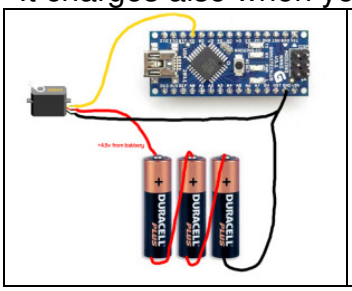
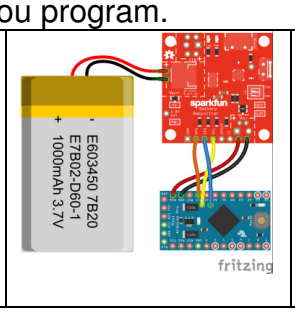
CH-1092 Belmont, Switzerland
Tel +41 21 728-6156, Fax 728-6157
Email info@didel.com, www.didel.com

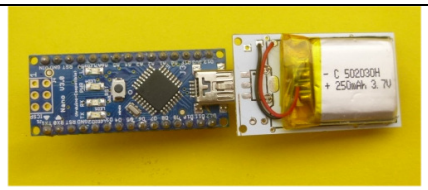
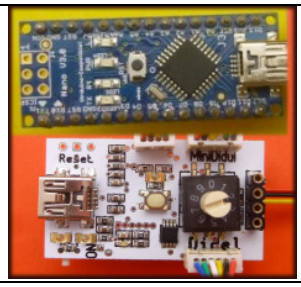
www.didel.com/Miniduipo.pdf

MiniDuiPo

Mini Diduino with embedded LiPo and charger

This new board is THE solution if you have to power some Arduino controlled wearables. It does include an Arduino compatible board, plus a Lipo with its USB charger. It is also a great replacement for the Nano and Micro-V3. Use a Nano to develop on your breadboard, cool. But.. do you need that forest of pins for your application?
The NanoLipo includes a 250mAh LiPo (easy to solder a smaller or larger one) and the charger. It charges also when you program.

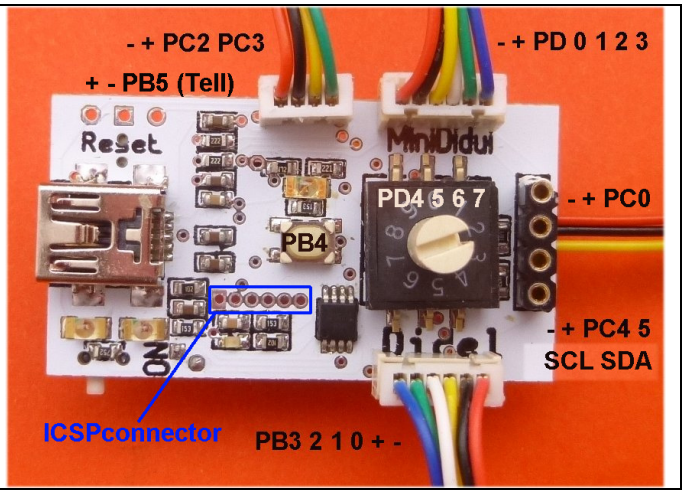



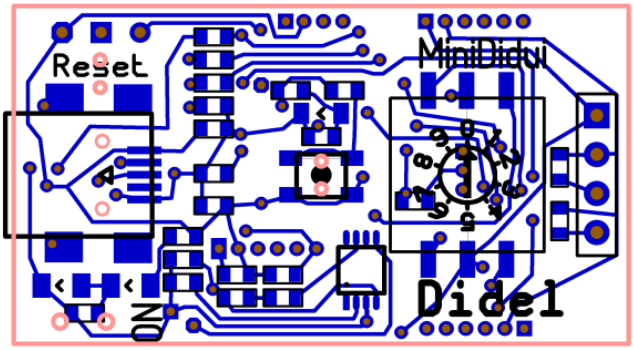
The NanoDui is compatible Arduino Duemillia and include that: 250mAh LiPo and charger.
We have assumed you will appreciate to connect a strip of WS2812B 0r WS2813 RGB leds, We assumed an USB connectore directly compatible with the cheap Oled SSD1306 will find its place on your application. We still provide all the ouputs of the AVR328, but the ports are available on three 1.27mm connectors. Easy connection with Molex connectors and cables. Small, compact, easy to build.

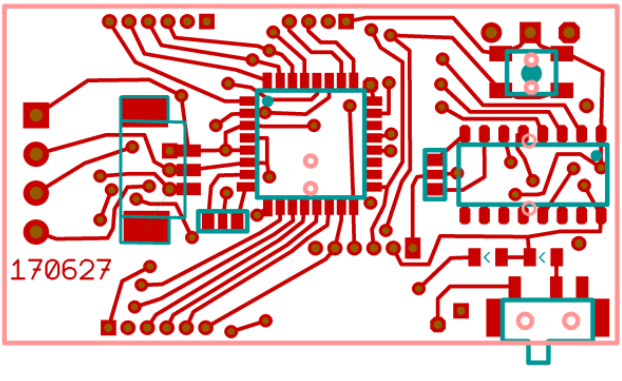
DS340G is the USB driver, easy to install.
Affectation of the AVR328 signals

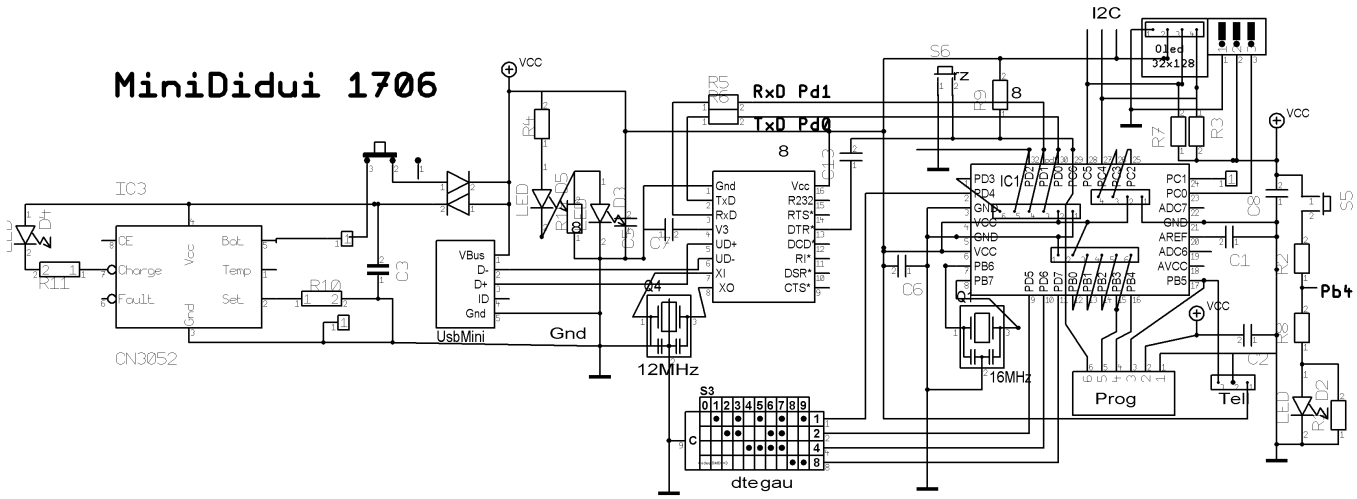
Pin	Port		Pin	Port	
0	PD0	Rx PdConn	10	PB2	PbConn
1	PD1	Tx id	11	PB3	id
2	PD2	Int0 id	12	PB4	Pous/Led
3	PD3	Int1 id	13	PB5	Tell
4	PD4	RotSw 1	14	PC0	2812b
5	PD5	RotSw 2	15	PC1	nc
6	PD6	RotSw 4	16	PC2	PcConn
7	PD7	RotSw 8	17	PC3	PcConn
8	PB0	PbConn	18	PC4	SCL
9	PB1	id	19	PC5	SDA



Software for the RGB strips, RGBCuve and Oled 128x64 and 128x32 are available from Didel, being documented now on Github.

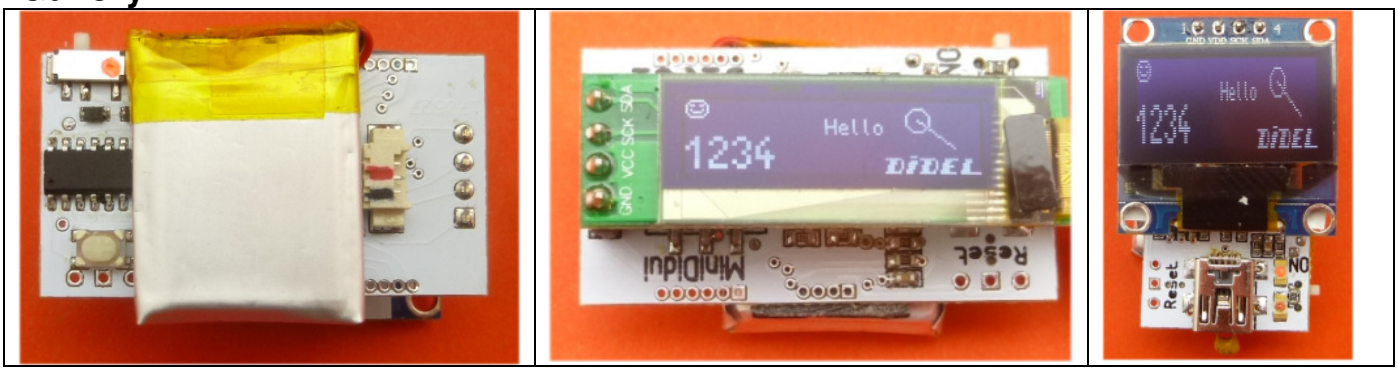






Reprogramming the AVR328 As usual - ICSP connector pin-out - + Ck Miso Mosi Rz

Gallery

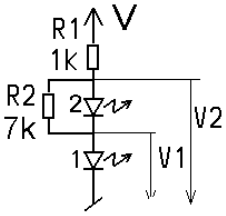


Note: The Oled32 software is compatible with the Oled64, as show. The OledPix64 soft gives of course the max resolution.

Interesting NanoLipo/DiduiPo features

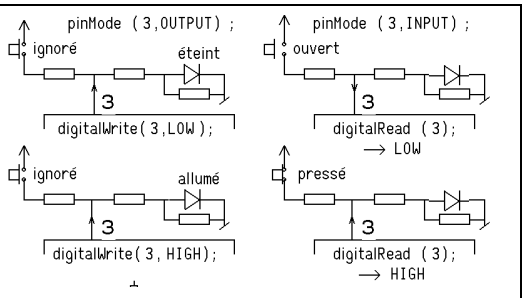
Two LEDs for voltage indication

One of the two LED **is Off** at 3V and **On** at 3.7V. It is not precise, but in a glance, you know if you need to recharge.



Push button/LED on pin 12 (PB4)

Pin 12 set as output control the LED, switch is ignored.
Pin 12 set as input allows to read the switch
Reading the push button can be done in different ways:
1) the program wait till the button is depressed, execute the action and continues when the button is released. This is a blocking action
2) the program check regularly if the key is depressed. If yes, the action is executed,
3) the interrupt mode is initialized; when the button is depressed, the action is executed. Usually, a flag is set the program will check, do the action and clear the flag.



Definition files

```
#define LP 12 // LED/Push pin
#define PushMode    pinMode (LP,INPUT)
#define LEDMode     pinMode (LP,OUTPUT)
#define  PushOn     (digitalRead(LP))
#define  LEDOn  digitalWrite (LP,1)
#define  LEDOff digitalWrite (LP,0)
```

```
#define bLP 4 // LED/Push bit on PORTB
#define PushMode  clearBit (DDRB,bLP)
#define LEDMode    setBit  (DDRB,bLP)
#define  PushOn    (PORTB & (1<<bLP))
#define  LEDOn  setBit  (PORTB,bLP)
#define  LEDOff clearBit(PORTB,bLP)
```

Library file `PushLED13.h` includes the definitions and useful functions.
For counting pushes see `TestCntPous.ino`

Rotary switch

The rotary switch is easy to read, pins are on bits PD4 5 6 7 in the order 1 2 4 8
Pull-ups are programmed on these pins

```
aig  = (~PIND>>4) & 0xF;
  switch (aig) {
    case 0:  //Id
     …
    break;
    etc
  }  // end switch
```

jdn 170711