Smatcarer必须要了解的一些软件知识

不管你选择软件还是硬件你都要懂的软件知识

单片机是什么

我们大概可以把单片机的功能拆分成**执行、感知、计算、计时**四大块(当然是我自己拆的Orz)

所谓的执行就是单片机可以控制一些末端的执行机构，比如单片机可以点亮LED、可以驱动舵机、可以驱动直流电机等等。

感知就是单片机可以通过一些传感器感知真实世界的信息，比如可以外接温湿度计来获取温湿度、可以通过ADC来获取施加在IO上的电压。

计算就是单片机可以完成一些计算的功能，比如单片机可以完成对信号的卡尔曼滤波、互补滤波等其他复杂的运算。

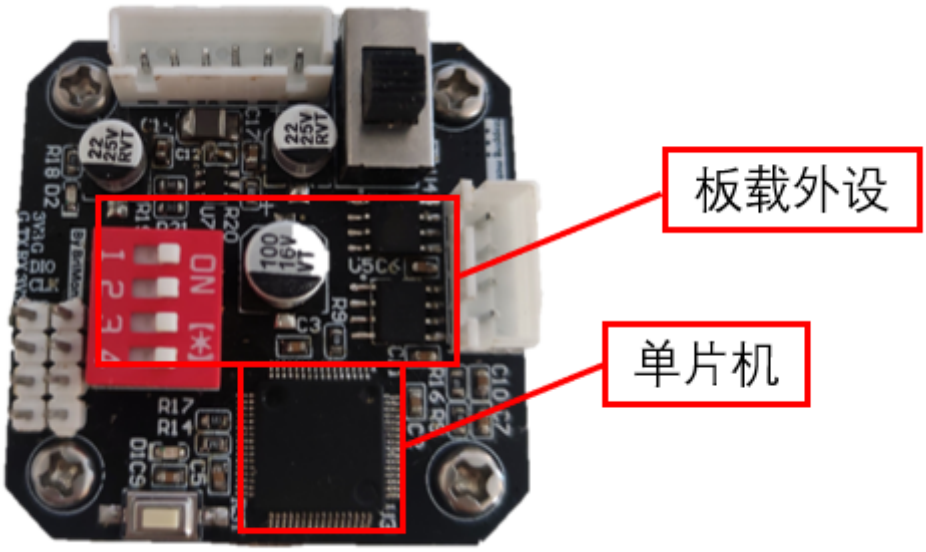
计时这个功能比较有意思，时间这个东西在RTOS还有各种外设中起着重要的作用。使用定时器或者系统时钟可以产生较为准确的时间，根据这些时间可以实现很多有意思的东西。

单片机仅仅是**控制器**的一部分，控制器包含**MCU(单片机)、板载外设**。当然这个控制器就指的是控制智能车或者其他系统的一个硬件整体。

LED、数码管、蜂鸣器、陀螺仪、屏幕、驱动电路、通讯电路等这些都是板载外设。

GPIO、定时器、PWM、URAT、CAN、USB等这些都是**片上外设**。(片上外设的外设是相对于MCU的内核来说的)

下面的图是一个步进电机驱动器，可以简单的区分出来那个是MCU那个是板载外设：



如果你对软件不感兴趣的话了解到这里就已经足够了。

如果你对软件感兴趣你可以看下去：

我们将以STM32为基础进行教学，我们学习一定要注重各种外设、功能的**原理**，当然硬件也是，只有懂得了深层次的原理，再去进行创新、开发才可以游刃有余。智能车竞赛是给你一个平台去应用和更好的掌握这些知识、提高你的工程能力，但是更核心的竞争力仍是你对这个原理有多么的了解。

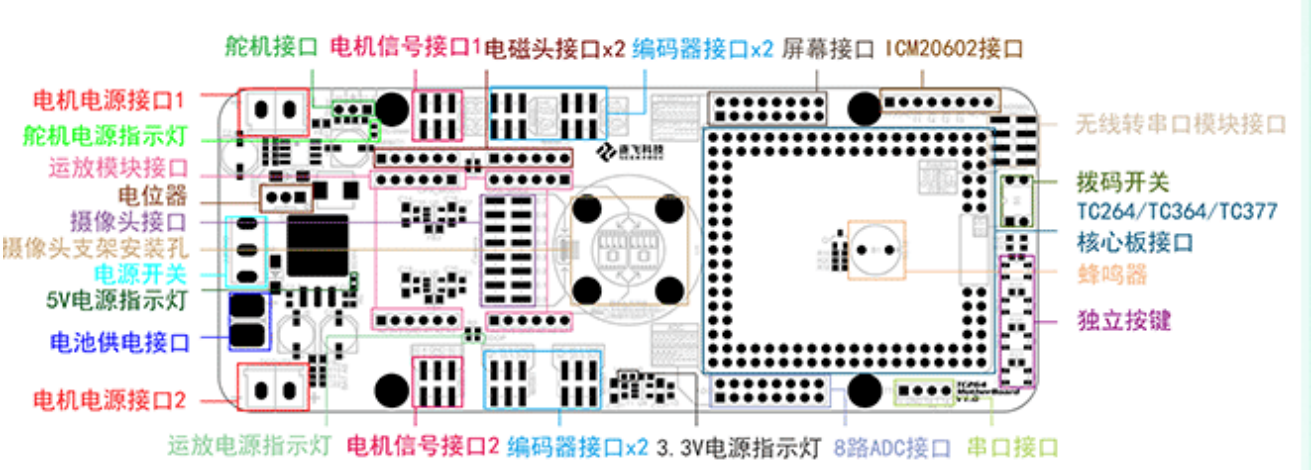
国内芯片标准库基本上和STM32的开发流程一致、外设的原理和设置参数也一致，所以学习STM32很容易就可以迁移到其他型号的MCU上。

我们将会学习**标准库**的开发形式，了解寄存器的开发形式，了解HAL库的开发形式（STM32独有）。当然现在的开发越来越趋向于STM32HAL库的这种开发模式，可以去了解一下CubeMX这个软件，ST把所有的配置图形化了，使得嵌入式工程师从开发底层驱动中解放了出来，更加注重于应用层的设计。但是不是说你用了MX就可以不用去学习标准库了，没有基础的开发终究是不可取的。

标准库开发是个标准化的开发流程，需要开发者去理解外设的各个参数，需要知道什么情况下、什么功能下需要用到什么外设，需要用到这个外设的哪个功能，会设置这些参数。

智能车竞赛需要使用的芯片有**不同的架构和不同的IDE**，比如RISC-V架构、cortex-M架构、8051架构还有英飞凌自己的tri-core架构，其中英飞凌的架构的库函数实现和其他的不是很相同，但是万变不离其宗，整体的学习思路是一致的，我们学习开发最重要的也是学习这个开发的思路和遇到不同问题的解决方法。

下图标注的是智能车需要用到的几乎所有的外设：



仿真器及其协议

了解完单片机是什么我们还要知道常用的仿真器和协议。

为什么要使用仿真器：

- 下载：可以将程序下载到单片机中
- 硬件仿真：单片机仿真器是一种在电子产品开发阶段代替单片机芯片进行软硬件调试的开发工具。配合集成开发环境使用仿真器可以对单片机程序进行单步跟踪调试，也可以使用断点、全速等调试手段，并可观察各种变量、RAM及寄存器的实时数据，跟踪程序的执行情况。利用单片机仿真器可以迅速找到并排除程序中的逻辑错误，大大缩短单片机开发的周期。硬件仿真是开发过程中所必须的。

接口协议

JTAG

JTAG（Joint Test Action Group,联合测试行动小组）是一种国际标准测试协议，用于系统仿真、调试及芯片内部测试。目前JTAG接口有两种连接标准：14针接口及20针接口。

具体的线序可以查看：https://blog.csdn.net/chenhuanqiangnihao/article/details/113835905

SWD

SWD是ARM公司提出的另一种**串行调试接口**，与JTAG相比，SWD只要两根线，分别为：SWCLK和SWDIO。减少了对单片机GPIO口的占用，SWD方式可以在线调试。但是应用范围没有JATG广。

SWDIO–串行数据线，用于数据的读出和写入

SWDCLK–串行时钟线，提供所需要的时钟信号

ARM芯片有两种调试模式，一种是JTAG，一种是SWD，二者在管脚上有复用。SWD模式比JTAG在高速模式下面更加可靠。所以常用SWD进行下载、仿真调试。

常用仿真器

J-Link

J-Link是德国SEGGER公司推出基于JTAG的仿真器。简单地说，是给一个JTAG协议转换盒，即一个小型USB到JTAG的转换盒，其连接到计算机用的是USB接口，而到目标板内部用的还是jtag协议。它完成了一个从软件到硬件转换的工作。

但是正版J-Link也是贵的离谱，甚至可以到千元。

DAP

CMSIS DAP是ARM官方推出的**开源仿真器**，支持所有的**Cortex-A/R/M**器件，支持JTAG/SWD接口。把固件写到单片机里面，单片机就具备了J-LINK的功能。其成本很低。市面上基本所有的离线下载器基本都是基于CMSIS_DAP方案来的，例如正点原子的离线下载器、无线下载器等，还有就是国产单片机厂家做的调试器，例如GD32出的GD-LINK，都是基于CMSIS DAP方案改的。

ST-Link

ST-LINK是专门针对意法半导体STM8和STM32系列芯片的仿真器。开发STM32时也是挺常见的。

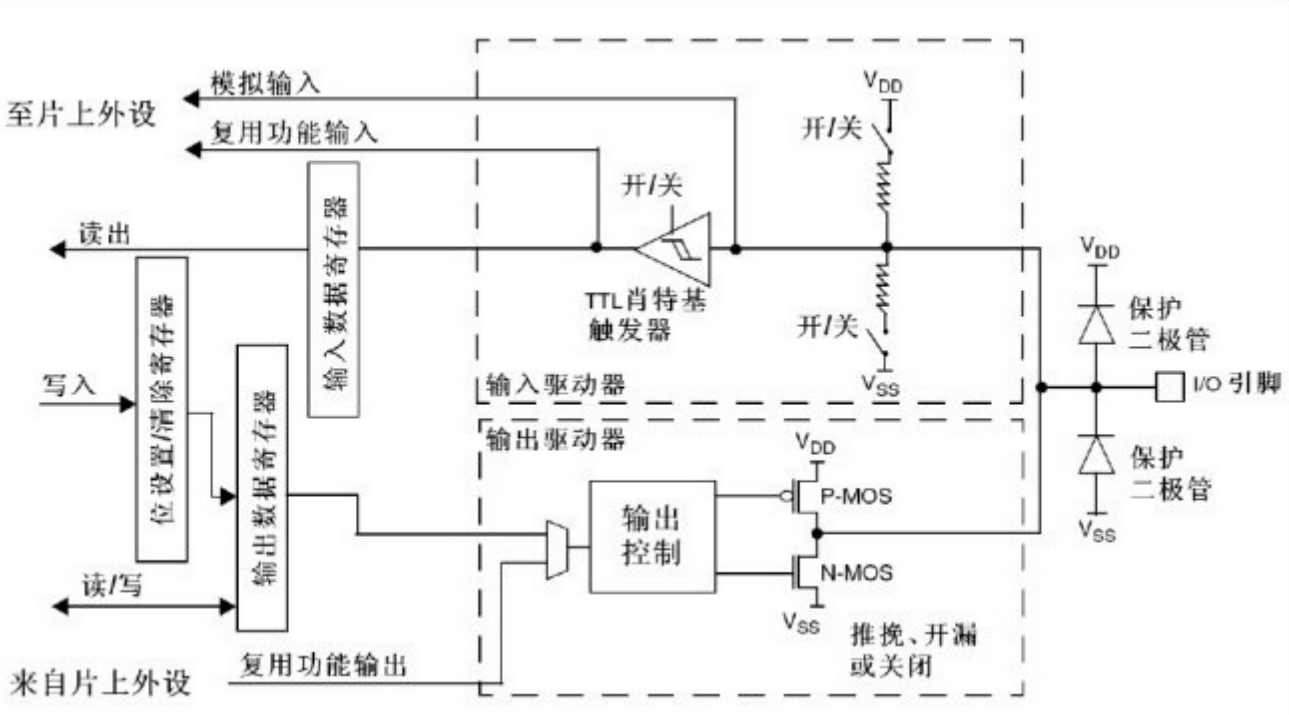
下面是智能车竞赛常用的下载器：

这些下载器为了方便都内置了USB-TTL功能，既可以当做调试器又可以连接串口。

IO典型结构

为啥要了解IO的结构？因为这是单片机连接外界的途径，我们了解了IO口的硬件电路就可以知道单片机是如何与外界沟通的了。

这是STM32 IO口的结构图：



- **保护二极管**：当I/O引脚存在过低的电压，保护二极管会从VSS到I/O给予一个保护电压。当I/O引脚存在过高的电压，保护二极管会从I/O将高电压引到VDD。
- **上下拉电阻**：编程配置控制，当驱动能力不够或者设置I/O引脚初始状态时使用程序配置。配置引脚的上下拉状态。
- **TTL肖特基触发器**：肖特基触发器作用将输入的连续信号转换为离散信号。从图中可以看到模拟输入从肖特基触发器前引出。肖特基触发器开启时引脚状态直接输送到寄存器中，CPU可以直接读取。
- **输出MOS**：控制P-MOS和N-MOS的状态可以控制IO的输出模式

- **推挽输出**：输出高电平时，P-MOS导通；低电平时，N-MOS管导通；两个管子轮流导通，一个负责灌电流，一个负责拉电流。低电平为0v，高电平为3.3V。
- **开漏输出**：输出低电平时，N-MOS管导通，使输出接地，若控制输出1（无法直接输出高电平），则既不输出高电平，也不输出低电平，为高阻态，需要输出高电平时必须外接上拉电阻。如果不接上拉电阻，就失去了输出高电平的能力。

具体的内容可以看这个blog：https://blog.csdn.net/zhuisaozhang1292/article/details/121454387

PWM

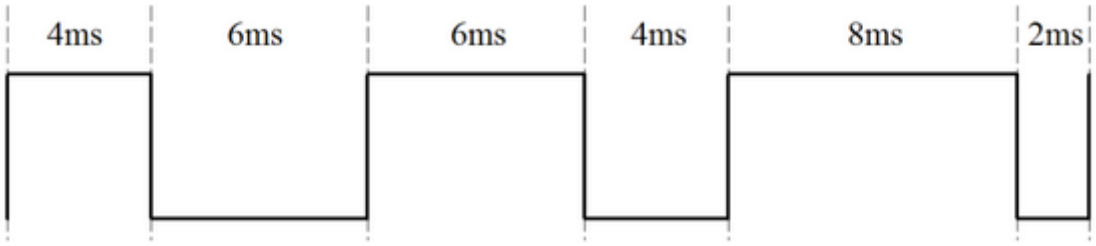
PWM的全称是**脉冲宽度调制**(Pulse-width modulation)，是通过将有效的电信号分散成离散形式从而来降低电信号所传递的平均功率的一种方式。所以根据**面积等效**法则，可以通过对改变脉冲的时间宽度，来等效的获得所需要合成的相应**幅值**和**频率**的波形，PWM就是通过这种原理实现D/A转换的。简单来说就是通过改变占空比的方式来改变输出的**有效电压**。

主要参数：

- 占空比：PWM信号保持高电平的时间百分比称为占空比。如果信号始终为高电平，则它处于100%占空比，如果它始终处于低电平，则占空比为0%。
- 频率：PWM信号的频率决定PWM完成一个周期的速度。

- 幅值：高低电平的电压值差值。

比如下面的PWM波：



第一个PWM周期为10ms，高电平时间为4ms，所以占空比为40%，第二个周期占空比为60%，第三个周期占空比为80%。

作用：

在软件上PWM可以用于电机调速、控制舵机、功率调制、PID调节、通信等。

在硬件上PWM也有很大的作用，例如PWM是开关电源必不可少的调制方式。

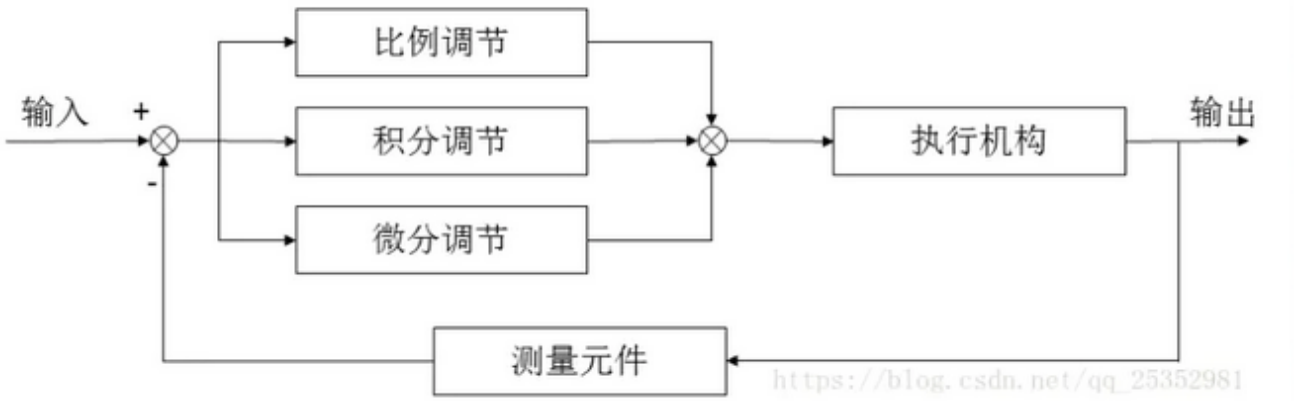
PID

PID控制应该算是应用非常广泛的控制算法了。小到控制一个元件的温度，大到无人机的飞行姿态和飞行速度等等，都可以使用PID控制。为了闭环地控制系统，我们常常引入PID，当然也可以使用现代的控制算法比如ADRC、LQR等，但是PID是使用最为广泛、调参较为方便的一种算法。当然PID也有很多变种，比如分段PID、模糊PID、Bang-Bang PID等。我们在这里仅介绍一下最基础的PID。

PID其实就是指**比例(P)，积分(I)，微分(D)**。

总的来说，当得到系统的输出后，将输出经过比例，积分，微分3种运算方式，叠加到输入中，从而控制系统的行为。

可以通过PID的框图来直观的理解一下这个过程，这样输入到输出的调控就闭环了：



抽象的理解

比例环节

比例控制是最简单的一种控制。比例控制成比例地反映控制系统的偏差信号e(t)，偏差一旦产生，控制器立即产生控制作用，以减小偏差。P参数越大比例作用越强，**动态响应**越快，消除误差的能力越强。但实际系统是有惯性的，控制输出变化后，实际y(t)值变化还需等待一段时间才会缓慢变化。由于实际系统是有惯性的，比例作用不宜太强，比例作用太强会引起系统**振荡**不稳定。

积分环节

控制器的输出与输入误差信号的积分成正比关系。主要用于**消除静差**。积分作用的强弱取决于积分时间常数T,T越大，积分作用越弱，反之则越强。积分作用消除静差的原理是：只要有误差存在，就对误差进行积分，使输出继续增大或减小，一直到误差为零，积分停止，输出不再变化。

微分环节

反映偏差信号的变化趋势，并能在偏差信号变得太大之前，在系统中引入一个有效的早期修正信号，从而**加快系统的动作速度，减少调节时间**。在微分控制中，控制器的输出与输入误差信号的微分（即误差的变化率）成正比关系。不论比例调节作用，还是积分调节作用都是建立在产生误差后才进行调节以消除误差，都是事后调节，因此这种调节对稳态来说是无差的，对动态来说肯定是有差的，因为对于负载变化或给定值变化所产生的扰动，必须等待产生误差以后，然后再来慢慢调节予以消除。但一般的控制系统，不仅对稳定控制有要求，而且对动态指标也有要求，通常都要求负载变化或给定调整等引起扰动后，恢复到稳态的速度要快，因此光有比例和积分调节作用还不能完全满足要求，必须引入微分作用。一发现y(t)有变大或变小的趋势，马上就输出一个阻止其变化的控制信号，以防止出现过冲或超调等。

我们将PID**离散化**后可以得到下面的**公式**：

增量式PID：

$$\Delta u(k) = K_p \times [e(k) - e(k-1)] + K_i \times e(k) + K_d \times [e(k) - 2e(k-1) + e(k-2)] \tag{1}$$

$$u(k) = \sum_{j=0}^{k} \Delta u(j) = u(k-1) + \Delta u(k) \tag{2}$$

位置式PID:

$$u(k) = k_p \times e(k) + K_i \times \sum_{j=0}^{k} e(j) + K_d \times [e(k) - e(k-1)] \tag{3}$$

更形象的理解

你可能认为上面的文字太过抽象，那我们来研究一个实际问题：**如何控制一辆车温度在50km/h?**

低于50km/h踩下油门，高于50km/h踩下刹车，应该没有人会愿意坐你的车。所以，在大多数场合中，用“**开关量**”来控制一个物理量，就显得比较简单粗暴了。有时候，是无法保持稳定的。因为单片机、传感器不是无限快的，采集、控制需要时间。而且，控制对象具有惯性。比如你将一个加热器拔掉，它的“余热”（即热惯性）可能还会使水温继续升高一小会。

PID的输出方向

输出量的方向一定是使误差减小的方向。例如当车速低于50时我们一定是踩油门而不是踩刹车，是使车速增大的方向而不是使车速减小的方向。如果输出量方向取反了，就会导致发散的控制效果。

比例项

我们期望车速为50，而如果当前的车速为49，他们之间的差为1，那我们踩油门的深度就为Kp * 1。当车速比较低加入40，我们踩油门的深度就为Kp * 10。如果速度略高为51，那么我们就去踩Kp * 1的刹车，如果速度为60了，这个时候我们刹车的深度就为Kp * 10。

假设我们从0起步，那么Kp越大，达到50所需要的时间就越短，但是由于人反应有延迟，车有惯性，速度会继续上升，超出50，我们即称为**超调**，超调之后系统会在50上下来回**抖动**，经过一段时间才能稳定。Kp越大，超调量越大，当Kp过于大时，系统会不稳定，汽车可能会熄火。所以我们会限制油门最多能踩多深，即**限制输出的幅度**，超过这个值就不允许了。

我们也注意到，只采用比例项只能稳定在50附近，永远达不到50的速度。因为汽车行驶过程中存在空气阻力，速度达到50时我们不踩油门车速就会下降，最终车速会平衡在一个比50略低的地方，即系统存在**静差**。

积分项

它是把过去的误差经行积分，再乘一个系数作为输出。

回到之前的情景，这时车速已经平衡在了比50低的某个值。这时我们加入积分项，我们会发现踩油门的深度随时间变深，直到速度达到50，达到消除静差的作用。

如果积分系数比较大呢，当速度达到50时由于系统惯性速度超过了50，一段时间积分后又会减速到50以下，以此重复，形成震荡。运气好的情况下速度收敛到50，如果比例系数太大，很有可能导致震荡发散。

但如果我们从0起步的时候加入积分项，由于这时候误差很大，将会产生很大的超调。解决方法有两个：

- 第一个我们设置一个阈值，当差值高于阈值时不进行积分，在域值范围内时才进行积分。这时对域值的选择就有一定要求了，如果太大，则达不到防止超调的效果，如果太小有可能导致不会启动积分。
- 第二个方法我们给积分限幅，防止积分过大导致超调量大。

如果对精度没有太大要求，可以完全去掉积分项，因为积分项会导致**超调**和**震荡**。

微分项

在离散公式里面就是差分，用当前误差减去前一次的误差。

微分项就是指的误差的变化率，微分项的作用即使误差变化率趋于0。对于速度，变化率就是加速度，抑制加速度，就可以抑制速度的变化，减小震荡。

这里我们举弹簧的例子，如果给悬挂重物的弹簧一个力，弹簧就会上下震荡，如果想让震荡停下来，可以给系统加上阻尼，比如放入水中。微分项就是这样一个阻尼的作用。

微分项会**放大误差的噪音**，微分项过大会产生**震荡**。

在低频段，主要是**PI**控制规律起作用，提高系统型别，消除或减少稳态误差；在中高频段主要是**PD**规律起作用，提高响应速度。因此，控制器可以全面地提高系统的控制性能。

所以一个控制过程不一定使用PID，可以单独使用P，或者PI、PD。

end

如果你对软件感兴趣的话可以按照教程安装keil