



Yun Shield User Manual

VERSION: 1.0

Version	Description	Date
0.1	Initiate	2014-Jun-21
1.0	Release	2014-Jul-08



Index:

1	Intro	oduction	n	3
	1.1	What	t is Yun Shield	3
	1.2	Speci	fications	3
	1.3	Featu	ıres	3
	1.4	Syste	m Structure	5
2	Cont	figure Y	un Shield	7
	2.1	Find	the ip addresses	7
	2.2	Confi	gure Method	8
		2.2.1	Access via web interface	8
		2.2.2	Access via SSH	8
	2.3	Web	Configure Pages	9
		2.3.1	General Set Up	9
		2.3.2	Arduino related set up	9
		2.3.3	Upgrade	10
3	Use	with Ar	duino boards	11
	3.1	Conn	ects to Arduino Boards	11
		3.1.1	Connect to Leonardo	11
		3.1.2	Connect to Arduino Uno	11
		3.1.3	Connect to Arduino Duemilanove/Diecimila	13
		3.1.4	Connect to Arduino Mega2560	13
	3.2	Dete	ct Yun Shield	14
	3.3	Uploa	ad Sketch	16
	3.4	Bridg	e Library	16
4	Exar	nples		17
	4.1	Exam	ple 1: Say hello to Linux	17
	4.2	Exam	ple 2: Upload data to IoT Server	19
	4.3	Exam	ple 3: Log Data to USB flash	22
5	FAQ			24
	5.1	What	t is the difference between the official Arduino Yun and Yur	Shield?24
	5.2	ls Yur	n Shield compatible with a variant Arduino Board?	24
	5.3	Ardui	ino IDE doesn't detect Yun Shield	25
	5.4	Whe	re can I find the source code of Yun Shield?	25
	5.5	How	to Upload, Download or Edit files in Yun Shield	25
	5.6	How	to recover the Yun Shield	27
6	Refe	rence		29



1 Introduction

1.1 What is Yun Shield

Yun Shield is one of the most powerful shields for Arduino Board. Yun Shield is designed to solve the Internet connectivity and storage issue for Arduino Board.

Yun Shield runs Open Source OpenWrt system (Same system as runs in Arduino Yun) and it is fully compatible with Arduino IDE v1.5.4 or later. Yun Shield is the ideally choice for Arduino Projects which require various internet connections and more storage.



Basically, Yun Shield + Leonardo equally to the official Arduino Yun, but Yun Shield is more flexible because it can work with other Arduino board such as Uno, Duemilanove, Mega etc. And Yun Shield uses external wifi antenna which provides stability and possibility for various environments.

1.2 Specifications

- Processor: 400MHz, 24K MIPS
- Flash: 16MBytes
- ➢ RAM: 64MBytes
- Power Input: 4.75v ~ 23v via Arduino VIN pin
- > 1 x 10M/100M RJ45 connector
- 150M WiFi 802.11 b/g/n
- External Antenna via I-Pex connector
- > 1 x USB 2.0 host connector, used for USB storage or 3G connection
- 1 x Reset button
- Compatible with 3.3v or 5v I/O Arduino.

1.3 Features

✓ Open source Linux (OpenWrt) inside



- ✓ Low power consumption
- ✓ Compatible with Arduino IDE 1.5.4 or later, user can program, debug or upload sketch to Arduino board via Arduino IDE.
- ✓ Managed by Web GUI, SSH via LAN or WiFi
- ✓ Software upgradable via network
- ✓ Built-in web server
- ✓ Support internet connection via LAN port, WiFi or 3G dongle.
- \checkmark Support USB flash to provide storage for Arduino projects.
- ✓ Failsafe design provides robustly system.
- ✓ Compatible with Arduino Leonardo, Uno , Duemilanove, Diecimila, Mega





1.4 System Structure





POWER:

The Dragino HE is the core module of Yun Shield. The HE module requires around 200ma current when in full load, so it is powered by the Arduino VIN pins to avoid overheated in the Arduino onboard 5v LDO. So when Yun shield is in used, the Arduino board should be powered by DC port instead of USB port. The DC input can be 7v ~ 15v.



The USB Host of Yun Shield gets power from the Arduino +5v pin, since the +5v from Arduino comes from the +5V LDO, to avoid overheated on the Arduino Board, when the USB host is in used, it is recommended to use +7v DC.

Interface:

The RJ45, WiFi, USB Host and Failsafe are connected to the Dragino HE module directly. And the Dragino HE module use SPI and UART to communicate with Arduino Board. Yun Shield is compatible with 3.3v and 5v Arduino board. The **on board jumper SV1** is used to set the SPI and UART to 3.3v or 5v level.

The SPI interface is used to upload the sketches comes from the Arduino IDE. SPI interface only connects to Dragino HE during uploading so the Arduino SPI can still be used to connect to other SPI slave devices.

The UART interface is used for the Bridge class in Arduino, there are lots of examples explain how to use the bridge class in the Arduino IDE. It is the core of Yun solution. We must make sure the serial Interface of Arduino is not used by other hardware.



2 Configure Yun Shield

2.1 Find the ip addresses

The Yun Shield 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 port



At the first boot of Yun Shield, it will auto generate an unsecure WiFi network call *Dragino2-xxxxx*

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

Fall Back IP

70%	
U果网络支持此功能,则可以 您需要从网络系统管理员处约	以获取自动指派的 IP 设置。否则, 获得适当的 IP 设置。
 目动获得 IP 地址(0) 使用下去的 TB 地址(6) 	
	172 31 255 253
子网播码的	255 255 255 252
默认网关 (0):	22 25 25
● 白劫萍得 mars 服冬哭₩	b+i+ (B)
● 使用下面的 DNS 服务器	影地址(E):
首选 DNS 服务器(P):	2 2 2
备用 DNS 服务器(A):	1 1 1 1
同語中国語にの異ない	宣纪(M)

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

Detect IP from Arduino IDE

If Yun Shield'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 Yun Shield's IP address as described in <u>Detected Yun Shield</u>.



2.2 Configure Method

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

2.2.1 Access via web interface

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

192.168.240.1/cpi-bin/luci/webpanel/homepage	Q (1 + C) - EX (0/4) P + S + Q + A (1 + K + 4 +
公司最多 🍯 火銀官方社会 🗌 新手上牌 🍯 常用同社 🖬 天服商地	C 12
$\Theta \Theta$	Welcome to your Arduano Yún. Please enter password to access the web control panel
ARDUINO	PASSWORD
YUN	Default Log In Password: arduino
	Please be sure you have cookies enabled before proceeding.
	LOC IN

Default User name and Password for Yun Shield is root/Arduino.

2.2.2 Access via SSH

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

RuTTY Configuratio	n	23		
Category:	Posio estions for usur PuITY asso		SSH Access:	
E-Session Logging E-Terminal	Specify the destination you want to connect the Host Name (or IP address) Provide the second	to	IP address:	IP Address of Yun Shield
Bell	172.31.255.254 2	22	D 1	22
Features	Connection type:	Serial	Port:	22
Appearance Behaviour Translation	Load, save or delete a stored session Saved Sessions		User Name:	root
- Colours - Colours - Data - Proxy - Telnet - Rlogin B - SSH	Default Settings 192.168.255.1teinet	Load Save Delete	Password:	Arduino (default)
I Serial	Close window on exit: Always Never Only on clear	n exit		

Open Cano



2.3 Web Configure Pages

2.3.1 General Set Up

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.

Finder 🖂 🗌 Yan 🛱	*		- 0' X'
(全) ③ 192.168.240.1/cgi-bin/luci/stok=73 多 访问最多 🤐 火活面方站点 []] 新手上路 🔐	78b0444b49af0408188e10f2dee167/webpanel/coofig 常用同址 12		
Ø	For more advanced network co	nfiguration features, see the <u>advanced configuration panel</u>	
YÚ	YÛN BOARD CONFIGURATION	0	
	YÛN NAME *	Arduino Device Host Name	
	PASSWORD	Password for WEB / SSH log in	
	CONFIRM PAGSWORD	Password for WEB / SSH log in	
	TIMEZONE *	Rest of the World (UIC) -	
Firefox -			
	WIRELESS PARAMETERS CONFIGURE A WRELEBS NETWORK		
	WRELESS NAME *		Select WiFi network and key in the password
	SECURITY	Xons •	
	DISCARD	CONFIGURE & RESTART	
	REST API ACCESS	0	
	REST API ACCESS	OPEN WITH PASSWORD REST APIs allow you to access your sketch from the web, sending commands or exchanging configuration values. If your Yinis on a public network, or controlling sensitive sequement, or both, we recommend you leave the REST API password protected.	

2.3.2 Arduino related set up

IOT SERVER CONFIGURAT	ION
SERVER TYPE	xively -
FEED ID	
API KEY	
DEVICE NAME	DRAGINO-A84041136FFC
ARDUINO BOARD TYPE	Leonardo, M32, M32W 🗸
OPERATION MODE	Arduino Bridge 🔻
DEBUG LEVEL	Disable ▼

Arduino Board Type: Define the bootloarder/mcu type/ fuse setting during Sketch upload. Operation Mode: make sure it is in <u>Arduino Bridge</u> mode so the Bridge class for Arduino Yun works.



2.3.3 Upgrade

Yun Shield firmware can be upgraded via the GUI for bug fixes / system improvement or new features.

Go to $GUI \rightarrow Upgrade$ page and select the correct firmware to upgrade. The firmware used for web upgrade should be a sysupgrade type firmware, user can choose if keep settings or not after upgrade.

UPGRADE I	FIRMWARE
	KEEP SETTINGS:
c	HOOSE FIRMWARE: jjgg ms14-arduino-yun-squashfs-sysupgrade-v1. 3. 3. bin
	UPLOAD FIRMWARE
NEW IMAGE	IS UPLOADED, PLEASE CONFIRM
NEW IMAGE	IS UPLOADED, PLEASE CONFIRM 913121b61dab04b43f25fec89eb8fc18
MEW IMAGE md5 checksum: size:	IS UPLOADED, PLEASE CONFIRM 913121b61dab04b43f25fec89eb8fc18 15.43 MB (15.56 MB valid)
NEW IMAGE md5 checksum: size: Settings:	IS UPLOADED, PLEASE CONFIRM 913121b61dab04b43f25fec89eb8fc18 15.43 MB (15.56 MB valid) Clear settings after upgrade.
NEW IMAGE md5 checksum: size: Settings:	IS UPLOADED, PLEASE CONFIRM 913121b61dab04b43f25fec89eb8fc18 15.43 MB (15.56 MB valid) Clear settings after upgrade.
NEW IMAGE md5 checksum: size: Settings:	IS UPLOADED, PLEASE CONFIRM 913121b61dab04b43f25fec89eb8fc18 15.43 MB (15.56 MB valid) Clear settings after upgrade.
NEW IMAGE md5 checksum: size: Settings:	IS UPLOADED, PLEASE CONFIRM 913121b61dab04b43f25fec89eb8fc18 15.43 MB (15.56 MB valid) Clear settings after upgrade. CANCEL UPGRADE
NEW IMAGE md5 checksum: size: Settings:	IS UPLOADED, PLEASE CONFIRM 913121b61dab04b43f25fec89eb8fc18 15.43 MB (15.56 MB valid) Clear settings after upgrade. CANCEL UPGRADE PROCESS UPGRADE

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.

The firmware version info can be check from this link: <u>Yun Firmware Change log</u>



3 Use with Arduino boards

The Yun Shield use SPI for uploading sketch and use UART port for Bridge class to talk to the AVR. While connects Yun Shield to Arduino Board, below points should be checked:

- Whether the Arduino Board is power by DC jack
- If the board type setting is correct in Yun Shield. ()
- If the board type setting is correct in Arduino IDE
- Whether the Arduino SPI and UART is not influenced by other hardware
- Make sure the UART mode is in Arduino Bridge mode.

3.1 Connects to Arduino Boards

3.1.1 Connect to Leonardo

Simply plug the Yun Shield on top of the Leonardo, and power the Leonardo via DC jack. In Arduino IDE, the **board type** should select **Arduino Yun.**

3.1.2 Connect to Arduino Uno

 In UNO, the uart connection between mega328P and mega16u2 will influence the bridge feature with Yun Shield. So we have to disconnect it by set the mega16u2 into reset mode. As below:



Note: USB upgrade/debug won't work after this change, User will have to upgrade sketch and debug via Arduino IDE via WiFi (see examples)

2) Add a "**Uno Yun**" board type in the file: Arduino\hardware\arduino\avr\board.txt. as below and reopen the Arduino IDE:



unoyun.name=Arduino Uno -- Dragino Yún unoyun.upload.via_ssh=true

unoyun.vid.0=0x2341 unoyun.pid.0=0x0043 unoyun.vid.1=0x2341 unoyun.pid.1=0x0001 unoyun.upload.tool=avrdude unoyun.upload.protocol=arduino unoyun.upload.maximum_size=32256 unoyun.upload.maximum_data_size=2048 unoyun.upload.speed=57600 unoyun.upload.disable_flushing=true unoyun.upload.use_1200bps_touch=true unoyun.upload.wait_for_upload_port=true

unoyun.bootloader.tool=avrdude unoyun.bootloader.low_fuses=0xff unoyun.bootloader.high_fuses=0xde unoyun.bootloader.extended_fuses=0x05 unoyun.bootloader.file=optiboot/optiboot_atmega328.hex unoyun.bootloader.unlock_bits=0x3F unoyun.bootloader.lock_bits=0x0F

3) Put the Yun Shield on top of Uno and power it via DC jack.





3.1.3 Connect to Arduino Duemilanove/Diecimila

1) In Duemilanove/Diecimila, the mega avr uart interface is connected to the FTDI chip, we have to disconnect them as shown in below picture:



- Add a "Duemilanove Yun" board type in the file: Arduino\hardware\arduino\avr\board.txt. user can use the <u>UnoYun board type</u> if they has mega328p. for mega328/mega168/mega168p, they can modify the upload.maximum_data_size/ upload.maximum_size and build.mcu accordingly.
- 3) Put the Yun Shield on top of Duemilanove and power it via DC jack.

3.1.4 Connect to Arduino Mega2560

 In Mega2560, the uart connection between mega2560 and mega16u2 will influence the bridge feature with Yun Shield. So we have to disconnect it by set the mega16u2 into reset mode. As below:



2) Add a "**Mega2560 Yun**" board type in the file: Arduino\hardware\arduino\avr\board.txt. as below and reopen the Arduino IDE:



mega2560Yun.name=Arduino Mega 2560 -- Dragino Yún mega2560Yun.upload.via_ssh=true

mega2560Yun.vid.0=0x2341 mega2560Yun.pid.0=0x0044 mega2560Yun.vid.1=0x2341 mega2560Yun.pid.1=0x003f mega2560Yun.upload.tool=avrdude mega2560Yun.upload.protocol=arduino mega2560Yun.upload.maximum_size=258048 mega2560Yun.upload.maximum_data_size=8192 mega2560Yun.upload.speed=57600 mega2560Yun.upload.disable_flushing=true mega2560Yun.upload.use_1200bps_touch=true mega2560Yun.upload.wait_for_upload_port=true

mega2560Yun.bootloader.tool=avrdude mega2560Yun.bootloader.low_fuses=0xff mega2560Yun.bootloader.high_fuses=0xd8 mega2560Yun.bootloader.extended_fuses=0xfd mega2560Yun.bootloader.file=stk500v2/stk500boot_v2_mega2560.hex mega2560Yun.bootloader.unlock_bits=0x3F mega2560Yun.bootloader.lock_bits=0x0F

3) Put the Yun Shield on top of Mega2560 and power it via DC jack.

3.2 Detect Yun Shield

Make sure your laptop and Yun Shield are in the same network. The Yun Shield will broadcast data in this network and the Arduino IDE will receive this data and show the Yun Shield in *Tools* \rightarrow *Port*.



💿 Blink Arduino 1.5.4			
File Edit Sketch Tool	s Help		
Blink	Auto Format Archive Sketch Fix Encoding & Reload	Ctrl+T	
#include <consol< td=""><td>Serial Monitor</td><td>Ctrl+Shift+M</td><td></td></consol<>	Serial Monitor	Ctrl+Shift+M	
const int ledPin	Board	•	
int incomingByte:	Port	٠	 Dragino at 10.130.1.228 (Arduino Yún)
<pre>void setup() { // initialize s</pre>	Programmer Burn Bootloader	Þ	
Bridge.begin();			

Console.begin();



3.3 Upload Sketch

- 1) In the Arduino IDE, choose the correct board type for the AVR module.
- 2) In Arduino IDE \rightarrow port, choose the correct port. (should be Arduino Yun port with an ip address)
- 3) In the Yun Shield GUI \rightarrow Sensor page, choose the correct board type for upload.
- 4) Compile the sketch and upload it to the Arduino Board. During upload, The Yun Shield will ask you to key in the password, by default, the password is arduino.

Blink Arduino 1.5.4	
File Edit Sketch Tools Help	
Blink Compile and upload firmware to avr	
#include ⟨Console.h⟩	
const int ladpin = 8. // the min that the LED is attached to	
int incomingButs: // survishls to vasal incoming ravial data into	
in incoming year of the incoming strike date inco	
void setup 0 {	
// 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:	
pirMode(ledFin, OUTPUT);	
Compile and Upload sketch in Arduino IDE	ī

3.4 Bridge Library

The Bridge Library simplifies the communication between the Arduino Board and Yun Shield.

Bridge commands from the AVR (Arduino Board) are interpreted by Python on the Yun Shield. Its role is to execute programs on the GNU/Linux side when asked by Arduino, provide a shared storage space for sharing data like sensor readings between the Arduino and the Internet, and receiving commands from the Internet and passing them directly to the Arduino.

There are detail explain and lots of example to show how to use Bridge in the Arduino Official Website. Reference link is: <u>http://arduino.cc/en/Reference/YunBridgeLibrary</u>



4 Examples

4.1 Example 1: Say hello to Linux

Introduction:

This example is a hello test between the Arduino and Yun Shield. The example can be found on the Arduino IDE--> File --> Examples --> Bridge --> ConsoleRead. Tutorial of this example can be found on http://arduino.cc/en/Tutorial/ConsoleRead. Below listing the code and add some detail to understand it with the Yun Shield:

Code:

#include <Console.h> //use Console class for Arduino IDE debug over WiFi, similar to Serial class,
String name;

void setup() {
 // Initialize Console and wait for port to open:
 Bridge.begin();
 Console.begin();

// Wait for Console port to connect
while (!Console);

Console.println("Hi, what's your name?"); //Data flow: Arduino --> Yun Shield --> Arduino IDE

```
}
```

```
void loop() {
  if (Console.available() > 0) {
     char c = Console.read(); //read the next char received, data flow: IDE --> Yun Shield--> Arduino
     // look for the newline character, this is the last character in the string
     if (c == ' n') {
        //print text with the name received
       Console.print("Hi ");
       Console.print(name);
       Console.println("! Nice to meet you!");
       Console.println();
       // Ask again for name and clear the old name
       Console.println("Hi, what's your name?");
       name = ""; // clear the name string
     }
     else {
                     // if the buffer is empty Cosole.read() returns -1
       name += c; // append the read char from Console to the name string
     }
  }
}
```

Screen Shot:



No line ending 🔻 300 baud

Send

ConsoleRead | Arduino 1.5.6-r2 File Edit Sketch Tools Help

•

ConsoleRead Console.println("Hi, what's your name?"); S Arduino at 172.31.255.254 (Arduino Yún) } void loop () { if (Console.available() > 0) { Hi, what's your name? char c = Console.read(); // read the next char received Hi edwin! Nice to meet you! $\ensuremath{{\prime}}\xspace$ // look for the newline character, this is the last c if (c == '\n') { Hi, what's your name? //print text with the name received Console.print("Hi "); Console.print(name); Console.println("! Nice to meet you!"); Console.println();

// Ark again for name and clear the old name Console.println("Hi, what's your name?"): name = "": // clear the name string } else { // if the buffer is empty Cosole.read() re name #= c; // append the read char from Console to th } }



4.2 Example 2: Upload data to IoT Server

Introduction:

This example shows how to log data to the public IoT server "Xively". The example is a modified version(change Serial to Console to fit for different Arduino Board and debug over WiFi) from Arduino IDE--> File --> Examples --> Bridge --> XivelyClient. Tutorial of this example can refer http://arduino.cc/en/Tutorial/YunXivelyClient.

Before upload the sketch, make sure:

- ✓ The Yun Shield already has internet access
- ✓ Input your FEED ID and API KEY according to the Tutorial. Note, The FEED ID should be within double quotation marks "".
- ✓ Change Serial Class to Console class to fit for different AVRs.

Below listing the code and add some detail to understand it with the Yun Shield:

Code:

// include all Libraries needed:
#include <Process.h> //Process lib use to call Linux Commands in Yun Shield
#include <Console.h> //Console lib, used to show debug info in Arduino IDE
#include "passwords.h" // contains my passwords, see below

/*

NOTE: passwords.h is not included with this repo because it contains my passwords. You need to create it for your own version of this application. To do so, make a new tab in Arduino, call it passwords.h, and include the following variables and constants:

#define APIKEY	"foo"	// replace your pachube api key here
#define FEEDID	"0000"	// replace your feed ID
#define USERAGENT	"my-project"	<pre>// user agent is the project name</pre>
*/		

// set up net client info: const unsigned long postingInterval = 60000; //delay between updates to xively.com unsigned long lastRequest = 0; // when you last made a request String dataString = "";

void setup() {

}

// start console:
Bridge.begin();
Console.begin();

while (!Console); // wait for Network Serial to open
Console.println("Xively client");

// Do a first update immediately
updateData();
sendData();
lastRequest = millis();

void loop() { *Yun Shield User Manual*



// get a timestamp so you can calculate reading and sending intervals: long now = millis();

```
// if the sending interval has passed since your
  // last connection, then connect again and send data:
  if (now - lastRequest >= postingInterval) {
     updateData();
     sendData();
     lastRequest = now;
  }
}
void updateData() {
  // convert the readings to a String to send it:
  dataString = "Temperature,";
  dataString += random(10) + 20;
  // add pressure:
  dataString += "\nPressure,";
  dataString += random(5) + 100;
}
// this method makes a HTTP connection to the server:
void sendData() {
  // form the string for the API header parameter:
  String apiString = "X-ApiKey: ";
```

```
apiString += APIKEY;
```

```
// form the string for the URL parameter:
String url = "https://api.xively.com/v2/feeds/";
url += FEEDID;
url += ".csv";
```

// Send the HTTP PUT request, form the linux command and use Process Class to send this command to Yun Shield

```
// Is better to declare the Process here, so when the
// sendData function finishes the resources are immediately
// released. Declaring it global works too, BTW.
Process xively;
Console.print("\n\nSending data...");
xively.begin("curl");
xively.addParameter("-k");
xively.addParameter("--request");
xively.addParameter("PUT");
xively.addParameter("--data");
xively.addParameter(dataString);
xively.addParameter("--header");
xively.addParameter(apiString);
xively.addParameter(url);
xively.run();
Console.println("done!");
// If there's incoming data from the net connection,
// send it out the Console:
while (xively.available() > 0) {
```

```
while (xively.available() > 0) {
    char c = xively.read();
    Console.write(c);
}
```

```
}
```



Screen Shot:



200	PUT	feed		03:33:46 L	ТС
200	PUT	feed		03:32:44 U	тс
💿 Ardui	no at 1	72.31.255.254 (Arduino	Yún)		
				Send	
Xively cl	lient				Â
Sending	data	done!			ш
Sending	data	done!			
Sending	data	done!			
Sending	data	done!			
Sending (data	done!	No line ending	▼ 300 baud	Ŧ

Triggers

Triggers provide 'push' capabilities by sending HTTP POST requests to a URL of your choice when a condition has been satisfied.



4.3 Example 3: Log Data to USB flash

Introduction:

This example shows how to log data to a USB flash. The sketch used in this example is same as http://wiki.dragino.com/index.php?title=Arduino_Yun_examples#Log_sensor_data_to_USB_flash h. And the source code is in this link.

The Yun Shield will auto mount the USB flash to directory /mnt/sda1. And the sketch will append the sensor data to the file /mnt/sda1/data/datalog.csv. So make sure there is such file in the USB flash before running the sketch

Code:

```
#include <FileIO.h>
                          //FileIO class allow user to operate Linux file system
#include <Console.h>
                         //Console class provide the interactive between IDE and Yun Shield
void setup() {
  // Initialize the Console
  Bridge.begin();
  Console.begin();
  FileSystem.begin();
  while(!Console); // wait for Serial port to connect.
  Console.println("Filesystem datalogger\n");
}
void loop () {
  // make a string that start with a timestamp for assembling the data to log:
  String dataString;
  dataString += getTimeStamp();
  dataString += " , ";
  // read three sensors and append to the string:
  for (int analogPin = 0; analogPin < 3; analogPin++) {
     int sensor = analogRead(analogPin);
     dataString += String(sensor);
     if (analogPin < 2) {
       dataString += ","; // separate the values with a comma
     }
  }
  // open the file. note that only one file can be open at a time,
  // so you have to close this one before opening another.
  // The USB flash card is mounted at "/mnt/sda1" by default
  File dataFile = FileSystem.open("/mnt/sda1/data/datalog.csv", FILE_APPEND);
  // if the file is available, write to it:
  if (dataFile) {
     dataFile.println(dataString);
     dataFile.close();
     // print to the serial port too:
     Console.println(dataString);
  }
  // if the file isn't open, pop up an error:
  else {
```



Console.println("error opening datalog.csv"); } delay(15000); //Write every 15 seconds } // getTimeStamp function return a string with the time stamp // Yun Shield will call the Linux "date" command and get the time stamp String getTimeStamp() { String result; Process time; // date is a command line utility to get the date and the time // in different formats depending on the additional parameter time.begin("date"); time.addParameter("+%D-%T"); // parameters: D for the complete date mm/dd/yy T for the time hh:mm:ss time.run(); // run the command // read the output of the command while(time.available()>0) { char c = time.read(); if(c != '\n')

```
return result;
```

result += c;

```
}
```

}

Screen Shot:

Datalog Arduino 1.5.6-r2	and a second second	0			
File Edit Sketch Tools Help			/mnt/sda1/data/datalog.csv - root	D172.31.255.254	
			🗴 🖩 🙆 🖻 🗴 🛍 🗙 🏽 🗠	🗠 🛤 🏦 🛝 🔒 🔷 🇼	
Datalog			07/09/14-07:35:53 , 360,403,395 07/09/14-07:36:08 , 143,129,131		^
}	o Arduino at 172.31.255.254 (Arduin	o Yún)	07/09/14-07:36:23 , 234,222,198 07/09/14-07:36:39 , 575,609,524 07/09/14-07:36:54 , 387,429,411 07/09/14-07:37:09 , 155,143,142		
<pre>// This function return a string // MS14 will call the Linux "date</pre>	07/00/14-07:57:20 550 572 494	Send	07/09/14-07:37:24 , 400,403,341 07/09/14-07:37:39 , 602,641,552		
String getTimeStamp() { String result:	07/09/14-07:57:54 , 389, 433, 417		07/09/14-07:37:54 , 539,594,543 07/09/14-07:38:10 , 387,432,417 07/09/14-07:38:25 , 214,239,258		E
Process time; // date is a command line util:	07/09/14-07:58:09 , 156,142,141 07/09/14-07:58:24 , 608,652,566		07/09/14-07:38:40 , 359,401,392 07/09/14-07:38:55 , 596,652,582		
<pre>// in different formats depend: time.begin("date"):</pre>	07/09/14-07:58:55 , 309,347,349		07/09/14-07:39:10 , 485,498,420 07/09/14-07:39:26 , 501,518,438 07/09/14-07:39:41 , 582,618,534		
time.addParameter("+%D-%T");	07/09/14-07:59:10 , 103, 94, 110 07/09/14-07:59:25 , 403, 406, 343	IDE Window	07/09/14-07:39:56 , 615,669,591 07/09/14-07:40:11 , 458,508,475	Winscp Windo	w
time.run(); // run the comman	07/09/14 07:59:56 , 589,624,536		07/09/14-07:40:26 , 671,919,954 07/09/14-07:40:41 , 375,374,319 07/09/14-07:40:57 , 613,671,601		
// read the output of the comm	07/09/14-08:00:26 362 404 395		07/09/14-07:41:12 , 145,155,183		
while(time.available()>0) {	07/09/14-08:00:41 , 509, 563, 519		07/09/14-07:41:27 , 121,109,122		
<pre>char c = time.read();</pre>	07/09/14-08:00:56 , 443, 450, 380		07/09/14-07:41:57 , 493,547,510		
if(e != '\n')	07/09/14-08:01:11 , 104,94,116	=	07/09/14-07:42:13 , 101,92,115		
result += c;		-	07/09/14-07:42:28 , 585,616,525		
}	Autoscroll	No line ending 👻 300 baud	07/09/14-07:42:58 , 183,203,230 07/09/14-07:43:13 , 109,107,133		
return result;			07/09/14-07:43:28 , 367,365,310 07/09/14-07:43:44 , 622,680,606		-
3			Line: 1/103 Column: 1	Character: 48 (0x30)	



5 FAQ

5.1 What is the difference between the official Arduino Yun and Yun Shield? In Hardware Aspect

Both Arduino Yun and Yun Shield have the same CPU, Memory size and RAM size for the Linux system. The Arduino Yun is an integrated of Linux Part and MCU part, Yun Shield is designed as a shield which can be used with exist Arduino boards.

Basically, The Yun Shield + Arduino Leonardo equally to an Arduino Yun, but Yun Shield is more flexible because it can be used with other Arduino boards such as Arduino Uno, Duemilanove, Diecimila etc.

Yun Shield is duplicable and producible: The design of Yun Shield is open and the most complicate and difficult parts are done in the Dragino HE module. User can purchase the Dragino HE module separately to customized their IoT project and release their variant Yun Solution.

Stable and Flexible WiFi performance: Arduino Yun use chip antenna design, if there is a shield on top of the Arduino Yun, the wifi will be greatly shielded and lead to a poor wifi performance. Instead, Yun Shield use external Antenna design, user can connect different type of antennas to the i-pex connector of Yun Shield, this make the installation is more flexible and possible to transfer the signal to several km distance.

In Software Aspect

The Yun Shield software is derived from Arduino Yun with some bugs fixed; feature added and support more board types.

5.2 Is Yun Shield compatible with a variant Arduino Board?

If the Arduino board is a variant from the boards described in <u>Support Board Type</u>, then it should be compatible. Below is the check list for the compatibility.

- ✓ The variant has 7~15v power in the VIN pin to power the Yun Shield.
- ✓ The variant has same definition and position of SPI pins in the ICSP header as the official board.
- ✓ The variant has same definition and position of D0 and D1 pins in the ICSP header in the official board.
- ✓ Check whether there are ICs connected to the SPI and UART of the AVR and evaluate if they will influence the communication between Yun Shield and the AVR MCU.

The <u>system structure</u> section well explains the working principle of Yun Shield, if user still not sure if Yun Shield is compatible with their board or having trouble in the compatibility. Then can send the board info to <u>support@dragino.com</u> and our support team will review and check it.



5.3 Arduino IDE doesn't detect Yun Shield

Check below points if this issue happens:

- ✓ The Arduino IDE version is 1.5.4 or later
- ✓ Your PC and Yun Shield are in the same network.
- ✓ If Yun Shield boot in advance than Arduino IDE, this may happen. So try to power off/on the Yun Shield and check again.

5.4 Where can I find the source code of Yun Shield?

The Yun Shield source can be found at: <u>https://github.com/dragino/linino</u>

5.5 How to Upload, Download or Edit files in Yun Shield

Yun Shield has a built-in Linux system and support SCP protocol. User can upload, download or edit the Linux files using SCP tools.

In windows OS, the scp tool is winscp. Install it and log into Yun Shield as below:

- ♦ Host Name: Yun Shield IP address
- ♦ User Name: root
- ♦ Password: arduino (default)
- ♦ Protocol: SCP

Session	Session				
Stored sessions Environment Directories SSH Preferences	Host name:	Port number:			
	192.168.255.1	22 🔶			
	User name: Password:				
	root	••••			
	Private key file:				
	Protocol File protocol: SCP				
			Select cold		

The log in process will alter two warning, just ignore it. After log in, a management panel will appear. The left part of this panel is your PC's directories and the right part shows the directories of Yun Shield, you can upload/download files by dragging, or double click the files to modify its content.



Local Mark Files Comma	ands Se	ssion Options	Remote Help - (영 전 종	2 Defa	ult - 1	а.			
My documents			0 4 8 %	1.1 41	nota	₩ •⊖31.3**	ina - In	a da za	12
Witers/edwin/Documents				1		1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1			
Name Ext	Size	Туре	Changed	Name	Ext	1	Size	Change	d
9. edwin #2) HTC Touch Myn Music EMp Fetures Myn Fotos EMp Fetures 30 Orebox #3(25) Orebox #3(25) 30 Orebox #3(25) Ulada BurnNow 31 Ulada BurnNow With Machines 31 Stable Stable Stables Stables 32 schlassaml 3	272 402	Parent direct 文件央 文件会 文社会 文目 文目 文目 文件会 文社会 文社会 文社会 文社会 文社会 文社会 文社会 文社会 文社会 文社会	2011/3/1 2011/3/1 2011/1/2 2011/1/2 2011/1/2 2011/3/1 2011/3/2 2011/3/2 2011/2/2 2011/2/2 2011/2/2 2011/2/1 2011/1/2	Su bin cost bin sbin bin sbin bin bin	ay			2011/1/ 2011/3/ 2011/3/ 2011/3/ 2011/3/ 2011/3/ 2011/3/ 2011/3/ 2011/3/ 2011/3/ 2011/3/ 2011/3/ 2011/3/ 2011/3/	1 0:00 22 1 0:00 1 0:12 22 22 22 1 0:00 1 0:00 22 22 22 22 1 0:00 1 0:01 22 22 22 22 22 22 22 22 22 22 22 22 22
< [. ,						
B d 674 Bin 0 d 12 2 52 Bename 17 54 5 dir 1	ß FS Ca	av S F6 Mave	CT F7 Create D		in 0 of 15 FS Delete 217 FS	Properties	F10 Ouit		
the second of the cost of		p) - Ja - O more	La ri circate o		to believe in the	A	SCP		0.00-20

WinSCP Panel

www.dragino.com



5.6 How to recover the Yun Shield

User may lost access to the Yun Shield by accidently change the Linux configure files to in correct setting, lost password or other caused. He or she is still able to recover the Yun Shield system by using the Failsafe u-boot of Yun Shield.

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 <u>http://www.dragino.com/downloads/index.php?dir=motherboards/ms14/Firmware/Yun/</u>. 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 <u>Hercules</u>, this is the tool we use to transfer commands to Yun Shield in Failsafe mode. Run Hercules and input correct parameters as below:



Connect your PC to Yun Shield

Connect the PC and Yun Shield via an Ethernet cable. Set up PC with below LAN IP **192.168.255.2** and netmask **255.255.0**. Disable PC's firewall.

Power up Yun Shield to Failsafe mode

Press the Failsafe button and power up Yun Shield; 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 Yun Shield has been in 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.

Yun Shield User Manual

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

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 Yun Shield and the device won't boot anymore. Or destroy the radio data of Yun Shield 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 Yun Shield has been in Failsafe Netconsole mode and detected by nc.

dr_boot> ?	
?	
?	- alias for 'help'
ootm	 boot application image from memory
clearclocks	 remove PLL and clocks configuration from FLASH
EP.	- memory copy
dhcp	 invoke DHCP client to obtain IP/boot params
erase	- erase FLASH memory
eraseenv	 erase environment sector in flash
30	 start application at address 'addr'
help	 print embedded help
httpd	 start www server for firmware recovery
iminfo	 print firmware header
nd	- memory display
าก	 memory modify (auto-incrementing)
ntest	 simple RAM test
nw	- memory write (fill)
חר	 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
ftpboot	 boot image via network using TFTP protocol
version	- print U-Boot version



6 Reference

- ✓ Yun Shield hardware source <u>https://github.com/dragino/modules/tree/master/hardware/YunShield</u>
- ♦ Yun Shield software source code <u>https://github.com/dragino/linino</u>
- Arduino Yun Bridge Official page <u>http://arduino.cc/en/Reference/YunBridgeLibrary</u>
- Arduino Yun Official Forum <u>http://forum.arduino.cc/index.php?board=93.0</u>