

MCP4725 Breakout Board

CHIP: MCP4725A0T-E

SIZE: 15.5mm x 13mm

DESCRIPTION:

The MCP4725 is a 12 bit digital to analog converter (DAC) with integrated EEPROM. This board breaks out GND, VDD, SDA, SCL & VOUT (output). The operating voltage of the DAC is 2.7-5.5v and typically has a 6 micro second settling time. This module features 1uF & 100nF capacitors on VDD, and an LED indicating good power. The A0 address pin is tied to GND by default, giving it a 0x60 I2C address.

This board takes out the tedious, unpredictable troubles that come from DAC's, and will let you streamline and debug your inventions & innovations

Typical applications include precise voltage generation, low frequency wave form generation, prototyping of analog circuits and of course: education!

Happy building!

BOARD PINOUT:

PIN	NAME	DOES THIS
1	GND	Ground
2	SCL	Serial Clock
3	SDA	Serial Data
4	VOUT	Output Voltage
5	VDD	+3.3v-5v typically

FORMULAS:

MODE	FORMULA	NOTES
Output Voltage Formula	$VOUT = (VDD * D) / 4096$	D is the input digital code from 0-4095 (12 bit resolution) VDD is supply voltage (2.7-5v5) VOUT is resulting voltage

APPLICATIONS:

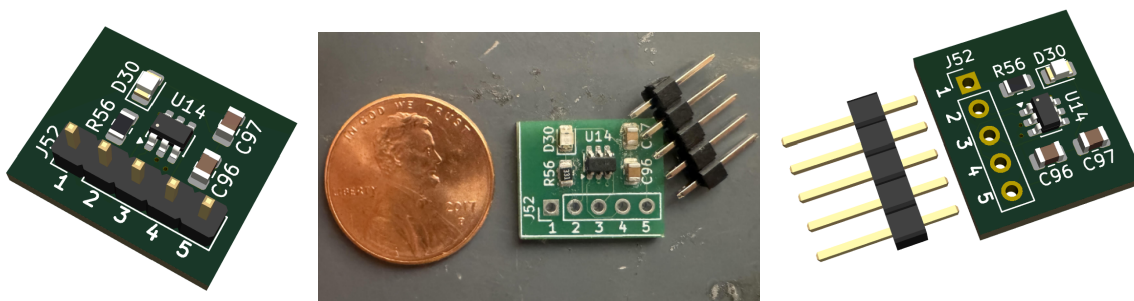
- Generating precise tuning, pitch, or modulation voltages for analog music gear and modular synthesizers
- Setting adjustable, software-controlled DC bias voltages or thresholds for analog comparators and measurement circuits

- Driving analog dimming inputs on LED drivers to provide flicker-free brightness control without PWM ripple
- Outputting steady, custom analog voltages to test, calibrate, or emulate analog sensor signals into microcontroller ADCs

CHIPS ON BOARD:

COMPONENT	PART #	JOB	DATASHEET
DAC	MCP4725A0T	Digital to Analog conversion	[Manufacturer Datasheet]

PHOTOS:



WIRING:

CONNECTION	DESTINATION
VDD	Arduino 5V
GND	Arduino GND
SDA	Arduino SDA (Pull up with external 4.7-10k resistor)
SCL	Arduino SCL (Pull up with external 4.7-10k resistor)

EXAMPLE CODE:

```

/*
  Meridian Electronic
  MCP4725A0 12-bit I2C DAC + Arduino Uno
*/

#include <Wire.h>

const uint8_t DAC_ADDR = 0x60;    // A0 = GND. Use 0x61 if A0 = VDD.
const float   VREF      = 5.0;    // VDD in volts (measure yours for accuracy)

```

```

// Fast Mode write (2 bytes): updates the DAC register only, not the EEPROM.
// Byte 1: 0 0 PD1 PD0 D11 D10 D9 D8   (PD = 00 -> normal operation)
// Byte 2: D7 ... D0
bool dacWrite(uint16_t value) {
    if (value > 4095) value = 4095;
    Wire.beginTransaction(DAC_ADDR);
    Wire.write((value >> 8) & 0x0F); // C2 C1 = 00, PD = 00, top 4 data bits
    Wire.write(value & 0xFF);       // low 8 data bits
    return Wire.endTransmission() == 0;
}

// Convenience: set an output voltage (clamped to 0..VREF).
bool dacWriteVolts(float volts) {
    if (volts < 0) volts = 0;
    if (volts > VREF) volts = VREF;
    return dacWrite((uint16_t)(volts / VREF * 4095.0 + 0.5));
}

void setup() {
    Serial.begin(115200);
    Wire.begin();
    Wire.setClock(400000); // chip supports 100k / 400k / 3.4M

    // Check the DAC acknowledges its address.
    Wire.beginTransaction(DAC_ADDR);
    if (Wire.endTransmission() != 0) {
        Serial.println(F("MCP4725 not found. Check wiring, A0 level and address."));
        while (true) {}
    }
    Serial.println(F("MCP4725 found."));
}

void loop() {
    // 1) Step through a few fixed voltages.
    const float steps[] = {0.0, 1.25, 2.5, 3.75, 5.0};
    for (uint8_t i = 0; i < sizeof(steps) / sizeof(steps[0]); i++) {
        dacWriteVolts(steps[i]);
        Serial.print(F("Output set to "));
        Serial.print(steps[i], 2);
        Serial.println(F(" V"));
        delay(1000);
    }

    // 2) Sawtooth ramp 0 -> 4095.
    for (uint16_t v = 0; v < 4096; v += 16) {
        dacWrite(v);
    }

    // 3) A couple of sine cycles, centered at mid-scale.
    for (uint8_t cycle = 0; cycle < 2; cycle++) {
        for (uint16_t deg = 0; deg < 360; deg += 4) {
            float s = sin(deg * PI / 180.0);
            dacWrite((uint16_t)(2047.5 + 2047.5 * s));
        }
    }
}

```

NOTES & DEBUGGING:

- This boards I2C Address is 0x60 (A0 is tied low)
- Don't forget 4.7-10k ohm pull up resistors on SDA and SCL, you can also opt to use Arduinos internal pull ups, but external are more standardized
- The resolution is 12 bits, giving 4096 codes
- Typical 6us settling time

- Be mindful of supply VDD and I2C logic levels (2.7 to 5.5v), VDD and reference are tied, not individually selectable
- Load limits: 5 k Ω minimum, 1000 pF maximum
- I2C speeds: 100 kHz, 400 kHz, 3.4 MHz
- LSB size at 5v: 1.22mV
- LSB at 3.3v: 0.81mV

Feel free to send any questions to sales@meridianelectronic.com

Check MeridianElectronic.com for datasheets, 3D printing, solder stencils & more!