Applied Electrical and Electronic Engineering: Construction Project (EEEE1039 UNNC)

Project Session I and Project Session II Individual Report

Student Name: Guan-Sheng Chen

Student ID: 20314386

Group: QY-19

Affiliation: Department of Electrical & Electronic Engineering,

University of Nottingham Ningbo China

● Abstract

In the session 1, the H bridge was used as a motor controller by connecting the motor and switches. Through H bridge and Arduino to complete the basics of building and testing the vehicle. In the session 2, the goal of the session was to establish serial communications between two microcontrollers, HMI (Human-Machine Interface) with Arduino, and real-time (RT) system operation. To further control the vehicle, relevant knowledge was used and the device in the topic of HMI with Arduino were connected. The sinewave generator was created for the system's real-time operation. The devices in this topic were used to connect to the vehicle in order to control it further through HMI with Arduino. Two Arduino nano boards were connected to transmit and receive messages from one microcontroller to the other for serial communication. Through h bridge and Arduino to complete the basics of building and testing the vehicle.

● Introduction

The H bridge project and the use of Arduino were the two major parts of session 1 of the basic knowledge of building and testing vehicles. For the first part, the H bridge as a motor controller was consisted of four switches and a motor, and the polarity of a voltage applied to a load could be switched by the circuit. The direction of rotation could be controlled by switching between switches; therefore, the rotation movement of the vehicle could be controlled by this circuit. In the second part, the Arduino nano board was an encoder used to execute the entered code to help the vehicle complete instructions. The code was input by Arduino IDE and saved on the Arduino nano board by connecting it to the computer with a transmission line.

In the session 2, four segments were included. Serial communications between two microcontrollers (Arduino), Human-Machine Interface (HMI) with Arduino, system operation in real time, and vehicle construction. For serial communications between two microcontrollers (UART), two Arduino nano boards were connected in serial part, and one was master and the other was the slave. Data and variables could be transmitted through two Arduino boards (connection of RX and TX). For HMI with Arduino, the system and the vehicle were consisted of six devices. The speed and the status of the task were set by the encoder. The system status would be displayed on the LCD (Liquid Crystal Display) screen. The success of the implementation of the task was indicated by the LED and the buzzer. The direction of the vehicle was controlled by the slide switch. The execution of the task is initiated by the button switch or the microphone. For real-time control, the frequency was sent by the serial monitor, and then the frequency would be controlled by RT, and then through the amplifying circuit and filter circuit to generate a sine wave. For a low battery voltage indicator, when the voltage was lower than about 5 V, the current would be from the positive electrode to the negative electrode of the LED, so the LED light would light up, when the voltage was higher than 6 V, the current would directly pass through the zenor diode, so that the LED light would not light up.

- **Method**

Session 1:

1. H bridge as a motor controller

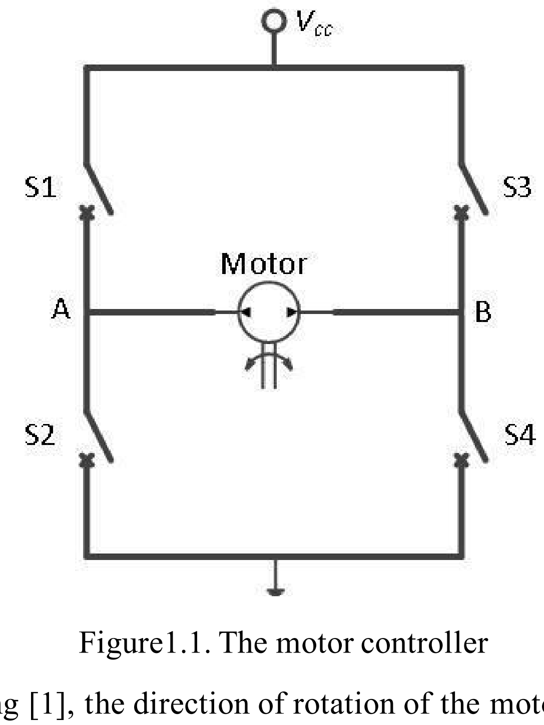


Figure1.1. The motor controller

According to Jing [1], the direction of rotation of the motor controller could be clockwise or counterclockwise by switching the switches. The voltage sources of 4 V and 8 V was chosen to determine whether the direction of motor rotation is affected by voltage. Under different voltages, three different situations were tested. The first was to close all switches. The second was to open the switches of S1 and S4. The third was to open the switches of S2 and S3.

Session 2:

1. Serial communications between 2 microcontrollers (Arduino)

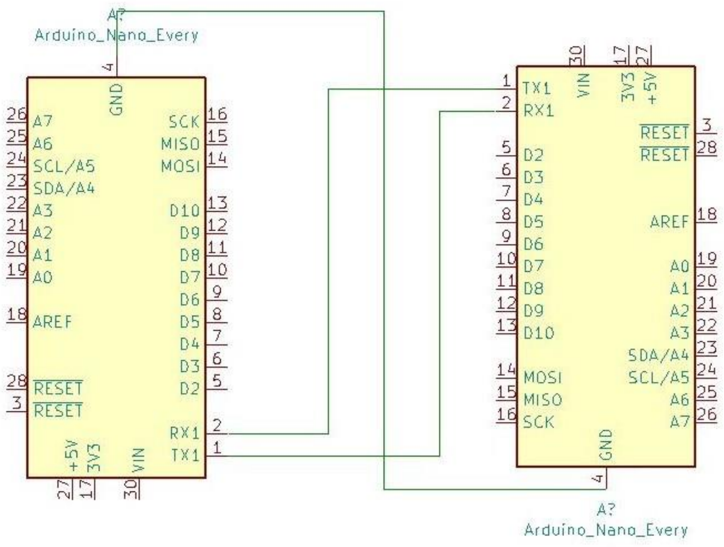


Figure1.2. Connection diagram

The Arduino board in Figure 1.2 on the left was the master, and the Arduino board on the right was the slave. The connection of the ground was used to protect the Arduino board. For the code in the Arduino board, “Hello” was defined in the master board, so “Hello” was expected to the output to the slave board.

2. HMI with Arduino

    (a.) Button switch

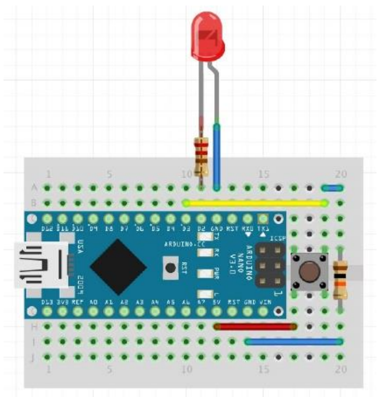


Figure1.3. Button switch

In Figure 1.3, the button switch was unpressed, and there was no connection between the two legs of the button switch, therefore the pin was connected to ground. When the button was pressed, two legs were linked to the pin, which was connected to 5 V.

(b.) Slide switch

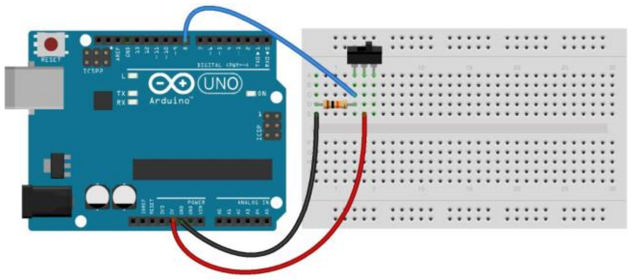


Figure1.4. Slide switch

In Figure 1.4, the resistor of 220 Ω was chosen to limit the current. One terminal of the slide switch was connected to pin 8, and the middle terminal was connected the voltage supply. One terminal of the resistor was connected to the ground.

(c.) Potentiometer

This device has not been tried.

(d.) Electret microphone

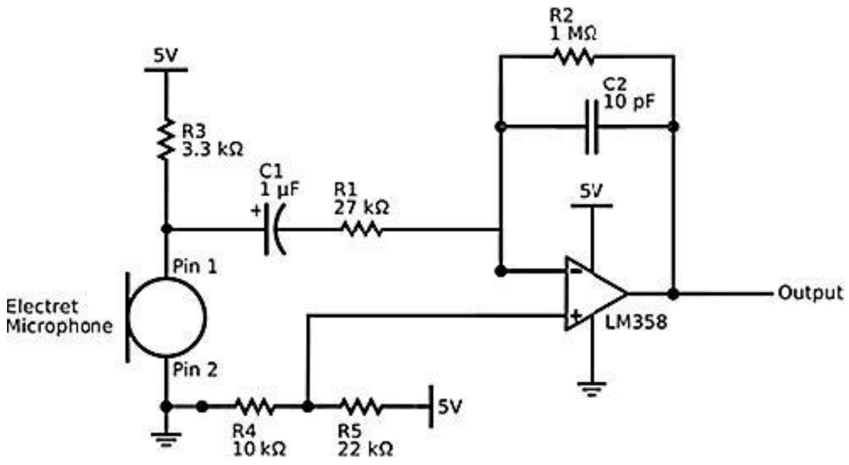


Figure1.5. Electret microphone

The electret microphone was the other option to start the task. In Figure 1.5, an LM358 operational amplifier was used to send the output of an electret microphone through a band-pass amplifier. The circuit was worked in 5 V. A high-pass (R1 and C1) and a low-pass (R2 and C2) component (R2 and C2) were in the op-amp circuit. The op-amp worked in the same way as an inverting amplifier. The output was provided by a voltage divider (R4 and R5). Both the positive and negative components of the input were amplified and presented in the output.

(e.) Encoder

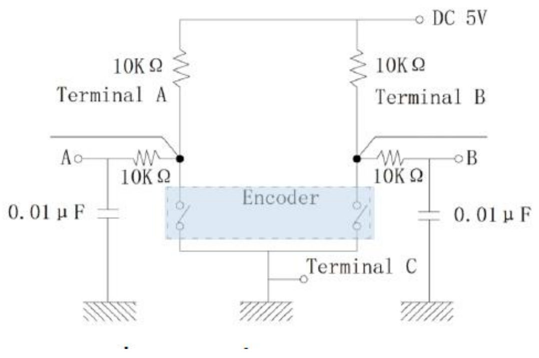


Figure1.6. Encoder

According to [2], the encoder was used to generate an electrical signal, either analog or digital, according to the rotational movement to select the task of the vehicle and set the status of the system. The schematic was in Figure 1.6.

(f.) LED

The voltage supply was linked to one terminal of the resistor. The LED was connected to the other terminal of the resistor. The positive voltage was applied to the LED's cathode. The other terminal of the LED was attached to the power supply's negative terminal. The Arduino microcontroller was used to provide the voltage.

(g.) LCD1602

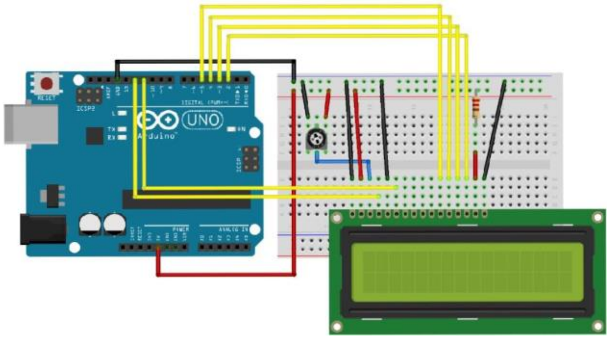


Figure1.7. LCD1602

The message was displayed on the LCD screen. The message displayed on the LCD was determined by the codes in the Arduino nano board. The resistor of 220 Ω was used to limit the current.

(h.) Passive buzzer

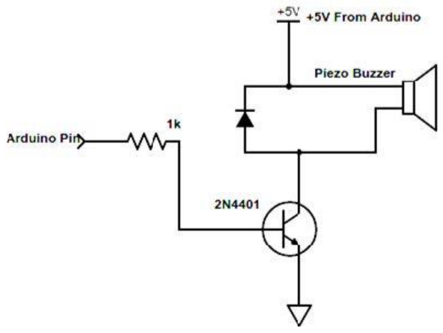


Figure1.8. Passive buzzer

In Figure 1.8, the 5V output of the Arduino microcontroller was connected to the positive pole of the buzzer and connected to the cathode of the diode. The negative pole of the buzzer was connected to the anode side of the diode. R1 was set to 1 kΩ to limit the current through the NPN transistor, when the digital output was high. The fast-recovery diode was added to provide a path for the passive buzzer's kickback current.

(i.) Active buzzer

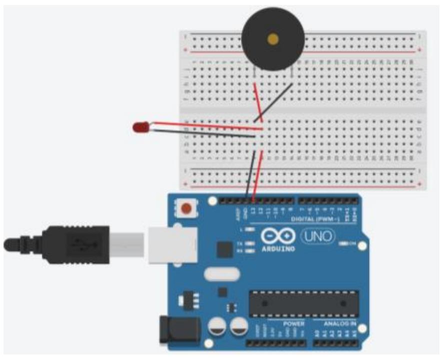


Figure1.9. Active buzzer

In Figure1.9, the LED positive terminal was connected to pin 13, and the negative terminal was connected to the ground.

3. System operation in real time (RT)

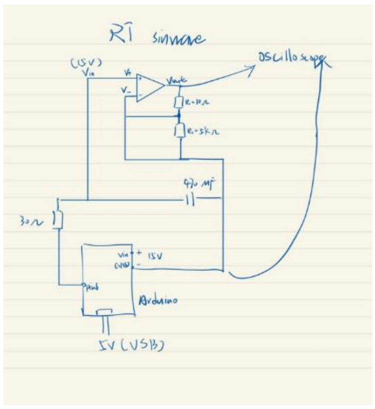


Figure1.10. RT controlled sinewave generator

The schematic of RT controlled sinewave generator was in Figure1.10. The frequency was entered from serial monitor. The resistor of R1 was 10 kΩ, and the resistor of R2 was 5 kΩ.

4. Low battery voltage indicator

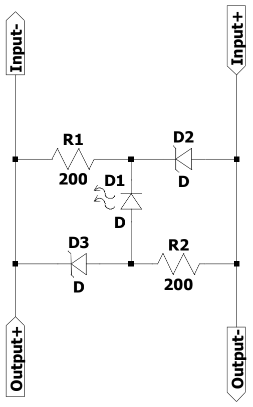


Figure1.11. Low battery voltage indicator

In Figure 1.11, the fuse was added as a protection, and the zenor diodes of 3 V were chosen to prevent low voltage current from passing. The resistors of R1 and R2 were chosen. The fuse was added after Input+, but the fuse could not be found in LTspice, there was no fuse in Figure 1.11.

5. Vehicle building and challenges

   The topic was consisted of three tasks. The first one was spin, the second was running one-meter squares, and the third was running forward and backward. Due to time constraints, this part has not been completed.

● Results

Session 1:

1. H bridge as a motor controller

   The results were in Table 1. The direction of rotation was not affected by the voltage, but the direction of rotation was affected by turning on or off different switches. S1 and S4 were turned on, current flowed through the motor, and the motor rotated in a clockwise direction. S2 and S3 were turned on, current flowed through the motor, and the motor rotated in a counterclockwise direction.

Table 1.

Vcc (V)	S1	S2	S3	S4	Direction of Rotation
4	off	off	off	off	N/A
4	on	off	off	on	Clockwise
4	off	on	on	off	Counterclockwise
8	off	off	off	off	N/A
8	on	off	off	on	Clockwise
8	off	on	on	off	Counterclockwise

Session 2:

1. Serial communications between 2 microcontrollers (Arduino)

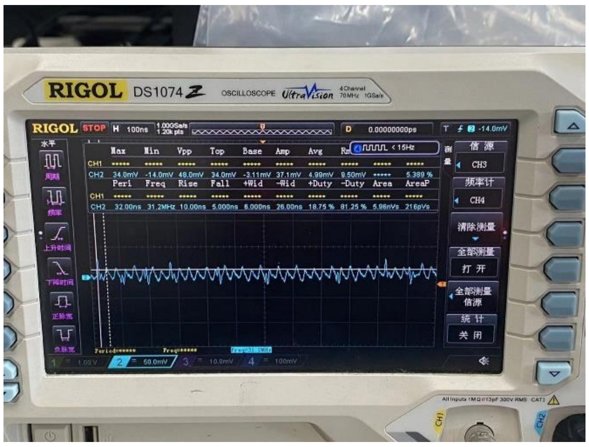


Figure2.1. The waveform of the master board

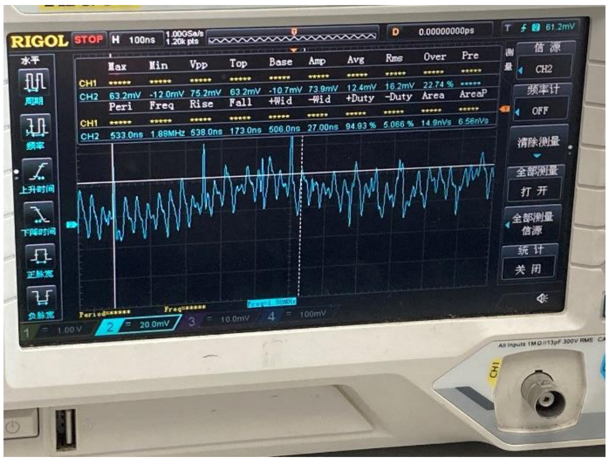


Figure2.2. The waveform of the slave board

The waveform of the master board and the slave board were recorded in Figure 2.1 and Figure 2.2 respectively.

2. HMI with Arduino

(a.) Button switch

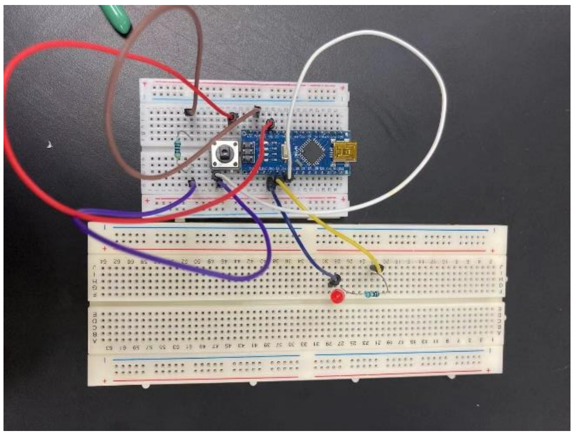


Figure2.3. The button switch circuit

The button was pressed, two legs were attached to the pin connected to 5 V, and the LED lit up. Therefore, the result was consistent with the expectations.

(b.) Slide switch

The current flowed because the slide switch was turned on.

(c.) Potentiometer

This device has not been tried.

(d.) Electret microphone

The circuit was correctly connected; however, the code was failed to be executed.

(e.) Encoder

By spinning the encoder, the serial monitor's message was changed.

(f.) LED

The LED was lighted up.

(g.) LCD1602

The message “Hello World” and the number of rows and columns were displayed on the LCD screen in Figure1.7. The output matches the Arduino code that was provided.

(h.) Passive buzzer

Different tones were emitted by the passive buzzer with different frequencies.

(i.) Active buzzer

The sound was emitted from the active buzzer, but the tone could not be changed.

3. System operation in real time (RT)

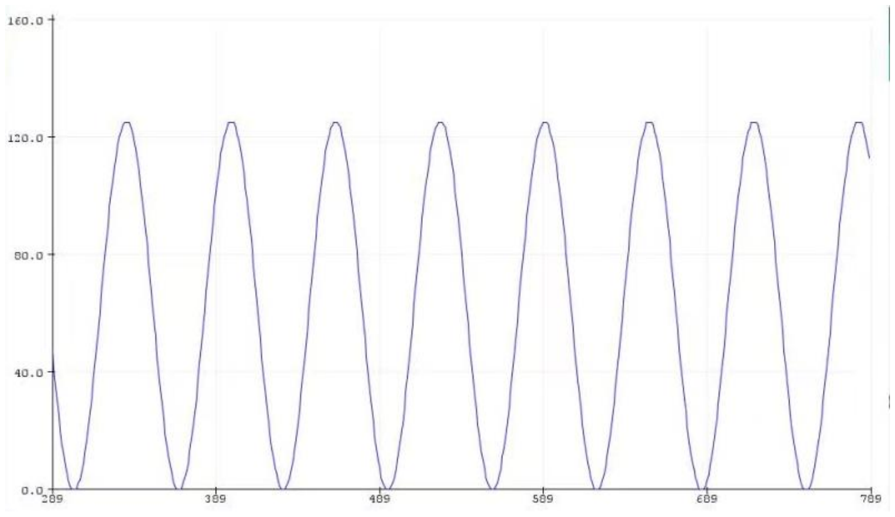


Figure2.4. Sine wave waveform in Arduino serial monitor

In Figure2.4, the vertical axis was in milliseconds and the horizontal axis was voltage. The constant peak-to-peak voltage output was 9 V, and the DC power supply voltage was adjusted to 15 V. The Arduino serial monitor delivered a basic frequency range of 1 Hz to 4 Hz (in 0.2 Hz increments).

4. Low battery voltage indicator

   For the circuit in Figure1.11, under the situation that the voltage was lower than 5V, the zenor diode would not be passed by the current, and the current would flow from R2 to the LED and then to R1. On the contrary, the voltage was higher than 5V, the current would flow through the zenor diode and then to R1. This circuit worked with a current of less than 0.1 amps and a low-voltage threshold of 5 volts.

5. Vehicle building and challenges

   This section has not been completed due to time constraints.

- **Discussions**

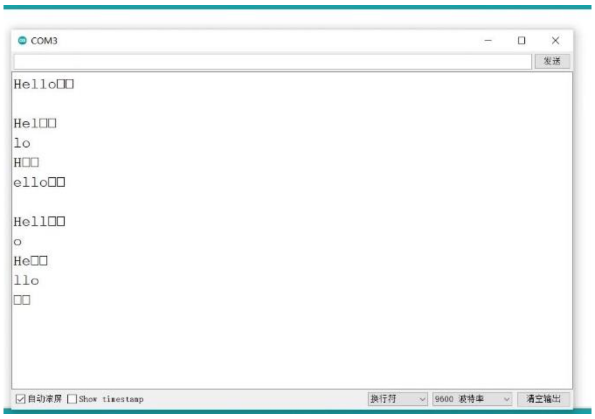
Session 1:

1. H bridge as a motor controller

   According to the data in Table 1, the results were consistent with the expectations.

Session 2:

1. Serial communications between 2 microcontrollers (Arduino)



```
Hello□□

Hel□□
lo
H□□
ello□□

Hell□□
o
He□□
llo
□□
```

Figure3.1. The output on the slave board

The output from serial port was displayed in Figure 3.1, which can prove that there is serial communication between the master board and the slave board.

2. HMI with Arduino

(a.) Button switch

The LED was lighted up when the button was pressed. Therefore, the result was consistent with the expectations.

(b.) Slide switch

Because the slide switch was switched on, current flowed. Therefore, the result was consistent with the expectations.

(c.) Potentiometer

The time was not effectively allotted, and there was no time to accomplish the task. The well preparation should be done in advance before the next experiment.

(d.) Electret microphone

The circuit in Figure1.7 was worked, and the requirements were met.

(e.) Encoder

The serial monitor's message was altered by spinning the encoder. The outcomes were in line with expectations.

(f.) LED

The LED was lighted up. Thus, the result was consistent with expectations.

(g.) LCD1602

The message "Hello world" was successfully displayed on the LCD screen.

The results meet with the requirement.

(h.) Passive buzzer

The passive buzzer emitted different tones with different frequencies. The expectations were meet by the results.

(i.) Active buzzer

The results of the active buzzer were consistent with the requirement.

3. System operation in real time (RT)

   The waveform in Figure 2.4 was consistent with the requirement.

4. Low battery voltage indicator

   The LED was not lighted up when the voltage was higher than 5 V, and it was lighted up when the voltage was lower than 5 V. The results were consistent with the requirement.

● Conclusions

In the session 1, because of the special situation, only the H bridge as a motor controller was completed in practice, and the other tasks were completed on the simulator. For the h bridge as a motor controller, the current path was modified by switching between switches, which changed the direction of motor spinning.

In the session 2, three major topics were completed. For the serial communications between two microcontrollers, the transmission and reception of messages between the master and slave boards has been finished. For HMI with Arduino, eight tasks were completed, and one task was not finished. The majority of the equipment necessary to connect to the vehicle have been built, with the exception of the potentiometer. To complete more tasks, further research was required to improve task completion efficiency and gained a better understanding of the tasks. For system operation in real time (RT), the sinewave generator was constructed, although the process was time-consuming. The reason for this was that the Arduino code was incorrect, and correcting it took a long time. To become conversant with Arduino programming, more research was required. Because of the time constraints, the final tasks were not completed. Time management and thorough task preparation were required.

- **References**

- [1] Jing W. H bridge project. 2021,9. [Accessed: November 25, 2021]
- [2] HowToMechatronics. How Rotary Encoder Works and How To Use It with Arduino. 2021, 10. Available: https://www.citationmachine.net/ieee/cite-a-website/confirm [Accessed: November 24, 2021]

- **Appendices**

Session 2:

1. Serial communications between 2 microcontrollers (Arduino)

(a.) The master board

```
char nb=[5]='hello';
void setup() {
  Serial.begin(9600);
}

void loop() {
    Serial.println(nb);
    delay(1000);
}
```

(b.) The slave board

```
char nb=[5];

void setup() {
    Serial.begin(9600);
}

void loop() {
    Serial.readBytes(nb,5);
    Serial.println(nb);
    delay(1000);
}
```

2. HMI with Arduino

   (a.) Button switch

```
#include <PinChangeInterrupt.h>

const byte ledPin = 13;
const byte Button_Switch = 10;
volatile byte state = LOW;

void setup() {
  pinMode(ledPin, OUTPUT);
  pinMode(Button_Switch, INPUT_PULLUP);
  attachPinChangeInterrupt(digitalPinToPCINT(Button_Switch),
blink, RISING);
}

void loop() {
  digitalWrite(ledPin, state);
}

void blink() {
  state = !state;
}
```

   (b.) Slide switch

```
const byte ledPin = 13;
const byte Slide_Switch = 2;
volatile byte state = LOW;
```

```
void setup() {
  pinMode(ledPin, OUTPUT);
  pinMode(Slide_Switch, INPUT_PULLUP);
  attachInterrupt(digitalPinToInterrupt(Slide_Switch), blink,
CHANGE);
}

void loop() {
  digitalWrite(ledPin, state);
}

void blink() {
  state = !state;
}
```

(c.) Electret microphone

```
int ledPin=13;
int Microphone = A0;

void setup()
{
  pinMode(ledPin, OUTPUT);
  pinMode(Microphone, INPUT);
  Serial.begin(57600);
}

void loop()
{
    long sum = 0;
    for(int i=0; i<100; i++)
```

```
        {
                sum += analogRead(Microphone);
        }

        sum = sum/100;

        Serial.println(sum);

    if (sum>=200){
        digitalWrite(ledPin, HIGH);
    }
    else {
        digitalWrite(ledPin, LOW);
    }

    delay(100);
}
```

(d.) Encoder

```
#define encoder0PinA 2
#define encoder0PinB 11
#define encoder0Btn 3
int 0Pos = 0;
void setup() {
    Serial.begin(9600);
    pinMode(2, INPUT_PULLUP);
    pinMode(11, INPUT_PULLUP);
    pinMode(3, INPUT_PULLUP);
    attachInterrupt(digitalPinToInterrupt(2),
doEncoder, CHANGE);
}
```

```
int Rot,lastRot;

bool btn = false;

void loop() {
   Serial.print(btn);
   Serial.print(" ");
   Serial.print(Rot);
   if(Rot>lastRot){

   Serial.print("    XY");

   }
   else{

      Serial.print("    XXY");

   }

   lastRot = Rot;

   Serial.println(" ");

   delay(250);
}
void Encoder()
{
   if (digitalRead(2) == digitalRead(11)){
   0Pos--;

   }

   else{

   0Pos++;

   }
   valRotary = encoder0Pos/2;
}

void botton(void)
{
   btn = true;
}
```

(e.) LED

```
const byte ledPin = 13;
const byte Slide_Switch = 2;
volatile byte state = LOW;

void setup() {
    pinMode(ledPin, OUTPUT);
    pinMode(Slide_Switch, INPUT_PULLUP);
    attachInterrupt(digitalPinToInterrupt(Slide_Switch),          blink,
CHANGE);
}

void loop() {
  digitalWrite(ledPin, state);
}

void blink() {
  state = !state;
}
```

(f.) LCD1602

```
/*
  LiquidCrystal Library - Hello World

 Demonstrates the use a 16x2 LCD display.  The LiquidCrystal
 library works with all LCD displays that are compatible with the
 Hitachi HD44780 driver. There are many of them out there, and
you
 can usually tell them by the 16-pin interface.

 This sketch prints "Hello World!" to the LCD
 and shows the time.

  The circuit:
 * LCD RS pin to digital pin 17
 * LCD Enable pin to digital pin 16
 * LCD D4 pin to digital pin 4
 * LCD D5 pin to digital pin 6
 * LCD D6 pin to digital pin 7
 * LCD D7 pin to digital pin 8
 * LCD R/W pin to ground
 * LCD VSS pin to ground
 * LCD VCC pin to 5V
 * 10K resistor:
```

```
 * ends to +5V and ground
 * wiper to LCD VO pin (pin 3)

*/

// include the library code:
#include <LiquidCrystal.h>

// initialize the library by associating any needed LCD interface pin
// with the arduino pin number it is connected to
const int rs = 17, en = 16, d4 = 4, d5 = 6, d6 = 7, d7 = 8;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

void setup() {
  // set up the LCD's number of columns and rows:
  lcd.begin(16, 2);
  // Print a message to the LCD.
  lcd.print("hello, world!");
}

void loop() {
  // set the cursor to column 0, line 1
  // (note: line 1 is the second row, since counting begins with 0):
  lcd.setCursor(0, 1);
  // print the number of seconds since reset:
  lcd.print(millis() / 1000);
}
```

(g.) Passive buzzer

```
int Passive_buzzer = 10;
int LED_pin = 13;
```

```
void setup() {
  pinMode(Passive_buzzer, OUTPUT);
  pinMode(LED_pin, OUTPUT);
}

void loop() {
    unsigned int i=0;
    for(i=0;i<=1000;i++)
    {
        digitalWrite(Passive_buzzer, HIGH);
        delayMicroseconds(500);
        digitalWrite(Passive_buzzer, LOW);
        delayMicroseconds(500);
    }
    digitalWrite(Passive_buzzer, LOW);
    delay(500);
    digitalWrite(LED_pin, HIGH);
    delay(500);
    digitalWrite(LED_pin, LOW);
}
```

(h.) Active buzzer

```
int Active_buzzer = 10;
int LED_pin = 13;

void setup() {
 pinMode(Active_buzzer, OUTPUT);
 pinMode(LED_pin, OUTPUT);
}
```

```
void loop() {
    digitalWrite(Active_buzzer, HIGH);
    digitalWrite(LED_pin, HIGH);
    delay(500);
    digitalWrite(Active_buzzer, LOW);
    digitalWrite(LED_pin, LOW);
    delay(500);
}
```

3. System operation in real time (RT)

```
#include <MsTimer2.h>
#define FREQ_CTL (1000)
float f=3.0;
float t=0;

void setup( )
{

    noInterrupts();
    Serial.begin(9600);    //connect to PC
    User_init_timer2();
    pinMode(6, OUTPUT);
    interrupts();
    delay(1000);
    Serial.println("Enter freq:");
    f=(float)Serial.read();
    delay(1000) ;


}
void loop()
```

```
{

}

inline void User_init_timer2()
{
   MsTimer2::set((1000/FREQ_CTL),Timer2ISR);
   MsTimer2::start();
   }


void Timer2ISR(){

   float v;
   v= (int)(90.*sin(2.*3.1415926*f*t)+127.);
   analogWrite(6, v);
   //Serial.println(v);
   if(t<1) t=t+0.001;
   else t=0.0;
   }
```