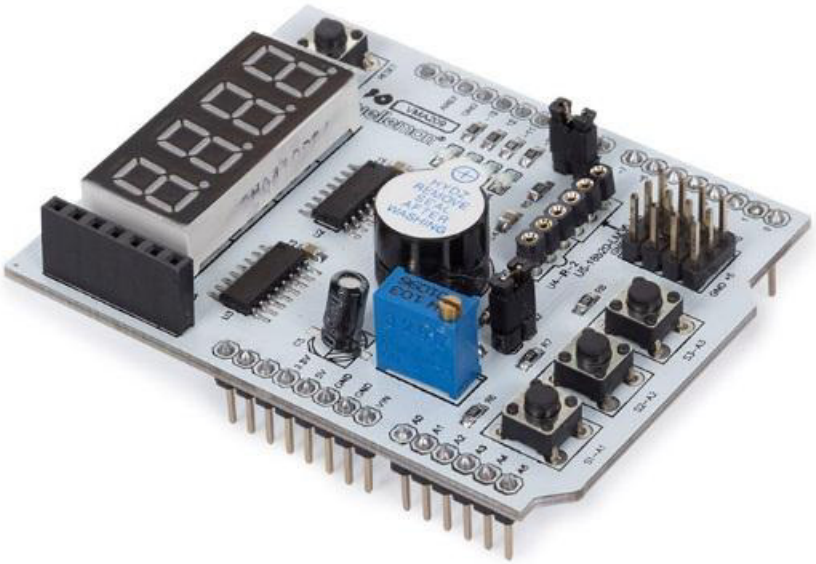
VMA209

MULTI-FUNCTION SHIELD EXPANSION BOARD FOR ARDUINO®



USER MANUAL

USER MANUAL

1. Introduction

To all residents of the European Union

Important environmental information about this product

This symbol on the device or the package indicates that disposal of the device after its lifecycle could harm the environment. Do not dispose of the unit (or batteries) as unsorted municipal waste; it should be taken to a specialized company for recycling. This device should be returned to your distributor or to a local recycling service. Respect the local environmental rules.

If in doubt, contact your local waste disposal authorities.

Thank you for choosing Velleman®! Please read the manual thoroughly before bringing this device into service. If the device was damaged in transit, do not install or use it and contact your dealer.

2. Safety Instructions

- This device can be used by children aged from 8 years and above, and persons with reduced physical, sensory or mental capabilities or lack of experience and knowledge if they have been given supervision or instruction concerning the use of the device in a safe way and understand the hazards involved. Children shall not play with the device. Cleaning and user maintenance shall not be made by children without supervision.

- Indoor use only.
  Keep away from rain, moisture, splashing and dripping liquids.

3. General Guidelines

- Refer to the Velleman® Service and Quality Warranty on the last pages of this manual.
- Familiarise yourself with the functions of the device before actually using it.
- All modifications of the device are forbidden for safety reasons. Damage caused by user modifications to the device is not covered by the warranty.
- Only use the device for its intended purpose. Using the device in an unauthorised way will void the warranty.
- Damage caused by disregard of certain guidelines in this manual is not covered by the warranty and the dealer will not accept responsibility for any ensuing defects or problems.
- Nor Velleman nv nor its dealers can be held responsible for any damage (extraordinary, incidental or indirect) – of any nature (financial, physical…) arising from the possession, use or failure of this product.
- Due to constant product improvements, the actual product appearance might differ from the shown images.
- Product images are for illustrative purposes only.
- Do not switch the device on immediately after it has been exposed to changes in temperature. Protect the device against damage by leaving it switched off until it has reached room temperature.
- Keep this manual for future reference.

4. What is Arduino®

Arduino® is an open-source prototyping platform based in easy-to-use hardware and software. Arduino® boards are able to read inputs – light-on sensor, a finger on a button or a Twitter message – and turn it into an output – activating of a motor, turning on an LED, publishing something online. You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so, you use the Arduino programming language (based on Wiring) and the Arduino® software IDE (based on Processing).

Surf to www.arduino.cc and www.arduino.org for more information.

5. Overview

dimensions ..... 69 x 54 x 11 mm
weight ..... 27 g

6. Connection

VMA209	Arduino®
10, 11, 12, 13	4 red LEDs
A1, A2, A3	3 buttons + reset button
A0	potentiometer (10 kΩ)
latch 4, clock 7, data 8	4-digit, 7-segment LED tube driven by 74HC595
3 (digital on-off)	buzzer
2	socket for IR receiver (remote control)
A4	socket for temperature sensor LM35 or DS18B20 (polarity!)
GND, +5V, 0, 1 (RX/TX)	header for APC220 shield
5, 6, 9, A5	free pins (PWM)

7. Examples

7.1 Blinking LEDs

```
//********************************************************************************
//Flashing LEDS on the Velleman VMA209
//Programmed by : Arduino IDE
//Compatible with  : Arduino Leonardo, Arduino UNO, MEGA
//********************************************************************************

char ledPin =  10; //digital pin 10 -> LED1 (D4)
char ledPin1 = 11; //digital pin 11 -> LED2 (D3)
char ledPin2 = 12; //digital pin 12 -> LED2 (D2)
char ledPin3 = 13; //digital pin 13 -> LED2 (D1)
void setup()
{
pinMode(ledPin, OUTPUT);//declare LedPin as output
pinMode(ledPin1, OUTPUT);
pinMode(ledPin2, OUTPUT);
pinMode(ledPin3, OUTPUT);
}
void loop()
{
digitalWrite(ledPin, HIGH); //Switch ON this LED
digitalWrite(ledPin1, HIGH);
digitalWrite(ledPin2, HIGH);
digitalWrite(ledPin3, HIGH);
delay(1000); //Wait for 1 second
digitalWrite(ledPin, LOW); //Switch OFF this LED
```

```
digitalWrite(ledPin1, LOW);
digitalWrite(ledPin2, LOW);
digitalWrite(ledPin3, LOW);
delay(1000); // Wait for 1 second
}
```

7.2 Running LEDs

```
//*****************************************************************************
//VMA209 RUNNING LED EXAMPLE
//Written on : Arduino IDE
//Compatible with  : Arduino Leonardo, Arduino UNO, Mega
//*****************************************************************************/
int BASE = 10 ;  //Digital Pin Base = 10, 10 = corresponding with D10
int NUM = 4;   // 4 Digital lines are used for 4 LEDs

void setup()
{
  for (int i = BASE; i < BASE + NUM; i ++)
  {
    pinMode(i, OUTPUT);   //declare digital pins 10 to 13 as output
  }
}
```

7.3 Push-Button and LED Test

```
//*****************************************************
//  VMA209 Push button and LED test
//  written by Patrick De Coninck / Velleman NV.
//  VMA209 contains 3 Push buttons, they are connected to the Arduino Analog inputs A1, A2, A3
//  in this example we will switch ON LED1 when pushing Push button 3 - please feel free to choose different
buttons or LEDS
//*****************************************************



int ledpin=13; //Define integer ledpin with a value of 13
int inpin=A3;  //Define integer inpin = analog line A3
int val;  // define variable VAL
void setup()
{
pinMode(ledpin,OUTPUT);//Declare ledpin (which has a value of 13) as OUTPUT
pinMode(inpin,INPUT);//Declare inpin (which is analog input A3) as INPUT
}
void loop()
{
val=digitalRead(inpin);//Read the value of Analog line 13 (push button)
```

```
if(val==LOW)          //If this value is LOW:
{ digitalWrite(ledpin,LOW);} // then ledpin (the led on digital line 13) is also LOW (off)
else
{ digitalWrite(ledpin,HIGH);} // in the other case (ledpin is not low) switch ON the LED on D13
}

void loop()
{
  for (int i = BASE; i < BASE + NUM; i ++)
  {
    digitalWrite(i, LOW);    //Switch OFF Digital lines 10 to 13 one by one
    delay(200);       // wait 0,2 seconds
  }
  for (int i = BASE; i < BASE + NUM; i ++)
  {
    digitalWrite(i, HIGH);    //Switch ON Digital lines 10 to 13 one by one
    delay(200);       //wait 0,2 seconds
  }
}
```

7.4 LED Start-Stop

```
//*****************************************************************************
//  VMA209 - LED START-STOP BY PUSH BUTTON EXAMPLE
//  function : press S1, LED D1 will light -- press S1 again, LED D1 will switch off
//  programmed on:Arduino IDE---------------------------------------
//  compatible with Arduino UNO, MEGA
//*****************************************************************************

#define LED 13  // LED is on digital 13, another value between line 10 and 13 can be chosen ! Just give it a try !
#define KEY A1  // we choose one of the available push buttons which are on A1, A2 or A3. In this case we choose A1 but You can try another one

int KEY_NUM = 0;

void setup()
{
  pinMode(LED,OUTPUT);                    // initialize LED (D13) as output
  pinMode(KEY,INPUT_PULLUP);              //initialize KEY (analog pin A1) as an input with the internal pull-up resistor enabled
}

void loop()
{
  ScanKey();            // check if there is a key pressed (see void ScanKey)
  if(KEY_NUM == 1)      // key 1 pressed
  {
    digitalWrite(LED,!digitalRead(LED));// Inverte the LED status
  }
}
```

```
void ScanKey() // ScanKey routine
{
 KEY_NUM = 0;
 if(digitalRead(KEY) == LOW)
 {
  delay(20);                                          // anti-bounce delay , this is the minimuum time You have to press the button
  if(digitalRead(KEY) == LOW)
  {
   KEY_NUM = 1;
   while(digitalRead(KEY) == LOW);
  }
 }
}
```

7.5 Potentiometer

```
//*****************************************************************************
//  VMA209 - Pot meter example
//  The VMA209 contains a blue potmeter (trimmer) which is connected to Analog 0
//  In this example we show You how to measure the voltage between 0 and 5V, and how to visualise it by the serial monitor
//  Programmed :Arduino IDE
//  Board  : Arduino Leonardo, UNO, MEGA
//*****************************************************************************/
#define Pot A0                    //declare Analog line 0 as Pot

int PotBuffer = 0;                //initialize variable PotBuffer as integer

void setup()
{
 Serial.begin(9600);                    //Set serial port to 9600 Baud
}

void loop()
{
 PotBuffer = analogRead(Pot); // Read the value of Pot (A0) and store it into PotBuffer
 Serial.print("Pot = ");// Write "Pot = " to the serial monitor
 Serial.println(PotBuffer);    // Now print the actual value measured by A0 (Pot or PotBuffer), TURN the small screw on top of the blue trimmer and You will see a value between 0 and about 1000
                        //
                        // This means that You have to devide PotBuffer by 200 to have the actual voltage on A0 (Serial.println(PotBuffer/200)), the variable Potbuffer , which is declared as Integer (int) should
                        // then be declared as Floating point variable or : float PotBuffer = 0;

 delay(500);                                                                    // wait 0,5 sec. between each measuring cycle
}
```

7.6 Potentiometer PWM

```
//*****************************************************************************
//  VMA209 - Pot meter + PWM example
//  The VMA209 contains a blue potmeter (trimmer) which is connected to Analog 0
//  In this example we show You how to measure the voltage between 0 and 5V,how to visualise it by the serial monitor, and how to adjust the intensity of 2 LED's by using PWM
//  Programmed :Arduino IDE
//  Board  : Arduino Leonardo, UNO,
//*****************************************************************************/

int potpin=A0; //Initialize integer potpin as Analog 0
int ledpin1=11;//Define digital interface 11 (PWM output)
int ledpin2=10;//Define digital interface 10 (PWM output)



int val=0;// initialize val as a integer with value 0
void setup()
{
Serial.begin(9600);//Set the communications baudrate to 9600 Baud

}
void loop()
{
val=analogRead(potpin);// Read the sensor's analog value and assign it to val
Serial.println(val);// Print this value to the serial monitor
analogWrite(ledpin1,val/4);// write this value to the LED and set its brightness by PWM (value between 0 and 255, that's why devide by 4)
analogWrite(ledpin2,val/4);// write this value to the LED and set its brightness by PWM (value between 0 and 255, that's why devide by 4)



delay(100);//wait for 0,1 second for the next measurement
}
```

7.7 Buzzer Test

```
//*****************************************************************************
// VMA209 Buzzer example
// The buzzer on the VMA209 is connected to digital pin 3
// with this small example, we show You how to create a siren
// compatible with  : Arduino Leonardo or Arduino UNO R3
//*****************************************************************************/

int buzzer=3;     //Set variable buzzer as integer and assign the value 3
void setup()
{
pinMode(buzzer,OUTPUT);// Initialize Pin3 (buzzer) as output
```

```
}
void loop()
{
unsigned char i,j;// Define variables
while(1)
{
for(i=0;i<80;i++)
{
digitalWrite(buzzer,HIGH);//Sound On
delay(1);// Wait 1ms
digitalWrite(buzzer,LOW);//Sound Off
delay(1);//Wait 1ms
}
for(i=0;i<100;i++)//second sound
{
digitalWrite(buzzer,HIGH);//Sound On
delay(2);//Wait 2ms
digitalWrite(buzzer,LOW);//Sound Off
delay(2);//Wait 2ms
}
}
}
```

7.8 Up-Down Counter

```
//*****************************************************************************
//----------------VMA209 UP-DOWN counter example-----------
//*****************************************************************************/
int latchPin = 4;
int clockPin =7;
int dataPin = 8; //Define latch, clock and data pins for the display
int KEY_ADD =A1;  //Define Switch 1 as count UP
int KEY_DEC=A2;  //Define Switch 2 as count DOWN
unsigned char Dis_table[] = {0xC0,0xF9,0xA4,0xB0,0x99,0x92,0x82,0xF8,0X80,0X90};        //LED status
display variables
unsigned char Dis_buf[]   = {0xF1,0xF2,0xF4,0xF8};
unsigned char disbuff[]  =  {0, 0, 0, 0};
int SUM = 0;
int Flag_up = 1;
int Flag_up1 = 1;

void setup ()
{
  pinMode(latchPin,OUTPUT);
  pinMode(clockPin,OUTPUT);
  pinMode(dataPin,OUTPUT); //define pins 4,7,8 as OUTPUT
}

void display()
{
```

```
  for(char i=0; i<=3; i++)
  {
    digitalWrite(latchPin,LOW);
    shiftOut(dataPin,clockPin,MSBFIRST,Dis_table[disbuff[i]]); //Send value to the 4 displays
    shiftOut(dataPin,clockPin,MSBFIRST,Dis_buf[i] );
    digitalWrite(latchPin,HIGH);
    delay(2);                                  //wait 2ms before accesing the next display, please try another value
(for example 100 ) to see how the multiplexing works
  }

}
unsigned char ScanKey()                        //Scan push button 1 (S1)
{
  if(Flag_up  && digitalRead(KEY_ADD) == LOW)
  {
    Flag_up = 0;
    display();
    display();
    if(digitalRead(KEY_ADD) == LOW)
    {
      return 1;
    }
  }
  if(digitalRead(KEY_ADD) == HIGH)
  {
    Flag_up = 1;
  }
  return 0;

}

unsigned char ScanKey1()                       //Scan push button 2 (S2)
{
  if(Flag_up1  && digitalRead(KEY_DEC) == LOW)
  {
    Flag_up1 = 0;
    display();
    display();
    if(digitalRead(KEY_DEC) == LOW)
    {
      return 1;
    }
  }
  if(digitalRead(KEY_DEC) == HIGH)
  {
    Flag_up1 = 1;
  }
  return 0;

}

void loop()
{
```

```
  display();
  if( ScanKey() == 1)                    // if a push button is pressed
 {
   SUM++;                                                    //Add 1
   if(SUM>9999)                                  //Maximum counter value is 9999 (try another value
!! )
   {
    SUM = 9999;  // remains at 9999
   }
   disbuff[0]=SUM/1000;              //Fill the 4 display buffers with the new value
   disbuff[1]=SUM%1000/100;
   disbuff[2]=SUM%100/10;
   disbuff[3]=SUM%10;
  }

  if( ScanKey1() == 1)                   //Button 2 pushed ?
 {
   SUM--;                                                    //Count down one value
   if(SUM<=0)                            //Value is zero ? than remain at 0
   {
    SUM = 0;
   }
   disbuff[0]=SUM/1000;              //Fill the 4 display buffers with the new value
   disbuff[1]=SUM%1000/100;
   disbuff[2]=SUM%100/10;
   disbuff[3]=SUM%10;
  }

 }
```

7.9 Temperature Test

```
//*****************************************************************************/
//---------------- VMA209 measuring temperature by using input A4 ---------------------------------------/
//--- ATTENTION -- The symbol on the PCB is for sensor 18B20 !! If using LM35 it has to be upside down !! --
--/
//---------------- CHECK THE POLARITY FIRST !!! ---------------------------------------------------------/
//*****************************************************************************/
#define LM35 A4

int val = 0;                           // initialising variable val with value 0
float temp = 0;                  // initialising variable temp as floating point

void setup()
{
 Serial.begin(9600);    // set the baudrate to 9600
}

void loop()
{
```

```
  val = analogRead(LM35);                       // read the value of A4
  temp = val * 0.48876;                                 // eventual correction factor
  Serial.print("LM35 = ");
  Serial.println(temp);                         // print the value to the serial  monitor
  delay(1000);   // wait one second for the next measurement
}
```

7.10 Voltmeter

```
//*****************************************************************************************
*********
//-- VMA209 Voltmeter example
//-- This example reads the value of the blue potmeter on the VMA209 and shows it on the display
//*****************************************************************************************
*********/

int  potpin=A0;//declare variable potpin as connected to analog input A0
int latchPin = 4;
int clockPin =7;
int dataPin = 8; //declare latchPin, clockpin and datapin for the display (data pins 4,7 and 8)

unsigned char Dis_table[] = {0xC0,0xF9,0xA4,0xB0,0x99,0x92,0x82,0xF8,0X80,0X90};         // This table
defines the 7 segments of the display , 0x is not used here.00 = all segments ON, FF = all segments OFF,
0x7F is used for the decimal point
unsigned char Dis_buf[]   = {0xF1,0xF2,0xF4,0xF8}; // this table sets a "selector" for what digit is selected
unsigned char disbuff[]  =  {0, 0, 0, 0}; // set display buffer to 0
int SUM = 0;



void setup ()
{
  pinMode(latchPin,OUTPUT);
  pinMode(clockPin,OUTPUT);
  pinMode(dataPin,OUTPUT); //set the 3 data pins as output
}

void display()
{
  for(char i=0; i<=3; i++)// This routine will write information to the 4 display digits, variable i will count
from 0 to 3
  {
    digitalWrite(latchPin,LOW); //Activate the Latch Pin , the Latch Pin allows data to be written to the shift
registers of the VMA209
    shiftOut(dataPin,clockPin,MSBFIRST,Dis_table[disbuff[i]]); //output Dis_table depending on the value of i ,
    shiftOut(dataPin,clockPin,MSBFIRST,Dis_buf[i] );  //output Dis_buf depending on the value of i
    digitalWrite(latchPin,HIGH); //De-activate the latch pin, the information for digit(i) has been written
    delay(2);                                   // take a break for 2mS
  }

}
```

8. More Information

Please refer to the VMA209 product page on www.velleman.eu for more information.

Use this device with original accessories only. Velleman nv cannot be held responsible in the event of damage or injury resulting from (incorrect) use of this device. For more info concerning this product and the latest version of this manual, please visit our website www.velleman.eu. The information in this manual is subject to change without prior notice.

Velleman® Service and Quality Warranty

Since its foundation in 1972, Velleman® acquired extensive experience in the electronics world and currently distributes its products in over 85 countries.

All our products fulfil strict quality requirements and legal stipulations in the EU. In order to ensure the quality, our products regularly go through an extra quality check, both by an internal quality department and by specialized external organisations. If, all precautionary measures notwithstanding, problems should occur, please make appeal to our warranty (see guarantee conditions).

General Warranty Conditions Concerning Consumer Products (for EU):

• All consumer products are subject to a 24-month warranty on production flaws and defective material as from the original date of purchase.

• Velleman® can decide to replace an article with an equivalent article, or to refund the retail value totally or partially when the complaint is valid and a free repair or replacement of the article is impossible, or if the expenses are out of proportion.

You will be delivered a replacing article or a refund at the value of 100% of the purchase price in case of a flaw occurred in the first year after the date of purchase and delivery, or a replacing article at 50% of the purchase price or a refund at the value of 50% of the retail value in case of a flaw occurred in the second year after the date of purchase and delivery.

• Not covered by warranty:

- all direct or indirect damage caused after delivery to the article (e.g. by oxidation, shocks, falls, dust, dirt, humidity...), and by the article, as well as its contents (e.g. data loss), compensation for loss of profits;

- consumable goods, parts or accessories that are subject to an aging process during normal use, such as batteries (rechargeable, non-rechargeable, built-in or replaceable), lamps, rubber parts, drive belts... (unlimited list);

- flaws resulting from fire, water damage, lightning, accident, natural disaster, etc.…;

- flaws caused deliberately, negligently or resulting from improper handling, negligent maintenance, abusive use or use contrary to the manufacturer's instructions;

- damage caused by a commercial, professional or collective use of the article (the warranty validity will be reduced to six (6) months when the article is used professionally);

- damage resulting from an inappropriate packing and shipping of the article;

- all damage caused by modification, repair or alteration performed by a third party without written permission by Velleman®.

• Articles to be repaired must be delivered to your Velleman® dealer, solidly packed (preferably in the original packaging), and be completed with the original receipt of purchase and a clear flaw description.

• Hint: In order to save on cost and time, please reread the manual and check if the flaw is caused by obvious causes prior to presenting the article for repair. Note that returning a non-defective article can also involve handling costs.

• Repairs occurring after warranty expiration are subject to shipping costs.

• The above conditions are without prejudice to all commercial warranties.

The above enumeration is subject to modification according to the article (see article's manual).

Made in PRC
Imported by Velleman nv
Legen Heirweg 33, 9890 Gavere, Belgium
www.velleman.eu