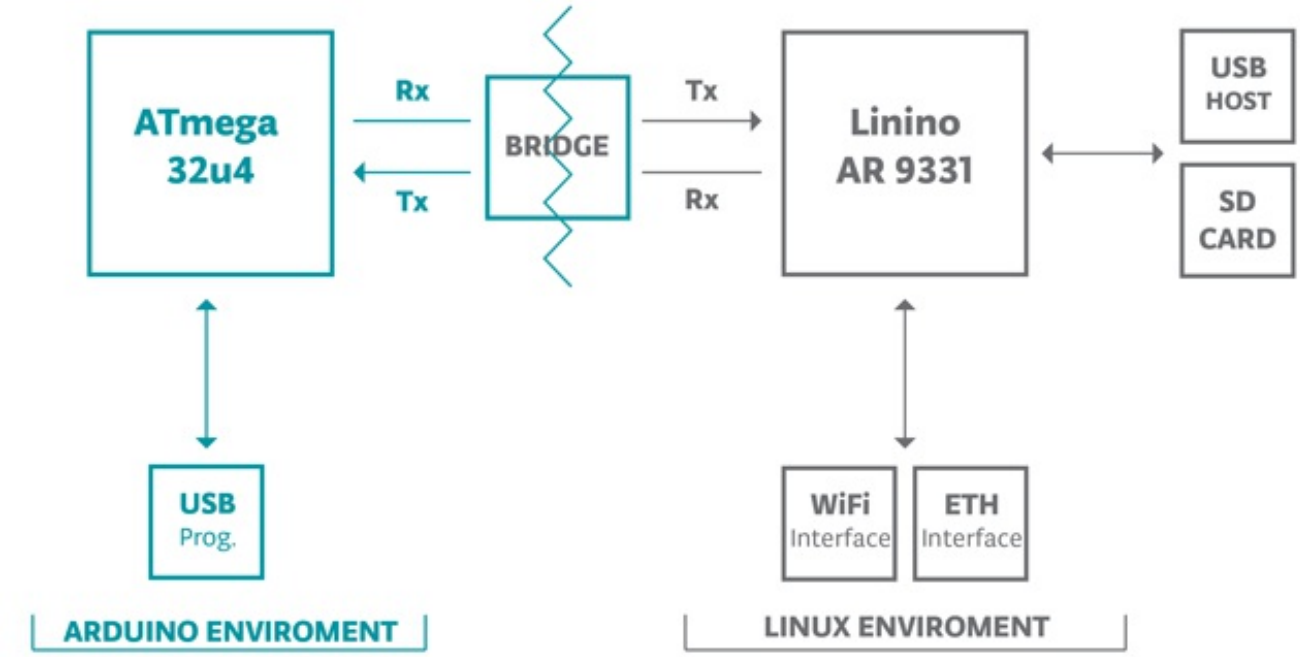
Guida utilizzo

Arduino YÙN

Capitolo Pagina

Arduino Yun

L'Arduino Yun è una scheda con incoroparti due sistemi indipendenti ma collegati fra loro. Uno è l'Arduino, con il chip ATmega 32u4 collegata alla porta MicroUSB ed hai pin, mentre il secondo chip è un Lilino AR 9331, il quale esegue una distro di linux molto leggera per sistemi embedded, OpenWRT. Il chip Lilino è collegato alla porta USB Type A, allo slot per la scheda MicroSD, alla porta Ethernet ed al modulo Wi-Fi.
I due moduli sono collegati fra solo tramite un bridge. Il quale permette di strasferire dei dati fra i due chip.



Inizializzare Arduino

Per iniziare a configurare l'Arduino Yun bisogna accedere alla shell, per configurare la password di root e vedere l'indirizzo IP. Per eseguire queste operazioni bisogna caricare sull'Arduino un programma che emuli la porta seriale tramite la porta seriale dell'Arduino, quindi caricare il seguente programma tramite l'IDE di Arduino.

```
long linuxBaud = 250000;

void setup() {
  SERIAL_PORT_USBVIRTUAL.begin(115200);  // open serial connection via USB-Serial
  SERIAL_PORT_HARDWARE.begin(linuxBaud); // open serial connection to Linux
}

boolean commandMode = false;

void loop() {
  // copy from USB-CDC to UART
  int c = SERIAL_PORT_USBVIRTUAL.read();    // read from USB-CDC
  if (c != -1) {                            // got anything?
    if (commandMode == false) {             // if we aren't in command mode...
      if (c == '~') {                       //    Tilde '~' key pressed?
        commandMode = true;                 //       enter in command mode
      } else {
        SERIAL_PORT_HARDWARE.write(c);      //    otherwise write char to UART
      }
    } else {                                // if we are in command mode...
      if (c == '0') {                       //     '0' key pressed?
        SERIAL_PORT_HARDWARE.begin(57600);  //        set speed to 57600
        SERIAL_PORT_USBVIRTUAL.println("Speed set to 57600");
      } else if (c == '1') {                //     '1' key pressed?
        SERIAL_PORT_HARDWARE.begin(115200); //        set speed to 115200
        SERIAL_PORT_USBVIRTUAL.println("Speed set to 115200");
      } else if (c == '2') {                //     '2' key pressed?
        SERIAL_PORT_HARDWARE.begin(250000); //        set speed to 250000
        SERIAL_PORT_USBVIRTUAL.println("Speed set to 250000");
      } else if (c == '3') {                //     '3' key pressed?
        SERIAL_PORT_HARDWARE.begin(500000); //        set speed to 500000
        SERIAL_PORT_USBVIRTUAL.println("Speed set to 500000");
    } else if (c == '~') {                 //     '~' key pressed?
        SERIAL_PORT_HARDWARE.write((uint8_t *)"\xff\0\0\x05XXXXX\x7f\xf9", 11); // send "bridge
shutdown" command
        SERIAL_PORT_USBVIRTUAL.println("Sending bridge's shutdown command");
      } else {                              //     any other key pressed?
        SERIAL_PORT_HARDWARE.write('~');    //        write '~' to UART
        SERIAL_PORT_HARDWARE.write(c);      //        write char to UART
      }
      commandMode = false;                  //     in all cases exit from command mode
    }
  }

  // copy from UART to USB-CDC
  c = SERIAL_PORT_HARDWARE.read();          // read from UART
  if (c != -1) {                            // got anything?
    SERIAL_PORT_USBVIRTUAL.write(c);        //    write to USB-CDC
  }
}
```

Quando il programma è caricato avviare il monitor seriale dell'IDE, con il bottone in alto a destra.
inviare un comando qualunque per controllare se la seriale funziona correttamente. Dopo di che inviare il comando `passwd`, dopo di che inserire due volte la stessa password, che sarà la password dell'utente root sul chip Lilino dell'Arduino Yun. Dopo di che eseguire il comando `ip addr show`, per potersi collegare in remoto con l'Arduino Yun,
Dopo di che collegarsi via SSH all'arduino utilizzando l'indirizzo IP prima letto, lo username `root` e la password prima inserita.
Questo per utilizzare Lilino, mentre per utilizzare la scheda Arduino, caricare tramite l'IDE ufficiale di Arduino i programmi che si desidera.

Dal lato Lilino invece bisogna utilizzare python. Per comunicare fra le due parti, Arduino ha sviluppato un sistema di bridge, che permette trasferire dei valori tramite delle variabili accessibili da entrambe le parti.

Lato arduino

Per capire come funziona, creare un esempio di blick sul pin 13, comandato tramite il bridge.
Utilizzare il seguente codice:

```
// importare libreria bridge.
#include <Bridge.h>
// importare libreria stdio per i metodi delle stringhe e caratteri
#include <stdio.h>

/**
 * Arduino blink on pin 13 sketch.
 *
 * @author giuliobosco (giuliobva@gmail.com)
 * @version 1.0 (2019-03-20 - 2019-03-20)
 */

// crea varibile in cui salvare i valori del bridge
char D13value[2];

void setup() {
  // setta la dimensione della varibile in cui salvare i valori del bridge
  memset(D13value, 0, 2);

  // setta il pin 13 come output
  pinMode(13, OUTPUT);

  // avvia il bridge
  Bridge.begin();
}

void loop() {
  // leggere il valore di D13 sul bridge e salvarlo nella variabile prima istanzianziata
  Bridge.get("D13", D13value, 2);
  // trasformare da int a stringa
  int D13int = atoi(D13value);
  // scrivere il valore della variabile sul pin 13
  digitalWrite(13, D13int);

  // ferma il programma per 10 ms per non sovraccaricare troppo il micro controllore
  delay(10);
}
```

Lato Lilino

Mentre dal lato di Lilino, bisogna creare un programma in python, che setti in maniera alternata il valore di D13 ad `1` e `0` ogni secondo, tramite il bridge.

In questo caso esegue questa operazione 100 volte, dopo di che stampa `I hope you enjoyed the light show`.

```
# importa sys per aggiungere la libreria bridge
import sys
# aggiungo libreria bridge
sys.path.insert(0, '/usr/lib/python2.7/bridge')

# importo la funzione sleep
from time import sleep
# importo libreria bridge
from bridgeclient import BridgeClient as bridgeclient

# inizializzo bridge
value = bridgeclient()

# per 100 volte
for idx in range(0, 100):
    # setta D13 a 1 (pin acceso)
    value.put('D13','1')
    # aspetta 1 secondo
    sleep(1)
    # setta D13 a 0 (pin spento)
    value.put('D13','0')
    # aspetta 1 secondo
    sleep(1)

# stampa messaggio
print("I hope you enjoyed the light show\n")
```