

# ArControl Guidance

---

- Author: Xinfeng Chen (陈昕枫)
- Email : [chenxinfeng@hust.edu.cn](mailto:chenxinfeng@hust.edu.cn)
- Copyright (C) 2017, Huazhong University of Science and Technology. GNU LGPL v2.1.
- Source-code download: <https://github.com/chenxinfeng4/ArControl>
- Binary-release download: <https://github.com/chenxinfeng4/ArControl/releases>
- PCB drafts download: <https://github.com/chenxinfeng4/ArControl/releases>
- The work is currently submitting to Frontiers.

## Reference codes

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

## 中文版说明 (Pages for Chinese)

---

请查看 `readme中文.md`。

## Introduction

---

### What's the ArControl

The goal of ArControl is to establish an Arduino-based (UNO only) behavioral platform (as Skinner box), which control devices to deliver stimulation and monitor behavioral response. The ArControl is not merely a cheap and powerful alternative to commercial behavioral platforms. Meanwhile, the ArControl is superior to the commercial systems in that it was a genuine real-time system with high temporal resolution. ArControl is supposed to be a powerful solution for behavioral researches in psychology and neuroscience.

Through both the technical parameters assessment and the practical behavioral experiments in mice, our ArControl platform was proven to be reliable and adaptive within various behavioral tasks.

The basic features of this platform are:

- Comprehensive – it combines software and hardware, behavioral task design and experimental data collection;
- Inexpensive – neither dedicated nor expensive hardware is essential;
- General purpose – it's applicable to multiple behavioral tasks;
- Easy to use – behavior task can be decomposed by the straightforward State Notation concept, and designed via a friendly GUI without need to master script language.
- Real-time performance – it has high temporal resolution (<1ms) and free from the load of computer.,

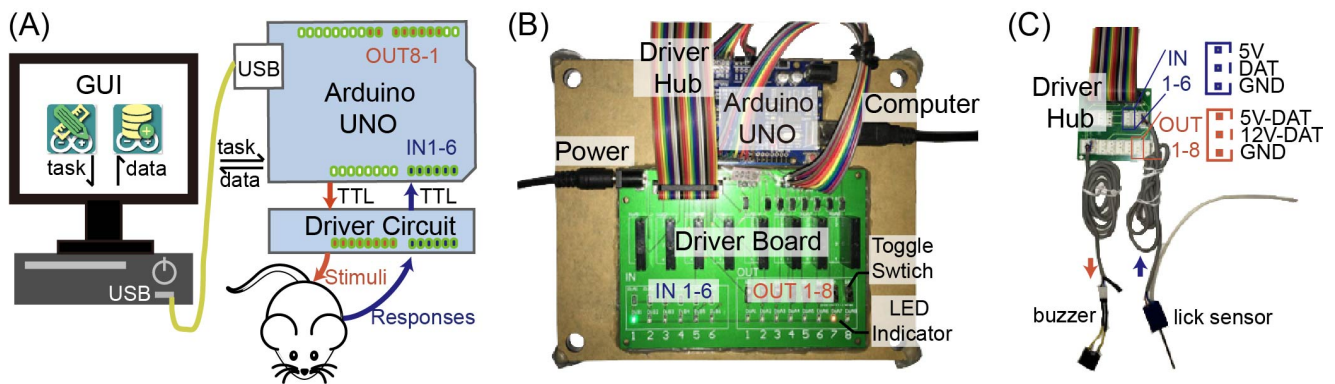


Figure 1

Figure 1. Diagram and hardware of the ArControl's electronic circuit. (A) Overview of the electronic circuit. Arduino is the core controller detects animal response, sends simulations via digital pin, and logs data to the host computer. ArControl supports 6 input and 8 output digital channels. (B) The hardware of ArControl: Arduino Uno R3, the driver circuit for voltage conversion. (C) The driver hub provides slots for terminal devices. Sensors can work at 5V, and stimulators can alternatively work at 5/12V.

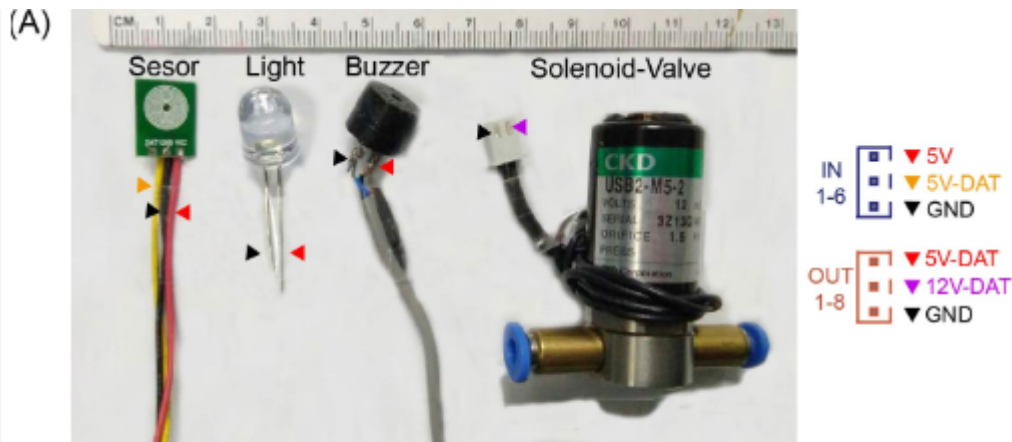


Figure S2

Common instruments for mice behavioural platform. "Sensor" is a captive sensor for detecting mice lick. "Light" is a LED lamp. "Buzzer" produces a buzzing sound as a tone signal. "Solenoid-Valve" control the air-puff and the water-drop.

## ArControl uses State Notation to design tasks

Inspired by Graphic State Notation, we succeeded in grafting State Notation Pattern into ArControl platform (Figure 2a, b). The schedule is constructed at hierarchical levels: Session (top), Component (middle), and State (lowest). The State is the basic structure using State Notion design pattern. The Component and the Session are the primary and the secondary collection of States (Figure 2b). Generally, the terms of Session, schedule, and behavior task are equal concepts indeed. Utilizing the State Notion Pattern, a schedule is dissembled as a serials of States. Each State specifies a stimulus configuration in subject's environment and a set of time and/or response requirements that cause state transition (Figure 2a). There is one and only one State to be active at any moment. Session will sequentially move from current State to next State, until reaching the terminal.

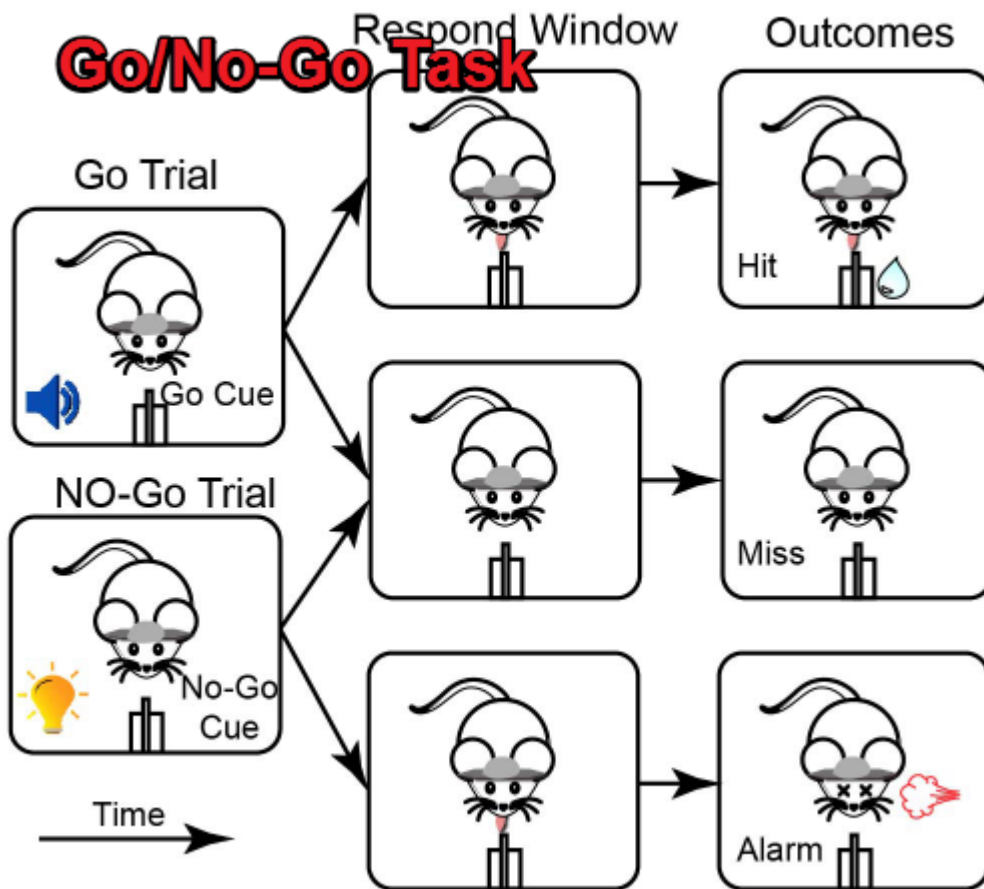


Diagram of Go/No-Go Task

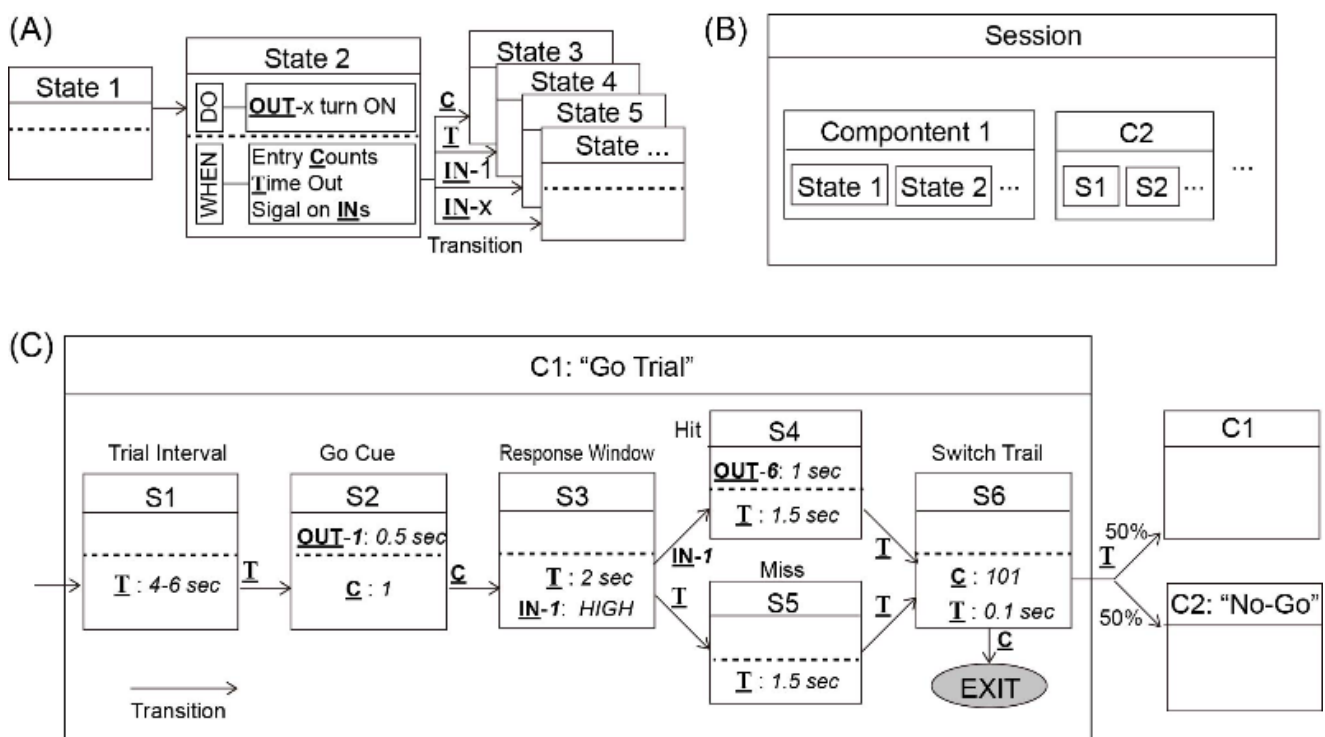


Figure 2

Figure 2 Diagram of State Notation Pattern utilized by ArControl. (A) The State has a *do-function* to deliver stimulus, and *when-function* to master transitions cross States. (B) Component (abbr. C) and Session structures are the primary and the secondary collection of States (abbr. S) . (C) Illustration of a Go Trial in the Go/No-Go task

## ArControl has two parts of GUI

The ArControl software suite consists of two individual parts -- the Designer (Figure 3A) and the Recorder (Figure S1A) -- for the purpose of designing and running a Session respectively. The ArControl Designer is core program, where users apply State Notation concept to achieve the Session construction.

The ArControl Recorder is the other part of ArControl software suite. The basic functions of it contain clicking to start/stop running ready Session, collecting the data flow and displaying as table and chart contents (Figure S1A). Besides, there is a straightforward utility tool plugged in —Firmata— intent to debug the input and output devices.

The two parts can run individually. In the source file, they are named as "arcDesigner.exe" and "arcRecorder.exe"



arcDesigner



arcRecorder

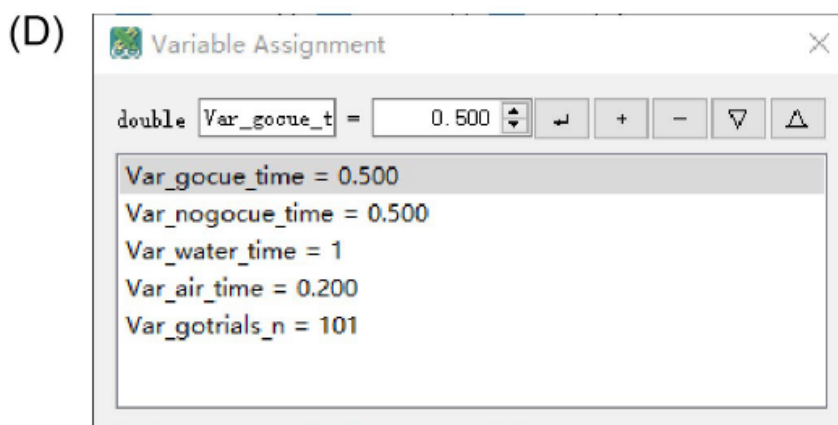
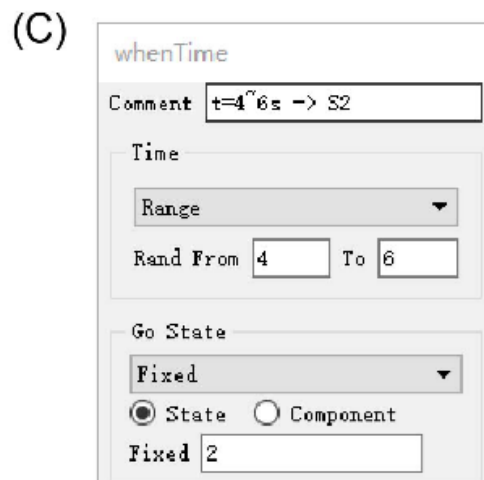
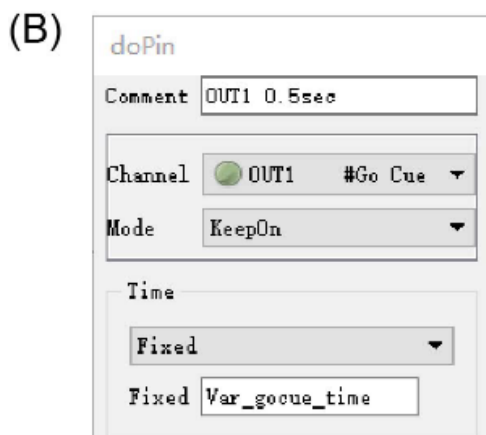


Figure 3

Figure 3. Apply ArControl-Designer to design Session. (A) Main window of the ArControl Designer shows an implementation of the Go/No-Go task. (B, C) Typical pop-up windows to configure a *do-function*(B) and a *when-function*(C). (D) Window defines and initializes the Global Variable.

The basic functions of the ArControl Recorder was clicking to start/stop the running of a ready Session, as well as collecting and displaying the data flow (Figure S1A-C). A useful tool (Firmata) was provided to directly debug the input and the output devices.

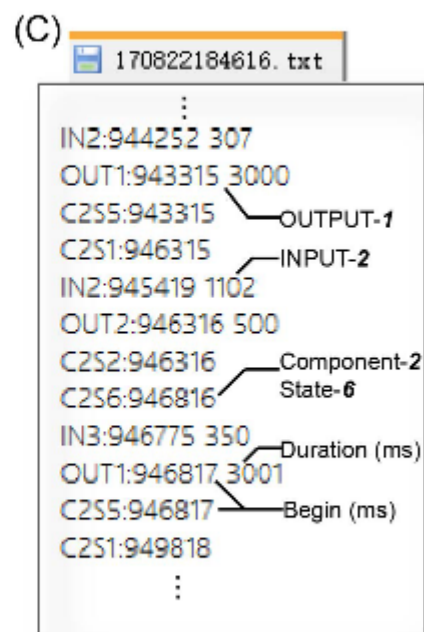
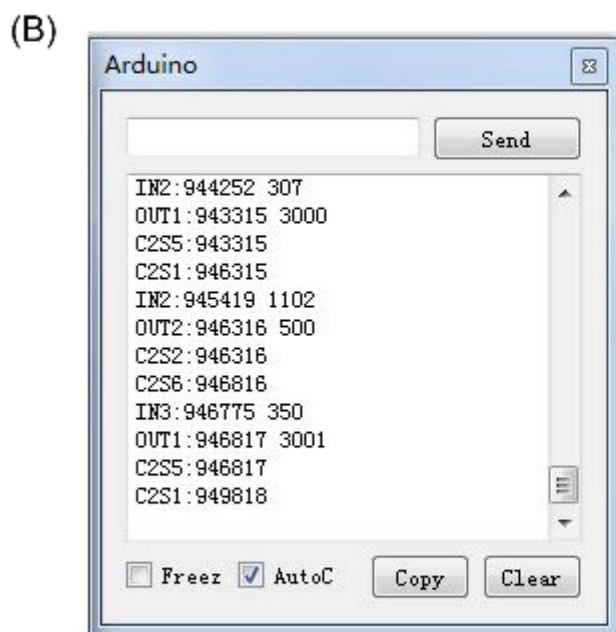
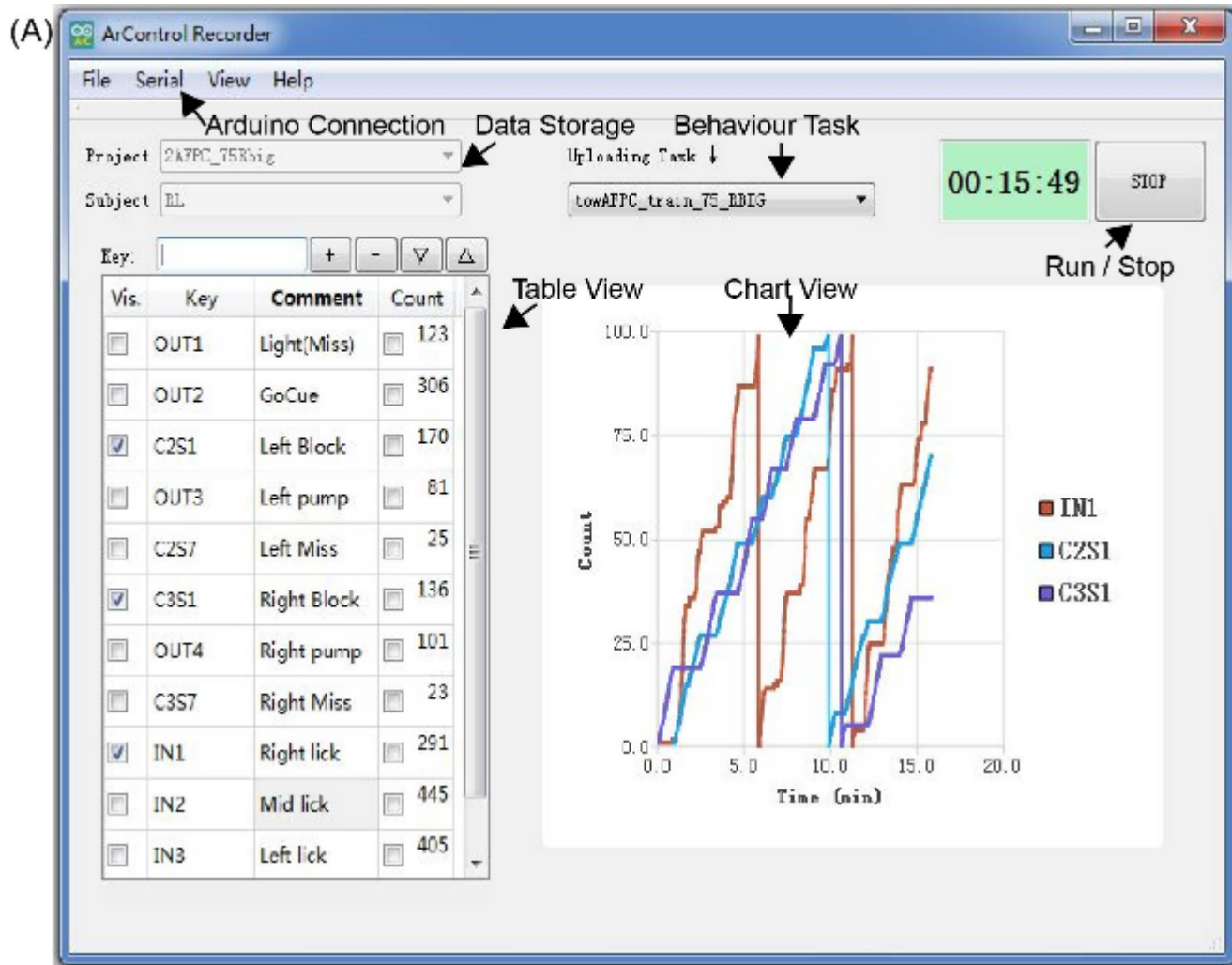


Figure S1



Supplementary Figure 1. The data collection from ArControl. (A) The main window of ArControl-Recorder. (B) The serial monitor window of ArControl-Recorder. Numbers after colon represent the beginning and duration time in milliseconds. (C) A matched segment extracted from the data file.

## Install ArControl

---

### Download and install Arduino IDE

ArControl relays on Arduino-IDE. You should download the IDE from official [site](#).

- The ArControl was verified on [Arduino IDE 1.6.11](#) . In theory, the latest versions are also eligible, except 1.6.12.
- You are recommended to download the "Windows Installer" version.



### Download and install ArControl

As for general purpose, the [binary-release](#) version are recommended. ArControl is only verified on Microsoft Windows PC. You can recompile the [source-code](#) to make adaptations.

- If you want to use binary-release directly, just download and unzip it.
- Onlyif you want to compile source-code, you should download and install [Qt5.7](#).
- ArControl will automatically look for any available "Arduino IDE". Make sure that you have installed the "Arduino IDE".



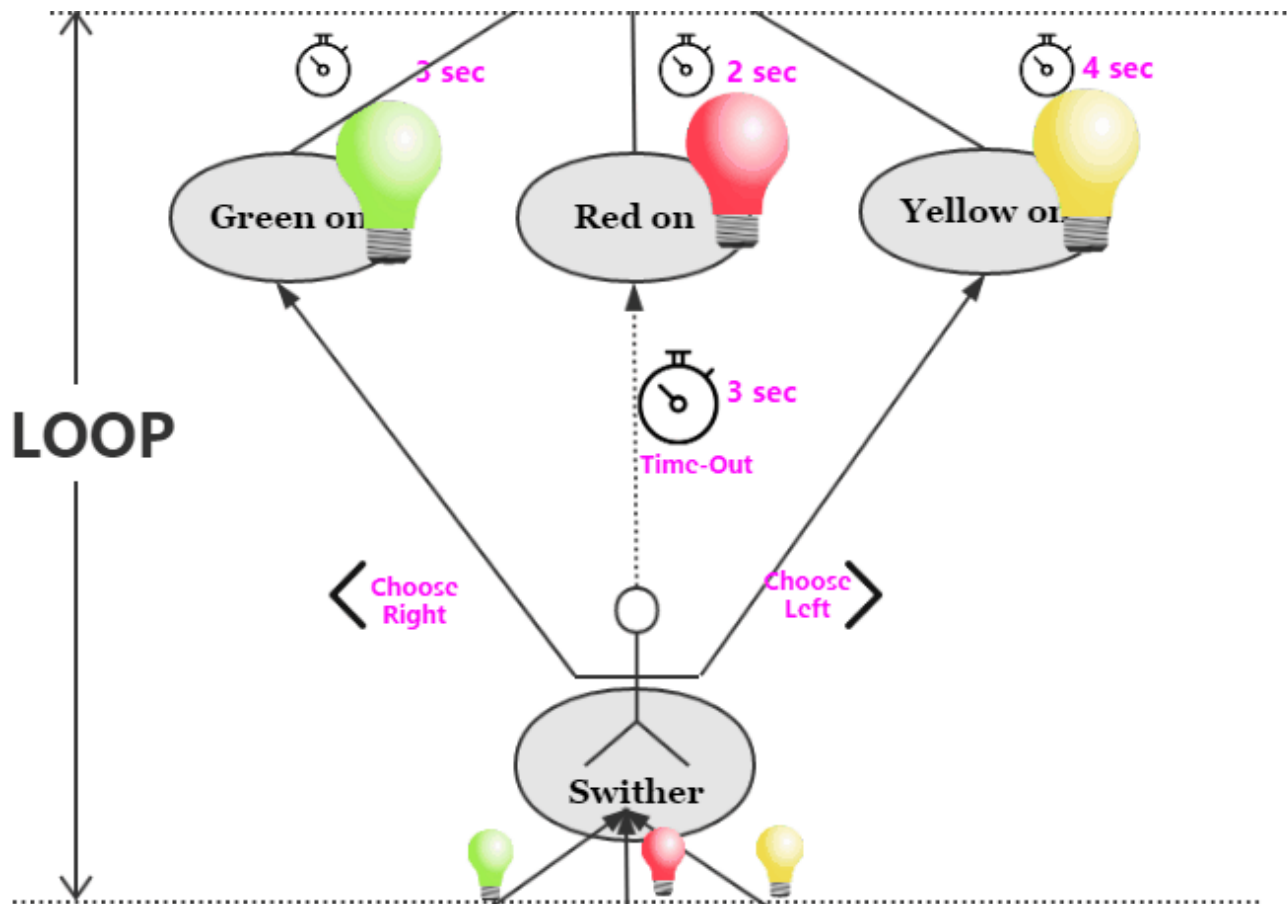
### Your first task -- "Light3"

---

### Compose your first task -- "Light3"

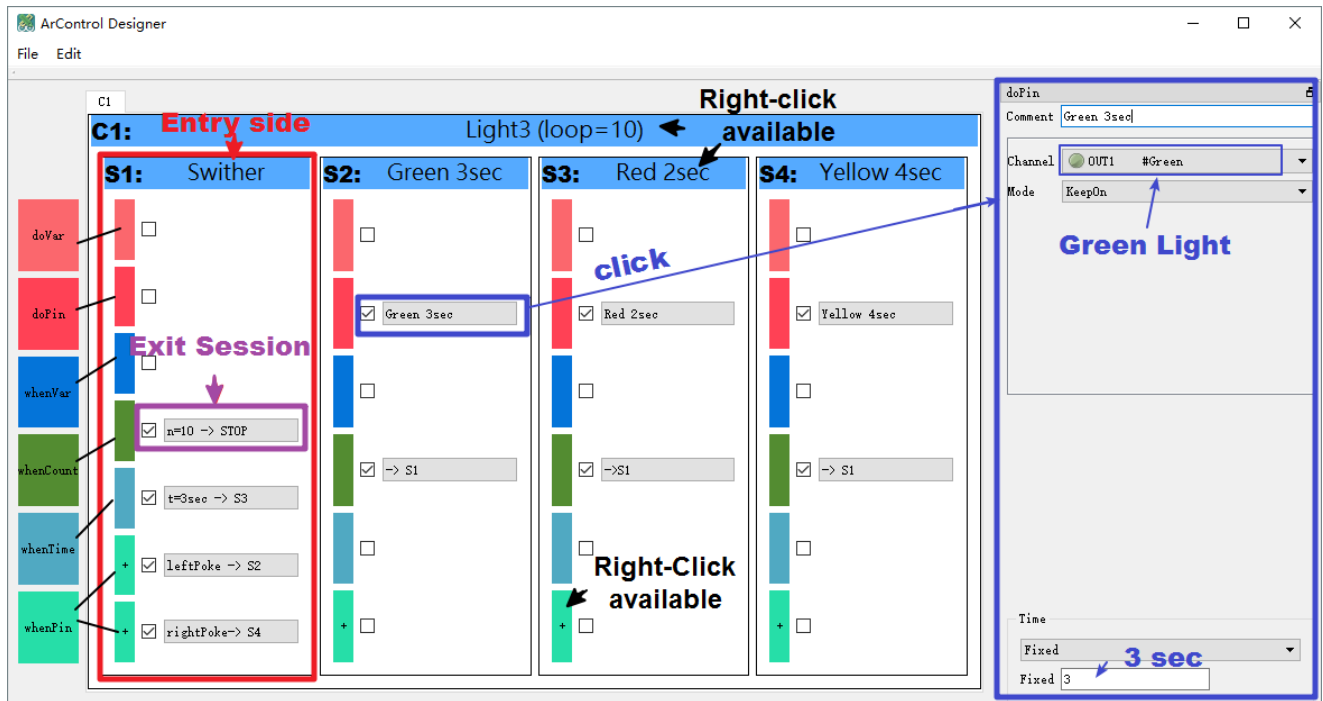


It's nature that different choices lead to different results. It's our first task to follow this schedule (below). Even a proficient programmer may take half an hour to implement such schedule by using native Arduino C++ language. However, ArControl makes things easier. Through a user-friendly GUI, there is no need to master any script language.

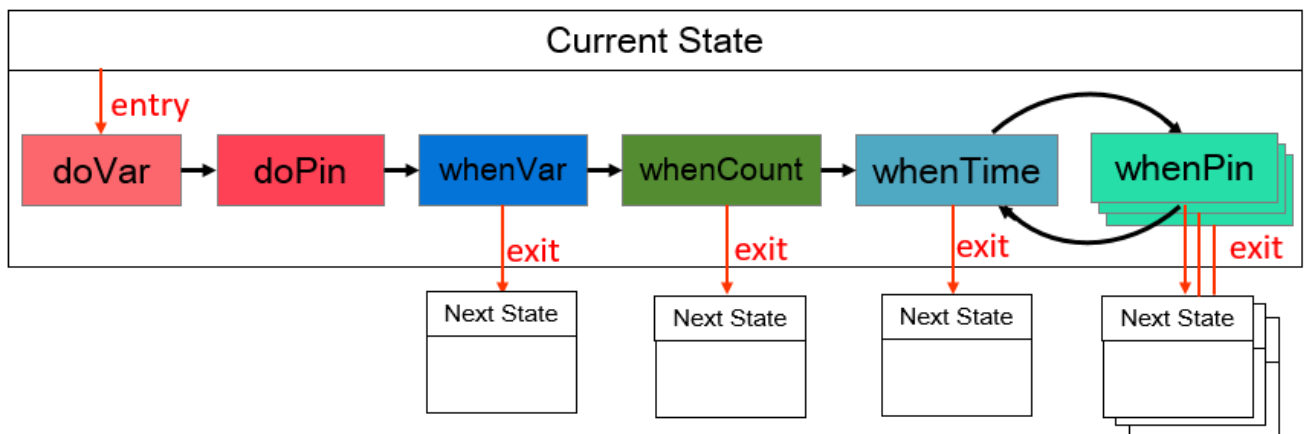


Sequent events of Light3 task. You will get Yellow Ligth (4sec) Green Ligth (3 sec) or Red Ligth (2 sec), depending on the your choice of Left, Right or Wait.

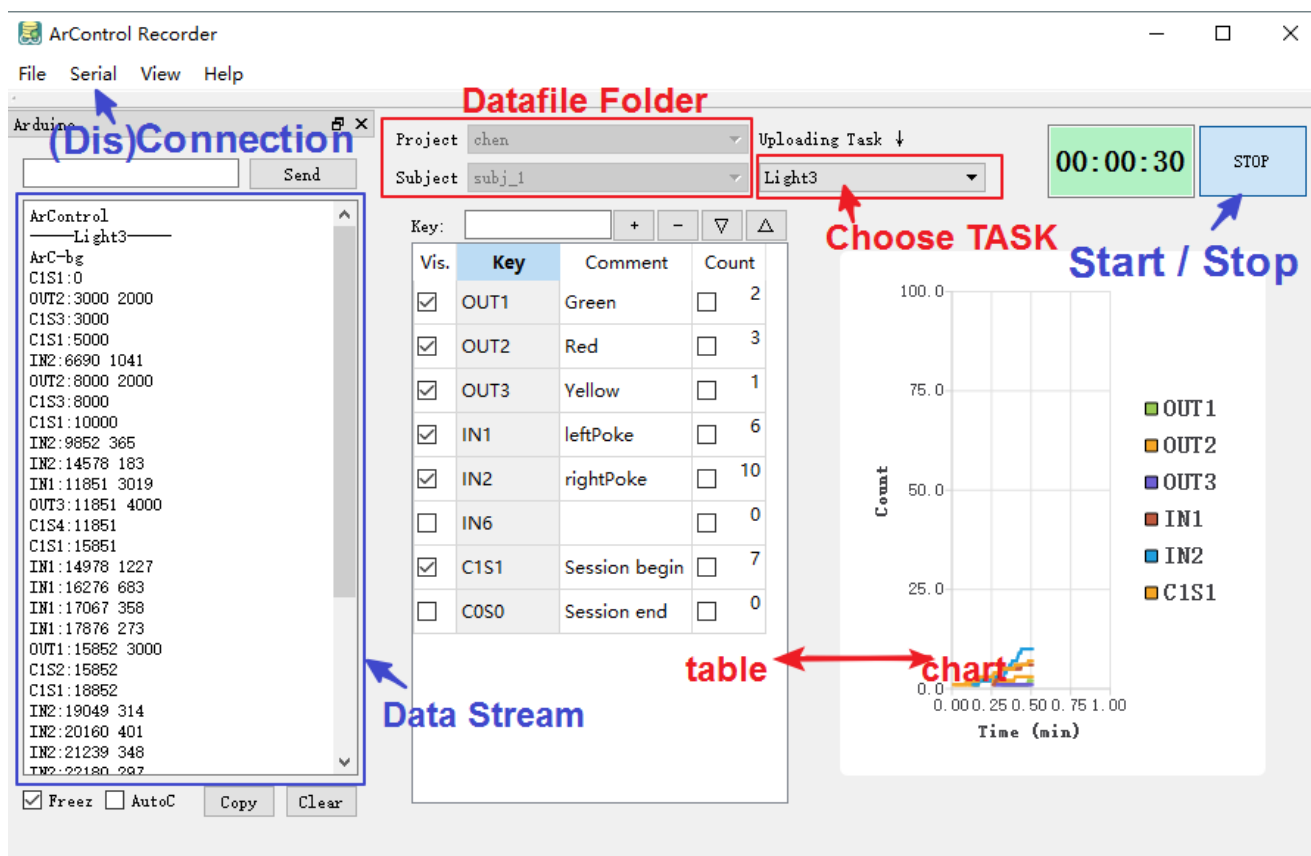
The schedule was decomposed with ArControl Designer as figure below. This demo has been embeded in release version, and it can also be found in `ArControl Designer> File > Open in > Light3`.



Light3 schedule is decomposed with ArControl Designer. Besides, an extra restraint ("max loop = 10 - > Exit Session") is added.



Priority ranks (sequences) of doXXX and whenXXX. Items in whenPin are in equal priorities.



Run a "Light3" task. Data was viewed as log in "Data Stream" panel, as table in middle panel, as chart in right panel, and was collected to data file under "Datafile Folder".

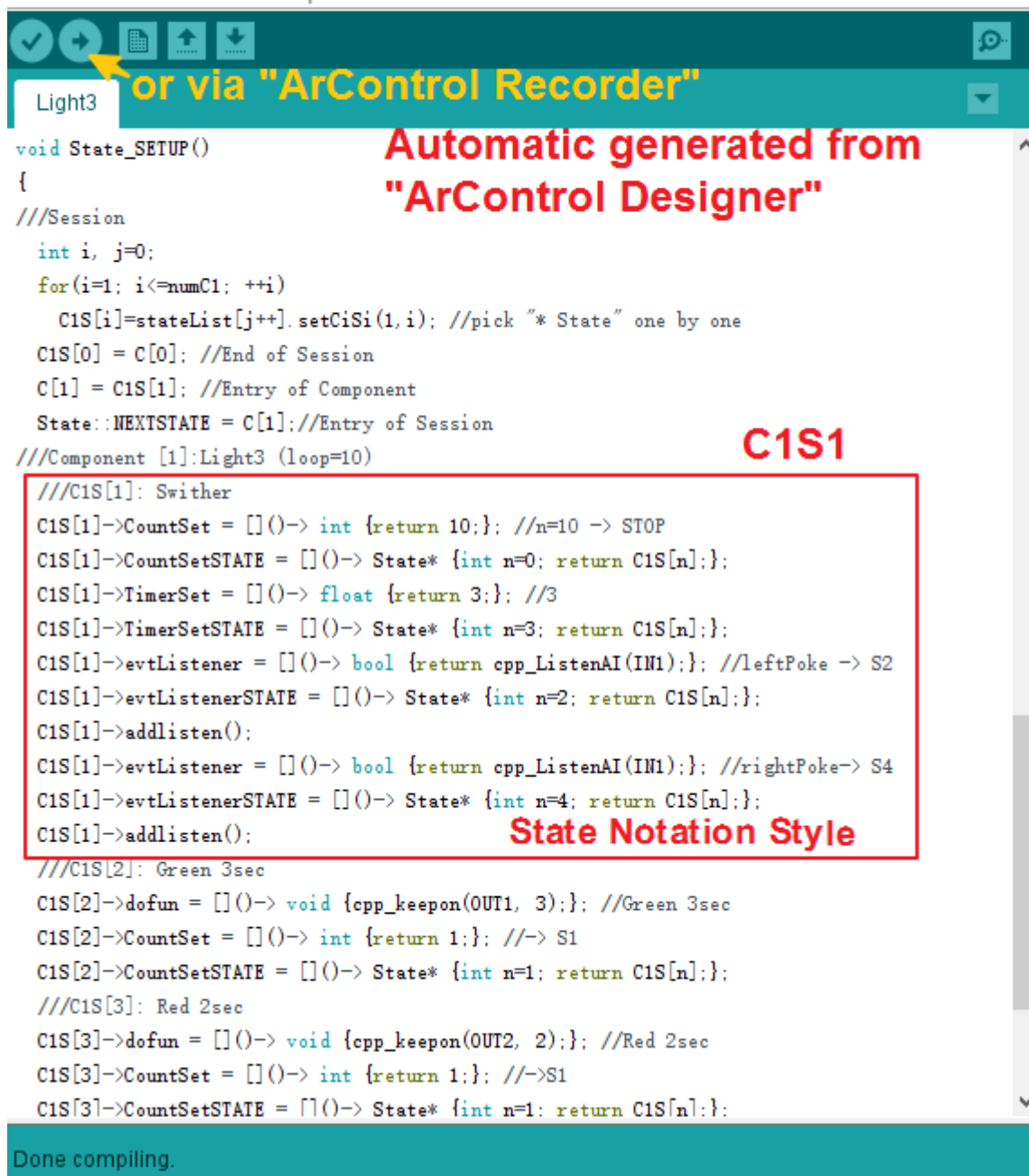
## Insights into the task -- "Light3"

The **ArControl Designer** saved the previous **Light3** layout as *Light3.aconf*, and automatically generated executable *Light3.ino*.

**ARCONTROL\_PATH/task/Light3**

**Designer Outputs**

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



Light3

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.

The ArControl Recorder automatically saved the log to data file. You can use my Matlab function `BF_arc2mat.m` to convert the file from TXT to MAT.

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

```
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
```

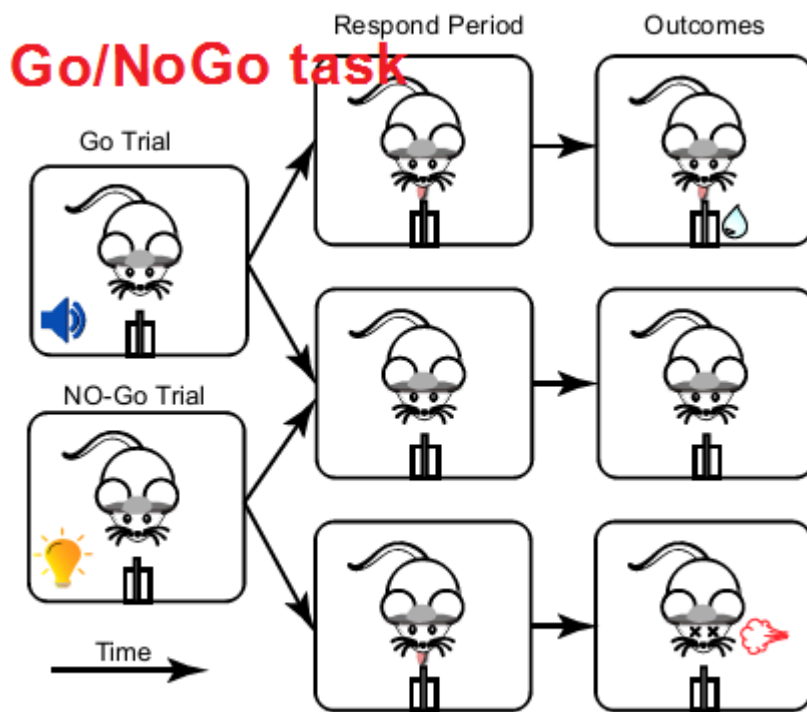
## Other ready demo tasks -- "Go\_NoGo" / "2AFPC" task

---

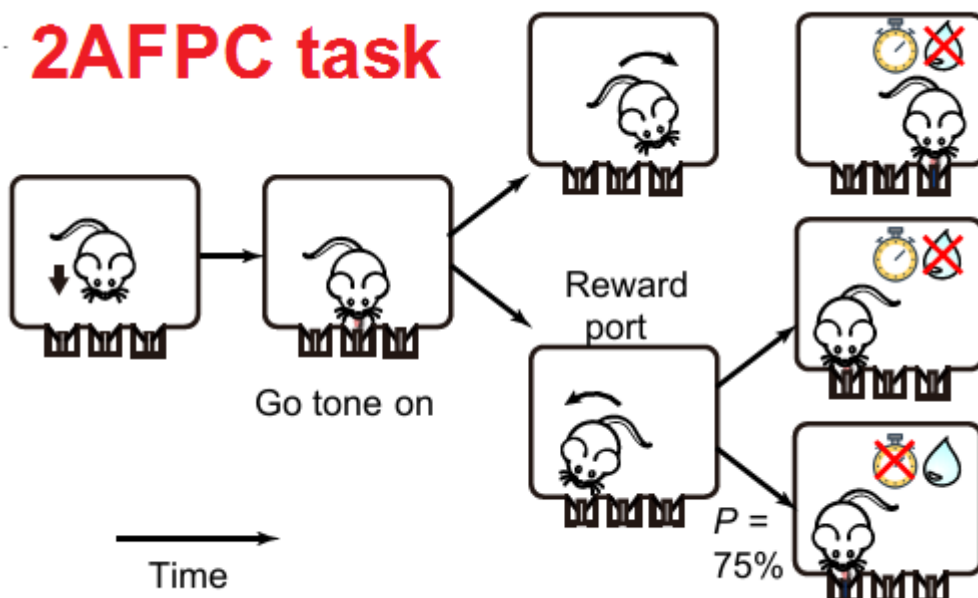
Two common behavior tasks, a Go/No-Go task (Gomez et al., 2007; Dolzani et al., 2013) and a two-choice procedure task (Tai et al., 2012; Stephenson-Jones et al., 2016) which have been widely applied to assess discrimination and memory. Here we implement their logics by ArControl.

### About this two task

The schedules were illustrated as figures below. This demos have been embeded in release version, and can be found in `ArControl Designer> File > Open in > Go_NoGo` and `ArControl Designer> File > Open in > twoAFPC` .



In the Go/NoGo task, mice were body restricted via a head bar and body tube (Guo et al., 2014) and were required to discriminate a go cue (tone) and a no-go cue (light). They would consequently get a reward (water-drop) or a punishment (air-puff) once they respond (lick) to the go cue (tone) or no-go cue (light) during a response window, respectively.



In the 2AFPC task, animals were required to initiate a trial by licking the central port, and sequentially move to a left or a right port to obtain a reward. Only one port was rewarded by 75% at a time. In 25% of trials, neither port was rewarded. If no reward was delivered, animals would be punished with an extra time out. Rewarded port was periodically switched across time.

## Resolution for Common Problems of ArControl Software

## Recorder Error: Messages are Messy Coded when uploading a task.

Arduino IDE should be configure as "English" language. Open `Arduino IDE > File > Perferences > Editor language`, choose `English (English)`.

## Recorder Error: Crash when uploading a task.

Make sure that you have installed Arduino IDE, the Arduino IDE path is valid. You should renew the path of Arduino IDE. Open `ArControl Designer > File > Profile > Arduino IDE path`, choose the valid `arduino_debugger.exe` (for Windows) under the root of Arduino IDE path.

## Recorder / Designer Error: Crash immediately when opening

It may caused by wrong configuration of `ArControl`. You should reset the software. Delete `default.bconf` (if exsits), `task\defalut\` (if exsits) and `profile.xml` (if exsits).

## Recorder Error: A "Time Out" Error when uploading a task.

It's common for a fresh task. Try to repeat, or try to use `ArControl IDE` to upload the task.

## Recorder Error: Cannot connect to Arduino board

It's common when the Arduino board is currently occupied by other program (etc., `Arduino IDE > Serial Monitor`). Reject the Arduino board from this program, or reconnect via the USB port.

## Recorder Error: "fatal error: xxxxx/ArControl\_AllinOne.h: No such file or directory" when uploading a task

Open `mytask.ino`, find the line `#include "xxxxxx/ArControl_AllinOne.h"`. The path may be invalid. Renew this path mannually, or resave the `mytask` via `ArControl Designer`.

## How to: Choose language of ArControl

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`.

## How to: Regard the "whenPin" INx as HIGH or LOW voltage.

Typical sensors reflect resting-status at LOW, and activating-status at HIGH voltage. However, some sensors are opposite coded. Open `Arcontrol Designer > File > Profile > Input signal as`, choose the proper option and it works for all INx.

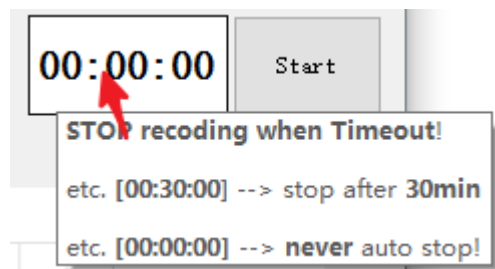
## How to: Start a task immediately or by Hardware-Trigger

A task is commonly triggered by `ArControl Recorder > START` button. However, the task can triggered from other style. Open `ArControl Designer > File > Profile > "Start" Model`. There are `immediately`, `Trigger by Hardware (pin_11_HIGH)`, `Trigger by: Software (serial communicate)`, `Trigger by: Software or Hardware`.

## How to: Timer-Stop a task

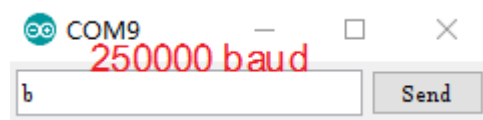


A task is stopped typically from 1. `ArControl Recorder > STOP` button; 2. The task has finished its' schedule (terminates at `c0` or `CxS0`). In addition, you can set a Timer-Stop from the `ArControl Recorder > Timer` window .



## How to: Use Arduino IDE to run a task

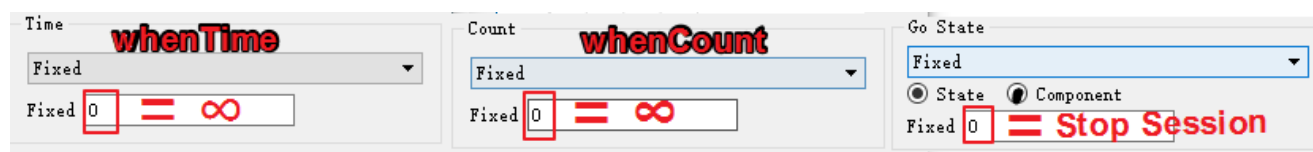
You can use the `ArControl Designer`, as well as `Arduino IDE` to run a task. Open the `Arduino IDE > Serial Monitor`, choose the `250,000 baud` and send the character `b`.



## Skills and Warns

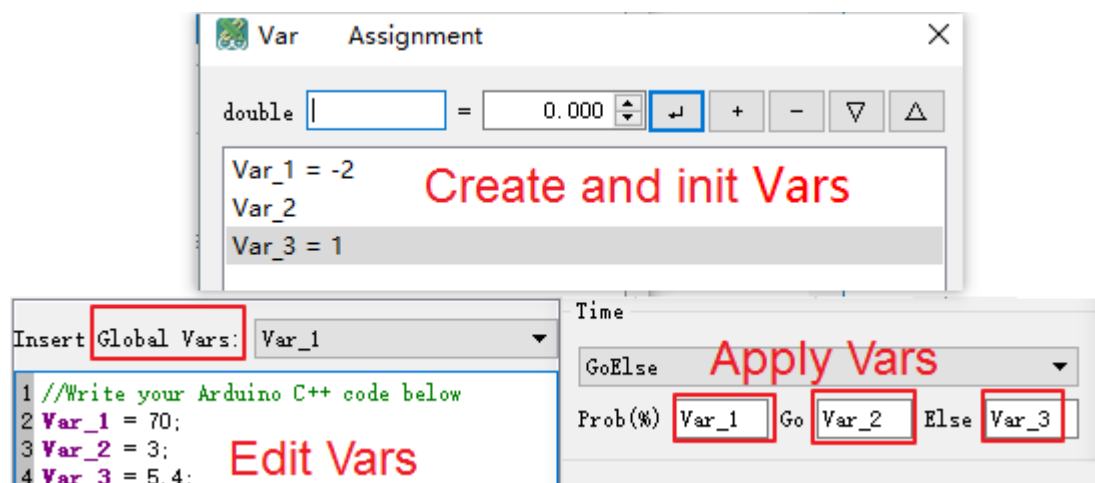
### Warn: Parameters value "0"-- special purposing

Avoid to use `0` as the value of parameter in `ArControl Designer`.



### Skills: Parameters value support Variables

Global Variables are defined in `Edit > Var Assignment`, dynamically refreshed in `doVar` or `whenVar`, sharable among States, and practicable for parameter value replacement.



## **Warn: 20 States are the maximum size of a task**

Limited with the 2Kb SRAM of Arduino UNO, 20 States are the maximum size of a Session.

## **Warn: 200 Hz is the maximum frequency of [INx]**

ArControl could record 200Hz TTL signal from all 6 input channels simultaneously, and 700Hz when focusing on a single channel, with  $\pm 1\text{ms}$  (95% confidence, round effect) accuracy.