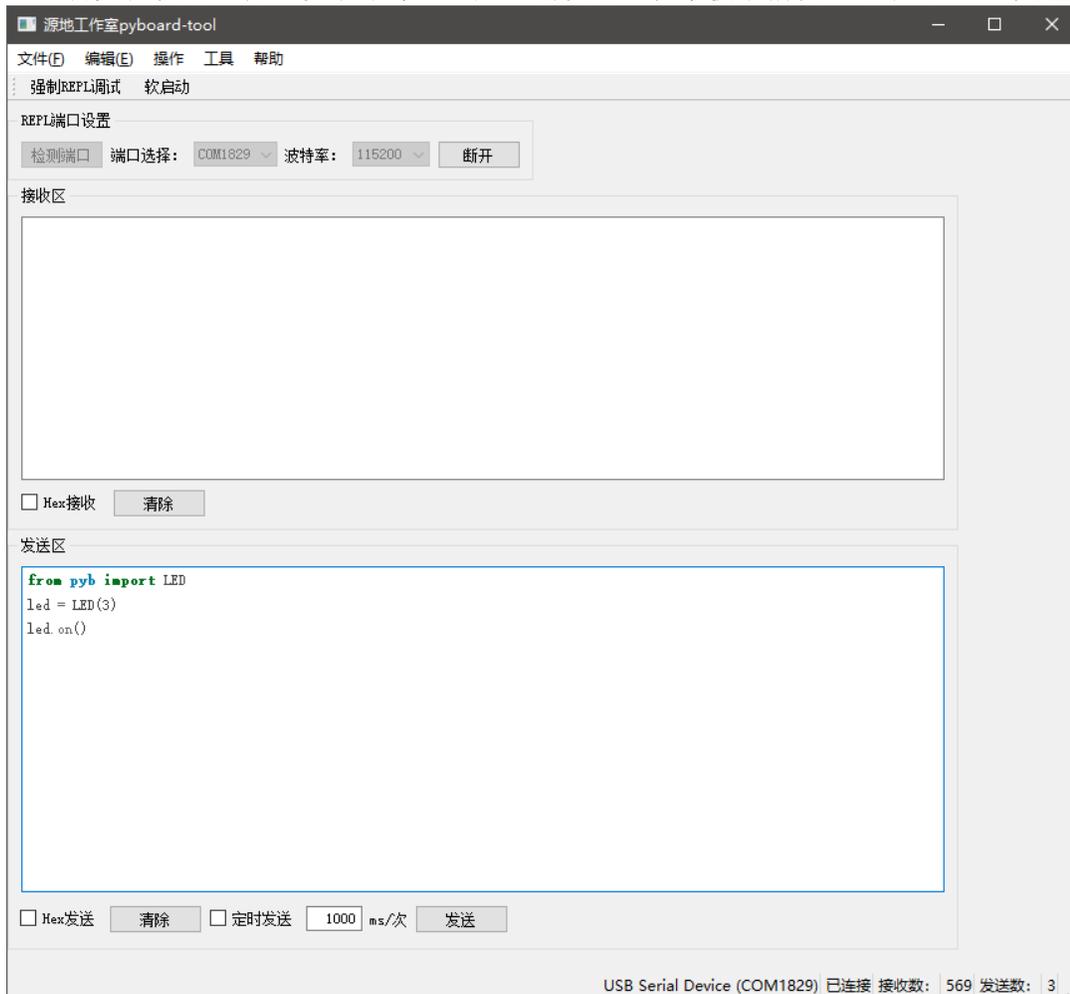


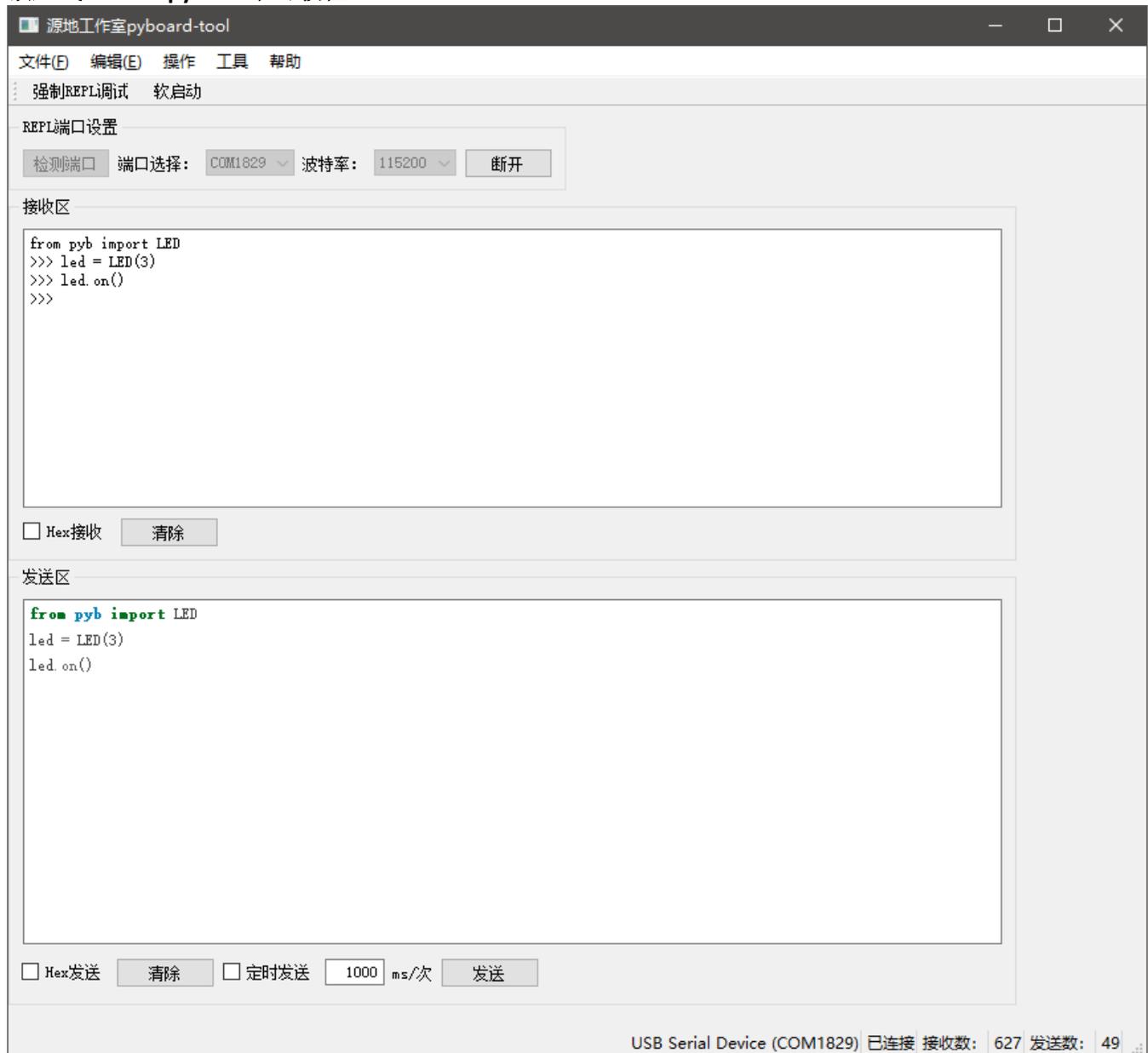
本课学习内容：

- 学会 REPL 调试方式，进行点灯，熄灯，翻转灯的操作。
- 学会 REPL 进行交互式调试
- 学会 REPL 交互调试发现 main.py 中的 bug

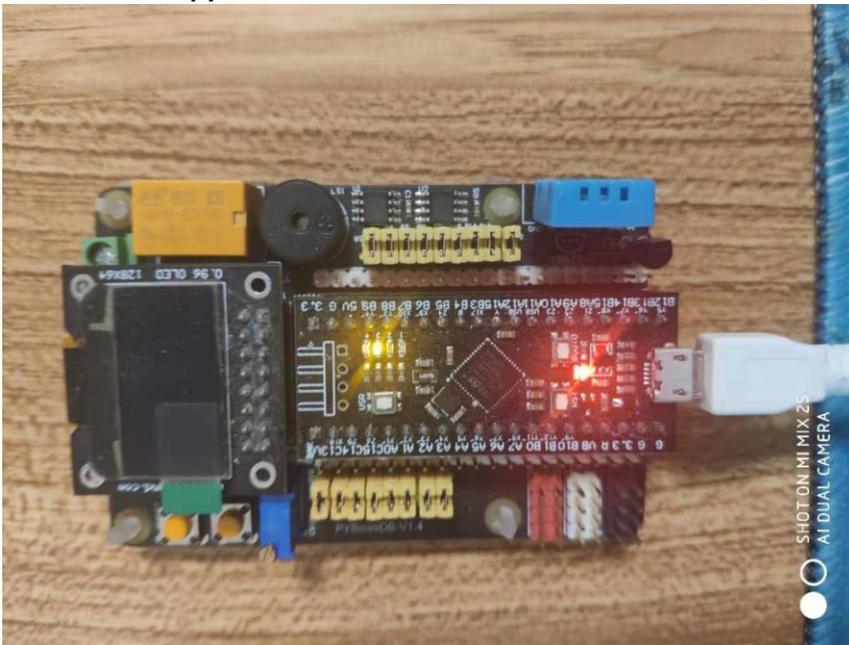
首先我们进入 Pyboard 生成的移动硬盘中，将 main.py 内容清空或者直接将 main.py 删除，保存好(再次强调一定等 LED1 灯熄灭后再进行下一步动作)，打开我们的 Pyboard-tool 工具，我们现在尝试下使用 REPL 直接控制我们板子。

首先在发送区键入以下命令。然后点击发送，观察板子情况，这就是 REPL 交互调试。

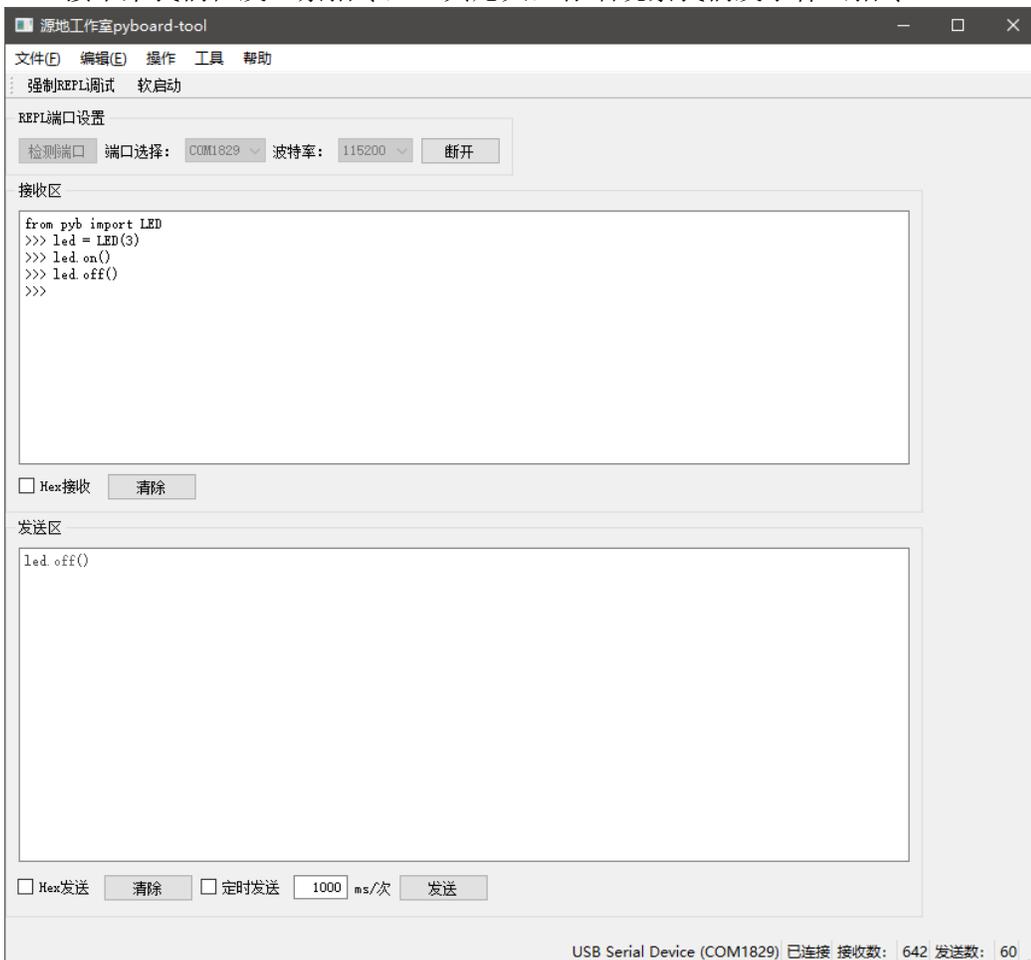




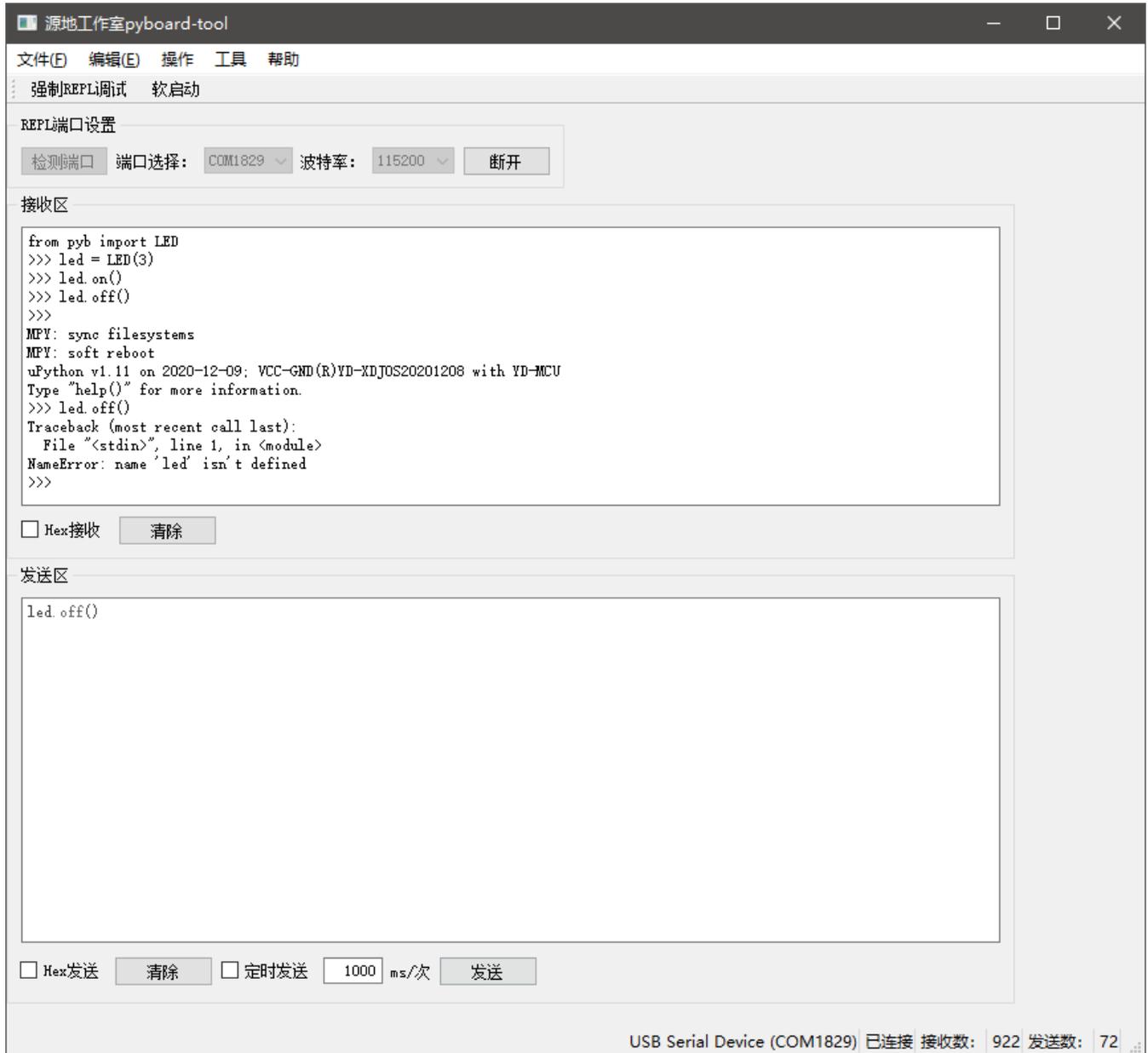
注意你点击发送后，会在接收区也有显示，并伴随着现象发生，这时候你会发现，第三个灯被点亮了，现场照片表现如下。



发现那个第 3 个 LED 被点亮，黄灯被点亮。
接下来我们在发一条指令，让其熄灭，仔细观察我们发了什么指令。

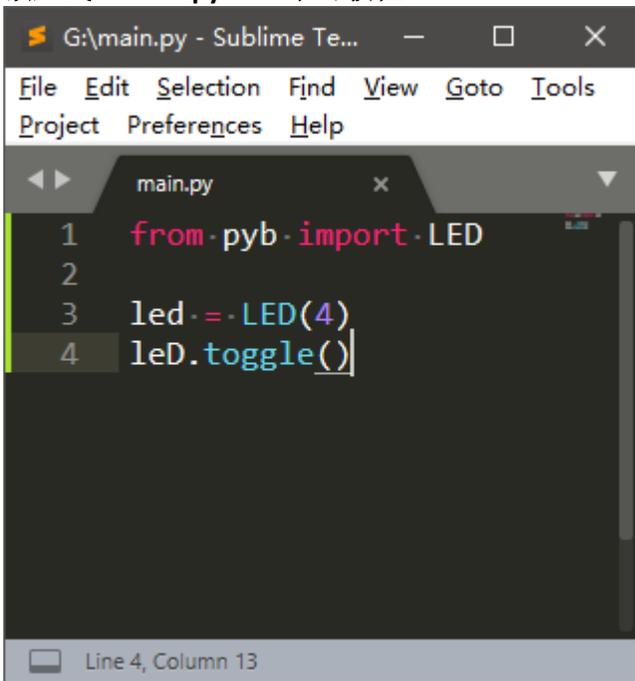


我们只发送了 led.off()这条语句就可以熄灭灯，如果没有上次一发送的那些语句，第一次就发 led.off()，REPL 交互是会报错的，不信来试试，首先按一下软复位，清除一下“内存”，咱们第一次就发 led.off()，看看情况。



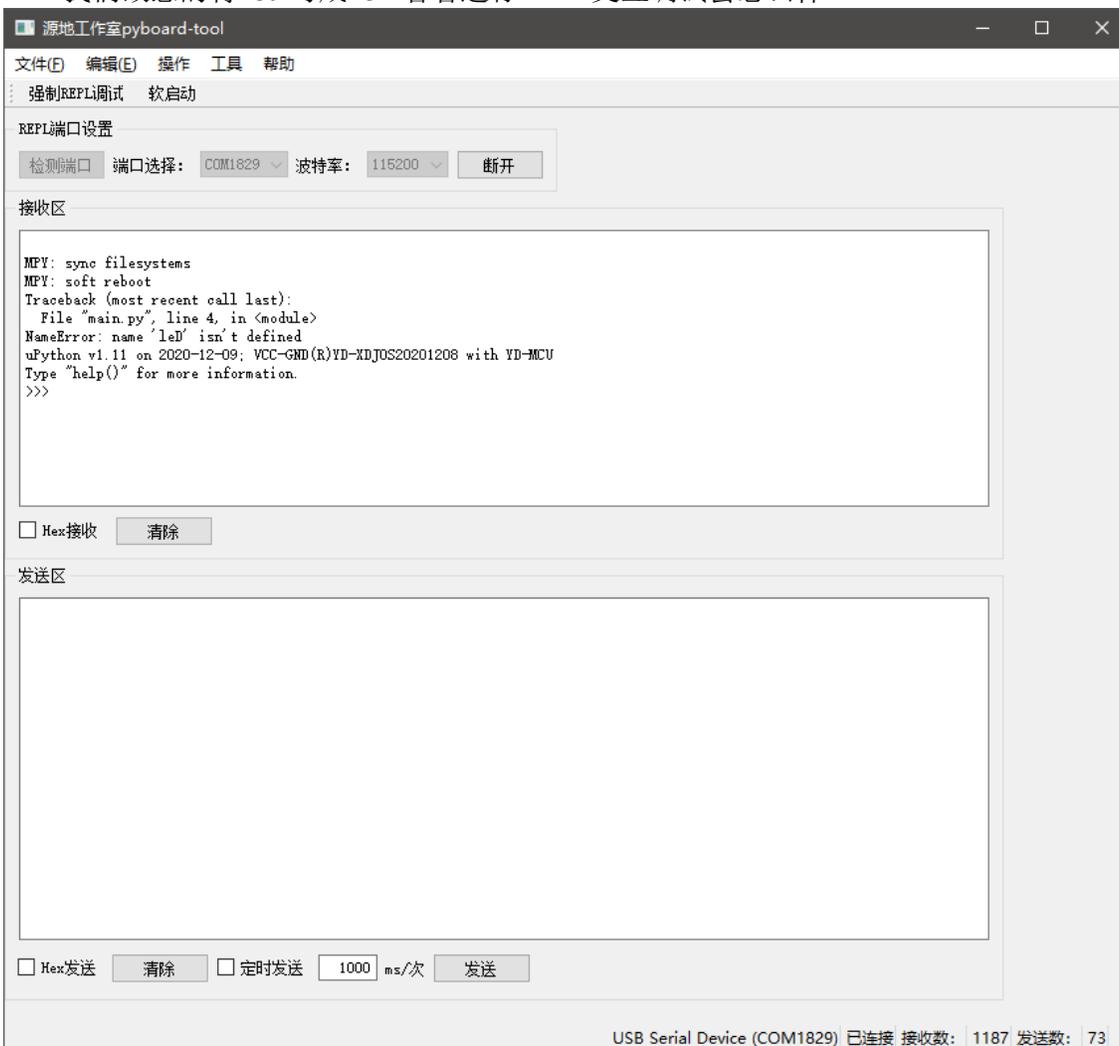
系统会报错，说 led 没有被定义。所以你应该懂了是怎么回事了吧。

下面我们将用 REPL 交互调试功能发现在用户脚本文件的 bug，来辅助我们进行脚本编程。首先在 main.py 写一个 bug，然后保存好带有 bug 的脚本 mian.py 文件，点击软启动就可以显示 bug 信息了。先故意做一个 bug 文件。



```
G:\main.py - Sublime Te...
File Edit Selection Find View Goto Tools
Project Preferences Help
main.py
1 from pyb import LED
2
3 led = LED(4)
4 led.toggle()
Line 4, Column 13
```

我们故意的将 led 写成 leD 看看进行 REPL 交互调试会怎么样。



```
源地工作室pyboard-tool
文件(F) 编辑(E) 操作 工具 帮助
强制REPL调试 软启动
REPL端口设置
检测端口 端口选择: COM1829 波特率: 115200 断开
接收区
MPY: sync filesystems
MPY: soft reboot
Traceback (most recent call last):
  File "main.py", line 4, in <module>
NameError: name 'leD' isn't defined
uPython v1.11 on 2020-12-09; VCC-GND (R) YD-XDJOS20201208 with YD-MCU
Type "help()" for more information.
>>>
Hex接收 清除
发送区
Hex发送 清除 定时发送 1000 ms/次 发送
USB Serial Device (COM1829) 已连接 接收数: 1187 发送数: 73
```

源地 PYBminiDB 学习板

嵌入式 STM32-python 学习教程

VCC-GND.COM

REPL 交互调试会显示第 4 行中的 `leD` 没有定义。你学会了用 REPL 交互排除脚本错误了吗？还是很 NICE 的。