开课吧

机器人控制系统

机器人动作控制

主讲：宋德锋　　助教：杨灿

开课吧

CONTENTS
目录

01 机器人控制系统

02 让机器人动起来

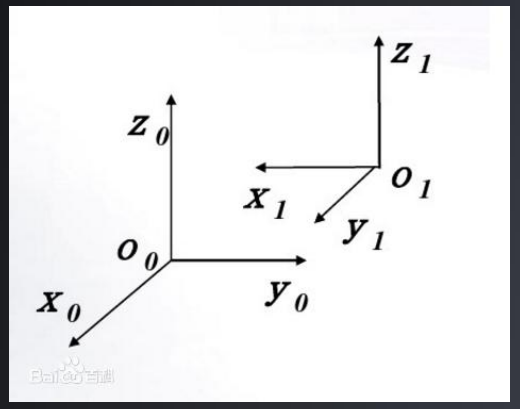
03 用程序控制机器人

机器人控制系统

机器人控制系统通过驱动和执行器共同完成

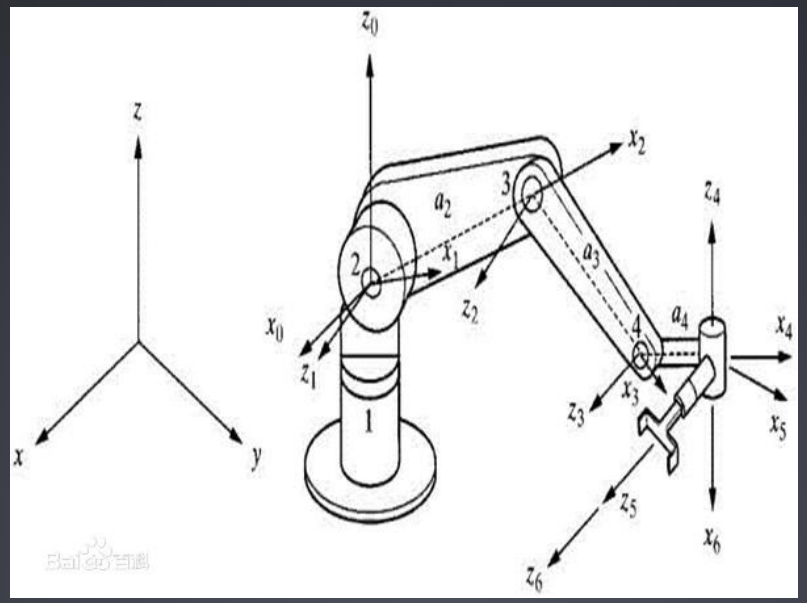
机器人控制系统

机械臂



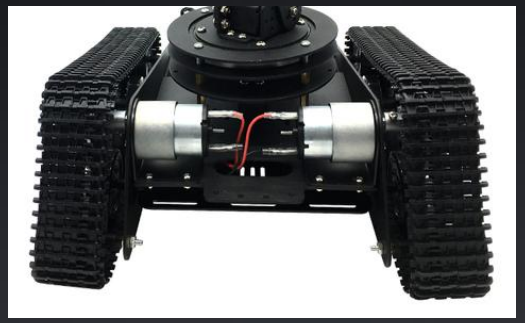
机器人正向运动学：知道每个关节的运动量，推出机械手末端位置和姿态

机器人逆向运动学：知道机械手末端位置和姿态，求解每个关节的运动量

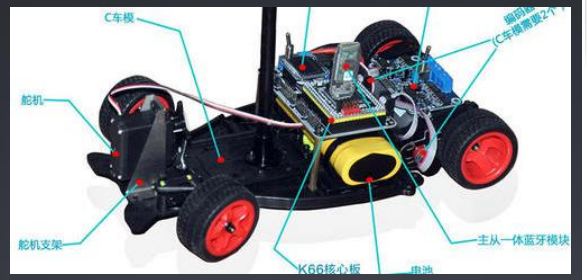


机器人控制系统

运动/移动控制

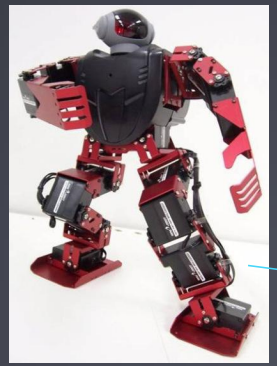


	左轮电机	右轮电机
左转	反转	正转
右转	正转	反转
直行	正转	正转
后退	反转	反转

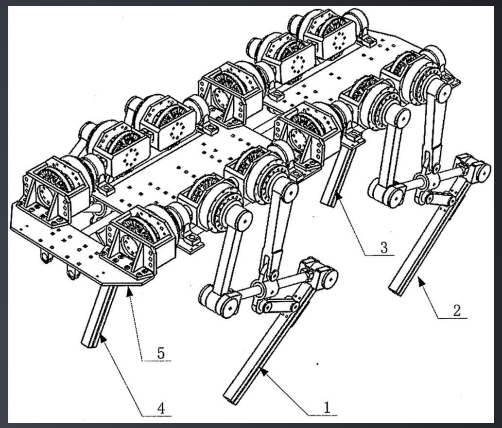


后轮驱动，前轮转向

每个轮单独控制转向、驱动



每个关节一个舵机



波士顿动力：活塞

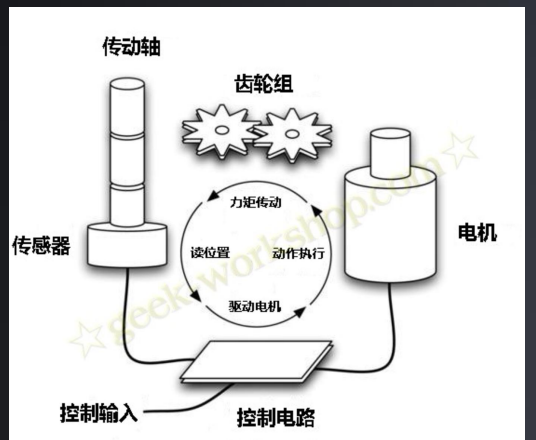
机器人控制系统

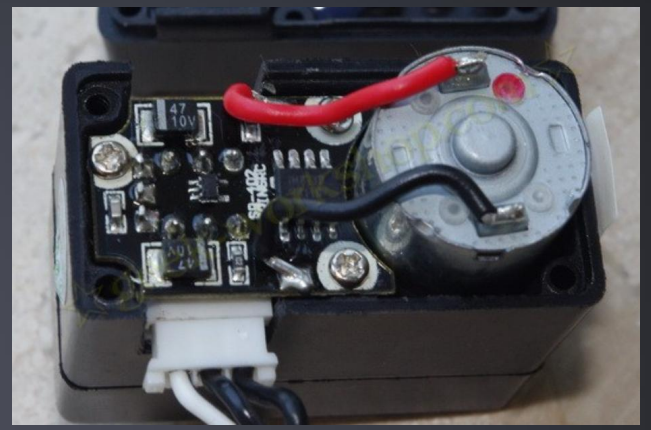
开课吧

舵机

- 一套自动控制系统，通过发送信号，指定输出轴旋转角度
- 如果从一个型号换成另一个型号，整个机械结构都需要重新设计

	舵机	普通直流电机
转动范围	在一定角度范围	连续转动
转动的角度信息反馈	可	不可
用途	控制某物体转动一定角度	整圈转动做动力





机器人控制系统

舵机

- MG995
  - 价格便宜，金属齿轮，耐用度好
  - 扭力较小

- SR 403
  - 价格便宜，全金属齿轮，扭力大
  - 做工很山寨

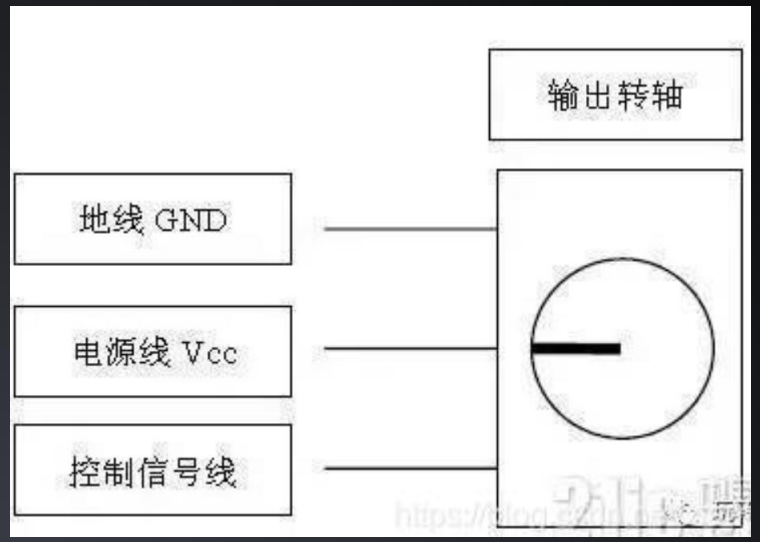
- AX12+ 机器人专用舵机，
  - 数字舵机
  - RS485串口通信

机器人控制系统

舵机

- 一般为3线制：红色为电源，棕色为地，黄色为信号
- 模拟舵机与数字舵机

	模拟舵机	数字舵机
结构		微处理器
控制方式	不停的发送PWM信号	只需要发送一次PWM信号
用途	控制某物体转动一定角度	整圈转动做动力

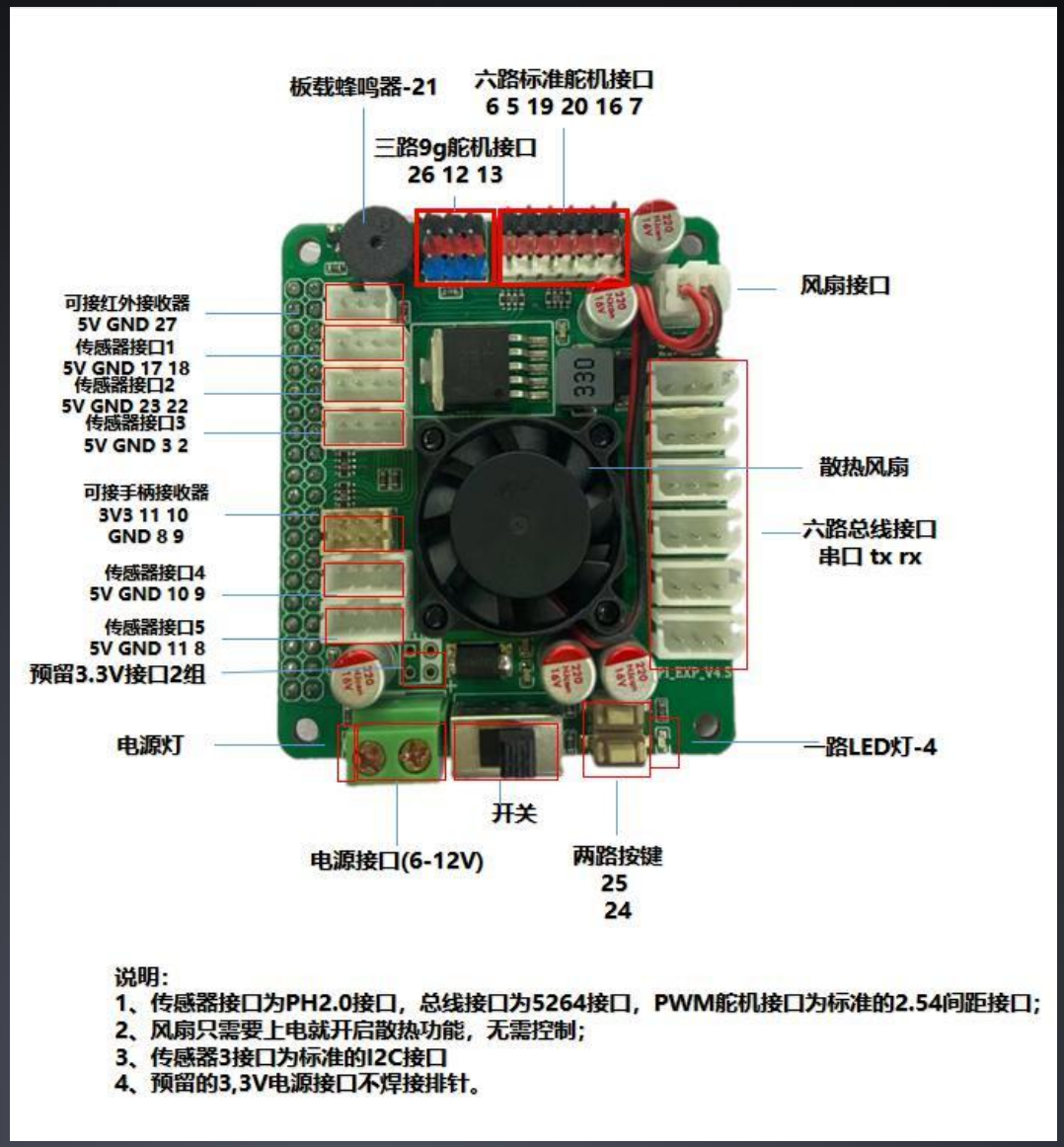


总线伺服舵机：模拟-->数字-->串行总线

机器人控制系统

驱动

- 电源
- 执行器驱动



说明：
1、传感器接口为PH2.0接口，总线接口为5264接口，PWM舵机接口为标准的2.54间距接口；
2、风扇只需要上电就开启散热功能，无需控制；
3、传感器3接口为标准的I2C接口
4、预留的3,3V电源接口不焊接排针。

让机器人动起来

图形化工具调试

让机器人动起来

开课吧

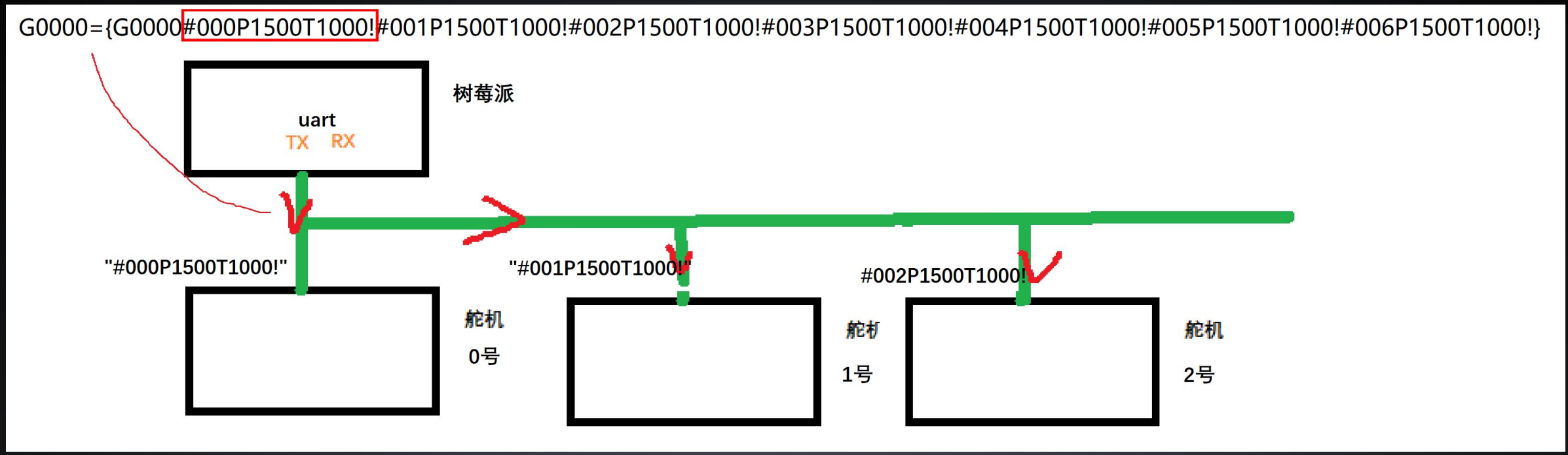
任务2-1：

- 尝试调到单个舵机
- 设计实现属于自己机器人的的动作组：

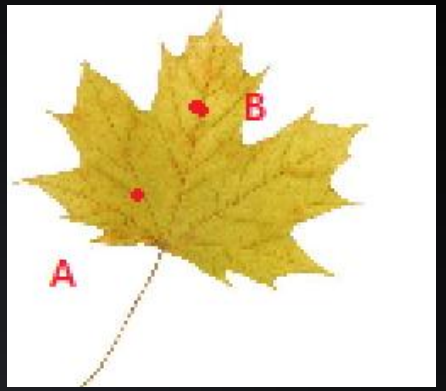
  前进、后退、左转、右转
- 将完成的动作视频和导出相应的工程

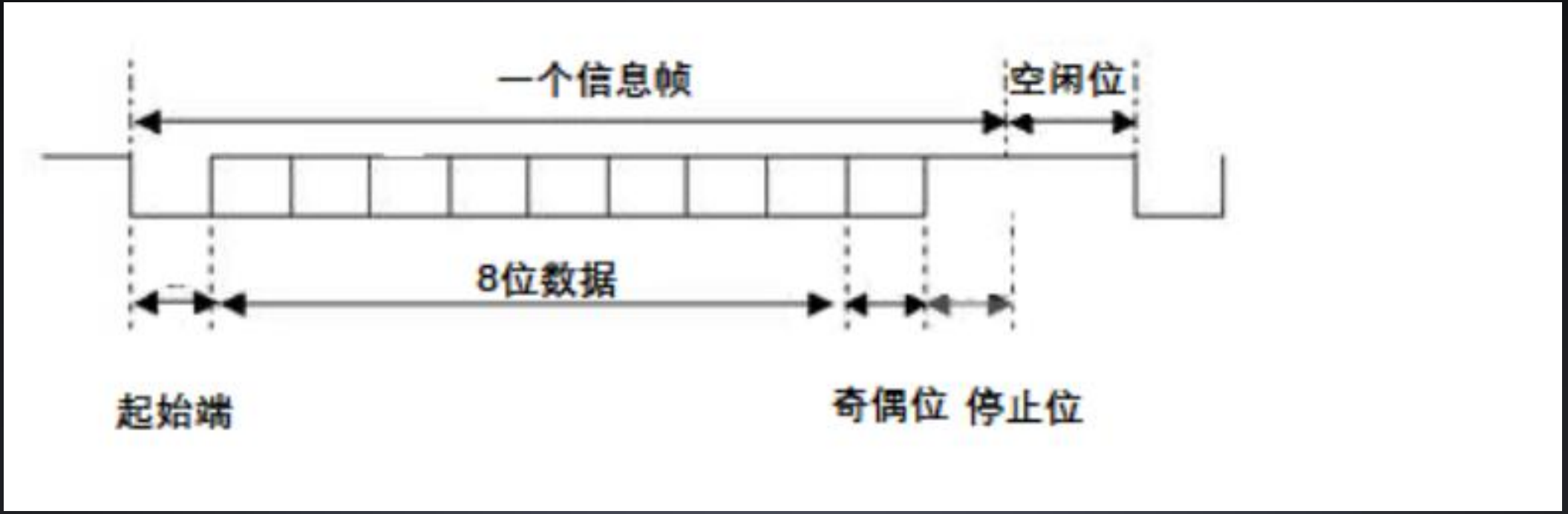
  ini文件发到群

从图形化工具调试动作到uart通信



串口基本概念





从图形化工具调试动作到uart通信

开课吧

- 串口通信几个重要的问题
  - 通信协议设计
    - 格式：

      头+内容+尾

      头+长度+内容
    - 操作：

      打包

      解析
  - 数据缓存机制：当很多个动作组传给舵机时，机器人如何有条不紊的完成动作？

从图形化工具调试动作到uart通信

开课吧

思考：

- Uart需要怎么配置呢？
- 怎么传输数据，传什么数据，传到哪儿？

从图形化工具调试动作到uart通信

开课吧

- 引脚
  - 串行通信接口
    - GPIO
    - uart
    - IIC
  - 电源
- 库函数
  - WiringPi
  - 步骤
    - 初始化：串口初始化——获取句柄
      - 打开串口设备（查看位置：ls /dev/tty*）
      - 波特率（115200）——对符号传输速率的一种度量
    - 对串口进行相应的操作
      - 发送（写）

从图形化工具调试动作到uart通信

开课吧

- 任务2-2：
  - 打开并初始化串口
  - 将机器人走路动作组通过串口发送给舵机，使得机器人能够执行相应的动作

uart通信

开课吧

```
#include <wiringSerial.h>
int uart_fd;

int init_action()
{
    uart_fd = serialOpen("/dev/ttyAMA0",115200);
    if(uart_fd < 0){
        printf("open serial failure\n");
        return 0;
    }
    return 1;
}

serialPrintf(fd, “#001P1600T1000!”);
```

动作文件robot.ini分析与处理

开课吧

```
[group]
G0000={G0000#001P1500T1000!#002P1500T1000!#003P1500T1000!#004P1500T1000!#005P1500T1000!#006P1500T1000!}
G0001={G0001#001P1500T1000!#002P1500T1000!#003P1500T1000!#004P1500T1000!#005P1500T1000!#006P1500T1000!}
G0002={G0002#001P1500T1000!#002P1500T1000!#003P1500T1000!#004P1500T1000!#005P1500T1000!#006P1500T1000!}
G0003={G0003#001P1350T0200!#002P1500T0200!#003P1500T0200!#004P1250T0200!#005P1500T0200!#006P1500T0200!}
G0004={G0004#001P1350T0200!#002P1400T0200!#003P1400T0200!#004P1400T0200!#005P1400T0200!#006P1400T0200!}
G0005={G0005#001P1500T0200!#002P1400T0200!#003P1400T0200!#004P1500T0200!#005P1400T0200!#006P1400T0200!}
G0006={G0006#001P1750T0200!#002P1400T0300!#003P1400T0300!#004P1650T0200!#005P1400T0300!#006P1400T0300!}
G0007={G0007#001P1600T0200!#002P1500T0300!#003P1500T0300!#004P1650T0200!#005P1500T0300!#006P1500T0300!}
G0008={G0008#001P1600T0200!#002P1600T0300!#003P1600T0300!#004P1650T0200!#005P1600T0300!#006P1600T0300!}
G0009={G0009#001P1500T0200!#002P1600T0200!#003P1600T0200!#004P1500T0200!#005P1600T0200!#006P1600T0200!}
G0010={G0010#001P1350T0300!#002P1600T0300!#003P1600T0300!#004P1250T0300!#005P1600T0300!#006P1600T0300!}
G0011={G0011#001P1350T0200!#002P1500T0200!#003P1500T0200!#004P1250T0200!#005P1500T0200!#006P1500T0200!}
G0012={G0012#001P1350T0200!#002P1600T0200!#003P1600T0200!#004P1400T0200!#005P1600T0200!#006P1600T0200!}
G0013={G0013#001P1500T0200!#002P1600T0200!#003P1600T0200!#004P1500T0200!#005P1600T0200!#006P1600T0200!}
```

动作文件分析与处理

开课吧

```
#include <unistd.h>
#include <fcntl.h>
#include <string.h>
char cmd[100][120];
int action_load(){
    int fd; char buf[1024*10]={0}; char *locate;
    fd = open("robot.ini", O_RDONLY);
    if(fd < 0) {
        perror("open file error");
        return 1;
    }
    read(fd, buf, sizeof(buf));
    locate = strstr(buf, "G0000");//locate指向第一个G0000开始的位置
    memcpy(cmd[0],locate,104);//将第一行动作指令填写到cmd[0]
    这里要完成全部动作读入到cmd
    return 0;
}
```

动作文件导入处理

开课吧

```
#include <unistd.h>
#include <stdio.h>
#include <string.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
char cmd[100][119];
int file_open(const char* file_name)
{
    int fd = open(file_name, O_RDONLY);
    if(fd < 0) {
        perror("open file error");
        return -1;
    }
    return fd;
}
```

```
int file_handle(int fd)
{
    char *locate = NULL;
    int i;
    char buf[1024 * 10] = {0};
    read(fd, buf, sizeof(buf));
    locate = strstr(buf, "G0000");
    for(i = 0; i < 88; i++) {
        memcpy(cmd[i], locate, 120);
        locate += 120;
         printf( “%s\n” ,cmd[i]);
    }
    return 0;
}
```

机器人动作控制函数

开课吧

```
void robot_act(int ud,int begin,int end,int dt,int times)
{
	int i = 0;
	while(times--)
	{
		for(i = begin; i < end; i++)
		{
			serialPrintf(ud,cmd[i]);
			// printf("cmd[%d]:[%s]\r\n",i,mycmd+(UART_PRO_LEN+1)*i);
			delay(dt);
		}
	}
}
```

机器人动作封装

开课吧

```
void robot_standby();
void robot_forward(int steps);
void robot_backward(int steps);
void robot_turnleft(int steps);
void robot_turnright(int steps);
void robot_climbup_f();
void robot_climbup_b();
void robot_climbup_l();
void robot_climbup_r();
```


开课吧
控制，便是王道

开课吧

THANK YOU FOR
WATCHING