFENQd7q5lYqLa1Ks64xksCRtHI69aXT/il4N1DT4boa9Z23mru8m5lVJE9mXPBpXRfJK17HW0VzX/CxfB3/QzaZ/4Er/jR/wALF8Hf9DNpn/gSv+NF0Lll2Olormv+Fi+Dv+hm0z/wJX/GsLxB8a/COhTQRRXL6qZeWbT9sixjPclhz7UXQ1CT0SPQqKgsb2DUtOtr6zffb3MSzRPjG5WAIP5Gp6ZAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFADJoY7iFop40ljcYZHUEMPcGuQvfhhohunvNBlu/Dt63WbS5jErf7yfdI9sCuyoosUpNbHBNF8SvD5/0efTfFVsvRZl+yXGO3I+Q/U0D4o/2f8AL4p8K63o3rN9n8+Ef8DT/Cu9opWHzJ7o8u1r4++FdOkhXTo7vU9/+tMcZi8of8DAyfpx71qL8YfDt3Gv9iWur6xMwyIbKwdmB9DnA/U1t+JvAXhzxfc29xr+ni5mtxtRxIyHbnO07SMjNb1vBFa20VvbRrHDEgSNFGAqgYAH4UtSm6dtEcL/AGz8QfETbNI0O28OWx4N1qknmy/VYl6H/eqSL4XWd/Mlx4x1bUPElwpzsuZfLgU/7MSYA/M13VFOxPO1toVNO0rT9ItxBpdjb2cWMbIIlQfpT/7Psvt/277HB9r27ftHlL5mPTdjOKsUUybsK5/XvAnhvxIudV0qBphytxEPLlU+odcGugooBNrVHm/9oa78MpVTXJ59b8KkhUvypa4sOwEmPvp2z1/QH0O1uoL20iurOZJ4JlDxyxsGV1PQgjqKfLFHPC8UyLJHIpV0cZDA8EEdxXnwhf4W6i0kReTwdeS/PH946VKx+8O/lMTyP4Tz35Wxek/U9DopqOssayRsrowBVlOQQe4NOpmYUUUUAFFFFABRRRQAUUUUAFFFFAEV1d29jZy3V5MkFvCheSWRsKijqSa87Go698TpiNBuLrQPCyllOoBdtzfnp+6B+4n+11/HIC6kZPib4qk0aBv+KV0iYf2hIjY+3TjkQgj+BT97H+Br0WKKOCFIYI1iijUKiIuFUDgAAdBS3NPgXmcLc/BfwXdaalo2nyq6tvN0s7ec5PUsxznP0rf0/wADeGNN0+Gzt9CsGjhXaplt1kY+5ZgST9a3qKLITnJ7syf+EU8O/wDQB0z/AMA4/wDCj/hFPDv/AEAdM/8AAOP/AArWopk8zMn/AIRTw7/0AdM/8A4/8Kw/EHwo8I+IpoJbrS0tnh4/0LEIcejBRg/Xr712VFKyGpSWqZwXhYSeDPF0vg6aSWTTLuNrrRnlbcYwP9ZBnqdv3h7V3tcX8UbWRPC8evWQ/wBO0G4S+hI7qpxIv0KE5+lddZ3cV/YwXls26G4jWWNvVWGQfyNC7DlqlImooopkBRRRQAUUUUAFFFeDfC3QvFkXxc1SfUNReeDT5JItRkM5ZbhmU7AB35w3tjFJuxcY8ybvse80UUUyAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAqK6tYL6zltbuJZoJkKSRuMhlIwQalooA+fvhX4n8VaR4+s/Cutedb6WzSwra3aY8kqrMqo7DJ5AAGTkH6V9A1znjrw8/iLwvNDZnZqNqwurCYdY505XB7Z6fjU/g3xHH4r8K2eqKvlyuuy4iIwYpV4dcfX9MVK00NajU/eSNyiiiqMgooooAKKKKACiiigArifiD4hvIvsvhfw23/E91jKI4GfssP8AHMfTAzj8fSul1/XLPw3odzqupSbILdM47u3ZR6knAFc54D8PXccl14p8SR413V8M0ZORaQ/wQr6YGM+/0pPsXGy95nQ+HNAs/DGgWuk6cuIbdMFiOZG/iY+5PNadFFMlu7uwooooEFFFFABRRRQBDeWkV/Yz2lwN0NxE0Ug9VYYP6GuM+Et3KfBr6Pdtm60O7l0+XIwSEbKn6bSB+FdzXA+HR/ZHxn8T6Z0j1S1g1OFT6j925/Fj+lJ7lx1i0d9RRRTICiiigAooooAZPMlvbyTzNtjjUu7egAya4f4QwvL4On1q4UifW7+e+fd1AZ9oH5Ln8au/FLU20r4a6u8JPn3EQtYgvUtIQnH4En8K3fD+lronhvTtMQAC0to4TjuVUAn86XU02h6mjRRRTMwooooAKKKKACiiigAorI8TatPouj/arVIS3mpGXuGIjjDHG5sc4FQ+HtWudSlnW5v9Gu9iggabMXK/72TwKpRbXMS5JOxu0UjMqIWdgqqMkk4AFVLfWNMvLeW4tNRtJ4YRmWSKdWWMYz8xBwOPWpKLlFRWt3b31utxZXEVxC+dskLh1ODg4I461LQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFedaM48I/F7UdEb5NP8RIdQs/Rbhf8AWqPqBu/KvRa4b4q2Ew8PWviKwGb3w9crfJj+KMECRfoV5P8Au0n3NIb8vc7mioLK8h1DT7e8tW3wXESyxt6qwyD+RqemZhRRRQAUUUUAFYXjDxdp3grQTquriVojIIo44VBaRyCQBkgdATye1bteb/E2OPxdqel+A7VA89zKt5dzjk2cCZG7/ebJA+vvSexcEnLXYg8P3/8AwtrxBDrTRPD4b0aQeRaTAb7i72glnAJGEDDA9/rXp9Y/hfwvpfhDRV0vRYmjgDmRi7bmdzjLE+vA/KtihBOSb02CiiimQFFFFABRRRQAUUUUAFcD4u/4lXxV8HawOI7ozaZOfXeuYx/31mu+rhfi/BIPAZ1O3XdPo95BfxY9UcA/oxNJ7F0/isd1RUdvPHc20U8J3RyoHQ+oIyKkpkBRRRQAUUUUAcD4/wD+Jv4w8H+HF+ZZb46hcL/0zhXIz7Ekj8K76uA8Pf8AE9+MfiLVz80OkW8el25J/iPzyY9wePxrv6SNJ6WQUUUUzMKKKKACiiigAooooA53xxDDL4c3XFzbWqwzxy+ZcxmRMg9No+9npjmq/gy7iu3umhbSPkChksbN7eRT/tB+cenFJ42v7FYbexuL9rG4Ei3STramYRBG+8wBGBnjJqh4YkvNW8RTX41l2lWKNZYZ9K8kyQ5JUqd54PPNbwX7t3/r8DGb99Ha3SJJZzJLu2NGwbYCWxjnAGcmuTEd9Poc8Ea3l1Y2dxbNCbm2MU0saMrSLs2qWwBx8oJ6c12NFYGxj6EGkn1K7WGWCC6uQ8KTRtGxARVLFGAK5KnqMnr3rUmWdseRJGnrvjLZ/IipKKAK2y+/5+Lf/vw3/wAXRsvv+fi3/wC/Df8AxdWaKAK2y+/5+Lf/AL8N/wDF0bL7/n4t/wDvw3/xdWaKAK2y+/5+Lf8A78N/8XRsvv8An4t/+/Df/F1ZooArbL7/AJ+Lf/vw3/xdGy+/5+Lf/vw3/wAXVmigCtsvv+fi3/78N/8AF0bL7/n4t/8Avw3/AMXVmigCtsvv+fi3/wC/Df8AxdGy+/5+Lf8A78N/8XVmigCtsvv+fi3/AO/Df/F0bL7/AJ+Lf/vw3/xdWaKAK2y+/wCfi3/78N/8XUdxa3N1bS288ts8UqFHU27YZSMEffq7RQBwXwzkv7XR77w3NcxefoF29p88JJaI/NG33hwVOB9K7TZff8/Fv/34b/4uuO1Ajw98YrC9+7a+IrU2cx7efF80ZPuVJWu6pIue9+5W2X3/AD8W/wD34b/4ujZff8/Fv/34b/4urNFMgrbL7/n4t/8Avw3/AMXRsvv+fi3/AO/Df/F1ZooAytX1GXRNGu9Tvru3S3tYmlc+Q3QDoPn6npXK/DbRtTNhc+KdWkiXVdeYXEgkhLGGL/lnGPmGAFwcfT0o8bn/AISrxZpPgqE7rbI1DVsdBCh+SM/7zY/Su+VQqhVAAAwAB0pbs0+GNu5X2X3/AD8W/wD34b/4ujZff8/Fv/34b/4urNFMzK2y+/5+Lf8A78N/8XRsvv8An4t/+/Df/F1ZooArbL7/AJ+Lf/vw3/xdGy+/5+Lf/vw3/wAXVmigCtsvv+fi3/78N/8AF0bL7/n4t/8Avw3/AMXVmigCtsvv+fi3/wC/Df8AxdGy+/5+Lf8A78N/8XVmigCtsvv+fi3/AO/Df/F1neINJu9Z8N6jprz27C7tZIceQRyykD+P1raooBaHGfDLULzVvhros4uIQY7cW7B4SWBjJj5+Yc/LXVbL7/n4t/8Avw3/AMXXGfDT/iX3/ivQDx/Z+sPLEv8AdimAdB+hrvKS2Ln8TK2y+/5+Lf8A78N/8XRsvv8An4t/+/Df/F1ZopkFbZff8/Fv/wB+G/8Ai6q6neXGlaTd6hc3NuIbWF5n/cEcKCT/AB+1adcJ8XbuX/hDY9GtDi71y7isIgBk4ZssfpgY/Gk9EVFXkkN+FGnX1v4Ehv5ZYVuNXmk1CbdCSS0jcHO4fwha7VUvAw3zwFc8gQEEj676dZ2sVjYwWluNsVvGsSD0VRgfoKmoWiCTu2wooopkhRRRQAUUUUAFFFFAGHrPhW01u7a4nur2Bmg8hxbzbA6ZzgjBz1qXR/Dlvo91Lcx3d5dSyRrEWupt+1VzgDgeta9FVzO1ieVN3CiiipKCiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKAON+KmmT3vgae9sB/p2kSpqNswGSGjOT/47urpNE1SHW9CstUtv9VeQJMo9Nwzj8OlXJI1ljaORQyOCrKRwQe1cL8MpDpP9s+Drhj5uiXbfZ9x5a2kO+M/qR+VLqabw9DvKKKKZmFVtRv7fStMub+9cR29tE0sjHsoGTVmvP8A4iyv4h1jSfA1m5H9ouLnUWU8x2sZyR143MMD6UmVFXdib4YWNxdWF94t1VNt/wCIJvPVT1itxxEg9sc/iK7qmxRJBCkUKBI41CqqjhQOAKdQtAk+Z3CiiimSFFFFABRRRQAUUUUAFFFFABRRRQBwUH/Ep+PF3H92LW9ISXP96WFtuP8Aviu9rgfiEf7M8V+C9fHC2+pGylb0SdNuT7DbXfUkXLVJhRRRTICuAvl/4SL43WNr9618N2TXMg7efLwo/BcMK7yaVIIXmmYJHGpZ2PQADJNcR8LInvtL1TxRcqVm1++e4QHqIVOyMfgAfzpM0jomzuqKKKZmFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFee+Mo28MfEHQ/GEbbLO4I0vUz2COf3bn6N1PsK9Cqhrej2uv6HeaVfrut7uIxv7Z6Ee4OCPpSZUXZ6l+iuJ8A6/co0vhHxI2zXNJUKGbgXkHRJU9eMA+/447RpY0jeR3VUQEsxOAoHXJ7UbhKLTsQ6hf22l6dcX19KsNvbxmSR2PCqBk1xnw3srnUm1Dxpq0bR3mtuPs8b9YLVeI1/HqfXis2+1D/hafib+xNJdn8Kac4k1K9QfJeyggrAh7r3JHX/vkt6YiLHGqRqFRRhVUYAHpRuyn7sbdWLRRRTMwooooAKKKKACiiigAooooAKKKKACiiigDj/irpj6n8NNWWDPn2sYu4mHVTEwcke+AR+NdHouoprGg2GpR423ltHOMdtyg4/WrU0KXFvJDMu6ORSjqe4IwRXF/Cm5ZPCU2iztm40O9msJM9SFYlT/AN8kflS6l7w9Dt6KKKZBxfxU1Ka18FvptgT9v1qZNOtlB5zIcMfptz+ddTpWnQ6Ro9pp1qMQ2kKQpxjhQBn9K4vjxT8ZgRl7Hwtbf8BN3L+nyoPwNd/SW9zSWiSCiiimZhRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQBzvi/whZ+KdPJx9n1SBGNjfxsUkt5MHB3Lztz1Hf615V4F+CuuJeXkfjG8nttMk+/aWl4cXjZ6vt/h69eee1e70UnFN3NI1JRVkVtO06z0jT4bHTLaO1tYV2xxRrgKP896s0UUzMKKKKACiiigAooooAKKKKACiiigAooooAKKKKACvP7T/im/jdd2p+S08S2YuIxnj7RDww/Fea9ArkviFo93e6Nbaro8Qk1XRLhb22TOPNC/fj/4EuR9QKTLg9bPqdbWT4o16Hwz4avdVnG7yI/3cfeSQ8Ig9yxAqLQvF+h+IbK1m0/UrVpLiLzRbGZfNQfxArnPBBB47VzglHxB8ZQtbMsnhvQZ95lByt7dgcAeqx5znuTRfsCjrr0NnwF4fl8P+F41vzv1O9dry/kPVppOWH4cD8K6WiimS3d3YUUUUCCiiigAooooAKKKKACiiigAooooAKKKKACiimSwRTY86JJMdN6g4oAfRUH2G0/59Yf+/Yo+w2n/AD6w/wDfsU9Bak9FQfYbT/n1h/79ij7Daf8APrD/AN+xRoGpPRUH2G0/59Yf+/Yo+w2n/PrD/wB+xRoGpPRUH2G0/wCfWH/v2Kz7TUNDvTN9nWEiFPMZntyilOfnUsoDLx95cj3o0Hqa9FZFlNperOpsYlHksGkSW2aJirKdp2uoJBPQ9OD6U6C50m51CSzhtS0sTFXb7E4jBHJHmbduefWiy2Fc1aKyrK60jUZnjs7cSbM/vPsbCNsHB2yFdrc+hNXvsNp/z6w/9+xRoGpPRUH2G0/59Yf+/Yo+w2n/AD6w/wDfsUaBqT0VB9htP+fWH/v2KPsNp/z6w/8AfsUaBqT0VQvjpmnW4muoI1UsEVUg3s7HoFVQSx9gOxqtJqGiRWMF35SyRXBIiEVo0jtgEn5FUsMYOcjjvijQepsUVmH+zUs5NUZbc2P2cSiRY9w2gFi2APQjp6UxLvS3sXu1sZvKVtpB02UOT7Js3Ec9QMUaIWprUVm6dLpWqwvLZ26lY5DG4ktWiZWHUbXUHvVv7Daf8+sP/fsUaBqT0VB9htP+fWH/AL9ij7Daf8+sP/fsUaBqT0VB9htP+fWH/v2KPsNp/wA+sP8A37FGgak9FZV1eaLZ3qWtxHEsrbeluWVNxwu5guFyeBuIz2pDe6Kup/YDFGJ9/l/8erbN+3dt37du7HOM5o0DU1qKyLiXTdJhtYr6DfNIuFENo8zOVA3HCKT+Jovb7RdPeNbuJULpv4tGYIucbnIU7B7tgdfQ0aBqa9FQfYbT/n1h/wC/Yo+w2n/PrD/37FGgak9FQfYbT/n1h/79ij7Daf8APrD/AN+xRoGpPRUH2G0/59Yf+/Yo+w2n/PrD/wB+xRoGpPRWRaahod6Zvs6wkQp5jM9uUUpz86llAZePvLke9FlNperOpsYlHksGkSW2aJirKdp2uoJBPQ9OD6UaBqeV3nwHub34j3GqzahbjRbi7a5eKMsk2GYsYwAMAc4yD07V7HYWFrpdhDZadbx21tAu2OKNcKoqlHc6TLqUlhHbFp422uRZP5anaGwZNuwHBHfvSWN7oupTtDaRxs6gsN1sUDqDgshZQHGe65HI9RSSS2NJVJSWvQ1qKg+w2n/PrD/37FH2G0/59Yf+/Yp6GepPRUH2G0/59Yf+/YpVs7ZWDLbQhgcghBxRoGpNRRRSGFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAR3EXn20sW4r5iFdw7ZGM1y1tpmrPbRR/Yo7aWw0uSyjaZkZJ5G2DIAJ+T93/EAfm6V1tFL+vz/zGcxoWm3Fqt2LnTbk2jmHyre8limlVlPJDbiNi/KQC2QQ2AOBU8VjdJ4leSytrqztn8xrqSa4DRzsQApRN7bTnnOF4HeugopvUS0OW8NaXqWnXdvFKl1Fb29qYZzPdCWOdwVCNEu47AAG7L1HB6jqaKKdwCiiikAUUUUAZWtW1w82n3lrA1y1lcGRoEZQzqUZDtLEDI3Z5I4BrDutG1B9Pt/Ms7g+ZNczyR2M6R3Fu8hJTDswBABIbBwTjgjNdjRSsMxYLJjogt9T01Z5zYCGc2+xBIMEGJTlSO/oozwaisItXtdFv/JglEjSH7Db3k4lkiUgD533HPzbmxuJxgZ7DfopvVsRXsLNLCxitoiWEa8s3V26lj7k5J9zViiigAooooAKKKKAOY1XTL+S61S3t7UzRaqIh9o3qFt8Da24EgngZG0HJ6460z+zNS/4SxbtbaVR9rLvMJU+zvD5e0ZTO7zRjG7bnqN23AHVUULR3DpY57WbFptOh/s/SbtLqMS/ZjbTpD9nc/xNhwCpODjDe47VS1/StWu7hGigkmmFsiRT28qIglDZbzlYjfGfl+UBuN3AODXXUUbAAzgZ698UUUUAFFFFABUdxF59tLFuK+YhXcO2RjNSUUmrqw07O5yVtpmrPbRR/Yo7aWw0uSyjaZkZJ5G2DIAJ+T93/EAfm6VNoWm3Fqt2LnTbk2jmHyre8limlVlPJDbiNi/KQC2QQ2AOBXT0VV9b/wBdybaWMD7DcL4qE1laXVvA7M17LLcBobgbAoCx7zhshedq8KeTnmto2jXMGtQSlb6CxsYJIYIbuWJsbiuAnl5O0Berkt0966iiktBhRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQB53tvP7D1zW01rUIrmyvJ/KRrktDtVuEKHjHatJvGdyms2FsUtHjuXiikjTzDJE7rn5jjYMH+HOcVqf8IZopupJpIJZBJMZmie4cxFyc5KZ2nn2p9x4Q0i5v2vHimWZpRN+7uJEUSD+MKDgHAxmt1KDtzGTjLWxz0PjDV7TR5J7+O2nlm1B7S3Kq5C4JzuVVJIAHGOTXVeH9Tn1bR47q6tmtpizKyFWXODjIDAHB68ioH8KaTJHcI0MuyebzyoncBJM53pz8hyeoxWjY2UdhaiCF5nUEktNK0jEn1ZiTUSlBrRajSknvoWKKKKzNAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigBk4laBxbOkcpU7HkQuqnsSARke2R9a5mHWtat7dJ7z7Leh757RYLS1MbnaXGQXlIydg64A5ya6mqq6baKECxYEc7XC/MeJGJJbr/ALR46c1EottNf1r/AJXN6dSMYtSV/wCn8+xljxRCYfPS3uJVeOBo4EjXzN0pYBcl8dRg9AOuSOinxXbfudtleMH8vzSqp/o5eQxqH+bn5gR8u7GMnjBN6LRNPhCCK327BGF+djjyySnfsWP9azb/AMJxXeoWk0Eq20ELo8kaiQtLtkMgBPmBSNxP3kbGTgg8hLn0ubReGb1TX9eX9dxLbxDcW8NydUjMl0kyRrY21sUkTexVPmd9rg/3xtXg5xggLF410yXULOxIkjubpihikaMNEwdk2su/LfMjDKBgMZJAINWLnwxZzwNGkl0hkmjkkla6leQhG3BVcvuQcn7pAGTxU8Xh/ToZYJIY5o3hXYClzIPMGSf3nzfvOWY/Pnlie5pJTsNywrTbTv8Ad0/z/wCDcym8bWb28dwgEUKXv2WffJHIc+WzAKY3YZJCjGc84x0rYvtUFja28jWlxLPcOI4rWPZ5jMVLFclgvAVifmxwetV4fC2kQwmL7PJNGcZS4uZJgcKyYw7Hja7DHpj0GJn0GxksltZBctGhUoxu5d6FRgFX3blOOCQRnJznNNKdtSZywzkuVO39ef6/cVpvE0FvM6zWV4saBwZsIUMixl2j4bO4AHnG3Ixmm2niiK6vorVtNvoDJL5W+UR7VJj8xc4cn5lBxxkY5A4qy/h3TJbhp5YZHdkKEPcSFeV2FtpbAYrxuxuwTzzU40qyWdZhD+8WRZQd5+8E2A9f7pxQlO+v9d/0sS5ULaJ7fj95cooorQ5QooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigD/9k=)**

Figure Things Speak Result evidence-1 for week-7 Lab

**Discussion on results:**

We are using AD8232 heart rate monitor sensor to measure the heart rate which is connected to the Arduino. The data is transferred to the mobile app using Bluetooth module and it is connected to the Things speak sever and it is visualised in the MATLAB. We used phantom 320 ECG Simulator [References](#_References) to import the BMP values to our spark fun ad8232 board.

Unsigned long variables are extended size variables for number storage, and store 32 bits (4 bytes). Unlike standard longs unsigned longs won’t store negative numbers, making their range from 0 to 4,294,967,295 (2^32 - 1).

**Syntax:** unsigned long var = Val;

### Parameters for var is variable name. and Val is the value you assign to that variable.

**Formula:**

BPM = (1.0/Pulse Interval) \* 60.0 \* 1000;

This is the formula used for calculating beats for minute.

**SOURCE OF ERRORS:**

The errors occurred during the lab was very minor errors like MIT app could not read Varchar.

# References

1. <https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf>
2. <https://www.ms-gmbh.de/wp-content/doc/manuals/MS-GEB-phantom-320-1-8-EN-20210510.pdf>
3. [1]

|  |  |
| --- | --- |
| [1] | V. V. Rankovska and G. D. G. , “Interrupts in Teaching Microcontrollers Using Arduino,” *2020 XXIX International Scientific Conference Electronics (ET),* pp. 1-4, 2020. |
| [2] | T. A. a. M. A. R. S. M. Ahsanuzzaman, “ Low Cost, Portable ECG Monitoring and Alarming System Based on Deep Learning,” *2020 IEEE Region 10 Symposium (TENSYMP),* 2020. |
| [3] | S. S. a. N. N. A. ,. 2. J. Riyazulla Rahman, “Health Monitoring and Predicting System using Internet of Things & Machine Learning,",” *2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS),* 2021. |

**APPENDIX:**

**Week-5**

**Test plan 1 Code:**

#include <SPI.h> //library for SPI communications

#include <Mcp3208.h> //library for MCP3208

volatile int SPI\_CS=10; // define the pin for ChipSelect(CS)/SlaveSelect(SS) (Active LOW)

#define ADC\_VREF 5000 // define the ADC Vreference voltage

#define ADC\_CLK 16000000 // SPI clock 16.0 MHz

#define interruptPin = 2

MCP3208 adc (ADC\_VREF, SPI\_CS); //pass on the values of ADC\_VREF and SPI\_CS to adc function

void setup() // put your setup code here, to run once:

{

pinMode (SPI\_CS, OUTPUT); //set the CS/SS as OUTPUT

digitalWrite (SPI\_CS, HIGH); //set the CS/SS as HIGH initially so no spurious data transfer

Serial.begin(115200); //set baud rate for serial tranfser

SPISettings mysettings (2000000, MSBFIRST, SPI\_MODE0); // The SPISettings object is used to configure the SPI port for your SPI device with mySettting(Maximum\_device\_speed, dataOrder, dataMode)

SPI.begin(); //initialise SPI

SPI.beginTransaction(mysettings); //All 3 parameters from mysettings are combined to a single SPISettings object, which is given to SPI.beginTransaction().

attachInterrupt (0, resetButton, HIGH);

}

void loop() // put your main code here, to run repeatedly:

{

uint16\_t raw = adc.read(MCP3208::SINGLE\_2); //read the raw ADC on channel 2 in SINGLE ENDED MODE

uint16\_t val = adc.toAnalog(raw); //get analog value

Serial.print ("value: ");

Serial.print (raw);

Serial.print (" (");

Serial.print (val);

Serial.println (" mV)");

//delay(1000);

}

void resetButton ()

{

pinMode(SPI\_CS, LOW);

return;

}

**Test plan 2 Code:**

#include <SPI.h> //library for SPI communications

#include <Mcp3208.h> //library for MCP3208

#include <avr/io.h>

#include <avr/interrupt.h>

volatile int SPI\_CS=10; // define the pin for ChipSelect(CS)/SlaveSelect(SS) (Active LOW)

#define ADC\_VREF 5000 // define the ADC Vreference voltage

#define ADC\_CLK 16000000 // SPI clock 16.0 MHz

#define interruptPin 2

#define Samples 6000

//#define Test\_Pin 8

MCP3208 adc (ADC\_VREF, SPI\_CS); //pass on the values of ADC\_VREF and SPI\_CS to adc function

int read\_adc = Samples + 1;

void setup() // put your setup code here, to run once:

{

cli(); //disable all the interrupts

pinMode (interruptPin, INPUT\_PULLUP);

attachInterrupt (digitalPinToInterrupt(interruptPin), isr\_start, LOW);

//Timer Interrupt

TCCR1A = 0; // set entire TCCR1A register to 0

TCCR1B = 0; // set entire TCCR1B register to 0

TIMSK1 |= (1 << TOIE1); // enable timer overflow interrupt ISR

TCCR1B |=(1<<CS11); //Set bit CS12 in TCCR1

TCNT1 = 0xD8F0;

sei(); //enables all interrupts again

pinMode (SPI\_CS, OUTPUT); //set the CS/SS as OUTPUT

//pinMode (Test\_Pin, OUTPUT); //set the Test\_Pin as OUTPUT data transfer

digitalWrite (SPI\_CS, HIGH); //set the CS/SS as HIGH initially so no spuri

Serial.begin(115200); //set baud rate for serial tranfser

SPISettings mysettings (2000000, MSBFIRST, SPI\_MODE0); // The SPISettings object is used to configure the SPI port for your SPI device with mySettting(Maximum\_device\_speed, dataOrder, dataMode)

SPI.begin(); //initialise SPI

SPI.beginTransaction(mysettings); //All 3 parameters from mysettings are combined to a single SPISettings object, which is given to SPI.beginTransaction().

}

void loop() // put your main code here, to run repeatedly:

{

}

ISR(TIMER1\_OVF\_vect)

{

TCNT1 = 0xD8F0;

if (read\_adc < Samples)

{

uint16\_t raw = adc.read(MCP3208::SINGLE\_2); //read the raw ADC on channel 2 in SINGLE ENDED MODE

uint16\_t val = adc.toAnalog(raw); //get analog value

Serial.print ("value: ");

Serial.print (raw);

Serial.print (" (");

Serial.print (val);

Serial.println (" mV)");

read\_adc+=1;

}

}

void isr\_start()

{

read\_adc=0;

}

**Week-6**

**Test plan 1 Code:**

#include <avr/sleep.h> //Need this library for using the functions of the sleep mode

#include <avr/io.h> //Need this library for using the functions of the sleep mode

#define Interrupt\_Pin 2 // The Interrupt pin at location 2 is being utilised

void setup ()

{

Serial.begin (115200); //Open the serial port at 115200 bps

pinMode (13, OUTPUT); // Use the on-board LED at pin 13 as a way of ascertaining what state the Arduino is in

digitalWrite (13, LOW); // Drive the LED to LOW i.e. switched OFF

pinMode (Interrupt\_Pin, INPUT\_PULLUP); // Drive the interrupt pin 2 via a PULLUP state so that it does not gets toggled between LOW and HIGH states

digitalWrite (Interrupt\_Pin, HIGH); // Drive the interrupt pin 2 to a HIGH i.e. it sees 5 V available across it

}

void loop()

{

delay (5000); // Keep the LED HIGH for 5 seconds

sleepsetup(); // Call out the sleepsetup function (Defined later on) to put the Arduino to sleep

}

void sleepsetup()

{

sleep\_enable(); //Enable the sleep function (pre-defined function within the AVR/Sleep library

attachInterrupt(0,out\_of\_sleep,LOW); //Attach the interrupt to pin 2 which when driven LOW will call out function "out\_of\_sleep"

set\_sleep\_mode (SLEEP\_MODE\_PWR\_DOWN); //Set up the sleep mode (SLEEP\_MODE\_PWR\_DOWN) is the maximum power saving state

digitalWrite (13, LOW); //Write the LED pin to LOW

sleep\_cpu(); //Sets the sleep enable bit putting the ATMEGA to sleep

// The program stays here until the wake up call arrives

Serial.println("Awake now"); //When awaken by the interrupt, the program executes from this instruction and prints

digitalWrite (13, HIGH); //Turn ON the LED to indicate WAKEUP

}

void out\_of\_sleep() //Interrupt Service ROutine called out

{

sleep\_disable(); //disables the sleep mode

detachInterrupt(0); //disables the interrupr i.e. debounce the interrupt mechanism to prevent multiple interrupt calls.

}

**Test plan 2 code:**

#include <SPI.h> //library for SPI communications

#include <Mcp3208.h> //library for MCP3208

#include <avr/io.h>

#include <avr/interrupt.h>

#include <avr/sleep.h>

#include <SoftwareSerial.h>

volatile int SPI\_CS=10; // define the pin for ChipSelect(CS)/SlaveSelect(SS) (Active LOW)

#define ADC\_VREF 5000 // define the ADC Vreference voltage

#define ADC\_CLK 16000000 // SPI clock 16.0 MHz

#define interruptPin 2

#define Samples 6000

SoftwareSerial SerialBT (5, 6); //Rx, Tx

// RX is digital pin 5 (connect to TX of other device)

//TX is digital pin 6 (connect to RX of other device)

MCP3208 adc (ADC\_VREF, SPI\_CS); //pass on the values of ADC\_VREF and SPI\_CS to adc function

int read\_adc = Samples + 1;

void setup() // put your setup code here, to run once:

{

cli(); //disable all the interrupts

pinMode (interruptPin, INPUT\_PULLUP);

attachInterrupt (digitalPinToInterrupt(interruptPin), isr\_start, LOW);

//Timer Interrupt

TCCR1A = 0; // set entire TCCR1A register to 0

TCCR1B = 0; // set entire TCCR1B register to 0

TIMSK1 |= (1 << TOIE1); // enable timer overflow interrupt ISR

TCCR1B |=(1<<CS11); //Set bit CS12 in TCCR1

TCNT1 = 0xD8F0;

sei(); //enables all interrupts again

pinMode (SPI\_CS, OUTPUT); //set the CS/SS as OUTPUT

//pinMode (Test\_Pin, OUTPUT); //set the Test\_Pin as OUTPUT data transfer

digitalWrite (SPI\_CS, HIGH); //set the CS/SS as HIGH initially so no spuri

Serial.begin(115200); //set baud rate for serial tranfser

SPISettings mysettings (2000000, MSBFIRST, SPI\_MODE0); // The SPISettings object is used to configure the SPI port for your SPI device with mySettting(Maximum\_device\_speed, dataOrder, dataMode)

SPI.begin(); //initialise SPI

SPI.beginTransaction(mysettings); //All 3 parameters from mysettings are combined to a single SPISettings object, which is given to SPI.beginTransaction().

SerialBT.begin (115200);

sleep\_enable();

set\_sleep\_mode (SLEEP\_MODE\_PWR\_DOWN);

sleep\_cpu();

}

void loop() // put your main code here, to run repeatedly:

{

}

ISR(TIMER1\_OVF\_vect)

{

TCNT1 = 0xD8F0;

if (read\_adc < Samples)

{

uint16\_t raw = adc.read(MCP3208::SINGLE\_2); //read the raw ADC on channel 2 in SINGLE ENDED MODE

uint16\_t val = adc.toAnalog(raw); //get analog value

val = map((val), 0, 4095.0, 0, 255);

// while (Serial.available()>0)

// {

// Serial.print(val);

// Serial.write(SerialBT.read());

// Serial.write("\n");

// SerialBT.write(SerialBT.read());

// SerialBT.write("\n");

// }

Serial.print ("value: ");

Serial.print (raw);

Serial.print (" (");

Serial.print (val);

Serial.println (" mV)");

SerialBT.write((byte)val);

//SerialBT.write('/n');

read\_adc+=1;

} else

{

sleep\_cpu();

}

}

void isr\_start()

{

sleep\_disable();

read\_adc=0;

//detachInterrupt(0); //do not use the detachInterrupt as it does not allow the interrupt to work

}

**Week-7**

**Test plan 1 code:**

int UpperThreshold = 518;

int LowerThreshold = 490;

int reading = 0;

float BPM = 0.0;

bool IgnoreReading = false;

bool FirstPulseDetected = false;

unsigned long FirstPulseTime = 0;

unsigned long SecondPulseTime = 0;

unsigned long PulseInterval = 0;

void setup()

{

Serial.begin(9600);

}

void loop(){

reading = analogRead(0);

// Heart beat leading edge detected.

if(reading > UpperThreshold && IgnoreReading == false)

{

if(FirstPulseDetected == false)

{

FirstPulseTime = millis();

FirstPulseDetected = true;

}

else

{

SecondPulseTime = millis();

PulseInterval = SecondPulseTime - FirstPulseTime;

FirstPulseTime = SecondPulseTime;

}

IgnoreReading = true;

}

// Heart beat trailing edge detected.

if(reading < LowerThreshold)

{

IgnoreReading = false;

}

BPM = (1.0/PulseInterval) \* 60.0 \* 1000;

Serial.print(reading);

Serial.print("\t");

//Serial.print(PulseInterval);

//Serial.print("\t");

Serial.print(BPM);

Serial.println(" BPM");

//Serial.flush();

// Please don't use delay() - this is just for testing purposes.

//delay(50);

}