**ΠΜΣ «Εκπαίδευση STEM & Συστήματα Εκπαιδευτικών Ρομποτικών Διατάξεων»**

Εργαστήριο Εκπαιδευτικής Ρομποτικής Ι *(Arduino)*

**Πηγαίοι κώδικες** από το Φύλλο Εργαστηριακής Άσκησης: **Ήχος** (και ολίγο φως)

*Γνωριμία με το buzzer στην ΠΡΑΞΗ*

**ΠΡΩΤΟ ΒΗΜΑ:**

void setup() {

pinMode(8,OUTPUT);

}

void loop() {

digitalWrite(8,HIGH);

delay(200);

digitalWrite(8,LOW);

delay(200);

}

**ΤΡΙΤΟ ΒΗΜΑ:**

int i;

void setup() {

pinMode(9,OUTPUT);

}

void loop() {

for (i=0;i<=255;i=i+10)

{

analogWrite(9, i);

delay(1000);

}

}

**ΠΕΜΠΤΟ ΒΗΜΑ:**

*int buzzer=8; //set the digital IO pin to control the buzzer*

*void setup()*

*{*

*pinMode(buzzer,OUTPUT); //set digital IO pin as output mode*

*}*

*void loop()*

*{*

*unsigned char i,j; //Define variable*

*while(1)*

*{*

*for(i=0;i<80;i++) //Make sound with same frequency*

*{*

*digitalWrite(buzzer,HIGH); //Make sound*

*delay(1); //Delay 1ms*

*digitalWrite(buzzer,LOW); //No sound*

*delay(1); //Delay 1ms*

*}*

*for(i=0;i<100;i++) // Make sound with other frequency*

*{*

*digitalWrite(buzzer,HIGH); //Make Sound*

*delay(2); //Delay2ms*

*digitalWrite(buzzer,LOW); //No sounds*

*delay(2); //Delay 2ms*

*}*

*}*

*}*

*Σύνδεση με παλιότερες εφαρμογές μας*

ΜΙΑ πιθανή λύση

*int buzzer=8;*

*int zzz;*

*void setup()*

*{*

*pinMode(buzzer,OUTPUT);*

*}*

*void loop()*

*{*

*unsigned char i;*

*zzz=analogRead(A0);*

*if (zzz>100)*

*{*

*for(i=0;i<100;i++)*

*{*

*digitalWrite(buzzer,HIGH);*

*delay(3);*

*digitalWrite(buzzer,LOW);*

*delay(3);*

*}*

*for(i=0;i<50;i++)*

*{*

*digitalWrite(buzzer,HIGH);*

*delay(10);*

*digitalWrite(buzzer,LOW);*

*delay(10);*

*}*

*}*

*}*

ΔΕΥΤΕΡΟ ΜΕΡΟΣ \_ ΜΟΥΣΙΚΗ?

**ΠΡΩΤΟ ΒΗΜΑ**:

#define NOTE\_C3 131

#define NOTE\_D3 147

#define NOTE\_G3 196

#define NOTE\_A3 220

#define NOTE\_B3 247

#define NOTE\_C4 262

#define BUZZER\_PIN 8

int melody[] = {

NOTE\_C4, NOTE\_G3, NOTE\_G3, NOTE\_A3, NOTE\_G3, 0, NOTE\_B3, NOTE\_C4

};

// note durations: 4 = quarter note, 8 = eighth note, etc.:

int noteDurations[] = {

4, 8, 8, 4, 4, 4, 4, 4

};

void setup() {

// iterate over the notes of the melody:

for (int thisNote = 0; thisNote < 8; thisNote++) {

// to calculate the note duration, take one second divided by the note type.

//e.g. quarter note = 1000 / 4, eighth note = 1000/8, etc.

int noteDuration = 1000 / noteDurations[thisNote];

tone(8, melody[thisNote], noteDuration);

// to distinguish the notes, set a minimum time between them.

// the note's duration + 30% seems to work well:

int pauseBetweenNotes = noteDuration \* 1.30;

delay(pauseBetweenNotes);

// stop the tone playing:

noTone(8);

}

}

void loop() {

// no need to repeat the melody.

}

**ΔΕΥΤΕΡΟ ΒΗΜΑ**: Μια πιθανή λύση

#define NOTE\_C3 131

#define NOTE\_D3 147

#define NOTE\_G3 196

#define NOTE\_A3 220

#define NOTE\_B3 247

#define NOTE\_C4 262

#define BUZZER\_PIN 8

int melody[] = {

NOTE\_C4, NOTE\_G3, NOTE\_G3, NOTE\_A3, NOTE\_G3, 0, NOTE\_B3, NOTE\_C4

};

// note durations: 4 = quarter note, 8 = eighth note, etc.:

int noteDurations[] = {

4, 8, 8, 4, 4, 4, 4, 4

};

void setup() {

}

void loop() {

// iterate over the notes of the melody:

for (int thisNote = 0; thisNote < 8; thisNote++) {

// to calculate the note duration, take one second divided by the note type.

//e.g. quarter note = 1000 / 4, eighth note = 1000/8, etc.

int noteDuration = 1000 / noteDurations[thisNote];

tone(8, melody[thisNote], noteDuration);

// to distinguish the notes, set a minimum time between them.

// the note's duration + 30% seems to work well:

int pauseBetweenNotes = noteDuration \* 1.30;

delay(pauseBetweenNotes);

// stop the tone playing:

noTone(8);

}

delay (2000);

}

**ΠΑΡΑΡΤΗΜΑ 1 ΡΟΖ ΠΑΝΘΗΡΑΣ**

/\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Public Constants

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*/

#define NOTE\_B0 31

#define NOTE\_C1 33

#define NOTE\_CS1 35

#define NOTE\_D1 37

#define NOTE\_DS1 39

#define NOTE\_E1 41

#define NOTE\_F1 44

#define NOTE\_FS1 46

#define NOTE\_G1 49

#define NOTE\_GS1 52

#define NOTE\_A1 55

#define NOTE\_AS1 58

#define NOTE\_B1 62

#define NOTE\_C2 65

#define NOTE\_CS2 69

#define NOTE\_D2 73

#define NOTE\_DS2 78

#define NOTE\_E2 82

#define NOTE\_F2 87

#define NOTE\_FS2 93

#define NOTE\_G2 98

#define NOTE\_GS2 104

#define NOTE\_A2 110

#define NOTE\_AS2 117

#define NOTE\_B2 123

#define NOTE\_C3 131

#define NOTE\_CS3 139

#define NOTE\_D3 147

#define NOTE\_DS3 156

#define NOTE\_E3 165

#define NOTE\_F3 175

#define NOTE\_FS3 185

#define NOTE\_G3 196

#define NOTE\_GS3 208

#define NOTE\_A3 220

#define NOTE\_AS3 233

#define NOTE\_B3 247

#define NOTE\_C4 262

#define NOTE\_CS4 277

#define NOTE\_D4 294

#define NOTE\_DS4 311

#define NOTE\_E4 330

#define NOTE\_F4 349

#define NOTE\_FS4 370

#define NOTE\_G4 392

#define NOTE\_GS4 415

#define NOTE\_A4 440

#define NOTE\_AS4 466

#define NOTE\_B4 494

#define NOTE\_C5 523

#define NOTE\_CS5 554

#define NOTE\_D5 587

#define NOTE\_DS5 622

#define NOTE\_E5 659

#define NOTE\_F5 698

#define NOTE\_FS5 740

#define NOTE\_G5 784

#define NOTE\_GS5 831

#define NOTE\_A5 880

#define NOTE\_AS5 932

#define NOTE\_B5 988

#define NOTE\_C6 1047

#define NOTE\_CS6 1109

#define NOTE\_D6 1175

#define NOTE\_DS6 1245

#define NOTE\_E6 1319

#define NOTE\_F6 1397

#define NOTE\_FS6 1480

#define NOTE\_G6 1568

#define NOTE\_GS6 1661

#define NOTE\_A6 1760

#define NOTE\_AS6 1865

#define NOTE\_B6 1976

#define NOTE\_C7 2093

#define NOTE\_CS7 2217

#define NOTE\_D7 2349

#define NOTE\_DS7 2489

#define NOTE\_E7 2637

#define NOTE\_F7 2794

#define NOTE\_FS7 2960

#define NOTE\_G7 3136

#define NOTE\_GS7 3322

#define NOTE\_A7 3520

#define NOTE\_AS7 3729

#define NOTE\_B7 3951

#define NOTE\_C8 4186

#define NOTE\_CS8 4435

#define NOTE\_D8 4699

#define NOTE\_DS8 4978

/\*

Melody

Plays a melody

circuit:

- 8 ohm speaker on digital pin 8

\*/

// #include "pitches.h"

#define BUZZER\_PIN 8

#define REST 0

int melody[] = {

REST, REST, REST, NOTE\_DS4,

NOTE\_E4, REST, NOTE\_FS4, NOTE\_G4, REST, NOTE\_DS4,

NOTE\_E4, NOTE\_FS4, NOTE\_G4, NOTE\_C5, NOTE\_B4, NOTE\_E4, NOTE\_G4, NOTE\_B4,

NOTE\_AS4, NOTE\_A4, NOTE\_G4, NOTE\_E4, NOTE\_D4,

NOTE\_E4, REST, REST, NOTE\_DS4,

NOTE\_E4, REST, NOTE\_FS4, NOTE\_G4, REST, NOTE\_DS4,

NOTE\_E4, NOTE\_FS4, NOTE\_G4, NOTE\_C5, NOTE\_B4, NOTE\_G4, NOTE\_B4, NOTE\_E5,

NOTE\_DS5,

NOTE\_D5, REST, REST, NOTE\_DS4,

NOTE\_E4, REST, NOTE\_FS4, NOTE\_G4, REST, NOTE\_DS4,

NOTE\_E4, NOTE\_FS4, NOTE\_G4, NOTE\_C5, NOTE\_B4, NOTE\_E4, NOTE\_G4, NOTE\_B4,

NOTE\_AS4, NOTE\_A4, NOTE\_G4, NOTE\_E4, NOTE\_D4,

NOTE\_E4, REST,

REST, NOTE\_E5, NOTE\_D5, NOTE\_B4, NOTE\_A4, NOTE\_G4, NOTE\_E4,

NOTE\_AS4, NOTE\_A4, NOTE\_AS4, NOTE\_A4, NOTE\_AS4, NOTE\_A4, NOTE\_AS4, NOTE\_A4,

NOTE\_G4, NOTE\_E4, NOTE\_D4, NOTE\_E4, NOTE\_E4, NOTE\_E4

};

int durations[] = {

2, 4, 8, 8,

4, 8, 8, 4, 8, 8,

8, 8, 8, 8, 8, 8, 8, 8,

2, 16, 16, 16, 16,

2, 4, 8, 4,

4, 8, 8, 4, 8, 8,

8, 8, 8, 8, 8, 8, 8, 8,

1,

2, 4, 8, 8,

4, 8, 8, 4, 8, 8,

8, 8, 8, 8, 8, 8, 8, 8,

2, 16, 16, 16, 16,

4, 4,

4, 8, 8, 8, 8, 8, 8,

16, 8, 16, 8, 16, 8, 16, 8,

16, 16, 16, 16, 16, 2

};

void setup()

{

pinMode(BUZZER\_PIN, OUTPUT);

}

void loop()

{

int size = sizeof(durations) / sizeof(int);

for (int note = 0; note < size; note++) {

//to calculate the note duration, take one second divided by the note type.

//e.g. quarter note = 1000 / 4, eighth note = 1000/8, etc.

int duration = 1000 / durations[note];

tone(BUZZER\_PIN, melody[note], duration);

//to distinguish the notes, set a minimum time between them.

//the note's duration + 30% seems to work well:

int pauseBetweenNotes = duration \* 1.30;

delay(pauseBetweenNotes);

//stop the tone playing:

noTone(BUZZER\_PIN);

}

}