PID 控制专题

主要内容：

1. 常用的控制算法与 PID 控制算法的异同点；

2.PID 控制算法的理论分析

3.基于单片机的 PID 算法实现

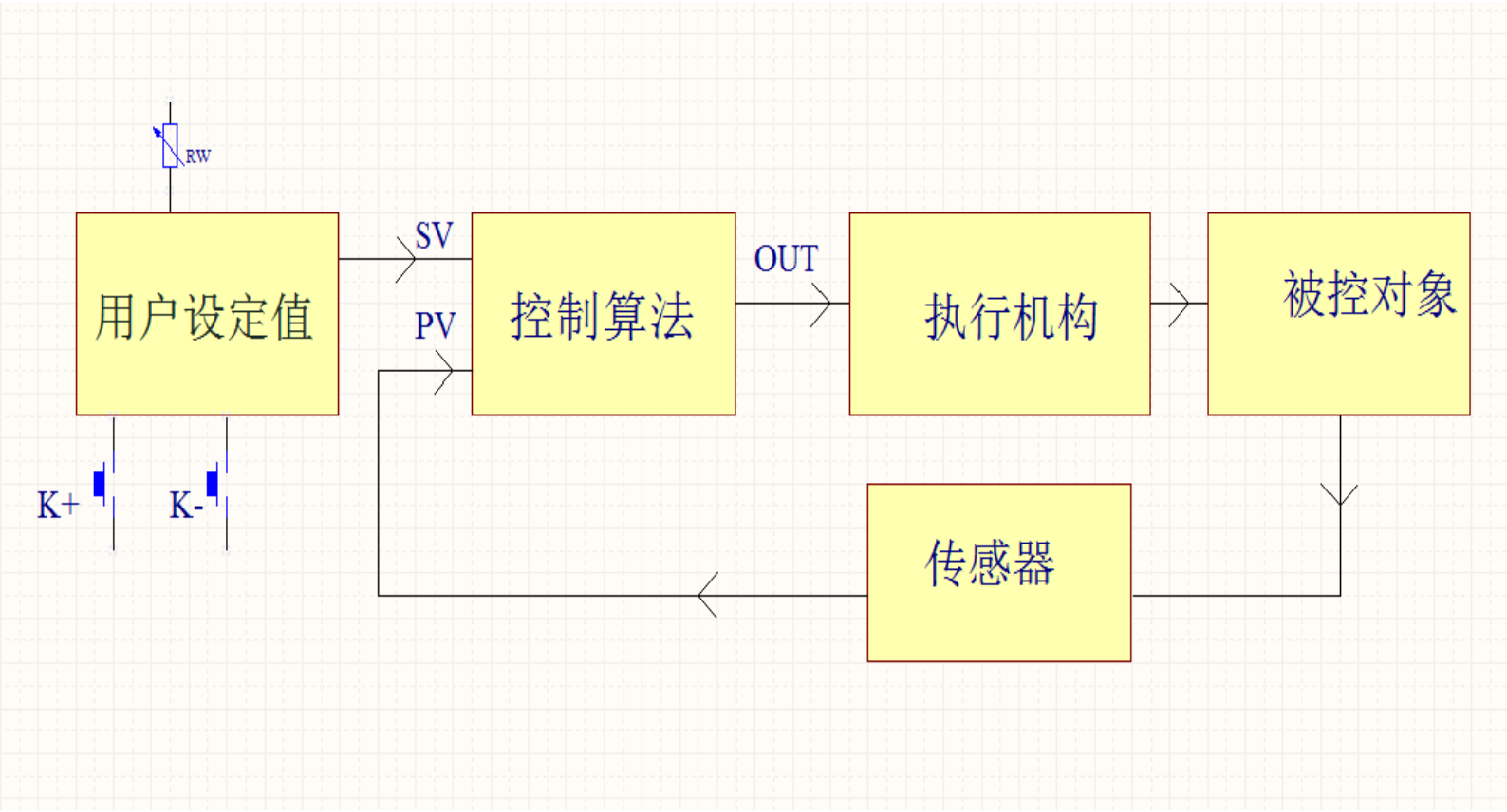
4.PID 算法的工程应用的一些注意事项

5.演示板电路分析

6.PID 算法 C 语言实现 --- 基于 ARM-CortexM3(STM32)的增量式 PID 温度控制

一、常用的控制算法：

1.控制系统的基本结构：



控制目的：

控制的根本目的就是要使控制对象当前的状态值与用户的设定值相同（最大限度的接近）。

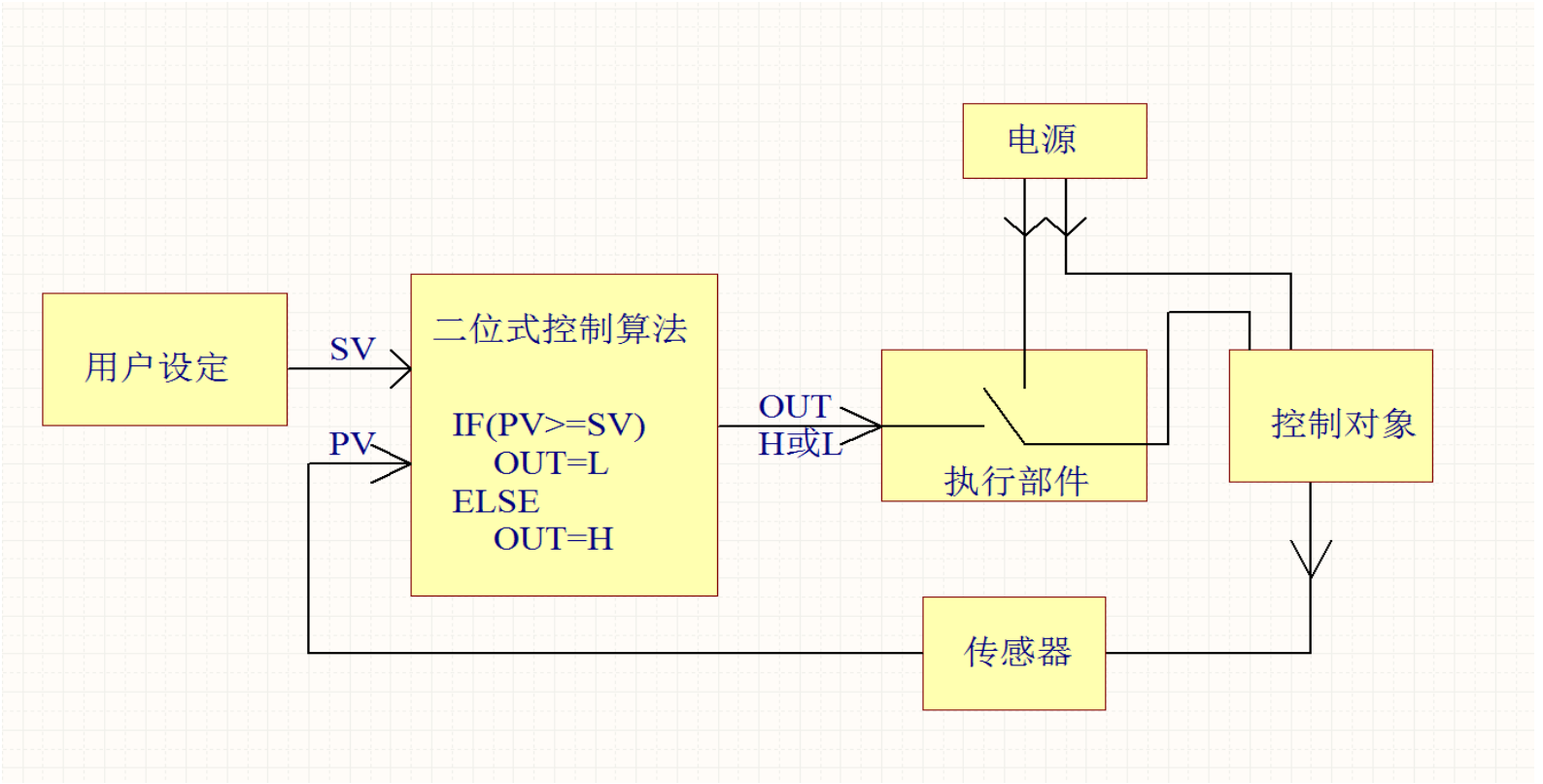
基本思想：

用户设定值 SV 与被控制对象当前的值 PV 两者同时送入由特定硬件电路模型或特定的软件算法组成的控制算法逻辑中，利用不同的控制算法对 SV 和 PV

进行分析、判断、处理，从而产生当前应该输出的控制信号 OUT,控制信号经过执行机构施加到控制对象上，从而产生预期的控制效果。

2.常用控制算法：---位式控制

1).二位式控制算法



特点：

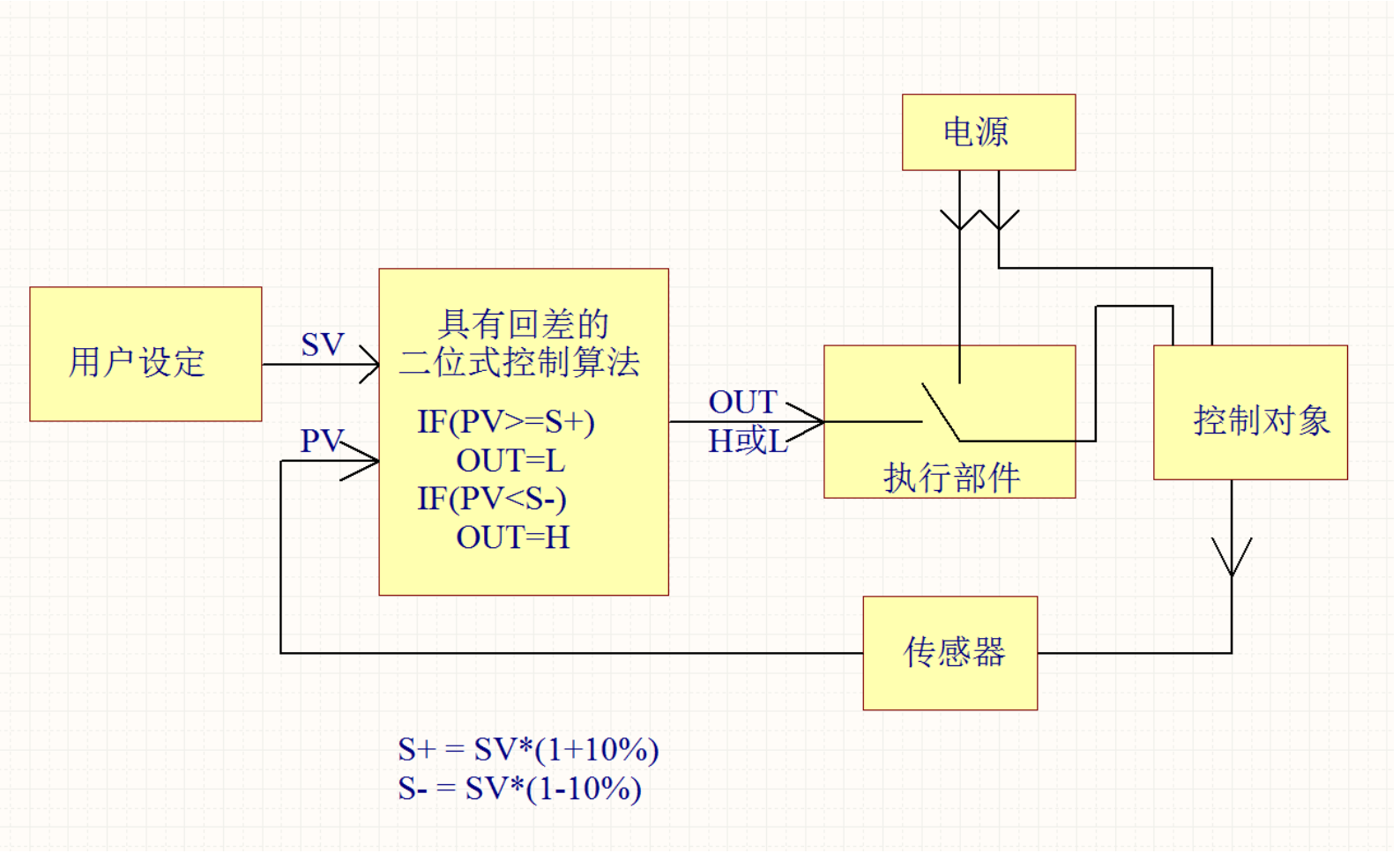
a.二位式控制算法输出的控制量只有高低 2 种状态。

b.执行机构使控制对象要不全额工作，要不就停止工作。当 PV 低于 SV 时全额工作，PV 大于或等于 SV 时就彻底停止工作。如果控制对象是一个 1000W 的加热器，温度不到时就 1000W 全功率运行，温度达到时就停止工作。

c.由于环境因素或控制系统传输延时或者控制对象本身的惯性等因素，控制效果往往是 PV 在 SV 的上下有一个较大的波动。

d.在 PV 接近 SV 的临界点时，控制输出信号 OUT 往往在 H 和 L 之间频繁转换，导致执行部件的触点频繁开关动作，易产生干扰及缩短执行部件的寿命。

2).具有回差的二位式控制算法



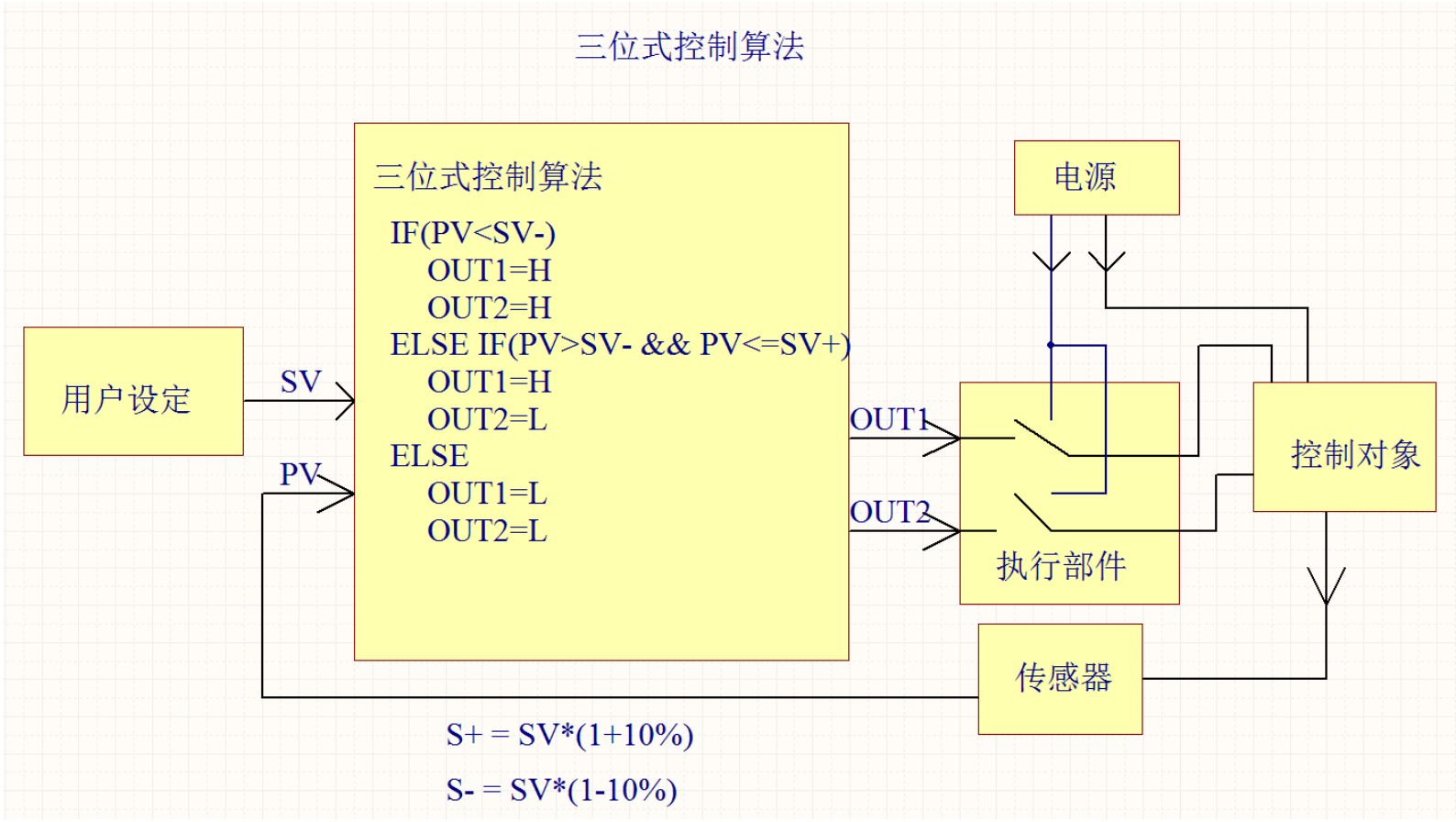
特点：

a. 取 SV 的正负 10%左右作为回差调节上下限，高于上限才开始输出 L,低于下限才开始输出 H;

b.避免了一般二位式控制算法在临界点时执行部件频繁动作。

c.因为控制对象只有全额运行或不运行两种状态，仍然存在一般二位式控制算法的缺点：PV 总是在 SV 附近波动。

3).三位式控制算法



特点：在二位式控制的基础上对控制对象的功率分成 0 功率（停止工作）、半功率、全功率三种情况（即三位）。

当前值低于设定值一定比例（一般 10%）时 OUT1 和 OUT2 同时起控制作用，控制对象全功率运行；

当前值在设定值的正负 10%范围时，OUT1 单独作用，工作于半功率状态；

当前值达到或超过设定值时 OUT1 和 OUT2 都停止输出，控制对象停止工作。

相对一般二位式控制算法，三位式算法对控制对象的当前状态值做了简单的分析，并根据不同的当前状态值输出不同的控制信号。能够较好的对输出产生控制效果。

小结：位式控制的主要特征：

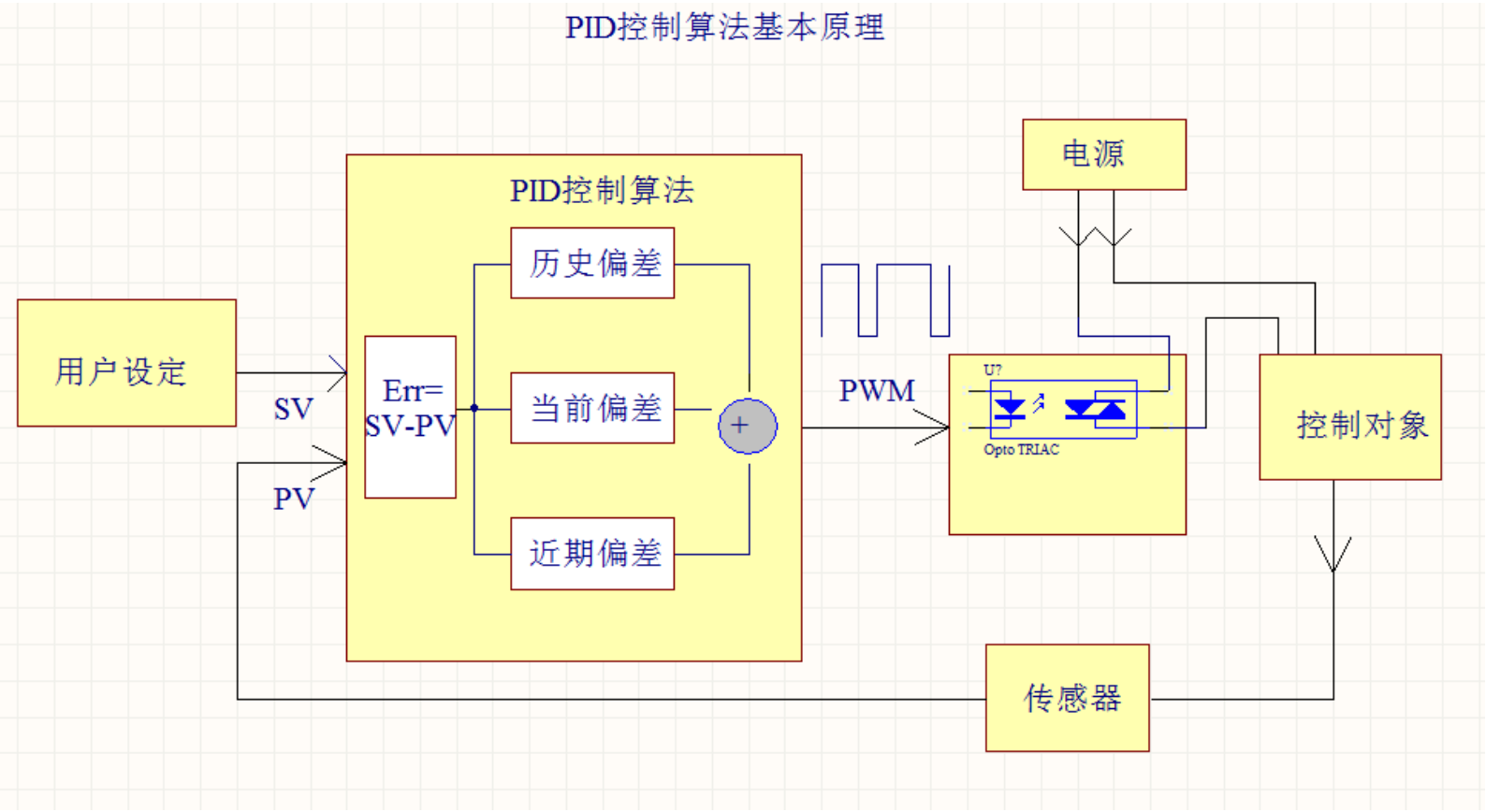
1.控制算法只关注控制当前的状态值（PV）与设定值之间的差值----二者当前有差值就输出控制信号，二者当前无差值就不输出控制信号。

2.位式控制算法的输出信号状态单一，只输出了高低两种状态，这两种状态对应着控制对象的工作与不工作----如果是温度控制系统，就是加热器加热与不加热。

3.由于控制系统自身的延时或者控制对象自身的惯性，位式控制算法只能使控制对象当前的状态值在设定值附件波动，不能很好的跟踪在设定值的附近甚至相等。

PID 控制算法

1. PID 控制算法的基本思想



PID 算法是一种具有预见性的控制算法，其核心思想是：

1>. PID 算法不但考虑控制对象的当前状态值（现在状态），而且还考虑控制对象过去一段时间的状态值（历史状态）和最近一段时间的状态值变化（预期），由这 3 方面共同决定当前的输出控制信号；

2>.PID 控制算法的运算结果是一个数，利用这个数来控制被控对象在多种工作状态(比如加热器的多种功率, 阀门的多种开度等)工作, 一般输出形式为 PWM, 基本上满足了按需输出控制信号，根据情况随时改变输出的目的。

2.PID 算法分析：

设某控制系统：用户设定值为 SV(也就是希望通过 PID 控制算法使被控制对象的状态值保持在 SV 的附件)。

1>从系统投入运行开始，控制算法每隔一段时间对被控制对象的状态值进行采样。由此，可得到开机以来由各个采样时间点被控制对象的状态值所形成的数据序列：

$$X1,X2,\ X3,\ X4,\ \ldots.\ Xk-2,Xk-1,Xk$$

说明：

X1：开机以来的第一次采样值

Xk:　目前的采样值（最近一次的采样值）

2>　从这个采样值序列中提取出三方面信息：

①当前采样值 Xk 与用户设定值 SV 之间的差值：**Ek**

$$Ek = Sv - Xk$$

分析 Ek:

$$Ek \begin{cases} >0: \text{说明当前状态值未达标} \\ =0: \text{说明当前控制状态值正好满足要求} \\ <0: \text{说明当前状态值已经超标} \end{cases}$$

结论：

Ek 反应了控制对象当前值与设定值的偏差程度，可以根据 Ek 的大小对输出信号 OUT 进行调整：偏差程度大 OUT 增大，偏差程度小 OUT 减小。即输出信号的强弱与当前偏差程度的大小成比例，所以根据 Ek 的大小来给出控制信号 OUT 的当前值的算法称为比例控制（Proportion）。

用数学模型可以表示为：

$$POUT=（Kp*Ek）+Out0$$

Kp:一般称之为比例系数，可以理解为硬件上的放大器（或衰减器），适当选取 Kp 将当前误差值 Ek
按一定的增益放大或缩小，以提高控制算法的相应速度。

Out0:是一个常数，目的是为了当 Ek 为 0 时，确保输出信号不为 0，以不至于在当前值与设定值相等时控制器输出信号 OUT 为 0，系统处于无控制信号的失控状态。

②将投入运行以来的各个采样值都与设定值相减，可得到开机以来每个采样时刻的偏差序列数据：

$$E1，E2，E3 .....Ek-2,Ek-1,Ek$$

说明：

E1：开机的第一个采样点与设定值的偏差

E1=SV-X1；

E2=SV-X2；

……

EK-2=SV-XK-2；

EK-1=SV-XK-1；

Ek:当前的采样值与设定值的偏差

EK=SV-XK

分析开机以来的误差序列：

每个偏差值可能有：>0，<0，=0 这三种可能的值，因为从开机到现在，控制算法不断输出控制信号对被控对象进行控制，导致了过去这段时间有时候超标（Ex<0）,有些时候未达标（Ex>0）,有时候正好满足要求（Ex=0）;如果将这些

偏差值进行累加求代数和得到 Sk,即：

$$Sk=E1+E2+E3+.....+Ek-2+Ek-1+Ek$$

分析 Sk:

$$Sk \begin{cases} >0: \text{过去大多数时候未达标} \\ =0: \text{过去控制效果较理想} \\ <0: \text{过去大多数时候已经超标} \end{cases}$$

结论：1.通过对 Sk 的分析，可以对控制算法过去的控制效果进行综合评估。体现了控制算法按照原来的方式输出的控制信号导致了现在的控制结果，所以应该利用这个值来对当前要输出的控制信号 OUT 进行修正，以确保控制对象会在将来的一小段时间尽快达到用户设定的值。

2.Sk 实际上是过去每个时间点的误差相加，与数学上的定积分运算类似，因此根据 Sk 对输出信号进行调节的算法称积分（**integral**）算法。所以积分控制的数学模型为：

$$IOUT=(kp*((1/Ti)\int Exdt))+Out0$$

Kp 是一常数，其目的类似硬件上的放大器，用于将 Sk 放大或衰减；

Out0 是一常数，为了在历史积分偏差值为 0 时确保系统有一个输出值，避免失控；

Ti 是积分时间常数,取值越大会导致输出量OUT会越小,可理解为历史上已经很久的误差值都影响了当前的输出信号。取值越小，输出 OUT 会越强烈，可理解为积分只考虑了最近一段时间的误差。

实际中，如果系统已经运行“很长”一段时间了，那些早期采样的偏差值可以忽略他们对当前控制的影响，所以应该根据情况选择合理的 Ti 值方能得到良好的控制效果。

③最近两次的偏差之差 Dk

$$Dk=Ek-Ek-1$$

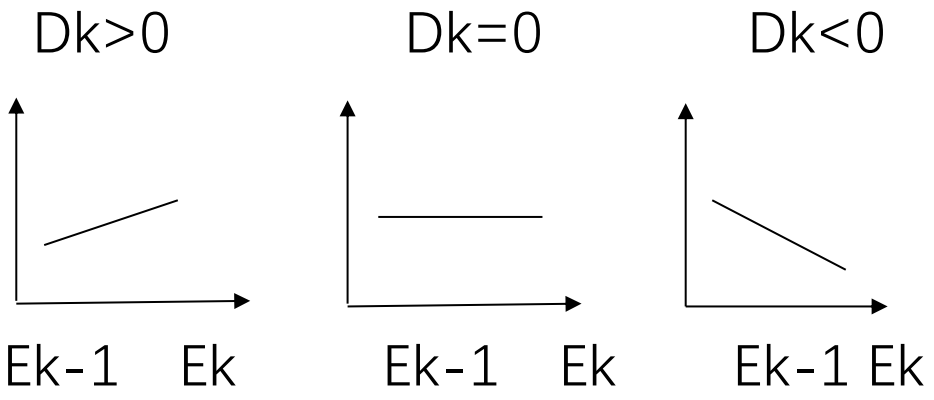
说明：

Ek： 当前的偏差

Ek-1: 基于当前的前一个采样时刻的偏差值 （即上一次的偏差值）；

分析 Dk:

Dk
- >0:说明从上一采样时刻到当前误差有增大趋势
- =0：说明从上一采样时刻到当前误差平稳
- <0:说明从上一采样时刻到当前误差有减小趋势



结论：

1. Dk 能够说明从上次采样到当前采样的这段时间被控制对象的状态变化趋势，这种变化的趋势很可能会在一定程度上延续到下一个采样时间点，所以可以根据这个变化趋势（Dk 的值）对输出信号 OUT 进行调整，达到提前控制的目的。

2. Dk 形如数学上的微分运算，反应了控制对象在一段时间内的变化趋势及变化量，所以利用 Dk 对控制器输出信号进行调节的算法称为微分（**differential**）算法。可以用数学模型表达为：

$$DOUT=Kp*(Td(de/dt))+Out0$$

Kp：为一常数，可理解为硬件上的放大器或衰减器，用于对输出信号 OUT 的增益进行调整；

Out0:为一常数，为了在 Dk 为 0 时确保 OUT 都有一个稳定的控制值，避免失控。

Td:叫微分时间常数，(犹如硬件上电感器的自感系数)Td 越大导致 OUT 增大，对输出信号产生强烈的影响。

3>PID 算法的形成

1.比例、积分、微分三种算法的优缺点分析：

POUT=（Kp*Ek）+ Out0　　--比例算法

IOUT=kp*（（1/Ti）$\int$ Exdt)+Out0　--积分算法

DOUT=Kp*(Td(de/dt))+Out0　　--微分算法

比例算法：只考虑控制对象当前误差，当前有误差才输出控制信号，当前没有误差就不输出控制信号，也就是说只要偏差已经产生了比例算法才采取措施进行调整，所以单独的比例算法不可能将控制对象的状态值控制在设定值上，始终在设定值上下波动；但是比例控制反应灵敏，有误差马上就反应到输出。

积分算法：考虑了被控制对象的历史误差情况，过去的误差状况参与了当前的输出控制，但是在系统还没有达到目标期间，往往会因为这些历史的误差对当前的控制产生了干扰（即拖后腿），使用不当反而搅乱当前的输出。但是在系统进入稳定状态后，特别是当前值与设定值没有偏差时，积分算法可以根据过去的偏差值输出一个相对稳定的控制信号，以防止产生偏离目标，起到打预防针的效果。

微分算法:单纯的考虑了近期的变化率，当系统的偏差趋近于

某一个固定值时（变化率为 0），微分算法不输出信号对其偏差进行调整，所以微分算法不能单独使用，它只关心偏差的变化速度，不考虑是否有偏差（偏差变化率为 0 时偏差不一定是 0）.但是微分算法能获得控制对象近期的变化趋势，它可以协助输出信号尽早的抑制控制对象的变化。可以理解为将要有剧烈变化时就大幅度调整输出信号进行抑制，避免了控制对象的大幅度变化。

以上三种算法综合起来产生一个当前的控制量对控制对象进行控制，它们的优缺点互补，即形成经典的 PID 算法。

2.PID 算法数学模型

OUT= POUT+ IOUT+ DOUT

即：

$$OUT=((Kp*Ek)+Out0)+(kp*((1/Ti)\int Exdt)+Out0)+(Kp*(Td(de/dt))+Out0)$$

整理该式子得到：将各项的 Out0 归并为 OUT0。

$$OUT=kp(\ Ek+\ (\ (1/Ti)\int Exdt))+\ (Td(de/dt)))+OUT0$$

3.PID 算法在单片机中的应用

1） PID 算法在单片机中应用时，对积分和微分项可以作近似变换：

对于积分项可改写成：

$$I=1/Ti\sum_{k=0}^{n} Ek*T$$

即用过去一段时间的采样点的偏差值的代数和的代替积分。

T 是采样周期，也叫控制周期，每隔 T 时间段进行一次 PID 计算。

对于微分项可改写成:

D=TD*((Ek-Ek-1)/T)

Ek：本次偏差，Ek-1 上次的偏差值

2)位置式 PID 算法数学模型

由此可得到单片机中 PID 算法的表达式：

$$OUT=kp(Ek+((1/Ti)\int Exdt))+(Td(de/dt)))+OUT0$$

=>

$$OUT=Kp(En+(1/Ti\sum_{k=0}^{n}Ek*T)+(TD*((Ek-Ek-1)/T)))+out0$$

进一步展开得：

$$OUT=(Kp*Ek)+(Kp*(T/Ti)\sum_{k=0}^{n}Ek)+(Kp*(TD/T)(EK-Ek-1))+OUT0$$

令 Ki= Kp*(T/Ti);

KD=(Kp*(TD/T)

故：

$$OUT=(Kp*Ek) + (Ki\sum_{k=0}^{n}Ek) + (KD(EK-Ek-1)) + OUT0$$

程序设计时利用 C 语言或汇编语言可以方便实现这个计算公式。OUT 即为本次运算的结果，利用 OUT 可以去驱动执行机构输出对应的控制信号，例如温度控制就可以控制 PWM 的宽度，电磁阀就可以改变电磁线圈电流以改变阀门开度，或者是可控硅的导通角度等；

这种 PID 算法计算出的结果（OUT 值）表示当前控制器应该输出的控制量，所以称为位置式（直接输出了执行机构应该达到的状态值）。

3）增量式 PID 算法

位置式 PID 算法计算量较大，比较消耗处理器的资源。在有些控制系统中，执行机构本身没有记忆功能，比如 MOS 管是否导通完全取决于控制极电压，可控硅是否导通取决于触发信号，继电器是否接通取决于线圈电流等，只要控制信号丢失，执行机构就停止，在这些应用中应该采用位置式 PID。

也有一些执行机构本身具有记忆功能，比如步进电机，即使控制信号丢失，由于其自身的机械结构会保持在原来的位置等，在这些控制系统中，PID 算法没有必要输出本次应

该到达的真实位置，只需要说明应该在上次的基础上对输出信号做多大的修正（可正可负）即可，这就是增量式 PID 算法。

增量式 PID 计算出的是应该在当前控制信号上的调整值，如果计算出为正，则增强输出信号；如果计算出为负则减弱输出信号。

增量式 PID 算法数学模型：

如果用 OUTK-1 表示上次的输出控制信号值，那么当前的输出值应该为 OUTk,这两者之间的关系为：

$$OUTK=OUTk-1+\triangle OUT$$

$\triangle$OUT 即为应该输出的增量值；

上式变形得：

$$\triangle OUT= OUTK- OUTk-1$$

本次的位置式算法输出：

$$OUTk=(Kp*Ek) + (Ki\sum_{k=0}^{n} Ek) + (KD(EK-Ek-1)) +OUT0 \quad \text{--1 式}$$

上次的位置式算法输出：

$$OUTk-1=(Kp*Ek-1) +(Ki\sum_{k=0}^{n-1} Ek) + (KD(EK-1-Ek-2)) +OUT0 \quad \text{--2 式}$$

上述 1 式减 2 式即得到相邻两次的增量：

如前所述：

Ki= Kp*(T/Ti);

KD=(Kp*(TD/T)

△OUT= OUTK- OUTk-1 =

kp(EK-EK-1)+((Kp*T)/Ti)Ek+(((Kp*TD)/T)*(Ek-2Ek-1+Ek-2))

EK： 本次的偏差；

Ek-1：上次的偏差

Ek-2：上上次的偏差

Kp： 算法增益调节

Ti ： 积分时间

TD: 微分时间常数

结论：

增量式 PID 的计算只需要最近 3 次的偏差（本次偏差，上次偏差，上上次偏差），不需要处理器存储大量的历史偏差值，

计算量也相对较少，容易实现。

4）关于 Ti 和 TD 的理解：

在 PID 控制算法中，当前的输出信号由比例项，积分项，微分项共同作用形成，当比例项输出不为 0 时，如果积分项对运算输出的贡献作用与比例项对运算对输出的贡献一样时（即同为正或同为负时），积分项相当于重复了一次比例项产生作用的时间，这个

时间就可以理解为积分时间。

当比例项不为 0 时，如果微分项在一段时间里计算的结果与比例项对输出的贡献相同（即同为正或同为负）时，微分项相当于在一段时间里重复了比例项的作用，这段时间可理解为就是微分时间。

实际应用中应该合理选择 Kp,Ti,Td 以确保三者对输出的贡献平衡，从而使控制对象在设定值的附近。