Department of Electrical & Electronic Engineering

University of Nottingham Ningbo China

EEE1039 Report1:

Basics of Building and Testing the Vehicle

&

Real-Time Control and Human Machine Interface

Shun Yao

20321234

ABSTRACT:

This experiment aims to study basis of electronic components and apply Arduino programme to a self-build vehicle that could be controlled and interacted with HMI. The hardware base and some related knowledge were studied before programming.

As a result, two vehicles were built, H-bridge and three serial communications were learned; several HMI, a RT-control sinewave generator, a low battery indicator were designed and successfully built. The programmed codes could drive the vehicle run 10m straight line, go around a 1m square, spin one cycle. The HMI on the vehicle could allow user to select the task, direction, set the speed, display running state; the buzzer could play different sound and LED could blink in different frequency when the vehicle is in different running state.

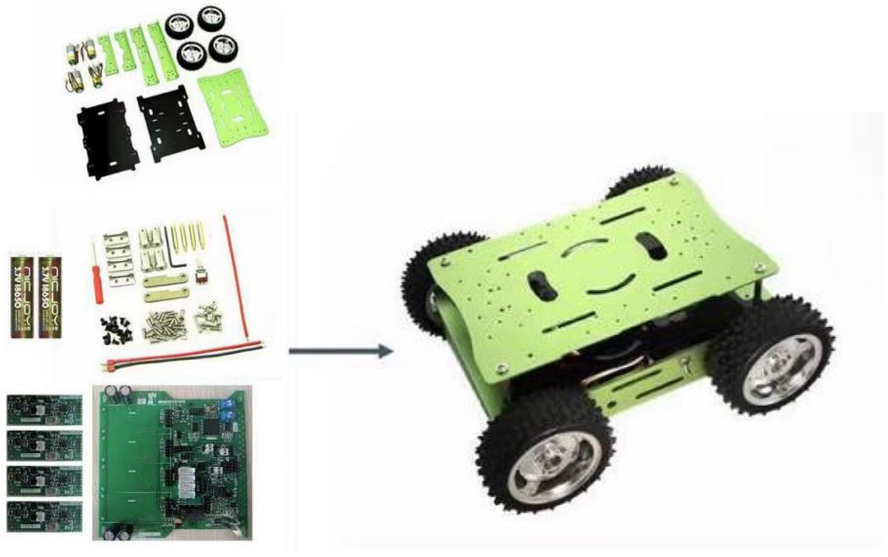
In conclusion, basis of electronic hardware and Arduino programming were studied in this experiment. Most of experiment was working properly, though the last challenge of programming the vehicle running strictly controlled by HMIs on it was not fully finished.

INTRODUCTION:

Task1: Vehicle assemble and H-bridge building

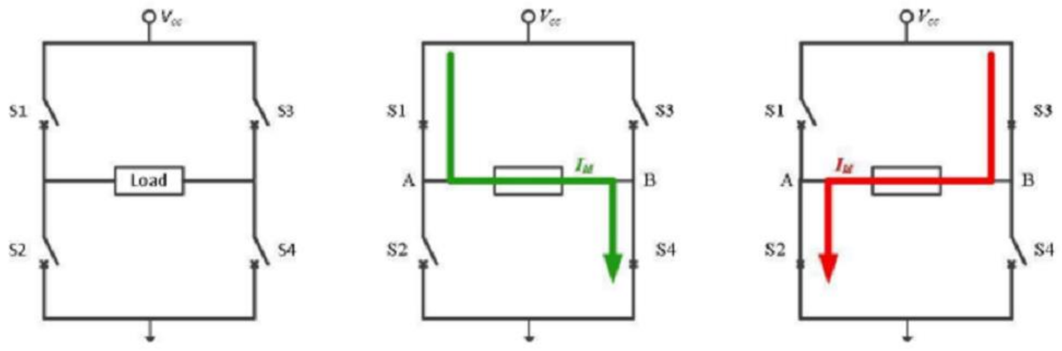
The first session was based on hardware of vehicle, check all the components, and assemble those together. Some of the circuit board needs to be soldered.

Figure 1. The components and an example vehicle (Chunyang, Jing 2021)



H-bridge is one of the control systems which could be used on motor to choose the direction of it. An H-bridge system was built on the breadboard which is easy to rebuild or debug. The basis of breadboard connection and the schematic diagram of H-bridge should be fully understood to finish building of H-bridge either with button switch or slide switch.

Figure 2. The H-bridge: motor controller schematic diagram



Task2: Programming for the vehicle by Arduino Nano

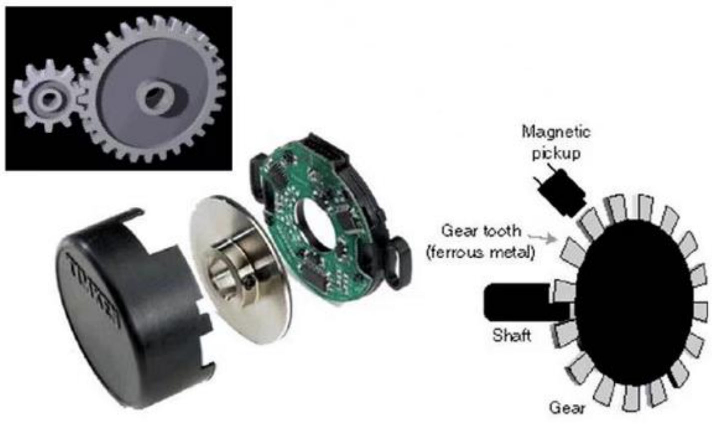
Three assignments need to be completed:

a. Run exactly 10m and automatically stop with one source code
b. Spin the vehicle for exactly one cycle and automatically stop
c. Run around a 1m square and automatically stop at the start position

The original codes from Moodle page should be understood by advance self-study in order to change some value or logic instruction that make the code suit in the assembled vehicle who has its own mechanic characteristic.

Basically, the encoders return the rotation angles of individual motor shafts as digital signals, which are sent to the processor. After conversion based on gear ratio and wheel radius, the distance travelled can be calculated (Juang and Lum 2013). In other word, the distance of vehicle travelled should counted by the encoder within each motor, and the value of encoders was sent back to Arduino board in real time which makes the vehicle could stop automatically once the encoder value reached the set number.
The instruction "delay()" was used as one of the main instructions in assignment c.

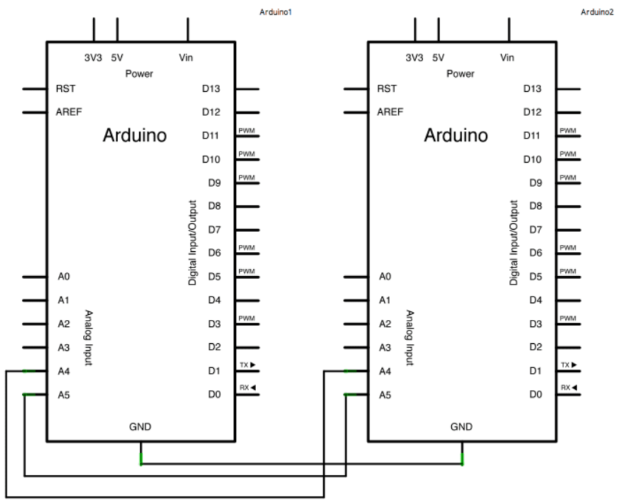
Figure 3. The encoder (Chunyang, Jing 2021)



Task3: Establish serial communication between two Arduino and design HMI

Serial communication is the process of sending data one bit at a time, sequentially, over a communication channel or computer bus. Three general serial communication protocols were studied here: UART, I^2C and SPI. Using the example codes from Arduino official website, a master writer and slaver receiver circuit was completed between two Arduino board.

Figure4. Schematic diagram of serial communication between two Arduino



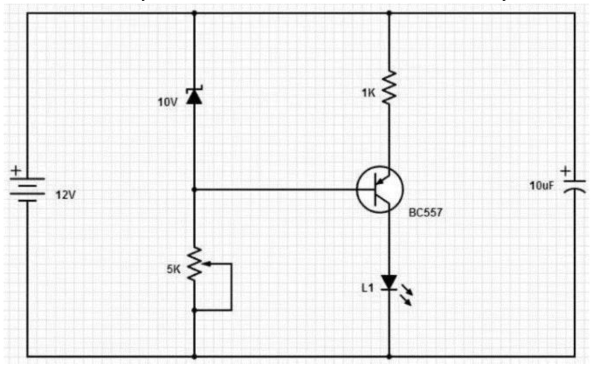
human–machine interface (HMI) that interfaces machines with physical input hardware such as keyboards, mice, or game pads, and output hardware such as computer monitors, speakers, and printers (Wikipedia 2021). In this session, the button switch was designed on a circuit to control a LED light on or off and the slide switch was used to control a LCD1062 display. The HMI with a microphone and a buzzer was more complex as it requires amplifier and capacitor in the circuit, which takes more effects.

Task4: Real-Time control system and low battery indicator

A RT controlled sinewave voltage generator was designed and verified on an Arduino base circuit whose frequency could be changed to the value sent by Serial Monitor from 1 Hz to 4 Hz. The code written on Arduino Nano and the circuit was build on breadboard. The circuit should include an amplifier and a low pass filter to remove the influence of noise.

A low battery indicator was designed, soldered, and verified individually. The LED on the indicator will light on if the voltage across it higher than 6V and light off if the voltage is lower than 6V. Current on it was controlled under 0.1A by adding resistor.

Figure5. Example of a 12V lead low battery indicator



Task5. Programming one source code to finish 2 of 3 tasks based on Arduino Nano

Assignments:

A. Spin + [sound alert + LED blinking (both solid ON)]
B. Run 1m squares + [sound alert + LED blinking (both 0.5s ON - 0.5s OFF)]
C. Run forward or backward [sound alert + LED blinking (both 1s ON - 1s OFF)]

Requirements:

1. Task selection and start by press button(s) or a microphone
2. Direction setting by a slide switch: +/- (Task A/B: clockwise/anticlockwise; Task C: forward/backward)
3. Speed setting (50% to 80%, incr 1%) and target setting (Task A: number of cycles; Task B/C: 2m to 4m, incr 0.1m) by an encoder
4. System status display by an LCD: Task #, speed set, target set, direction set, Status: Running or Stopped
5. System status indication by an LED: Running Task # or stopped
6. System status indication by a buzzer: Running Task # or stopped
7. Try to play different tunes or frequencies of beeps, or different music

Assignment A and C was chosen. Code related to buzzer, LCD, encoder, and other components need to combine together to fulfil the requirements, which need an effective teamwork.

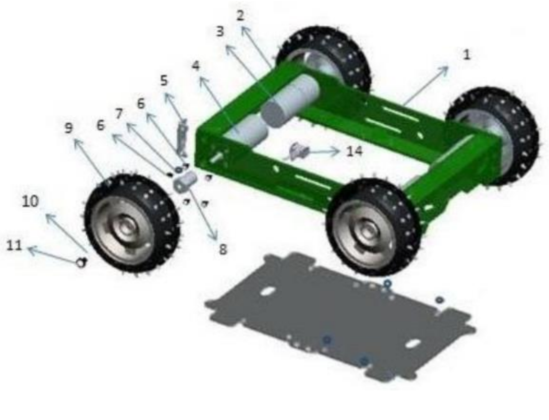
METHODS:

Task1: Vehicle assemble and H-bridge building

Before vehicle assemble starts, the number and the correction of components were checked with a given table to make sure the basis of assemble work be confirmed. The soldering work was done after the frame of vehicle finished and the circuit was carefully checked, because it is hard to debug a soldered circuit.

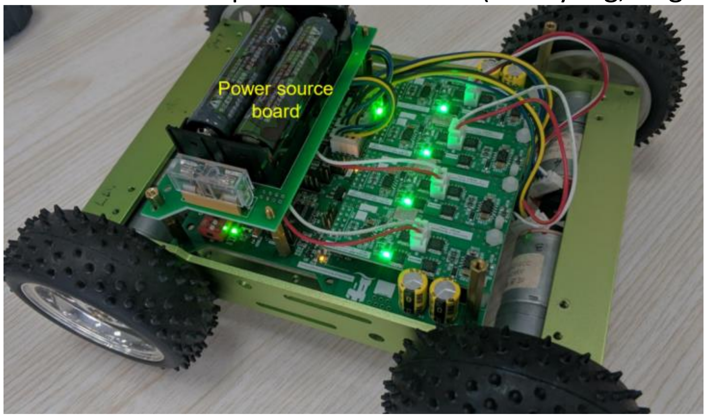
The given materials are sufficient to build two vehicles. The first vehicle was built from four steel plate which form a rectangle as vehicle's frame. Four motors were added into the frame using screw, then four wheels fixed to those four motors. After the frame combined, an acrylic plate was fix on it by four nylon hex pillars to provide a suitable environment for baseboard build.

Figure6. Vehicle frame (Chunyang, Jing 2021)



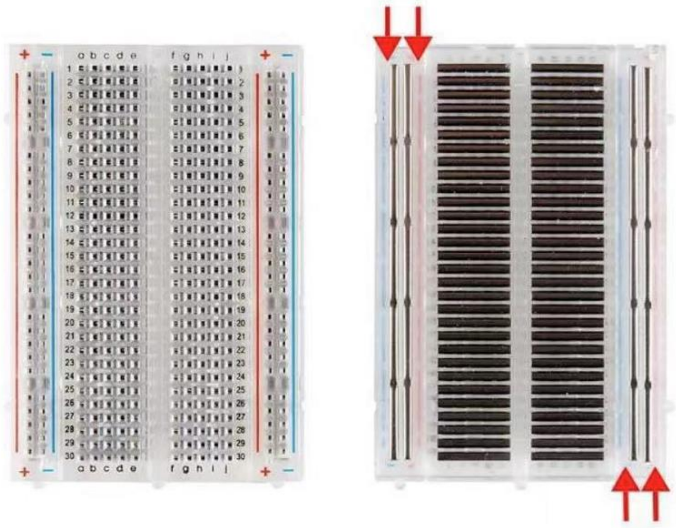
The provided H-bridge board and baseboard could be added on the acrylic plate directly and connected with four motors. However, the power source board need to be design and solder before assembling. This board should combine a switch, a fuse, a wiring port, a LED power indicator, and a battery tray. Soldering started once the circuit design completed, but the misconnection of the switch leads the first soldering work failed. After a new switch replaced the misconnected one, the power source board completed and successfully combined on the vehicle. The second vehicle was built as essentially same as the first one, the only different is the frame of it was only one steel plate instead of four.

Figure7. Photo of completed first vehicle (Chunyang, Jing 2021)



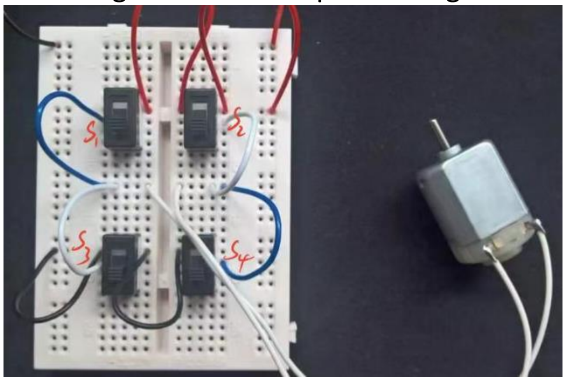
Construction of H-bridge have fully explained on the H bridge project that was completed step by step. Before build a H-bridge in the lab, basis of breadboard should learn. A breadboard is a laboratory device that permits flexible building and testing of circuits without having to permanently solder everything together. It is very convenient, as it allows the electronics designer to test their ideas without having to go through the expensive and time-consuming process of manufacturing 'finished' products.

Figure8. A general breadboard



An H-bridge circuit could content four button switches or two slide switch and a load. Four button switch was chosen to build on breadboard, supply voltage set on 5V and load circuit was a motor. In figure 9, when turn the switch S1 and S4 on, the motor will run clockwise. When turn the switch S2 and S3 on, the motor will run anticlockwise. This circuit was based on the schematic diagram in figure 2.

Figure9. An example H-bridge



Task2: Programming for the vehicle by Arduino Nano

Assignment a: Run exactly 10m and automatically stop with one source code

Download the Arduino IDE on PC and learn the basic usage of it in advance, including the syntax. An example code of 10m race was given on Moodle, but it can not perfectly fit in the built vehicle. In that case, in order to run exactly 10m, some values in the example code needs to be adjust. Finally, the define value “TEN_METER” was changed from 11129 to 8000 based on experiment’s feedback. The speed set of each motor was decided to use different power to run a straight line as the mechanic characteristic of each motor is not the same. To make vehicle running as quick as possible, right forward motor was set in “85”, right backward motor in “84”, left forward motor in “79”, left backward motor in “78”. Those values were decided through experiments of running the vehicle, data shows on table 1.

Experiment	Left forward	Left backward	Right forward	Right backward	Running pathway
1	50	50	50	50	To the left
2	50	50	60	60	To the right
3	50	50	57	57	Straight
4	80	80	90	90	To the right
5	79	78	85	84	Straight

Table1. Vehicle speed values

Upload the code to Arduino Nano board and connect the Arduino board onto the vehicle using four wires.

Assignment b: Spin the vehicle for exactly one cycle and automatically stop

This assignment requires to set different direction and speed of different motor, in that case, different command ID needs to be combined. The command ID “fa” was changed into “baffrr” that makes one side of motors of vehicle running clockwise and the other side of motors running anticlockwise. There was no requirement of speed, so the speed could only relate to the encoder value of motors. The define value "TEN_METER" was changed to 3750 and the speed set in 45.

Assignment c: Run around a 1m square and automatically stop at the start position

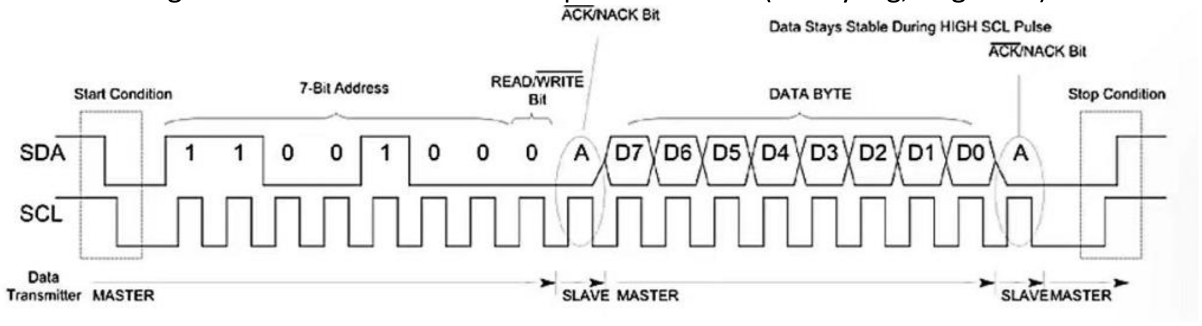
This assignment requires not only set different direction and speed of each motor also to change the stage of each motor in different position. Basically, run 1m straight line, spin a quarter of circular at a point, repeat both two procedures for four times to complete a cycle of 1m square. The speed set of each state: right forward and backward motors were both set in "50", and the left forward and backward motors were both set in "45". Command ID used in straight line was "baffff", and "baffrr" was used in turning point. In order to decrease the mechanic error in each turning points, a short stay between every change of states was added(from straight line to turning) using the instruction "delay()". The short delay was set in 0.5 second.

Task3: Establish serial communication between two Arduino and design HMI

In this session, three general serial communication protocols were studied: I^2C, UART, and SPI.

I^2C(Inter-Integrated Circuit): Synchronous, multi-master, multi-slave, packet switched, single-ended, serial communication bus, half-duplex

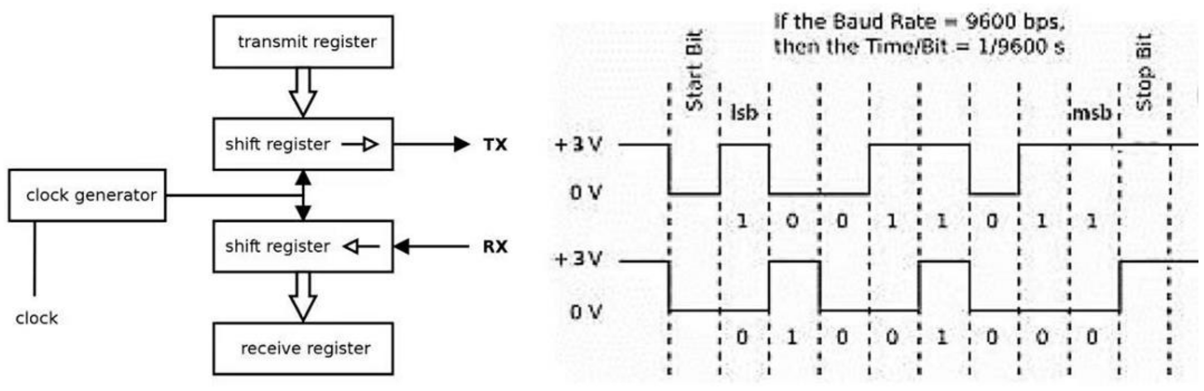
Figure10. Serial communication protocol: I^2C (Chunyang, Jing 2021)



Using the code from Arduino official web, a serial communication circuit between two Arduino board was built. The Wire Master Writer code was written into Arduino1 as a massage sender and the Wire Slave Receiver code was written into Arduino2 as a massage reader. In this case, the variable "x" was given value in Arduino1 and send those values to Arduino2 so the serial monitor of Arduino2 could display the values generated in Arudino1.

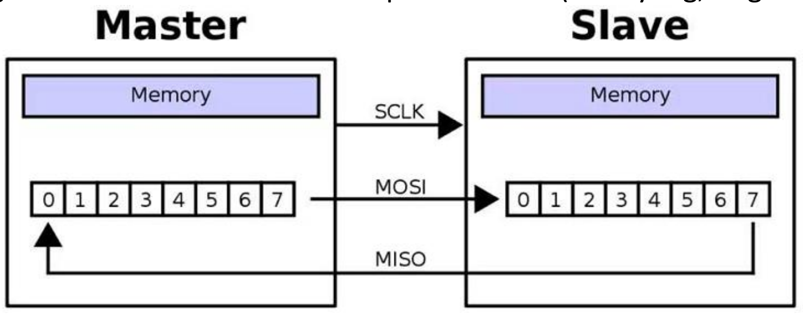
UART (Universal Asynchronous Receiver-Transmitter): Asynchronous, point-to-point, serial communication bus, half-duplex or full-duplex.

Figure11. Serial communication protocol: UART (Chunyang, Jing 2021)



SPI (Serial Peripheral Interface): Synchronous, multi-slave serial communication bus, full-duplex.

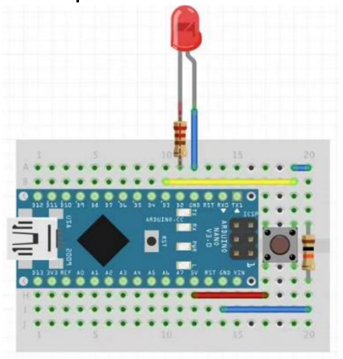
Figure12. Serial communication protocol: SPI (Chunyang, Jing 2021)



Human Machine Interface in this session are based on the following components, the inputs: button switch, slide switch, microphone, and potentiometer. The outputs: active buzzer, passive buzzer, LED, and LCD 1602. The example codes were provided, so the main point is to build the circuit of each HMI. The breadboard was chosen to build those circuit as it permits flexible building and testing of circuits without having to permanently solder everything together.

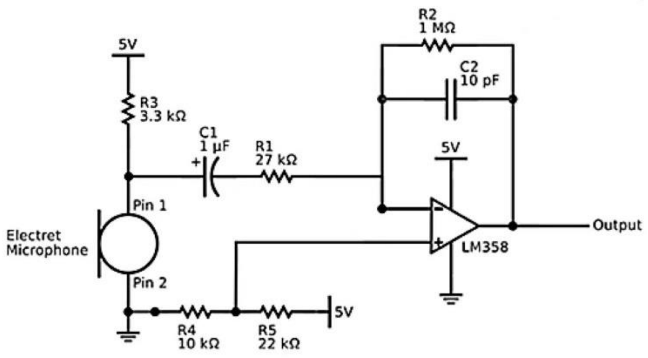
The first HMI was using a button switch to turn on or off a LED. In the circuit, a 150 Ohm resistor was used to protect the circuit which is in series with LED, and a button switch was added on the circuit in series with LED. Using Arduino to provide a 5V DC voltage with positive pole connected with positive pole of diode. Similarly, a slide switch could also be used to control a LED or buzzer.

Figure13. An example of LED and button switch HMI



For the electret microphone input, an amplifier circuit was needed to make it work Figure 14 shows a typical amplifier circuit of a microphone.

Figure14. A typical amplifier circuit of microphone



This circuit was built on breadboard using the same values from the figure 1. Connect it into an HMI circuit as input, it could light a LED or active a buzzer.

Running LCD 1602 should use a library from Arduino IDE called “LiquidCrystal” that could handle all communications with the LCD. What required to declare were:

• Which Arduino pins are connected to which pins on the display
• Set up the display - how many rows/columns
• Send the data to be displayed

Task4: Real-Time control system and low battery indicator

A Real-Time controlled sinewave generator needs to be designed with the following requirements:

• DC source voltage 15 V; Constant peak-to-peak voltage output (9 V)
• Fundamental frequency of 1 Hz to 4 Hz (incr 0.2Hz) sent by Serial Monitor
• RT control frequency 1 kHz or 500 Hz
• Design an amplification (e.g. push-pull) circuit and filter circuit

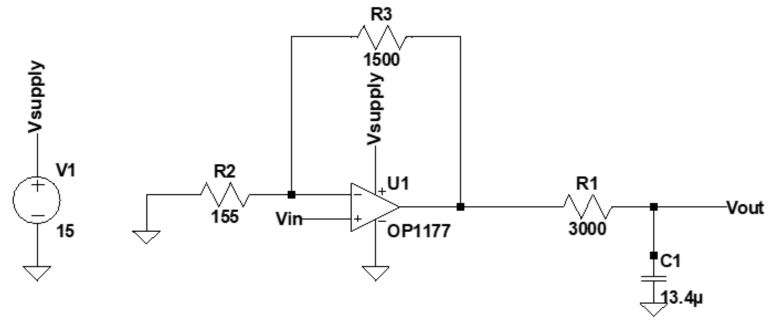
Arduino code needs include the library “MsTimer2.h” to generate a sinewave signal. A variable: curve, was defined

$$curve = (int)\,(24 * (sin(2 * PI * f * t) + 1))$$

The curve was to generate a string value of sinewave and using “analogWrite (6, curve)” to set pin 6 as signal output. The magnitude of curve was set in 24 to fit the peak-to-peak voltage into constant 9V. Plus 1 aims to make all values of the sinewave positive. “f” stands for frequency and “t” stand for time. In order to set the exact same value sent from serial monitor into the frequency in Hz, some mathematical calculation was added to the codes.

As the “TX” pin on Arduino could only sending “0” or “1”, the designing and building of amplification and filter circuit was needed. Amplification circuit to amplifier signal and filter to decrease noises. The usage of amplifier LM358 needs to be learned in advance and the values of resistor and capacitor needs calculation and experiments to verify.

Figure15. Schematic of the amplification circuit and low pass filter
(All voltage unit in V, current unit in A, resister unit in Ohm, capacitor unit in F)



This diagram shows the schematic of the amplification circuit and the low pass filter. Values of R1 and C1 were calculated by:

$$f = \frac{1}{2\pi RC} \qquad Equ1$$

$$f = \frac{1}{2\pi RC} = \frac{1}{2\pi \times R1 \times C1} = 4Hz$$

$$R1 \times C1 = 0.0398$$

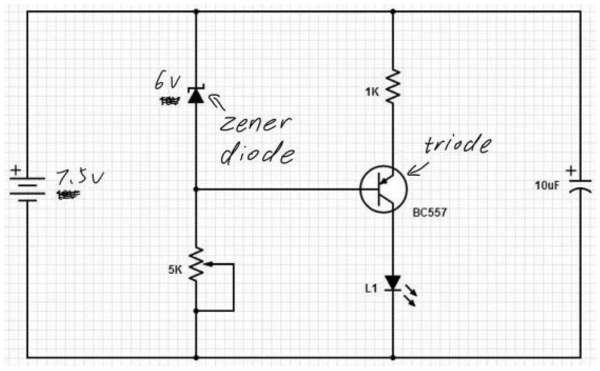
Given R1=3000 Ohm, then C1=$13.4 \times 10^{-6}F$.
The value of R2 and R3 were based on Equ2 and experiments.

$$\frac{V_{out}}{V_{in}} = \frac{R_2 + R_3}{R_2} \qquad Equ2$$

After repeatedly experiment, R3 chosen to be 1500 Ohms and R2 chosen to be 155 Ohms.

The building of low battery indicator was a pure hardware experiment which needs to design, build, solder, and verifier a circuit on stripboard. The schematic diagram of a 12V indicator has been shown in figure 5. The same working theory was applied on this 6V indicator.

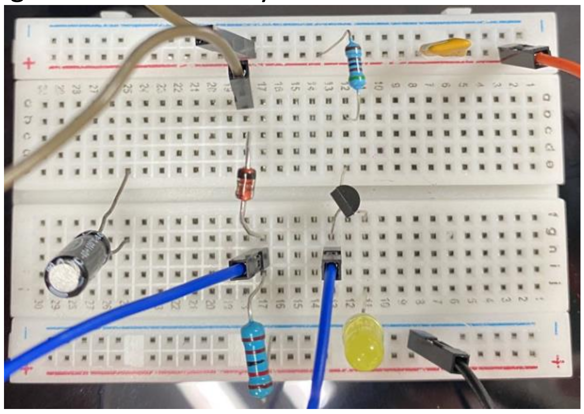
Figure15. Schematic diagram for 6V indicator



As schematic shows, the break down voltage of this Zener diode was set in 6V, which means when the voltage on it goes higher than 6V, the Zener diode will be broken down. In that case, the wire connected with triode will given voltage to it that leads the 1k resistor and the LED connected, and then, the LED will be lighted. The capacitor that in parallel with circuit was aims to protect the circuit.

Build and test the designed circuit on breadboard. After all the components checked, soldering started. A fuse was added to the circuit protect the circuit.

Figure16. Low battery indicator on breadboard



Task5. Programming one source code to finish 2 of 3 tasks based on Arduino Nano

Task A (Spin + [sound alert + LED blinking (both solid ON)]) and task C (Run forward or backward [sound alert + LED blinking (both 1s ON - 1s OFF)]) were chosen to be completed. The two tasks could be divided into two parts: programming by Arduino IDE and connecting HMIs with the vehicle.

In first part, it is required to using interrupt function within the code, so the frame of this code was built on one setup, one loop function and some external functions that could be active by “attachInterrupt()”. Running forward or backward, spin clockwise or anticlockwise were also involved in external functions. As it is needed to sound alert and blinking LED during the running state, sound playing code and LED blinking code were also written in the external functions of different running states.

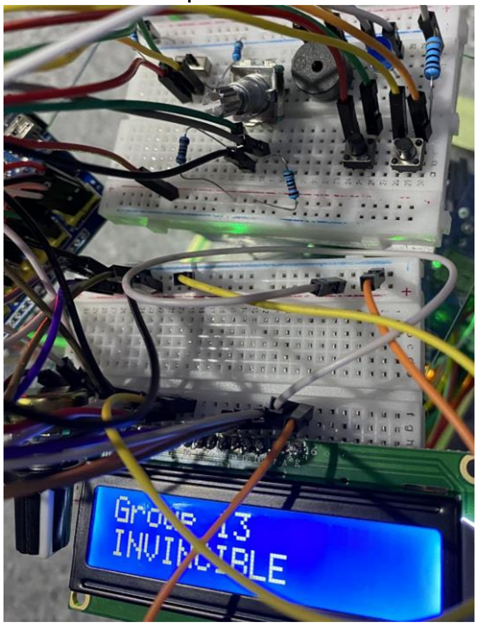
In order to set the speed and target of running by turning the provided encoder, two external functions: “doencoder” and “readencoder” were written. For the LCD 1062 display (Task #, speed set, target set, direction set, Status: Running or Stopped), the specific text was located on different functions.

As it will have too many arguments in code if different music playing in every different running state, it is decided to change the music playing speed of different running state.

For the second hardware part, two button switch, one slide switch, one encoder, one buzzer, and one LED was combined. Two button switches were designed to choose different task, slide switch designed to choose forward/backward or clockwise/anticlockwise.

Two 1k Ohm resister were connect in series with the two button switch as the load to call the “RISING” in the instruction “attachPinChangeInterrupt()”.

Figure17. The completed HMIs on the vehicle



After the preparation, the display on LCD, blink of LED, and sound of buzzer was working properly in the previous experiments but the vehicle always in stable. The wire connection on Arduino was checked and the code was reuploaded, after that, the vehicle start work.

RESULTS:

Task1: Vehicle assemble and H-bridge building

Two vehicles were successfully built, all the components have been used on the vehicles. The green one frame on four steel plates (vehcile1) was the one that used through the whole experiment because of its high preferments and controllability. The second built vehicle with one steel plate as frame (vehicle2) was the backup one.

An H-bridge with four button switch was built on the breadboard, a motor was added into this circuit as the load. Using power supply to provide a 5V voltage to H-bridge, in that case, if the switch S1 and S3 (shown in figure 9) were closed, the motor will run clockwise, if the switch S2 and S4 were closed, the motor will run anticlockwise.

This task was successfully fulfilled.

Task2: Programming for the vehicle by Arduino Nano

Three assignments in this task:

a. Run exactly 10m and automatically stop with one source code
b. Spin the vehicle for exactly one cycle and automatically stop
c. Run around a 1m square and automatically stop at the start position

All the three assignments were completed based on the given code from Moodle page. At the beginning, original code was directly uploaded to Arduino board and the vehicle1 was runed somewhere farther than 10m. So, the define value “TEN_METER” was changed after experiments. Finally, the vehicle1 could run a straight line and stop at a point between 10m and 10.5m.

After some further study of command ID and “delay()”, task b was finished by using command ID: “baffrr” and time controlled by “delay()”. Task c was finished by combine different command ID (“baffff”, “baffrr”) and using “delay()” control the time set on each running state. During the changing mechanic error of vehicle1, task c was a time-consuming work with a bulk of experiments required.

This task was successfully fulfilled. (See Appendix A for full codes in this task)

Task3: Establish serial communication between two Arduino and design HMI

In this part, three serial communication protocols were studied: I2C, UART, SPI. A master writer, slaver receiver code was written into Arduino IDE from Arduino official web. Pin 4 was used to transmission, and the circuit has been built on breadboard. Two computers were used on this experiment. Master writer code was written to Arduino1, slaver receiver code written to Arduino2. Open the serial monitor of Arduino2, “x” values sent from Arduino1 was displayed, which could be changed with the code in Arduino1.

All the outputs and inputs components were built. Maxima 16 connections could be completed on breadboard. As the requirements of task, the final combined HMI was contained two button switch, one slide switch, one LED, one LCD 1062 and one encoder, which was later used on vehicle1.

Inputs	Outputs
Button switch	Active buzzer
Slide switch	Passive buzzer
Microphone	LED
Potentiometer	LCD 1062

Table2. Inputs and outputs of HMI

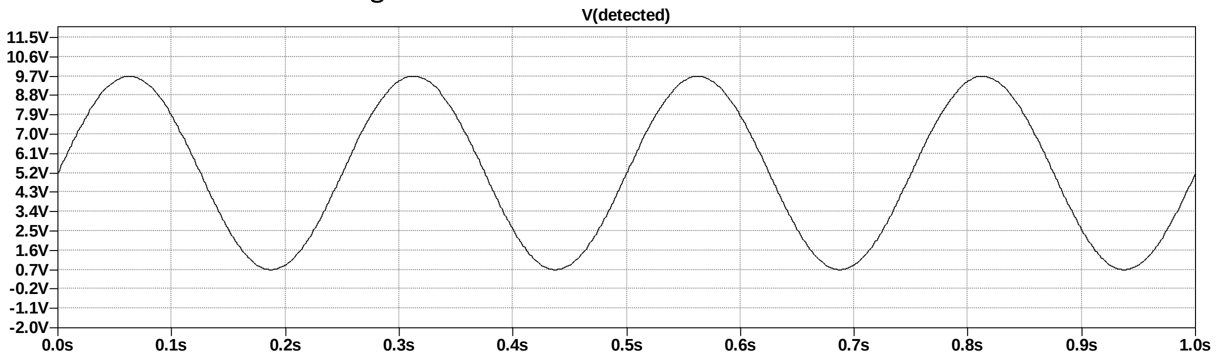
This task was successfully fulfilled. (See Appendix B for full codes in this task)

Task4: Real-Time control system and low battery indicator

Aiming to generate a sinewave voltage signal, a string of sinewave numbers was generated by Arduino and sent to amplification circuit to amplifier the magnitude to meet the requirement of 9V peak-to-peak voltage. The output of amplification circuit was connected to a low pass filter that could decrease the noise.

An oscilloscope was connected in parallel with the load to detect whether the generated voltage signal meets all the requirements. After constant experiments on the circuit and debugs on code, the final detected voltage signal when given frequency was 4Hz shows in figure 18.

Figure18. RT-controlled sinewave in 4Hz



Send a number (1 to 4, incr0.2) using Arduino serial monitor, this number will be automatically set as the frequency of the generated sinewave.

Building of a low battery indicator was a pure hardware project. After design on theory, debug on breadboard, and solder on stripboard, a low battery indictor was completed. This indicator contains a 1k Ohm resistor, a 5k Ohm resistor, a fuse, a triode, a zener diode, a capacitor, and a LED. When the voltage on the indicator is higher than 6V, LED will be light.

This task was successfully fulfilled. (See Appendix C for full codes in this task)

Task5. Programming one source code to finish 2 of 3 tasks based on Arduino Nano

At the end of all the experiments, it is decided to running task A and task C separately as the limitation of remained time. In that case, the vehicle could spin clockwise or anticlockwise (controlled by slide switch) and running forward or backward (controlled by slide switch).

Using task A for example. At the beginning of vehicle start, a user-friendly info displayed on LCD screen “please select task”, press the button, the screen will show “Task A”, and then, turning the encoder clockwise to set the target (encoder value) anticlockwise to set the speed (0-100) of vehicle. Press the button again, the screen will display “Input direction:", shift the slide switch to choose the direction (clockwise or anticlockwise). After direction chosen, press the button, the screen will display the direction and the vehicle will start spinning. At the same time of vehicle running, LED lighted, sound alerted.

Here, if shift the slide switch to the other slide and press the button during the vehicle spinning clockwise, the vehicle will start to spin anticlockwise, the screen display will change to “anticlockwise”, and the sound will alert in a way faster. It is also possible to change the speed during the vehicle is running. Turning the encoder anticlockwise, the screen will automatically display the speed value that could be changed by turning encoder. After a new value of speed set, press the button, the speed of vehicle will change to the set value and the screen display will back to the direction of vehicle.

In order to fulfil task C, several small changes were needed on the code. A point of the change: the blinking frequency of LED becomes 1s on and 1s off.

Although it’s recommended to use microphone to start the vehicle, and the microphone circuit has already been built, it didn’t used on the vehicle because of the time limitation.

One issue of the code was that the vehicle cannot stop when the encoder reached the target, which means the target set of the vehicle was invalid.

This task was not completely fulfilled. (See Appendix D for full codes in this task)

DISCUSSION:

In summary, 4 of 5 tasks were completed. However, there are many methodologies could be improved through the whole experiment.

Task 1: At the very beginning of assemble, it should be considered where are the components that easily loose. In this experiment, the screw between the motor and vehicle frame usually loose during experiment, so its better to tighten the screws on motors.

There are a lot of wires using on breadboard to build an H-bridge, which is easily to be confused. So, it is recommended to use different colour to identify wires (eg. Positive pole using rad wire).

Task 2: It takes a big amount of time to complete assignment c: run around a 1m square because it was using “delay()” to control the distance that vehicle travelled. It is facing several issues here: battery voltage will change as it consumes, mechanic error in every experiment was unpredictable. Using the value of encoder to control the distance will be better as it is actually setting the number of turns of each motor.

Task 3: The building of serial communication and human machine interface was working properly except the microphone building. The typical circuit of microphone was in a way too complex, and it was also hard to find an exact same value capacitors and resistors as the circuit shows. Several previous built circuits were theoretically same as the figure 14, but it could not work properly. After replacing some components on that circuit, the microphone starts to work. The previous issue was considered as the poor contact of components and the hardware mismatch.

Task 4: The real-time control sinewave generator was working properly as it designed to. It is recommended to do not be too strict to use one resistor that perfectly fit the required value. For example, if a 155 Ohms resistor needed, it is fine to use a 150 Ohm one combine with a 5 Ohm one.

Task 5: This task has not been fully completed. Theoretically, the written code should be able to drive the vehicle finish the task, but problem happens on the encoder. Although it is feasible to set an encoder value as target by turning the encoder on breadboard, the vehicle can not stop after the motor encode value reached target. This problem should related to the external function “readEncoder()”, some further changes may needs to apply on this function. As time limited, the final solution of this issue was not found at the end of experiment.

Even though the microphone circuit has been built, it was not used in this task as the majority of time was spent on programming.

CONCLUSION:

In conclusion, this experiment is from hardware building to software designing and applying. Start from assembly vehicle, investigated the function of each component, and then, the basis of H-bridge was learned. How a sample vehicle form and how it could run to different direction have been figured out in the first task. After hardware prepared, some simple programming began, 10m running forward example code could been downloaded directly from Moodle. Several variable needs be changed to fit in the mechanic characteristic of each motor. By using “delay()” to change between different running state, the 1m square task was completed.

After above experiment, some further study on Arduino started. Three serial communication protocols: I2C, UART, SPI were learned, which also including the knowledge of Arduino Nano pins. Some basic HMI were designed and built on breadboard at the same time. In addition, a RT-control system and a low battery indicator was designed and built. This enhanced the understanding of Arduino programming and improved hardware building skill. The finial challenge was based on programming, a long, complex programme should be written to meet all the requirements, which makes this task couldn’t easily be completed.

In the coming weeks before next experiment begin, further study about Arduino programming is essential. The issue in function “readEncoder()” should also be discussed.

References

Wikipedia contributors
2021 https://en.wikipedia.org/wiki/User_interface

Chunyang GU, Jing Wang
2021 Session II: Real-Time (RT) Control and Human Machine Interface (HMI).

Juang, Hau-Shiue, and Kai-Yew Lum
2013 Design and control of a two-wheel self-balancing robot using the arduino microcontroller board. 2013 10th IEEE International Conference on Control and Automation (ICCA), 2013, pp. 634-639. IEEE.

Appendix A

Task a: run 10m

```
#include <Wire.h>
{Cheng, 2011 #1}
#define METER (8000)      // 234.256*10/(0.067*3.1415926)
long unsigned int encoder1Value = 0;
long unsigned int encoder2Value = 0;
long unsigned int encoder3Value = 0;
long unsigned int encoder4Value = 0;

long unsigned int target1 = 0;

void setup() {
   unsigned char i=0;
   // put your setup code here, to run once:

   pinMode(13, OUTPUT);    // Configure the reset pin as an output and hold the robot in reset
   digitalWrite(13,LOW);

   //Setup the serial ports
   Serial.begin(57600);        // This is the hardware port connected to the PC through the USB
connection
   Wire.begin();
   target1 = 0x80000000 + METER;

   // wait 5 seconds
   delay(1000);

   Serial.println("software serial simple test!");    // Send some text to the PC

   digitalWrite(13,HIGH);    // Release the robot from reset
   delay(100);    // A short delay to allow the robot to start-up

Wire.begin();
   Wire.beginTransmission(42);
   Wire.write("fa");
   for (i=0;i<=3;i++)
   {
      Wire.write(79);//left forward
      Wire.write(0);
      Wire.write(78);//left back
      Wire.write(0);
      Wire.write(85);//right forward
      Wire.write(0);
      Wire.write(84);//right back
      Wire.write(0);
```

```
    }
      Wire.endTransmission();

}


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

    delay(10);
    readEncoder();
    Serial.println("Encoder 1 read text!");    // Send some text to the PC
    Serial.println(encoder1Value);    // Send some text to the PC
    Serial.println("Encoder 2 read text!");    // Send some text to the PC
    Serial.println(encoder2Value);    // Send some text to the PC
    Serial.println("Encoder 3 read text!");    // Send some text to the PC
    Serial.println(encoder3Value);    // Send some text to the PC
    Serial.println("Encoder 4 read text!");    // Send some text to the PC
    Serial.println(encoder4Value);    // Send some text to the PC
    delay(500);

    if (encoder1Value > 0x80040000)
    {
        encoder1Value=0x80000000;
    }
    else if (encoder1Value < 0x7FFC0000)
    {
        encoder1Value=0x80000000;
    }

    if (encoder1Value > target1)
    {
        //Stop all the motors
        Wire.beginTransmission(42);
        Wire.write("ha");//ha: stop
        Wire.endTransmission();
        digitalWrite(13,HIGH);
        while(1);
    }
}

void readEncoder()
{
    long unsigned int encoder1 = 0;
    long unsigned int encoder2 = 0;
    long unsigned int encoder3 = 0;
    long unsigned int encoder4 = 0;
    Wire.beginTransmission(42);
    Wire.write("i");
```

```
   Wire.endTransmission();
   delay(1);
   Wire.requestFrom(42,8);
   delay(10);
   //if(Wire.available()==8)
   {
      encoder1 = (long unsigned int) Wire.read();
      encoder1 += ((long unsigned int) Wire.read() <<8);
      encoder1 += ((long unsigned int) Wire.read() <<16);
      encoder1 += ((long unsigned int) Wire.read() <<24);
      encoder2 = (long unsigned int) Wire.read();
      encoder2 += ((long unsigned int) Wire.read() <<8);
      encoder2 += ((long unsigned int) Wire.read() <<16);
      encoder2 += ((long unsigned int) Wire.read() <<24);
   }
   encoder1Value = encoder1;
   encoder2Value = encoder2;
   Wire.requestFrom(42,8);
   delay(10);
   //if(Wire.available()==8)
   {
      encoder3 = (long unsigned int) Wire.read();
      encoder3 += ((long unsigned int) Wire.read() <<8);
      encoder3 += ((long unsigned int) Wire.read() <<16);
      encoder3 += ((long unsigned int) Wire.read() <<24);
      encoder4 = (long unsigned int) Wire.read();
      encoder4 += ((long unsigned int) Wire.read() <<8);
      encoder4 += ((long unsigned int) Wire.read() <<16);
      encoder4 += ((long unsigned int) Wire.read() <<24);
   }
   encoder3Value = encoder3;
   encoder4Value = encoder4;
}
```

Task b: spin

```
#include <Wire.h>

#define TEN_METER (3750)      // 234.256*10/(0.067*3.1415926)
long unsigned int encoder1Value = 0;
long unsigned int encoder2Value = 0;
long unsigned int encoder3Value = 0;
long unsigned int encoder4Value = 0;

long unsigned int target1 = 0;

void setup() {
    unsigned char i = 0;
    // put your setup code here, to run once:
    pinMode(13, OUTPUT);    // Configure the reset pin as an output and hold the robot in reset
    digitalWrite(13, LOW);

    //Setup the serial ports
    Serial.begin(57600);        // This is the hardware port connected to the PC through the USB
connection
    Wire.begin();
    target1 = 0x80000000 + TEN_METER;

    // wait 1 seconds
    delay(1000);

    Serial.println("software serial simple test!");    // Send some text to the PC

    digitalWrite(13, HIGH); // Release the robot from reset
    delay(100);    // A short delay to allow the robot to start-up

    Wire.beginTransmission(42);
    Wire.write("baffrr");
        Wire.write(45);
        Wire.write(0);
    Wire.endTransmission();
}

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

    delay(10);
    readEncoder();
    Serial.println("Encoder 1 read text!");    // Send some text to the PC
    Serial.println(encoder1Value);    // Send some text to the PC
    Serial.println("Encoder 2 read text!");    // Send some text to the PC
    Serial.println(encoder2Value);    // Send some text to the PC
    Serial.println("Encoder 3 read text!");    // Send some text to the PC
    Serial.println(encoder3Value);    // Send some text to the PC
    Serial.println("Encoder 4 read text!");    // Send some text to the PC
    Serial.println(encoder4Value);    // Send some text to the PC
    delay(500);

    if (encoder1Value > 0x80040000)
    {
        encoder1Value = 0x80000000;
    }
    else if (encoder1Value < 0x7FFC0000)
    {
        encoder1Value = 0x80000000;
    }
    if (encoder1Value > target1)
    {
        //Stop all the motors
        Wire.beginTransmission(42);
        Wire.write("ha");
        Wire.endTransmission();
        digitalWrite(13, HIGH);
        while (1);
    }
}

void readEncoder()
{
    long unsigned int encoder1 = 0;
    long unsigned int encoder2 = 0;
    long unsigned int encoder3 = 0;
    long unsigned int encoder4 = 0;
    Wire.beginTransmission(42);
    Wire.write("i");
    Wire.endTransmission();
    delay(1);
    Wire.requestFrom(42, 8);
    delay(10);
    //if(Wire.available()==8)
    {
        encoder1 = (long unsigned int) Wire.read();
        encoder1 += ((long unsigned int) Wire.read() << 8);
        encoder1 += ((long unsigned int) Wire.read() << 16);
        encoder1 += ((long unsigned int) Wire.read() << 24);
        encoder2 = (long unsigned int) Wire.read();
        encoder2 += ((long unsigned int) Wire.read() << 8);
        encoder2 += ((long unsigned int) Wire.read() << 16);
        encoder2 += ((long unsigned int) Wire.read() << 24);
    }
    encoder1Value = encoder1;
    encoder2Value = encoder2;
    Wire.requestFrom(42, 8);
    delay(10);
    //if(Wire.available()==8)
    {
        encoder3 = (long unsigned int) Wire.read();
        encoder3 += ((long unsigned int) Wire.read() << 8);
        encoder3 += ((long unsigned int) Wire.read() << 16);
        encoder3 += ((long unsigned int) Wire.read() << 24);
        encoder4 = (long unsigned int) Wire.read();
        encoder4 += ((long unsigned int) Wire.read() << 8);
        encoder4 += ((long unsigned int) Wire.read() << 16);
        encoder4 += ((long unsigned int) Wire.read() << 24);
    }
    encoder3Value = encoder3;
    encoder4Value = encoder4;
}
```

Task c: run around a 1m square

```
#include <Wire.h>

#define TEN_METER (10000)
long unsigned int encoder1Value = 0;
long unsigned int encoder2Value = 0;
long unsigned int encoder3Value = 0;
long unsigned int encoder4Value = 0;
long unsigned int target1 = 0;

void setup() {
    unsigned char i=0;
    pinMode(13, OUTPUT);
    digitalWrite(13,LOW);

    //Setup the serial ports
    Serial.begin(57600);
    Wire.begin();
    target1 = 0x80000000 + TEN_METER;
    delay(1000);

    Serial.println("software serial simple test!");

    digitalWrite(13,HIGH);
            delay(100);

Wire.beginTransmission(42);
Wire.write("baffff");
        Wire.write(48);
        Wire.write(0);
        Wire.write(48);
        Wire.write(0);
        Wire.write(50);
        Wire.write(0);
        Wire.write(50);
        Wire.write(0);
Wire.endTransmission();
delay(1430);

Wire.beginTransmission(42);
Wire.write("baffff");
```

```
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
Wire.endTransmission();
delay(500);

Wire.beginTransmission(42);
Wire.write("baffrr");
        Wire.write(45);
        Wire.write(0);
        Wire.write(45);
        Wire.write(0);
        Wire.write(50);
        Wire.write(0);
        Wire.write(50);
        Wire.write(0);
Wire.endTransmission();
delay(880);

Wire.beginTransmission(42);
Wire.write("baffff");
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
Wire.endTransmission();
delay(500);

Wire.beginTransmission(42);
Wire.write("baffff");
        Wire.write(45);
        Wire.write(0);
        Wire.write(45);
        Wire.write(0);
        Wire.write(50);
        Wire.write(0);
        Wire.write(50);
        Wire.write(0);
Wire.endTransmission();
delay(1400);
```

```
Wire.beginTransmission(42);
Wire.write("baffff");
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
Wire.endTransmission();
delay(500);

Wire.beginTransmission(42);
Wire.write("baffrr");
        Wire.write(45);
        Wire.write(0);
        Wire.write(45);
        Wire.write(0);
        Wire.write(50);
        Wire.write(0);
        Wire.write(50);
        Wire.write(0);
Wire.endTransmission();
delay(880);

Wire.beginTransmission(42);
Wire.write("baffff");
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
Wire.endTransmission();
delay(500);

Wire.beginTransmission(42);
Wire.write("baffff");
        Wire.write(45);
        Wire.write(0);
        Wire.write(45);
        Wire.write(0);
        Wire.write(50);
        Wire.write(0);
        Wire.write(50);
        Wire.write(0);
Wire.endTransmission();
delay(1360);

Wire.beginTransmission(42);
Wire.write("baffff");
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
Wire.endTransmission();
delay(500);

Wire.beginTransmission(42);
    Wire.write("baffrr");
        Wire.write(45);
        Wire.write(0);
        Wire.write(45);
        Wire.write(0);
        Wire.write(50);
        Wire.write(0);
        Wire.write(50);
        Wire.write(0);
Wire.endTransmission();
delay(790);//third corner

Wire.beginTransmission(42);
Wire.write("baffff");
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
Wire.endTransmission();
delay(500);

Wire.beginTransmission(42);
Wire.write("baffff");
        Wire.write(45);
        Wire.write(0);
        Wire.write(45);
        Wire.write(0);
        Wire.write(50);
        Wire.write(0);
        Wire.write(50);
        Wire.write(0);
Wire.endTransmission();
delay(1350);

Wire.beginTransmission(42);
Wire.write("baffff");
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
Wire.endTransmission();
delay(500);

Wire.beginTransmission(42);
    Wire.write("baffrr");
          Wire.write(45);
        Wire.write(0);
        Wire.write(45);
        Wire.write(0);
        Wire.write(50);
        Wire.write(0);
        Wire.write(50);
        Wire.write(0);
    Wire.endTransmission();
delay(820);

  Wire.beginTransmission(42);
  Wire.write("baffff");
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
      Wire.endTransmission();
      delay(500);

Wire.beginTransmission(42);
Wire.write("baffff");
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
        Wire.write(0);
Wire.endTransmission();
delay(500);
}
      Wire.endTransmission();

}


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

    delay(10);
    readEncoder();
    Serial.println("Encoder 1 read text!");    // Send some text to the PC
    Serial.println(encoder1Value);    // Send some text to the PC
    Serial.println("Encoder 2 read text!");    // Send some text to the PC
    Serial.println(encoder2Value);    // Send some text to the PC
    Serial.println("Encoder 3 read text!");    // Send some text to the PC
    Serial.println(encoder3Value);    // Send some text to the PC
    Serial.println("Encoder 4 read text!");    // Send some text to the PC
    Serial.println(encoder4Value);    // Send some text to the PC
    delay(500);

    if (encoder1Value > 0x80040000)
    {
        encoder1Value=0x80000000;
    }
    else if (encoder1Value < 0x7FFC0000)
    {
        encoder1Value=0x80000000;
    }

    if (encoder1Value > target1)
    {
        //Stop all the motors
        Wire.beginTransmission(42);
        Wire.write("ha");//ha: stop
        Wire.endTransmission();
        digitalWrite(13,HIGH);
        while(1);
    }
}
```

```
void readEncoder()
{
   long unsigned int encoder1 = 0;
   long unsigned int encoder2 = 0;
   long unsigned int encoder3 = 0;
   long unsigned int encoder4 = 0;
   Wire.beginTransmission(42);
   Wire.write("i");
   Wire.endTransmission();
   delay(1);
   Wire.requestFrom(42,8);
   delay(10);
   //if(Wire.available()==8)
   {
      encoder1 = (long unsigned int) Wire.read();
      encoder1 += ((long unsigned int) Wire.read() <<8);
      encoder1 += ((long unsigned int) Wire.read() <<16);
      encoder1 += ((long unsigned int) Wire.read() <<24);
      encoder2 = (long unsigned int) Wire.read();
      encoder2 += ((long unsigned int) Wire.read() <<8);
      encoder2 += ((long unsigned int) Wire.read() <<16);
      encoder2 += ((long unsigned int) Wire.read() <<24);
   }
   encoder1Value = encoder1;
   encoder2Value = encoder2;
   Wire.requestFrom(42,8);
   delay(10);
   //if(Wire.available()==8)
   {
      encoder3 = (long unsigned int) Wire.read();
      encoder3 += ((long unsigned int) Wire.read() <<8);
      encoder3 += ((long unsigned int) Wire.read() <<16);
      encoder3 += ((long unsigned int) Wire.read() <<24);
      encoder4 = (long unsigned int) Wire.read();
      encoder4 += ((long unsigned int) Wire.read() <<8);
      encoder4 += ((long unsigned int) Wire.read() <<16);
      encoder4 += ((long unsigned int) Wire.read() <<24);
   }
   encoder3Value = encoder3;
   encoder4Value = encoder4;
}
```

Appendix B

Master writer:

```
// Wire Master Writer
// by Nicholas Zambetti <http://www.zambetti.com>

// Demonstrates use of the Wire library
// Writes data to an I2C/TWI slave device
// Refer to the "Wire Slave Receiver" example for use with this

// Created 29 March 2006

// This example code is in the public domain.


#include <Wire.h>

void setup()
{
  Wire.begin(); // join i2c bus (address optional for master)
}

byte x = 0;

void loop()
{
  Wire.beginTransmission(4); // transmit to device #4
  Wire.write("x is ");        // sends five bytes
  Wire.write(x);              // sends one byte
  Wire.endTransmission();    // stop transmitting

  x++;
  delay(500);
}
```

Slave Receiver:

```
// Wire Slave Receiver
// by Nicholas Zambetti <http://www.zambetti.com>

// Demonstrates use of the Wire library
// Receives data as an I2C/TWI slave device
// Refer to the "Wire Master Writer" example for use with this

// Created 29 March 2006
// This example code is in the public domain.


#include <Wire.h>

void setup()
{
   Wire.begin(4);                          // join i2c bus with address #4
   Wire.onReceive(receiveEvent); // register event
   Serial.begin(9600);                // start serial for output
}

void loop()
{
   delay(100);
}

// function that executes whenever data is received from master
// this function is registered as an event, see setup()
void receiveEvent(int howMany)
{
   while(1 < Wire.available()) // loop through all but the last
   {
      char c = Wire.read(); // receive byte as a character
      Serial.print(c);              // print the character
   }
   int x = Wire.read();      // receive byte as an integer
   Serial.println(x);            // print the integer
}
```

Appendix C

Real-Time control sinewave voltage signal generator:

```
#include<MsTimer2.h>

#define FREQ_CTL (1000)

float t=0;
int curve;
double f=0.2;
int f1,f2;

void TimerISR()
{
   t=t+0.01;
   curve= (int) (24*(sin(2*PI*f*t) +1));
   analogWrite (6, curve) ;
}

void setup ()
{
   noInterrupts () ;
   MsTimer2::set (1000/FREQ_CTL, TimerISR);
   MsTimer2::start () ;
   pinMode(6,OUTPUT) ;
   interrupts ();
   delay (100);
   Serial.begin (9600);
}

void loop ()
{

if (Serial.available ()==2)
{
   f1=-(Serial.read () -48+4.4);
   delay (10);
   f2=-(Serial.read () -48-2);
   Serial. read ();
   f=(((double) f1+ (double) f2/10)/10);
   Serial.println (f);
}

   Serial. flush () ;
}
```

Appendix D

Task A: Spin + [sound alert + LED blinking (both solid ON)]

```
#include <PinChangeInterrupt.h>
#include <LiquidCrystal.h>
#include <MsTimer2.h>
#include <Wire.h>

#define encoder0PinA 2
#define encoder0PinB 3        //encoder pins

void anticlockwise();
void readEncoder();
void clockwise();
void doEncoder() ;
void Timer2ISR();      //function-prototype declaration


long unsigned int encoder1Value = 0;
long unsigned int encoder2Value = 0;
long unsigned int encoder3Value = 0;
long unsigned int encoder4Value = 0;
long unsigned int target1 = 0;        //original codes

unsigned int second;

volatile byte state = LOW;    //LED state

const byte interruptPin0 = 11;        //slideswitch pins
const byte interruptPin1 = 10;        //slideswitch pins
const byte switcnButtonAB = 20; //A6
const byte switcnButtonC = 5;    //A7 pushbuttonpins

int encoder0Pos = 0;
int encoder1Pos = 0;        //encoder counter
int speed0, target0;        //encoder output

int speakerPin = A0;
int length = 15; // the number of notes
char notes[] = "ccggaagffeeddc "; // a space represents a rest
int beats[] = { 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 2, 4 };
int tempo = 300;                //buzzer

const int rs = 17, en = 16, d4 = 4, d5 = 6, d6 = 7, d7 = 8;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);            //LCD pins
```

```
void setup()
 {

   pinMode(speakerPin, OUTPUT);        //speaker init

   pinMode(switcnButtonAB, INPUT_PULLUP);
   pinMode(switcnButtonC, INPUT);        //switch init

   lcd.begin(16, 2);
   lcd.setCursor(0, 0);
   lcd.print("Please select ");      //LCD init
   lcd.setCursor(0, 1);
   lcd.print("task:");      //LCD init

   pinMode(9, OUTPUT);        //LED init

   unsigned char i = 0;
   pinMode(13, OUTPUT);
   digitalWrite(13, LOW);      //original codes

   Serial.begin(57600);
   Wire.begin();
   delay(1000);                        //original codes

   pinMode(encoder0PinA, INPUT_PULLUP);
   pinMode(encoder0PinB, INPUT_PULLUP);        //encoder init
   attachInterrupt(0, doEncoder, RISING);      //use encoder to set speed0&target0

   int r = 0;
   while ((digitalRead(switcnButtonC)) != HIGH)
 {
      r++;
      Serial.print("r = ");
      Serial.println(r);
      delay(200);
   }          //wait for push button

   delay(500);

   lcd.clear();
   lcd.setCursor(0, 0);
   lcd.print("TaskC");        //LCD display task type C

   target1 = 0x80000000 + target0;    // set target1

   delay(500);

   lcd.clear();
   lcd.setCursor(1, 1);
```

```
  lcd.print(speed0);
  lcd.setCursor(1, 6);
  lcd.print(target0);      //LCD display speed & target

  pinMode(interruptPin0, INPUT);
  pinMode(interruptPin1, INPUT);        // slideswitch init

  int c = 0;
  while ((digitalRead(switcnButtonC)) != HIGH) {
    c++;
    Serial.print("c = ");
    Serial.println(c);
    delay(200);
  }         //wait for push button

  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Input direction:");


  attachPinChangeInterrupt(digitalPinToPCINT(interruptPin0), clockwise, RISING);
  attachPinChangeInterrupt(digitalPinToPCINT(interruptPin1),      anticlockwise,      RISING);
//external intterupts >> clockwise & anticlockwise

  delay(500);

  int a = 0;
  while ((digitalRead(switcnButtonC)) != HIGH) {
    a++;
    Serial.print("a = ");
    Serial.println(a);
    delay(200);      //wait for push button >> led blink & motor could sync
  }

  int FREQ_CTL1 = 1;
  MsTimer2::set((1000 / FREQ_CTL1), Timer2ISR);
  MsTimer2::start();                    //LED blinks

}


void loop() {

  delay(10);
  readEncoder();
  Serial.println("Encoder 1 read text!");    // Send some text to the PC
  Serial.println(encoder1Value);    // Send some text to the PC
  Serial.println("Encoder 2 read text!");    // Send some text to the PC
  Serial.println(encoder2Value);    // Send some text to the PC
```

```
    Serial.println("Encoder 3 read text!");    // Send some text to the PC
    Serial.println(encoder3Value);    // Send some text to the PC
    Serial.println("Encoder 4 read text!");    // Send some text to the PC
    Serial.println(encoder4Value);    // Send some text to the PC
    delay(500);

    if (encoder1Value > 0x80040000)
    {
       encoder1Value = 0x80000000;
    }
    else if (encoder1Value < 0x7FFC0000)
    {
       encoder1Value = 0x80000000;
    }


    if (encoder1Value > target1)
    {
       Wire.beginTransmission(42);
       Wire.write("ha");
       Wire.endTransmission();
       MsTimer2::stop();
       while (1);
    }     //Stop all motors & stop blink

}


void readEncoder()
{

    long unsigned int encoder1 = 0;
    long unsigned int encoder2 = 0;
    long unsigned int encoder3 = 0;
    long unsigned int encoder4 = 0;
    Wire.beginTransmission(42);
    Wire.write("i1");
    Wire.endTransmission();
    delay(1);
    Wire.requestFrom(42, 4);
    delay(10);
    if (Wire.available() == 4)
    {
       encoder1 = (long unsigned int) Wire.read();
       encoder1 += ((long unsigned int) Wire.read() << 8);
       encoder1 += ((long unsigned int) Wire.read() << 16);
       encoder1 += ((long unsigned int) Wire.read() << 24);
    }
    encoder1Value = encoder1;
```

```
Serial.println(encoder1Value);

}


void anticlockwise()
{
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("anticlockwise");      //LCD display task type C

  int e = 0;
  while ((digitalRead(switcnButtonC)) != HIGH)
 {
     e++;
     Serial.println(e);
     delay(200);
  }     //wait for push button >> led blink & motor could sync


  interrupts();
  Wire.beginTransmission(42);
  Wire.write("baffrr");

  Wire.write(speed0);
  Wire.write(0);
  Wire.write(speed0);
  Wire.write(0);
  Wire.write(speed0);
  Wire.write(0);
  Wire.write(speed0);
  Wire.write(0);

  Wire.endTransmission();

  while (1) {
    for (int i = 0; i < length; i++) {
      if (notes[i] == ' ') {
        delay(beats[i] * tempo / 100); // rest
      } else {
        playNote(notes[i], beats[i] * tempo);
      }
      // pause between notes
      delay(tempo / 3);
    }//music
  }


}
```

```
  void clockwise() {
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("clockwise");      //LCD display task type C

    int f = 0;
    while ((digitalRead(switcnButtonC)) != HIGH) {
      f++;
      Serial.println(f);
    }     //wait for push button >> led blink & motor could sync

    interrupts();
    Wire.beginTransmission(42);
    Wire.write("barrff");
    Wire.write(speed0);
    Wire.write(0);
    Wire.write(speed0);
    Wire.write(0);
    Wire.write(speed0);
    Wire.write(0);
    Wire.write(speed0);
    Wire.write(0);
    Wire.endTransmission();

    while (1) {
      for (int i = 0; i < length; i++) {
        if (notes[i] == ' ') {
          delay(beats[i] * tempo); // rest
        } else {
          playNote(notes[i], beats[i] * tempo);
        }

        // pause between notes
        delay(tempo / 2);
      }//music
    }

  }


  void doEncoder()
 {

    if (digitalRead(encoder0PinA) == digitalRead(encoder0PinB))
    {
      encoder0Pos++;
      speed0 = encoder0Pos;
    }
    else
    {
        encoder1Pos++;
        target0 = (encoder1Pos) * 50;

}

    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("speed: ");
    lcd.setCursor(7, 0);
    lcd.print(speed0);
    lcd.setCursor(0, 1);
    lcd.print("target: ");
    lcd.setCursor(8, 1);
    lcd.print(target0);
}

void Timer2ISR() {

    state = ! state;
    digitalWrite(9, state);

}



//buzzer
void playTone(int tone, int duration) {
    for (long i = 0; i < duration * 1000L; i += tone * 2) {
        digitalWrite(speakerPin, HIGH);
        delayMicroseconds(tone);
        digitalWrite(speakerPin, LOW);
        delayMicroseconds(tone);
    }
}

void playNote(char note, int duration) {
    char names[] = { 'c', 'd', 'e', 'f', 'g', 'a', 'b', 'C' };
    int tones[] = { 1915, 1700, 1519, 1432, 1275, 1136, 1014, 956 };

    // play the tone corresponding to the note name
    for (int i = 0; i < 8; i++) {
        if (names[i] == note) {
            playTone(tones[i], duration);
        }
    }
}//end
```

Task C: Run forward or backward [sound alert + LED blinking (both 1s ON - 1s OFF)]

```
#include <PinChangeInterrupt.h>
#include <LiquidCrystal.h>
#include <MsTimer2.h>
#include <Wire.h>

#define encoder0PinA 2
#define encoder0PinB 3        //encoder pins

void backward();
void readEncoder();
void forward();
void doEncoder() ;
void Timer2ISR();      //function-prototype declaration


long unsigned int encoder1Value = 0;
long unsigned int encoder2Value = 0;
long unsigned int encoder3Value = 0;
long unsigned int encoder4Value = 0;
long unsigned int target1 = 0;        //original codes

unsigned int second;

volatile byte state = LOW;    //LED state

const byte interruptPin0 = 11;        //slideswitch pins
const byte interruptPin1 = 10;        //slideswitch pins
const byte switcnButtonAB = 20; //A6
const byte switcnButtonC = 5;    //A7 pushbuttonpins

int encoder0Pos = 0;
int encoder1Pos = 0;        //encoder counter
int speed0, target0;        //encoder output

int speakerPin = A0;
int length = 15; // the number of notes
char notes[] = "ccggaagffeeddc "; // a space represents a rest
int beats[] = { 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 2, 4 };
int tempo = 300;                //buzzer

const int rs = 17, en = 16, d4 = 4, d5 = 6, d6 = 7, d7 = 8;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);            //LCD pins


void setup()
  {
```

```
  pinMode(speakerPin, OUTPUT);        //speaker init

  pinMode(switcnButtonAB, INPUT_PULLUP);
  pinMode(switcnButtonC, INPUT);        //switch init

  lcd.begin(16, 2);
  lcd.setCursor(0, 0);
  lcd.print("Please select ");      //LCD init
  lcd.setCursor(0, 1);
  lcd.print("task:");      //LCD init

  pinMode(9, OUTPUT);        //LED init

  unsigned char i = 0;
  pinMode(13, OUTPUT);
  digitalWrite(13, LOW);      //original codes

  Serial.begin(57600);
  Wire.begin();
  delay(1000);                        //original codes

  pinMode(encoder0PinA, INPUT_PULLUP);
  pinMode(encoder0PinB, INPUT_PULLUP);        //encoder init
  attachInterrupt(0, doEncoder, RISING);      //use encoder to set speed0&target0

  int r = 0;
  while ((digitalRead(switcnButtonC)) != HIGH)
 {
    r++;
    Serial.print("r = ");
    Serial.println(r);
    delay(200);
  }         //wait for push button

  delay(500);

  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("TaskC");        //LCD display task type C

  target1 = 0x80000000 + target0;    // set target1

  delay(500);

  lcd.clear();
  lcd.setCursor(1, 1);
  lcd.print(speed0);
  lcd.setCursor(1, 6);
  lcd.print(target0);     //LCD display speed & target

  pinMode(interruptPin0, INPUT);
  pinMode(interruptPin1, INPUT);       // slideswitch init

  int c = 0;
  while ((digitalRead(switcnButtonC)) != HIGH) {
    c++;
    Serial.print("c = ");
    Serial.println(c);
    delay(200);
  }       //wait for push button

  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Input direction:");


  attachPinChangeInterrupt(digitalPinToPCINT(interruptPin0), forward, RISING);
  attachPinChangeInterrupt(digitalPinToPCINT(interruptPin1),       backward,       RISING);
//external intterupts >> foward & backward

  delay(500);

  int a = 0;
  while ((digitalRead(switcnButtonC)) != HIGH) {
    a++;
    Serial.print("a = ");
    Serial.println(a);
    delay(200);    //wait for push button >> led blink & motor could sync
  }

  int FREQ_CTL1 = 1;
  MsTimer2::set((1000 / FREQ_CTL1), Timer2ISR);
  MsTimer2::start();                //LED blinks

}


void loop() {

  delay(10);
  readEncoder();
  Serial.println("Encoder 1 read text!");   // Send some text to the PC
  Serial.println(encoder1Value);   // Send some text to the PC
  Serial.println("Encoder 2 read text!");   // Send some text to the PC
  Serial.println(encoder2Value);   // Send some text to the PC
  Serial.println("Encoder 3 read text!");   // Send some text to the PC
  Serial.println(encoder3Value);   // Send some text to the PC
```

```
    Serial.println("Encoder 4 read text!");    // Send some text to the PC
    Serial.println(encoder4Value);    // Send some text to the PC
    delay(500);

    if (encoder1Value > 0x80040000)
    {
        encoder1Value = 0x80000000;
    }
    else if (encoder1Value < 0x7FFC0000)
    {
        encoder1Value = 0x80000000;
    }


    if (encoder1Value > target1)
    {
        Wire.beginTransmission(42);
        Wire.write("ha");
        Wire.endTransmission();
        MsTimer2::stop();
        while (1);
    }     //Stop all motors & stop blink

}

void readEncoder()
{

    long unsigned int encoder1 = 0;
    long unsigned int encoder2 = 0;
    long unsigned int encoder3 = 0;
    long unsigned int encoder4 = 0;
    Wire.beginTransmission(42);
    Wire.write("i1");
    Wire.endTransmission();
    delay(1);
    Wire.requestFrom(42, 4);
    delay(10);
    if (Wire.available() == 4)
    {
        encoder1 = (long unsigned int) Wire.read();
        encoder1 += ((long unsigned int) Wire.read() << 8);
        encoder1 += ((long unsigned int) Wire.read() << 16);
        encoder1 += ((long unsigned int) Wire.read() << 24);
    }
    encoder1Value = encoder1;
    Serial.println(encoder1Value);

    }
```

```
void backward()
{
   lcd.clear();
   lcd.setCursor(0, 0);
   lcd.print("backward");      //LCD display task type C

   int e = 0;
   while ((digitalRead(switcnButtonC)) != HIGH)
  {
      e++;
      Serial.println(e);
      delay(200);
   }     //wait for push button >> led blink & motor could sync


   interrupts();
   Wire.beginTransmission(42);
   Wire.write("barrrr");

   Wire.write(speed0);
   Wire.write(0);
   Wire.write(speed0);
   Wire.write(0);
   Wire.write(speed0);
   Wire.write(0);
   Wire.write(speed0);
   Wire.write(0);

   Wire.endTransmission();

   while (1) {
      for (int i = 0; i < length; i++) {
         if (notes[i] == ' ') {
            delay(beats[i] * tempo / 100); // rest
         } else {
            playNote(notes[i], beats[i] * tempo);
         }
         // pause between notes
         delay(tempo / 3);
      }//music
   }


}

void forward() {
   lcd.clear();
   lcd.setCursor(0, 0);
```

```
    lcd.print("forward");      //LCD display task type C

    int f = 0;
    while ((digitalRead(switcnButtonC)) != HIGH) {
      f++;
      Serial.println(f);
    }     //wait for push button >> led blink & motor could sync

    interrupts();
    Wire.beginTransmission(42);
    Wire.write("baffff");
    Wire.write(speed0);
    Wire.write(0);
    Wire.write(speed0);
    Wire.write(0);
    Wire.write(speed0);
    Wire.write(0);
    Wire.write(speed0);
    Wire.write(0);
    Wire.endTransmission();


    while (1) {
      for (int i = 0; i < length; i++) {
        if (notes[i] == ' ') {
          delay(beats[i] * tempo); // rest
        } else {
          playNote(notes[i], beats[i] * tempo);
        }

        // pause between notes
        delay(tempo / 2);
      }//music
    }

  }


  void doEncoder()
 {

    if (digitalRead(encoder0PinA) == digitalRead(encoder0PinB))
    {
      encoder0Pos++;
      speed0 = encoder0Pos;
    }
    else
    {
      encoder1Pos++;
```

```
        target0 = (encoder1Pos) * 50;

    }

    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("speed: ");
    lcd.setCursor(7, 0);
    lcd.print(speed0);
    lcd.setCursor(0, 1);
    lcd.print("target: ");
    lcd.setCursor(8, 1);
    lcd.print(target0);
}

void Timer2ISR() {

    state = ! state;
    digitalWrite(9, state);

}


//buzzer
void playTone(int tone, int duration) {
    for (long i = 0; i < duration * 1000L; i += tone * 2) {
        digitalWrite(speakerPin, HIGH);
        delayMicroseconds(tone);
        digitalWrite(speakerPin, LOW);
        delayMicroseconds(tone);
    }
}

void playNote(char note, int duration) {
    char names[] = { 'c', 'd', 'e', 'f', 'g', 'a', 'b', 'C' };
    int tones[] = { 1915, 1700, 1519, 1432, 1275, 1136, 1014, 956 };

    // play the tone corresponding to the note name
    for (int i = 0; i < 8; i++) {
        if (names[i] == note) {
            playTone(tones[i], duration);
        }
    }
}//end
```