

- ✓ Full resource of UNO R3
- ✓ Onboard Virtual USB HID device
 - The USB HID can be programming the board as a keyboard / mice / etc... with HID class device
- ✓ Ground test terminal
- ✓ TQFP-48 MD-3248P MCU
- ✓ Analog power and digital power and ground are separated
- ✓ Onboard 4.096V/2.048V voltage reference source
 - 4.096V for PWM-DAC
 - AREF for MCU ADC selectable between 4.096V and 2.048V
- ✓ 4 channel onboard PWM-DAC
 - CH0/1 with 16bit resolution
 - CH2/3 with 12bit resolution
- ✓ 5 channel High Voltage Analog input
 - Range 32V with 4.096V AREF
 - Range 16V with 2.048V AREF
- ✓ Onboard 3 tack switch buttons and 3 LEDs for fast programming

Features & Parameters



- Massduino MD-3248P with 32KB Flash , 2KB SRAM
- Fully Arduino IDE development
- Full arduino UNO resource support
- Rich additional resource
 - 3 user programmable LEDs
 - 3 tack switch buttons
 - 4 CH pwm DAC (2 ch fast mode and 2 ch high resolution mode)
 - 5 CH HV ADC (Max input 32V)
 - Onboard 4.096V and 2.048V voltage reference source
- Virtual USB , make the board as a HID device (Keyboard / Mice etc.)
- **DC input voltage** : DC 9 to 24V
- **System voltage** : 3.3V or 5V
- **PWM DAC Range** : 0 ~ 4.096V
- **PWM Resolution (CH 1/2)**: 16bit
- **PWM Resolution (CH 3/4)**: 10bit
- **HV ADC input range**: 0~32V
- **HV ADC resolution**: 1mV w/s 15bit ADC (`analogRead_15bits()`)

Install Masduino support package

Inorder to use Masduino UNO Pro , you need to download masduino support package from below link:

http://www.inhaos.com/downcount.php?download_id=218

After you download and unzip the file , you will got two folder:

 For Arduino-1.6.2~1.6.5	2018/5/19 12:06
 For Arduino-1.6.6-Later	2018/5/19 11:48

Find the directory that suits your Arduino IDE version , copy to target directory . Here is the default sketchbook directory for most popluar system:

- > Windows: C:\Users\\Documents\Arduino
- > Mac OSX: /Users/user/Documents/Arduino
- > LINUX: /home/<Userame>/sketchbook

And then restart the Arduino IDE , you will see the device appear in the menu :

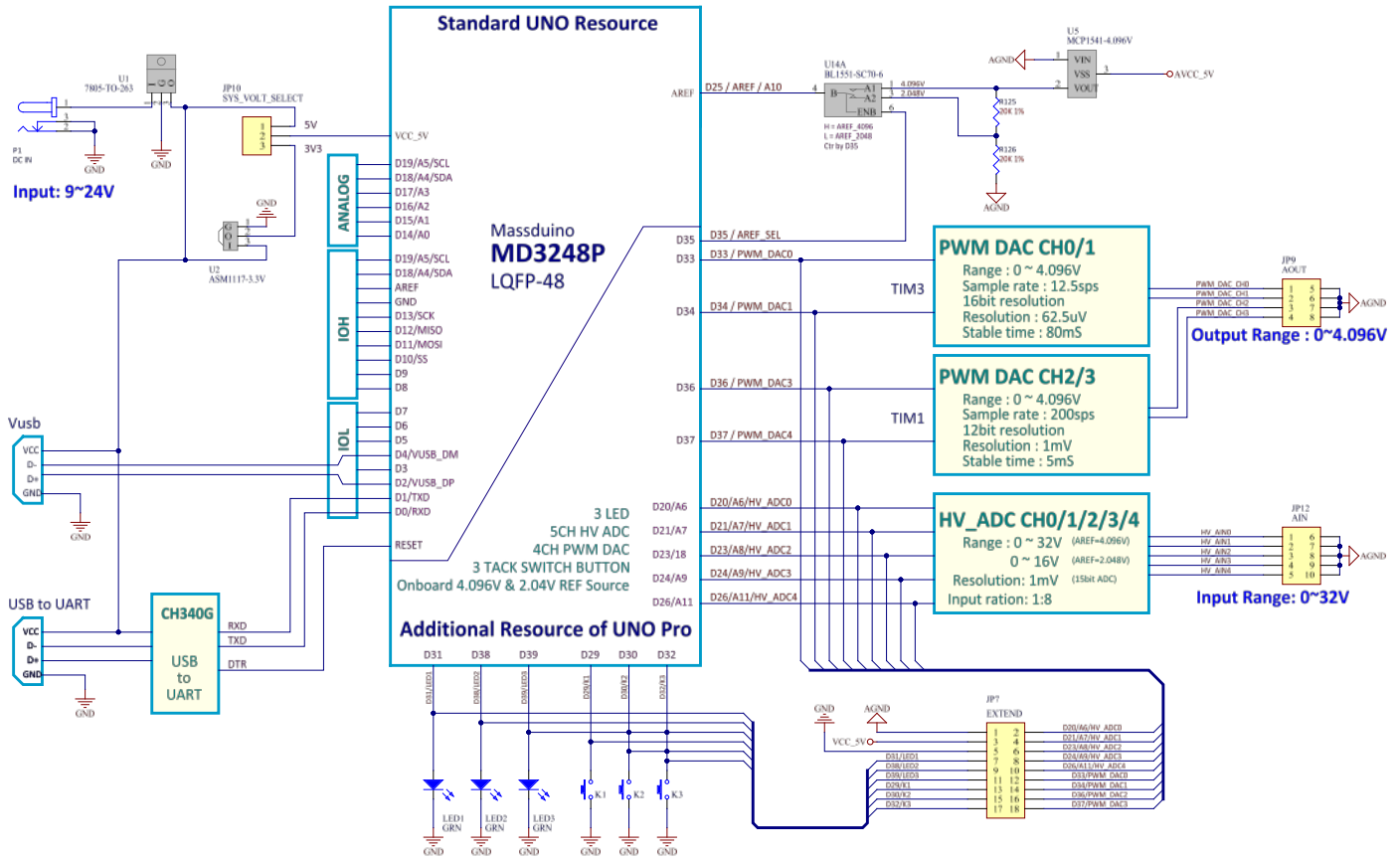
tools - > board - > Masduino Ev board -> MD-3248P-LQFT48

Then , you can start to play with Uno Pro.

Masduino UNO Pro

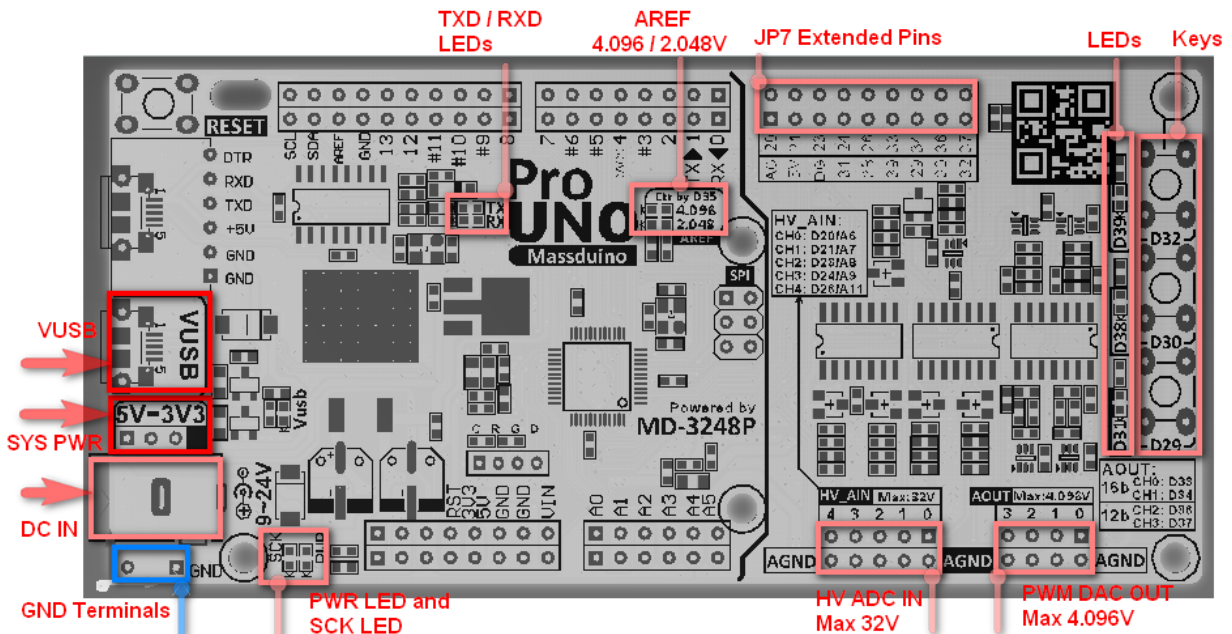
High performance enhanced development board compatible with Arduino UNO

Block diagram



Here is the block diagram of UNO Pro , as you can see , UNO Pro support full resource of standard UNO , and additional below resource :

- 3 user programmable LEDs
- 3 tack switch buttons
- 4 CH pwm DAC (2 ch fast mode and 2 ch high resolution mode)
- 5 CH HV ADC (Max input 32V)
- Onboard 4.096V and 2.04V voltage reference source



PWM DAC

UNO Pro have 4ch onboard PWM DAC , they are connection to D33/D34/D36/D37 , when you config those pin for PWM function, the onboard LPF will convert the PWM signal to a analog voltage , the analog level is depended on the duty cycle of the PWM signal.

The reference source of the DAC is connect to onboard 4.096V reference, which means the max output of DAC is 4.096V.

The D33/D34 are using TIM3 and D36/D37 using TIM1 , so the PWM DAC0 (D33) and PWM DAC1 (D34) must have same resolution and PWM2 and PWM3 are too.

The DAC resolution is depended on the PWM resolution , if the PWM resolution set to 12bit , the DAC resolution is 12bit too , the max level is $2^{12} = 4096$ levels.

For PWM signal, the resolution is inversely proportional with the frequency , which means if you using higher resolution , you must lower the frequency. So if you set PWM resolution to 16bit , the frequency will be 16times lower than 12bit.

The LPF (Low Pass Filter) will lead the analog signal delay, the 16bit PWM signal will lead 16 times delay time than 12bit. In UNO Pro , the 12bit PWM delay time is 5mS and 16bit PWM delay is 80mS.

So that's why we setup two type of DACs , CH0/1 is high resolution but with lower sample rate , CH2/3 is higher sample rate but resolution is lower than DAC0/1.

Below is recommended setting for DACs:

No.	DAC Chn.	Pin number	Timer	Resolution	Resolution mV	Delay Time	Sample Rate	Remark
1	DAC 0	D33	TIM3	16	62.5uV	80mS	12.5sps	
2	DAC 1	D34						
3	DAC 2	D36	TIM1	12	1mV	5mS	200sps	
4	DAC 3	D37						

The function `"pwmResolution(PWM_PIN, 16);"` is used to setup the resolution of the PWM signal , for D33/D34 , the resolution have to setup to same , and D36/D37 too , if you using `"pwmResolution(36, 12);"` to setup D36 to 12bit , the D37 will be setup to 12bit resolution too automatic.

As resolution suggestion above , the DAC0/1 can be setup to any resolution of 1 to 16bit and DAC2/3 can be setup to any resolution of 1 to 12bit , if the resolution is over the suggestion level , the output signal will be have big ripple.

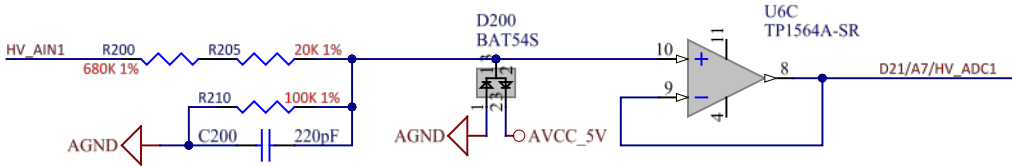
When you setup PWM = 0 , the PWM value sill still keep to 1 , so if you want get real 0V of output , you can disable

PWM function and write the IO to LOW, reference code below :

```
pwmTurnOff ( PWM_PIN ); // trun off the PWM function
digitalWrite ( PWM_PIN , LOW ) ;
delay(5);
```

HV ADC (High Voltage ADC)

The input circuit as below , the input ration is 1:8.



If you are using 4.096V as the AREF , the max input range is $4.096V * 8 = 32.768V$, since the full scale is non linear , so we only use 0~32V range. In this case, we suggestion you use “analogRead_15bits()” function , the ADC resolution is 1mV.

If you are using 3.3V system voltage , the AREF must lower than 3.3V , in this case , you can use 2.048V as the AREF , (write D36 to LOW) , the max input is 16V. we suggestion you use “analogRead_14bits()” function , the ADC resolution is 1mV too.

Since the input divider / OP amp / ADC , every stage have it’s own fixed error , so if you want to get more precise measurement result , you need calibration algorithm , the simple calibration algorithm below:

- Step1:** Select a calibration point , this point better is close to full scale .
- Step2:** Connected the voltage source to one of input channel.
- Step3:** use high precision voltmeter to measurement the input voltage, record it to Vol_A.
- Step4:** read form ADC , record it to VolB
- Step5:** then the calibration factor is $VolA/VolB$.
- Step6:** All input readings, multiplied by the calibration factor, are the calibrated data.

After the simple calibration, the input channel will easy to reach 0.5% accuracy.

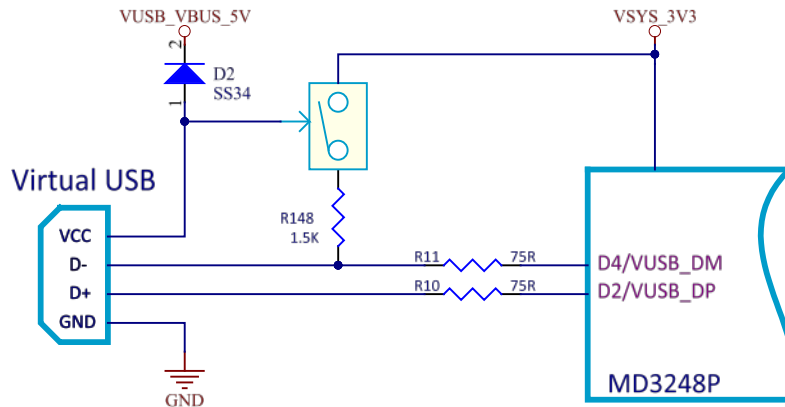
We strong recommend use external reference source to replace default reference.

```
analogReference (EXTERNAL) ;
```

Virtual USB

UNO Pro have one virtual USB connector, it can be working as an USB HID device , such as keyboard / mice .

For virtual USB , the D2 will working as USB D+ and D4 will working as USB D- , during USB low speed device specification , after power up , the D- will be pull up by a 1.5K resistor.

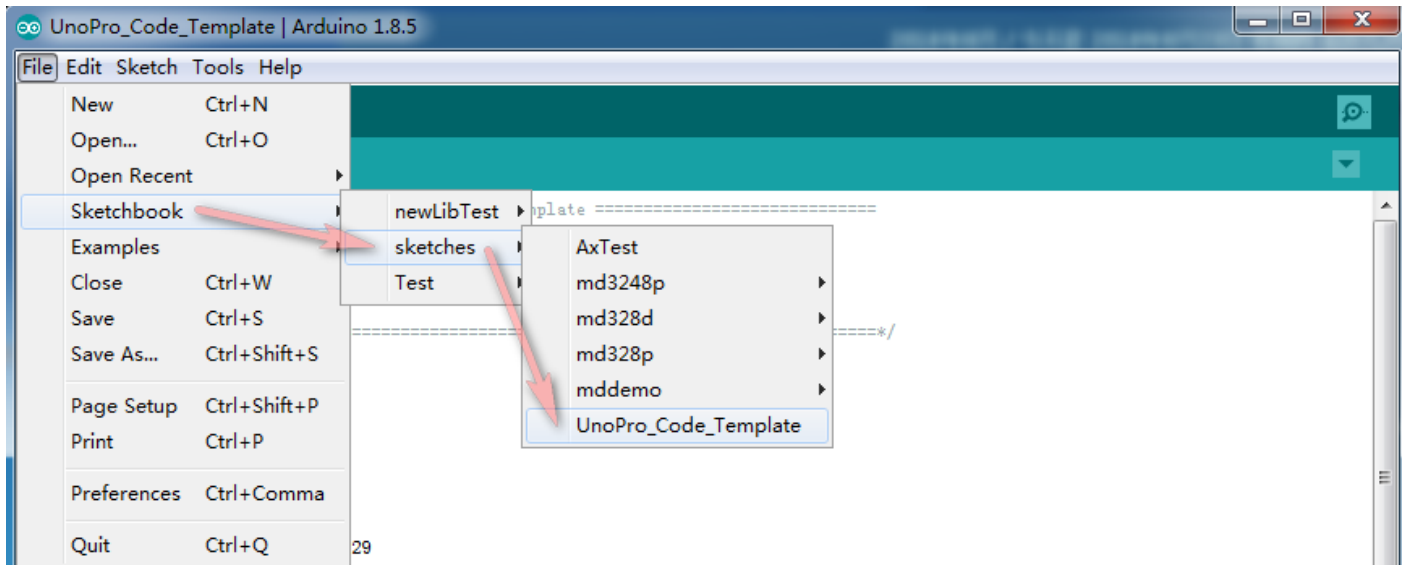


When virtual USB attached to the host , the pull up resistor “R148 1.5K” will auto apply to D4/VUSB_DM pin . if you do not use virtual USB function , the R148 will not connected to 3.3V , D2 and D4 can be used as the original function.

In order to use virtual USB function , the SYS_PWR must switch to 3V3 , since USB data level is 3.3V.

Code Template

Since the UNO Pro have a lot of additional resource than the standard UNO , so this template will help user to fast start coding.



```
/* =====MassduinoUnoPro_Code_Template=====
Version: 1.0
Release: 22 May.2018
www.inhaos.com
===== */
#include <WDT.h>

#define pin_SCK_LED      13

#define pin_BTN1        29
#define pin_BTN2        30
#define pin_BTN3        32
#define pin_LED1        31
#define pin_LED2        38
#define pin_LED3        39

#define pin_PWM_DAC_CH0  33
#define pin_PWM_DAC_CH1  34
#define pin_PWM_DAC_CH2  36
#define pin_PWM_DAC_CH3  37

#define pin_HV_AIN_CH0   A6
#define pin_HV_AIN_CH1   A7
#define pin_HV_AIN_CH2   A8
#define pin_HV_AIN_CH3   A9
#define pin_HV_AIN_CH4   A11
```



```
#define pin_AREF_SOURCE      35
#define fun_SetARF_4096     { digitalWrite ( 35 , HIGH ) ; }
#define fun_SetARF_2048     { digitalWrite ( 35 , LOW ) ; }

void setup() {
  sysClock ( EXT_OSC );           // switch system clock source to 16MHz ext osc
  pinMode ( pin_SCK_LED, OUTPUT );
  pinMode ( pin_BTN1 , INPUT_PULLUP );
  pinMode ( pin_BTN2 , INPUT_PULLUP );
  pinMode ( pin_BTN3 , INPUT_PULLUP );
  pinMode ( pin_LED1 , OUTPUT );
  pinMode ( pin_LED2 , OUTPUT );
  pinMode ( pin_LED3 , OUTPUT );
  pinMode ( pin_AREF_SOURCE , OUTPUT );

  // initial analog refernce source
  analogReference ( EXTERNAL );   //switch analog reference source to AVREF
  fun_SetARF_4096 ;

  // enabel watchdog
  wdt_enable(WTO_256MS);         // enable the watchdog , period 256mS
}

void loop() {
  wdt_reset();                  // remember to reset the watchdog in main loop if you enabel the watchdog.
  // wdt_disable(); // disable the watchdog if you do not want it work anymore

  digitalToggle ( pin_SCK_LED );

  digitalWrite ( pin_LED1 , !digitalRead ( pin_BTN1 ) );
  digitalWrite ( pin_LED2 , !digitalRead ( pin_BTN2 ) );
  digitalWrite ( pin_LED3 , !digitalRead ( pin_BTN3 ) );
}
```

Massduino UNO Pro

High performance enhanced development board compatible with Arduino UNO



INHAOS China office: :

No.1 XinWei First Alley, GaoTianFang, DongCheng District,
DongGuan City, GuangDong 523112 China

E-mail: Support@inhaos.com