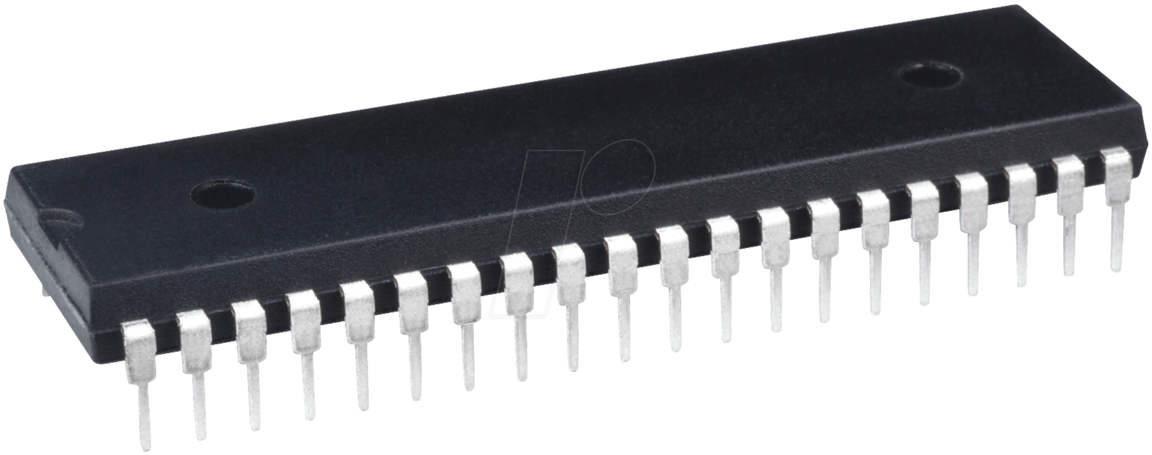


Ping Pong

16.05.2017

—

Md. Jakaria
1305024

Kowshika Sarkar
1305034

Ping Pong Game

. A 2D game resembling table tennis.

Features

Game Modes

1. Single player
2. Dual player

In Single Player mode, AI will play as the opponent.

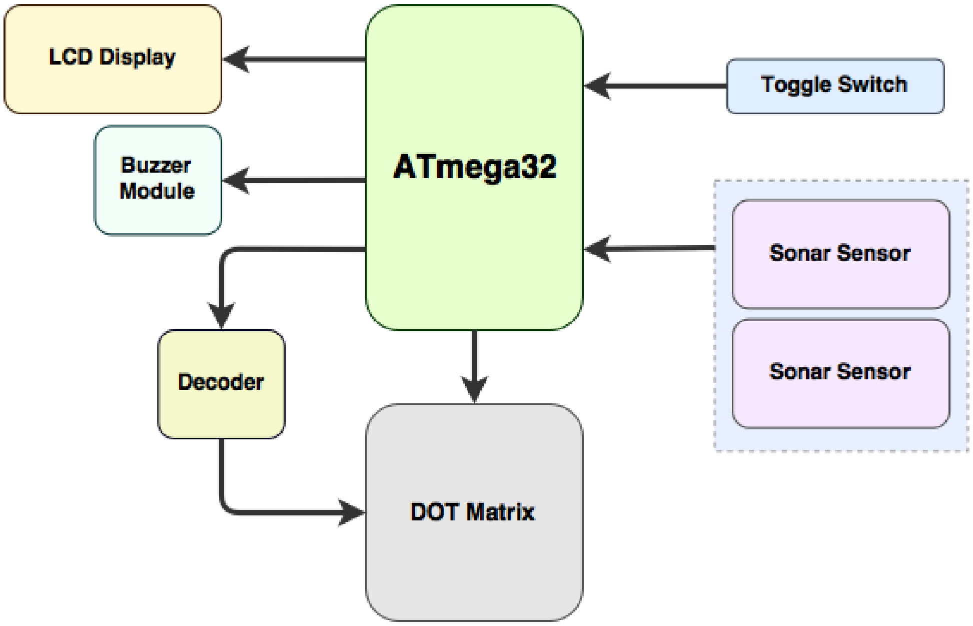
Game Levels

Total 3 levels in the game. The ball will have a greater speed in a higher level. As players' scores increase, the level will rise.

Hand Gesture

Players will move their paddles using hand gestures. To move the paddle to left/right, player has to move his/her hand accordingly.

Block Diagram



Working Principle

Ultrasonic Sensor (HC-Sr04)

Ultrasonic ranging module HC - Sr04 provides 2cm-400cm non-contact measurement function, the ranging accuracy can reach to 3mm.

Basic working principle:

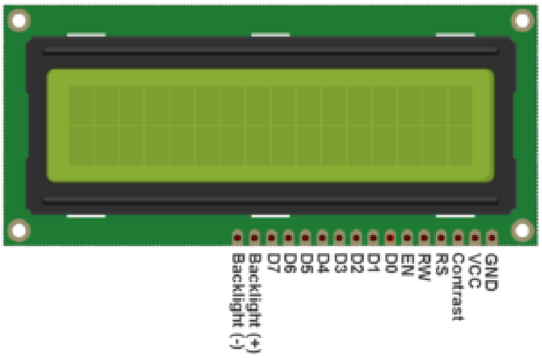
1. Using I/O trigger for at least 10 microseconds high level signal,
2. The module automatically sends eight 40KHz square wave, automatically detects whether any signal return,
3. If the signal is back, through high output, time of high output I/O duration is the time from sending ultrasonic to returning.

   Distance of the object = (high level time ✕ velocity of sound (340 m/s)) / 2

4 Pins

- Vcc (5V)
- Trigger (Pulse input)
- Echo (Pulse output)
- GND (0V)

16×2 LCD Display



- 2 line display
- 16 characters in each line
- 5x7 pixel matrix for each character
- Two registers
  - Command instruction to LCD
  - Data data to display
- 2 modes
  - 8 bit data is passed 8 bits (D0 to D7) at a time
  - 4 bit data is passed 4 bits (D4 to D7) at a time

16 Pins

GND	0v
Vcc	5V
V_{EE}	contrast adjustment by variable resistor
RS	**Logic 0 :** command register;**Logic 1 :** data register
RW	**Logic 0 :** write to register; **Logic 1 :** read from register
EN	sends data to data pins when high to low transition is given
D0 - D7	8-bit data register
Backlight(+)	Led+
Backlight(-)	Led-

We will use a header **lcd.h** to interface LCD display with ATMega32 usig Atmel Studio.

Functions of **lcd.h**

- **Lcd8_Init() & Lcd4_Init() :** These functions will initialize the 16×2 LCD module connected to the microcontroller pins defined by the following constants.

  For 8 Bit Mode :

  ```
  #define D0 eS_PORTD0
  #define D1 eS_PORTD1
  #define D2 eS_PORTD2
  #define D3 eS_PORTD3
  #define D4 eS_PORTD4
  #define D5 eS_PORTD5
  #define D6 eS_PORTD6
  #define D7 eS_PORTD7
  #define RS eS_PORTC6
  #define EN eS_PORTC7
  ```

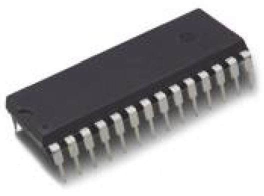
  For 4 Bit Mode :

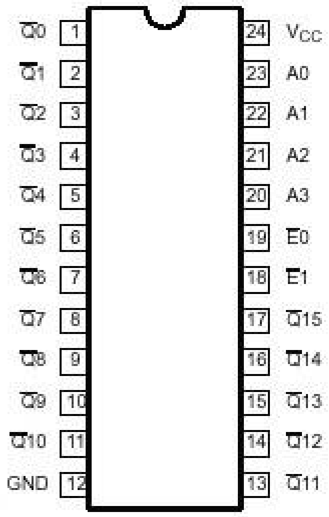
  ```
  #define D4 eS_PORTD4
  #define D5 eS_PORTD5
  #define D6 eS_PORTD6
  #define D7 eS_PORTD7
  #define RS eS_PORTC6
  #define EN eS_PORTC7
  ```

- **Lcd8_Clear() & Lcd4_Clear() :** Calling these functions will clear the 16×2 LCD display screen when interfaced with 8 bit and 4 bit mode respectively.

- **Lcd8_Set_Cursor() & Lcd4_Set_Cursor() :** These function will set the cursor position on the LCD screen by specifying its row and column. By using these functions we can change the position of character and string displayed by the following functions.
- **Lcd8_Write_Char() & Lcd4_Write_Char() :** These functions will write a single character to the LCD screen and the cursor position will be incremented by one.
- **Lcd8_Write_String() & Lcd8_Write_String() :** These function will write string or text to the LCD screen and the cursor position will be incremented by length of the string plus one.
- **Lcd8_Shift_Left() & Lcd4_Shift_Left() :** This function will shift data in the LCD display without changing data in the display RAM.
- **Lcd8_Shift_Right() & Lcd8_Shift_Right() :** This function will shift data in the LCD display without changing data in the display RAM.

Decoder (IC-74154)




$\overline{Q0}$ 1 24 Vcc
$\overline{Q1}$ 2 23 A0
$\overline{Q2}$ 3 22 A1
$\overline{Q3}$ 4 21 A2
$\overline{Q4}$ 5 20 A3
$\overline{Q5}$ 6 19 $\overline{E0}$
$\overline{Q6}$ 7 18 $\overline{E1}$
$\overline{Q7}$ 8 17 $\overline{Q15}$
$\overline{Q8}$ 9 16 $\overline{Q14}$
$\overline{Q9}$ 10 15 $\overline{Q13}$
$\overline{Q10}$ 11 14 $\overline{Q12}$
GND 12 13 $\overline{Q11}$

This IC has **4 input** pins (A, B, C, D), **16 output** pins (0-15), **2 enable** pins (G1, G2), and **2 power** pins (Vcc, GND). Output and enable pins are active low.

When both G1 and G2 are at logic level 0, one of 16 output lines is selected based on 4 input pins, e.g, if $A_3A_2A_1A_0$ is 0000, output Q_0 is selected (logic 0) and other 15 output pins are at logic 1. If either of G1, G2 is at logic 1, no output line is selected i.e, all of them are at logic 1.

DOT Matrix

- A two dimensional array of LEDs.
- **Two ways of configuration :**
  - Row anode-column cathode
  - Row cathode-column anode

We will use 4 8✕8 bicolor matrices to combine into a 16✕16 matrix.

There are 24 pins in each 8✕8 matrix to select LED(s). 8 pins for 8 rows, and 16 pins for columns (2 pins for 2 colors of each column).

Description of pins:

Row4	R4	G4	Row3	R3	G3	Row2	R2	G2	Row1	R1	G1
Row8	R8	G8	Row7	R7	G7	Row6	R6	G6	Row5	R6	G5

Text side here - Upside Down

1) Rows are anode (+ve is to be given). Rows are numbered from up to down.

2) Columns are cathode (GND is to be given). Columns are numbered from left to right (Assuming this time you are facing the LED (front side of the matrix)).

3) RowX denotes row number.

4) GX is used to light up green LEDs in column X.

5) RX is used to light up red LEDs in column X.

For example if we want to light the 2nd column of 3rd row with red, we will have to connect VCC to Row3 pin and GND to R2 pin.

Showing a symbol in dot matrix:

We will use multiplexing to light the LEDs in matrix.

1. Turn on only the topmost row (i.e., provide +5V)

2. Turn on the necessary columns of the topmost row by connecting the corresponding column pins to ground. This lights up the selected LEDs in the topmost row.

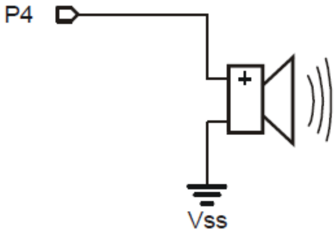
3. Do steps 1 and 2 one by one for all rows and keep repeating.

If we do this slowly, you would see the LED rows turning on one after another. However, if it is done fast enough, human eye will see the whole pattern together. This phenomenon is called persistence of vision.

Buzzer

A buzzer or beeper is an audio signaling device, which may be mechanical, electromechanical, or piezoelectric. Typical uses of buzzers and beepers include alarm devices, timers, and confirmation of user input such as a mouse click or keystroke.

Buzzer Connections:



Here P4 is a logic switch i.e. switch between 5V and 0V.