Getting Start with Arduino Yun

From Wiki for Dragino Project

Contents

Introduction

What is Dragino Yun Firmware

The Dragino Yun firmware is designed to provide the easy communication between Linux and Atmel AVR mcu. In Dragino Yun firmware, developer can remote upgrade the AVR sketch via Arduino IDE, call Linux command . access Linux system in MCU.

Dragino Yun Firmware is derived from the official Arduino Yun firmware with some bugs fixed and support more avrs. The firmware source follows the development on OpenWrt Yun (https://github.com/arduino/openwrt-yun). The firmware can be downloaded from downloaded from **This Link (http://www.dragino.com/downloads/index.php?dir=motherboards/ms14/Firmware/Yun/)**

This page provides necessary info of how to use Dragino Yun, how to design a plug-in module for Arduino Yun. For detail of how to use generic Arduino Yun software please consult to **Arduino Yun Offical Link** (http://arduino.cc/)

What devices can run Dragino Yun firmware

- Appliance: MS14S (http://www.dragino.com/products/mother-board/item/72-ms14-s.html), MS14-P (http://www.dragino.com/products/mother-board/item/71-ms14-p.html) ,
- Linux WiFi, IoT module: HE (http://www.dragino.com/products/linux-module/item/87-he.html)
- Any devices design with HE, such as Yun Shield (http://www.dragino.com/products/yunshield/item/86-yun-shield.html)

Requirement to get Arduino Yun alike feature

Yun firmware provide great feature such as program the AVR via Arduino IDE and easy communication between Linux and MCU. If developer want to design their customized hardware module for Dragino boards to use Yun feature, developer should make sure the correct hardware connection between dragino boards (MS14, HE , Yun Shield) and the AVR boards (Arduino UNO etc) . Features include:

- Remote upgrade sketch to AVR: Make sure the SPI connection is fine between AVR and Dragino boards.
- Communication between avr and linux:Make sure the UART connection is fine between AVR and Dragino boards.

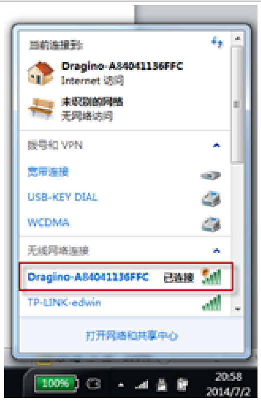
Reference hardware connection design of the SPI/ UART connection can check Yun Shield (https://github.com/dragino/modules/tree/master/hardware/YunShield) and Arduino UNO (http://arduino.cc/en/Main/ArduinoBoardUno)

Configure Dragino Yun

Find the ip addresses

The Dragino Yun has a WiFi interface and a LAN port. Either of them has IP address, can be used for internet connection and device management.

Factory IP of WiFi interface



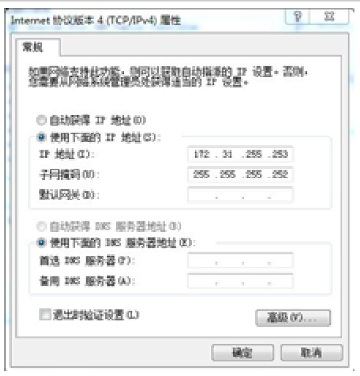
当前连接到:
Dragino-A84041136FFC
Internet 访问
未识别的网络
无网络访问
拨号和 VPN
宽带连接
USB-KEY DIAL
WCDMA
无线网络连接
Dragino-A84041136FFC 已连接
TP-LINK-edwin
打开网络和共享中心
100%
20:58
2014/7/2

Dragino Yun acts as an AP at the first boot

At the first boot of Dragino Yun, it will auto generate an unsecure WiFi network call Dragino2-xxxxxx

User can use their laptop to connect to this WiFi network. The laptop will get an IP 192.168.240.xxx and the Dragino Yun has the default IP 192.168.240.1

Fall Back IP



Internet 协议版本 4 (TCP/IPv4) 属性

常规

如果网络支持此功能，则可以获取自动指派的 IP 设置。否则，您需要从网络系统管理员处获得适当的 IP 设置。

自动获得 IP 地址(O) [✓ 使用下面的 IP 地址(S):]

IP 地址(I): 172 . 31 . 255 . 253
子网掩码(U): 255 . 255 . 255 . 252
默认网关(D): . . .

自动获得 DNS 服务器地址(B) [✓ 使用下面的 DNS 服务器地址(E):]

首选 DNS 服务器(P): . . .
备用 DNS 服务器(A): . . .

☐ 退出时验证设置(L)

高级(V)... 确定 取消

PC setting to use FallBack IP

A fall back IP 172.31.255.254/255.255.255.252 is assigned to Dragino Yun's LAN port so user can always access Dragino Yun with this ip if their laptop has: IP IP: 172.31.255.253 and Netmask:255.255.255.252.

Detect IP from Arduino IDE

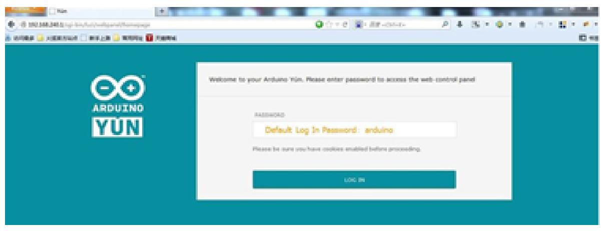
If Dragino Yun's Ethernet port is connected to the uplink router or the WiFi interface is associated to the WiFi router. The PC in the same network can use Arduino IDE to detect Dragino's IP address as described in Detected Dragino Yun.

Configure Method

The Dragino Yun runs Open Source Linux system. If user has a PC at the same network as Dragino Yun, user can access its system via either Web Interface or Secure Shell (SSH).

Access via web interface

The recommended browsers to configure Dragino Yun are Firefox and Chrome. Simply type the IP address into your browser and you will see the log in page of Dragino Yun.


Welcome to your Arduino Yún. Please enter password to access the web control panel

PASSWORD

Default Log In Password : arduino

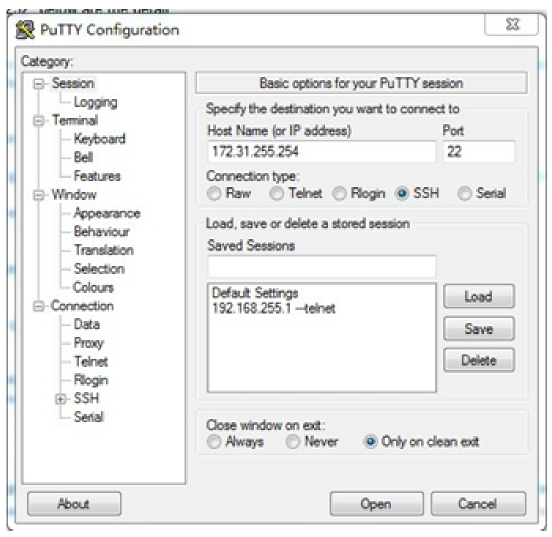
Please be sure you have cookies enabled before proceeding.

Dragino Yun Log In Page

Default User name and Password for Dragino Yun is root/dragino.

Access via SSH

Via SSH access, user can access to the Linux system directly and customized the system to support more features and applications.


PuTTY Configuration

Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address): 172.31.255.254

Port: 22

Connection type: Raw Telnet Rlogin [✓ SSH] Serial

Load, save or delete a stored session

Saved Sessions

Default Settings
192.168.255.1 --telnet

Close window on exit: Always Never [✓ Only on clean exit]

Access Dragino Yun via SSH

SSH Access:

IP address: IP Address of Yun Shield

Port: 22

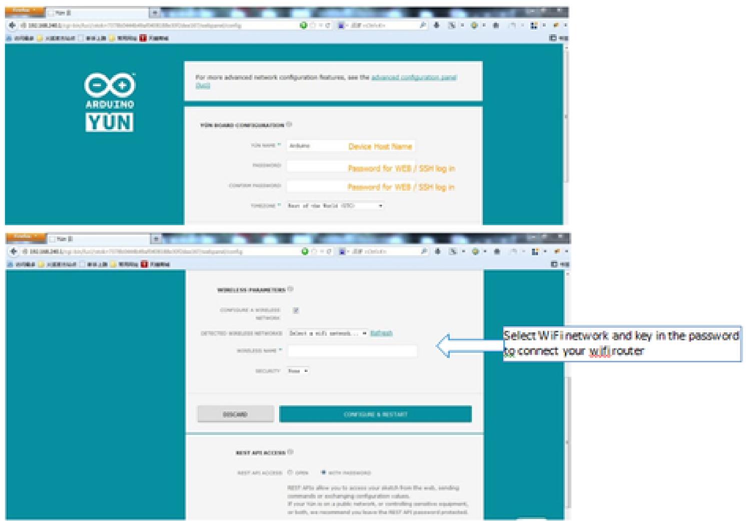
User Name: root

Password: dragino (default)

Web Configure Pages

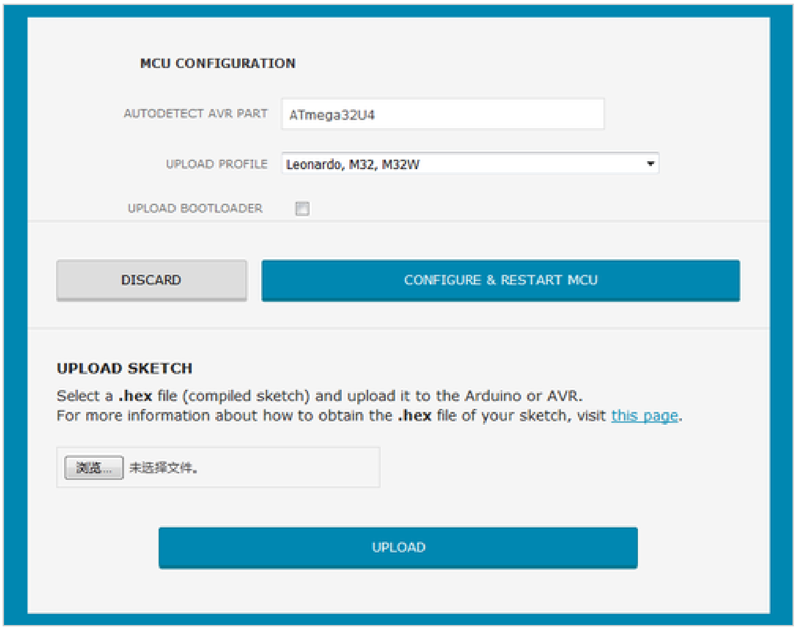
General Set Up for Internet Access

After log in, the GUI will show the WIFI / ETH interface status. Click the Configure button and now user can configure the device password and network parameters.


ARDUINO YÚN
For more advanced network configuration features, see the advanced configuration panel (luci)
YÚN BOARD CONFIGURATION
YÚN NAME * Arduino
Device Host Name
PASSWORD
Password for WEB / SSH log in
CONFIRM PASSWORD
Password for WEB / SSH log in
TIMEZONE *
WIRELESS PARAMETERS
CONFIGURE A WIRELESS NETWORK
DETECTED WIRELESS NETWORKS Select a wifi network... Refresh
WIRELESS NAME *
SECURITY None
DISCARD
CONFIGURE & RESTART
REST API ACCESS
REST API ACCESS OPEN WITH PASSWORD
Select WiFi network and key in the password to connect your wifi router

Configure WiFi for internet Access

Arduino related set up


MCU CONFIGURATION

AUTODETECT AVR PART ATmega32U4

UPLOAD PROFILE Leonardo, M32, M32W

UPLOAD BOOTLOADER ☐

UPLOAD SKETCH

Select a **.hex** file (compiled sketch) and upload it to the Arduino or AVR.
For more information about how to obtain the **.hex** file of your sketch, visit this page.

Select the correct boardtype

Arduino Board Type: Define the bootloader/mcu type/ fuse setting during Sketch upload.

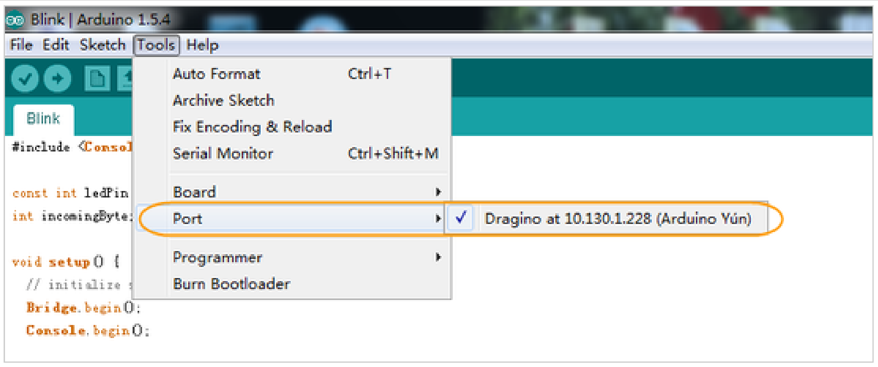
Operation Mode: make sure it is in Arduino Bridge mode so the Bridge class for Arduino Yun works.

Upgrade Firmware

Detail please see Upgrade firmware via web UI

Detect Dragino Yun

Once the device running Dragino Yun firmware is configured and get IP address from WiFi router. It will send broadcast data in the network and if you run the latest Arduino IDE. It will auto detect Yun service and show in the *port menu* .



```
#include <Consol
const int ledPin
int incomingByte;
void setup() {
  // initialize s
  Bridge.begin();
  Console.begin();
```

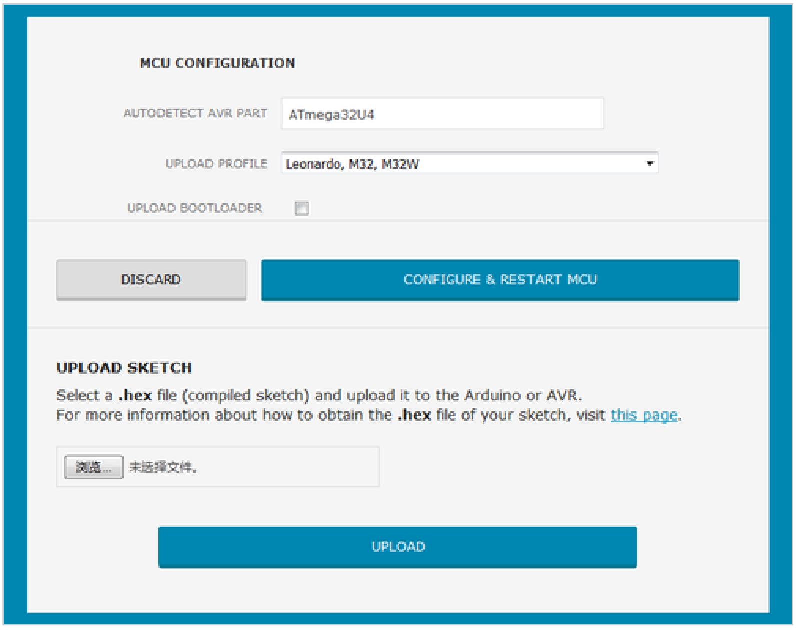
Arduino Detect Yun in the same network

Upload Sketch to Arduino / AVR

In Dragino Yun software, user can upgrade the Arduino / AVR remotely in Arduino IDE. The Dragino Yun GUI supports different kind of Arduino/AVR boards includes:

- Arduino Leonardo or Mega32u4
- Arduino Uno or Mega328p
- Arduino Duemilanove/Diecimila or Mega168
- Arduino Mega2560 or Mega2560

User can check them in the GUI--> Sensor page. The Dragino Yun use avrdude to program so it can program more AVRs via SSH access.


MCU CONFIGURATION

AUTODETECT AVR PART: ATmega32U4

UPLOAD PROFILE: Leonardo, M32, M32W

UPLOAD BOOTLOADER: ☐

UPLOAD SKETCH

Select a **.hex** file (compiled sketch) and upload it to the Arduino or AVR.
For more information about how to obtain the **.hex** file of your sketch, visit this page.

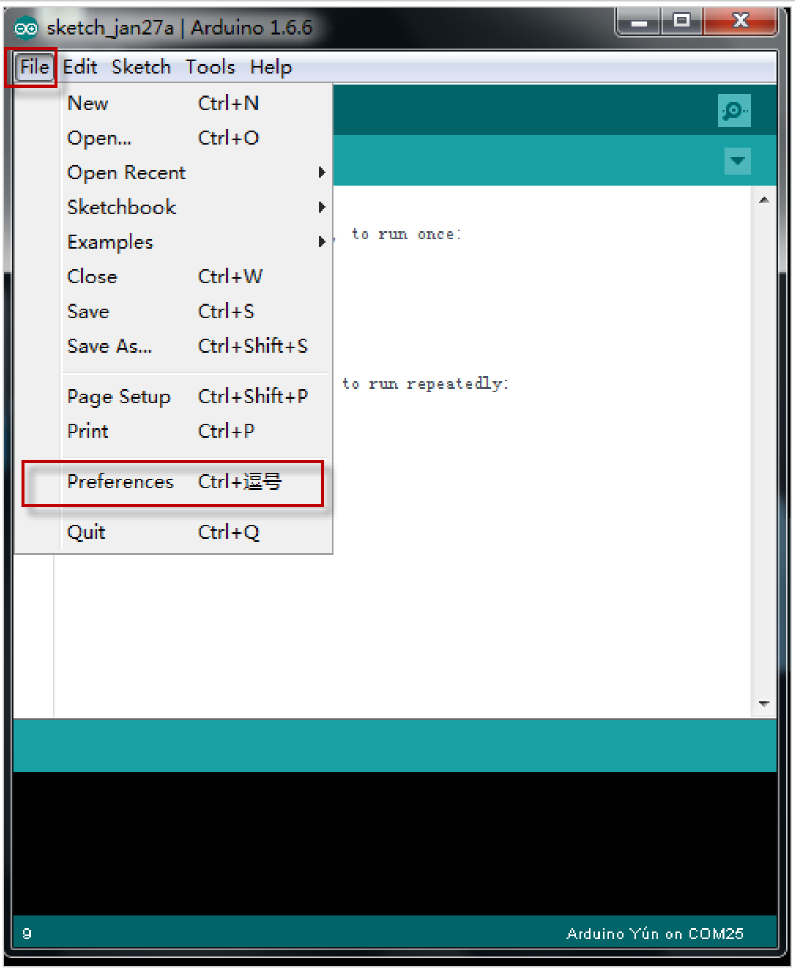
Select the correct boardtype

Arduino Board Type: Define the bootloarder/mcu type/ fuse setting during Sketch upload.

Automatically Add Board Profile in Arduino IDE

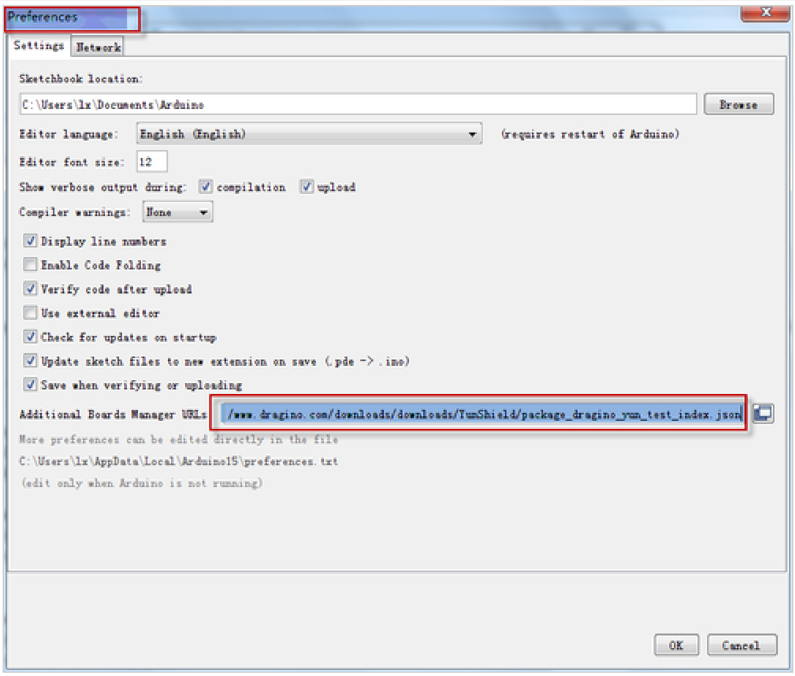
To use Arduino IDE to upgrade the Arduino / AVR via local area network. User should configure the correct board type.

To do this, user can follow the operations as below(Arduino IDE version after 1.6.4): Open the IDE and click the File option to choose Preferences.



Choose the Preferences

Type the Additional Boards Manager URLs: http://www.dragino.com/downloads/downloads/YunShield/package_dragino_yun_test_index.json Save settings.


Preferences

Settings | Network

Sketchbook location:
C:\Users\lx\Documents\Arduino [Browse]

Editor language: English (English) (requires restart of Arduino)

Editor font size: 12

Show verbose output during: ✓ compilation ✓ upload

Compiler warnings: None

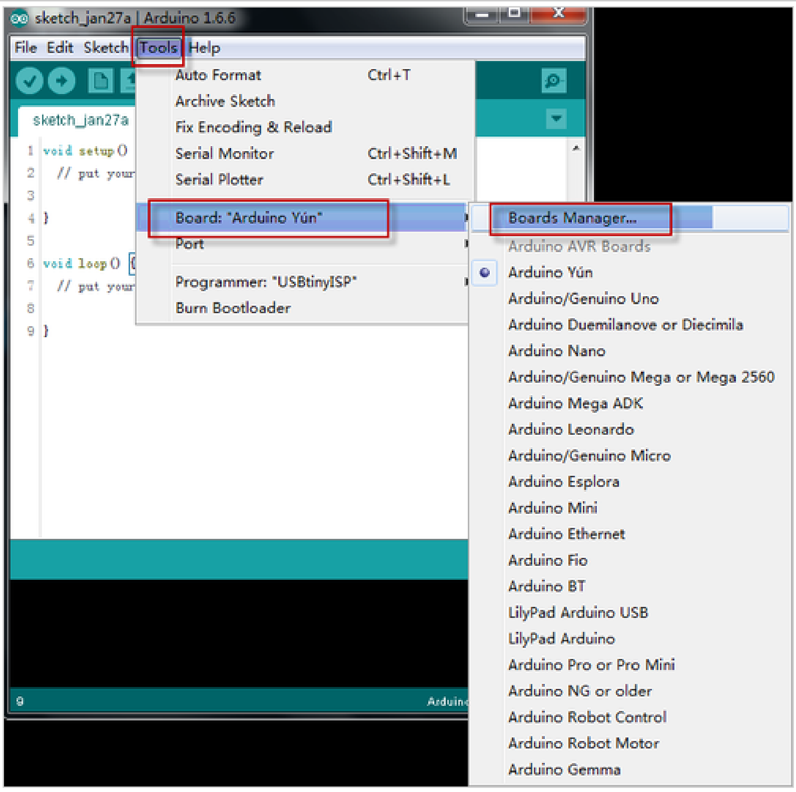
✓ Display line numbers
☐ Enable Code Folding
✓ Verify code after upload
☐ Use external editor
✓ Check for updates on startup
✓ Update sketch files to new extension on save (.pde -> .ino)
✓ Save when verifying or uploading

Additional Boards Manager URLs: /www.dragino.com/downloads/downloads/YunShield/package_dragino_yun_test_index.json

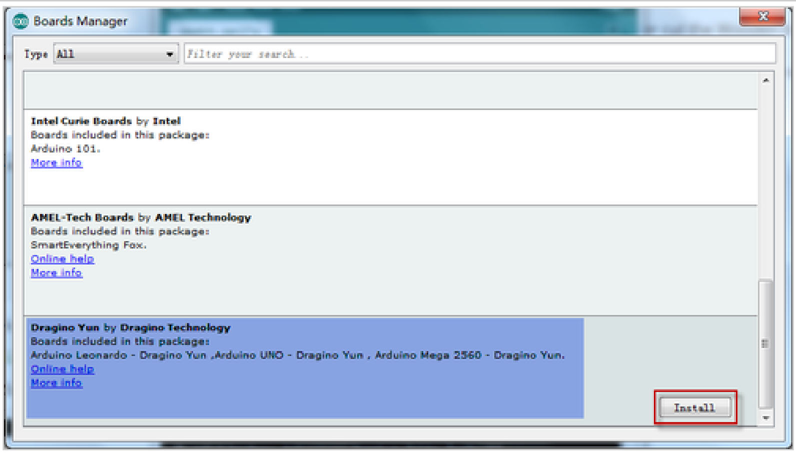
More preferences can be edited directly in the file
C:\Users\lx\AppData\Local\Arduino15\preferences.txt
(edit only when Arduino is not running)

[OK] [Cancel]

Type URL



Enter the Boards Manager



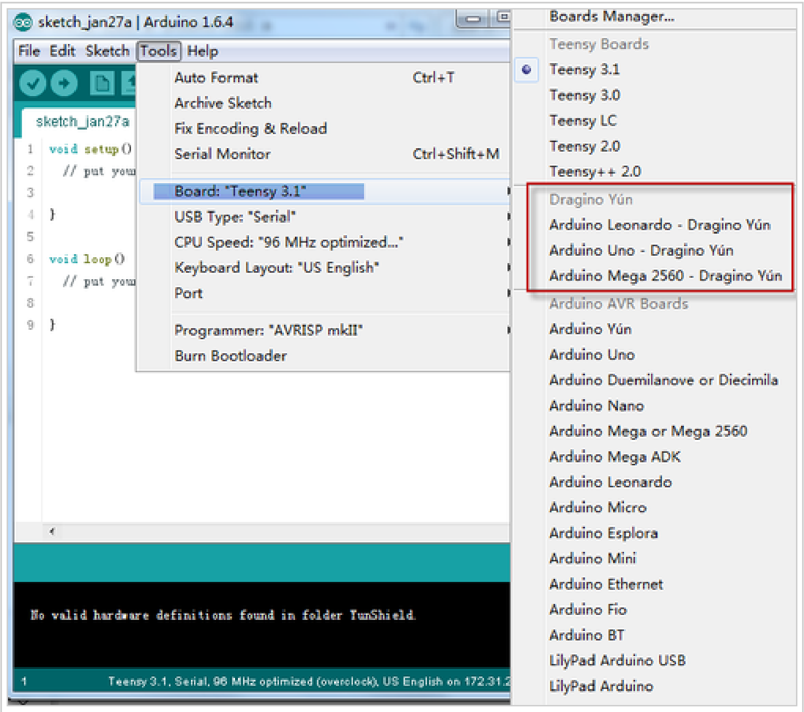
Intel Curie Boards by **Intel**
Boards included in this package:
Arduino 101.
More info

AMEL-Tech Boards by **AMEL Technology**
Boards included in this package:
SmartEverything Fox.
Online help
More info

Dragino Yun by **Dragino Technology**
Boards included in this package:
Arduino Leonardo - Dragino Yun ,Arduino UNO - Dragino Yun , Arduino Mega 2560 - Dragino Yun.
Online help
More info

Install the selected item

Then restart the Arduino IDE, the new hardware info should be found as below:

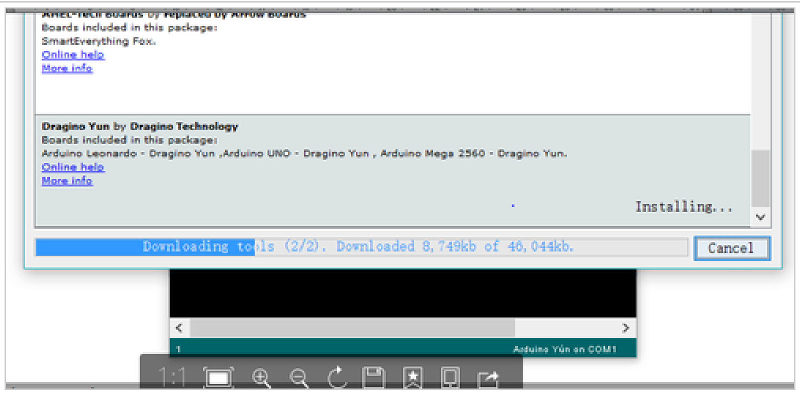


```
No valid hardware definitions found in folder YunShield.
```

Yun Shield Hardware

Manually Add Board Profile in Arduino IDE

If IDE quite slowly while downloading the Dragino profile in board manager and stuck somewhere. As show below, it is because your network has slow connection to some packages from Arduino IDE.


Dragino Yun by Dragino Technology
Boards included in this package:
Arduino Leonardo - Dragino Yun ,Arduino UNO - Dragino Yun , Arduino Mega 2560 - Dragino Yun.
Online help
More info

Installing...

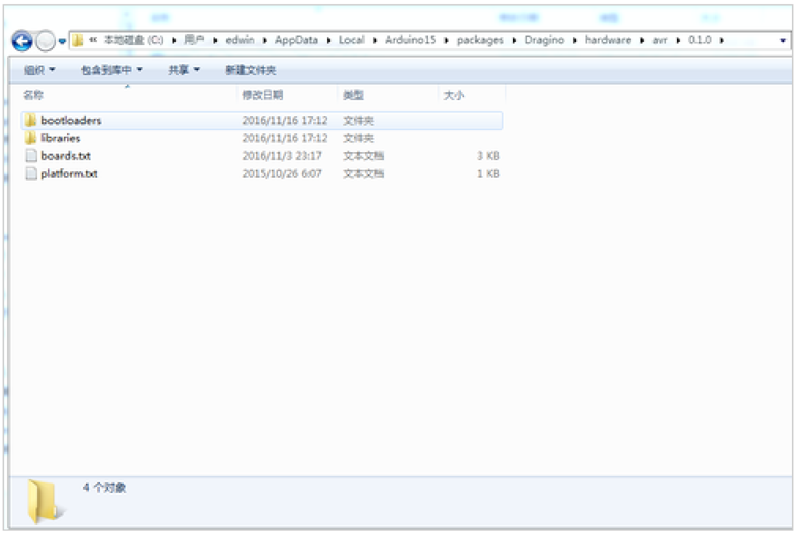
Downloading tools (2/2). Downloaded 8,749kb of 46,044kb.

Arduino IDE download slow

To solve this, user can manually add Dragino profile. Below is the step:

1. Download the profile from https://github.com/dragino/Arduino-Profile-Examples
2. Unzip it and put the content under this directory: C:\Users\xxx\AppData\Local\Arduino15\packages\Dragino\hardware\avr\0.1.0

Note: Different system may have different directory for Arduino15, if you can’t see Dragino\hardware\avr\0.1.0, just create it in your Arduino15 directory. The final directory content should as below.



« 本地磁盘 (C:) ▸ 用户 ▸ edwin ▸ AppData ▸ Local ▸ Arduino15 ▸ packages ▸ Dragino ▸ hardware ▸ avr ▸ 0.1.0 ▸

名称	修改日期	类型	大小
bootloaders	2016/11/16 17:12	文件夹	
libraries	2016/11/16 17:12	文件夹	
boards.txt	2016/11/3 23:17	文本文档	3 KB
platform.txt	2015/10/26 6:07	文本文档	1 KB

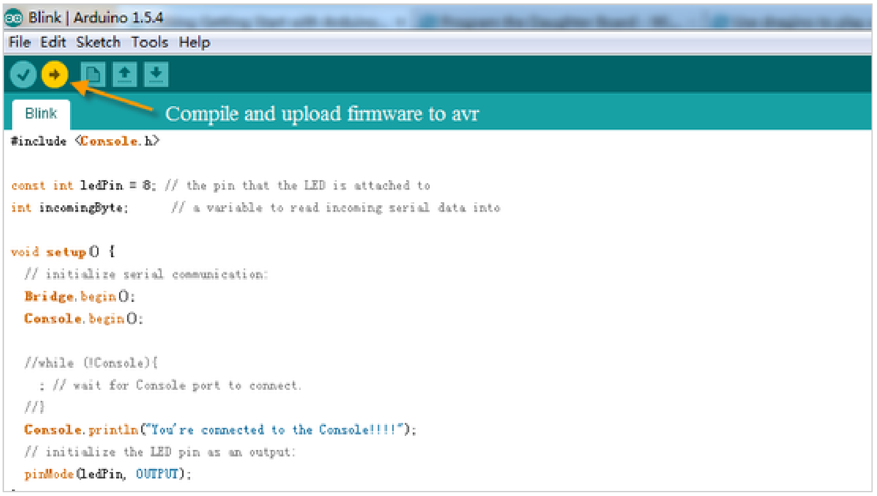
4 个对象

Add Dragino Profile

Compile and Upload via Arduino IDE

After we have correctly configure board info. We can use the Arduino IDE to compile and upload the firmware to Arduino or AVR mcu.

During upload, Arduino IDE will ask for a password. This is the password we configure in the Web GUI. By default, it is dragino.



Blink | Arduino 1.5.4
File Edit Sketch Tools Help
Blink
Compile and upload firmware to avr

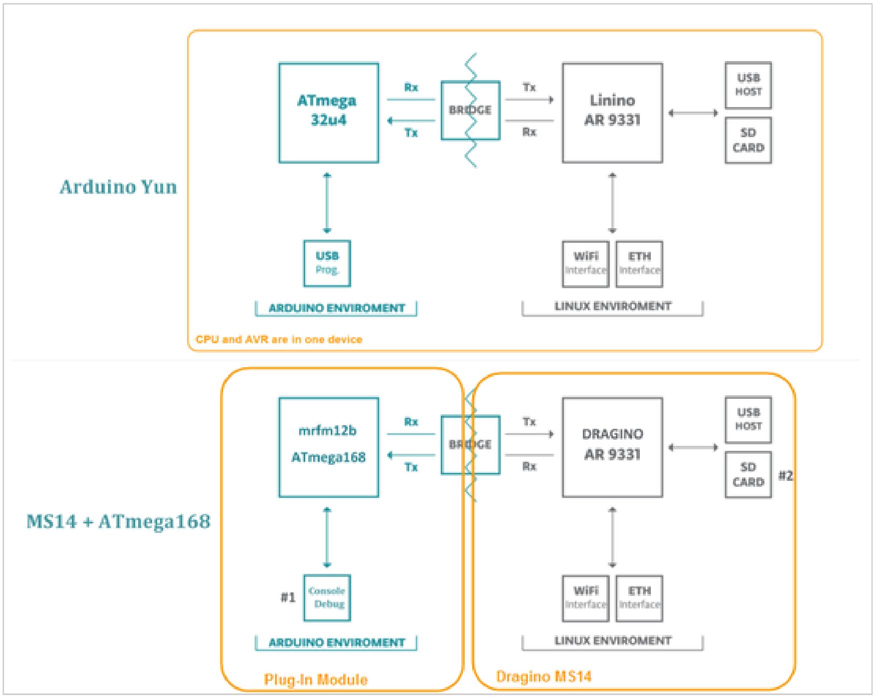
```
#include <Console.h>

const int ledPin = 8; // the pin that the LED is attached to
int incomingByte;      // a variable to read incoming serial data into

void setup() {
  // initialize serial communication:
  Bridge.begin();
  Console.begin();

  //while (!Console){
    ; // wait for Console port to connect.
  //}
  Console.println("You're connected to the Console!!!!");
  // initialize the LED pin as an output:
  pinMode(ledPin, OUTPUT);
```

Compile and Upload sketch in Arduino IDE

Arduino Bridge Library

The bridge library is the most important part of Arduino Yun. Bridge library defines a mechanisms how the AVR talk to the CPU (ar9331). With the bridge library, the avr can send data to CPU, get commands from CPU or call commands in CPU.

The bridge Library use UART port to communicate between avr and ar9331. Below is the block diagram which shows the difference between ms14 and the official Arduino Yun device. With Dragino ms14, we can use different types of avr , even use other brand mcu (PIC) with the Bridge Library.



Bridge Diagram Difference between Arduino Yun and Dragino MS14

Detail instruction of how to use the bridge library can be found in Arduino Official Website (http://www.arduino.cc). There are several points we should consider while writing the avr software:

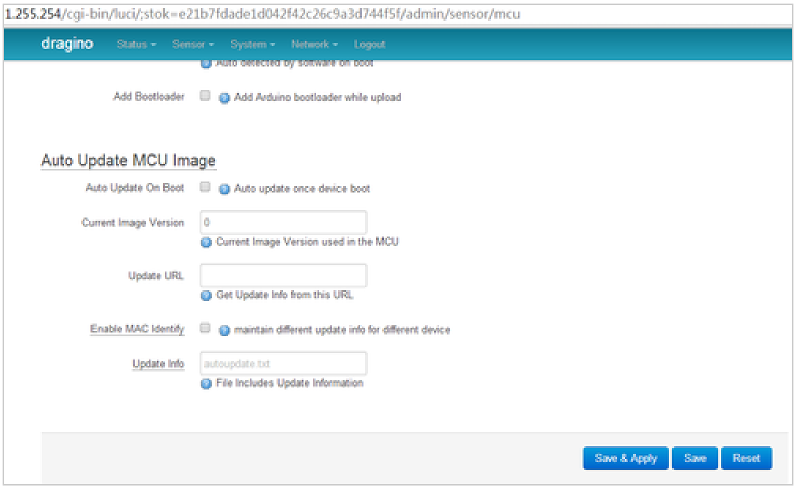
- In the default bridge examples from Arudino IDE, it uses **Serial** class to print debug info. This is ok if you connect ar9331 to ATmega32u4 because it will call USB Virtule COM Port to print the debug info, but it will have problem if you use it mcu such as ATmega168,328 because they don't have Virtual USB Com port. The serial will call mega168, 328's hardware serial port , it will be conflict with the Bridge Library.
- Arduino use ledPin 13 to show LED, user can also control SYS led on Dragino ms14, but the pin may be different depends on how user connect the avr to the ms14 sys led.

The examples sketch codes for ms14 can be found at **This Link (https://github.com/dragino/modules/tree/master/examples/Bridge)**.

Auto Update Sketch

Since Firmware 2.0.4, auto update sketch is supported. With this feature, the Dragino's will connect to a http server and get the latest sketch version and upload the sketch with this version. The purpose of this feature is to reduce the tech support cost / time for remote installation.

The feature can be configured in the page system -> advanced configuration panel -> sensors -> micro controller.


1.255.254/cgi-bin/luci/;stok=e21b7fdade1d042f42c26c9a3d744f5f/admin/sensor/mcu
dragino
Status
Sensor
System
Network
Logout
Save & Apply
Save
Reset

Auto detected by software on boot

Add Bootloader ☐ Add Arduino bootloader while upload

Auto Update MCU Image

Auto Update On Boot ☐ Auto update once device boot

Current Image Version 0
Current Image Version used in the MCU

Update URL
Get Update Info from this URL

Enable MAC Identify ☐ maintain different update info for different device

Update Info autoupdate.txt
File Includes Update Information

Auto Update MCU Settings

- Auto Update On Boot: While this option is enabled. Device will connect to the auto server on every boot and check if there is new version of Sketch to be update. If Device find newer version on the auto update server, device will download it from the server and and

update the mcu with this new version.
- Current Image Version: Shows the current sketch version. By default it is 0. Device will update this version to the latest version number only after auto update successful.
- Update URL: This URL contact the update infomation and the sketch.hex file. device will connect to this URL to check if there is newer version in the server.
- Update Info: The text file include the update information. an example for this file can be found here: example for update information file (http://www.dragino.com/downloads/downloads/tmp/autoupdate/update_info). It should includes:
  - **image**: the sketch used for auto update
  - **md5sum**: md5sum for this sketch.
  - **version**: the lastest version number.
- Enable MAC Identify: Instead of geting update information as specified in Update Info, The device will looking for the update information from the file: wifi_mac.txt. which means, if the device has mac address A840417867AF, device will download the file: $Update_URL/A840417867AF.txt for auto update information.

Procedure for Auto Update Sketch:

Assume we have below configured:

```
Auto Update On Boot: checked
Update URL: http://www.dragino.com/downloads/downloads/tmp/autoupdate/
Update Info: update_info
Enable MAC Identify: unchecked
```

Then after reboot, the device will do auto update as below:

1. Download the update information from http://www.dragino.com/downloads/downloads/tmp/autoupdate/update_info
2. Compare the Latest version and the version on the device
3. If server has a higer version, Device will download http://www.dragino.com/downloads/downloads/tmp/autoupdate/sketch.hex
4. Do a md5sum check to verify the downloaded sketch is fine
5. Update the MCU with the newer version sketch
6. Update the version number to the latest version number

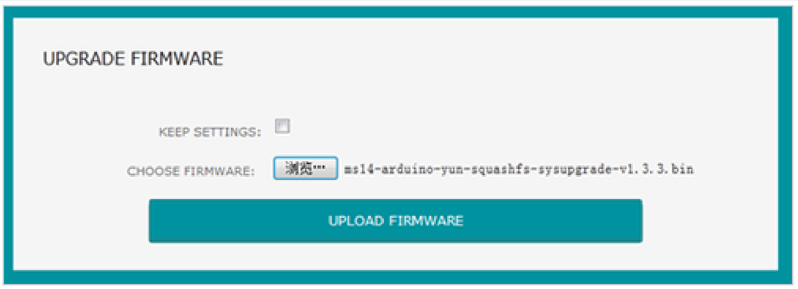
Upgrade Firmware to devices

The Yun firmware from Arduino official website won't work at ms14 because of different pin mapping. We have maitained a Dragino version Yun firmware for MS14. They can be downloaded from **firmware download link** (http://www.dragino.com/downloads/index.php?dir=motherboards/ms14/Firmware/Yun/). Firmware release note is **HERE**.

If you already have a Yun firmware running in the dragino devices. Then you can upgrade the firmware from WEB GUI. Otherwise, you will have to upgrade the firmware according to below instruction.

Upgrade via WEB GUI

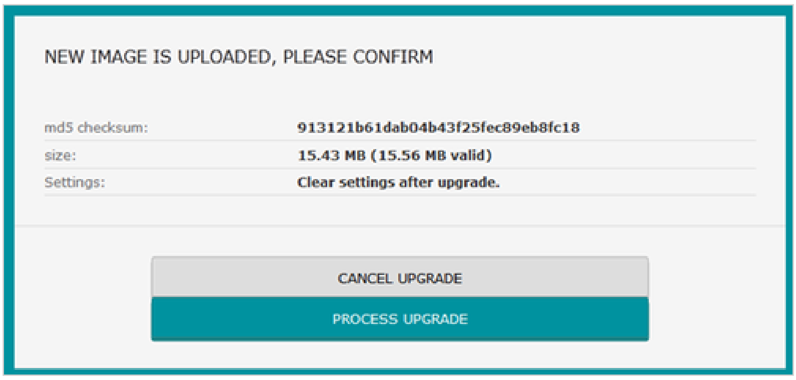
Go to GUI → Upgrade page and select the correct firmware to upgrade. The firmware used for web upgrade should be a xxx.squashfs-sysupgrade.bin type firmware, user can choose if keep settings or not after upgrade.


UPGRADE FIRMWARE

KEEP SETTINGS: ☐

CHOOSE FIRMWARE: 浏览... ms14-arduino-yun-squashfs-sysupgrade-v1.3.3.bin

Upload Firmware



NEW IMAGE IS UPLOADED, PLEASE CONFIRM

md5 checksum: 913121b61dab04b43f25fec89eb8fc18
size: 15.43 MB (15.56 MB valid)
Settings: Clear settings after upgrade.

Upload Firmware

Normally it will take about 2 minutes to flash the new firmware. Then all the LEDS will blink together which indicates that the system reboot with the new firmware.

Upgrade via U-Boot

If the ms14 doesn't have a Yun firmware, you need to upgrade a Yun firmware into it first.

Set up your tftp server with ip 192.168.255.2 and put the firmware on tftp server root.

In u-boot, both kernel and rootfs need to be upgraded. The commands is as below, replace the xxx with the exactly version you have on the firmware.

Upgrade Commands in U-Boot

Upgrade Kernel

```
dr_boot>tftpboot 0x81000000 ms14-arduino-yun-kernel-xxx.bin
dr_boot>erase 0x9fea0000 +0x140000
dr_boot>cp.b 0x81000000 0x9fea0000 $filesize
```

Upgrade rootfs

```
dr_boot>tftpboot 0x81000000 ms14-arduino-yun--rootfs-squashfs-xxx.bin
dr_boot>erase 0x9f050000 +0xe50000
dr_boot>cp.b 0x81000000 0x9f050000 $filesize
```

Change boot address for Yun firmware

```
dr_boot> setenv bootcmd bootm 0x9fea0000
dr_boot> saveenv
dr_boot> reset
```

Then you will have Yun firmware after reboot.

Upgrade to the IoT Mesh firmware

The Yun firmware and IoT mesh firmware has different flash layout and boot address. Switch between these firmware can't be complete via Web UI.

Below is the instruction of how to upgrade the device from Dragino Yun firmware to IoT Mesh Firmware:

1. Upgrade the device to the **middleware(download (http://www.dragino.com/downloads/downloads/motherboards/ms14/Firmware/Yun/middleware_to_switch_to_IoT_firmware/))** via the Web UI (don't check keep settings).
2. Access to the linux system and run: switch_firmware_from_Yun_to_IoT . Device will upgrade and reboot to Version: Dragino-v2 IoT-1.3.4. After that, user can continue to use other IoT Mesh Firmware by upgrade in Web UI.

output of this command is as below:

```
root@dragino:~# switch_firmware_from_Yun_to_IoT
Start Flashing, Don't power off the device until it finish !!!
Unlocking u-boot-env ...
```

```
Writing from /var/IoT_firmware_part1 to u-boot-env ...
Unlocking rootfs ...

Writing from /var/IoT_firmware_part2 to rootfs ...
Flashing finish. Reboot the device to boot to IoT firmware.
```

Examples

CLICK HERE for some detail examples.

Or consult **Arduino Yun Example Git Source** (https://github.com/dragino/modules/tree/master/examples/Bridge) for more examples source.

Trouble Shooting

Bridge examples doesn't work

The bridge examples from Arduino IDE are tested base on Arduino Yun (Ar9331 + Mega32u4 mcu). Mega32u4 has a hardware serial (Serial1 Class) which use for Bridge library, and has a USB port use as serial (Serial Class) to print out debug info. While in UNO (Mega328P) and Mega2560, they don't have the CDC USB port, the USB port in them use the hardware TXD and RXD (Serial Class),which also use for Bridge.

With reference to the Bridge Class in Arduino, it will use Serial1 for Mega32u4 and Serial for Mega328P/Mega2560.

So in the bridge examples such as bridge --> httpclient, it use both bridge (serial1) and serial (usb serial). It is ok for Mega32u4, but if user use this sketch on MCU such as mega328p, it won't work, because the bridge and serial are now use the same TXD/RXD. If user want to see debug info, they can use the Console Class instead of Serial. This will print the debug info via network to Arduino IDE.

Update Sketch Fail

Arduino IDE shows signature mismatch

an example output from Arduino IDE as below:

```
Programmer Type : linuxgpio
Description : Use the Linux sysfs interface to bitbang GPIO lines

avrdude: Calibrating delay loop... calibrated to 49 cycles per us
avrdude: AVR device initialized and ready to accept instructions

Reading | ################################################## | 100% 0.00s

avrdude: Device signature = 0x1e9801
avrdude: Expected signature for ATmega32U4 is 1E 95 87
Double check chip, or use -F to override this check.
```

Above result shows that Dragino try to program the Arduino with mega32u4 mcu type but actaully the Arduino is mega2560 which has signature 0x1e9801. So it fails. To solve this issue, user can choose the mcu/board type to the correct one in the **Dragino Web Interface --> Sensor --> Arduino Board** Type.

AVR device not responding

Typical output in Arduino IDE for this issue is as below:

```
        Programmer Type : linuxgpio
        Description     : Use the Linux sysfs interface to bitbang GPIO lines

avrdude: Calibrating delay loop... calibrated to 48 cycles per us
avrdude: AVR device not responding
avrdude: initialization failed, rc=-1
        Double check connections and try again, or use -F to override
        this check.
```

```

avrdude done.  Thank you.
```

This means that there is something wrong on the SPI interface between Dragino devices and AVR(Arduino Board). So please check if the SPI connection between dragino and avr is ok.

See wrong strings in serial during booting

The Arduino Bridge Library use Baud Rate 250000 for UART communication. So in Arduino Yun Firmware, the kernel will switch the baud rate to 250000 during the booting process. Because u-boot used runs Baud Rate 115200, if use serial monitor to see the boot up process in 115200 baud rate mode, you can see messy code in UART after kernel chande the baud rate.

A typical output as below:

```
U-Boot 1.1.4 (Aug 12 2012 - 20:14:51)

AP121-2MB (ar9330) U-boot
DRAM: #### TAP VALUE 1 = f, 2 = 10
64 MB
Top of RAM usable for U-Boot at: 84000000
Reserving 212k for U-Boot at: 83fc8000
Reserving 192k for malloc() at: 83f98000
Reserving 44 Bytes for Board Info at: 83f97fd4
Reserving 36 Bytes for Global Data at: 83f97fb0
Reserving 128k for boot params() at: 83f77fb0
Stack Pointer at: 83f77f98
Now running in RAM - U-Boot at: 83fc8000
id read 0x100000ff
flash size 16777216, sector count = 256
Flash: 16 MB
In: serial
Out: serial
Err: serial
Net: ag7240_enet_initialize...
No valid address in Flash. Using fixed address
No valid address in Flash. Using fixed address
: cfg1 0x5 cfg2 0x7114
eth0: 00:03:7f:09:0b:ad
eth0 up
: cfg1 0xf cfg2 0x7214
eth1: 00:03:7f:09:0b:ad
athrs26_reg_init_lan
ATHRS26: resetting s26
ATHRS26: s26 reset done
eth1 up
eth0, eth1
Hit any key to stop autoboot: 4     3     2     1     0
## Booting image at 9fea0000 ...
Image Name: MIPS OpenWrt Linux-3.8.3
Created: 2014-02-12 3:16:34 UTC
Image Type: MIPS Linux Kernel Image (lzma compressed)
Data Size: 1172210 Bytes = 1.1 MB
Load Address: 80060000
Entry Point: 80060000
Verifying Checksum at 0x9fea0040 ...OK
Uncompressing Kernel Image ... OK
No initrd
## Transferring control to Linux (at address 80060000) ...
## Giving linux memsize in bytes, 67108864

Starting kernel ...

[ 0.000000] Linux version 3.8.3 (edwin@edwin-test) (gcc version 4.6.4 20121210 (prerelease) (Linaro GCC 4.6-2012.12) ) #17 Wed Feb 12 11:15:36 CST 2014
[ 0.000000] bootconsole [early0] enabled
[ 0.000000] CPU revision is: 00019374 (MIPS 24Kc)
[ 0.000000] SoC: Atheros AR9330 rev 1
[ 0.000000] Clocks: CPU:400.000MHz, DDR:400.000MHz, AHB:200.000MHz, Ref:25.000MHz
[ 0.000000] Determined physical RAM map:
[ 0.000000] memory: 04000000 @ 00000000 (usable)
[ 0.000000] Initrd not found or empty - disabling initrd
[ 0.000000] Zone ranges:
[ 0.000000] Normal [mem 0x00000000-0x03ffffff]
[ 0.000000] Movable zone start for each node
[ 0.000000] Early memory node ranges
[ 0.000000] node 0: [mem 0x00000000-0x03ffffff]
[ 0.000000] Primary instruction cache 64kB, VIPT, 4-way, linesize 32 bytes.
[ 0.000000] Primary data cache 32kB, 4-way, VIPT, cache aliases, linesize 32 bytes
[ 0.000000] Built 1 zonelists in Zone order, mobility grouping on. Total pages: 16256
[ 0.000000] Kernel command line: board=Linino console=ttyATH0,250000 mtdparts=spi0.0:256k(u-boot)ro,64k(u-boot-env)ro,14656k(rootfs),1280k(kernel),64k(nvra
[ 0.000000] PID hash table entries: 256 (order: -2, 1024 bytes)
[ 0.000000] Dentry cache hash table entries: 8192 (order: 3, 32768 bytes)
[ 0.000000] Inode-cache hash table entries: 4096 (order: 2, 16384 bytes)
[ 0.000000] __ex_table already sorted, skipping sort
[ 0.000000] Writing ErrCtl register=00000000
[ 0.000000] Readback ErrCtl register=00000000
[ 0.000000] Memory: 60900k/65536k available (2569k kernel code, 4636k reserved, 651k data, 232k init, 0k highmem)
[ 0.000000] SLUB: Genslabs=9, HWalign=32, Order=0-3, MinObjects=0, CPUs=1, Nodes=1
[ 0.000000] NR_IRQS:51
[ 0.000000] Calibrating delay loop... 265.42 BogoMIPS (lpj=1327104)
[ 0.080000] pid_max: default: 32768 minimum: 301
[ 0.080000] Mount-cache hash table entries: 512
[ 0.090000] NET: Registered protocol family 16
[ 0.090000] MIPS: machine is Arduino Yun
[ 0.540000] Setting DogStick2 GPIO
[ 0.560000] bio: create slab at 0
[ 0.560000] SCSI subsystem initialized
[ 0.570000] usbcore: registered new interface driver usbfs
[ 0.570000] usbcore: registered new interface driver hub
[ 0.570000] usbcore: registered new device driver usb
[ 0.580000] Switching to clocksource MIPS
[ 0.580000] NET: Registered protocol family 2
[ 0.590000] TCP established hash table entries: 512 (order: 0, 4096 bytes)
[ 0.590000] TCP bind hash table entries: 512 (order: -1, 2048 bytes)
[ 0.590000] TCP: Hash tables configured (established 512 bind 512)
[ 0.600000] TCP: reno registered
[ 0.600000] UDP hash table entries: 256 (order: 0, 4096 bytes)
[ 0.610000] UDP-Lite hash table entries: 256 (order: 0, 4096 bytes)
[ 0.620000] NET: Registered protocol family 1
[ 0.640000] squashfs: version 4.0 (2009/01/31) Phillip Lougher
[ 0.640000] jffs2: version 2.2 (NAND) (SUMMARY) (LZMA) (RTIME) (CMODE_PRIORITY) (c) 2001-2006 Red Hat, Inc.
[ 0.650000] msgmni has been set to 118
[ 0.660000] Block layer SCSI generic (bsg) driver version 0.4 loaded (major 253)
[ 0.660000] io scheduler noop registered
[ 0.670000] io scheduler deadline registered (default)
[ 0.670000] Serial: 8250/16550 driver, 16 ports, IRQ sharing enabled
[ 0.680000] ar933x-uart: ttyATH0 at MMIO 0x18020000 (irq = 11) is
-q
侤塈%堃| 8c ?M? 鹍\贛噘]? H貭劝my o7^:H筣
J泑艉h.靥O j ?雀z截辿,雝.□NA衟 .岑摶&
霳 j  @L圝
&瘖¬?(Q蠳 涥AJ侤塝圝ba攧&垧釘J葝僼瀵    j ?
}+~涀(渓> }7?
 沺溟j ' j
??偽 )N6?Q鍏 j ?垟JH欽孬坹圝圚& 蘋 j ?垟J圝圛擩H沟圚嚮L 隸 j ?,L8  屇>a~a怇N j ? @L圝
&瘖¬?(' 5i?蔗a MH i?駐Hq榅?' ja?蔗醥?A塏6堷J巠哥kL? ? j%?I塏>娖Io%k} h (菻¬?鵗?
¬莉℉z黓埜 柭O j%? 鴝a?H j?a E'bMy瑦 O j=?J竘b棖 跱 篘x楉|良籦a垨L ` j=?K\&cn?X &闥t搸H塝 壂?J岦綹湅餗撉Mj湉z9 H貏?
8澵j鱇&
l  k=?f? 擬~訫?j8 H8圝 q X&¬m6??-x H 圝'坹怞? P¬磨狪告厅O k-?B?d達?h?tX67? ¬8OH 舭%皃O j-?偤l 歒傫歖¬烗H癿滅晦jb躭懈~
```

Recover Dragino Yun firmware

User may lost access to the Dragino Yun by accidently change the Linux configure files to in correct setting, lost password or other caused. There are different methods to recover.

Reset to factory default

Dragino Yun has a toggle button (GPIO11) which can be used for reset. When the Dragino Yun is running in OpenWrt System, user can press the toggle button to reset the device. When press this button, the WLAN LED will blink.

- If pressing the toggle button and release after 5 seconds, it will reset the WiFi setting and other settings will be kept.
- If pressing the toggle button and release after 30 seconds, it will reset ALL the setting to factory default .

Recover via RJ45 port in Failsafe u-boot

This mehod LAN connection direclty between Dragino and PC.

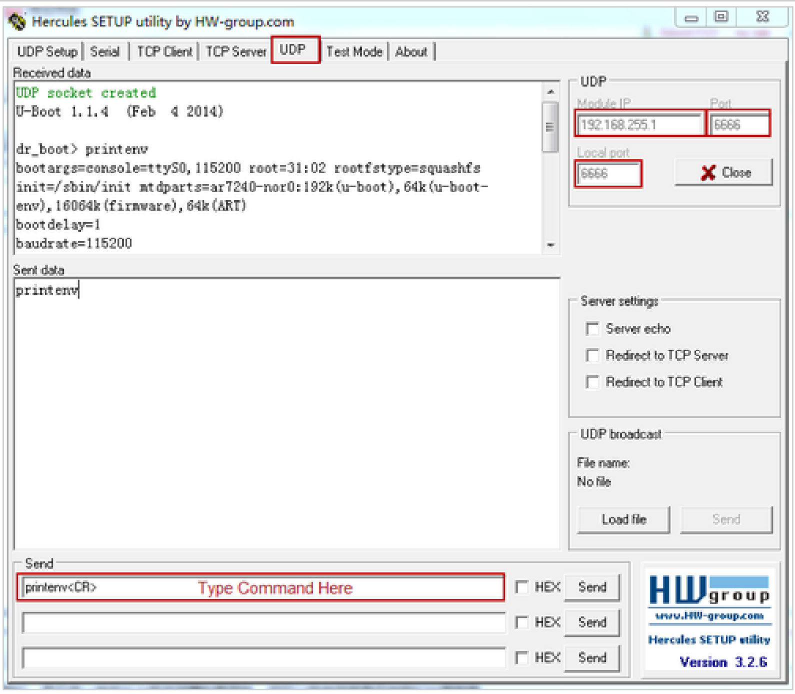
An instruction in Windows is as below

Set up TFTP server

Download the tftp server (recommend tftp32d.exe). And download the latest Yun firmware from http://www.dragino.com/downloads/index.php?dir=motherboards/ms14/Firmware/Yun/. The firmware we need is the kernel and rootfs-squashfs files. Put these firmware files and tftp32d at the same directory. Start the tftp server.

Download Hercules utility

Download Hercules, this is the tool we use to transfer commands to Yun Shield in Failsafe mode. Run Hercules and input correct parameters as below:



Received data

```
UDP socket created
U-Boot 1.1.4  (Feb  4 2014)

dr_boot> printenv
bootargs=console=ttyS0,115200 root=31:02 rootfstype=squashfs
init=/sbin/init mtdparts=ar7240-nor0:192k(u-boot),64k(u-boot-
env),16064k(firmware),64k(ART)
bootdelay=1
baudrate=115200
```

Sent data

```
printenv
```

UDP

Module IP: 192.168.255.1
Port: 6666
Local port: 6666

Server settings

- [ ] Server echo
- [ ] Redirect to TCP Server
- [ ] Redirect to TCP Client

UDP broadcast

File name:
No file

Send

printenv<CR>

Hercules Configure

Protocol: UDP Module IP: 192.168.255.1 Port: 6666 Local port: 6666

Connect your PC to MS14

Connect the PC and MS14 LAN port via an Ethernet cable. Set up PC with below LAN IP 192.168.255.2 and netmask 255.255.255.0. Disable PC’s firewall.

Power up MS14 to Web Failsafe mode

Press the Failsafe button and power up MS14; user will see all the LEDs blink together, release the button after 10 seconds and there are some messages will pop up in the Hercules panel, which means the MS14 has been in Web Failsafe Netconsole mode and ready to access commands.

User can type the commands in Hercules to transfer and upgrade Yun Shield to the latest firmware with factory settings.

The update commands are as below, replace the xxx with the actually version. Note: when typing command in Hercules: user must add **<CR>** at the end of each command; $ is a special char in Hercules, user should double it (key two $$) when typing.

Upgrade Kernel

```
tftpboot 0x81000000 ms14-arduino-yun-kernel-xxx.bin
erase 0x9fea0000 +0x140000
cp.b 0x81000000 0x9fea0000 $filesize
```

Upgrade rootfs

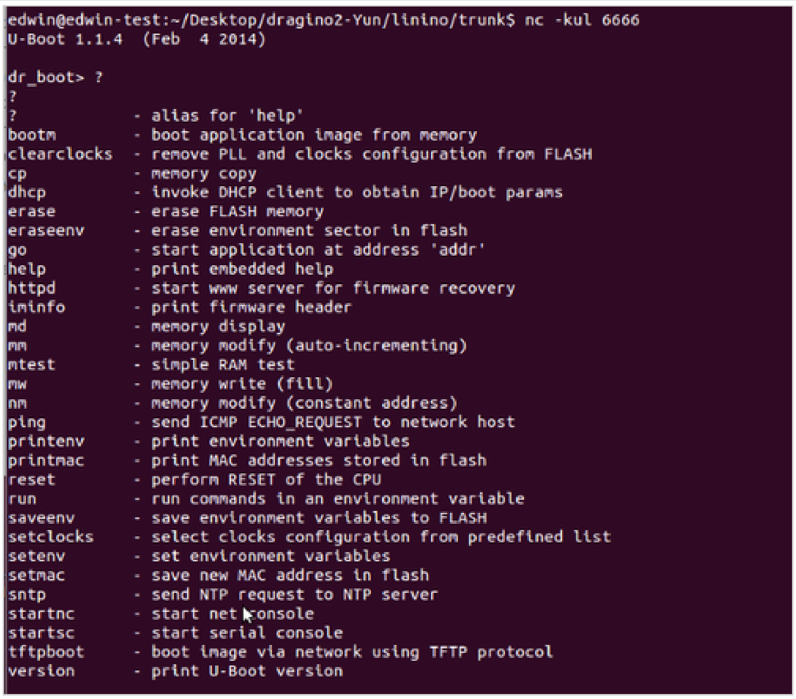
```
tftpboot 0x81000000 ms14-arduino-yun--rootfs-squashfs-xxx.bin
erase 0x9f050000 +0xe50000
cp.b 0x81000000 0x9f050000 $filesize
```

Reset to the new firmware

```
setenv bootcmd bootm 0x9fea0000
saveenv
reset
```

Warning: User should use exactly address number in the erase and cp.b shows, wrong address number may properly destroy the boot-loader of MS14 and the device won’t boot anymore. Or destroy the radio data of MS14 which may lead to a poor wifi performance or incorrect MAC addresses.

Recover in Linux is similar with Windows, the mail different is that the tool use in Linux is nc and runs with nc -kul 6666. Below shows that the MS14 has been in Failsafe Netconsole mode and detected by nc.



```
edwin@edwin-test:~/Desktop/dragino2-Yun/linino/trunk$ nc -kul 6666
U-Boot 1.1.4  (Feb  4 2014)

dr_boot> ?
?
?             - alias for 'help'
bootm         - boot application image from memory
clearclocks   - remove PLL and clocks configuration from FLASH
cp            - memory copy
dhcp          - invoke DHCP client to obtain IP/boot params
erase         - erase FLASH memory
eraseenv      - erase environment sector in flash
go            - start application at address 'addr'
help          - print embedded help
httpd         - start www server for firmware recovery
iminfo        - print firmware header
md            - memory display
mm            - memory modify (auto-incrementing)
mtest         - simple RAM test
mw            - memory write (fill)
nm            - memory modify (constant address)
ping          - send ICMP ECHO_REQUEST to network host
printenv      - print environment variables
printmac      - print MAC addresses stored in flash
reset         - perform RESET of the CPU
run           - run commands in an environment variable
saveenv       - save environment variables to FLASH
setclocks     - select clocks configuration from predefined list
setenv        - set environment variables
setmac        - save new MAC address in flash
sntp          - send NTP request to NTP server
startnc       - start net console
startsc       - start serial console
tftpboot      - boot image via network using TFTP protocol
version       - print U-Boot version
```

Ubuntu to access netconsole

Retrieved from "http://wiki.dragino.com/index.php?title=Getting_Start_with_Arduino_Yun&oldid=3033"

- This page was last modified on 15 January 2017, at 04:26.