**Arduino入门指南**

作为全球优秀的硬件开发平台及最流行的开源微控制器，Arduino以其简单的开发方式吸引开发者的到来，在关注创意与项目实现的基础上更快完成项目开发，极大节约了学习成本，缩短了开发周期。

初探轮式机器人微控制器，Arduino板可以说是一个不错的选择！

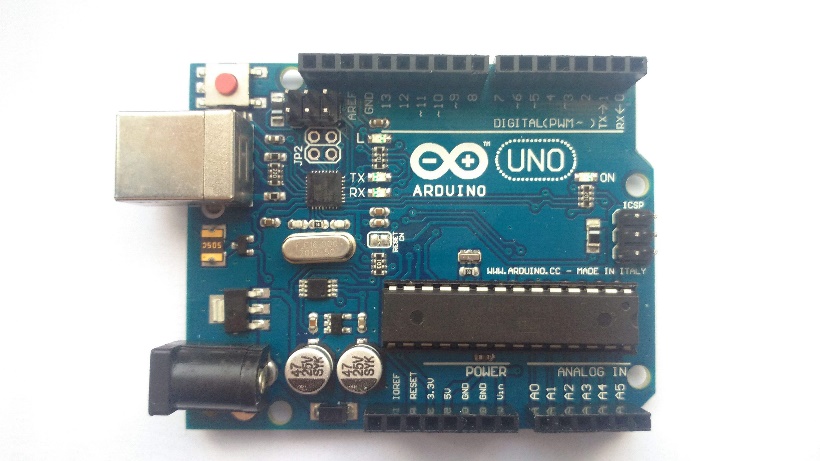
**Arduino究竟是什么？**

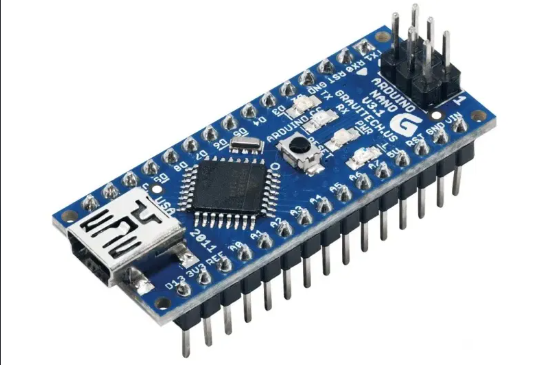
一本极佳的Arduino入门书籍《爱上Arduino》中这么解释：“Arduino是一个开源电子原型制作平台，包括一个简单易用的电路板以及一个软件开发环境。”我们日常所说的Arduino并不是单纯的一种东西，它既可以指硬件，也可以指代软件。往往我们用这个名字去指代特定的电路板、编程语言和一个编辑器。

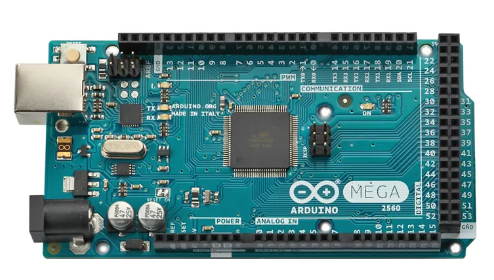
**Arduino主板**

根据型号的不同，电路板的外貌、引脚数与主控芯片的性能都有所不同。不过需要注意的是，只有这整个板子合起来才叫Arduino，而这个板子上的任何一个零件都不叫Arduino。~~冷知识：平时所说的单片机其实是指这块板子上的主控芯片，但主控芯片并不叫Arduino。~~

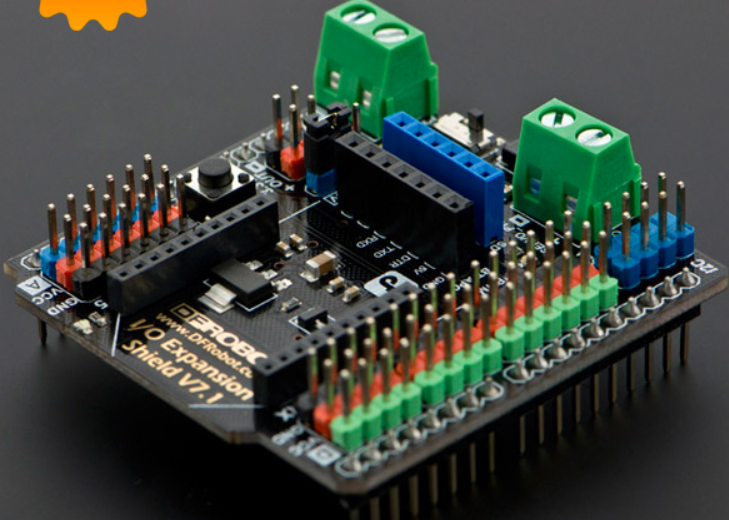
下面是几个常用的Arduino主板

 Arduino UNO R3

 Arduino Nano

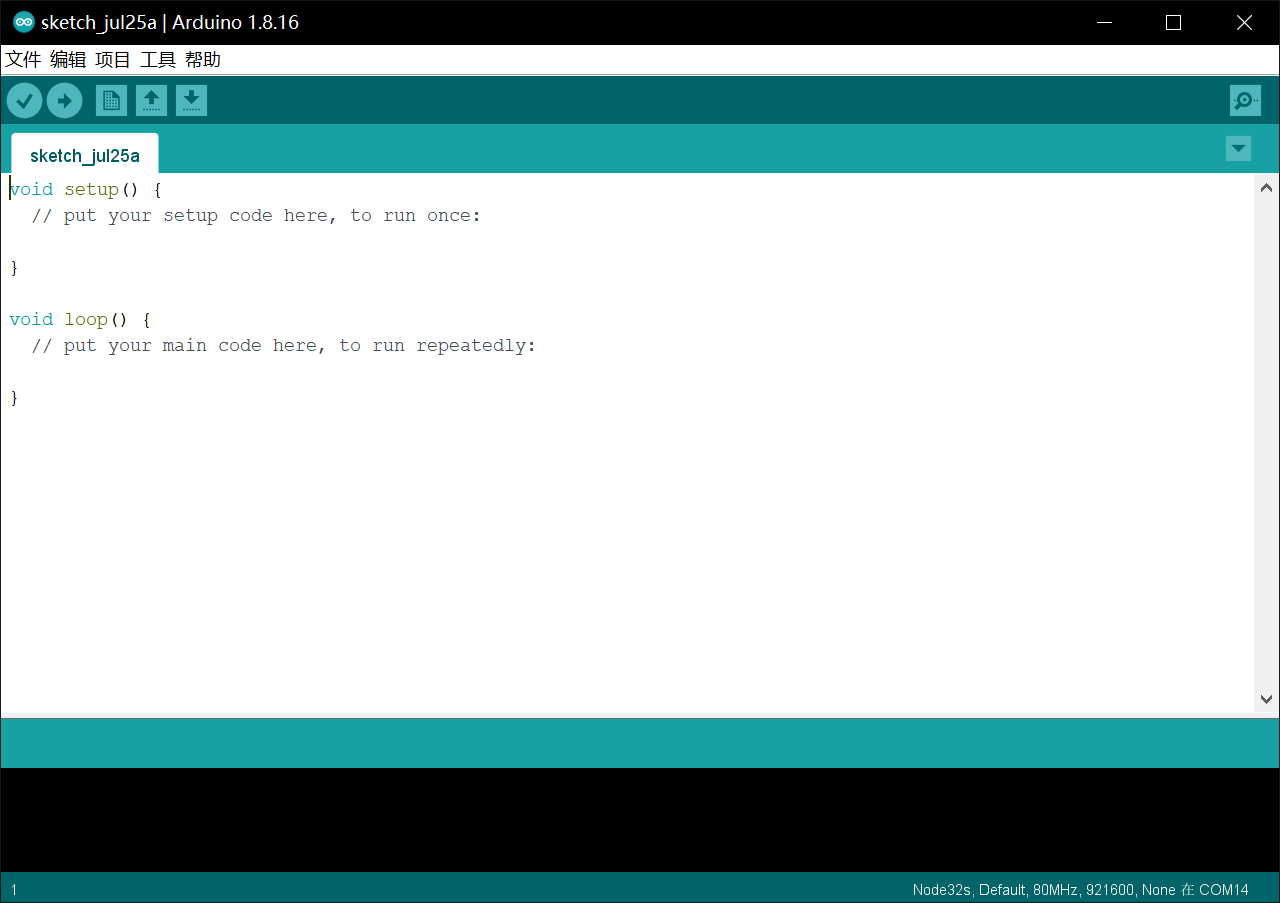
Arduino Mega2560

主板之外，我们还可以装上Arduino兼容的许多扩展版，在这里不作具体展开。

DFRduino IO 传感器扩展板 V7.1

**Arduino IDE (集成开发环境)与Arduino语言**

Arduino 语言是建立在 C/C++基础上的,其实也就是基础的C语言,Arduino 语言只不过把 AVR 单片机(微控制器)相关的一些参数设置都函数化，方便我们进行使用。Arduino IDE则是我们编写Arduino程序与下载程序的软件。下面，就让来了解一下一个Arduino程序的结构吧！

 Arduino IDE

**Arduino程序的结构**

与我们日常所见的C语言程序不同，Arduino程序结构中并没有main函数。取而代之的是在新建Arduino项目时自带的setup函数与loop函数。Arduino将main函数包装进了这两个函数之中，所以在我们的Arduino程序之中并不需要额外添加一个main函数。

Arduino程序由在程序开始时只执行一次的setup函数与会循环执行的loop函数组成。一般而言，我们会在setup函数中进行一些参数的设置，引脚的初始化，串口波特率的设置等行为。然后在loop函数中具体实现我们的任务目标。

**Arduino常用函数**

1. **pinMode**

pinMode是一个几乎能在所用setup函数中见到的一个函数，pinMode的功能是设置一个引脚的输入输出模式，在对任意一个引脚操作前都需要对引脚的输入输出模式进行初始化。使用方法是：

pinMode(引脚编号,模式)

在我们的初期学习中，对于引脚的模式，我们仅需了解从引脚处读入数据的INPUT模式与向引脚处输出数据的OUTPUT模式。

//pinMode(引脚编号,模式)

#define IN1 7//定义引脚编号的别名，下同

#define IN2 5

#define IN3 2

#define IN4 4

#define R2 11

#define R1 10

#define L1 9

#define L2 8

void setup()

{

pinMode(IN1, OUTPUT); //设置IN1引脚为输出模式

pinMode(IN2, OUTPUT); //设置IN2引脚为输出模式

pinMode(IN3, OUTPUT); //设置IN3引脚为输出模式

pinMode(IN4, OUTPUT); //设置IN4引脚为输出模式

pinMode(L2,INPUT); //设置L2引脚为输入模式

pinMode(L1,INPUT); //设置L1引脚为输入模式

pinMode(R1,INPUT); //设置R1引脚为输入模式

pinMode(R2,INPUT); //设置R2引脚为输入模式

}

1. **digitalRead/digitalWrite**

digitalRead/digitalWrite，也就是是数字输入输出函数，可以实现电路中高电平与低电平和程序中1与0（即HIGH与LOW）数字信号之间的转换。也就是我们通过digitalRead函数读取电路中的高低电平状态，而使用digitalWrite函数向电路中输出高低电平。使用pinmode函数初始化的引脚高电平为5V或3.3V，低电平为0V（GND）。使用方法是：

digitalRead(引脚编号)

digitalWrite(引脚编号, 值)

//pinMode(引脚编号,模式)

int l1,l2,r1,r2;

void loop()

{

r2=digitalRead(R2);

r1=digitalRead(R1);

l1=digitalRead(L1);

l2=digitalRead(L2);

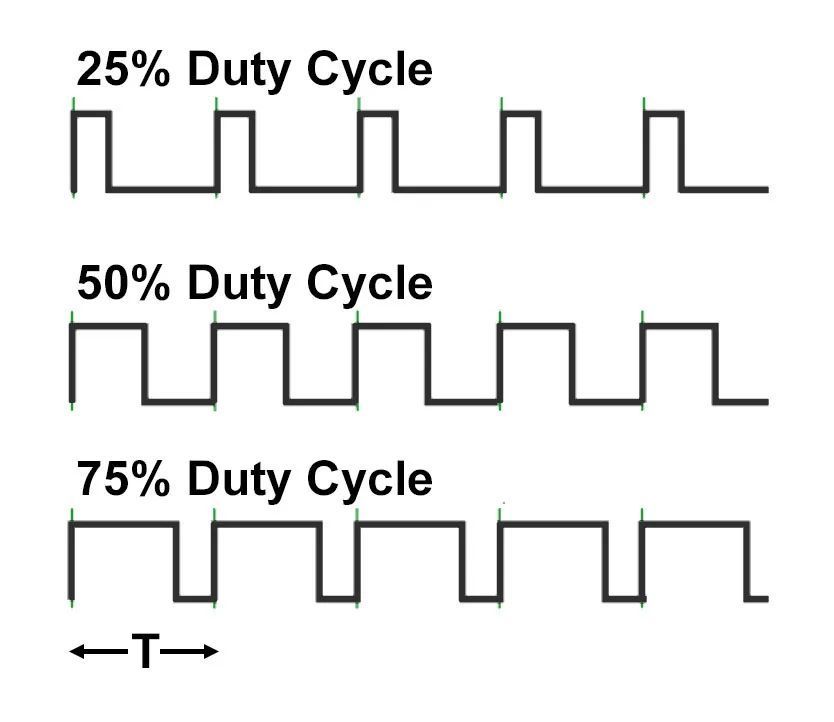
}

1. **analogRead/analogWrite**

analogRead/analogWrite,即模拟输入输出函数，是针对电路中的模拟信号而出现的两个函数。需要注意的是，analogRead与analogWrite没有什么相关性，并不是用analogWrite输出的信号可以使用analogRead接收。两个函数都是相互独立的。

对于analogRead，它只能使用主板上有着ANALOG IN标识的模拟信号引脚，并将引脚处的电压按0~5V映射到0~1023的形式读取。

而对于analogWrite而言，它只能使用主板上有PWM功能的引脚（即有着标号“~”的引脚）。PWM（脉冲宽度调制）是一种能将稳压电源的输出改变的一种模拟控制方式，PWM控制一段较短时间内的开关的闭合，使得开关开与闭的时间比值不同，从而让等效电压发生改变。在Arduino实现的analogWrite中，程序将0~255共256个数字映射到0%~100%占空比当中。因此使用analogWrite函数合理地调节占空比，不仅可以控制电机等驱动装置的不同转速，还可以控制LED灯的明暗程度。

PWM波

尽管两者含义不同，并不对应，但两个函数的使用方法依旧与数字输入输出函数类似：

analogRead(引脚编号)

analogWrite(引脚编号, 值)

值得注意的是analogWrite函数在使用前可以不用对相应的引脚进行pinMode初始化，这是analogWrite比较特殊的一点。

1. **delay**

delay函数，即延时函数，用法较为简单，需要注意的是delay的时间单位是毫秒。

delay(时间)

该函数还有另一个少用的变体，delayMicroseconds，函数的时间单位为微秒。

delayMicroseconds(时间)

1. **Serial成员函数**

Serial，意为串口。在Arduino程序开发的过程中我们经常会遇到需要使用串口通信的情况。Serial是一个包装好的类，通过Serial中的成员函数，我们即可以使用串口进行通讯。最通常的情况，我们会使用串口来和电脑进行通讯，写出Arduino的“Hello World”。

对于不同的硬件，串口的数量是不同的。对于Arduino Uno R3，其上仅仅只有一个硬件串口，并且被usb所占据。而对于Arduino Mega2560，则有着4个硬件串口，我们就可以使用其余三个硬件串口来进行通讯。

硬件串口一般共需要四个引脚，VCC，TX，RX，GND。其中，VCC是供电，GND是接地，依次连接VCC与GND即可。TX与RX一个是传输，一个是接收，在连接时，注意TX要连接RX，RX连接TX。

|  |
| --- |
| USB转TTL模块 |

在软件方面，我们经常使用Serial下的begin, print, println, read, available这几个函数。用法如下：

Serial.begin(波特率)

Serial.print(常量或变量)

Serial.println(常量或变量)

Serial.read()

Serial.available()

其中，begin用于初始化串口及规定波特率，而使用Serial.begin(波特率, config)可以配置数据位、校验位、停止位等。例如Serial.begin(9600,SERIAL\_8E2)是将串口波特率设为9600，数据位8，偶校验，停止位2。如没有config，默认配置为8数据位，无校验位，停止位1。



config可用配置如上

print与println都用于向串口中传递信息，区别在于print在执行后就结束，println在执行后会自动换行；read用于从串口中读取信息；available用于返回串口缓冲区中当前剩余的字符个数。

需要注意，在使用串口时，选取越高的波特率并不越好。串口波特率越高，传输距离越短。实际上能使用多少的波特率，跟处理器的主频有关；而且主频最好是波特率的整数倍，否则的话可能会增加错误率。Arduino支持的波特率包括：300, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400, 57600, 和 115200。

|  |
| --- |
| 一个简单的Hello World |

**Arduino实例**

1. **Blink示例**

作为最简单的Arduino示例程序，往往也是大家第一个学习的程序，Blink相当简单，它可以控制Arduino板上的板载LED闪烁。

我们也可以通过改变延时的时间，达到PWM的效果——一盏不完全亮起的LED灯。

//当按下重置或接通电路板电源时，setup函数运行一次

#define LED\_BUILTIN 13//对于Arduino UNO R3

void setup() {

// 初始化数字引脚LED\_BUILTIN作为输出。

pinMode(LED\_BUILTIN, OUTPUT);

}

// loop函数永远反复运行

void loop() {

digitalWrite(LED\_BUILTIN, HIGH); // 点亮LED（HIGH是电平）

delay(1000); // 等待一秒

digitalWrite(LED\_BUILTIN, LOW); // 通过设置电平为LOW来熄灭LED

delay(1000); // 等待一秒

}

调节LED灯明亮度的另一种方法：

//当按下重置或接通电路板电源时，setup函数运行一次

#define LED\_BUILTIN 13//对于Arduino UNO R3

void setup() {

// 初始化数字引脚LED\_BUILTIN作为输出。

pinMode(LED\_BUILTIN, OUTPUT);

}

// loop函数永远反复运行

void loop() {

digitalWrite(LED\_BUILTIN, HIGH); // 点亮LED（HIGH是电平）

delay(8); // 等待8ms

digitalWrite(LED\_BUILTIN, LOW); // 通过设置电平为LOW来熄灭LED

delay(2); // 等待2ms

}

1. **电机控制实例**

电机的控制就需要稍微复杂一些，在这里，我们使用一块L298N电机驱动芯片。一块普通的L298N芯片可以同时控制两个电机。L298N上，IN1~IN4接到Arduino上的4个任意的数字输入输出引脚；ENA与ENB接两个有PWM功能的数字输入输出引脚；5V接Arduino上的板载5V；12V接电源的正极；GND接电源负极或板载接地；A1、A2、B1、B2分别接两个电机。

对于直流电机，电流正向流入电机，电机就会正转；电流反向流入电机，电机就会反转。在例程中，我们就对于电机的两个逻辑控制的引脚分别设置为高电平与低电平来实现正转反转的控制。在使用L298N电机驱动芯片后，我们可以调整使能（ENA、ENB）来调节电机的速度，利用analogWrite我们就可以利用PWM的原理控制电机运行的速度。

#define IN1 7

#define IN2 5

#define ENA 6

#define IN3 2

#define IN4 4

#define ENB 3//引脚别名定义

void MotorA(int speed)//电机A控制函数

{

Serial.print("MotorR=");Serial.println(speed);//输出调试信息

if(speed > 0)//正转

{

digitalWrite(IN1, LOW);//IN1引脚设置为低电平

digitalWrite(IN2, HIGH);//IN2引脚设置为高电平

analogWrite(ENA, speed);//ENA引脚PWM输出speed

}

else if(speed < 0)//反转

{

digitalWrite(IN1, HIGH);//IN1引脚设置为高电平

digitalWrite(IN2, LOW);//IN2引脚设置为低电平

analogWrite(ENA, -speed);//ENA引脚PWM输出speed绝对值

}

else//静止

{

digitalWrite(IN1, LOW);//IN1引脚设置为低电平

digitalWrite(IN2, LOW);//IN2引脚设置为低电平

analogWrite(ENA, 0);//ENA引脚输出0

}

}

void MotorB(int speed)//电机B控制函数

{

Serial.print("MotorL=");Serial.println(speed);//输出调试信息

if(speed > 0)//正转

{

digitalWrite(IN3, LOW);//IN3引脚设置为低电平

digitalWrite(IN4, HIGH);//IN4引脚设置为高电平

analogWrite(ENB, speed);//ENB引脚PWM输出speed

}

else if(speed < 0)//反转

{

digitalWrite(IN3, HIGH);//IN3引脚设置为高电平

digitalWrite(IN4, LOW);//IN4引脚设置为低电平

analogWrite(ENB, -speed);//ENA引脚PWM输出speed绝对值

}

else//静止

{

digitalWrite(IN3, LOW);//IN3引脚设置为低电平

digitalWrite(IN4, LOW);//IN4引脚设置为低电平

analogWrite(ENB, 0);//ENA引脚输出0

}

}

void setup()

{

Serial.begin(9600);//串口波特率设置为9600

pinMode(IN1, OUTPUT);

pinMode(IN2, OUTPUT);

pinMode(IN3, OUTPUT);

pinMode(IN4, OUTPUT);//依次设置引脚输入输出模式

}

void loop()

{

MotorA(200);//A电机以200速度正转

MotorB(200);//B电机以200速度正转

}

