**Arduino code examples used for the unit ‘SMB–Science Magic Box’**

**Program that converts Temperature into Light Intensity**

int RedLedPin = 6; // Connect Led on Digital Pin 6 ; Resistance 1 KOhm

int LEDbrightness;

void setup()

{

Serial.begin(9600);//Set Baud Rate to 9600 bps

pinMode(LED\_BUILTIN, OUTPUT);

pinMode(RedLedPin, OUTPUT);

}

unsigned int intdelay=1000;

void loop()

{

uint16\_t val;

double dat;

uint16\_t InAnalogLight;

val=analogRead(A0);//Connect LM35 on Analog 0

dat = (double) val \* (3.3/10.24); // Vref su Due 3.3 V

Serial.print("Temp:"); //Display the temperature on Serial monitor

Serial.print(dat);

Serial.println("°C");

Serial.println(val);

delay(intdelay);

LEDbrightness = map(val, 60, 85, 0, 255);

analogWrite(RedLedPin, LEDbrightness); //Analog Write to Digital PWM pin

delay(200);

digitalWrite(LED\_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)

delay(intdelay); // wait for a second

digitalWrite(LED\_BUILTIN, LOW); // turn the LED off by making the voltage LOW

}

**Program that reads the pressure force sensor on analogue pin A0, transduces the signal to a PWM-modulated Buzzer for an audio signal and into a light signal on a LED, controlled in intensity thanks to the PWM feature on digital out.**

int buzzPin = 8; //Connect Buzzer on Digital Pin3

int RedLedPin = 6; // Connect Led on Digital Pin 6 ; Resistance 1 KOhm

int LEDbrightness;

#define NOTE\_B3 493

#define NOTE\_C4 523

#define NOTE\_D4 587

#define NOTE\_E4 659

#define NOTE\_F4 698

#define NOTE\_G4 783

#define NOTE\_A4 880

#define NOTE\_B4 987

#define NOTE\_C5 1046

#define NOTE\_D5 1174

#define NOTE\_E5 1318

#define NOTE\_F5 1318

#define NOTE\_DURATION 1000

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

int noteDurations[] = {

2, 2, 2, 2,2,2,2,2 };

uint16\_t prova;

int fsrPin = 0; // the FSR and 10K pulldown are connected to a0

int fsrReading; // the analog reading from the FSR resistor divider

void setup(void) {

Serial.begin(9600);

pinMode(LED\_BUILTIN, OUTPUT);

pinMode(buzzPin, OUTPUT);

pinMode(RedLedPin, OUTPUT);

}

void loop(void) {

fsrReading = analogRead(fsrPin);

delay(10);

fsrReading = analogRead(fsrPin);

digitalWrite(LED\_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)

LEDbrightness = map(fsrReading, 0, 1023, 0, 255);

// LED gets brighter the harder you press

analogWrite(RedLedPin, LEDbrightness); //Analog Write to Digital PWM pin

prova= pulseIn(1,HIGH);

Serial.print("Analog reading = ");

Serial.print(fsrReading); // the raw analog reading

if (fsrReading == 0) {

Serial.println(" - No pressure");

} else if (fsrReading < 50) {

Serial.println(" - Light touch");

tone(8, NOTE\_B3,NOTE\_DURATION);

} else if (fsrReading < 100) {

Serial.println(" - Light touch");

tone(8, NOTE\_C4,NOTE\_DURATION);

} else if (fsrReading < 200) {

Serial.println(" - Light touch");

tone(8, NOTE\_D4,NOTE\_DURATION);

} else if (fsrReading < 300) {

Serial.println(" - Light touch");

tone(8, NOTE\_E4, NOTE\_DURATION);

} else if (fsrReading < 400) {

tone(8, NOTE\_F4,NOTE\_DURATION);

Serial.println(" - Light squeeze");

} else if (fsrReading < 500) {

tone(8, NOTE\_G4,NOTE\_DURATION);

Serial.println(" - Medium squeeze");

} else if (fsrReading < 600) {

tone(8, NOTE\_A4,NOTE\_DURATION);

Serial.println(" - Medium squeeze");

} else if (fsrReading < 700) {

tone(8, NOTE\_B4,NOTE\_DURATION);

Serial.println(" - Medium squeeze");

} else if (fsrReading < 800) {

tone(8, NOTE\_C5,NOTE\_DURATION);

Serial.println(" - Medium squeeze");

} else if (fsrReading < 900) {

tone(8, NOTE\_D5,NOTE\_DURATION);

Serial.println(" - Medium squeeze");

} else if (fsrReading < 1000) {

tone(8, NOTE\_E5,NOTE\_DURATION);

Serial.println(" - Medium squeeze");

} else {

tone(8, NOTE\_F5,NOTE\_DURATION);

Serial.println(" - Big squeeze");

}

// delay(300);

// noTone(8);

digitalWrite(LED\_BUILTIN, LOW); // turn the LED off by making the voltage LOW

// delay(200); // wait for a second

}

/\*

Tone generator

v1 use timer, and toggle any digital pin in ISR

funky duration from arduino version

TODO use FindMckDivisor?

timer selected will preclude using associated pins for PWM etc.

could also do timer/pwm hardware toggle where caller controls duration

\*/

// timers TC0 TC1 TC2 channels 0-2 ids 0-2 3-5 6-8 AB 0 1

// use TC1 channel 0

#define TONE\_TIMER TC1

#define TONE\_CHNL 0

#define TONE\_IRQ TC3\_IRQn

// TIMER\_CLOCK4 84MHz/128 with 16 bit counter give 10 Hz to 656KHz

// piano 27Hz to 4KHz

static uint8\_t pinEnabled[PINS\_COUNT];

static uint8\_t TCChanEnabled = 0;

static boolean pin\_state = false ;

static Tc \*chTC = TONE\_TIMER;

static uint32\_t chNo = TONE\_CHNL;

volatile static int32\_t toggle\_count;

static uint32\_t tone\_pin;

// frequency (in hertz) and duration (in milliseconds).

void tone(uint32\_t ulPin, uint32\_t frequency, int32\_t duration)

{

const uint32\_t rc = VARIANT\_MCK / 256 / frequency;

tone\_pin = ulPin;

toggle\_count = 0; // strange wipe out previous duration

if (duration > 0 ) toggle\_count = 2 \* frequency \* duration / 1000;

else toggle\_count = -1;

if (!TCChanEnabled) {

pmc\_set\_writeprotect(false);

pmc\_enable\_periph\_clk((uint32\_t)TONE\_IRQ);

TC\_Configure(chTC, chNo,

TC\_CMR\_TCCLKS\_TIMER\_CLOCK4 |

TC\_CMR\_WAVE | // Waveform mode

TC\_CMR\_WAVSEL\_UP\_RC ); // Counter running up and reset when equals to RC

chTC->TC\_CHANNEL[chNo].TC\_IER=TC\_IER\_CPCS; // RC compare interrupt

chTC->TC\_CHANNEL[chNo].TC\_IDR=~TC\_IER\_CPCS;

NVIC\_EnableIRQ(TONE\_IRQ);

TCChanEnabled = 1;

}

if (!pinEnabled[ulPin]) {

pinMode(ulPin, OUTPUT);

pinEnabled[ulPin] = 1;

}

TC\_Stop(chTC, chNo);

TC\_SetRC(chTC, chNo, rc); // set frequency

TC\_Start(chTC, chNo);

}

void noTone(uint32\_t ulPin)

{

TC\_Stop(chTC, chNo); // stop timer

digitalWrite(ulPin,LOW); // no signal on pin

}

// timer ISR TC1 ch 0

void TC3\_Handler ( void ) {

TC\_GetStatus(TC1, 0);

if (toggle\_count != 0){

// toggle pin TODO better

digitalWrite(tone\_pin,pin\_state= !pin\_state);

if (toggle\_count > 0) toggle\_count--;

} else {

noTone(tone\_pin);

}

}

**The following program converts an input environment noise signal into a light signal, using LEDs with different colours, turning on LEDs according to noise intensity, being RED, YELLOW and GREEN used in this order for increasing noise level.**

int PIN\_LED\_GREEN = 4; // Connect Led on Digital Pin 6 ; Resistance 1 KOhm

int PIN\_LED\_RED = 6; // Connect Led on Digital Pin 6 ; Resistance 1 KOhm

int PIN\_LED\_YELLOW = 7; // Connect Led on Digital Pin 6 ; Resistance 1 KOhm

void setup()

{

Serial.begin(9600);//Set Baud Rate to 9600 bps

pinMode(LED\_BUILTIN, OUTPUT);

pinMode(PIN\_LED\_RED, OUTPUT);

pinMode(PIN\_LED\_GREEN, OUTPUT);

pinMode(PIN\_LED\_YELLOW, OUTPUT);

}

unsigned int intdelay=200;

void loop()

{

uint16\_t val;

double dat;

uint16\_t InAnalogNoise;

// The trick when using multiple analog sensors is to read them twice,

// with a small delay after each read (10ms is good), then discard the first reading.

// This is because the ADC multiplexer needs switching time and the voltage needs time to stabilize after switching..

// Basically the first analogRead call causes the multiplexer to switch, the delay gives the voltage time to stabilize,

// then your second read should be much more accurate with less jitter.

InAnalogNoise=analogRead(0); //connect mic sensor to Analog 0

delay(10);

InAnalogNoise=analogRead(0); //connect mic sensor to Analog 0

Serial.print("Noise : ");

Serial.println(InAnalogNoise,DEC);//print the sound value to serial

Serial.println();

delay(intdelay);

if (InAnalogNoise>100)

{

digitalWrite(PIN\_LED\_RED, HIGH);

if (InAnalogNoise>400)

{

digitalWrite(PIN\_LED\_YELLOW, HIGH);

if (InAnalogNoise>700)

{

digitalWrite(PIN\_LED\_GREEN, HIGH);

}

else

{

digitalWrite(PIN\_LED\_GREEN, LOW);

}

}

else

{

digitalWrite(PIN\_LED\_YELLOW, LOW);

}

}

else

{

digitalWrite(PIN\_LED\_RED, LOW);

digitalWrite(PIN\_LED\_YELLOW, LOW);

digitalWrite(PIN\_LED\_GREEN, LOW);

}

delay(80);

digitalWrite(LED\_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)

delay(intdelay); // wait for a second

digitalWrite(LED\_BUILTIN, LOW); // turn the LED off by making the voltage LOW

}

**The following program reads Ambient Analog Light sensor from A0 Analog Pin used as INPUT. Transduces the signal into an acoustic signal on Buzzer connected on Digital PIN 8 frequency modulated thanks to its PWM feature. Transduces the signal into a light intensity signal on digital pin 6 connected RED LED, written as ANALOG OUTPUT USING DIGITAL PIN PWM feature: more light from LED, when ambient luminosity decreases. Issues a sound with increasing frequency when light intensity increases (see the above code for sound emission).**

int PIN\_LED\_RED = 6; // Connect Led on Digital Pin 6 ; Resistance 1 KOhm

int buzzPin = 8; //Connect Buzzer on Digital Pin3

int RedLedPin = 6; // Connect Led on Digital Pin 6 ; Resistance 1 KOhm

int LEDbrightness;

// Musical notes frequencies mapping

#define NOTE\_B3 493

#define NOTE\_C4 523

#define NOTE\_D4 587

#define NOTE\_E4 659

#define NOTE\_F4 698

#define NOTE\_G4 783

#define NOTE\_A4 880

#define NOTE\_B4 987

#define NOTE\_C5 1046

#define NOTE\_D5 1174

#define NOTE\_E5 1318

#define NOTE\_F5 1318

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

int noteDurations[] = {

2, 2, 2, 2,2,2,2,2 };

uint16\_t prova;

int fsrPin = 0; // the FSR and 10K pulldown are connected to a0

int fsrReading; // the analog reading from the FSR resistor divider

void setup(void) {

Serial.begin(9600);

pinMode(LED\_BUILTIN, OUTPUT);

pinMode(buzzPin, OUTPUT);

pinMode(RedLedPin, OUTPUT);

}

void loop(void) {

fsrReading = analogRead(fsrPin);

delay(10);

fsrReading = analogRead(fsrPin);

prova= pulseIn(1,HIGH);

Serial.print("Analog reading = ");

Serial.println(fsrReading); // the raw analog reading

LEDbrightness = map(fsrReading, 0, 1023, 255, 0);

// LED gets brighter the less luminosity you get from ambient

analogWrite(RedLedPin, LEDbrightness); //Analog Write to Digital PWM pin

Serial.print("Luce = ");

Serial.println(LEDbrightness);

digitalWrite(LED\_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)

delay(200); // wait for 200 msec

if (fsrReading <= 100) {

} else if (fsrReading < 200) {

tone(8, NOTE\_B3,500);

} else if (fsrReading < 300) {

tone(8, NOTE\_C4,500);

} else if (fsrReading < 400) {

tone(8, NOTE\_D4,500);

} else if (fsrReading < 500) {

tone(8, NOTE\_E4,500);

} else if (fsrReading < 600) {

tone(8, NOTE\_F4,500);

} else if (fsrReading < 700) {

tone(8, NOTE\_G4,500);

} else if (fsrReading < 800) {

tone(8, NOTE\_A4,500);

} else if (fsrReading < 900) {

tone(8, NOTE\_B4,500);

} else if (fsrReading < 1000) {

tone(8, NOTE\_C5,500);

} else {

tone(8, NOTE\_D5,500);

}

delay(300);

noTone(8);

digitalWrite(LED\_BUILTIN, LOW); // turn the LED off by making the voltage LOW

delay(200); // wait for a second

}