使用 **Python** 与你的 **Arduino Yun** 进行交互

Phodal Huang

September 8, 2017

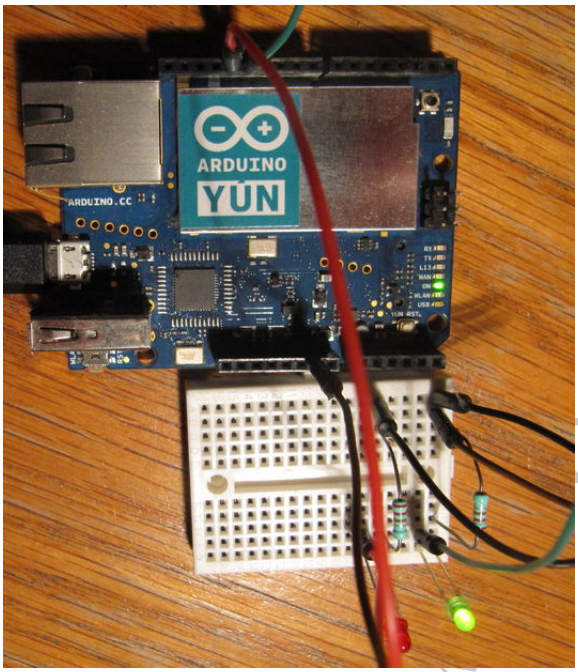
目录

原文链接：https://www.wandianshenme.com/play/arduino-yun-linino-use-python-control-led

如果你想使用 Python 2.x （运行在 Arduino Yun 上的 Linino）来控制你的 Arduino 部分的 GPIO 引脚。那么，你就应该阅读本篇的内容。

步骤 1：连接电路

其实实物图的连线如下所示：



在上图中，您可以看到串联 220 欧姆电阻，并连接到数字引脚 12 和 13 的两个 LED。

为了从 Arduino Yun 的 Linino 一侧与 Arduino 通信，您将使用 Bridge 库。它是 Arduino 安装的标准部分。要将其包含在您的 Arduino 代码中：

```
#include <Bridge.h>
```

要启动桥（bridge），请在 setup() 例程中初始化它：

```
// Start using the Bridge.
Bridge.begin();
```

在这个例子中，Python 将通过两个变量与桥接器进行交互：

```
// Here we will hold the values coming from Python via Bridge.
char D12value[2];

char D13value[2];
```

那么现在就只剩下一个问题，求当前的值：

```
Bridge.get("D12", D12value, 2);

Bridge.get("D13", D13value, 2);
```

在下一步中，您将看到 Arduino 的完整代码。

步骤 2：**Arduino** 代码

```
// Arduino Yun listens to python script via Bridge library to turn digital
    pins on/off.
// H.Zimmerman, 9-12-2014.
// Arduino Yun.

#include <Bridge.h>
#include <stdio.h>

// Here we will hold the values coming from Python via Bridge.
char D12value[2];
char D13value[2];

void setup() {
  // Zero out the memory we're using for the Bridge.
  memset(D12value, 0, 2);
  memset(D13value, 0, 2);

  // Initialize digital pins 12 and 13 as output.
  pinMode(12, OUTPUT);
  pinMode(13, OUTPUT);

  // Start using the Bridge.
  Bridge.begin();
}

void loop() {
  // Write current value of D12 to the pin (basically turning it on or off).
```

```
27   Bridge.get("D12", D12value, 2);
28 int D12int = atoi(D12value);  digitalWrite(12, D12int);
29
30   // An arbitrary amount of delay to make the whole thing more reliable.
         YMMV
31   delay(10);
32
33   // Write current value of D13 to the pin (basically turning it on or off).
34   Bridge.get("D13", D13value, 2);
35   int D13int = atoi(D13value);
36   digitalWrite(13, D13int);
37
38   // An arbitrary amount of delay to make the whole thing more reliable.
         YMMV
39   delay(10);
40 }                                                    }
```

步骤 3：实现 **Python** 部分

运行在您的 Arduino Yun 上的 Linux Linino 的 python 脚本，也使用 Bridge 库。下一步的例子，就是连续闪烁灯光。我相信你可以让它做一些更有趣的事情。

事实上，在为我女儿做了 Arduino Yun 夜灯之后，我做出了这样的东西：

引脚 12 连接到一个橙色 LED，告诉她可以醒来，引脚 13 连接到一个红色的 LED，告诉她它还在困倦的时间。她像父母那样爱她！因为 Arduino Yun 连接到家庭网络，所以我可以从任何位置简单地进入它，并改变 LED 的开启和关闭的时间。而这一切都由 cron 脚本来处理。

为了确保可以执行 python 脚本，你必须做两件事情。首先，请确保如下的代码是您的 python 脚本中的第一行：

```
1 #!/usr/bin/python
```

接下来，使脚本可执行。例如，如果它被称为 leds.py，您将在连接到 Arduino Yun 的 Linino shell 的终端中键入此命令：

```
1 chmod +x leds.py
```

现在你应该可以执行 python 脚本了：

```
1 ./leds.py
```

或者：

```
1 python leds.py
```

完整的 python 脚本在下一步，也是最后一步。Enjoy it！

步骤 4：Python 代码

如下：

```
#!/usr/bin/python

# Test to (un)set pin 12 and 13 on the Arduino Yun.
# H.Zimmerman, 09-12-2014.
# henszimmerman@gmail.com

import sys
sys.path.insert(0, '/usr/lib/python2.7/bridge')

from time import sleep

from bridgeclient import BridgeClient as bridgeclient
value = bridgeclient()

for idx in range(0, 100):
    value.put('D12','0')
    value.put('D13','1')
    sleep(0.1)
    value.put('D12','1')
    value.put('D13','0')
    sleep(0.1)

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

原文链接：https://www.wandianshenme.com/play/arduino-yun-linino-use-python-control-led