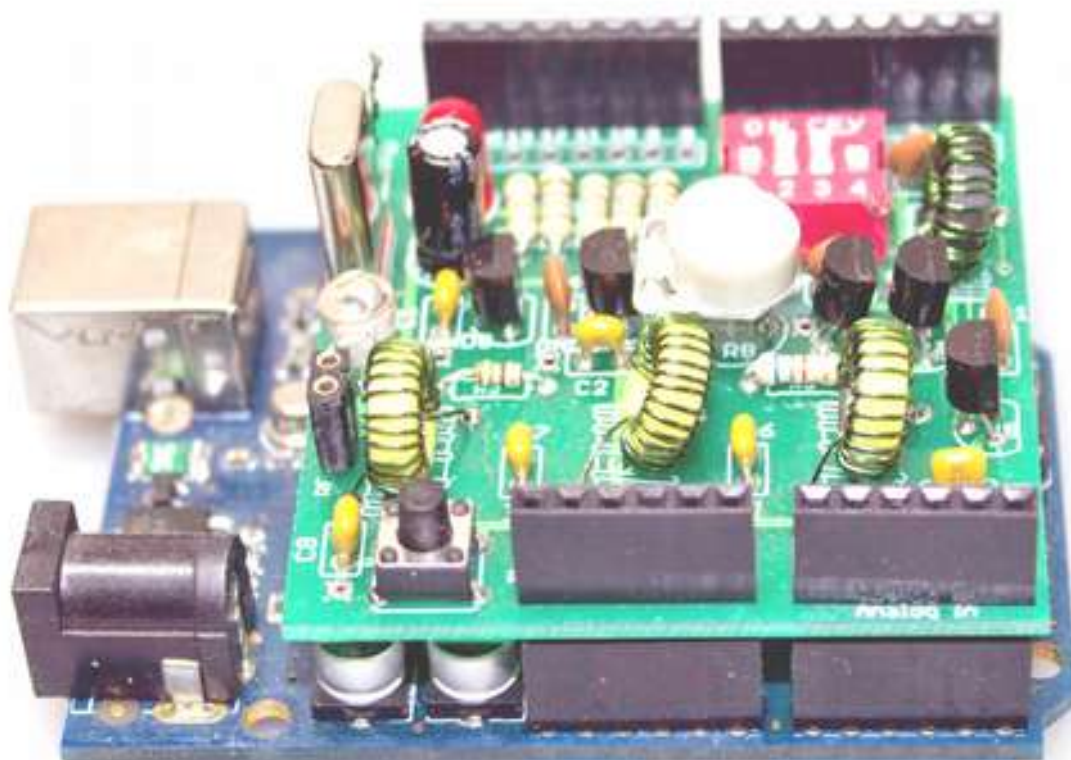




Introduzione

# Arduino per Radioamatori





# Open Electronics



*"Promoting the advancement  
of radio art and science"*



*"Sharing knowledge to  
the advancement  
of software/hardware technology"*

Fondata nel 1998 da Bruce Perens, **K6BP**

*"The strategic marketing paradigm of Open Source  
is a massively-parallel drunkard's walk  
filtered by a Darwinistic process."*

*Bruce Perens, K6BP*





# Arduino nel mondo hamradio?



## DIY

"If You Can't Open It,  
You Don't Own It"  
Maker's Motto



## Commercial

- C'è posto per entrambi i mondi
- Un ritorno ad un radiantismo non solo operativo
- Verso prodotti commerciali anche Open Electronics
- Ibridazione, customizzazione ed estensione delle caratteristiche





# Arduino: cos'è

- Una scheda a microcontroller
- Un insieme di tool
- Una community
- Un modello di marketing



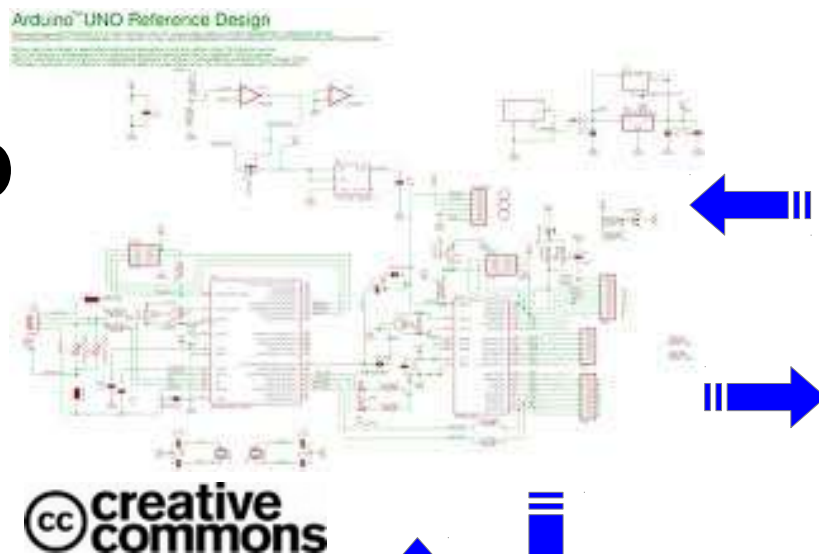




# Un Ciclo Virtuoso

**Makers**  
Community

**Arduino**  
LLC



Ladyada, Adafruit LLC

**Users**  
Community



**FLORENCE HAMFEST**

# Arduino: più in dettaglio



- L'ambiente Arduino è sia una **architettura open-electronics** che una struttura di **astrazione dall'hardware** (hardware abstraction layer)
- La programmazione è **C/C++ basato sul framework "Wiring"** integrato nell'ambiente di sviluppo Arduino
- Permette un alto grado di **astrazione dall'hardware** con una programmazione ad alto livello.
- Lo stesso codice dovrebbe supportare **diversi microcontrollori** e anche **diverse schede** basate sullo "standard" Arduino
- La programmazione ad alto livello rende meno efficiente la generazione del codice macchina.
- Gli strumenti di debugging ufficiali sono poco potenti e primitivi.
- E' comunque possibile (quando necessario) introdurre codice a più a basso livello (per es. accesso diretto ai registri) fino al codice assembly direttamente nello sketch
- E' possibile usare ambienti **di sviluppo integrati specifici** di terze parti o del costruttore del microcontrollore come Atmel Studio.
- Le versioni base di Arduino sono **costose** e **non sufficientemente** robuste per un impiego per prodotti finiti essendo destinate allo scopo di **prototipazione/sviluppo rapido**





# Arduino: la storia

*Il nome Arduino: da Re Arduino d'Ivrea, incoronato re d'Italia nell'anno 1002.*



**Ivrea**, Interaction Design Institute

Già 10 anni di vita!



**Massimo Banzi: How Arduino is open-sourcing imagination**

FILMED JUN 2012 • POSTED JUN 2012 • TEDGlobal 2012



Massimo Banzi, David Cuartielles,  
Tom Igoe, Gianluca Martino,  
David Mellis



**“Arduino: creare è un gioco da ragazzi”**

<http://www.wired.it>



# Da vari progetti Open Source: un ambiente interconnesso



- Processing (2001)
  - Ideato da MIT Media Lab
  - Linguaggio di programmazione per PC
  - Simile al C opera su piattaforma Java
  - Estremamente user-friendly, ricco di widget
  - Utile per l'interfacciamento con Arduino



- Wiring (2003)
  - Basato su Processing con target schede a MCU
  - "Sketching with hardware"



*Wiring Environment  
+ linguaggio*



*C/C++ e Wiring  
framework*



- Fritzing
  - EDA ideato da Postdam University (ancora versione beta)
  - Supporto alla progettazione dei collegamenti e del PCB
  - Estremamente user-friendly
  - Utile per documentazione e velocizzare l'apprendimento







# Programmazione lato embedded

<http://arduino.cc/>



- Arduino IDE

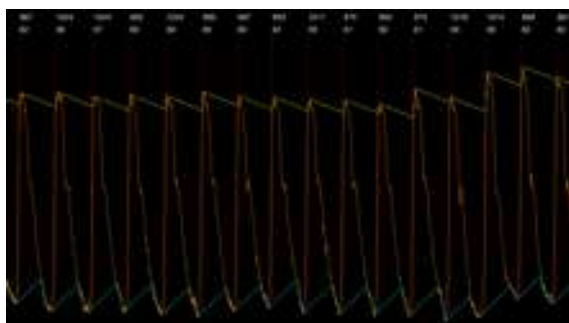


# Programmazione lato PC

<http://processing.org/>



- Processing



*"a tremendous platform for visualizing data"*

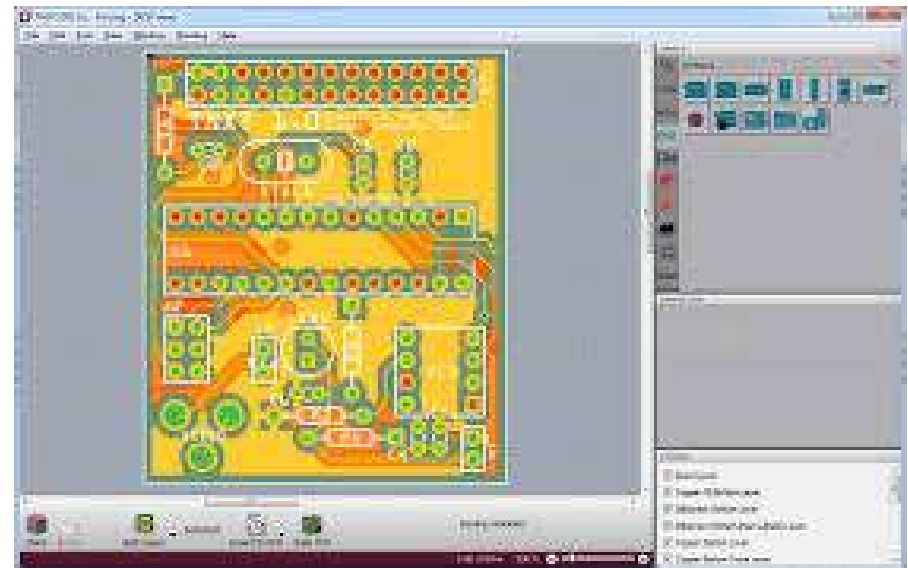
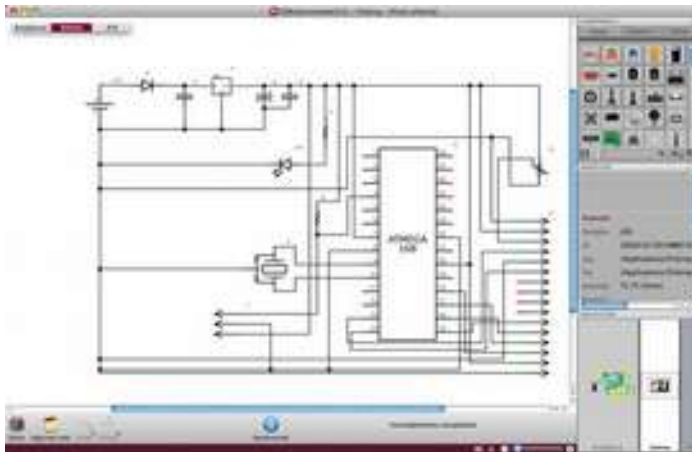


# Schemi e layout

<http://fritzing.org/>



- Breadboard View – uso didattico
- Schematic capture
- PCB design
- 80% documentazione di uso Arduino

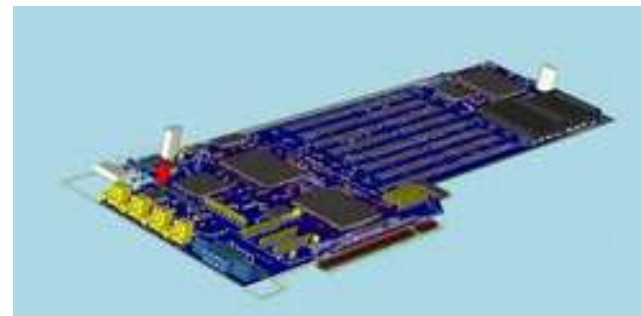


# KiCAD

<http://www.kicad.org>



- Uso anche professionale
- Nessuna limitazione
- GPL

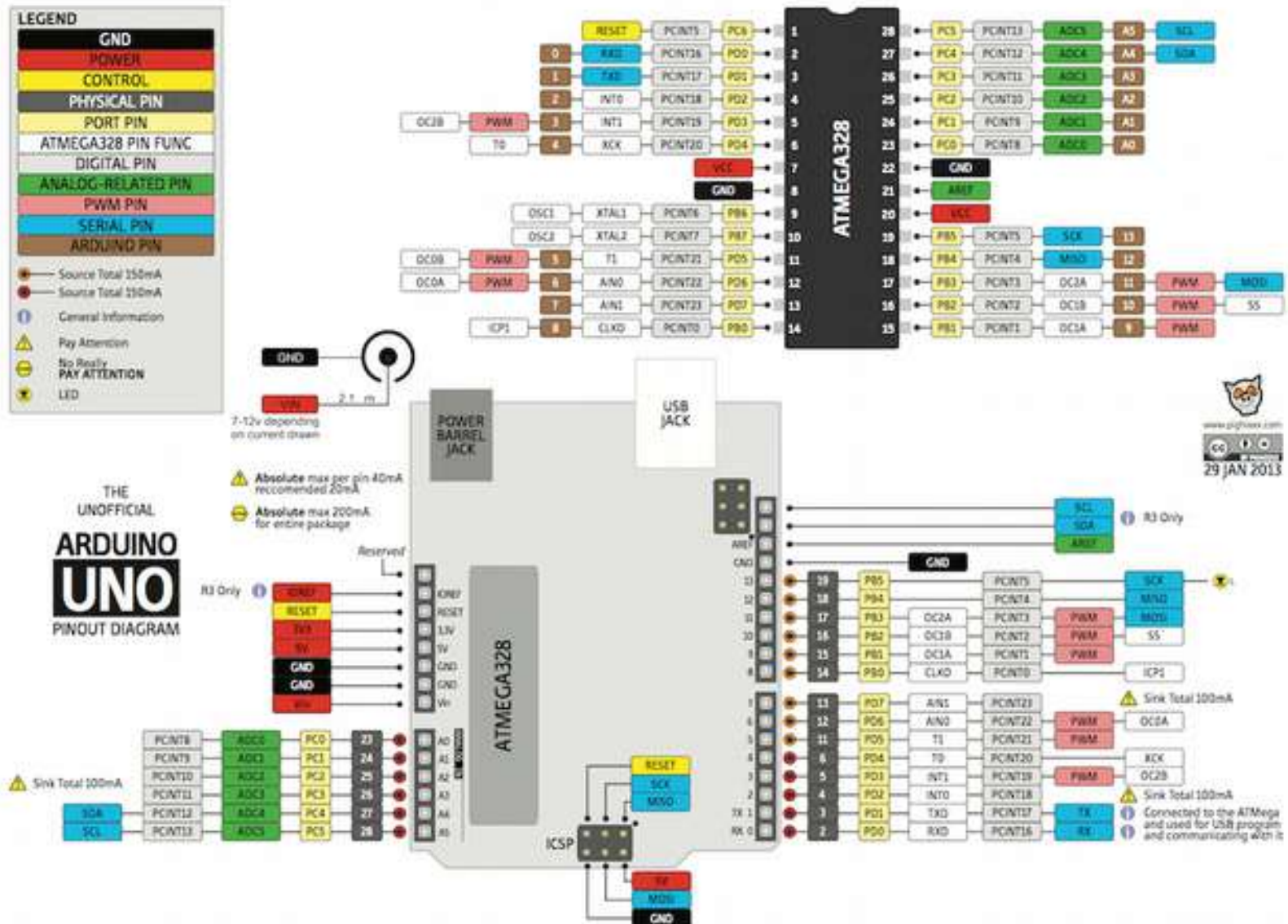




# Le schede: Ufficiali, approvate e DIY



# Arduino UNO







# Schede Arduino Standard





Item	uC	Input Voltage	System Voltage	Clock Speed	Digital I/O	Analog Inputs	PWM	UART	Flash Space	Bootloader	Programming Interface
 Arduino Due	AT91SAM3X8E	7-12V	3.3V	84MHz	54*	12	12	4	512Kb	Due	USB native
 Arduino Leonardo	ATmega32U4	7-12V	5V	16MHz	20*	12	7	1	32Kb	Leonardo	USB native
 Arduino Uno - R3	ATmega328	7-12V	5V	16MHz	14	6	6	1	32Kb	Optiboot	USB via ATmega16U2
 RedBoard	ATmega328	7-15V	5V	16MHz	14	6	6	1	32Kb	Optiboot	USB via FTDI








# Schede Arduino Standard

Item	uC	Input Voltage	System Voltage	Clock Speed	Digital I/O	Analog Inputs	PWM	UART	Flash Space	Bootloader	Programming Interface
 <b>Arduino Pro 3.3V/8MHz</b>	ATmega328	3.3V - 12V	3.3V	8MHz	14	6	6	1	32Kb	AtmegaBOOT	FTDI-Compatible Header
 <b>Arduino Pro 5V/16MHz</b>	ATmega328	5 - 12V	5V	16MHz	14	6	6	1	32Kb	AtmegaBOOT	FTDI-Compatible Header
 <b>Ethernet Pro (retired)</b>	ATmega328	7-12V	5V	16MHz	14	6	6	1	32Kb	AtmegaBOOT	FTDI-Compatible Header
 <b>Arduino Mega 2560 R3</b>	ATmega2560	7-12V	5V	16MHz	54	16	14	4	256Kb	STK500v2	USB via ATmega16U2





# Schede Arduino Mini

Item	uC	Input Voltage	System Voltage	Clock Speed	Digital I/O	Analog Inputs	PWM	UART	Flash Space	Bootloader	Programming Interface
 <b>Arduino Mini 5</b>	ATmega328	7-9V	5V	16MHz	14	6	8	1	32Kb	AtmegaBOOT	Serial Header
 <b>Arduino Pro Mini 3.3V/8MHz</b>	ATmega328P	3.3V - 12V	3.3V	8MHz	14	6	6	1	32Kb	AtmegaBOOT	FTDI-Compatible Header
 <b>Arduino Pro Mini 5V/16MHz</b>	ATmega328	5 - 12V	5V	16MHz	14	6	6	1	32Kb	AtmegaBOOT	FTDI-Compatible Header
 <b>Arduino Fio</b>	ATmega328P	3.3V - 12V	3.3V	8MHz	14	8	6	1	32Kb	AtmegaBOOT	FTDI-Compatible Header or Wireless via XBee <sup>1</sup>
 <b>Mega Pro Mini 3.3V</b>	ATmega2560	3.3-12V	3.3V	8MHz	54	16	14	4	256Kb	STK500v2	FTDI-Compatible Header
 <b>Pro Micro 5V/16MHz</b>	ATmega32U4	5 - 12V	5V	16MHz	12	4	5	1	32Kb	DiskLoader	Native USB





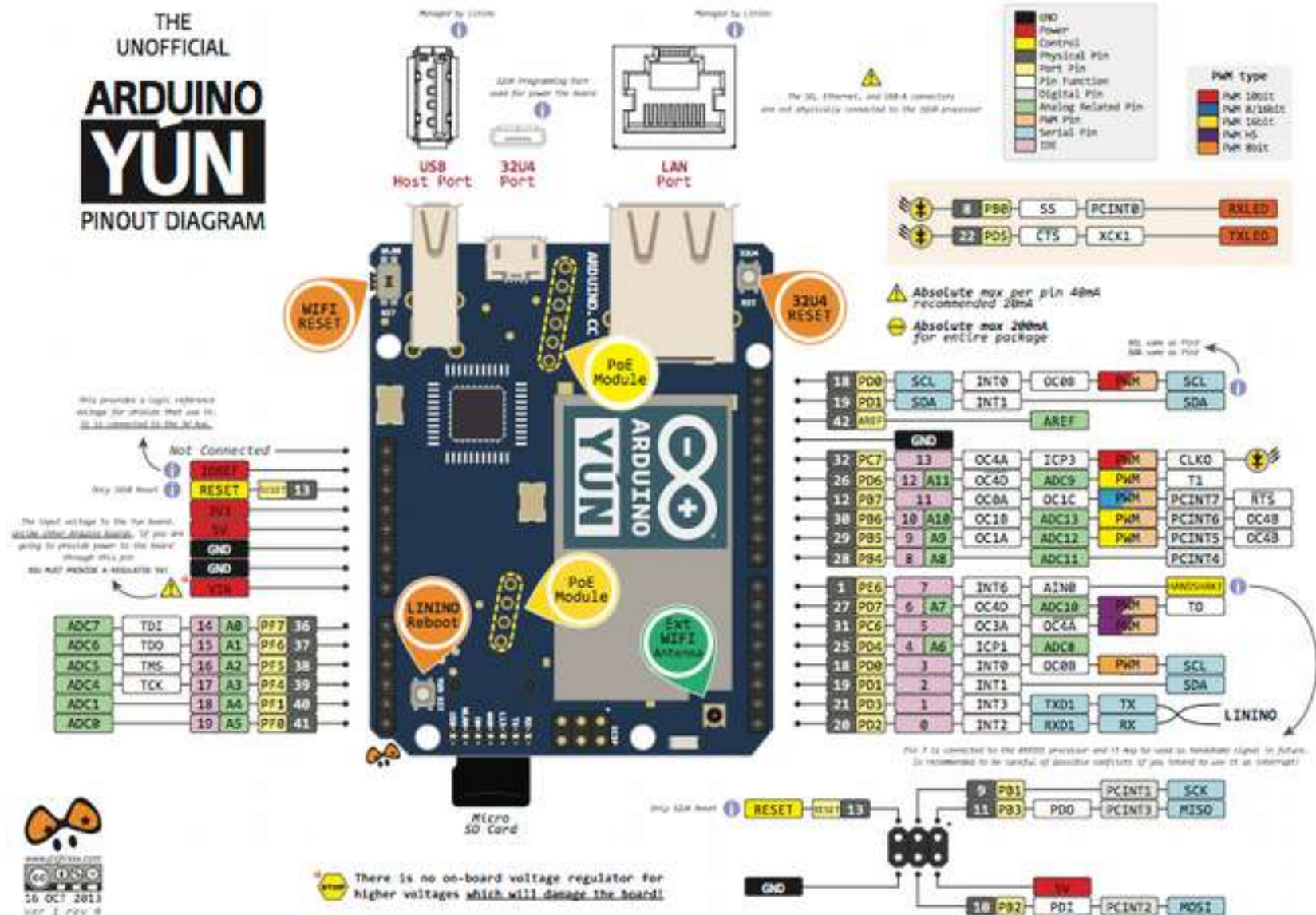
# Arduino con CPU Linux Embedded

- Accoppiata CPU **Embedded Linux** + **MCU** Atmel
  - Compatibili al 90% con i precedenti
  - Pronti per l'**Internet of Things**
  - Si avvicinano a Raspberry PI come applicazioni, mantenendosi però **orientati anche al controllo elettronico a basso livello**
- 
- Arduino Yun (32U4 + AR 9331)
  - Arduino TRE (32U4 + Sitara ARM Cortex A8)



# Arduino Yun

## THE UNOFFICIAL ARDUINO YUN PINOUT DIAGRAM

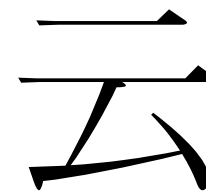


www.radioamatori.it  
16 OCT 2013  
ver 1.00.0



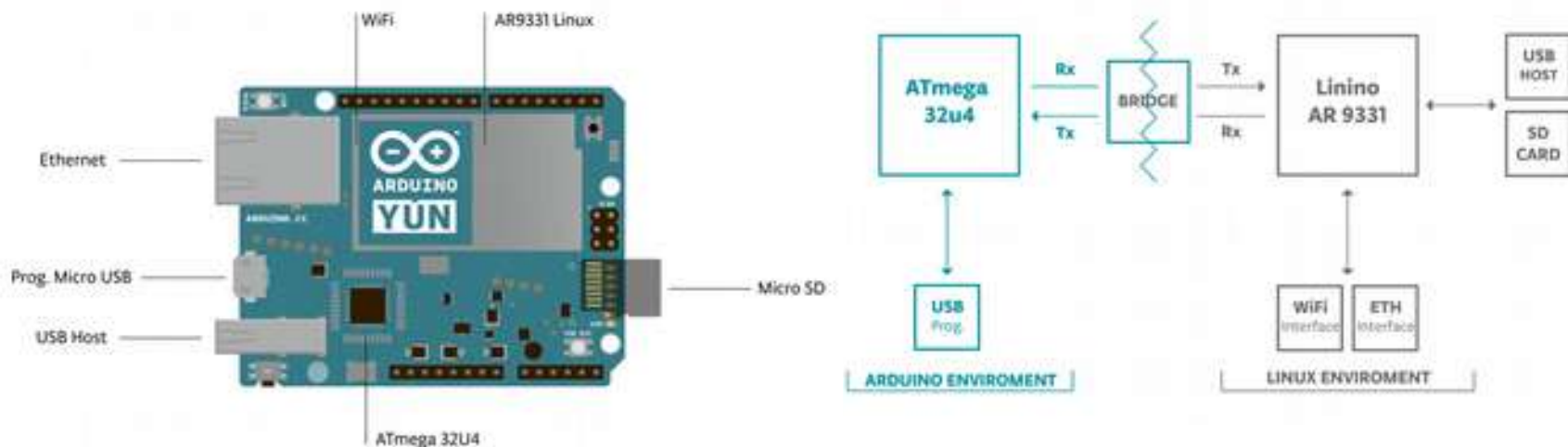


# Arduino Yùn



Yùn = Internet + Arduino → “Internet of Things”

“cloud”  
(in  
mandarino)



MCU simile a Leonardo ma la seriale 1 è riservata per comm con il SoC AR 9331



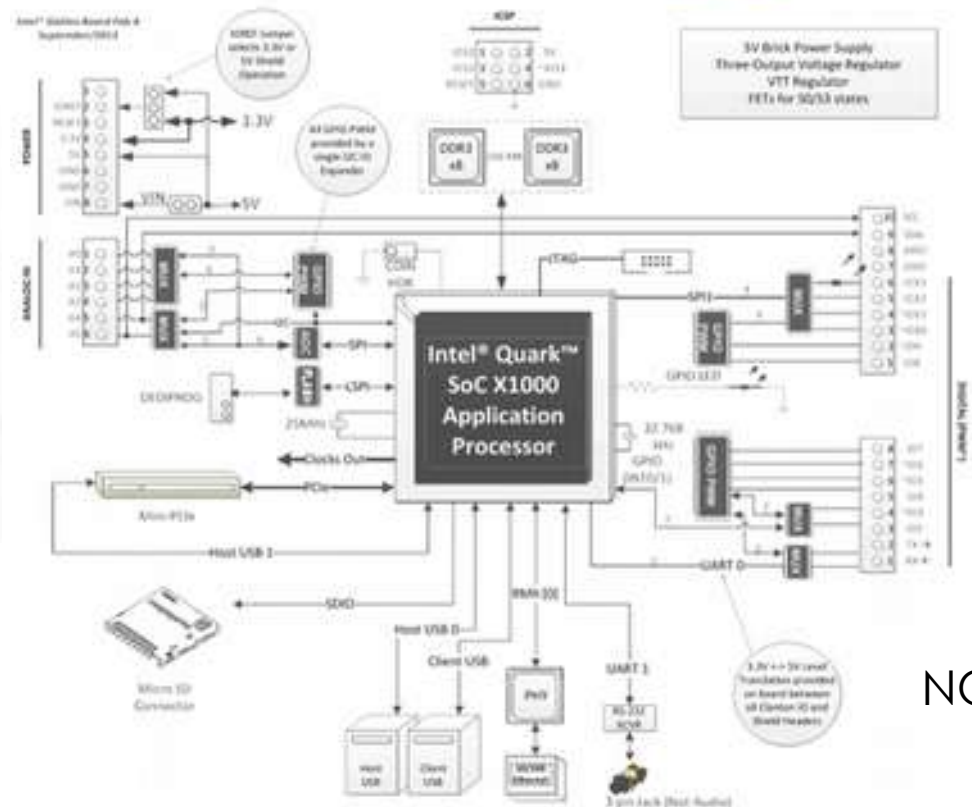
# Linux Embedded Board

# Intel Galileo

(Arduino Certified)



- SoC Intel Quark X1000 400 MHz
- CPU x86 compatible, low power for Internet of things
- ACPI, PCI Express, Ethernet, USB, UART, RS-232, RTC



NOV 2013

# Alternative: Teensy

## Key Features:

- USB can be any type of device
- AVR processor, 16 MHz
- Single pushbutton programming
- Easy to use Teensy Loader application
- Free software development tools
- Works with Mac OS X, Linux & Windows
- Tiny size, perfect for many projects
- Available with pins for solderless breadboard
- Very low cost & low cost shipping options

Specification	Teensy 2.0	Teensy++ 2.0	Teensy 3.0	Teensy 3.1
Processor	ATMEGA32U4 8 bit AVR 16 MHz	AT90USB1286 8 bit AVR 16 MHz	MK20DX128 32 bit ARM Cortex-M4 48 MHz	MK20DX256 32 bit ARM Cortex-M4 72 MHz
Flash Memory	32256	130048	131072	262144
RAM Memory	2560	8192	16384	65536
EEPROM	1024	4096	2048	2048
I/O	25, 5 Volt	46, 5 Volt	34, 3.3 Volt	34, 3.3V, 5V tol
Analog In	12	8	14	21
PWM	7	9	10	12
UART,I2C,SPI	1,1,1	1,1,1	3,1,1	3,1,1
Price	<a href="#">\$16.00</a>	<a href="#">\$24.00</a>	<a href="#">\$19.00</a>	<a href="#">\$19.80</a>



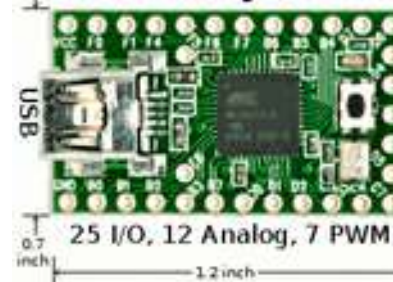
**Teensy 3.0**



## Software Development Tools

- WinAVR C compiler
- Teensyduino, add-on per Arduino IDE.

**Teensy 2.0**

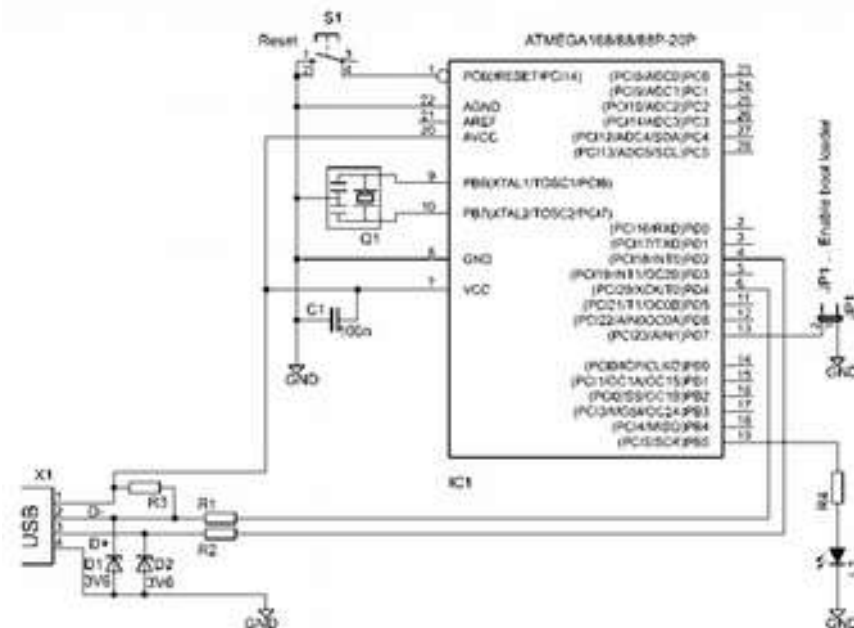


**Teensy++ 2.0**



# Arduino minimale e fatto-in-casa

One Chip (sized) Arduino



Kimio Kosaka

[http://make.kosakalab.com/arduino/obaka/project-5/index\\_en.html](http://make.kosakalab.com/arduino/obaka/project-5/index_en.html)

USBaspLoader (2 kB)  
Non è necessario un chip FTDI!

<http://www.obdev.at/products/vusb/usbasploader.html>







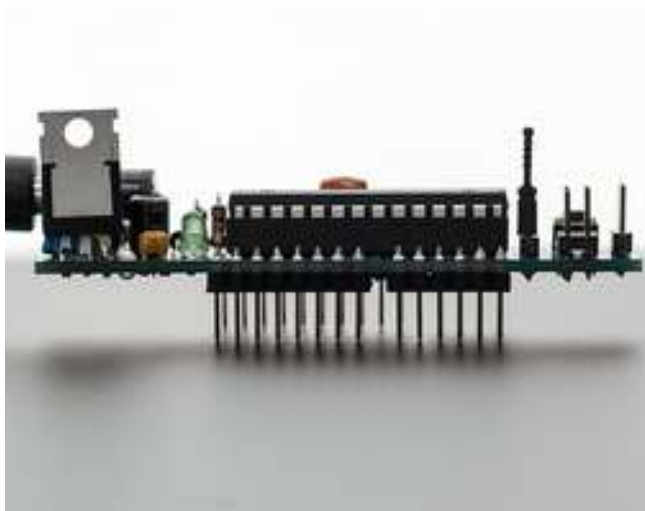
# Altre Versioni



- **Arduino at Heart:** qualsiasi prodotto con MCU supportato dall' IDE di Arduino che su richiesta del fabbricante può essere marchiato "Arduino at Heart" previo pagamento di una piccola royalty.

- **Arduino clone:** copie franche delle schede originali ma costruite da altri fabbricanti con minime modifiche e marchio diverso.
- **Arduino approved/certified/compatible:** prodotti approvati per l'uso con l'IDE Arduino e compatibili con alcuni shield originali.

- Ruggeduino
- Luigino
- Thindduino
- Ardweeny
- Boarduino
- Olimexino
- Meaple Leaf
- Hamstack



Adafruits: Boarduino. Arduino compatibile per uso su breadboard.





# Programmazione, sketch, librerie e configurazione

```
10 // Pin 13 has an LED connected on most Arduino boards
11 // give it a name:
12 int led = 13;
13
14 // the setup routine runs once when you press reset:
15 void setup() {
16   // initialize the digital pin as an output.
17   pinMode(led, OUTPUT);
18 }
19
20 // the loop routine runs over and over again forever:
21 void loop() {
22   digitalWrite(led, HIGH); // turn the LED on (HIGH
23   delay(1000);             // wait for a second
24   digitalWrite(led, LOW);  // turn the LED off by m
25   delay(1000);             // wait for a second
26 }
27
```





# Processo di generazione del codice



- Controllo del codice e riadattamento
- Compilazione con avr-gcc
- Generazione dei file oggetto
- Linking
- Upload del codice hex con AVRDUDE





# Le due vie alla programmazione di Arduino

## • Base

- IDE di Arduino
- Librerie già pronte
- Progetti già pronti da modificare
- Alto livello

Hobbisti e anche utenti “non tecnici”

## • Avanzata

- IDE e compilatori a discrezione
- Debugging
- **Compatibile con librerie già pronte e progetti già pronti da modificare**
- By-pass di funzionalità per accedere a basso livello
- **Uso completo delle funzionalità e prestazioni della MCU**
- HW già pronto solo da acquistare e standardizzato

Prototipazione, ricerca, sperimentazione.





# Librerie Arduino

## Core Distribution Libraries

EEPROM  
SD cards

GSM  
WiFi  
Ethernet

LiquidCrystal  
TFT

Stepper  
Servo

SPI  
Wire (TWI and I2C)  
SoftwareSerial

Audio\*  
Scheduler\*  
USBHost\*

NewSoftSerial  
OneWire  
Webduino  
Sserial2Mobile  
X10  
Xbee  
SerialControl

FFT  
Tone  
TLC5940  
IRRemote

Keypad

FT-857D  
RTTY

Messenger

## Arduino Community Libraries







# Esempi sketch

```
#include <Arduino.h>

void setup()
{
  pinMode(13, OUTPUT);
}

void loop()
{
  dah(); dit(); dah(); dit();
  space();
  dah(); dah(); dit(); dah();
  space(); space();
}
```

```
void dit()
{
  digitalWrite(13,HIGH);
  delay(100);
  digitalWrite(13, LOW);
  delay(100);
}
```

```
void dah()
{
  digitalWrite(13,HIGH);
  delay(100*3);
  digitalWrite(13, LOW);
  delay(100);
}
```

```
void space()
{
  delay(100);
}
```





# Esempi sketch

ryryry.ino:

```
#include <RTTY.h>

void setup()
{
    RTTY.attach(9,3);
}

void loop()
{
    RTTY.tx("CQ DX");
}
```

↓  
aFSK RTTY  
sul pin 9  
(salvo armoniche!)

RTTY.cpp

RTTY.cpp:

```
...
void RTTY5::rtty_txbit (int bit)
{
    if (bit)
    {
        // high
        tone(pa,2295,BAUD_RATE*1000);
    }
    else
    {
        // low
        tone(pa,2125,BAUD_RATE*1000);
    }
    delayMicroseconds(BAUD_RATE);
}
...
```

RTTY.h:

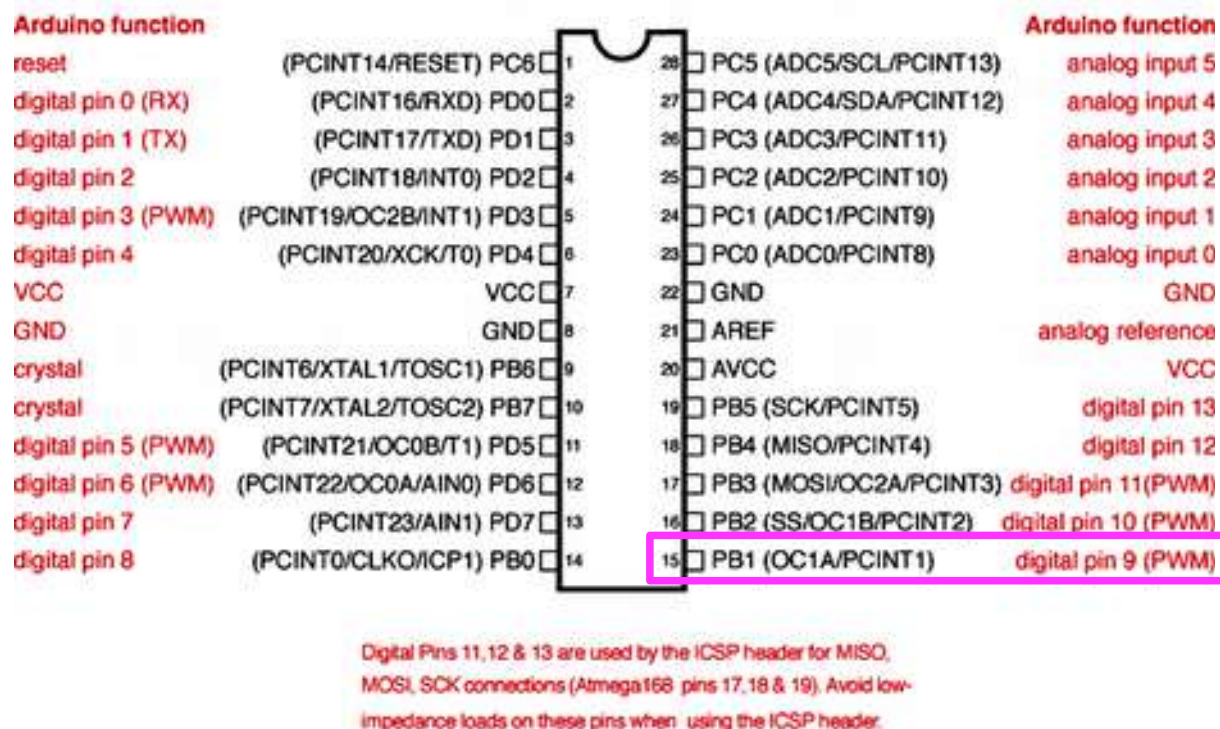
```
...
class RTTY5
{
public:
    RTTY5();
    void attach(int pina, int pinb);
    void tx (char * string);
private:
    void rtty_txbyte (char c);
    void rtty_txbit (int bit);

};
...
```

Tone.cpp



# Mappa dei pin su Arduino



## Arduino framework

```
pinMode(9, OUTPUT);  
...  
digitalWrite(9, HIGH);
```

PIN 9 → PB1

## AVR C code

```
DDRB |= B00000010;  
...  
PORTB |= B00000010;
```



# Yun: Linino

- Embedded Linux per router/access point, basato su OpenWRT

```
10.1.1.7 - PuTTY
login as: root
root@10.1.1.7's password:

BusyBox v1.19.4 (2019-08-07 16:16:02 CEST) built-in shell (ash)
Enter 'help' for a list of built-in commands.

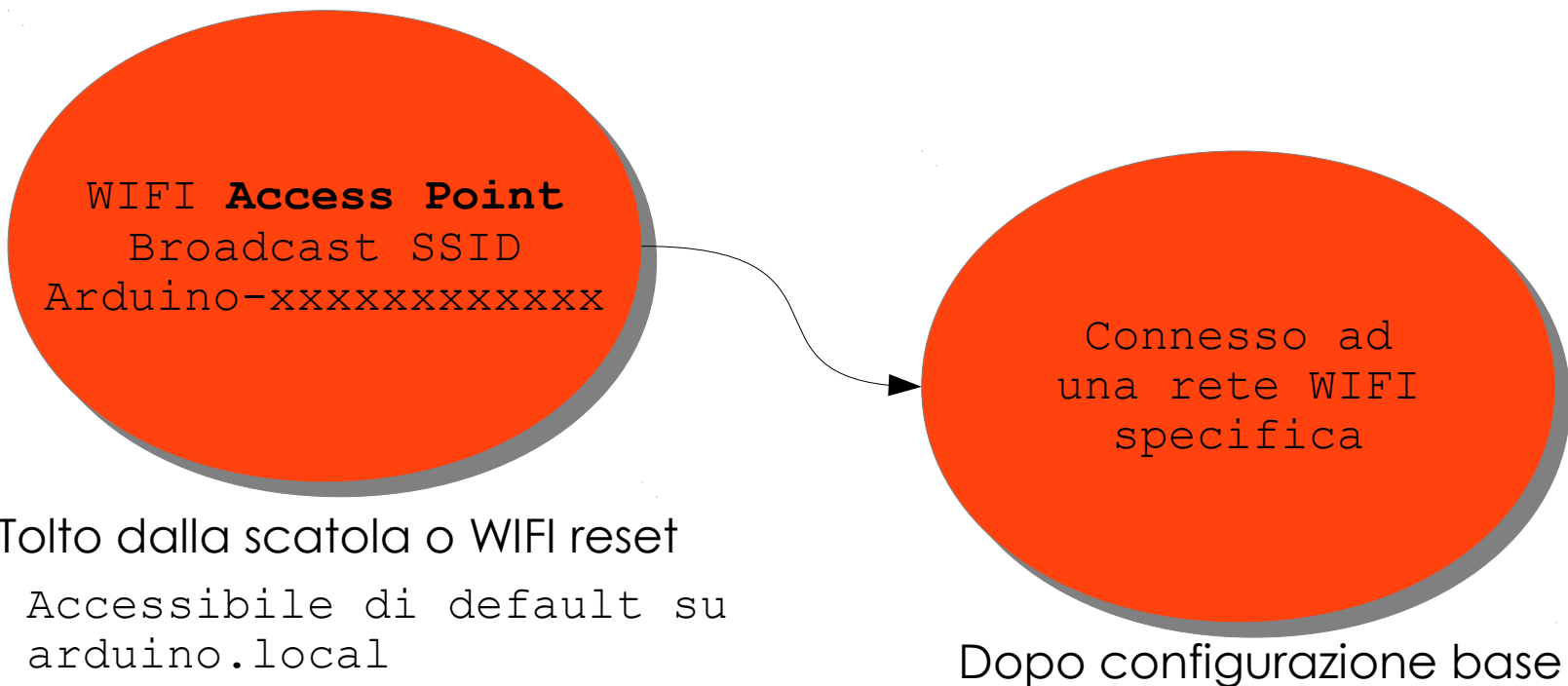
YUN
WIRELESS FREEDOM

root@ArduinnoYun:~# telnet localhost 6571
```





# Configurazione Yun



```
# ssh 192.168.240.1
```

Oppure con  
Interfaccia Web

- WIFI
- Ethernet
- miniCOM 

```
# ifconfig eth0
```





# Yun: Tool gestione Linino



- Qualsiasi emulatore di terminale (da Linux)
- PuTTY: client TELNET, SHH (da Windows)
- WinSCP: FTP, SFTP, SCP (da Windows)
- Arduino IDE 1.5 (beta)





# Yun: Linino UCI

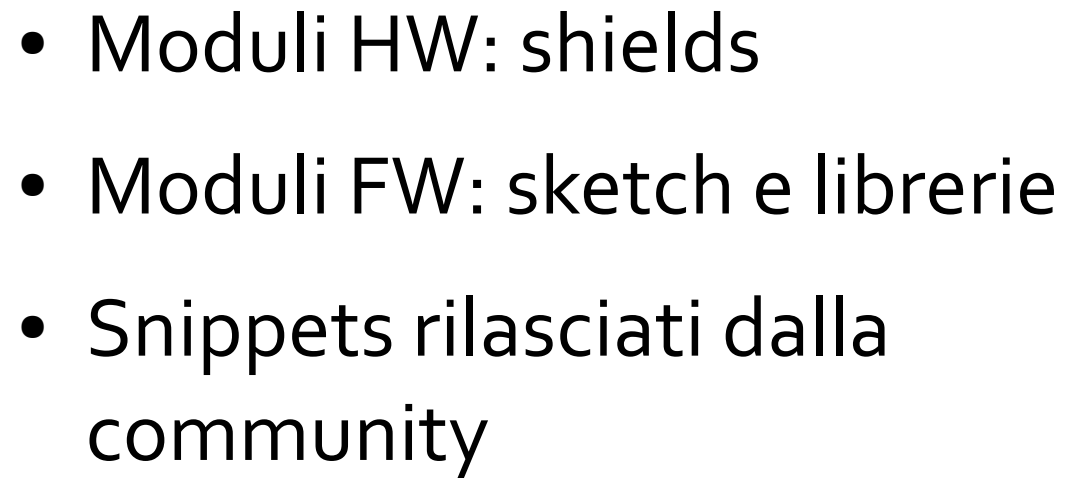
- Unified Configuration Interface

uci files in /etc/config:

arduino  
dhcp  
dropbear  
firewall  
fstab  
luci  
network  
system  
ubootenv  
ucitrack  
uhttpd  
wireless

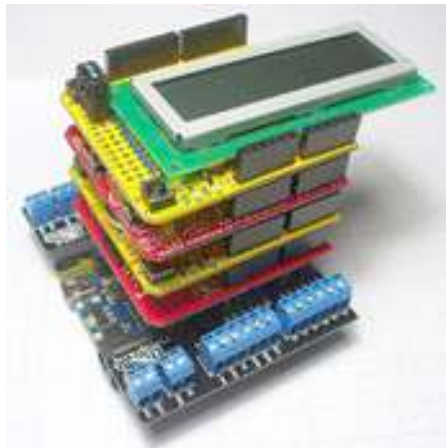
<http://linino.org>





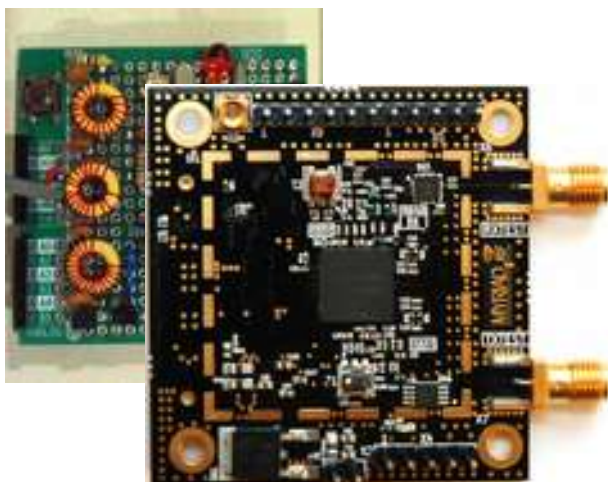
*Braun Lectron, 1967*

# Shield



<http://www.shieldlist.org>

... controllo motori DC, RC servo, stepper, WiFi, display LCD, Ethernet, CANBus, Bluetooth, XBee, gestione memorie, matrici a LED, relay, GPS, prototipazione rapida ...



Shield radio

- **Shield: daughter board impilabili su Arduino**
- **Modularità**
- di terze parti o customizzati



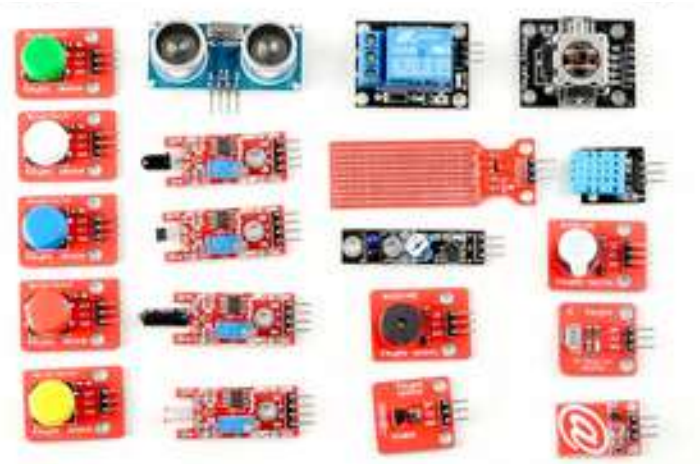
Myriad RF

Open-source FP-RF

<http://myriadrf.org>



# Mini-moduli pronti da montare



Sensori di distanza a US, microfonici, IR, relay ecc.



LED allarme, fotodiodi, pulsanti, potenziometri, sensori di temperatura, fotoresistenze.







# Librerie Arduino

## Core Distribution Libraries

EEPROM  
SD cards

GSM  
WiFi  
Ethernet

LiquidCrystal  
TFT

Stepper  
Servo

SPI  
Wire (TWI and I2C)  
SoftwareSerial

Audio\*  
Scheduler\*  
USBHost\*

NewSoftSerial  
OneWire  
Webduino  
Sserial2Mobile  
X10  
Xbee  
SerialControl

FFT  
Tone  
TLC5940  
IRRemote

Keypad

FT-857D  
RTTY

Messenger

## Arduino Community Libraries





# Applicazioni Ham Radio



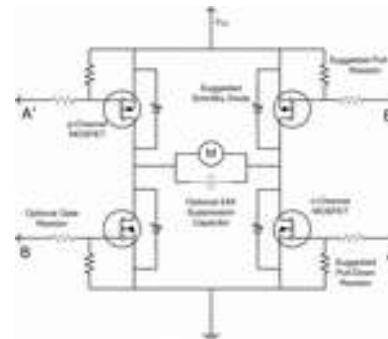
*W1AW, ARRL Headquarter*



# Principali funzioni svolte da Arduino in applicazioni OM

## Gestione di segnali per controllo motori

- Rotori d'antenna
- Rotori per condensatori variabili in retroazione su ROS



“Cascata”  
Leigh, WA5ZNU

## Generazione di segnali e ricezione digitale (audio)

- Keyer con funzionalità avanzate
- Modulatore FSK e per semplici modi digitali con DDS
- Audioprocessing: modi digitali

## Interfacciamento con moduli e/o PC

- Gestione ricetrasmittitori attraverso seriale o porte proprietarie
- Gestione moduli e schede a RF o per applicazioni specifiche tramite SPI, I2C
- Gestione moduli per applicazioni di controllo motori avanzate, su CAN bus
- Comunicazione con moduli Ethernet e WiFi 802.11g
- Uso su reti TCP/IP, webserver

Moduli RTX

Moduli QRSS

Moduli APRS

Moduli misure automatiche

Internet / Cloud

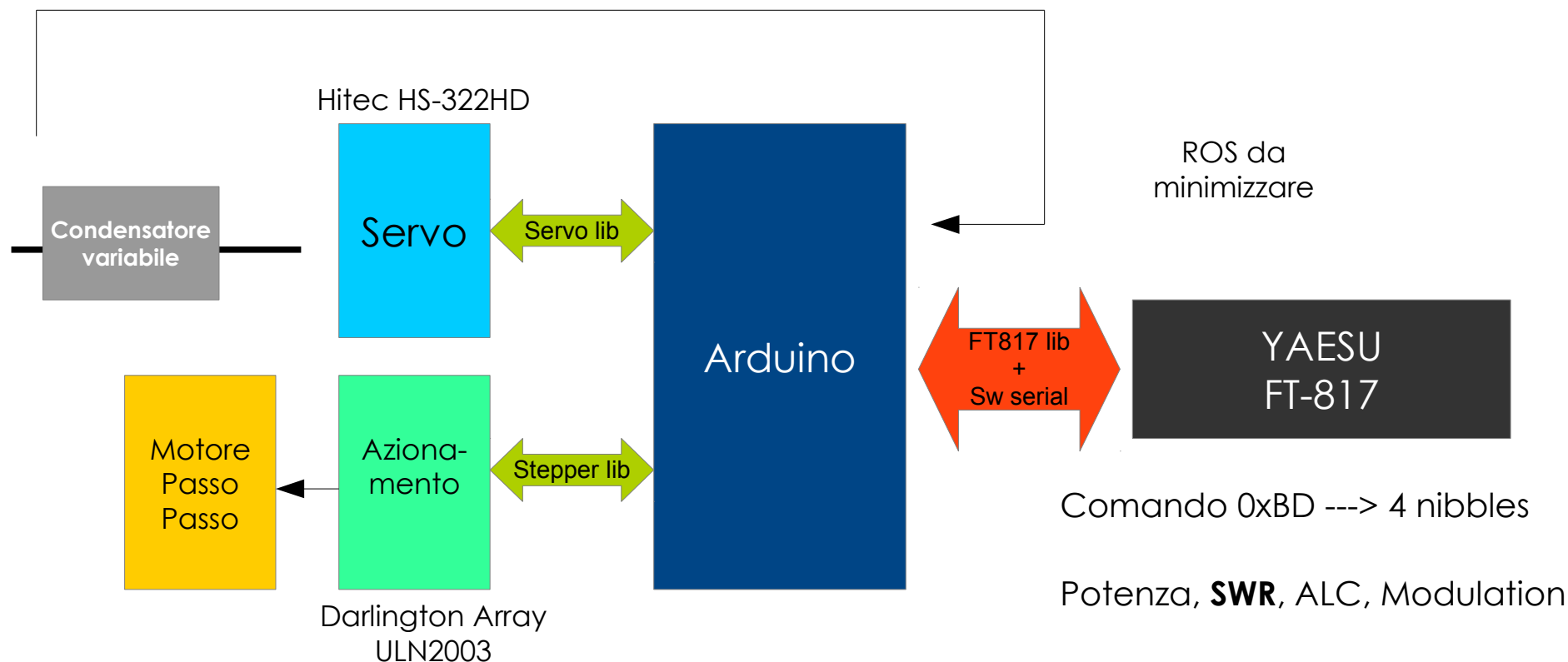
... DITUTTO !!!





# Applicazioni

## Accordatore Loop Magnetico



[http://www.ka7oei.com/ft817\\_meow.html](http://www.ka7oei.com/ft817_meow.html)

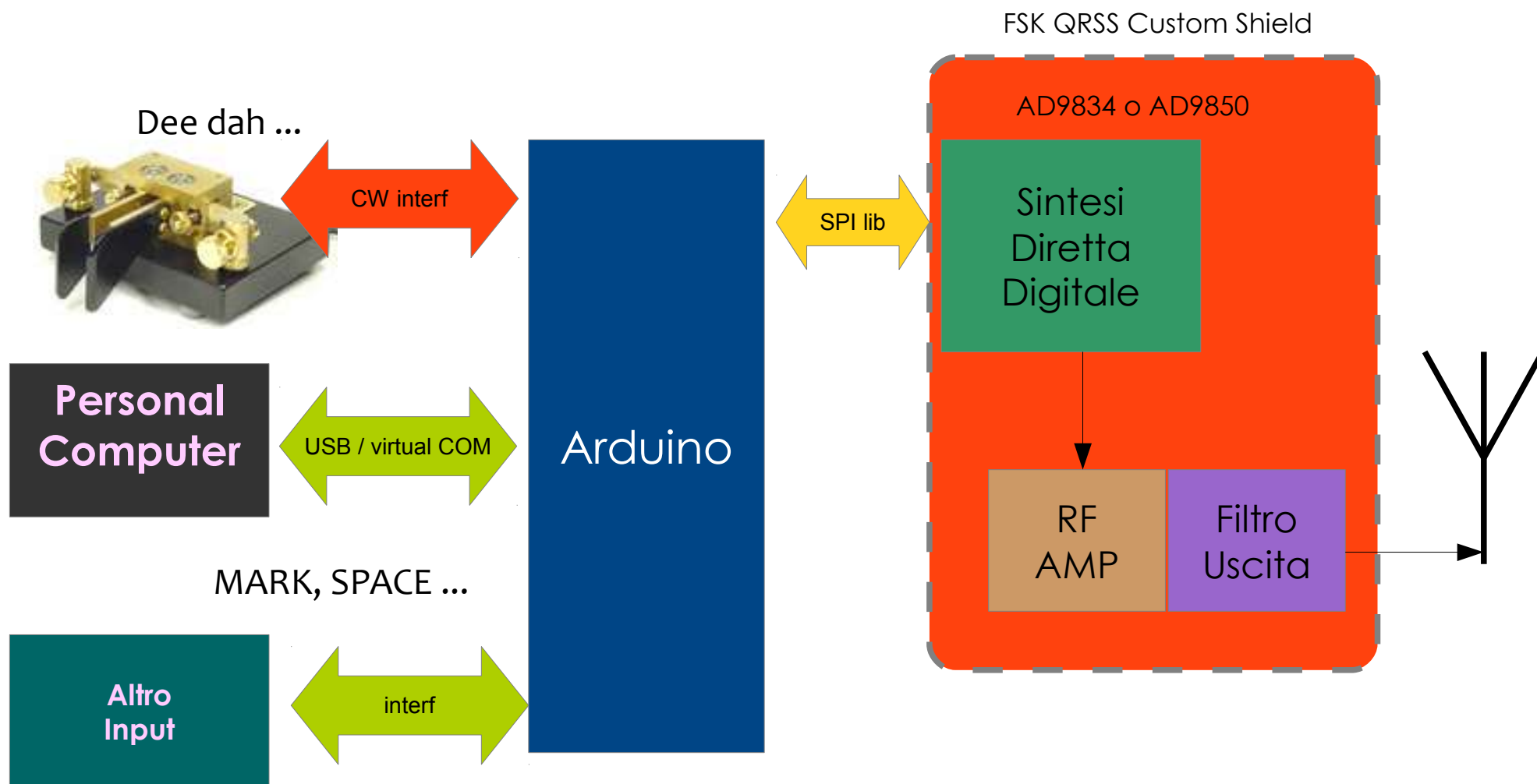
<http://code.google.com/p/ft-817-automatic-loop/>





# Applicazioni

## Trasmittitore FSK per QRSS

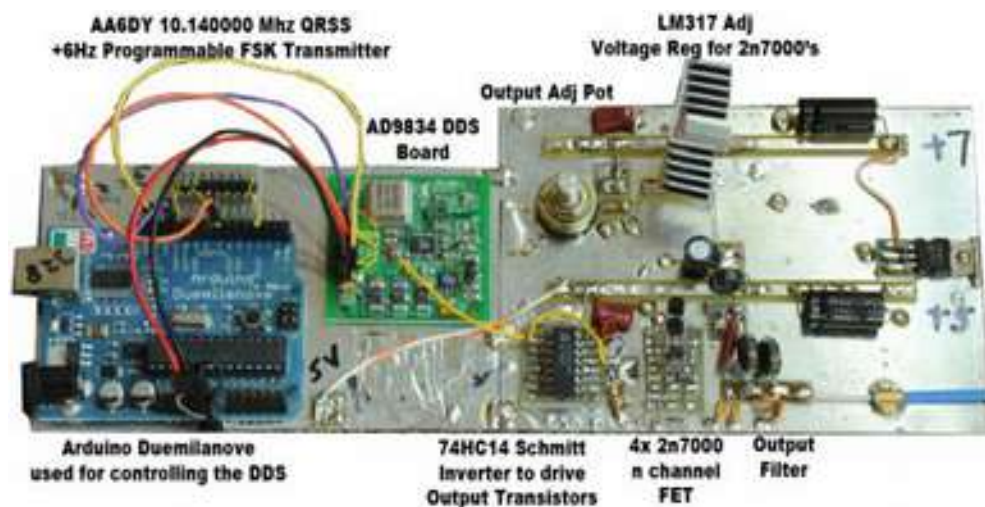


# ... dal web ...



- 10.140 MHz
- Shield per QRSS con FSK CW (F1A)
- Realizzata su ProtoShield (Sparkfun.com)

- Interfacciamento SPI
- con Direct Digital Synthesizer AD9834



Michael Seedman, AA6DY - Highland Park, IL <http://aa6dy.com>



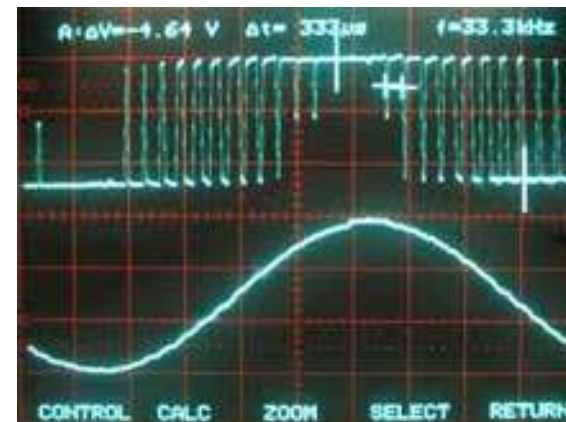


# Applicazioni

# Generazione di portanti audio

## con Sintesi Digitale Diretta tramite PWM

- Generazione di portanti audio a 2000-3000 Hz
- Possiamo evitare l'uso di un IC ad-hoc per la generazione
- Usiamo la periferica di Atmega328: compare/timer/counter per generare un PWM a duty cycle variabile in forma sinusoidale
- Per questo è più agevole by-passare il controllo dei timer di Arduino (non usare analogWrite)
- Impostando opportunamente i registri di configurazione della periferica è possibile avere un periodo del clock al timer pari a 1/16 MHz
- E' possibile controllare il decadimento del segnale CW evitando quindi i click



Nanokeyer di N6SN, Bud Tribble

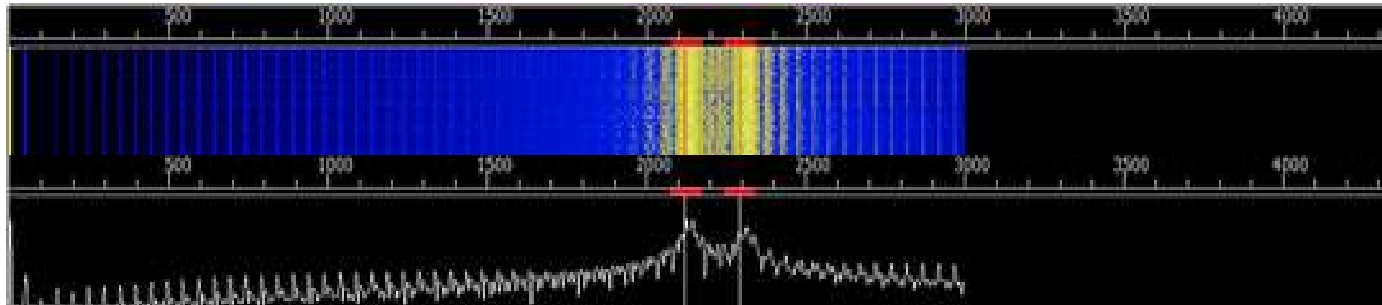
Sperimentazione!

<http://hamradioprojects.com/authors/wa5znu/+nanokeyer/>

[http://en.wikipedia.org/wiki/Bud\\_Tribble](http://en.wikipedia.org/wiki/Bud_Tribble)

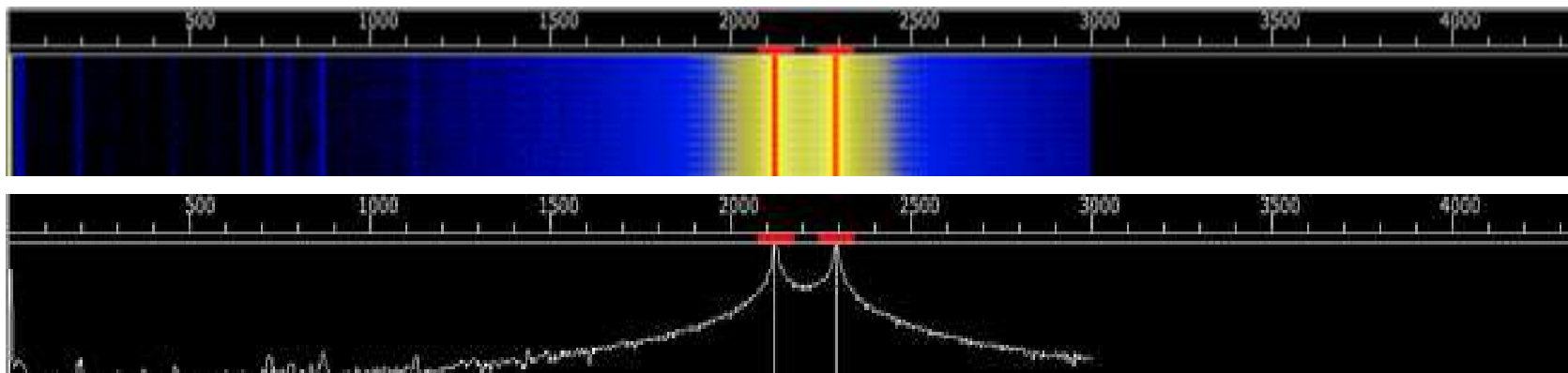


# Applicazioni RTTY



- Usando le librerie: RTTY modificata e Tune

ONDE QUADRE (esempio visto)



Spettri audio del  
segnale generato.

Esperimenti ricezione,  
eseguiti con fldigi di  
<http://www.w1hkj.com>

**Codice disponibile  
agli interessati**

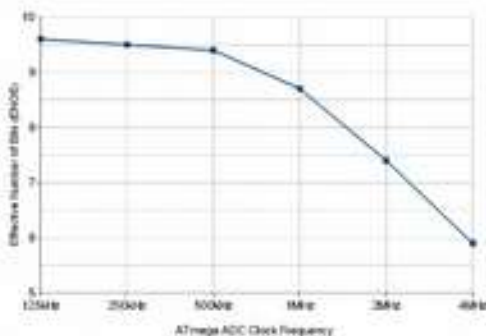
- Usando la libreria RTTY modificata e una DDS (usando i registri dei timer in modo avanzato)
- Segnale audio sinusoidale senza armoniche realizzato con una onda quadra a frequenza ultrasonica con D.C. variabile opportunamente (DDS con PWM)



# Applicazioni

## Processing sul segnale audio

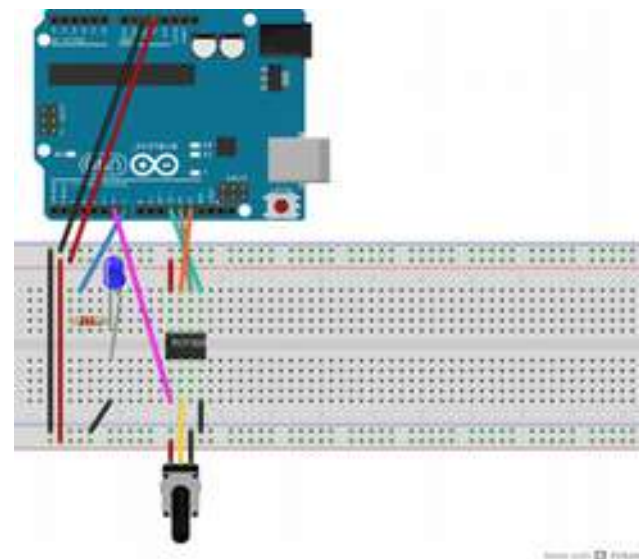
- Gli ADC della MCU di Arduino sono semplici e non sono progettati per l'audio
- Di default campiona a circa 9600 Hz. E' possibile far lavorare l'ADC fino a 1 MHz ottenendo frequenze di campionamento di 77 kHz. Incrementando la frequenza dell'ADC la risoluzione viene compromessa.



- Nonostante questo è possibile campionare a 40 kHz un solo canale con discreti risultati.
- E' possibile usare un componente esterno come Microchip MCP3002 Dual Channel 10-Bit A/D Converter con interfaccia SPI arrivando a frequenze di campionamento di 200 kHz.

<http://www.openmusiclabs.com/>

<http://interface.khm.de/>



### Applicazioni:

- Demodulazione audio segnali in uscita da ricevitore SSB eventuale decodifica: RTTY... PSK31 (?)
- Prestazioni difficilmente comparabili con una buona scheda audio, ma possibile uso anche in assenza PC...

## Sperimentazione!







"A radio in which some or all of the physical layer functions are software defined"



SDR Forum and IEEE

# RX SDR con Arduino?

## Direct sampling

*RF*

High Performance  
Software Defined Radio  
openhpsdr.org, tapr.org

## Direct Conversion Quadrature Sampling (DCQS)

*baseband*

Esempi: SoftRock  
Lite, EZCAP DVB-  
T/FM/DAB ...

## Digital Baseband

*IF*

Esempi: FLEXRadio  
SDRs, USRP, HPSDR,  
AMRAD Charleston  
SDR, DSP-IP

Fonte: F. Doremberg N4SPP [http://www.nonstopsystems.com/radio/frank\\_radio\\_sdr.htm](http://www.nonstopsystems.com/radio/frank_radio_sdr.htm)

Arduino UNO: troppo lento anche per processare i segnali I e Q.

Con tecniche avanzate è possibile usarlo per audio processing (con poche pretese)

- Arduino DUE (SAM<sub>3</sub>X8E)
- Olimexino (STM32F103)

Full audio 20 kHz real time processing capable!

Sperimentazione di semplici algoritmi per SDR





# Applicazioni: tracker con APRS



Arduino  
UNO

TTL 9600 bps  
NMEA protocol

Modulo  
GPS



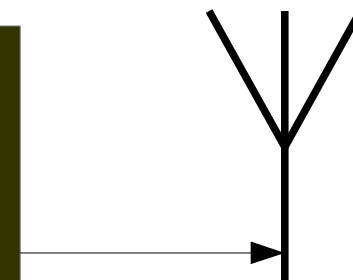
AINs

Sensori

Temp.: LM60

~OUT

Filtro  
&  
TX



Reg. livello aFSK

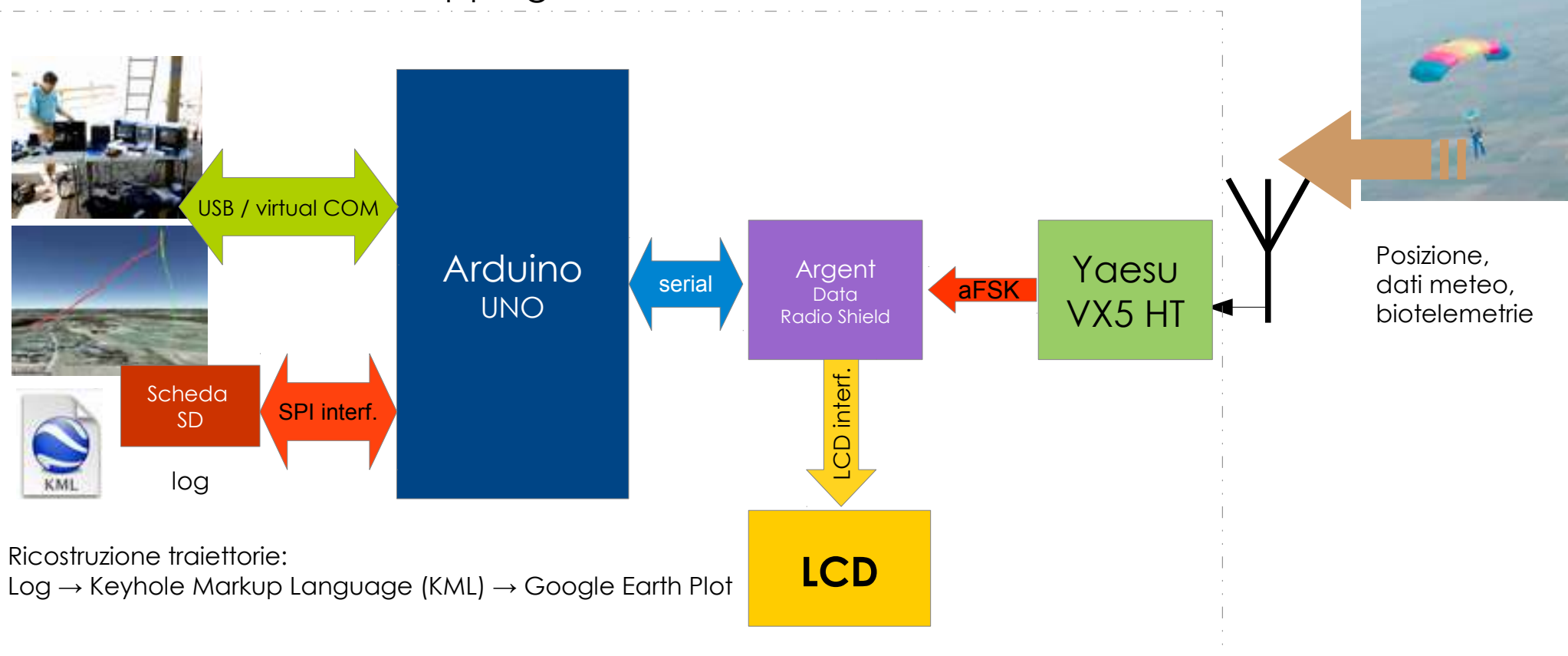
```
DJ7OO-11>APRS,WIDE2-1:/  
113801h  
4957.60N/  
00811.98E  
0000/000/  
A=000895/Ti=58/Te=107/V=8439  
Klaus JN49CX via Trackuino
```

<http://www.kh-gps.de/trackuino.htm>

# Applicazioni: logger con APRS



## Stazione alla Dropping Zone

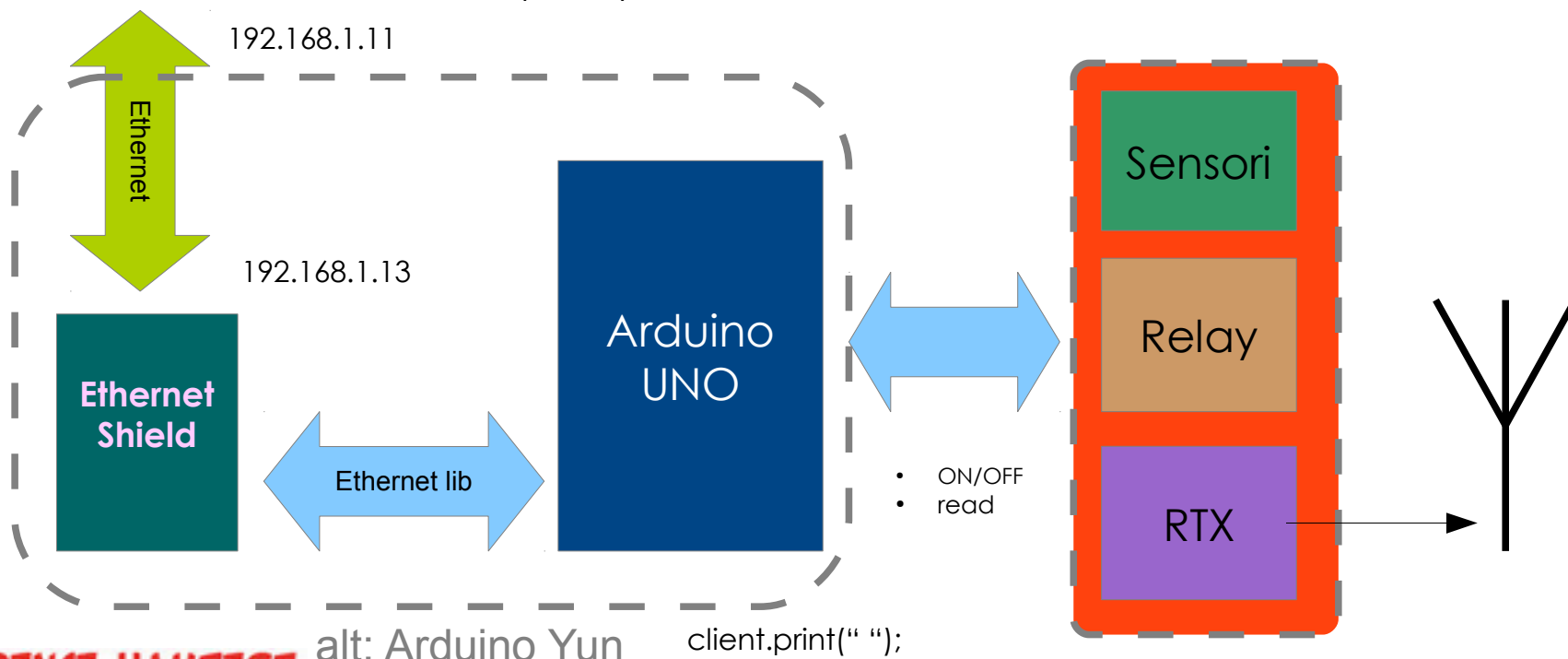
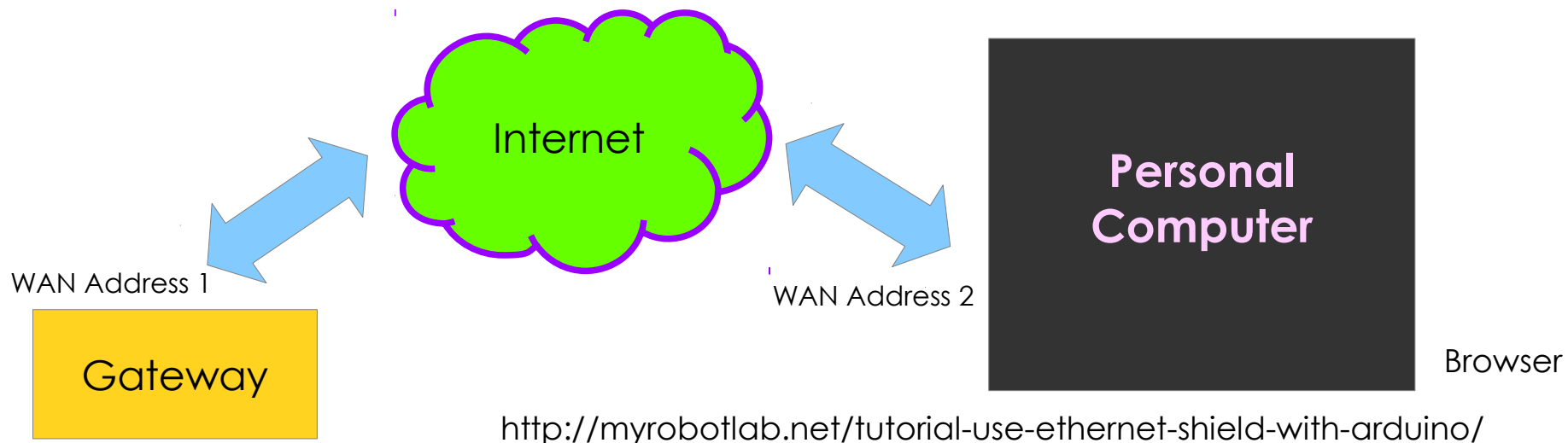


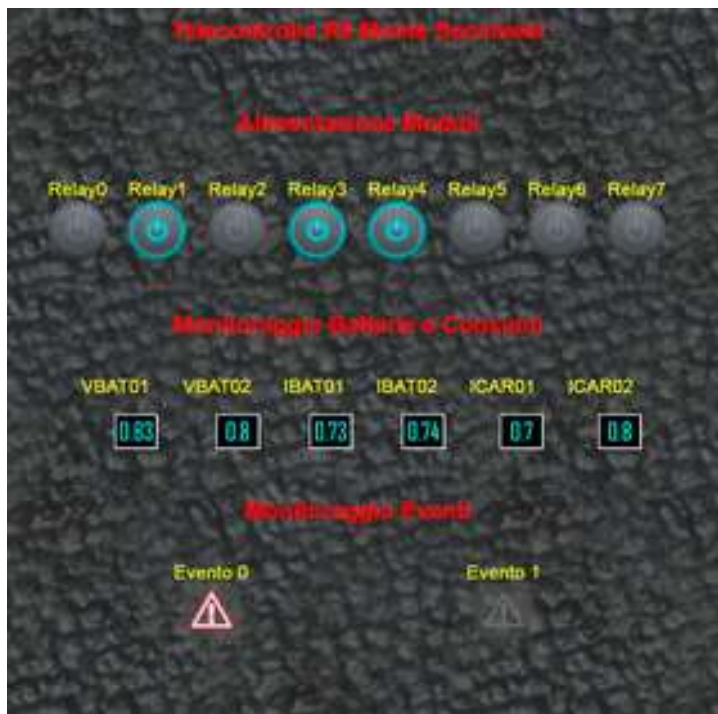
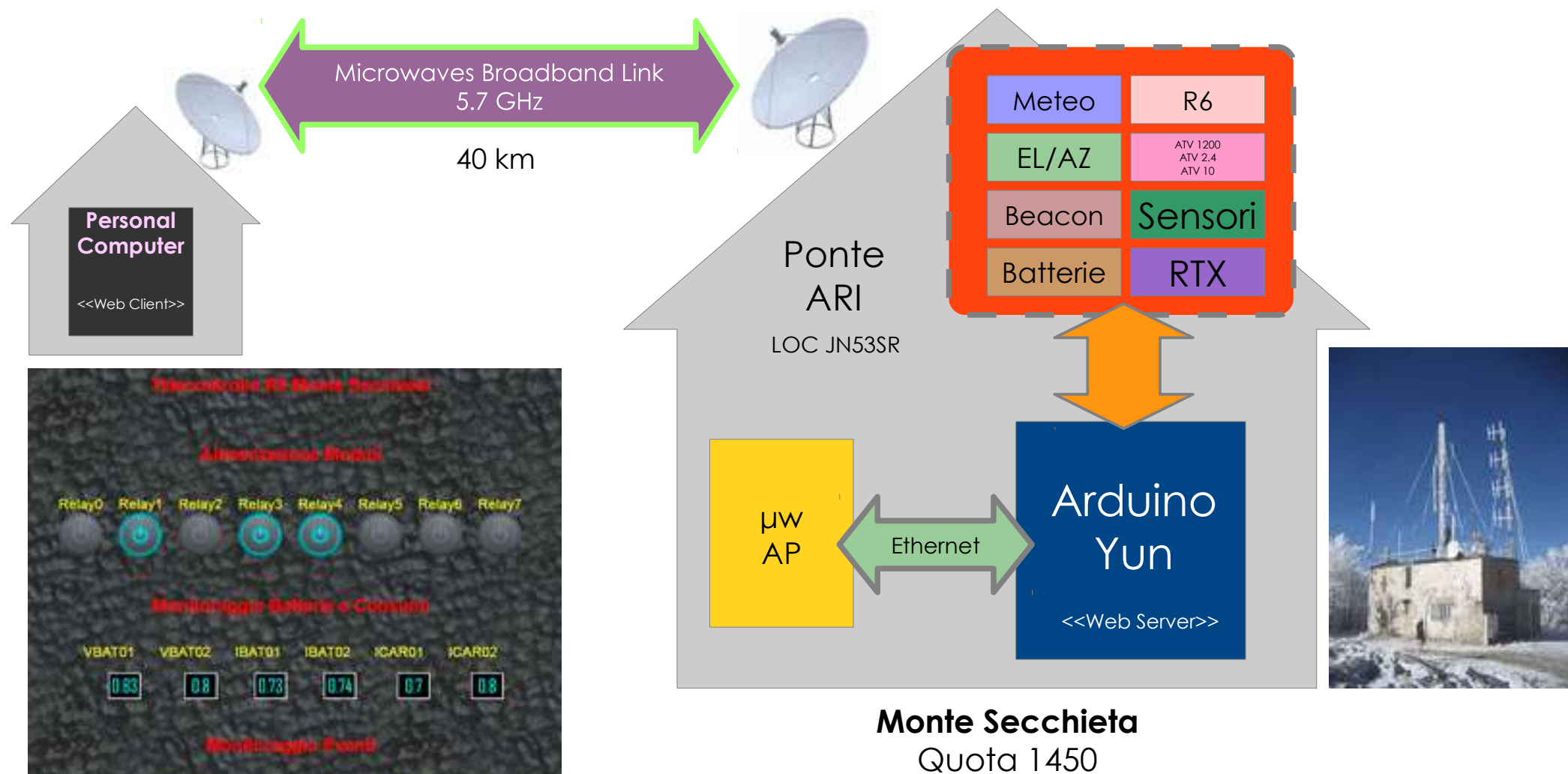
Fonti: Michael D. Pechner, NE6RD, Mark Meltzer, AF6IM  
<http://parachutemobile.org>



# Applicazioni

## Controllo remoto su Internet





**Sistema di telecontrollo** in sperimentazione, realizzato da I5XFD Franco, IK5FGJ Raffaele, IZ5IPB Fabio e IW5EKN Francesco.







# Web consigliato

Imparare  
Arduino

- <http://forum.arduino.cc>
- <http://scuola.arduino.cc/>
- <http://learn.adafruit.com>
- <http://www.open-electronics.org>



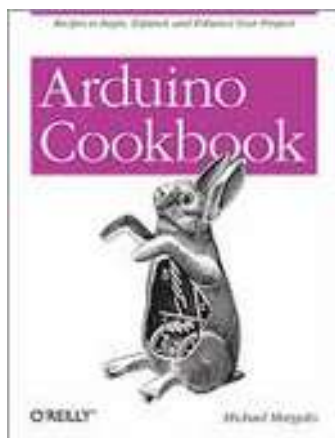
- <http://www.amrad.org>

Community di  
Sperimentazione

Associazione affiliata  
ARRL



# Libri Consigliati





# 73

**[iw5ekn@qsl.net](mailto:iw5ekn@qsl.net)**

