

```

/*
 * Euro Mega Control
 * A red case with Euro size cards for interfacing with Analog Synths
 * Contains: 6 Pulse Inputs, 6 Pulse Outputs, 6 Control Voltage Inputs
 *           (with Envelope Followers), 6 Control Voltage Outputs
 *           (RC filtered PWM), MIDI IN, MIDI OUT, 8-bit DAC Out,
 *           Diode Gate.
 *
 */

//~~~~~
//                               CONSTANTS and Variables
//~~~~~

#include <MIDI.h> //version 4.x
MIDI_CREATE_INSTANCE(HardwareSerial, Serial1, MIDI); //see midi_Defs.h

const int MIDI_TX = 18; //on SERIAL1
const int MIDI_RX = 19; //on SERIAL1

const int PULSE_IN1 = 2; //Interrupt 4
const int PULSE_IN2 = 3; //Interrupt 5
const int PULSE_IN3 = 16;
const int PULSE_IN4 = 17;
const int PULSE_IN5 = 20; //Interrupt 0
const int PULSE_IN6 = 21; //Interrupt 1

const int PULSE_OUT1 = 9;
const int PULSE_OUT2 = 10;
const int PULSE_OUT3 = 12;
const int PULSE_OUT4 = 13;
const int PULSE_OUT5 = 14;
const int PULSE_OUT6 = 15;

const int CV_IN1 = 8; //Analog inputs for analogRead();
const int CV_IN2 = 9;
const int CV_IN3 = 10;
const int CV_IN4 = 11;
const int CV_IN5 = 12;
const int CV_IN6 = 13;

const int SIGNAL_IN1 = 14; //2 Signal Inputs Jacks, either side of Reset Button
const int SIGNAL_IN2 = 15; //Biased at 2.5v for 0-5v Analog Inputs

const int CV_OUT1 = 4; //Timer0
const int CV_OUT2 = 5; //Timer3
const int CV_OUT3 = 6; //Timer4
const int CV_OUT4 = 7; //Timer4
const int CV_OUT5 = 8; //Timer4
const int CV_OUT6 = 11; //Timer

const int DAC_WR = 37; //AD7224 DAC Write pulse (active low)
// DAC inputs on A0 through A7, register F

const int GATE_IN1 = 39; //Two Arduino Inputs to Diode Gate
const int GATE_IN2 = 38; //3 more external Gate inputs surround the Reset Button

//~~~~~
//                               Callback MIDI_In Test function
//~~~~~

```

```

// user created Callback Function for MIDI Input Test

void myHandleNoteOn(byte channel, byte note, byte velocity){

    int x = analogRead(CV_IN1); //CV_IN1 pot sets arpeggio speed
    int y = analogRead(CV_IN2); //CV_IN2 pot sets arpeggio pitch range
    y = map(y, 0, 900, 0, 20);

    MIDI.sendNoteOn(note, velocity, 1);
    delay(x);
    MIDI.sendNoteOn(note + y, velocity, 1);
    delay(x);
    MIDI.sendNoteOn(note + y + y, velocity, 1);
    delay(x);

    MIDI.sendNoteOff(note + y, velocity, 1);
    MIDI.sendNoteOff(note + y + y, velocity, 1);
    MIDI.sendNoteOff(note, velocity, 1);
}

//~~~~~
//                               SETUP()
//~~~~~

void setup() {

    //Serial.begin(9600);

    MIDI.setHandleNoteOn(myHandleNoteOn); //for Callback MIDI In Test
    MIDI.begin(MIDI_CHANNEL_OMNI);

    pinMode(PULSE_IN1, INPUT_PULLUP);
    pinMode(PULSE_IN2, INPUT_PULLUP);
    pinMode(PULSE_IN3, INPUT_PULLUP);
    pinMode(PULSE_IN4, INPUT_PULLUP);
    pinMode(PULSE_IN5, INPUT_PULLUP);
    pinMode(PULSE_IN6, INPUT_PULLUP);

    pinMode(PULSE_OUT1, OUTPUT);
    pinMode(PULSE_OUT2, OUTPUT);
    pinMode(PULSE_OUT3, OUTPUT);
    pinMode(PULSE_OUT4, OUTPUT);
    pinMode(PULSE_OUT5, OUTPUT);
    pinMode(PULSE_OUT6, OUTPUT);

    digitalWrite(PULSE_OUT1, HIGH); //Final Jack Outputs are inverted
    digitalWrite(PULSE_OUT2, HIGH); //LEDs are inverted, Low jack output is On
    digitalWrite(PULSE_OUT3, HIGH);
    digitalWrite(PULSE_OUT4, HIGH);
    digitalWrite(PULSE_OUT5, HIGH);
    digitalWrite(PULSE_OUT6, HIGH);

    pinMode(A0, OUTPUT); //8-bit DAC data inputs on Port F
    pinMode(A1, OUTPUT);
    pinMode(A2, OUTPUT);
    pinMode(A3, OUTPUT);
    pinMode(A4, OUTPUT);
    pinMode(A5, OUTPUT);
    pinMode(A6, OUTPUT);
    pinMode(A7, OUTPUT);

    pinMode(DAC_WR, OUTPUT); //DAC write pulse, active low
    digitalWrite(DAC_WR, HIGH); //normally high

```

```

    pinMode(GATE_IN1, OUTPUT); //Diode OR Gate inputs
    pinMode(GATE_IN2, OUTPUT);

    digitalWrite(GATE_IN1, LOW); //a HIGH closes the gate to any other input signal
    digitalWrite(GATE_IN2, LOW);

}

//~~~~~
//          LOOP() --- uncomment (/*---*/) the test section you want to try
//~~~~~

void loop() {

    //Test Pulse Inputs
    /*
    Serial.print(digitalRead(PULSE_IN1));
    Serial.print(digitalRead(PULSE_IN2));
    Serial.print(digitalRead(PULSE_IN3));
    Serial.print(digitalRead(PULSE_IN4));
    Serial.print(digitalRead(PULSE_IN5));
    Serial.print(digitalRead(PULSE_IN6));
    Serial.println();
    */
    // Test CV Inputs
    /*
    Serial.print(analogRead(CV_IN1));
    Serial.print("\t");
    Serial.print(analogRead(CV_IN2));
    Serial.print("\t");
    Serial.print(analogRead(CV_IN3));
    Serial.print("\t");
    Serial.print(analogRead(CV_IN4));
    Serial.print("\t");
    Serial.print(analogRead(CV_IN5));
    Serial.print("\t");
    Serial.print(analogRead(CV_IN6));
    Serial.print("\t");
    Serial.println();
    */

    //Test Pulse Outs. Watch the LEDs.
    /*
    digitalWrite(PULSE_OUT1, HIGH);
    delay(20);
    digitalWrite(PULSE_OUT2, HIGH);
    delay(20);
    digitalWrite(PULSE_OUT3, HIGH);
    delay(20);
    digitalWrite(PULSE_OUT4, HIGH);
    delay(20);
    digitalWrite(PULSE_OUT5, HIGH);
    delay(20);
    digitalWrite(PULSE_OUT6, HIGH);
    delay(50);
    digitalWrite(PULSE_OUT1, LOW);
    digitalWrite(PULSE_OUT2, LOW);
    digitalWrite(PULSE_OUT3, LOW);
    digitalWrite(PULSE_OUT4, LOW);
    digitalWrite(PULSE_OUT5, LOW);
    digitalWrite(PULSE_OUT6, LOW);
    */
}

```

```

delay(500);
*/

//Test CV Outputs and Inputs. Connect CV out in CV ins.

/*
for (int i=0; i<256; i++){
  analogWrite(CV_OUT1, i);
  analogWrite(CV_OUT2, i);
  analogWrite(CV_OUT3, i);
  analogWrite(CV_OUT4, i);
  analogWrite(CV_OUT5, i);
  analogWrite(CV_OUT6, i);

  Serial.print(analogRead(CV_IN1));
  Serial.print("\t");
  Serial.print(analogRead(CV_IN2));
  Serial.print("\t");
  Serial.print(analogRead(CV_IN3));
  Serial.print("\t");
  Serial.print(analogRead(CV_IN4));
  Serial.print("\t");
  Serial.print(analogRead(CV_IN5));
  Serial.print("\t");
  Serial.print(analogRead(CV_IN6));
  Serial.print("\t");
  Serial.println();
  delay(200);
}
*/

// Test Diode Gate with Arduino outputs, top 2 CV_IN pots control frequencies.

/*
tone(GATE_IN1, analogRead(CV_IN1));
int mod_val = analogRead(CV_IN2);
mod_val = mod_val + 10;
digitalWrite(GATE_IN2, HIGH);
delayMicroseconds(mod_val);
digitalWrite(GATE_IN2, LOW);
delayMicroseconds(mod_val);
*/

//Test 8-Bit DAC, with 32 step staircase waveform
//Processor is not fast enough to do 256 steps with analogRead setting freq.

/*
for (int i=0; i<32; i++){ // 32 step staircase
  PORTF = i << 3;
  digitalWrite(DAC_WR, LOW);
  digitalWrite(DAC_WR, HIGH);
  delayMicroseconds(analogRead(CV_IN1)); //frequency set with CV_IN1 pot
}
*/

// Test MIDI Output. Uses MIDI Library v4 or later.

/*
// before setup() use: #include <MIDI.h>
// and MIDI_CREATE_INSTANCE(HardwareSerial, Serial1, MIDI);
// in setup() use: MIDI.begin(MIDI_CHANNEL_OMNI);

// CV_IN1 is duration, CV_IN2 is note velocity,

```

```

// PULSE_IN1 is Program Change, PULSE_IN2 is all notes Off
for (int i=30; i<60; i++){ //play notes going up

    int dur = analogRead(CV_IN1) + 20; // note duration from CV_IN1 pot
    int velocity = analogRead(CV_IN2); // note velocity from CV_IN2 pot
    velocity = map(velocity, 0, 1000, 5, 127);

    MIDI.sendNoteOn(i, velocity, 1); //main note_on, note_off
    delay(dur);
    MIDI.sendNoteOff(i, velocity, 1);
    delay(dur);

    if (!digitalRead(PULSE_IN2)){ //All notes off from PULSE_IN2 button
        for (int j=0; j<127; j++){
            MIDI.sendNoteOff(i, 0, 1);
        }
    }

    if (!digitalRead(PULSE_IN1)){ //Random Program Change from PULSE_IN1 button
        int prog = random(0, 127);
        MIDI.sendProgramChange(prog, 1);
    }
}
*/

// Test MIDI Input/Output with MIDI Callback

/*
// On MIDI.read() MIDI class will call Callback functions.
// User created callback function myHandleNoteOn() in section before setup()
// and MIDI.setHandleNoteOn(myHandleNoteOn) in setup() section

MIDI.read();
*/

} // end of loop()

```