

# ArControl 使用指南

---

- 作者: 陈昕枫, 华中科技大学, 武汉光电国家实验室
- 邮箱 : [chenxinfeng@hust.edu.cn](mailto:chenxinfeng@hust.edu.cn)
- Copyright (C) 2017, 华中科技大学. GNU LGPL v2.1.
- 源代码下载: <https://github.com/chenxinfeng4/ArControl>
- 可执行文件下载: <https://github.com/chenxinfeng4/ArControl/releases>
- PCB草图下载: <https://github.com/chenxinfeng4/ArControl/releases>
- 该工作正在Frontier审稿中

涉及到的三方源代码

- QFirmata: <https://github.com/firmata/protocol>
- SCPP\_ASSERT from Vladimir Kushnir

## 中文版特有前言

---

不管是说明文档, 还是软件界面, 我都会努力给份中文版本, 方便国人使用。

## 介绍

---

### 什么是ArControl

神经科学研究需要往往需要涉及动物行为学实验, 而这个实验平台 (又称作 **Skinner box**) 在商业上价格昂贵。作者的ArControl是一套建立在Arduino (当前只支持UNO) 上的动物行为学平台, 它可以控制硬件给予动物刺激, 检测动物的行为并做出反应。ArControl 不仅仅便宜又强大, 可以替代商业系统。而且因为系统设计具备实时特性, ArControl 甚至具有商业系统都难以媲美的高时间精准度。

ArControl系统已经通过了相关的验证工作, 包括硬性参数指标和实际的动物行为学实验验证, 可以安心使用。

ArControl具有以下特点:

- 全方面 - 它包含硬件设计和软件设计, 行为学任务设计和实验数据采集。
- 便宜 - 不需要特定的或昂贵的硬件
- 普适性 - 适用于多种行为学任务
- 使用简单 - 行为学任务的设计可以借助强大的“状态机”模式分解, 并通过我的GUI来实现。这些操作都是在图形界面下完成, 设计者不需要为之手动编写任何代码。
- 实时操作系统 -- 编写的任务文件最终会自动翻译成原生态 Arduino 脚本“ino”。任务是直接运行在 Arduino 板上, 不需要与电脑交互就完整地被执行。所以它具有极高的时间精度(<1ms), 并且不受电脑负载的影响。

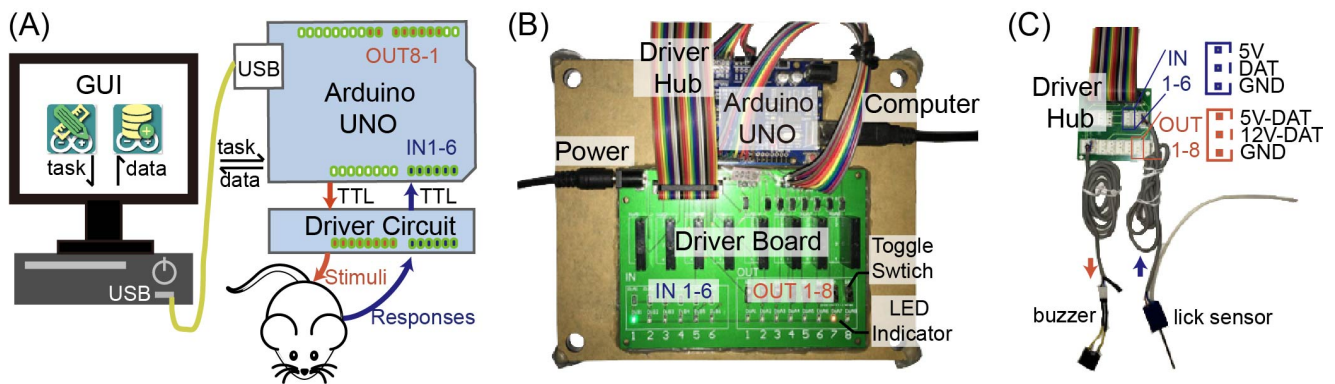


Figure 1

图一 ArControl的连接图示和硬件电子电路。(A) 电路的总览。Arduino板在其中起到了核心的作用。它通过数字引脚检测动物的响应、输出刺激信号，并把这些数据传回电脑再记录下载。ArControl利用定义了6个输入通道和8个输出通道。(B) ArControl硬件：Arduino UNO板和做电压转换的驱动电路。(C) 连接驱动电路的接线器，终端电器在这里插入。传感器(输入电器)设定用5V驱动，刺激器(输出电器)设定可用 5V/12V 驱动。

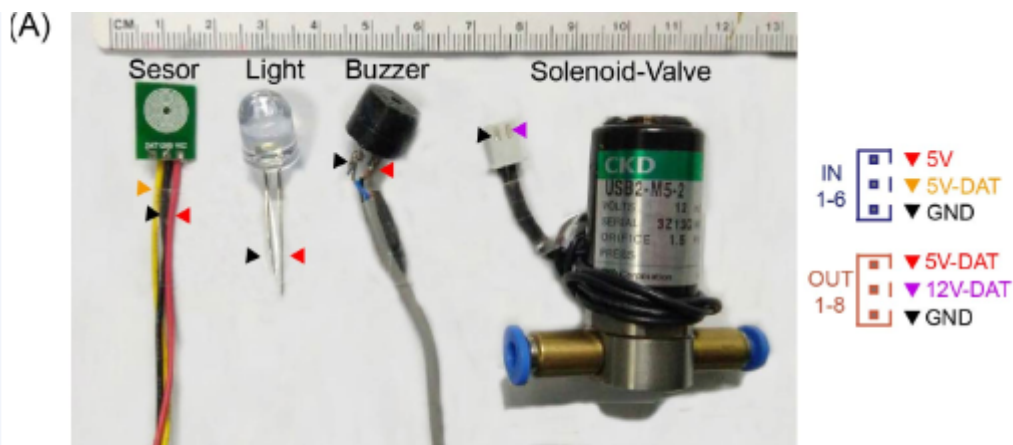
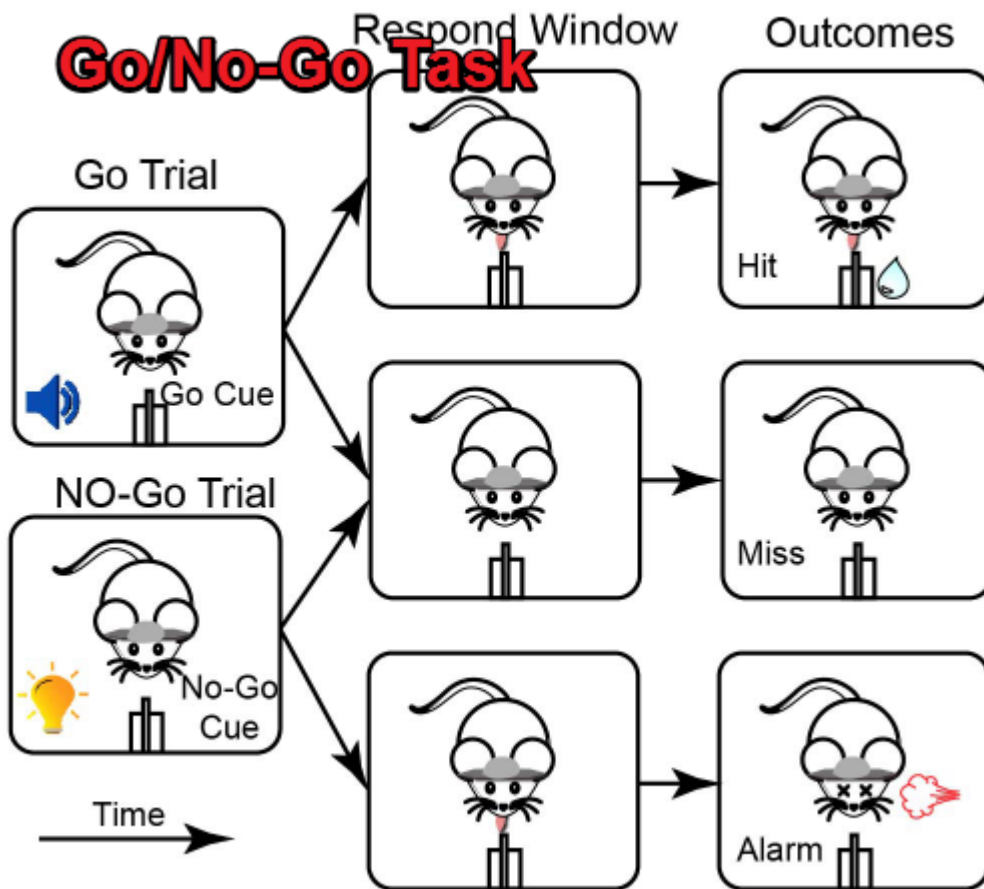


Figure S2

附图 小鼠行为学常用的终端电器。"Sesor" 传感器（接输入通道）用来检测小鼠的 "舔" 行为; LED灯和扬声器用于给予条件刺激(接输出通道)；电磁阀用于发放水滴奖赏或吹气惩罚(接输出通道)。

## ArControl利用“状态机”进行编程

受到商业动物行为学系统 Graphic State Notation 启发，我们成功的将“状态机”理论移植到了ArControl平台上 (Figure 2a, b)。一个时序任务 (既行为学任务、Session) 可以通过一系列的状态机表示。每个状态机都指定“完成什么样的任务，接受什么条件，通过改变自身内在状态来做出反馈”。也就是“do-function, when-function, state-transition” (Figure 2a)。例如在Go/No-GO辨别学习中，可以指定状态来完成“延时”，“给予开始信号”，“检测时间窗范围内是否有行为响应”，“有行为响应而给予奖赏刺激”，或“没有做出响应不给予任何刺激”等等。仅仅修改状态机中的某些参数，这些看似复杂的程式都可以被状态机建模 (Figure 2c)。通过状态机之间的跳转，Go/No-GO”时序可以被完整的表示出来。进一步为了归纳整理，我们可把一系列紧密联系的状态机放在一个小组 (Component, 组) 里面。例如“Go-trial”和“NoGo trial”的设计很类似，可以分别创建这两个组，方便管理。注意，任意一个时间尤且仅有一个状态是处于激活状态的，状态之间会不断的跳转直到程序的结束（到达设定的出口状态机，或被暴力强制退出）。从编程上讲，程序的第一个状态机是 C1S1，每个组 (Cx) 的入口都是该组的第一个状态机 (CxS1)，C0或CxS0有特殊含义 表示退出程序。



示例 “Go/No-Go” 任务

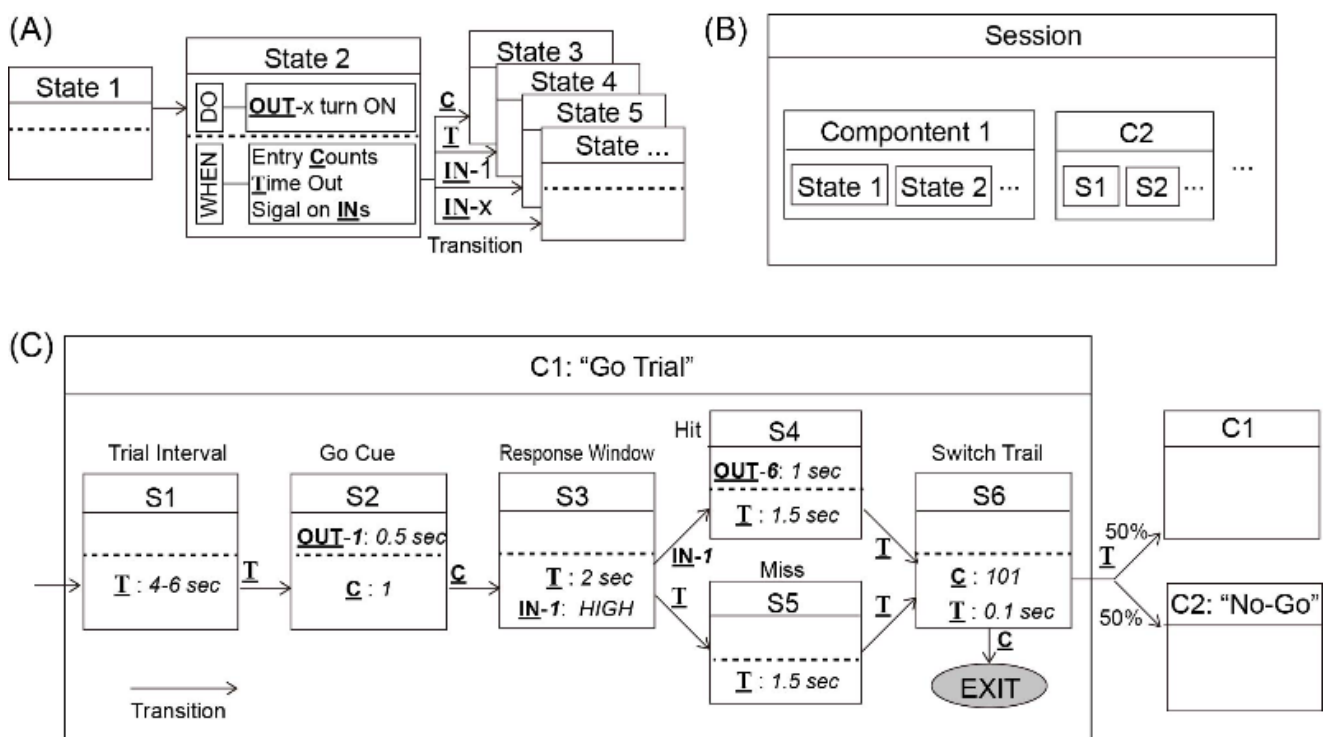
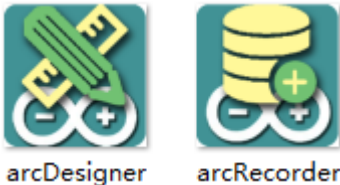


Figure 2

图2 图示ArControl使用的状态机语言。(A)每个状态机都是一个实体,拥有系统的“do-function”用来给予输出信号和“when-function”来控制状态机之间的转化。(B)一个时序任务(既行为学任务、Session)和次级的Component都是状态机集合的虚拟概念。(C)“Go/No-Go”辨别学习的“Go-trial”可以利用状态机概念建模。

## ArControl包含两个独立软件

ArControl软件包含两个独立部分 -- ArControl设计师(Figure 3a) 和 ArControl记录器(Figure S1a)。它们的用途顾名思义。设计师用来实现状态机的建模,记录器用来运行状态机并记录状态机产生的数据。



设计师中可以使用用户自定义的“全局变量”来存储重要的配置参数,比如,输出端口点亮的时长等等。另外,这些变量可以在运行时被动态的更改(参照下面的do-var, when-var),因此可以作为状态机之前“交流”的信号因子。相比与之前的“状态机”讲解中提到的,设计师还为do-function和when-function分别追加了do-var 和 when-var。这是为了方便用户以手动写脚本的方式定制输入或转化。你可以在里面使用Arduino原生态的功能,比如PWM、伺服电机、中断。但是作者非常不期望你这样用。一方面这些函数会影响执行速度,另一方面甚至会扰乱状态机的时钟、扰乱其它端口的状态。作者建议do-var 和 when-var 仅用于改变和检测“全局变量”。

最重要的是,设计师本身不运行状态机,它只是在进行配置(建模)界面。设计师会把设计好的该时序任务,翻译成Arduino 原生态的ino执行文件。真正的状态机群是直接完整存储和运行在Arduino板上的,这省去了运行时与电脑通讯(指令传输)。所以它是一个实时系统!这也使得这个系统具有极高的时间精度(<1ms),并且不受电脑负载的影响。

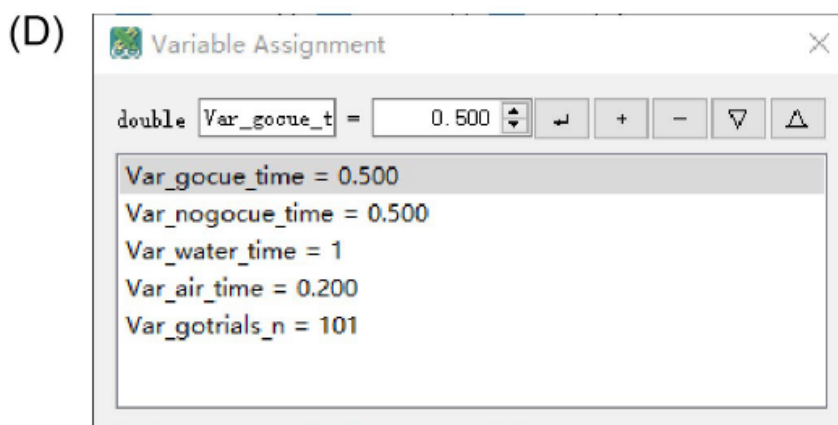
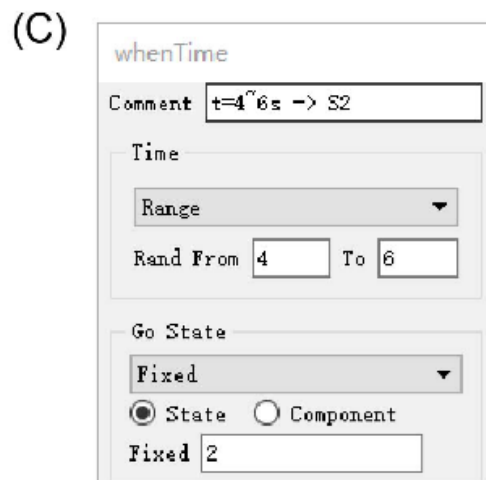
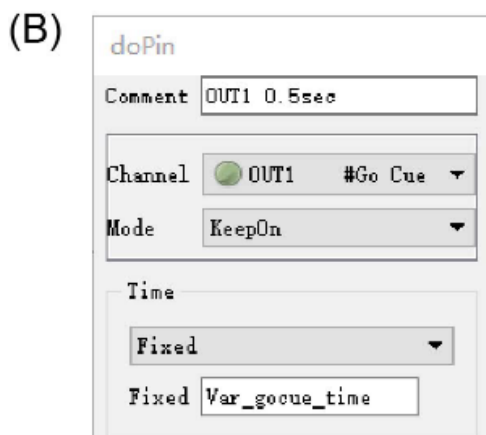
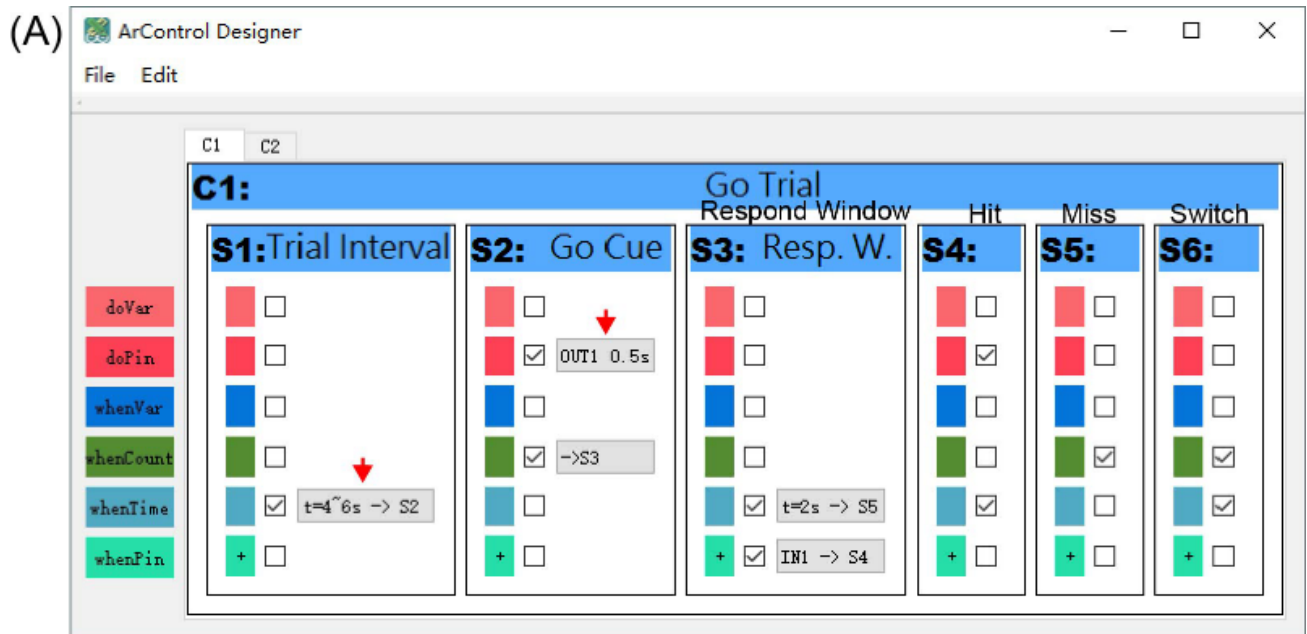


Figure 3

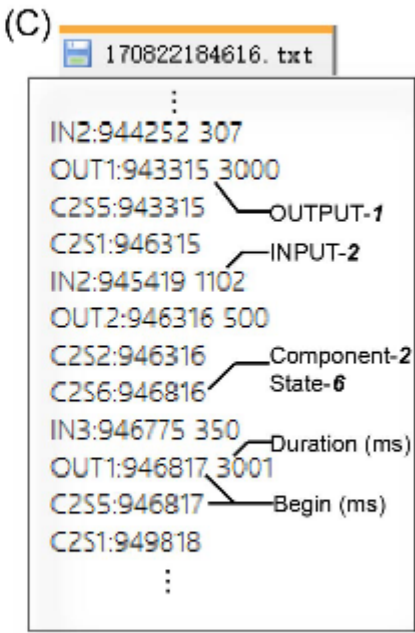
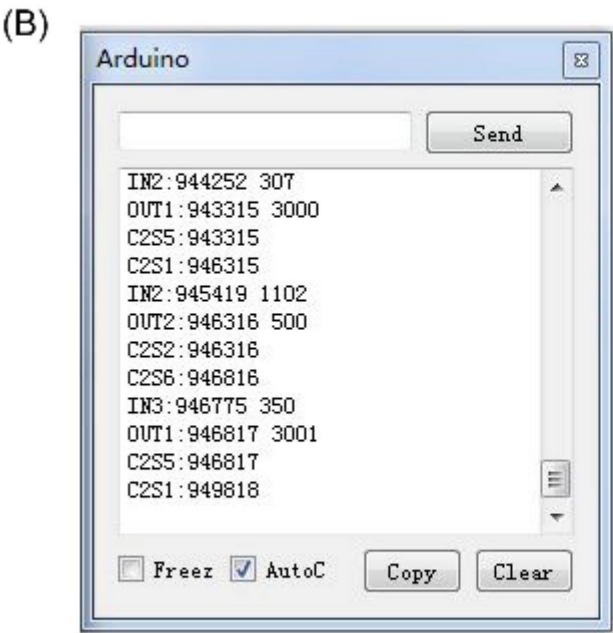
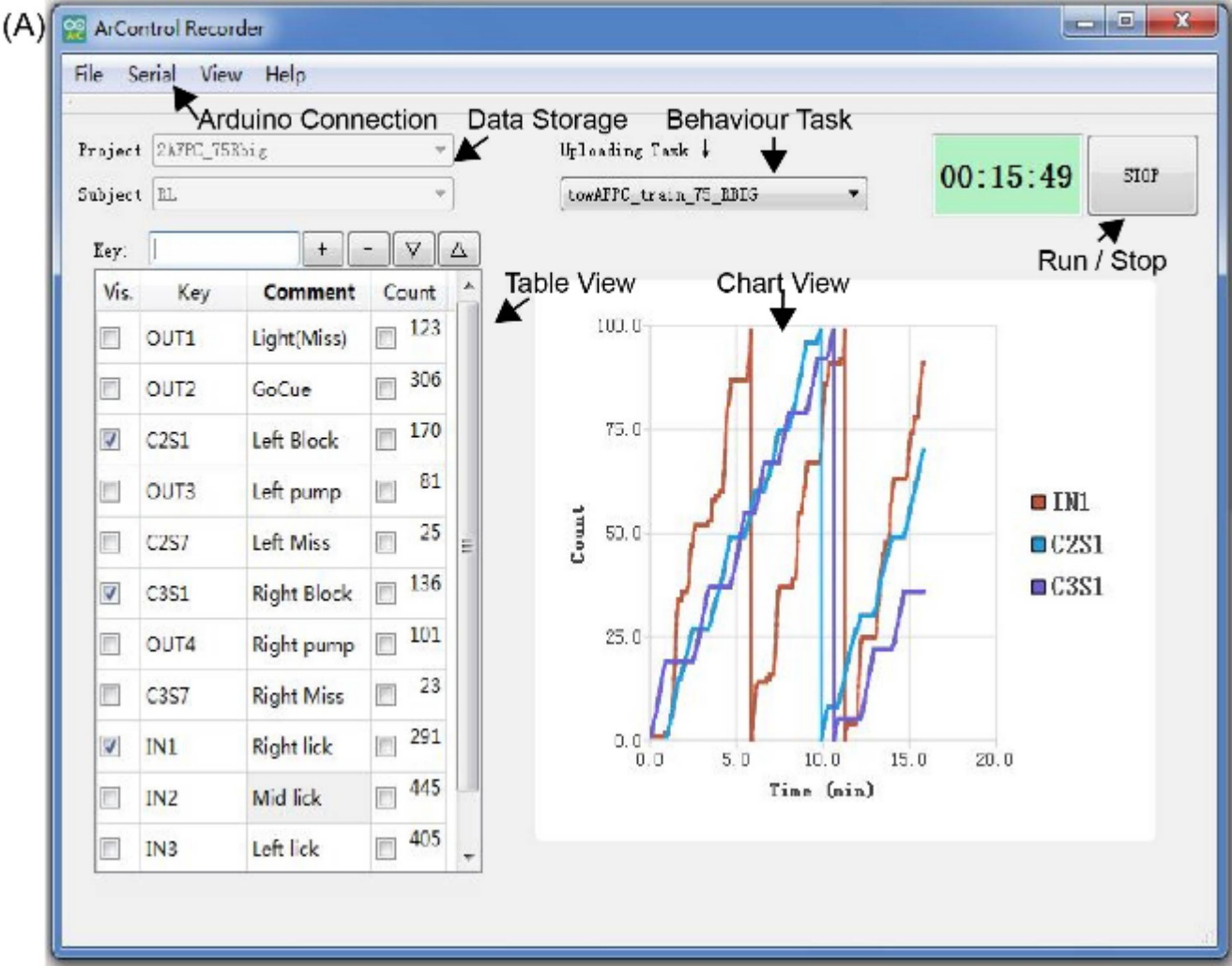
图 3. 利用ArContro设计师来构建一个Session (任务). (A) ArControl设计师的主窗口。它的每个状态机都支持 2种 *do-function* 和 4种 *when-function*。图中的示例与 图2c 一致。(B, C) 典型的弹出窗口来配置 *do-function* \* 和 *\*when-function*. (D) 全局变量添加和初始化。这些全局变量可以用来设定状态机的参数，并在状态机之间共享。





记录器的功能相对单一。只是开始/停止运行；记录运行时产生的实验数据；并把实时数据以表和图的形式显示出来。

此外记录器还提供“Firmata”功能，用于直接调试硬件--查看输入和点亮输出端口。



## Figure S1

副图 1. ArControl记录器和数据采集. (A) ArControl记录器的界面。(B) ArControl记录器的串口监视器。(c) 同步生成的数据文件，记录了包含 状态机/输入通道/输出通道 的状态更迭。

## 安装ArControl

---

### 下载并安装 Arduino IDE

ArControl依赖于Arduino-IDE。所以你应该下载并安装Arduino-IDE，[官方地址](#)。

- ArControl是在[Arduino IDE 1.6.11](#) 上测试通过的. 理论上是兼容之后版本的IDE，但是 1.6.12 除外.
- 推荐你下载 "Windows Installer" 版本.



### 下载并安装ArControl

一般意义上，作者推荐你下载 [可执行版](#) . ArControl当前是在 Microsoft Windows PC上测试的. 你需要重新编译 [源代码](#)，如果你想要linux，Mac版.

- 如果你想要可执行版，只需下载并解压它。
- 仅当你要编译源代码时，你需要下载 [Qt5.7](#).
- ArControl会自动搜寻 "Arduino IDE". 请确保Arduino-IDE已经事先装好，并首先运行ArControl的 arcDesigner.exe.



## 你的第一个编程 -- "Light3"

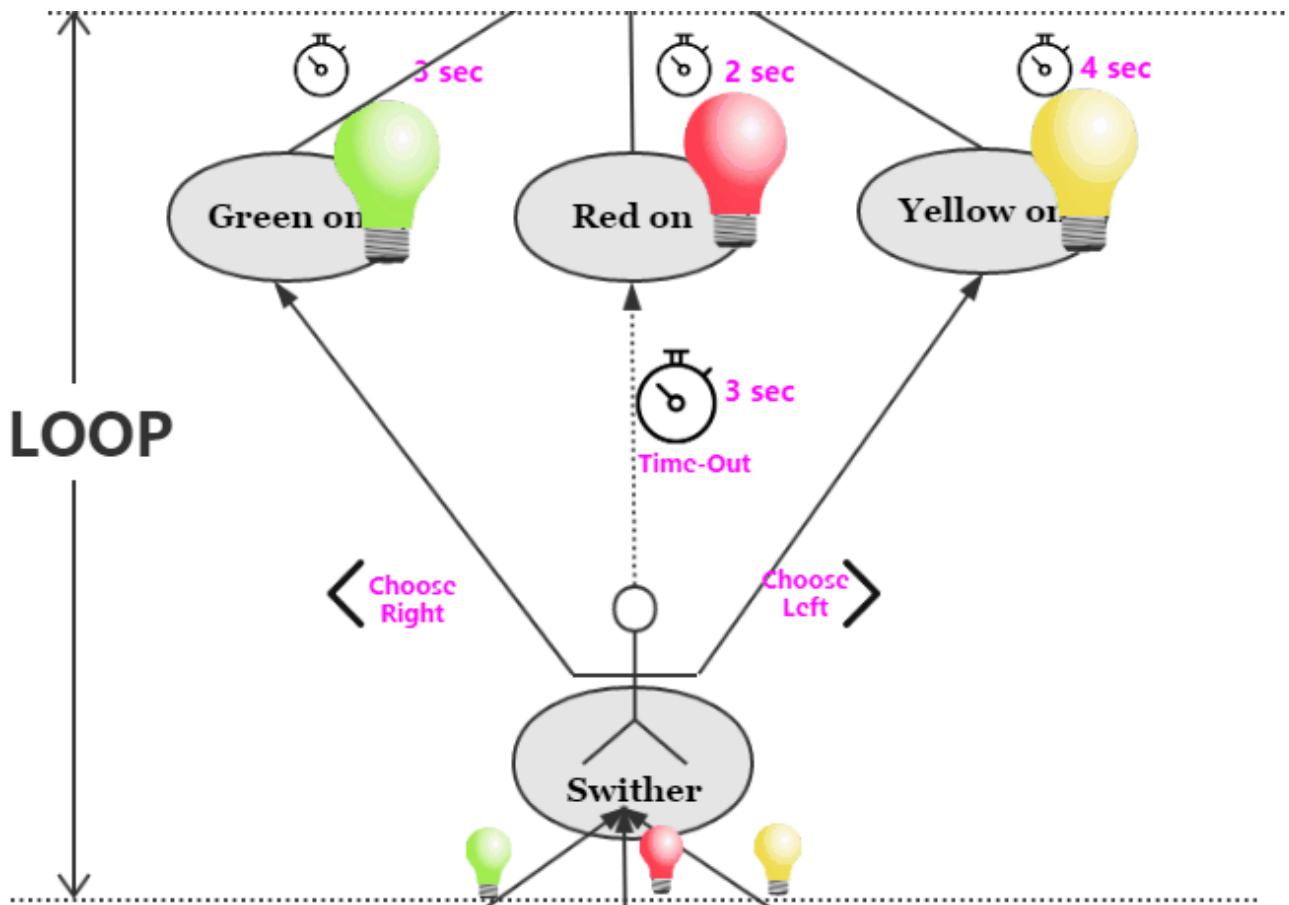
---

### 编写你的一个时序任务 -- "Light3"



在生活中，做出不同的选择往往导致不同的结果，连“静观其变”--等待其实也是选择的一种。我们第一个时序任务就是在ArControl中实现这种生活现象（下图）。

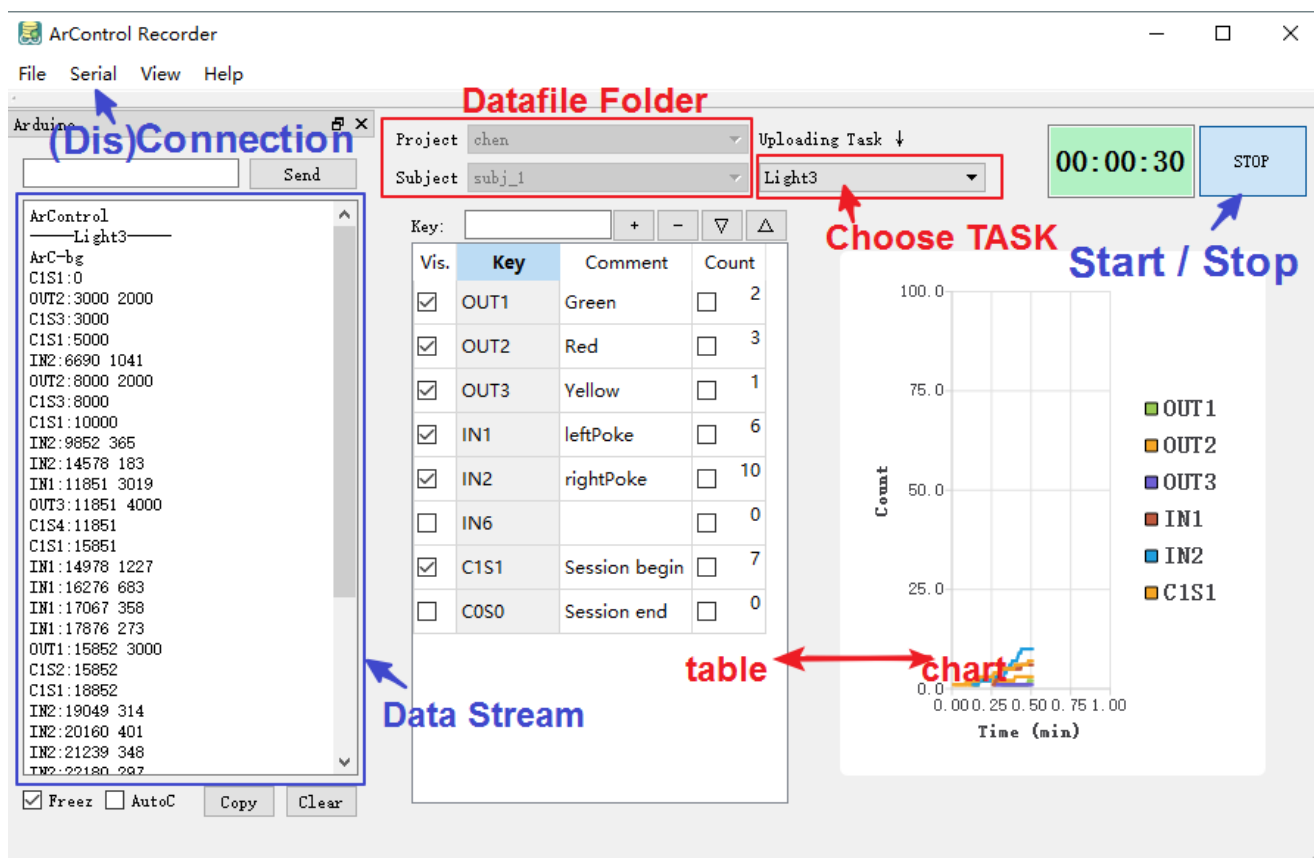
即使是老练的编程人员至少也需要半个小时，在Arduino平台上来完成这个时序操作。但是，ArControl能让即使新手都能快速完成编写。



Light3 任务的时序安排。根据你选左/右/不选, 你将得到各自的反馈信号 (例如, 三种不同颜色的灯光)。

这个时序任务可以被ArControl设计师编写如下图。这个演示代码已经在可执行版的文件中, 你可以通过 `ArControl1 设计师 > 文件 > 打开 > Light3`。





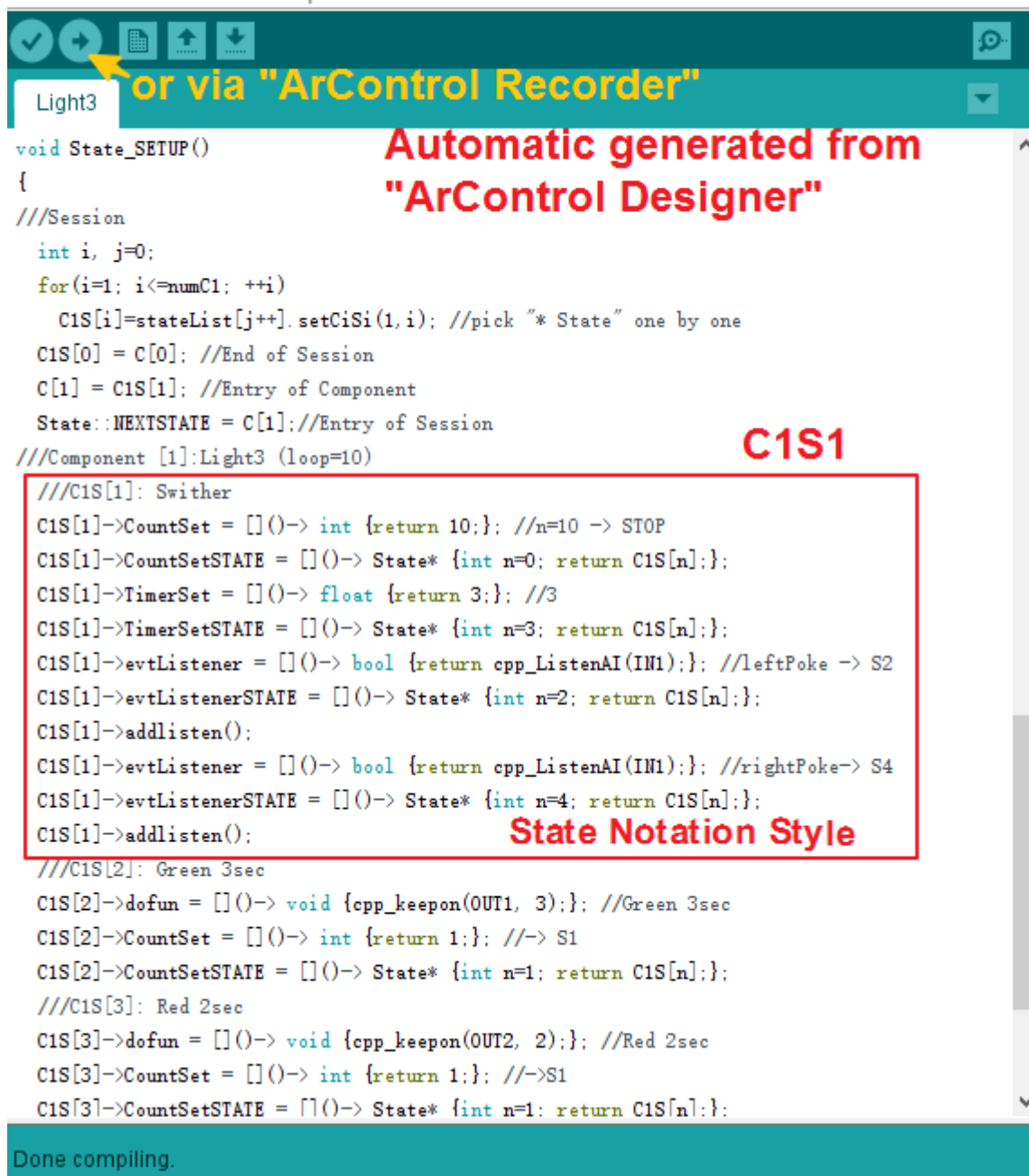
用ArControl记录器执行一个 Light3 任务。数据流在左侧框图显示，同时又以表格(中间框图) 和 图形(右侧框图) 展示。数据最后将保存在指定的 "Datafile Folder" 文件夹中。

## 深入观察 -- "Light3"

ArControl设计师 将配置 Light3 的界面元素保存在了 *Light3.acnf* 文件里, 并且自动将它的状态机建模完整地翻译成Arduino可执行的 *Light3.ino* 脚本。

ARCONTROL_PATH/task/Light3			
名称	修改日期	类型	大小
Light3.acnf	2017/10/7 星期...	ACONF 文件	10 KB
Light3.ino	2017/10/7 星期...	INO 文件	4 KB

**Designer Outputs**



Light3

or via "ArControl Recorder"

Automatic generated from "ArControl Designer"

C1S1

```

void State_SETUP()
{
  ///Session
  int i, j=0;
  for(i=1; i<=numC1; ++i)
    C1S[i]=stateList[j++].setCiSi(1,i); //pick "* State" one by one
  C1S[0] = C[0]; //End of Session
  C[1] = C1S[1]; //Entry of Component
  State::NEXTSTATE = C[1]; //Entry of Session
  ///Component [1]:Light3 (loop=10)
  ///C1S[1]: Swither
  C1S[1]->CountSet = []()-> int {return 10;}; //n=10 -> STOP
  C1S[1]->CountSetSTATE = []()-> State* {int n=0; return C1S[n];};
  C1S[1]->TimerSet = []()-> float {return 3;}; //3
  C1S[1]->TimerSetSTATE = []()-> State* {int n=3; return C1S[n];};
  C1S[1]->evtListener = []()-> bool {return cpp_ListenAI(IN1);}; //leftPoke -> S2
  C1S[1]->evtListenerSTATE = []()-> State* {int n=2; return C1S[n];};
  C1S[1]->addlisten();
  C1S[1]->evtListener = []()-> bool {return cpp_ListenAI(IN1);}; //rightPoke-> S4
  C1S[1]->evtListenerSTATE = []()-> State* {int n=4; return C1S[n];};
  C1S[1]->addlisten();
  ///C1S[2]: Green 3sec
  C1S[2]->dofun = []()-> void {cpp_keepon(OUT1, 3);}; //Green 3sec
  C1S[2]->CountSet = []()-> int {return 1;}; //-> S1
  C1S[2]->CountSetSTATE = []()-> State* {int n=1; return C1S[n];};
  ///C1S[3]: Red 2sec
  C1S[3]->dofun = []()-> void {cpp_keepon(OUT2, 2);}; //Red 2sec
  C1S[3]->CountSet = []()-> int {return 1;}; //->S1
  C1S[3]->CountSetSTATE = []()-> State* {int n=1; return C1S[n];};
  
```

Done compiling.

State Notation Style

ArControl 记录器 可以自动的采集数据。你可以用作者附带的 Matlab 函数 `BF_arc2mat.m`，将该数据文件从 TXT 转化为 MAT 格式。

ARCONTROL_PATH/data/chen/subj_1			
名称	修改日期	类型	大小
171007212558.txt	2017/10/7 星期...	文本文档	1 KB

Datafile

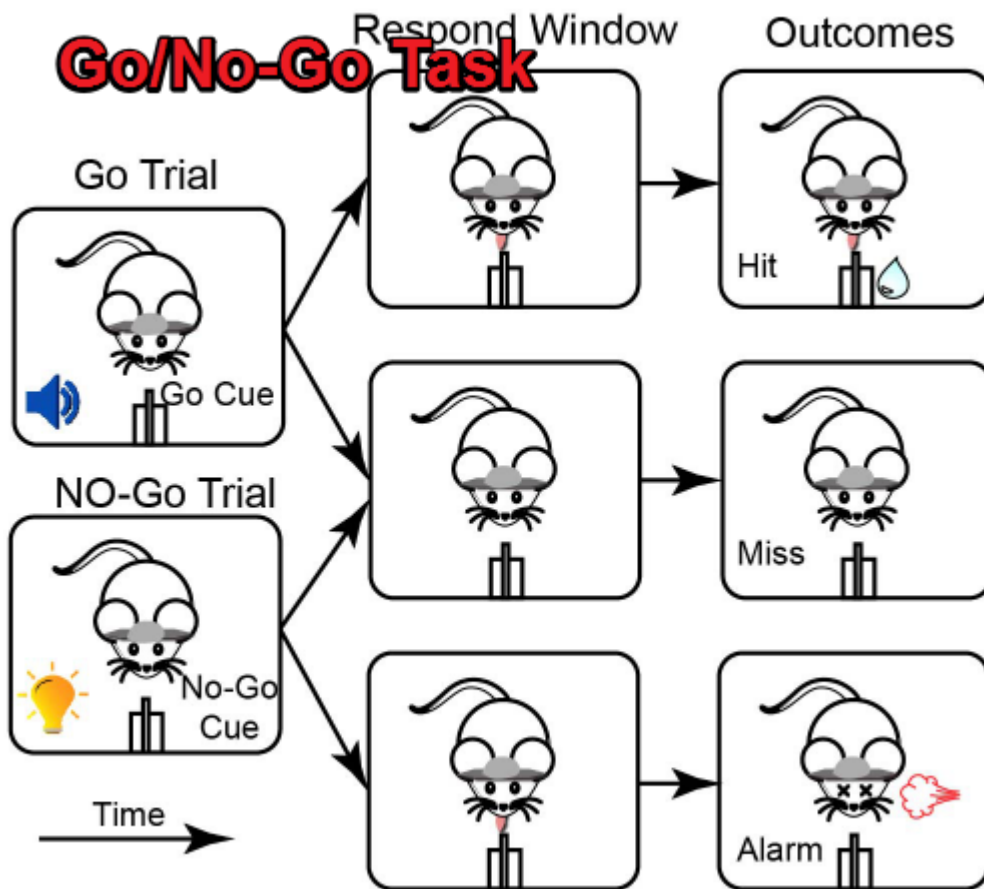
```
171007212558.txt
1  ArC-bg
2  C1S1:0
3  OUT2:3000·2000
4  C1S3:3000
5  C1S1:5000
6  IN2:6690·1041
7  OUT2:8000·2000
8  C1S3:8000
9  C1S1:10000
10 IN2:9852·365
11 IN2:14578·183
12 IN1:11851·3019
13 OUT3:11851·4000
14 C1S4:11851
15 C1S1:15851
16 IN1:14978·1227
17 IN1:16276·683
18 IN1:17067·358
19 IN1:17876·273
```

## 其它的演示 -- "Go\_NoGo" / "2AFPC" 任务

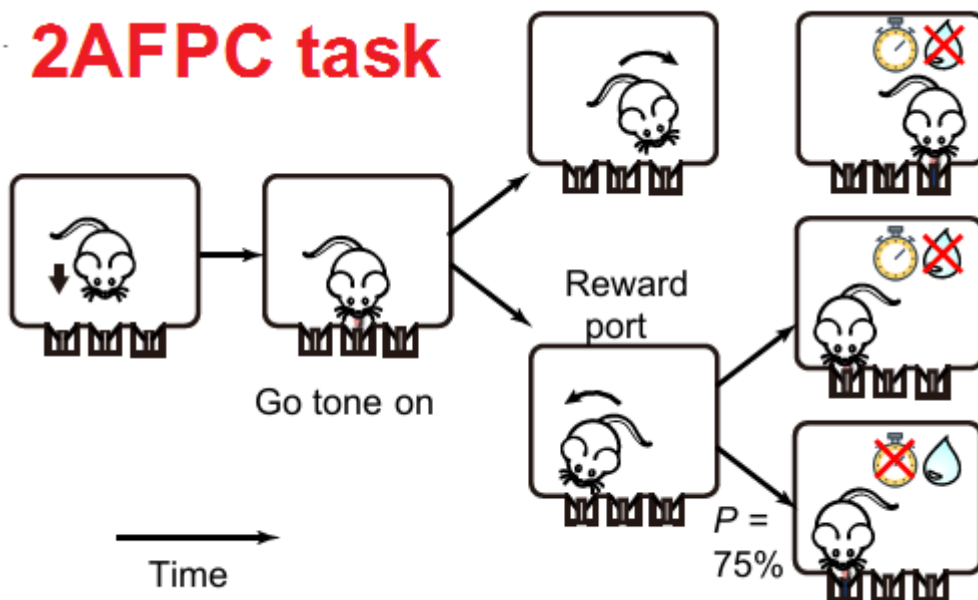
---

两种常见的行为学任务-- Go/No-Go 任务 (Gomez et al., 2007; Dolzani et al., 2013) 和 双向选择任务 (Tai et al., 2012; Stephenson-Jones et al., 2016), 它们已经广泛被用来研究辨别过程和记忆过程。作者在这里展示如何利用 ArControl 来实现这些逻辑。

这两个时序任务的展示如下。这个演示代码已经在可执行版的文件中, 你可以查看 [ArControl 设计师 > 文件 > 打开 > Go\\_NoGo](#) 和 [ArControl 设计师 > 文件 > 打开 > twoAFPC](#)。



在 Go/NoGo 任务中, 头固定喝水小鼠 (Guo et al., 2014) 需要去辨别 go 信号 (声音) 和 no-go 信号 (灯光)。根据小鼠在反应时间窗里的行为 (对 go 信号响应 / 对 no-go 信号响应 / 不响应) 来进行相应的输出 (水滴奖赏 / 吹气惩罚 / 不输出)。



在 2AFPC 任务中, 自由活动的小鼠需要通过舔中间位点来触发 trial 的开始, 随后移动到左边或右边的位点来尝试获得水滴奖赏。一侧位点的奖赏率设定在75%, 另外25%和另一侧将不予于奖赏。如果没有奖赏, 就会有一个延迟作为惩罚。奖赏所在的位点会周期性的交替。



# ArControl软件常见问题的解决方案

## 记录器 Error: 上载任务代码到Arduino板的界面出现乱码

你应该把Arduino IDE 换成英文语言. 打开 **Arduino IDE > 文件> 选项> 语言** , 选择 **English (English)** 。

## 记录器 Error: 上载任务代码到Arduino板时闪退

确保你已经安装了 Arduino IDE, 并且这个路径在ArControl上被正确设定。打开 **ArControl 设计师> 文件> 选项> Arduino IDE 路径** , 在Arduino IDE的安装目录下选择 **arduino\_debugger.exe** (for Windows).

## 记录器 / 设计师 Error: 程序刚打开就闪退

很可能是 **ArControl** 被错误的配置了。你可以重置该软件。删除 **default.bconf** (如果存在), **task\defalut\** (如果存在) 和 **profile.xml** (如果存在)。

## 记录器 Error: 上载任务代码到Arduino板的界面提示"Time Out" 错误

这对于刚刚创建的任务来说很常见。再试一遍, 或利用 **ArControl IDE** 来上载代码。

## 记录器 Error: 不能连接到Arduino板

很有可能是Arduino板当前正在被其它软件使用 (例如, **Arduino IDE > 串口监视器**)。从软件中弹出该Arduino板, 或者重新拔插USB连接线。

## 记录器 Error: 上载任务代码到Arduino板的界面提示"fatal error: xxxxx/ArControl\_AllinOne.h: No such file or directory"

打开 *mytask.ino*, 找到行 **#include "xxxxxx/ArControl\_AllinOne.h"** . 这个路径可能已经失效。手动地更新该路径, 或者利用 **ArControl 设计师** 的保存操作来自动更新该路径。

## 如何: 选择 ArControl 的语言

ArControl now merely supports **简体中文** and **English** languages. Swithing language by -- **ArControl Designer > File > Profile > Language** , **ArControl设计师 > 文件 > 选项 > 语言** .

## 如何: 设置 "whenPin" 里的 INx 是高/低电平触发.

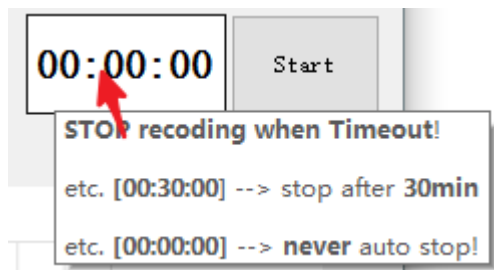
大多数的传感器组件 (INx) 都是用低电平表示休息, 高电平表示活跃。但是有些恰好相反。打开 **Arcontrol 设计师> 文件> 选项 > 输入信号** , 选择合适的类型。当心, 这个操作对全体的 INx 生效。

## 如何: 跳过“开始”按钮直接开始任务执行, 或利用硬件的触发信号来开始任务执行

任务的开始执行往往是通过 **ArControl 记录器> 开始** 按钮。但是也有其它的开始方式。打开 **ArControl 设计师> 文件> 选项> "开始运行"方式** , 你可以选择 **马上开始** , **触发由: 硬件 (引脚D11 高电平)** , **触发由: 软件 (串口通讯)** - 默认, **触发由: 硬件或软件** 。

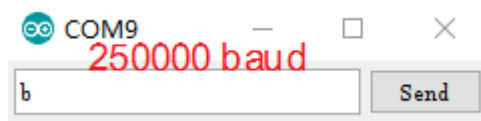
## 如何: 为结束任务的执行设定倒计时

结束任务通常是从 1. ArControl 记录器 > 结束 按钮; 2. 任务圆满完成, 状态机进入终点 (C0 or CxS0)。此外, 你可用通过 ArControl 记录器 > 时间窗 来设置结束倒计时。



## 如何: 利用Arduino-IDE来运行任务

Arduino IDE 可以代替 ArControl 记录器 来运行任务。打开 Arduino IDE > 串口监视器, 选择 250,000 baud 并发送字符 b。



## 技巧和警告

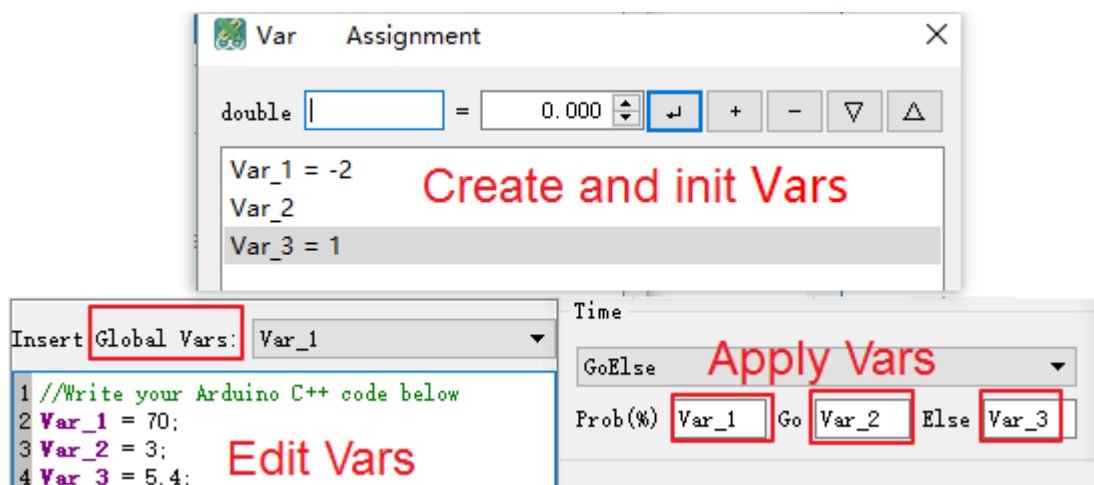
### 警告: 参数值 "0" 的特殊含义

尽量避免使用参数 0。它在 ArControl 设计师 中是有特殊含义的。



### 技巧: 参数值可以被“变量”替代

定义“全局变量”能更好地梳理任务中的重要参数。“全局变量”定义在 Edit > Var Assignment, 并且可以在 doVar 和 whenVar 中动态的更新, 可以被状态机群共享, 可以替代数字参数值。



### 警告: 任务最多支持 20个状态机 (State)

目前ArControl只支持Arduino UNO板。而UNO是入门级的硬件，仅仅拥有 2Kb SRAM , 20 状态机是它理论上承载的上限。作者正在计划适配高级的Arduino板，这将提高State的容量和运行速度。但作者精力有限，这项工作可能被搁置。如果你又有这方面需求，最快的办法就是自己重写 `ArControl PATH > ino` 下的 `.h` 文件，`ArControl` 设计师 和 `ArControl` 记录器 还是可以使用，但必须改用 `Arduino IDE` 而非 `Arduino` 记录器 来烧录任务代码。

## 警告: [INx] 输入通道最多支持200 Hz的输入信号

ArControl 最高能同时负载6个通道的200Hz, 或1个通道的700Hz TTL 信号。并且记录的数据与真实的数据比较后，证明精度可到  $\pm 1\text{ms}$  (95% 置信区间, 由于四舍五入效应)。