Digital Controllers

Prof. Anjan Rakshit and Dr. Amitava Chatterjee
Electrical Measurements and Instrumentation Laboratory,
Electrical Engineering Department,
Jadavpur University, Kolkata, India.

Digital Controller in a Process Control Loop

Digital Controller in a Process Control Loop

All the desirable features of a process controller *(e.g. anti-integral wind-up, auto/manual modes of operation with bump-less transfer etc.)* **may be easily incorporated while maintaining the high accuracy and precision of digital systems.**

Digital Controller in a Process Control Loop

All the desirable features of a process controller *(e.g. anti-integral wind-up, auto/manual modes of operation with bump-less transfer etc.)* **may be easily incorporated while maintaining the high accuracy and precision of digital systems.**

No drift problem is encountered in a digital system.

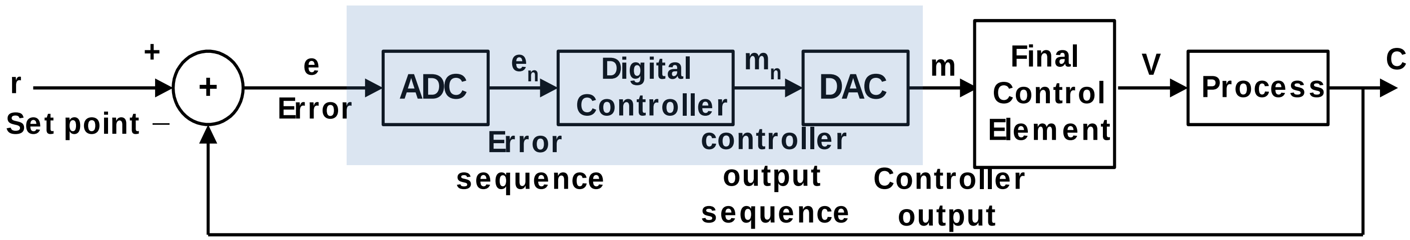
Digital Controller in a Process Control Loop

All the desirable features of a process controller *(e.g. anti-integral wind-up, auto/manual modes of operation with bump-less transfer etc.)* **may be easily incorporated while maintaining the high accuracy and precision of digital systems.**

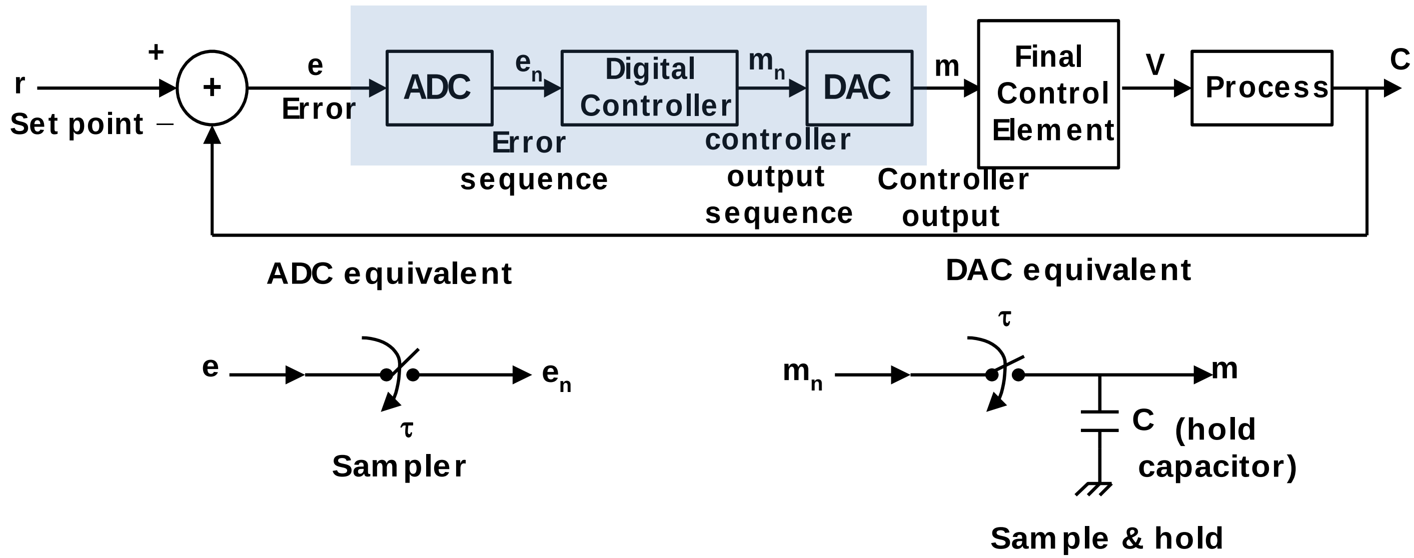
No drift problem is encountered in a digital system.

In a processor based digital controller, rapid switching from one algorithm to another *(e.g. a P controller to a PID controller)* **and automatic tuning of controller parameters are possible.**

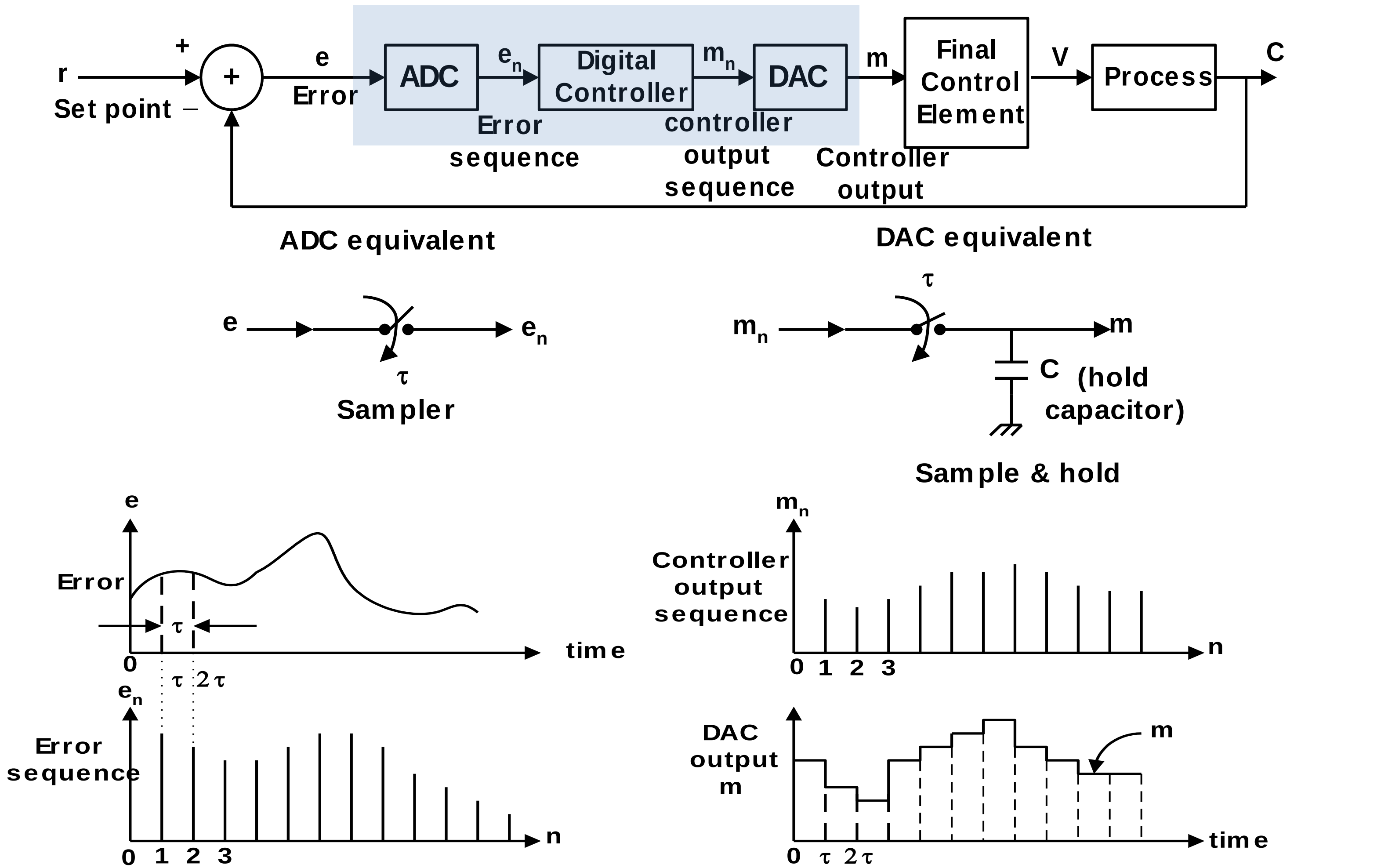
Digital Controller in a Process Control Loop



Digital Controller in a Process Control Loop



Digital Controller in a Process Control Loop



τ → Sampling interval

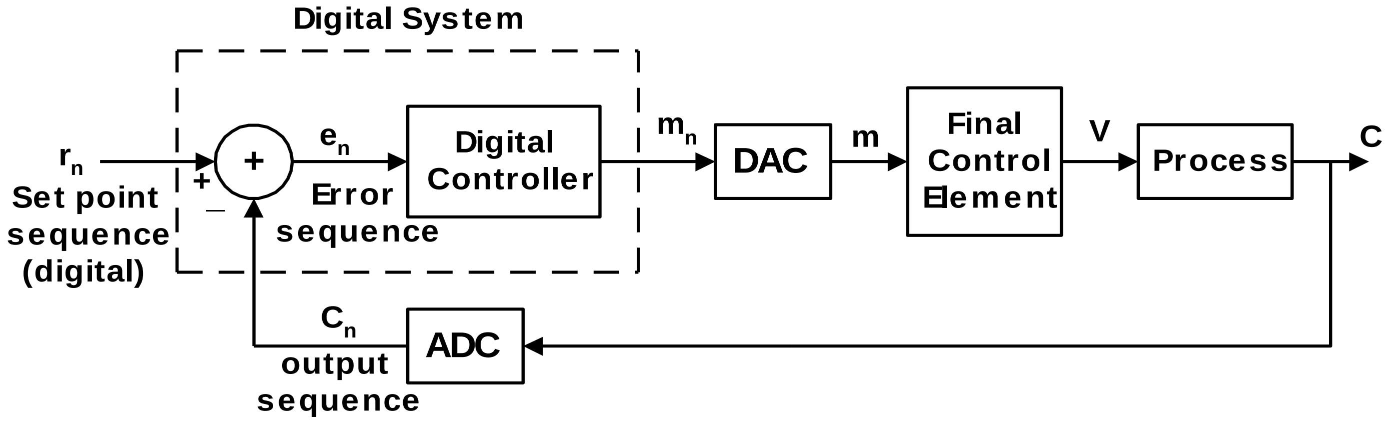
τ ≥ (ADC conversion time + computation time)

Digital Controller in a Process Control Loop

The error computation may be performed digitally if the set-point is available in digital form (say, from digital keyboard) and the measured variable is digitized with the ADC.

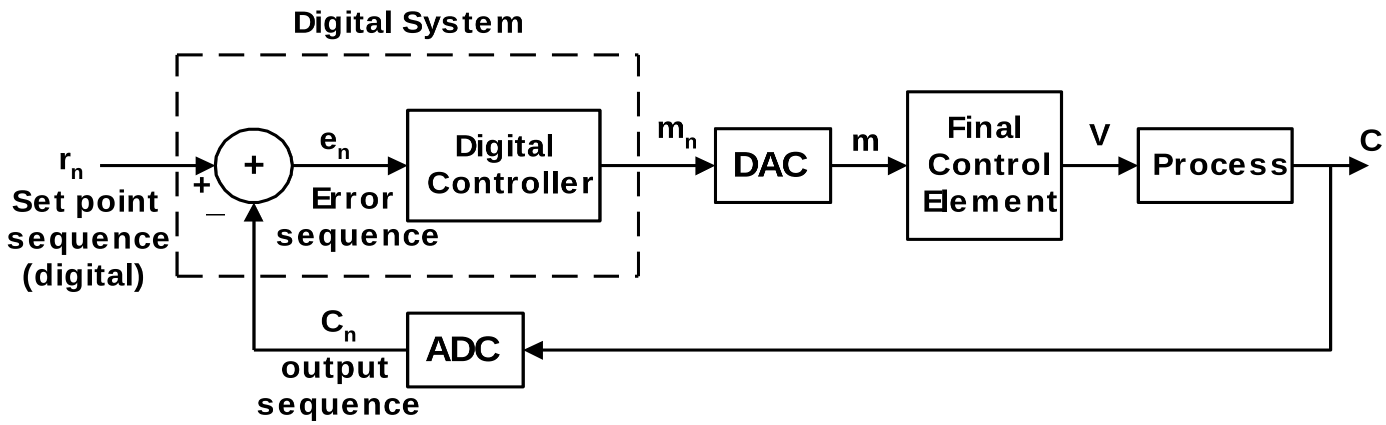
Digital Controller in a Process Control Loop

The error computation may be performed digitally if the set-point is available in digital form (say, from digital keyboard) and the measured variable is digitized with the ADC.



Digital Controller in a Process Control Loop

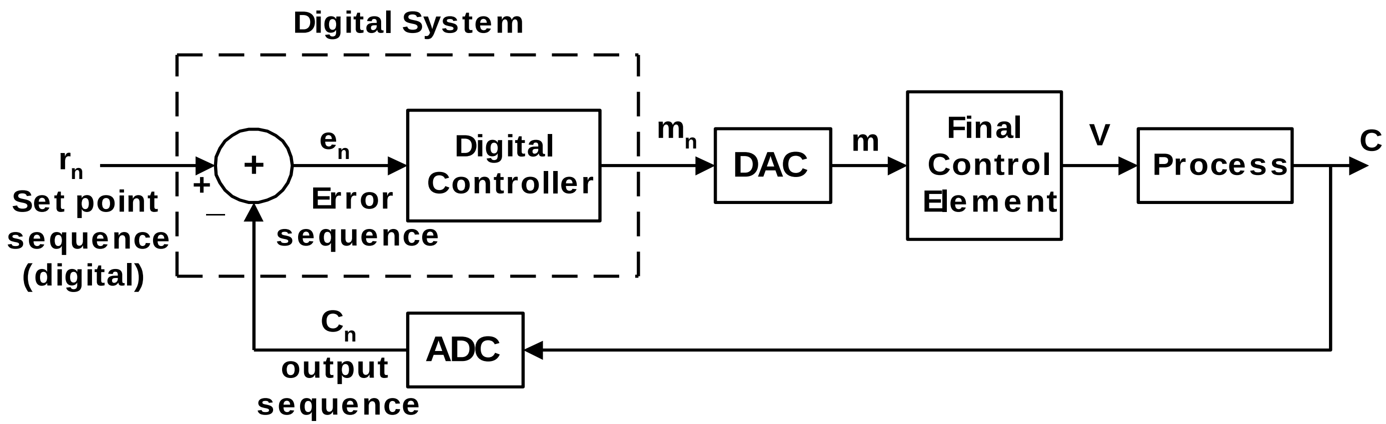
The error computation may be performed digitally if the set-point is available in digital form (say, from digital keyboard) and the measured variable is digitized with the ADC.



Proper selection of the **sampling interval 'τ'** is necessary for satisfactory operation of the process control loop.

Digital Controller in a Process Control Loop

The error computation may be performed digitally if the set-point is available in digital form (say, from digital keyboard) and the measured variable is digitized with the ADC.



Proper selection of the **sampling interval 'τ'** is necessary for satisfactory operation of the process control loop.

A large 'τ' may lead to **unstable operation** of the loop (because of the extra lag introduced in the loop), whereas a very small 'τ' requires a **high speed digital hardware** (hence high cost) to implement the controller.

Realization of Digital Controllers
(through discrete approximation of analog controllers)

Realization of Digital Controllers
(through discrete approximation of analog controllers)

Proportional (P) Controller

The analog controller output is

$$m = K_p e + b$$

Realization of Digital Controllers
(through discrete approximation of analog controllers)

Proportional (P) Controller

The analog controller output is

$$m = K_p e + b$$

The controller output at the nth instant is given by

$$m_n = K_p e_n + b_n = m_n' + b_n$$

where, K_p = proportional gain,
b_n = fixed bias.

Realization of Digital Controllers
(through discrete approximation of analog controllers)

Proportional (P) Controller

The analog controller output is

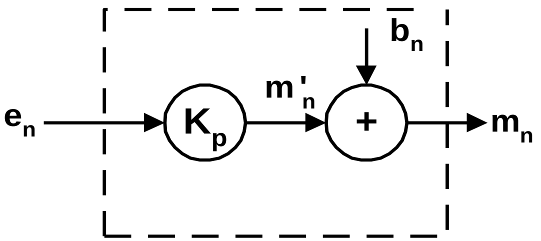
$$m = K_p e + b$$

The controller output at the nth instant is given by

$$m_n = K_p e_n + b_n = m'_n + b_n$$

where, K_p = proportional gain,
b_n = fixed bias.

Realization of the P controller:



Realization of Digital Controllers
(through discrete approximation of analog controllers)

Proportional-Integral (PI) Controller

The analog controller output is

$$m = K_p \left[e + \frac{1}{T_i} \int_o^t e.dt \right] + b$$

Realization of Digital Controllers
(through discrete approximation of analog controllers)

Proportional-Integral (PI) Controller

The analog controller output is

$$m = K_p \left[e + \frac{1}{T_i} \int_o^t e.dt \right] + b$$

$= m' + b$, where T_i is the integral time and

$$m' = K_p \left[e + \frac{1}{T_i} \int_o^t e.dt \right]$$

Realization of Digital Controllers
(through discrete approximation of analog controllers)

Proportional-Integral (PI) Controller

The analog controller output is

$$m = K_p \left[e + \frac{1}{T_i} \int_o^t e.dt \right] + b$$

$= m' + b$, where T_i is the integral time and

$$m' = K_p \left[e + \frac{1}{T_i} \int_o^t e.dt \right]$$

Let the integration term be $I = \int_o^t e.dt$

Realization of Digital Controllers
(through discrete approximation of analog controllers)

Proportional-Integral (PI) Controller

The analog controller output is

$$m = K_p\left[e + \frac{1}{T_i}\int_o^t e.dt\right] + b$$

= $m' + b$, where T_i is the integral time and $$m' = K_p\left[e + \frac{1}{T_i}\int_o^t e.dt\right]$$

Let the integration term be $$I = \int_o^t e.dt$$

The integration term at the nth instant is $$I_n = \int_o^{n\tau} e.dt$$ where τ is the sampling interval

Realization of Digital Controllers
(through discrete approximation of analog controllers)

Proportional-Integral (PI) Controller

The analog controller output is

$$m = K_p \left[e + \frac{1}{T_i} \int_o^t e.dt \right] + b$$

$= m' + b$, where T_i is the integral time and

$$m' = K_p \left[e + \frac{1}{T_i} \int_o^t e.dt \right]$$

Let the integration term be $I = \int_o^t e.dt$

The integration term at the nth instant is $I_n = \int_o^{n\tau} e.dt$ where τ is the sampling interval

$$= \int_o^{n\tau-\tau} e.dt + \int_{n\tau-\tau}^{n\tau} e.dt$$ for n = 1,2,3,..

Realization of Digital Controllers
(through discrete approximation of analog controllers)

Proportional-Integral (PI) Controller

The analog controller output is

$$m = K_p\left[e + \frac{1}{T_i}\int_o^t e.dt\right] + b$$

$= m' + b$, where T_i is the integral time and

$$m' = K_p\left[e + \frac{1}{T_i}\int_o^t e.dt\right]$$

Let the integration term be $I = \int_o^t e.dt$

The integration term at the nth instant is $I_n = \int_o^{n\tau} e.dt$ where τ is the sampling interval

$$= \int_o^{n\tau-\tau} e.dt + \int_{n\tau-\tau}^{n\tau} e.dt$$ for n = 1,2,3,..

Now, $\int_o^{n\tau-\tau} e.dt = \int_o^{(n-1)\tau} e.dt = I_{n-1}$

Realization of Digital Controllers
(through discrete approximation of analog controllers)

Proportional-Integral (PI) Controller

The analog controller output is

$$m = K_p\left[e + \frac{1}{T_i}\int_o^t e.dt\right] + b$$

$= m' + b$, where T_i is the integral time and $m' = K_p\left[e + \frac{1}{T_i}\int_o^t e.dt\right]$

Let the integration term be $I = \int_o^t e.dt$

The integration term at the nth instant is $I_n = \int_o^{n\tau} e.dt$ where τ is the sampling interval

$$= \int_o^{n\tau-\tau} e.dt + \int_{n\tau-\tau}^{n\tau} e.dt \quad \text{for n = 1,2,3,..}$$

Now, $\int_o^{n\tau-\tau} e.dt = \int_o^{(n-1)\tau} e.dt = I_{n-1}$

Then, $I_n = I_{n-1} + \int_{(n-1)\tau}^{n\tau} e.dt$

PI Controller

$$I_n = I_{n-1} + \int_{(n-1)\tau}^{n\tau} e.dt$$

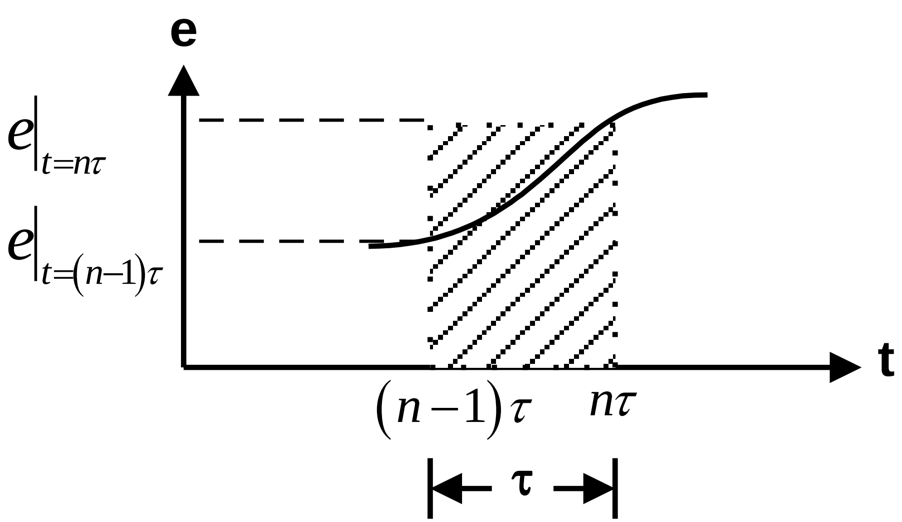
The second term of the above relation represents the area under the curve 'e' for $(n-1)\tau \leq t \leq n\tau$.

PI Controller

$$I_n = I_{n-1} + \int_{(n-1)\tau}^{n\tau} e.dt$$

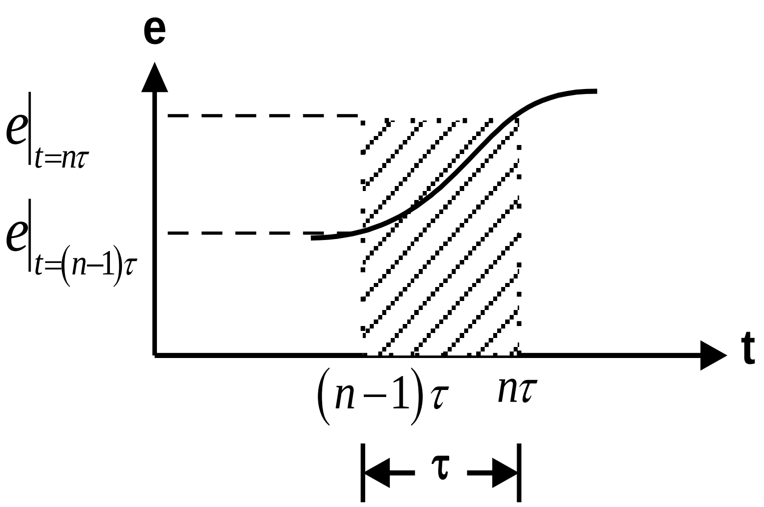
The second term of the above relation represents the area under the curve 'e' for $(n - 1)\tau \leq t \leq n\tau$.

This area may be approximated by the shaded rectangle (called the **method of rectangular integration**) as



PI Controller

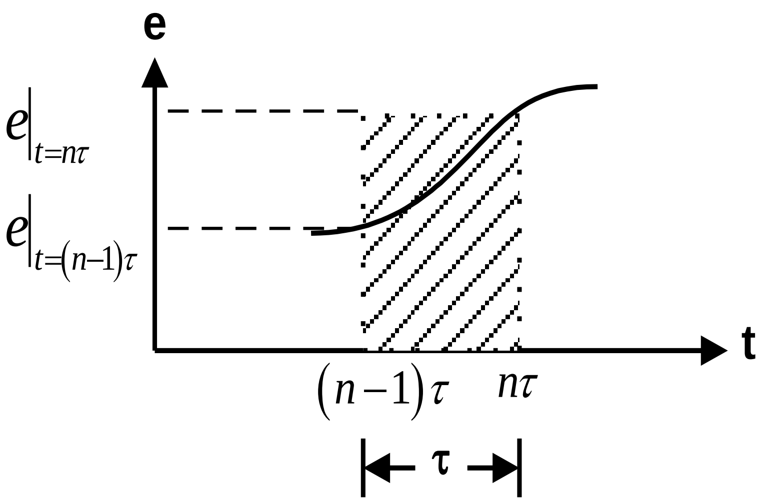
$$I_n = I_{n-1} + \int_{(n-1)\tau}^{n\tau} e.dt$$



Thus, $\int_{(n-1)\tau}^{n\tau} e.dt \approx \tau e|_{t=n\tau}$

PI Controller

$$I_n = I_{n-1} + \int_{(n-1)\tau}^{n\tau} e.dt$$



Thus, $$\int_{(n-1)\tau}^{n\tau} e.dt \approx \tau e|_{t=n\tau}$$

Using the notation e_n for $e|_{t=n\tau}$ **,** $$I_n = I_{n-1} + \tau e_n$$

PI Controller

Now, output m′ at the nth instant may be expressed as

$$m'_n = K_p \left[e_n + \frac{I_n}{T_i} \right]$$ where $$I_n = I_{n-1} + \tau e_n$$

PI Controller

Now, output m′ at the nth instant may be expressed as

$$m'_n = K_p\left[e_n + \frac{I_n}{T_i}\right] \quad \text{where} \quad I_n = I_{n-1} + \tau e_n$$

Similarly the output at the (n – 1)th instant may be expressed as

$$m'_{n-1} = K_p\left[e_{n-1} + \frac{I_{n-1}}{T_i}\right]$$

PI Controller

Now, output m′ at the nth instant may be expressed as

$$m'_n = K_p \left[e_n + \frac{I_n}{T_i} \right] \quad \text{where} \quad I_n = I_{n-1} + \tau e_n$$

Similarly the output at the (n – 1)th instant may be expressed as

$$m'_{n-1} = K_p \left[e_{n-1} + \frac{I_{n-1}}{T_i} \right]$$

The difference between these two outputs is

$$m'_n - m'_{n-1} = K_p \left[e_n - e_{n-1} + \frac{1}{T_i} \left(I_n - I_{n-1} \right) \right]$$

PI Controller

$$m'_n - m'_{n-1} = K_p\left[e_n - e_{n-1} + \frac{1}{T_i}\left(I_n - I_{n-1}\right)\right]$$

Substituting the value of $\boldsymbol{(I_n - I_{n-1})}$ for rectangular integration, $\boldsymbol{[(I_n - I_{n-1}) = \tau e_n]}$

PI Controller

$$m'_n - m'_{n-1} = K_p\left[e_n - e_{n-1} + \frac{1}{T_i}\left(I_n - I_{n-1}\right)\right]$$

Substituting the value of **($I_n - I_{n-1}$)** for rectangular integration, **[($I_n - I_{n-1}$) = τe_n]**

$$m'_n - m'_{n-1} = K_p\left[e_n - e_{n-1} + \frac{\tau e_n}{T_i}\right]$$

PI Controller

$$m'_n - m'_{n-1} = K_p\left[e_n - e_{n-1} + \frac{1}{T_i}\left(I_n - I_{n-1}\right)\right]$$

Substituting the value of $\boldsymbol{(I_n - I_{n-1})}$ for rectangular integration, $\boldsymbol{[(I_n - I_{n-1}) = \tau e_n]}$

$$m'_n - m'_{n-1} = K_p\left[e_n - e_{n-1} + \frac{\tau e_n}{T_i}\right]$$

or, $$m'_n = e_n K_p\left(1 + \frac{\tau}{T_i}\right) - K_p e_{n-1} + m'_{n-1}$$

PI Controller

$$m'_n - m'_{n-1} = K_p\left[e_n - e_{n-1} + \frac{1}{T_i}\left(I_n - I_{n-1}\right)\right]$$

Substituting the value of $\mathbf{(I_n - I_{n-1})}$ for rectangular integration, $\mathbf{[(I_n - I_{n-1}) = \tau e_n]}$

$$m'_n - m'_{n-1} = K_p\left[e_n - e_{n-1} + \frac{\tau e_n}{T_i}\right]$$

or, $$m'_n = e_n K_p\left(1 + \frac{\tau}{T_i}\right) - K_p e_{n-1} + m'_{n-1}$$

$$= a_o e_n + a_1 e_{n-1} + m'_{n-1}, \qquad \text{say}$$

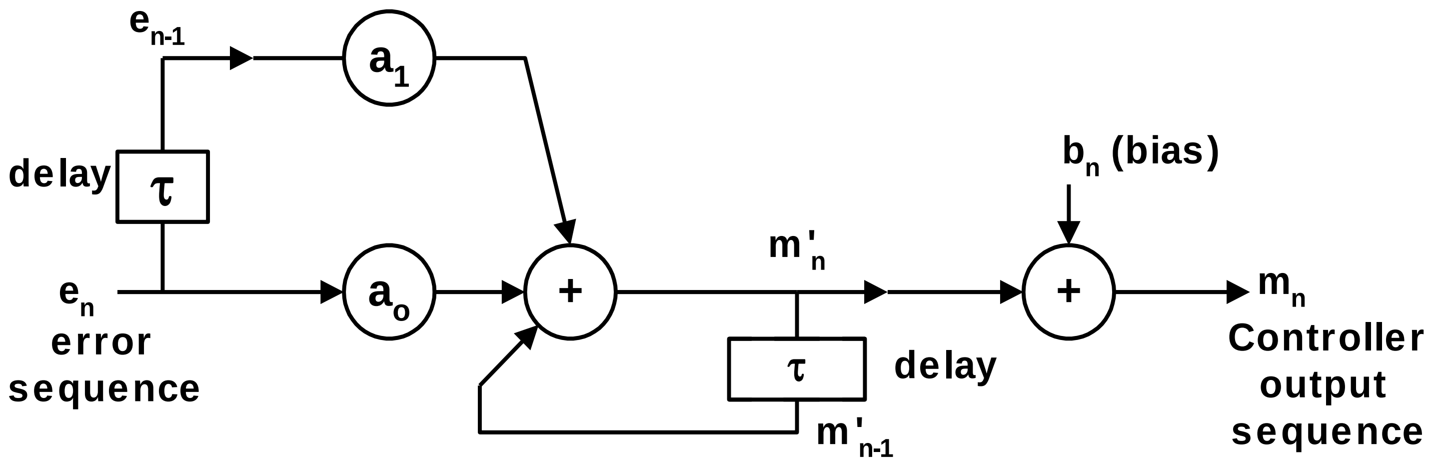
where, $a_o = K_p\left[1 + \dfrac{\tau}{T_i}\right]$ and $a_1 = -K_p$

Realization of the PI Controller

Controller output at the nth instant:

$$m_n = m'_n + b_n$$

$$m'_n = a_0 e_n + a_1 e_{n-1} + m'_{n-1}$$

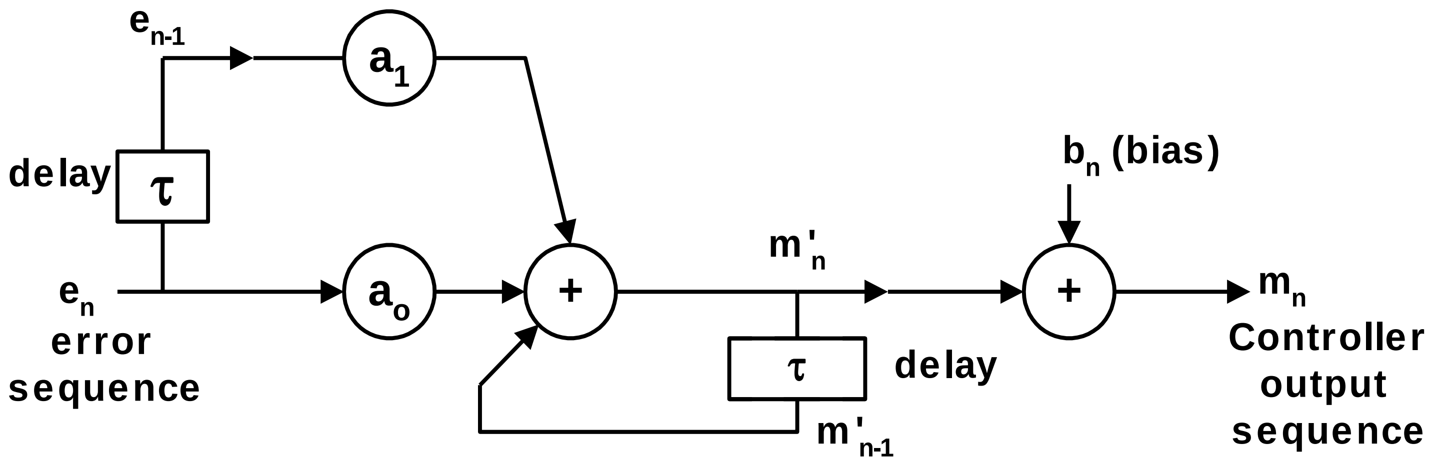


Realization of the PI Controller

Controller output at the nth instant:

$$m_n = m'_n + b_n$$

$$m'_n = a_0 e_n + a_1 e_{n-1} + m'_{n-1}$$



Problem: Develop a digital PI Controller using Trapezoidal rule for integration.

Velocity or incremental form of PI controller

The controller output is proportional to the derivative of a standard PI controller and it may be expressed as (without bias):

$$\Delta m_n = m_n - m_{n-1} = m'_n - m'_{n-1} = a_o e_n + a_1 e_{n-1}$$

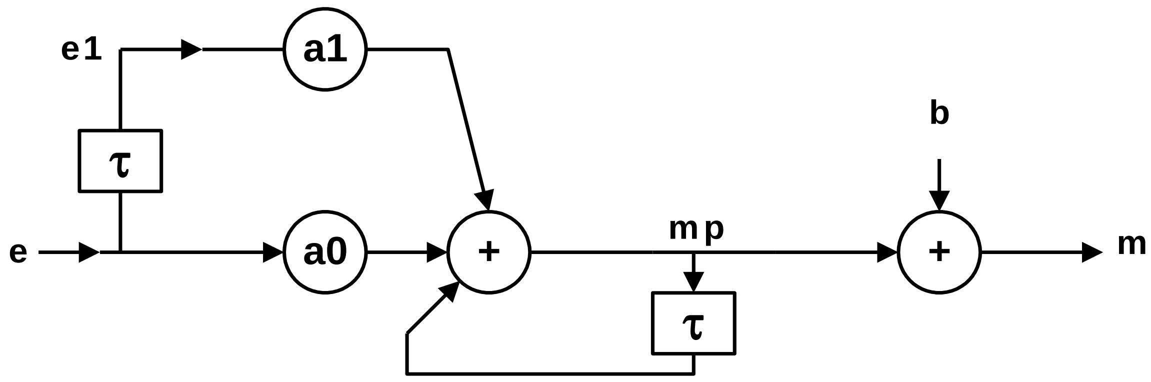
Velocity or incremental form of PI controller

The controller output is proportional to the derivative of a standard PI controller and it may be expressed as (without bias):

$$\Delta m_n = m_n - m_{n-1} = m'_n - m'_{n-1} = a_o e_n + a_1 e_{n-1}$$

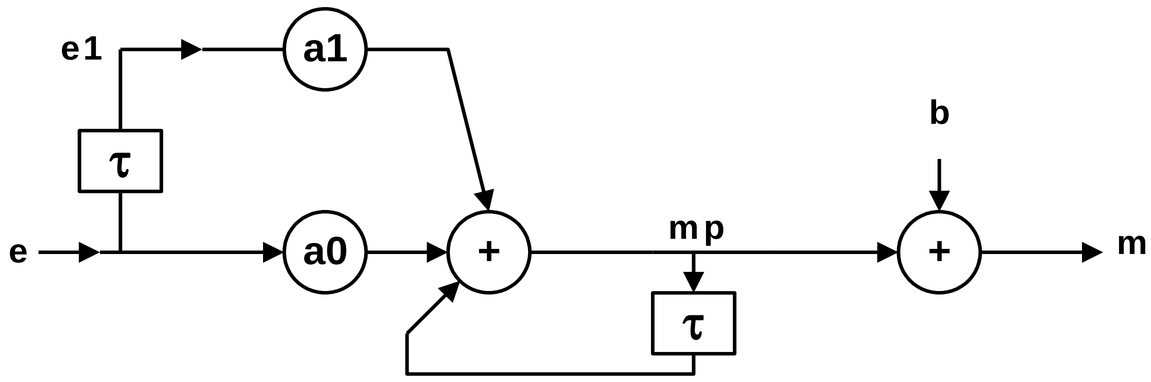
Velocity form of controller is useful when the actuator is some kind of adder (integral action), like a stepping motor.

Software Realization of the PI Controller



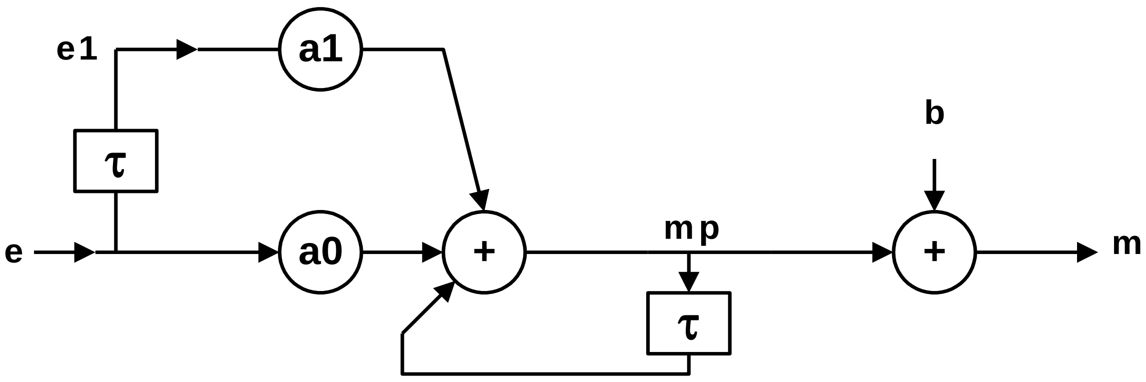

```
# include < studio.h>
void main (void)
      {
      float e = 0, e1, m, mp = 0;
      float a0, a1, b;
      float adc (void) ;        // digitized error
      void dac (float m ) ;   // analog output
      a0 = - - - - - - - - -  ;   // Kp [1 + τ/Ti ]
      a1 = - - - -  --  - - ;   // -Kp
      b = - - - - - - - - - - ;    // bias
```

Software Realization of the PI Controller




```
for (;;)   // continuous loop
      {    // loop time is the sampling interval τ
      e1 = e;
      e = adc ();
      mp = mp + a0*e + a1*e1;
      m = mp + b;
          // provision for saturation
      if (m < 0) m = 0;
      if (m > 100) m = 100;
      dac (m);
      }
}
```

Software Realization of the PI Controller




```
float adc (void)      // Analog-to-digital conversion
      {
      float  v;
      scanf (″%f″, &v);  // for (keyboard) simulation
      return  v;         // (to be replaced for actual
      }                  // realization)

void dac (float m)  // Digital-to-analog conversion
      {
      printf (″%f\n″, m);   // for (VDU) simulation
      }    // (to be replaced for actual realization)
```

Realization of Digital Controllers
(through discrete approximation of analog controllers)

Proportional-Derivative (PD) Controller

The analog controller output is

$$m = K_p \left[e + T_d \frac{de}{dt} \right] + b$$

where T_d is the derivative time.

Realization of Digital Controllers
(through discrete approximation of analog controllers)

Proportional-Derivative (PD) Controller

The analog controller output is

$$m = K_p\left[e + T_d\frac{de}{dt}\right] + b$$

where T_d is the derivative time.

Let the derivative of error 'e' at the nth instant be

$$D_n = \left.\frac{de}{dt}\right|_{t=n\tau}$$

Realization of Digital Controllers
(through discrete approximation of analog controllers)

Proportional-Derivative (PD) Controller

The analog controller output is

$$m = K_p \left[e + T_d \frac{de}{dt} \right] + b$$

where T_d is the derivative time.

Let the derivative of error 'e' at the nth instant be $D_n = \left. \frac{de}{dt} \right|_{t=n\tau}$

D_n may be approximated using the **backward difference algorithm** as

$$D_n \approx \frac{e_n - e_{n-1}}{\tau}$$

Realization of Digital Controllers
(through discrete approximation of analog controllers)

Proportional-Derivative (PD) Controller

The analog controller output is

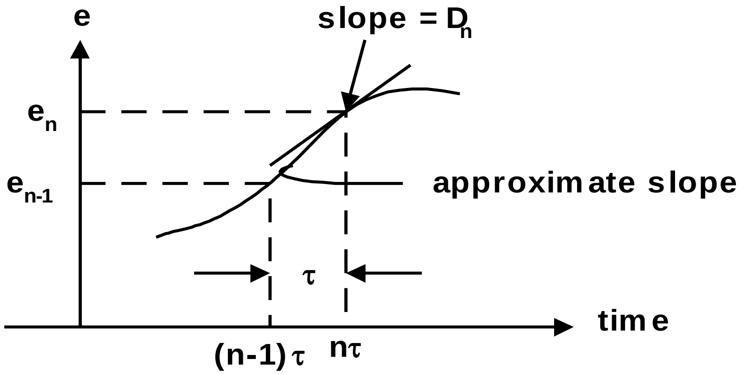
$$m = K_p\left[e + T_d\frac{de}{dt}\right] + b$$

where T_d is the derivative time.

Let the derivative of error 'e' at the nth instant be $D_n = \left.\frac{de}{dt}\right|_{t=n\tau}$

D_n may be approximated using the **backward difference algorithm** as

$$D_n \approx \frac{e_n - e_{n-1}}{\tau}$$



PD Controller

Thus the controller output at the nth instant is

$$m_n = K_p \left[e_n + T_d \left(\frac{e_n - e_{n-1}}{\tau} \right) \right] + b_n$$

PD Controller

Thus the controller output at the nth instant is

$$m_n = K_p \left[e_n + T_d \left(\frac{e_n - e_{n-1}}{\tau} \right) \right] + b_n$$

$$\text{or} \quad m_n = K_p \left[1 + \frac{T_d}{\tau} \right] e_n - \left[\frac{K_p T_d}{\tau} \right] e_{n-1} + b_n$$

PD Controller

Thus the controller output at the nth instant is

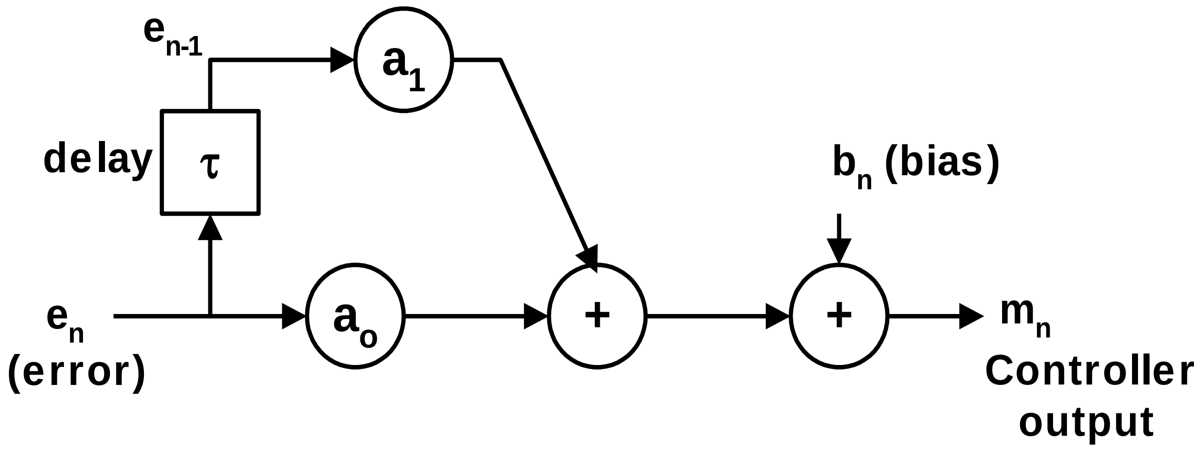
$$m_n = K_p \left[e_n + T_d \left(\frac{e_n - e_{n-1}}{\tau} \right) \right] + b_n$$

$$\text{or} \quad m_n = K_p \left[1 + \frac{T_d}{\tau} \right] e_n - \left[\frac{K_p T_d}{\tau} \right] e_{n-1} + b_n$$

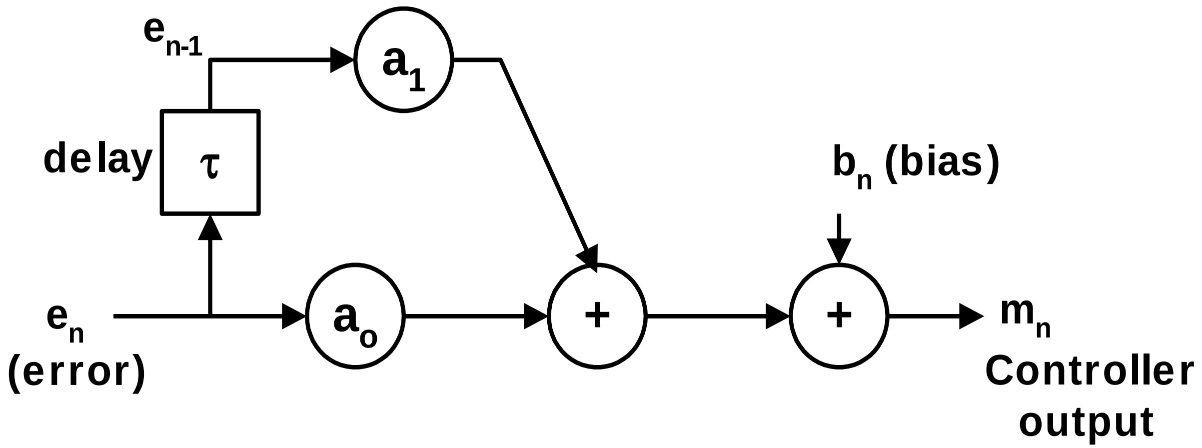
$$= a_o e_n + a_1 e_{n-1} + b_n$$

where $a_o = K_p \left(1 + \frac{T_d}{\tau} \right)$ and $a_1 = -\frac{K_p T_d}{\tau}$

Realization of the PD Controller



Realization of the PD Controller



Problem: Develop a 'c' program for software realization of the PD Controller.

PD Controller

Provision for anti-derivative kick

To avoid derivative action from a sudden change in set-point, the derivative action is generally derived from the measured output.

PD Controller

Provision for anti-derivative kick

To avoid derivative action from a sudden change in set-point, the derivative action is generally derived from the measured output.

Now, $e = r - c$

PD Controller

Provision for anti-derivative kick

To avoid derivative action from a sudden change in set-point, the derivative action is generally derived from the measured output.

Now, $e = r - c$

then, $$\frac{de}{dt} = \frac{dr}{dt} - \frac{dc}{dt},$$

PD Controller

Provision for anti-derivative kick

To avoid derivative action from a sudden change in set-point, the derivative action is generally derived from the measured output.

Now, $e = r - c$

then, $$\frac{de}{dt} = \frac{dr}{dt} - \frac{dc}{dt},$$

$$= -\frac{dc}{dt},$$ assuming set-point ‘r’ is constant.

PD Controller

Provision for anti-derivative kick

To avoid derivative action from a sudden change in set-point, the derivative action is generally derived from the measured output.

Now, $e = r - c$

then, $$\frac{de}{dt} = \frac{dr}{dt} - \frac{dc}{dt},$$

$$= -\frac{dc}{dt},$$ assuming set-point 'r' is constant.

Thus, the controller output may be expressed as,

$$m = K_p\left(e - T_d\frac{dc}{dt}\right) + b$$

PD Controller

Provision for anti-derivative kick

$$m = K_p\left(e - T_d\,\frac{dc}{dt}\right) + b$$

using **backward difference** algorithm, the controller output at the nth instant is

$$m_n = K_p\left[e_n - T_d\left(\frac{c_n - c_{n-1}}{\tau}\right)\right] + b_n$$

PD Controller

Provision for anti-derivative kick

$$m = K_p \left(e - T_d \frac{dc}{dt} \right) + b$$

using **backward difference** algorithm, the controller output at the nth instant is

$$m_n = K_p \left[e_n - T_d \left(\frac{c_n - c_{n-1}}{\tau} \right) \right] + b_n$$

$$\text{or} \quad m_n = K_p e_n - \frac{K_p T_d}{\tau} c_n + \frac{K_p T_d}{\tau} c_{n-1} + b_n$$

PD Controller

Provision for anti-derivative kick

$$m = K_p\left(e - T_d\frac{dc}{dt}\right) + b$$

using **backward difference** algorithm, the controller output at the nth instant is

$$m_n = K_p\left[e_n - T_d\left(\frac{c_n - c_{n-1}}{\tau}\right)\right] + b_n$$

$$\text{or} \quad m_n = K_p e_n - \frac{K_p T_d}{\tau}c_n + \frac{K_p T_d}{\tau}c_{n-1} + b_n$$

$$= a_o e_n + p_o c_n + p_1 c_{n-1} + b_n$$

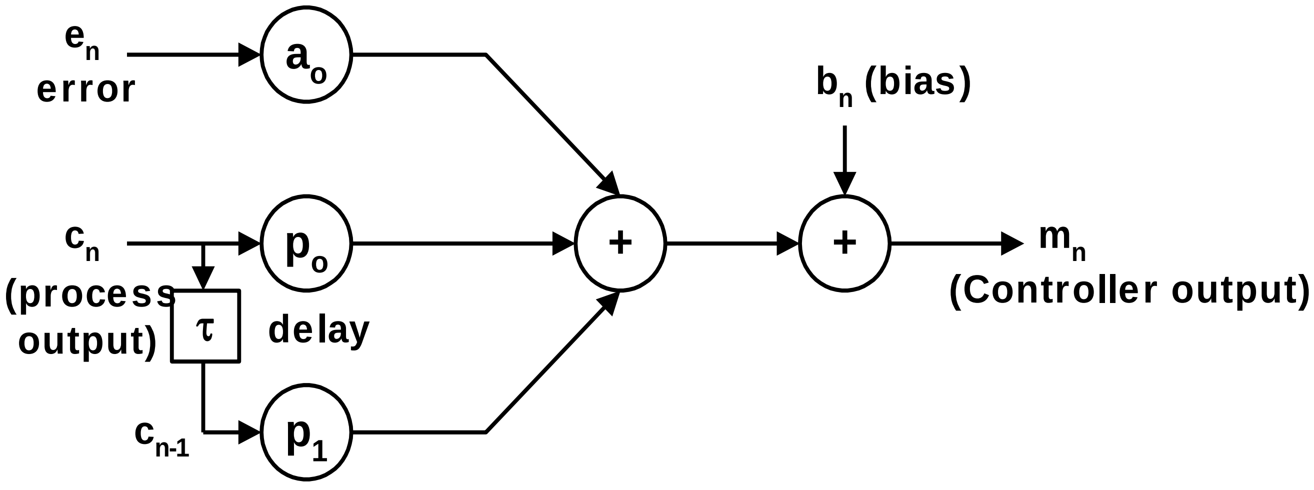
where

$$a_o = K_p$$

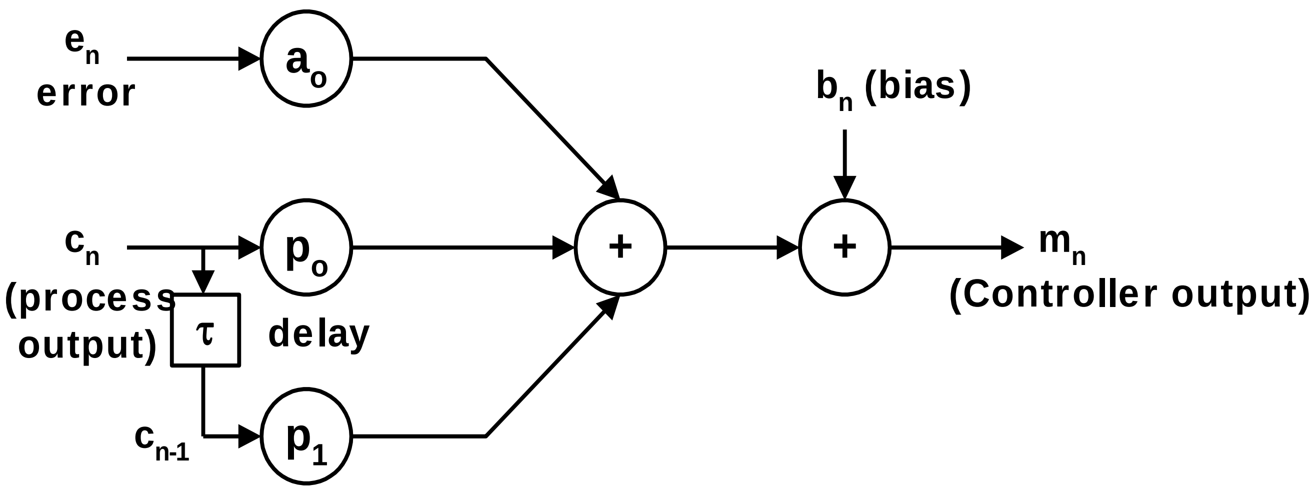
$$p_o = -\frac{K_p T_d}{\tau}$$

$$p_1 = \frac{K_p T_d}{\tau} = -p_o$$

Realization of PD Controller with anti-derivative kick



Realization of PD Controller with anti-derivative kick



Problem: Develop a ‘c’ program for software realization of the PD Controller with anti-derivative kick

Realization of Digital Controllers
(through discrete approximation of analog controllers)

Proportional-Integral-Derivative (PID) Controller

The analog controller output is

$$m = K_p \left[e + \frac{1}{T_i}\int_o^t e\,dt + T_d \frac{de}{dt} \right] + b$$

$$= m' + b \quad \text{(say)}$$

where $$m' = K_p \left[e + \frac{1}{T_i}\int_o^t e\,dt + T_d \frac{de}{dt} \right]$$

Realization of Digital Controllers
(through discrete approximation of analog controllers)

Proportional-Integral-Derivative (PID) Controller

The analog controller output is

$$m = K_p\left[e + \frac{1}{T_i}\int_o^t e\,dt + T_d\frac{de}{dt}\right] + b$$

$$= m' + b \quad \text{(say)}$$

where $$m' = K_p\left[e + \frac{1}{T_i}\int_o^t e\,dt + T_d\frac{de}{dt}\right]$$

The controller output (without bias) at the nth instant, using backward difference algorithm, is

$$m'_n = K_p\left[e_n + \frac{I_n}{T_i} + T_d\left(\frac{e_n - e_{n-1}}{\tau}\right)\right]$$

PID Controller

The controller output (without bias) at the nth instant is

$$m'_n = K_p\left[e_n + \frac{I_n}{T_i} + T_d\left(\frac{e_n - e_{n-1}}{\tau}\right)\right]$$

The controller output at the (n-1)th instant is

$$m'_{n-1} = K_p\left[e_{n-1} + \frac{I_{n-1}}{T_i} + T_d\left(\frac{e_{n-1} - e_{n-2}}{\tau}\right)\right]$$

PID Controller

The controller output (without bias) at the nth instant is

$$m'_n = K_p\left[e_n + \frac{I_n}{T_i} + T_d\left(\frac{e_n - e_{n-1}}{\tau}\right)\right]$$

The controller output at the (n-1)th instant is

$$m'_{n-1} = K_p\left[e_{n-1} + \frac{I_{n-1}}{T_i} + T_d\left(\frac{e_{n-1} - e_{n-2}}{\tau}\right)\right]$$

Subtracting,

$$m'_n - m'_{n-1} = K_p\left[e_n - e_{n-1} + \frac{I_n - I_{n-1}}{T_i} + \frac{T_d}{\tau}\left(e_n + e_{n-2} - 2e_{n-1}\right)\right]$$

PID Controller

$$m'_n - m'_{n-1} = K_p\left[e_n - e_{n-1} + \frac{I_n - I_{n-1}}{T_i} + \frac{T_d}{\tau}\left(e_n + e_{n-2} - 2e_{n-1}\right)\right]$$

Using rectangular integration algorithm, $I_n - I_{n-1} = \tau e_n$

PID Controller

$$m'_n - m'_{n-1} = K_p\left[e_n - e_{n-1} + \frac{I_n - I_{n-1}}{T_i} + \frac{T_d}{\tau}\left(e_n + e_{n-2} - 2e_{n-1}\right)\right]$$

Using rectangular integration algorithm,

$$I_n - I_{n-1} = \tau e_n$$

then,

$$m'_n - m'_{n-1} = K_p\left[e_n - e_{n-1} + \frac{\tau}{T_i}e_n + \frac{T_d}{\tau}\left(e_n - 2e_{n-1} + e_{n-2}\right)\right]$$

PID Controller

$$m'_n - m'_{n-1} = K_p\left[e_n - e_{n-1} + \frac{I_n - I_{n-1}}{T_i} + \frac{T_d}{\tau}\left(e_n + e_{n-2} - 2e_{n-1}\right)\right]$$

Using rectangular integration algorithm, $I_n - I_{n-1} = \tau e_n$

then, $m'_n - m'_{n-1} = K_p\left[e_n - e_{n-1} + \frac{\tau}{T_i}e_n + \frac{T_d}{\tau}\left(e_n - 2e_{n-1} + e_{n-2}\right)\right]$

or, $m'_n = e_n\left(1 + \frac{\tau}{T_i} + \frac{T_d}{\tau}\right)K_p - e_{n-1}\left(\frac{2T_d}{\tau} + 1\right)K_p + e_{n-2}\left(\frac{K_p T_d}{\tau}\right) + m'_{n-1}$

PID Controller

$$m'_n - m'_{n-1} = K_p\left[e_n - e_{n-1} + \frac{I_n - I_{n-1}}{T_i} + \frac{T_d}{\tau}\left(e_n + e_{n-2} - 2e_{n-1}\right)\right]$$

Using rectangular integration algorithm, $I_n - I_{n-1} = \tau e_n$

then, $$m'_n - m'_{n-1} = K_p\left[e_n - e_{n-1} + \frac{\tau}{T_i}e_n + \frac{T_d}{\tau}\left(e_n - 2e_{n-1} + e_{n-2}\right)\right]$$

or, $$m'_n = e_n\left(1 + \frac{\tau}{T_i} + \frac{T_d}{\tau}\right)K_p - e_{n-1}\left(\frac{2T_d}{\tau} + 1\right)K_p + e_{n-2}\left(\frac{K_pT_d}{\tau}\right) + m'_{n-1}$$

or, $$m'_n = a_o e_n + a_1 e_{n-1} + a_2 e_{n-2} + m'_{n-1}$$

PID Controller

$$m'_n = a_o e_n + a_1 e_{n-1} + a_2 e_{n-2} + m'_{n-1}$$

where, $$a_o = K_p \left(1 + \frac{\tau}{T_i} + \frac{T_d}{\tau} \right)$$

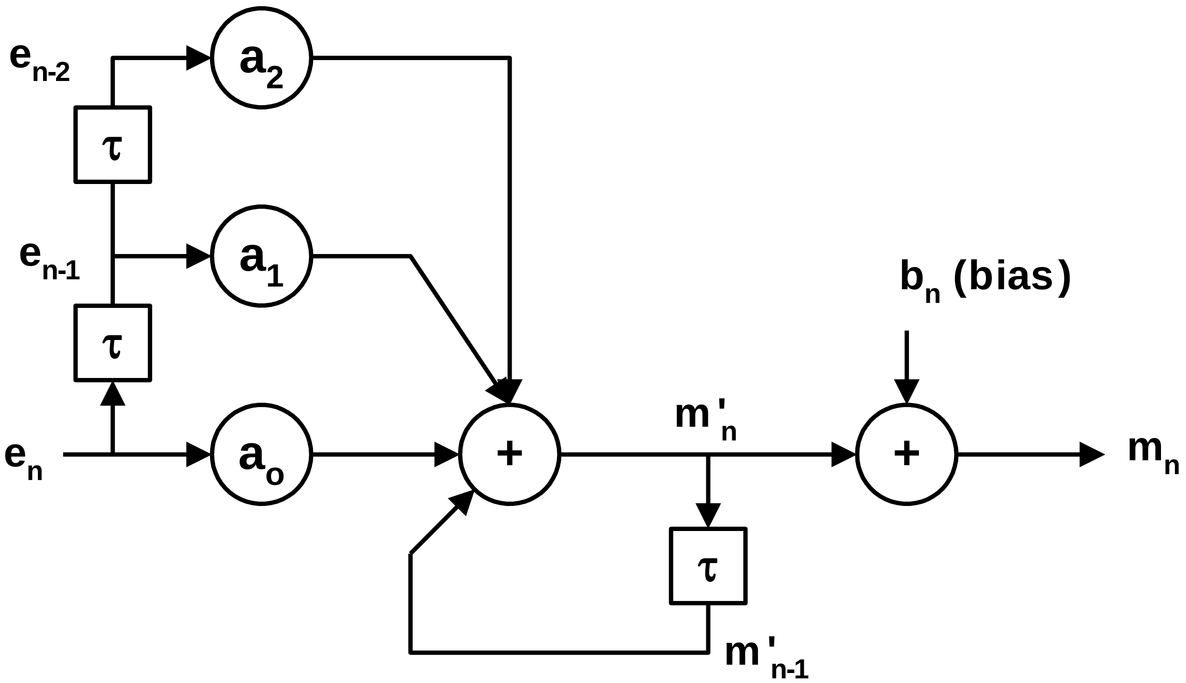
$$a_1 = -K_p \left(\frac{2T_d}{\tau} + 1 \right)$$

and $$a_2 = \frac{K_p T_d}{\tau}$$

Realization of the PID Controller

$$m'_n = a_o e_n + a_1 e_{n-1} + a_2 e_{n-2} + m'_{n-1}$$

and $m_n = m'_n + b_n$



PID Controller

Problems:

1. Develop a digital PID controller using trapezoidal rule for integration
2. Develop a program in ‘C’ for software realization of the PID controller
3. Modify the above controller to provide anti-derivative kick feature

Techniques used for anti-integral windup

Techniques used for anti-integral windup

- **By saturating or limiting the integral value**

Techniques used for anti-integral windup

- **By saturating or limiting the integral value**
- **By resetting the integral value to zero**

Techniques used for anti-integral windup

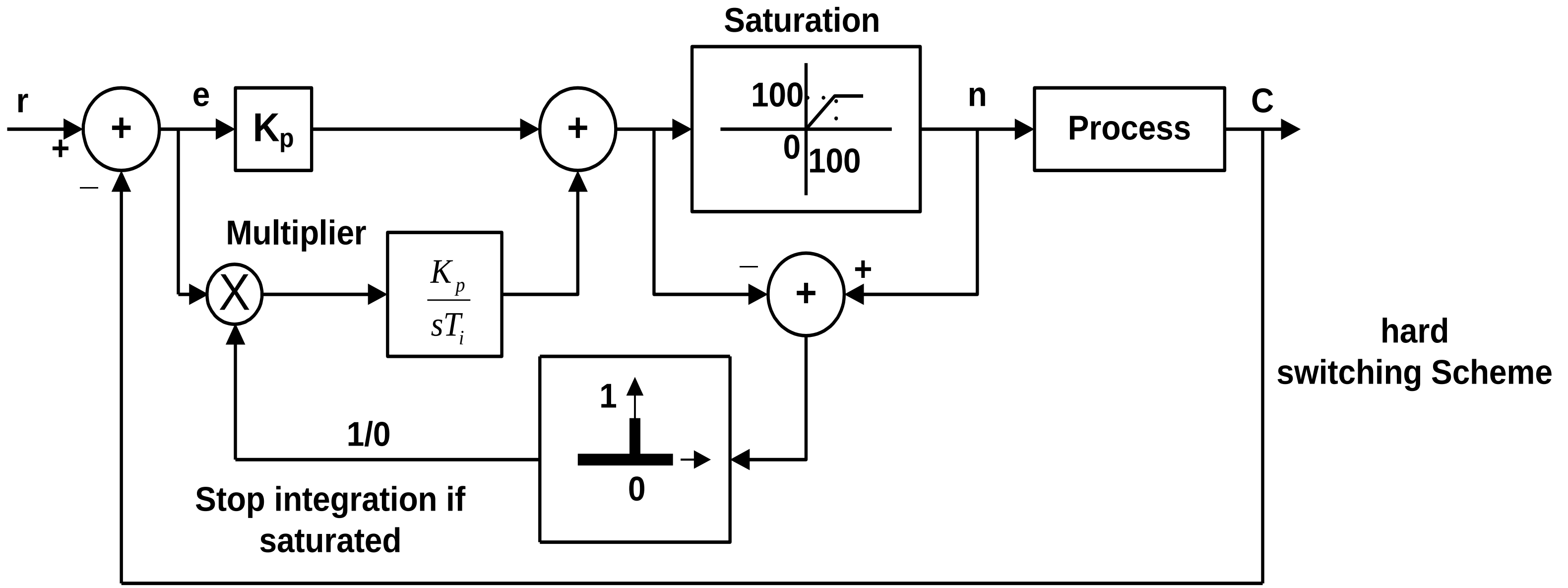
- **By saturating or limiting the integral value**
- **By resetting the integral value to zero**
- **By omitting the integral term**

Techniques used for anti-integral windup

- **By saturating or limiting the integral value**
- **By resetting the integral value to zero**
- **By omitting the integral term**
- **By adaptive adjustment of controller parameters**

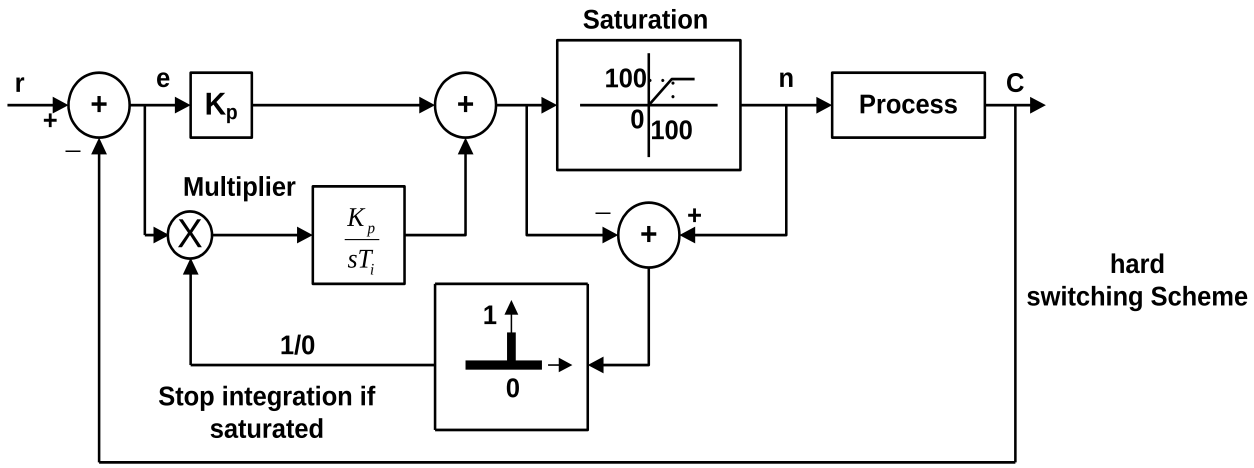
Some anti-integral windup schemes

Stop integration when PI/PID controller internal output (prior to the saturation block) exceeds the saturation limits

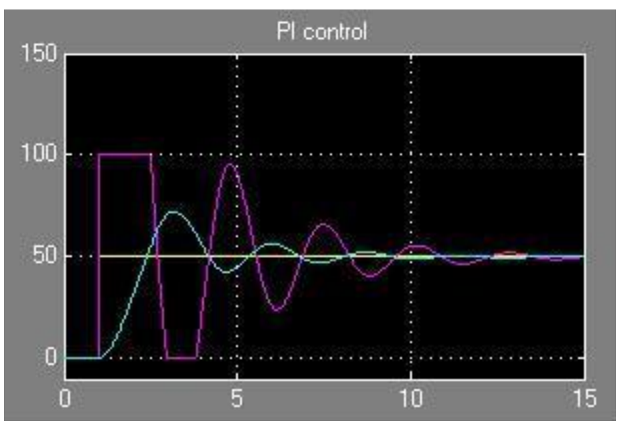


Scheme for PI Controller

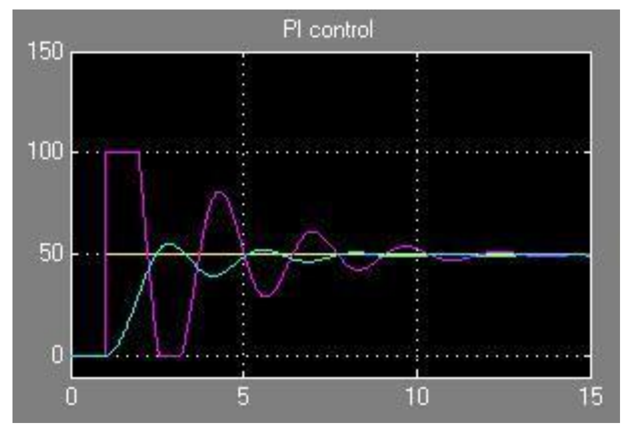
Performance of anti-integral windup scheme



Scheme for PI Controller



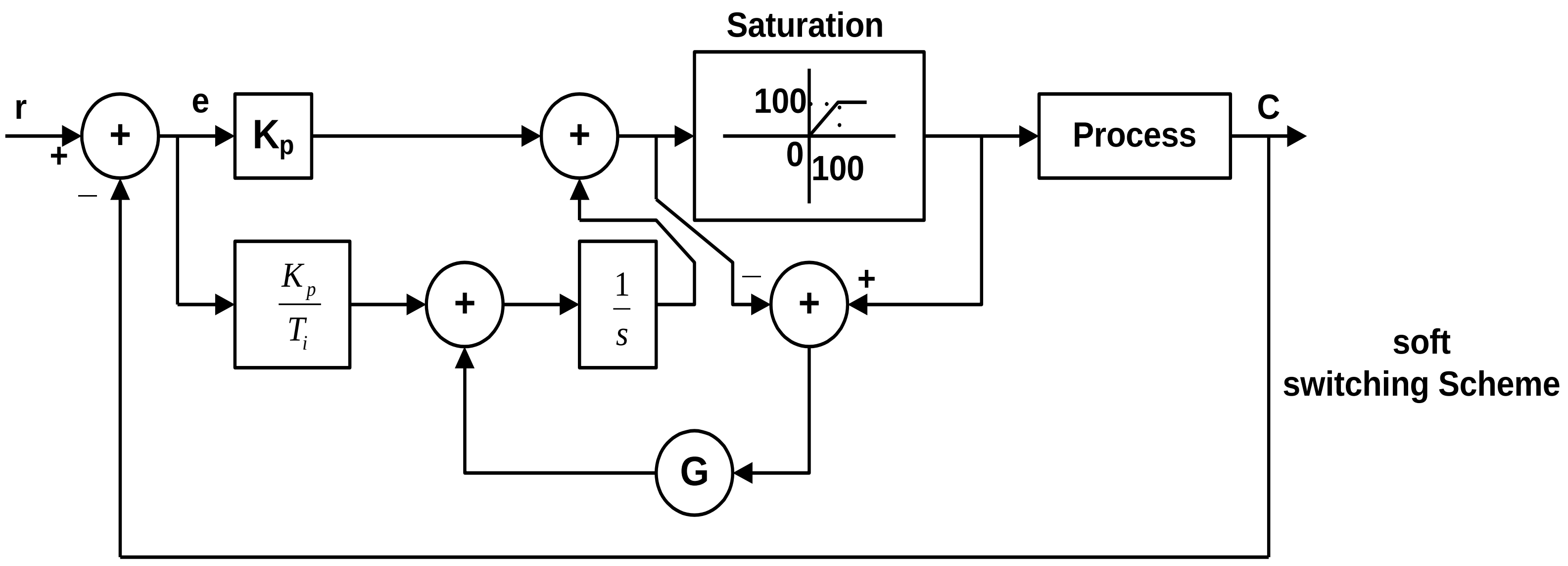
PI control without anti-integral windup



PI control with anti-integral windup

Some anti-integral windup schemes

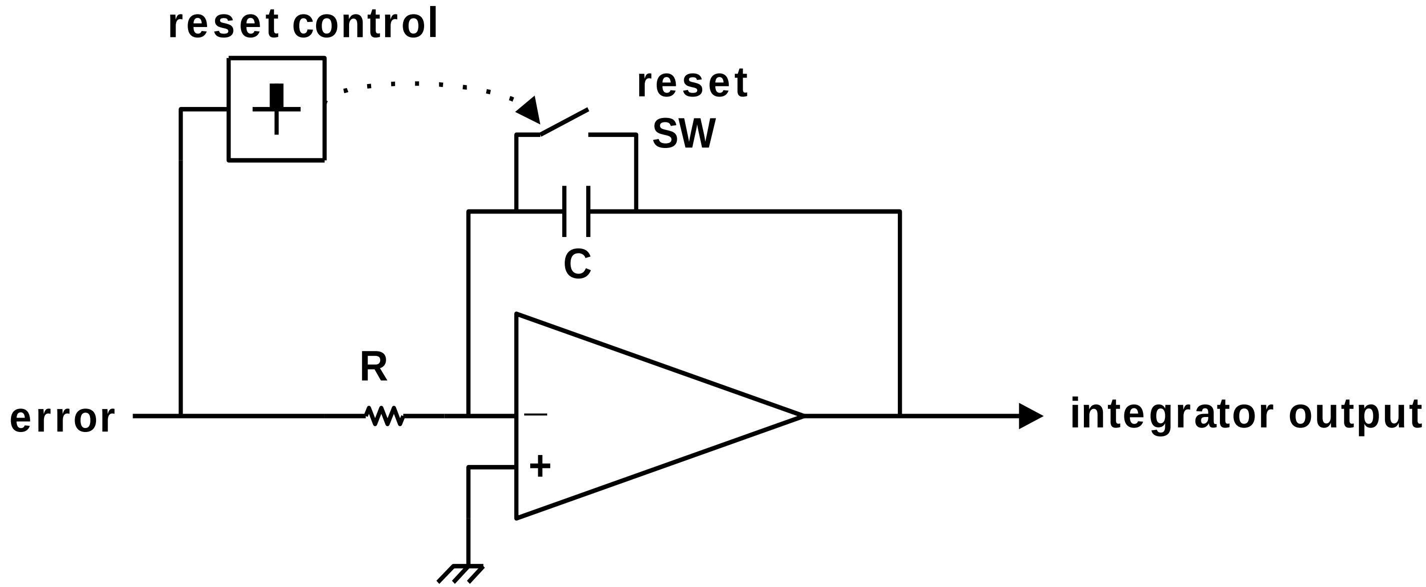
Reduce integration gradually as PI/PID controller internal output exceeds the saturation limits



G : a constant

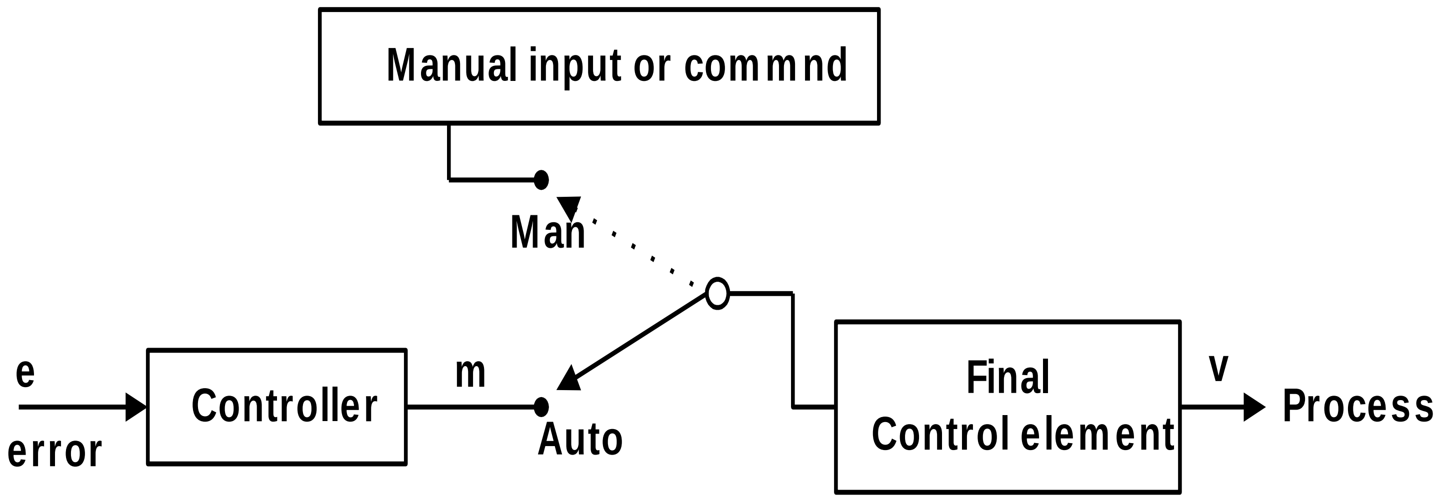
Some anti-integral windup schemes

Clegg integrator – the integrator is set to zero (reset) when the error crosses zero

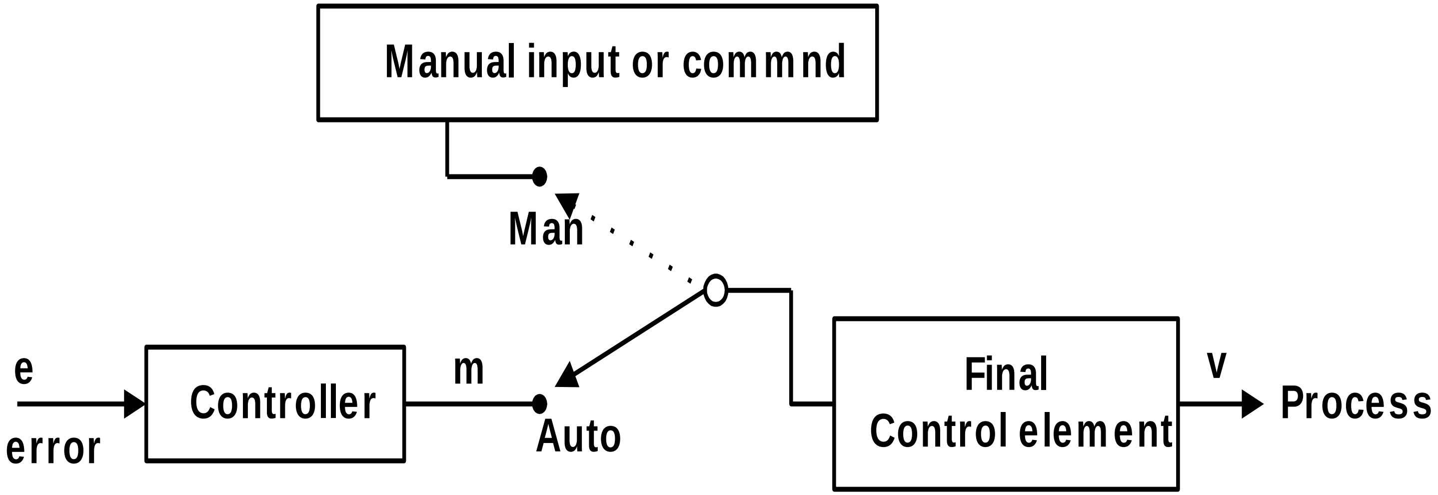


Automatic/Manual modes of Operations

- Automatic mode – means automatic closed loop operation
- Manual mode – means open loop manual control

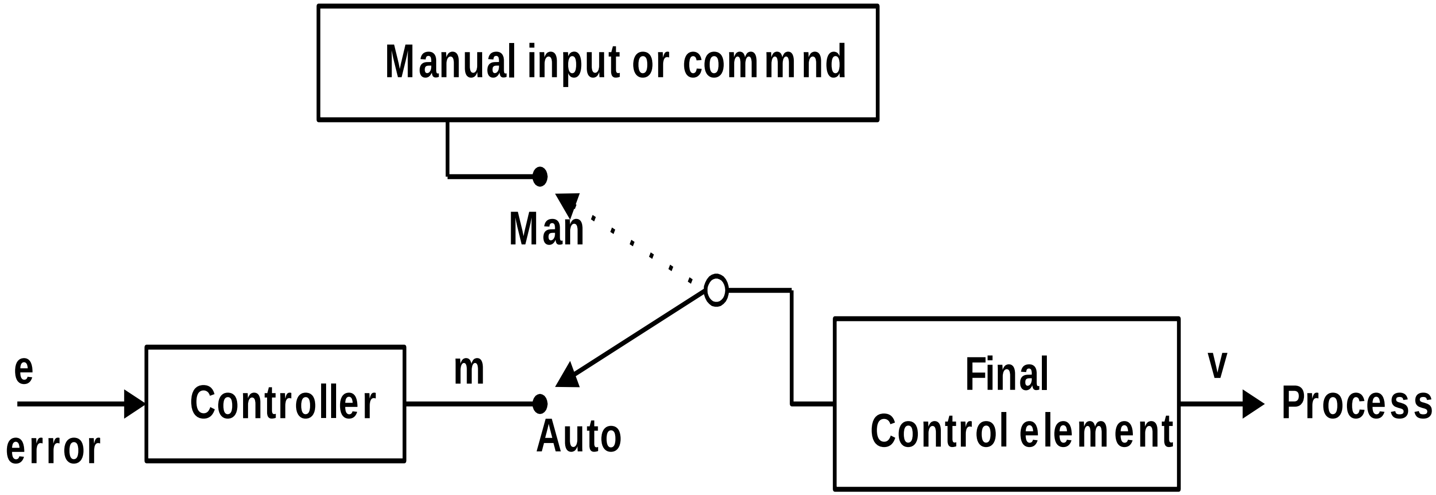


Automatic/Manual modes of Operations



If there is any difference between the controller output and the manual command, a *bump* occurs in the process output when the switch position is altered.

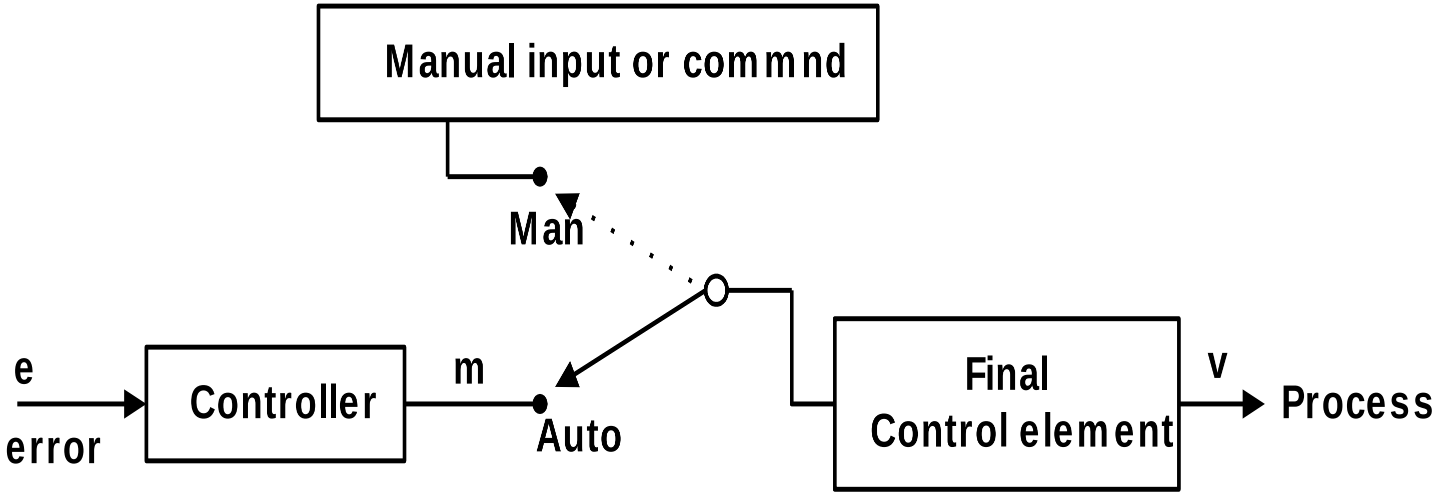
Automatic/Manual modes of Operations



If there is any difference between the controller output and the manual command, a *bump* occurs in the process output when the switch position is altered.

To provide *'bump-less transfer'* from auto-to-manual change over, special arrangements may be made for *'set-point initialization'*.

Automatic/Manual modes of Operations

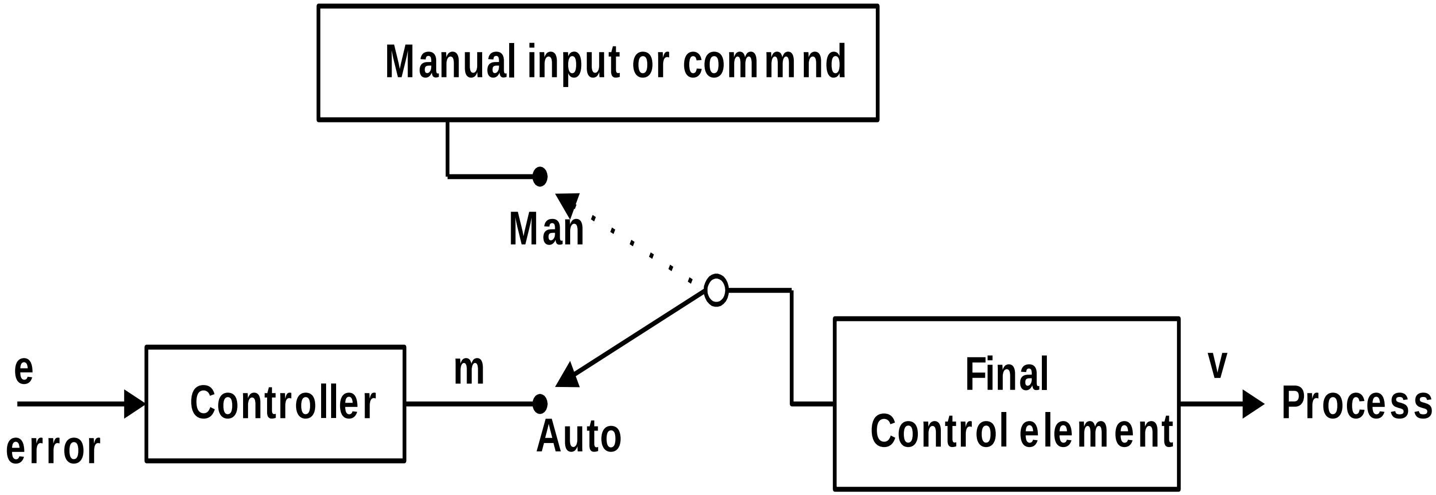


If there is any difference between the controller output and the manual command, a *bump* occurs in the process output when the switch position is altered.

To provide *'bump-less transfer'* from auto-to-manual change over, special arrangements may be made for *'set-point initialization'*.

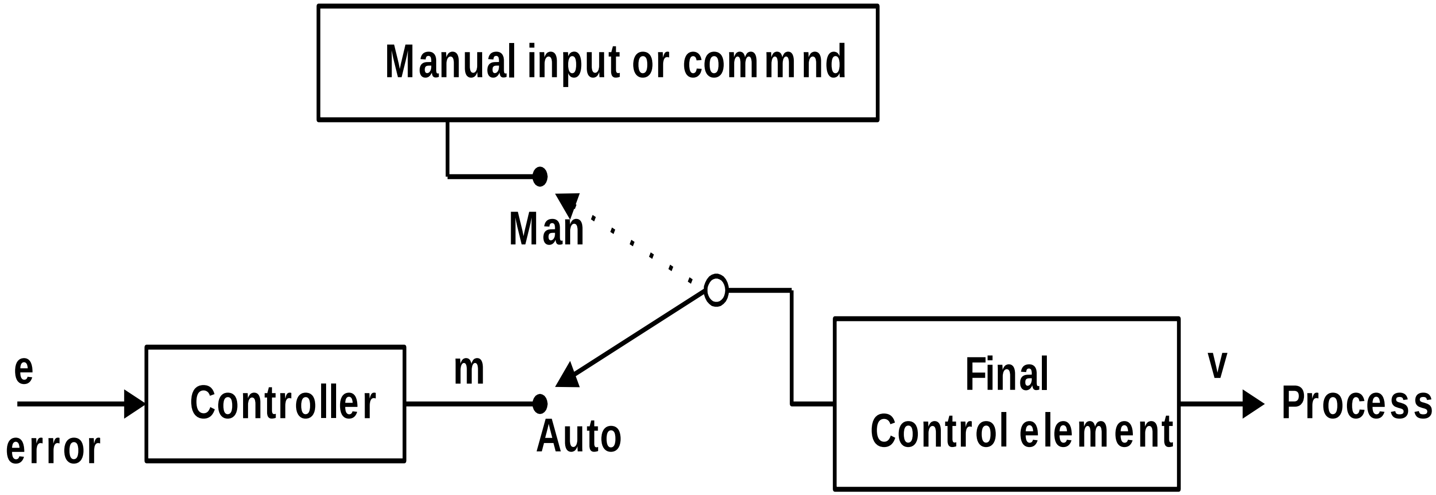
The manual command is driven to equal the controller output when the loop is in AUTO mode.

Automatic/Manual modes of Operations



When the loop is in MANual mode, if there is a steady error existing due to any difference between the set-point of the controller and the process output (under manual control), integral term, in case of PI and PID controllers, may wind-up to a large value, and consequently anti-integral wind-up is necessary for such situations.

Automatic/Manual modes of Operations

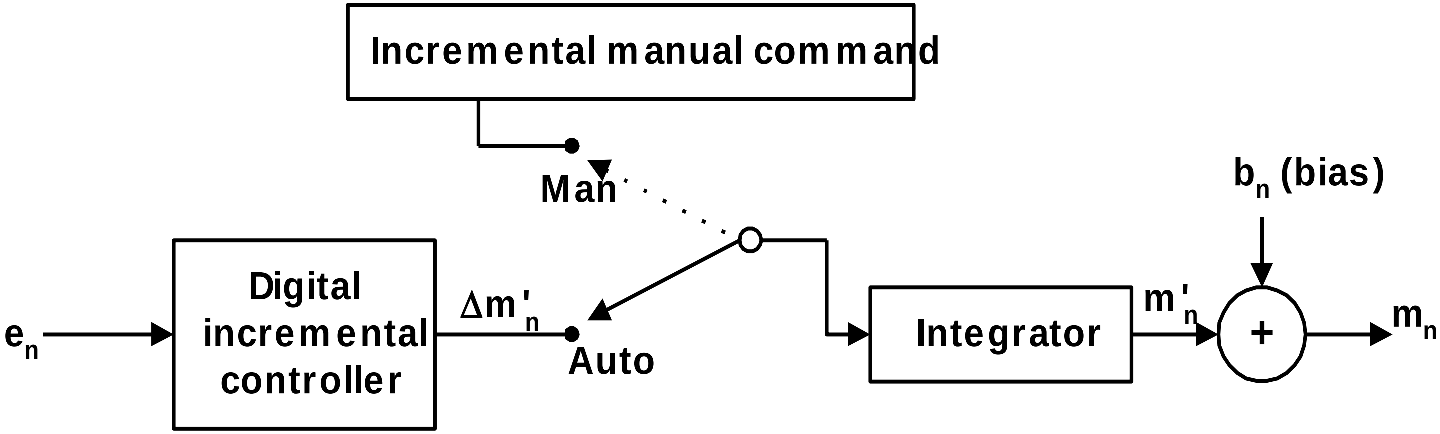


When the loop is in MANual mode, if there is a steady error existing due to any difference between the set-point of the controller and the process output (under manual control), integral term, in case of PI and PID controllers, may wind-up to a large value, and consequently anti-integral wind-up is necessary for such situations.

To provide *bump-less transfer* for all the operating modes, *incremental or velocity* from of controller is used with an additional integrator.

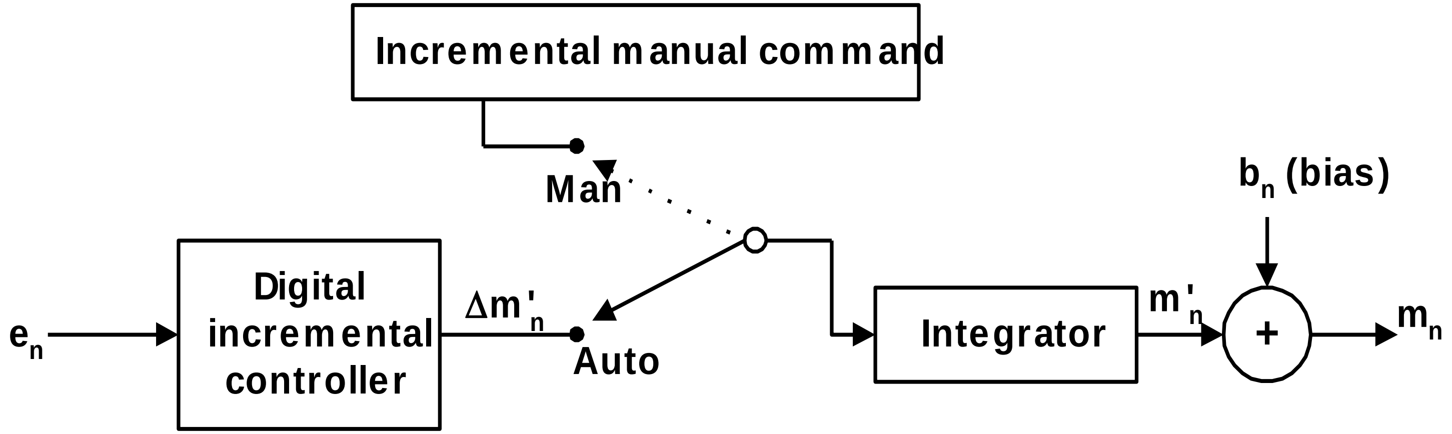
Automatic/Manual modes of Operations

Scheme for bump-less transfer



Automatic/Manual modes of Operations

Scheme for bump-less transfer



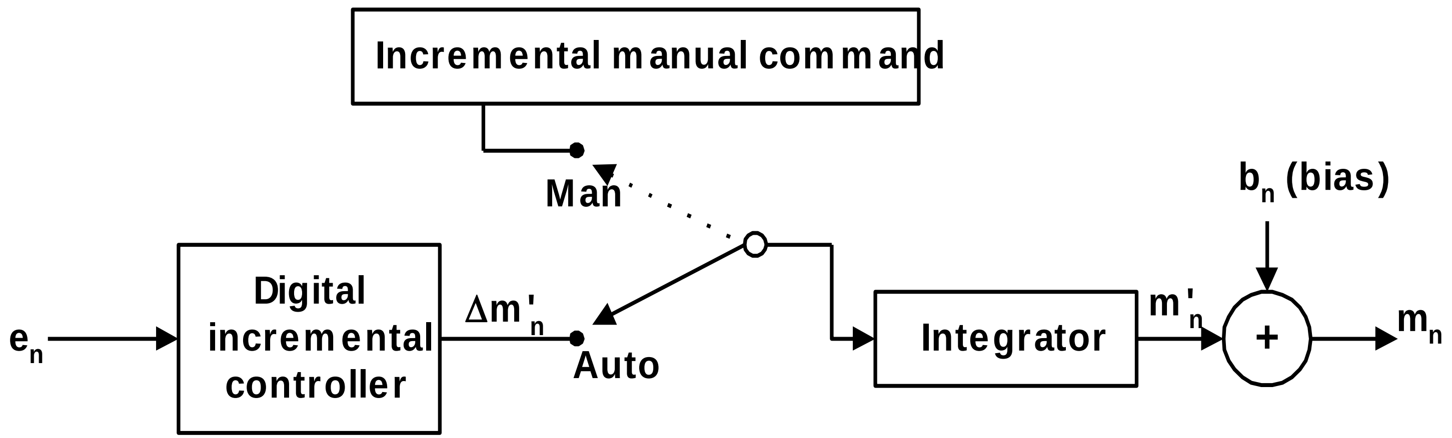
The incremental controller output (without bias) at the nth instant may be expressed as

$$\Delta m'_n = m'_n - m'_{n-1}$$

$$= e_n K_p \left(1 + \frac{\tau}{T_i} + \frac{T_d}{\tau}\right) - e_{n-1} K_p \left(\frac{2T_d}{\tau} + 1\right) + e_{n-2} \frac{K_p T_d}{\tau}$$ **for a PID controller**

Automatic/Manual modes of Operations

Scheme for bump-less transfer



The incremental controller output (without bias) at the nth instant may be expressed as

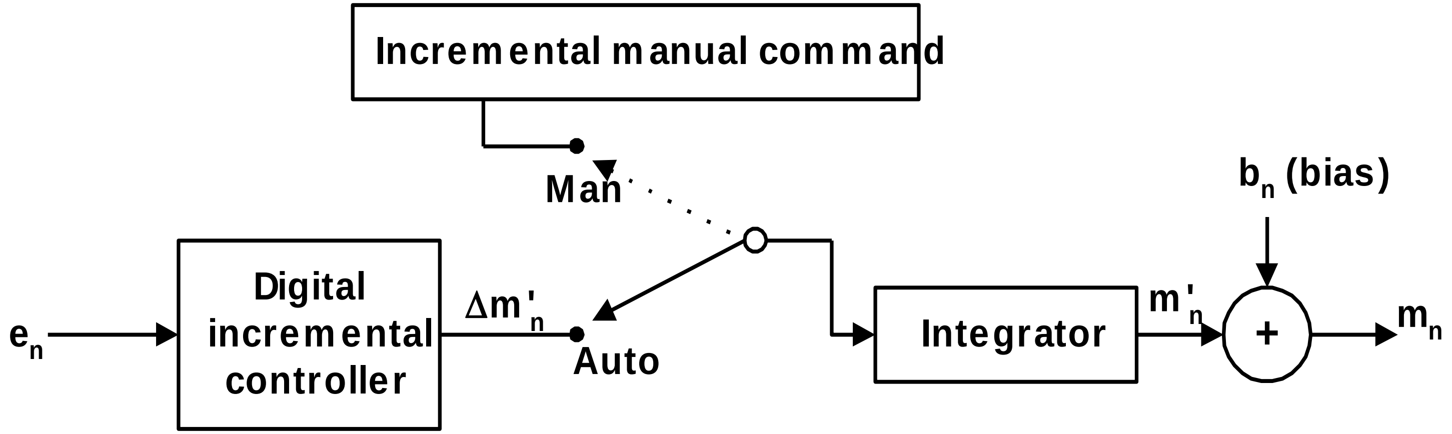
$$\Delta m_n' = m_n' - m_{n-1}'$$

$$= e_n K_p \left(1 + \frac{\tau}{T_i} + \frac{T_d}{\tau}\right) - e_{n-1} K_p \left(\frac{2T_d}{\tau} + 1\right) + e_{n-2} \frac{K_p T_d}{\tau}$$ **for a PID controller**

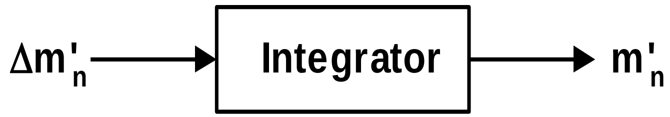
$$= a_o e_n + a_1 e_{n-1} + a_2 e_{n-2}$$

Automatic/Manual modes of Operations

Scheme for bump-less transfer

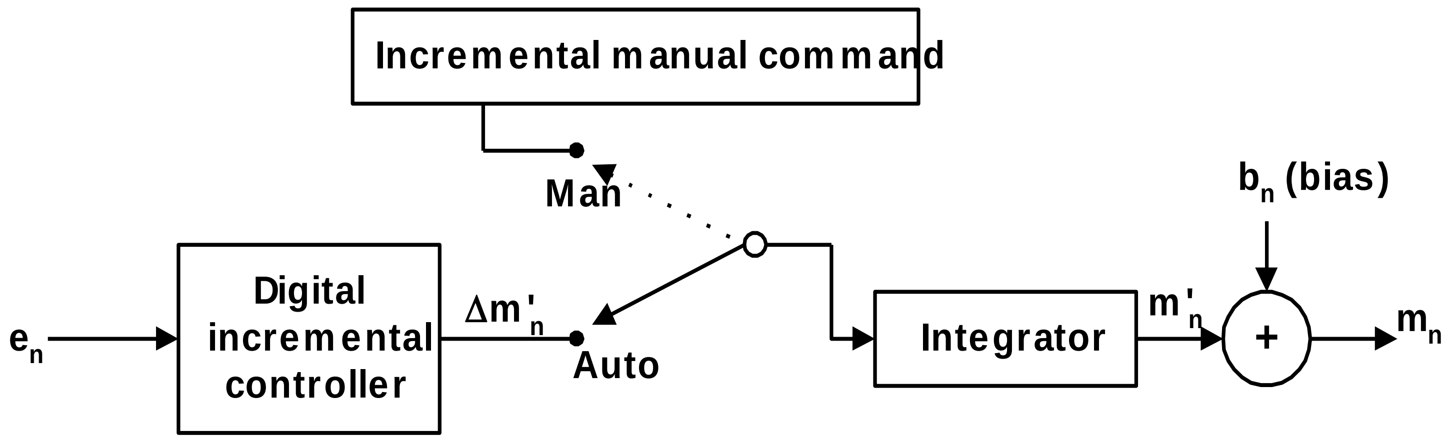


The integrator may be represented as

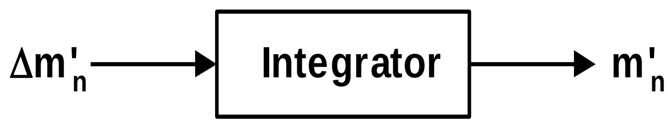


Automatic/Manual modes of Operations

Scheme for bump-less transfer



The integrator may be represented as

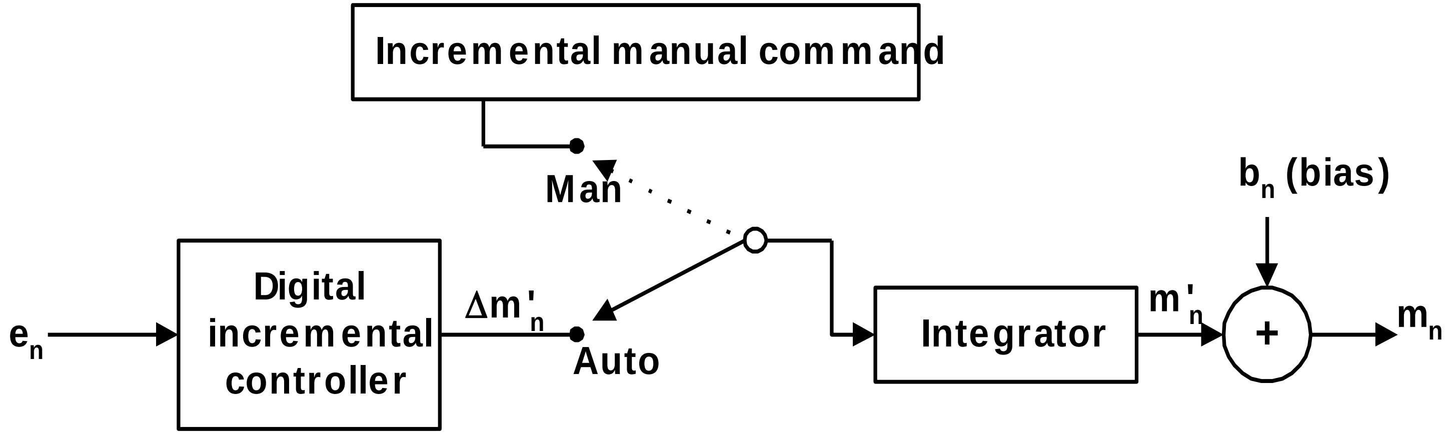


The integrator output may be represented as

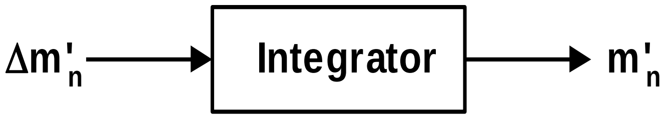
$$
\begin{aligned}
m'_n &= m'_n - m'_{n-1} + m'_{n-1} \\
&= \Delta m'_n + m'_{n-1}
\end{aligned}
$$

Automatic/Manual modes of Operations

Scheme for bump-less transfer

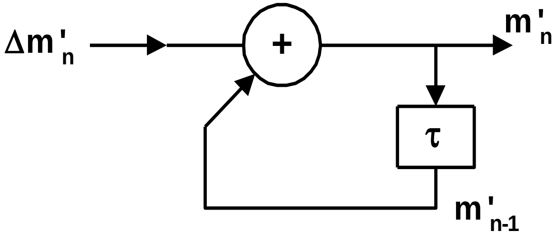


The integrator may be represented as



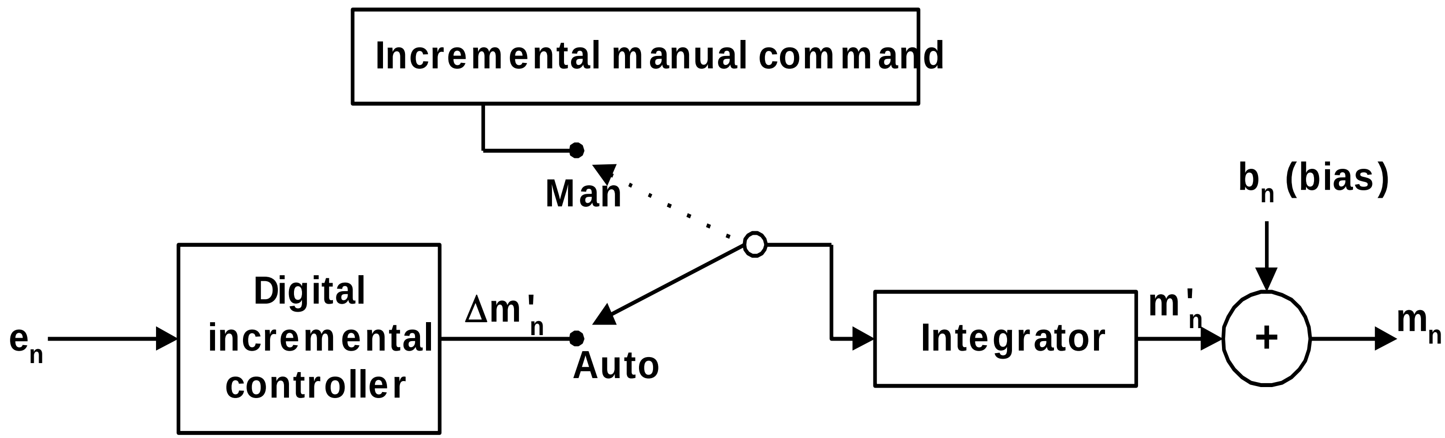
The integrator output may be represented as

$$m'_n = m'_n - m'_{n-1} + m'_{n-1}$$
$$= \Delta m'_n + m'_{n-1}$$



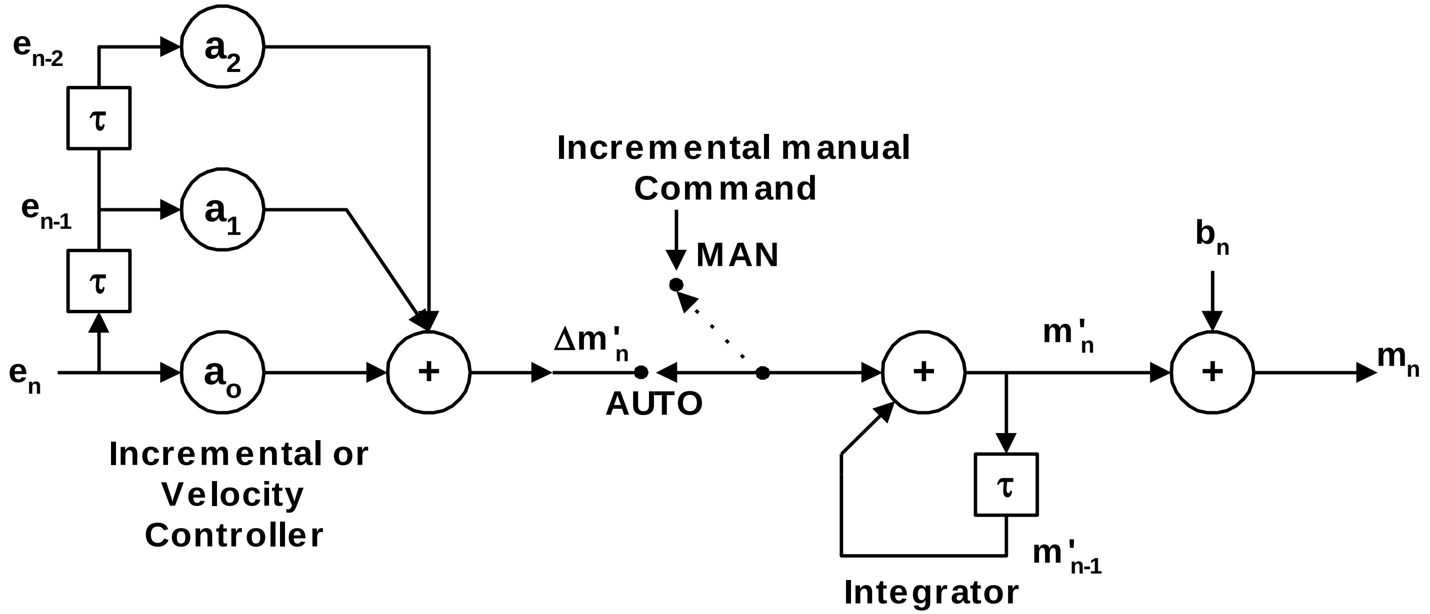
Automatic/Manual modes of Operations

Scheme for bump-less transfer



The presence of integrator at the output ensures a smooth output variation even when the actual manual command is different from the actual controller output under closed-loop control.

Realization of the incremental type PID Controller



Automatic tuning of PID Controllers – the Relay autotuner

Automatic tuning of PID Controllers – the Relay autotuner

Suitable for processes with non-zero dead-time.

Automatic tuning of PID Controllers – the Relay autotuner

Suitable for processes with non-zero dead-time.

This is based on a special technique for determining the critical gain K_c and critical time period T_c of the process loop.

Automatic tuning of PID Controllers – the Relay autotuner

Suitable for processes with non-zero dead-time.

This is based on a special technique for determining the critical gain K_c and critical time period T_c of the process loop.

K_c is the gain margin of the process loop.

Automatic tuning of PID Controllers – the Relay autotuner

Suitable for processes with non-zero dead-time.

This is based on a special technique for determining the critical gain K_c and critical time period T_c of the process loop.

K_c is the gain margin of the process loop.

Controller parameters K_p, T_i and T_d are calculated according to *Ziegler – Nichols (Z-N) rule* for a stable time response.

Automatic tuning of PID Controllers – the Relay autotuner

Suitable for processes with non-zero dead-time.

This is based on a special technique for determining the critical gain K_c and critical time period T_c of the process loop.

K_c is the gain margin of the process loop.

Controller parameters K_p, T_i and T_d are calculated according to *Ziegler – Nichols (Z-N) rule* for a stable time response.

Z-N settings

Controller	K_p	T_i	T_d
P – Controller	$0.5\ K_c$		
PI – Controller	$0.45\ K_c$	$T_c/1.2$	
PID – Controller	$0.6\ K_c$	$T_c/2$	$T_c/8$

Automatic tuning of PID Controllers – the Relay autotuner

Suitable for processes with non-zero dead-time.

This is based on a special technique for determining the critical gain K_c and critical time period T_c of the process loop.

K_c is the gain margin of the process loop.

Controller parameters K_p, T_i and T_d are calculated according to *Ziegler – Nichols (Z-N) rule* for a stable time response.

Z-N settings

Controller	K_p	T_i	T_d
P – Controller	$0.5\ K_c$		
PI – Controller	$0.45\ K_c$	$T_c/1.2$	
PID – Controller	$0.6\ K_c$	$T_c/2$	$T_c/8$

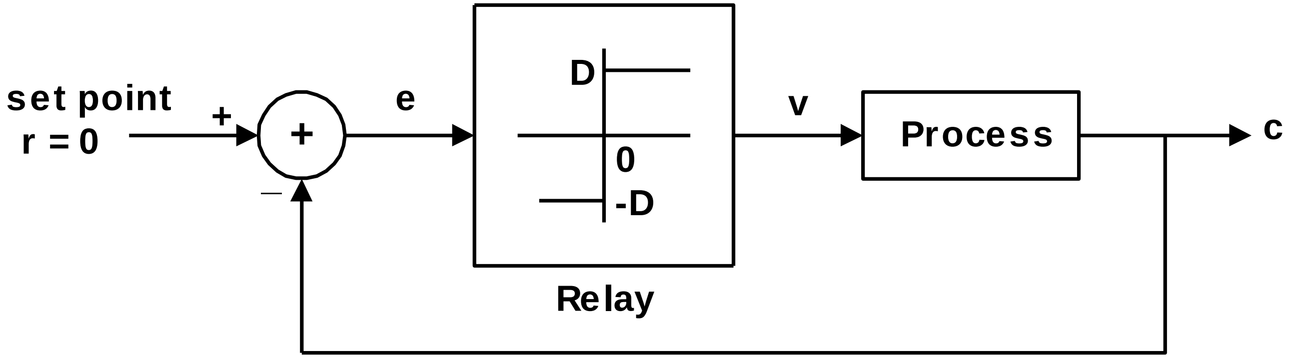
⇐ **gain margin: 2** (P – Controller row)

Automatic tuning of PID Controllers – the Relay autotuner

The critical gain K_c and critical time period T_c are determined from an experiment with relay (switching element) feedback.

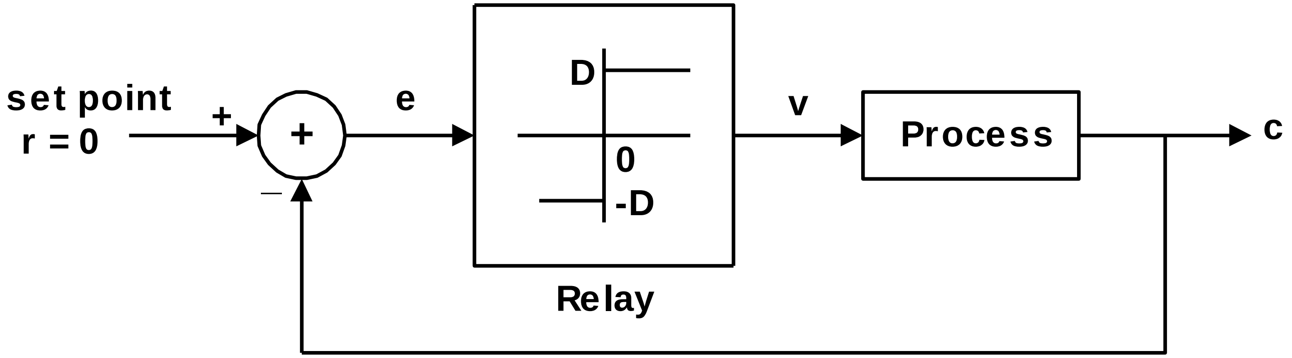
Automatic tuning of PID Controllers – the Relay autotuner

The critical gain K_c and critical time period T_c are determined from an experiment with relay (switching element) feedback.



Automatic tuning of PID Controllers – the Relay autotuner

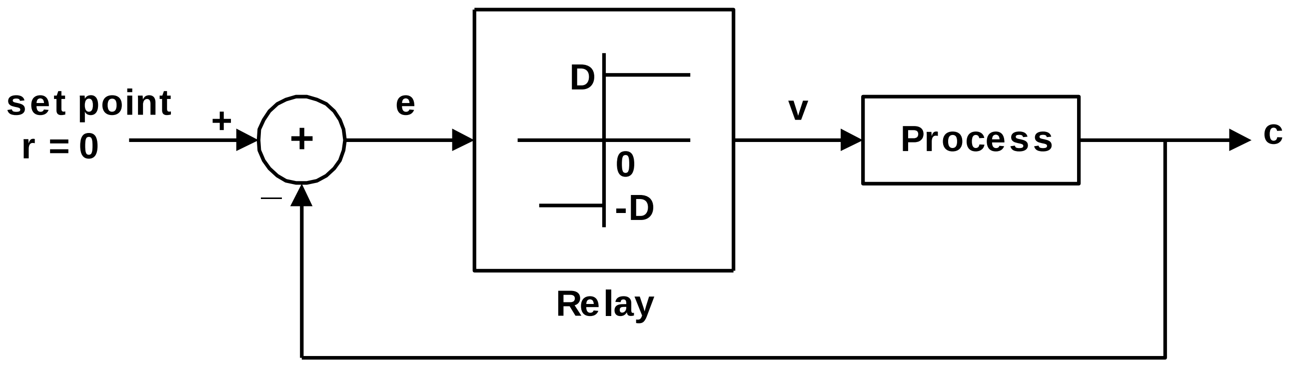
The critical gain K_c and critical time period T_c are determined from an experiment with relay (switching element) feedback.



✓**The relay control provides ON / OFF control of the process.**

Automatic tuning of PID Controllers – the Relay autotuner

The critical gain K_c and critical time period T_c are determined from an experiment with relay (switching element) feedback.

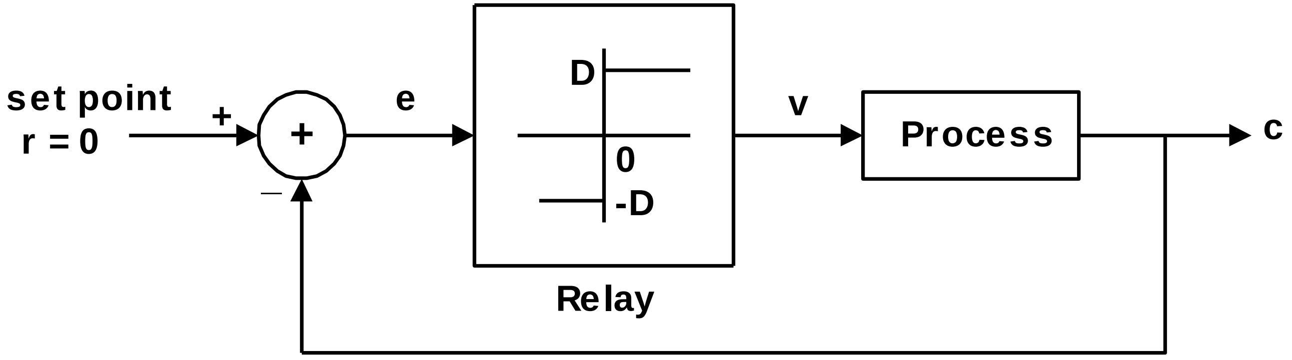


✓**The relay control provides ON / OFF control of the process.**

✓**The input 'r' is set to zero.**

Automatic tuning of PID Controllers – the Relay autotuner

The critical gain K_c and critical time period T_c are determined from an experiment with relay (switching element) feedback.



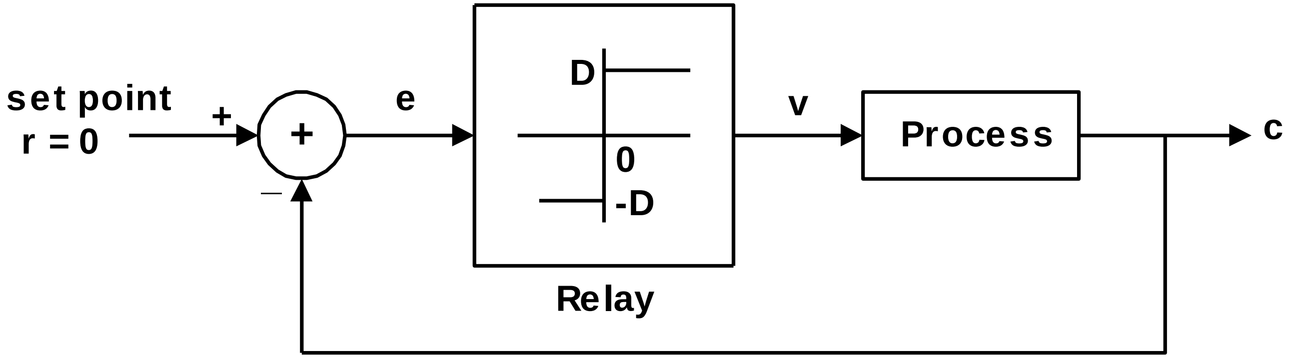
✓The relay control provides ON / OFF control of the process.

✓The input 'r' is set to zero.

✓The output 'c' oscillates around a mean value of zero (limit-cycle oscillations).

Automatic tuning of PID Controllers – the Relay autotuner

The critical gain K_c and critical time period T_c are determined from an experiment with relay (switching element) feedback.



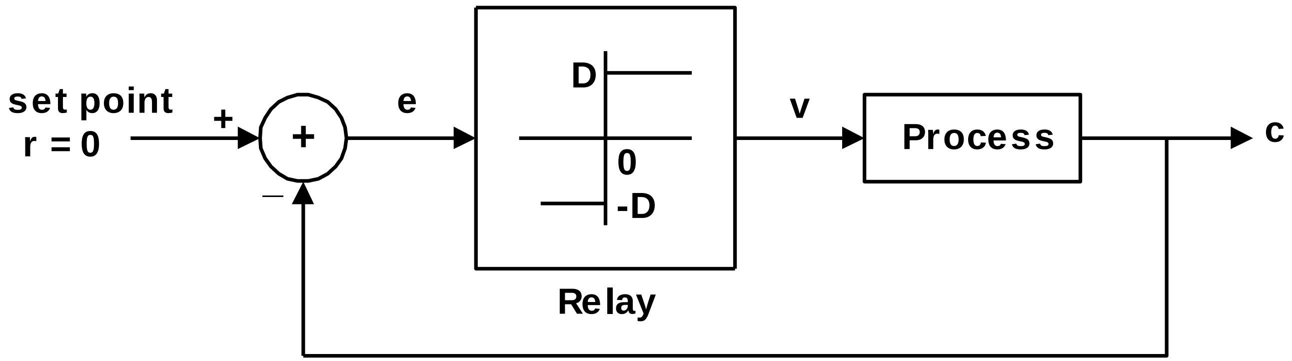
✓The relay control provides ON / OFF control of the process.

✓The input 'r' is set to zero.

✓The output 'c' oscillates around a mean value of zero (limit-cycle oscillations).

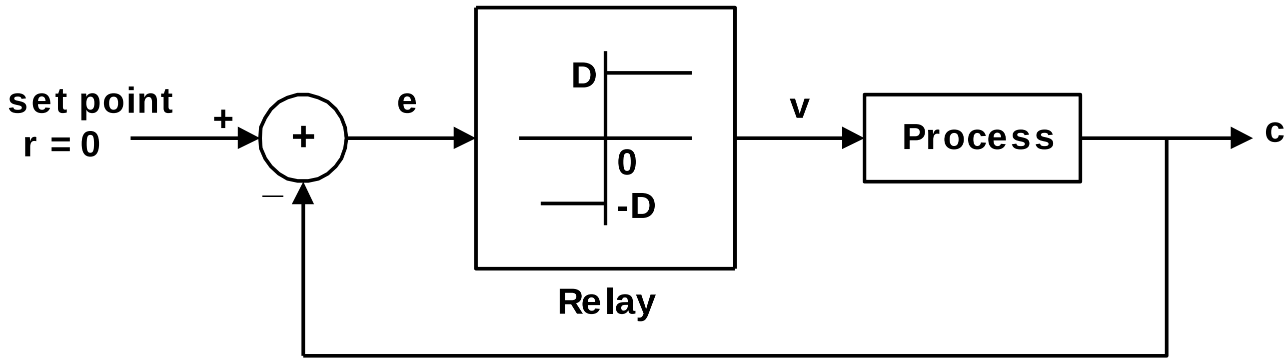
✓The process is driven by a square wave of amplitude 'D'.

Automatic tuning of PID Controllers – the Relay autotuner



Assuming the process to be a low-pass system, the process output 'c' contains mainly the fundamental component.

Automatic tuning of PID Controllers – the Relay autotuner

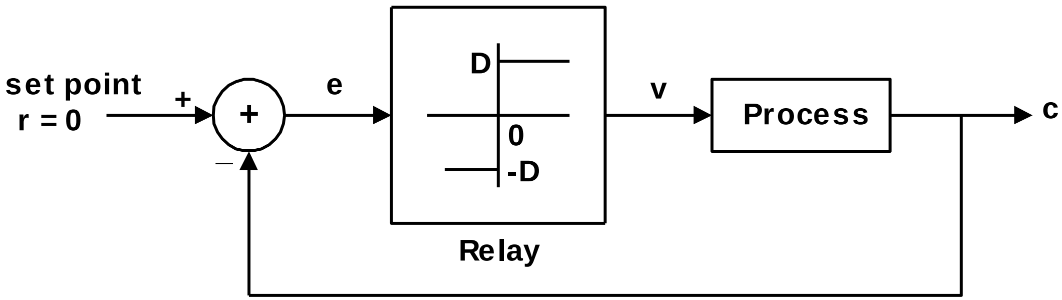


Assuming the process to be a low-pass system, the process output ‘c’ contains mainly the fundamental component.

Thus the error signal ‘e’ becomes sinusoidal,

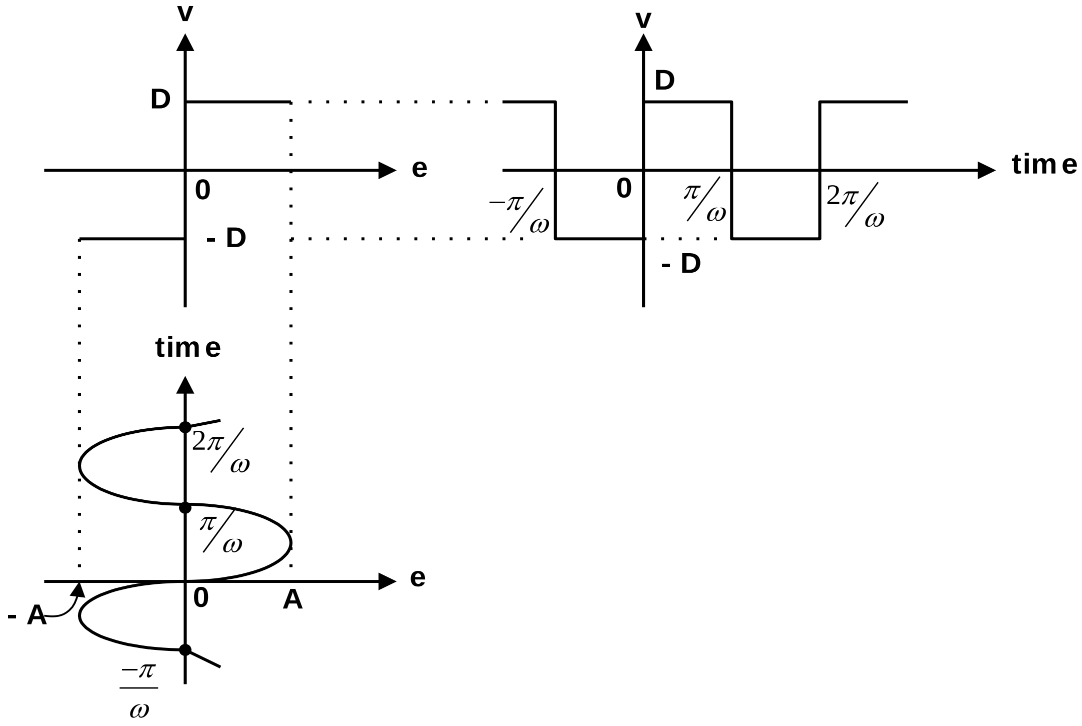
$$e = A \sin \omega t$$

Automatic tuning of PID Controllers – the Relay autotuner

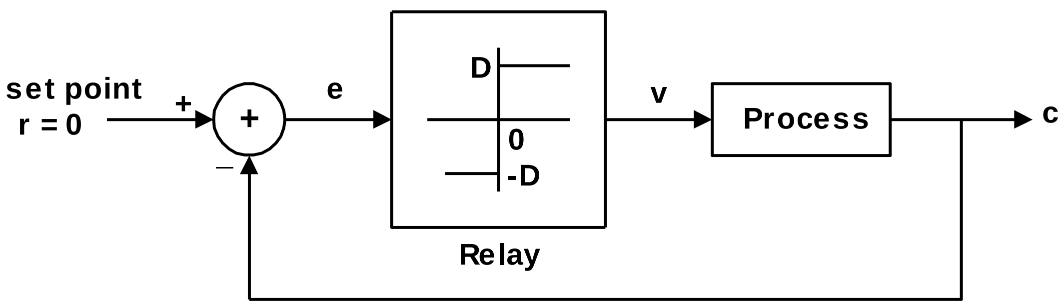


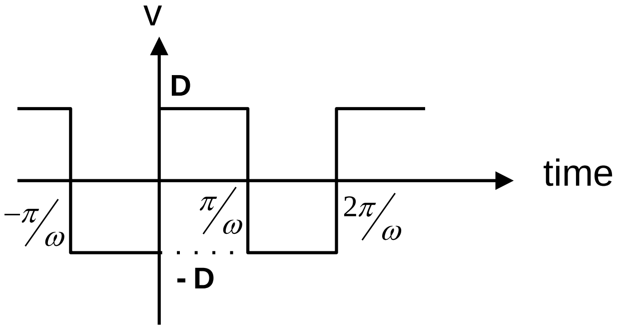
$e = A \sin \omega t$

The relay output 'v' may be found out as follows:

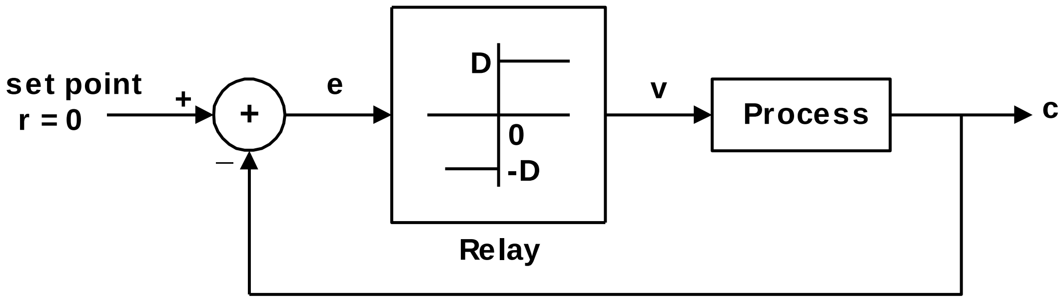


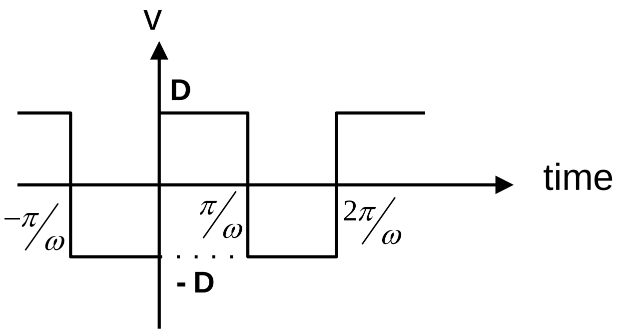
Automatic tuning of PID Controllers - the Relay autotuner





Automatic tuning of PID Controllers – the Relay autotuner

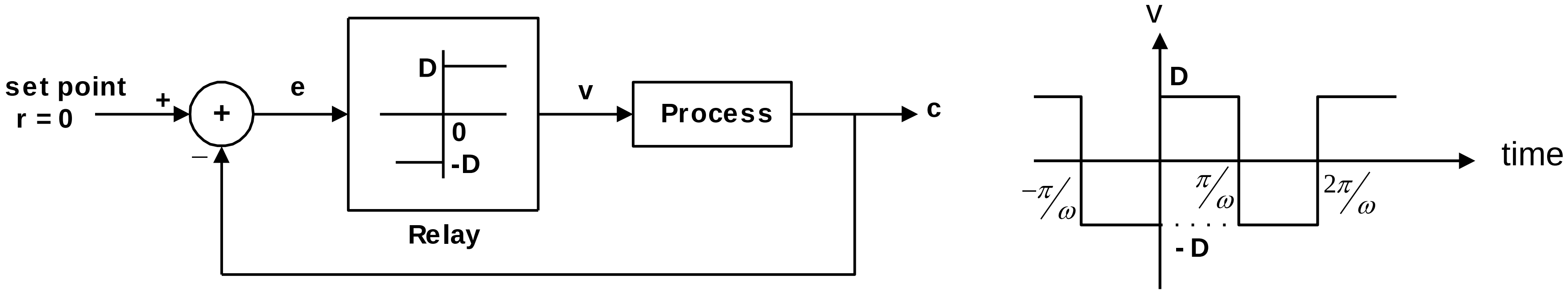




The Fourier series of the relay output (v) may be expressed as:

$$v = \frac{4D}{\pi}\left(\sin\omega t + \frac{1}{3}\sin 3\omega t + \frac{1}{5}\sin 5\omega t + ...\right)$$

Automatic tuning of PID Controllers – the Relay autotuner

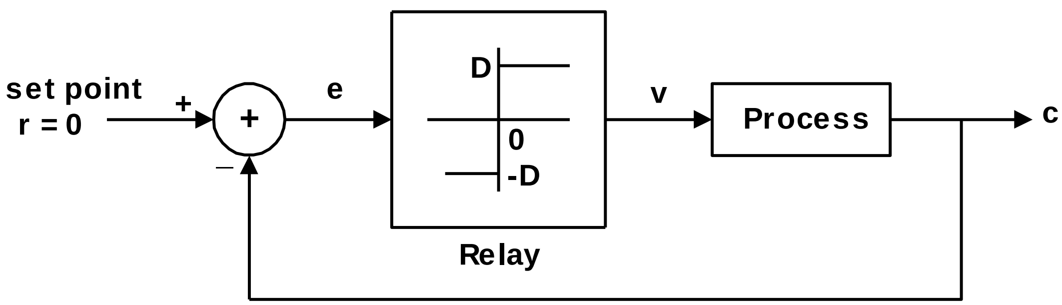


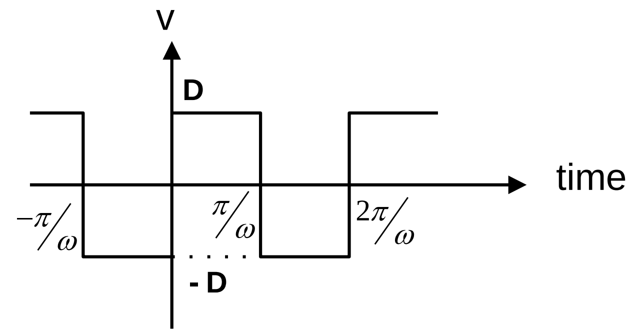
The Fourier series of the relay output (v) may be expressed as:

$$v = \frac{4D}{\pi}\left(\sin\omega t + \frac{1}{3}\sin 3\omega t + \frac{1}{5}\sin 5\omega t + ...\right)$$

The process practically attenuates all higher harmonics other than the fundamental.

Automatic tuning of PID Controllers – the Relay autotuner





The Fourier series of the relay output (v) may be expressed as:

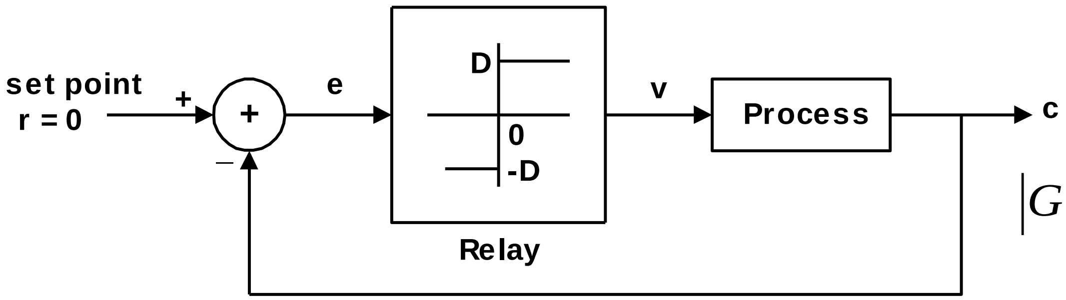
$$v = \frac{4D}{\pi}\left(\sin\omega t + \frac{1}{3}\sin 3\omega t + \frac{1}{5}\sin 5\omega t + ...\right)$$

The process practically attenuates all higher harmonics other than the fundamental.

Then the process gain at frequency 'ω' becomes

$$\left|G(\omega)\right| = \frac{output\ \ amplitude}{input\ \ amplitude} = \frac{A}{\frac{4D}{\pi}} = \frac{\pi A}{4D}$$

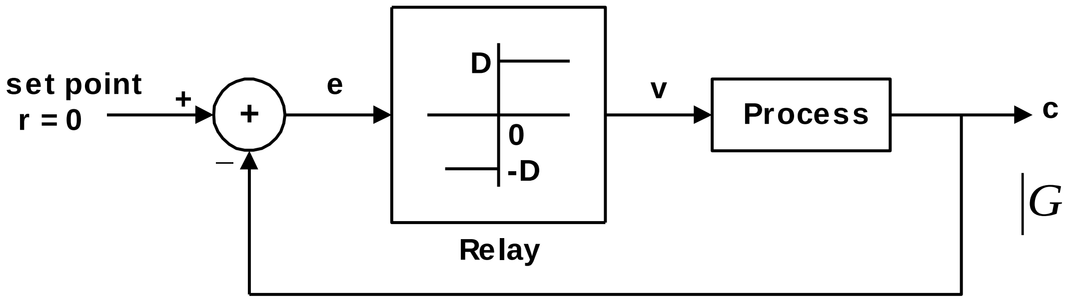
Automatic tuning of PID Controllers – the Relay autotuner



Process gain at frequency 'ω':

$$|G(\omega)| = \frac{output\ \ amplitude}{input\ \ amplitude} = \frac{A}{\frac{4D}{\pi}} = \frac{\pi A}{4D}$$

Automatic tuning of PID Controllers – the Relay autotuner

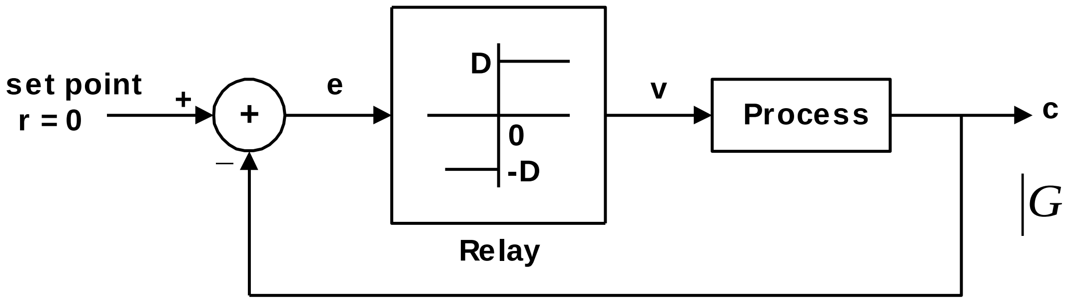


Process gain at frequency 'ω':

$$|G(\omega)| = \frac{output \quad amplitude}{input \quad amplitude} = \frac{A}{\dfrac{4D}{\pi}} = \frac{\pi A}{4D}$$

Now, to maintain steady oscillations at $\omega = \omega_c$, the loop gain is $1 \angle \pi$ (considering negative feedback).

Automatic tuning of PID Controllers – the Relay autotuner



Process gain at frequency 'ω':

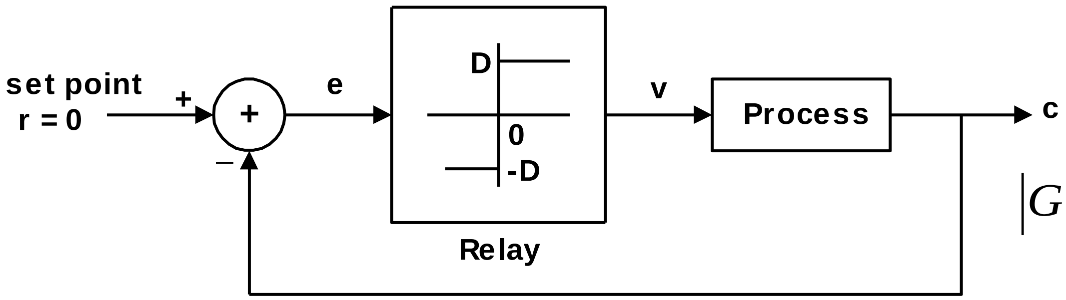
$$\left|G(\omega)\right| = \frac{output \quad amplitude}{input \quad amplitude} = \frac{A}{\frac{4D}{\pi}} = \frac{\pi A}{4D}$$

Now, to maintain steady oscillations at $\omega = \omega_c$, the loop gain is $1 \angle \pi$ (considering negative feedback).

Thus the gain of the relay controller (i.e. the critical gain) at $\omega = \omega_c$ is

$$K_c = \frac{1}{\left|G(\omega_c)\right|} = \frac{4D}{\pi A} \qquad \left[\text{as } K_c . \left|G(\omega_c)\right| = 1\right]$$

Automatic tuning of PID Controllers – the Relay autotuner



Process gain at frequency 'ω':

$$|G(\omega)| = \frac{output \;\; amplitude}{input \;\; amplitude} = \frac{A}{\frac{4D}{\pi}} = \frac{\pi A}{4D}$$

Now, to maintain steady oscillations at $\omega = \omega_c$, the loop gain is $1 \angle \pi$ (considering negative feedback).

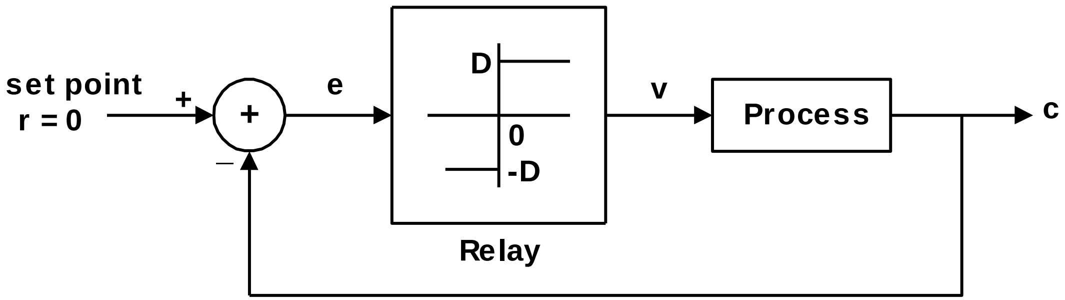
Thus the gain of the relay controller (i.e. the critical gain) at $\omega = \omega_c$ is

$$K_c = \frac{1}{|G(\omega_c)|} = \frac{4D}{\pi A} \qquad \left[\text{as } \mathrm{K_c}.|\mathrm{G}(\omega_c)| = 1\right]$$

Also $\angle G(\omega_c) = \pi$, as relay phase shift is zero.

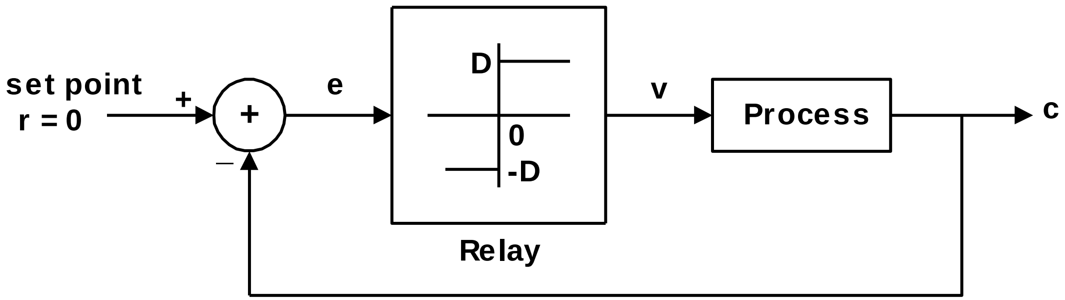
.

Automatic tuning of PID Controllers – the Relay autotuner



Thus by knowing the relay amplitude 'D' and by measuring the amplitude 'A' of the process output 'c', critical gain K_c may be determined $\left[K_c = \dfrac{4D}{\pi A} \right]$**.**

Automatic tuning of PID Controllers – the Relay autotuner

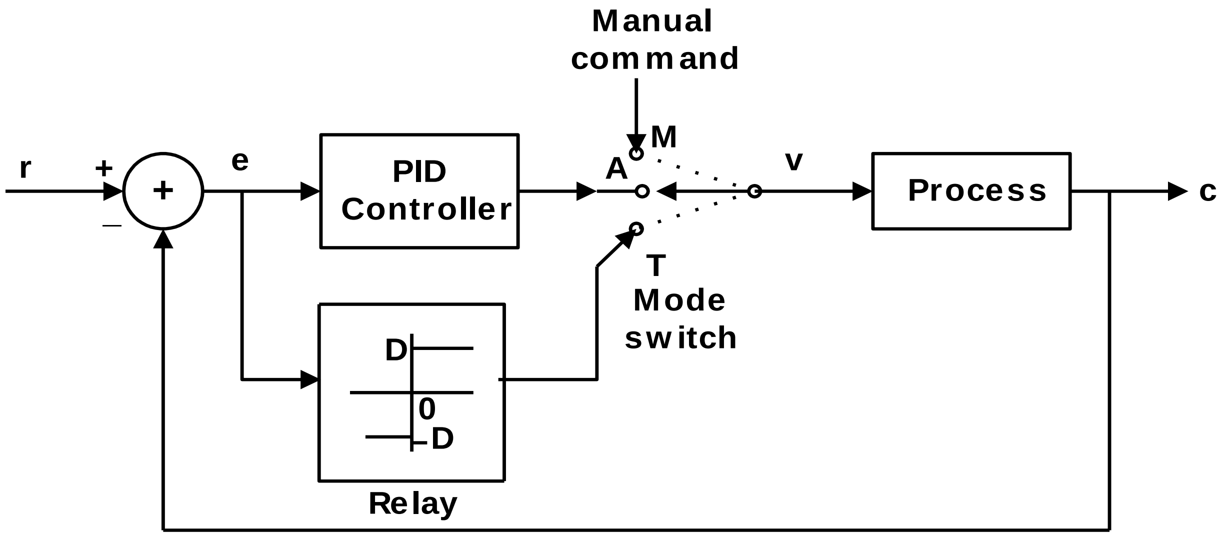


Thus by knowing the relay amplitude 'D' and by measuring the amplitude 'A' of the process output 'c', critical gain K_c may be determined $\left[K_c = \frac{4D}{\pi A}\right]$**.**

T_c **may be estimated by measuring the frequency of the output oscillation** $\left[T_c = \frac{2\pi}{\omega_c}\right]$**.**

Block diagram of the Relay autotuner

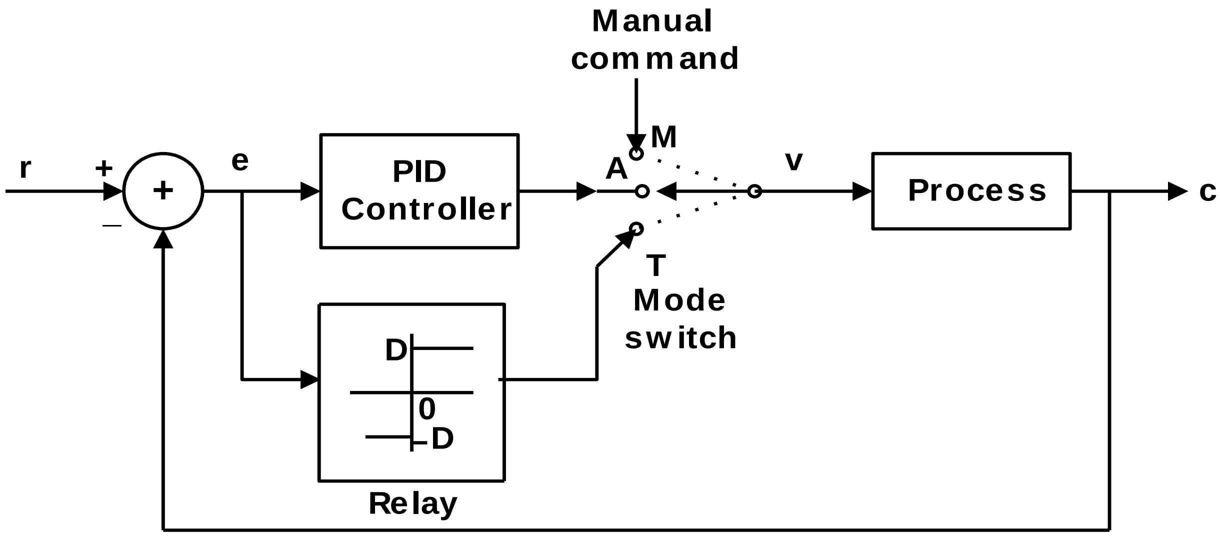
(The Satt Control Autotuner by Satt Control, Sweden)



M → Manual position
A → Auto position
T → Tune position

Block diagram of the Relay autotuner

(The Satt Control Autotuner by Satt Control, Sweden)

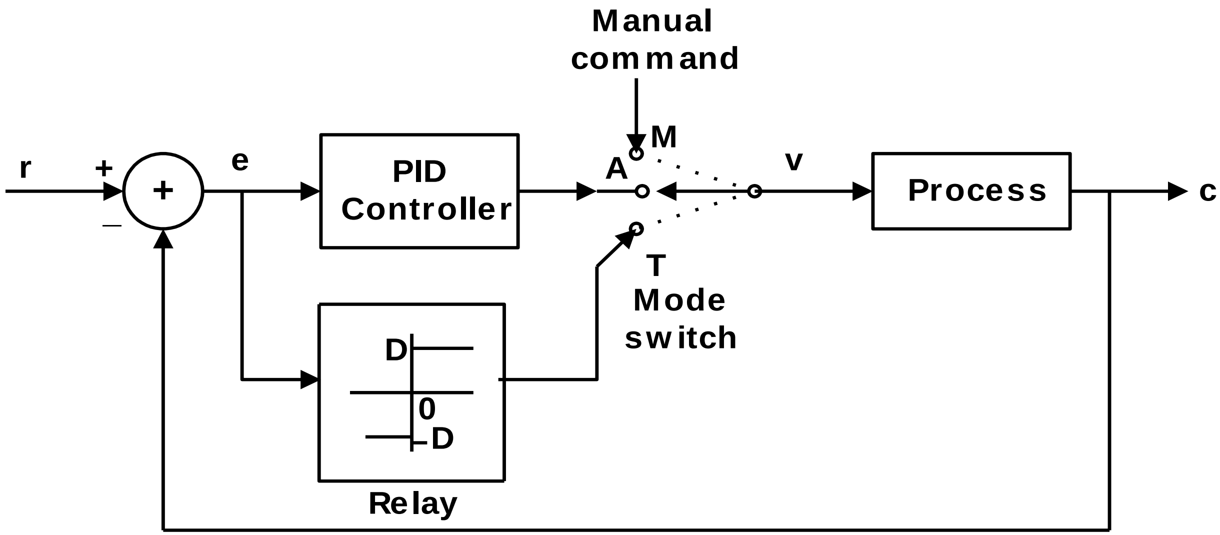


M → Manual position
A → Auto position
T → Tune position

At first, the process is brought to equilibrium state (≈ zero error), by setting a constant control signal in manual mode.

Block diagram of the Relay autotuner

(The Satt Control Autotuner by Satt Control, Sweden)

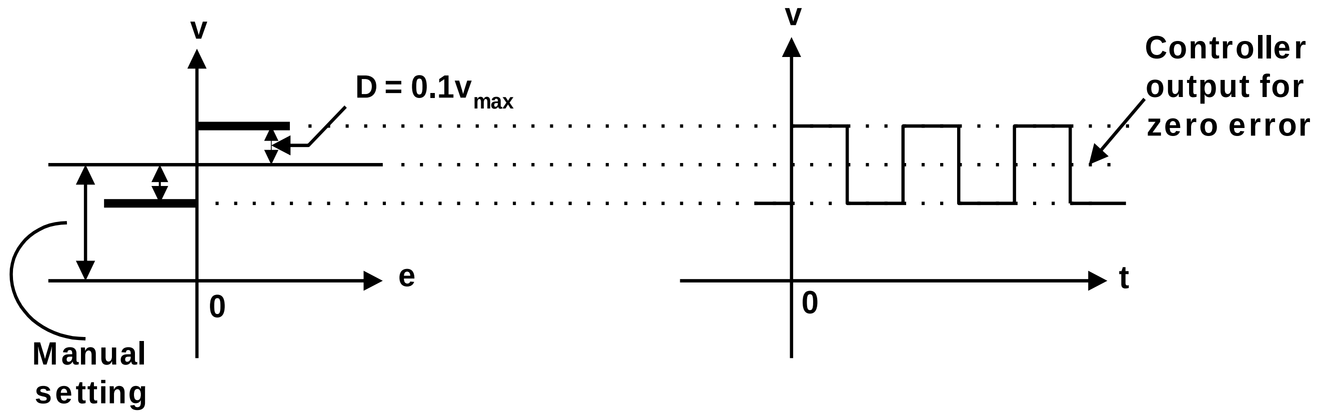


M → Manual position
A → Auto position
T → Tune position

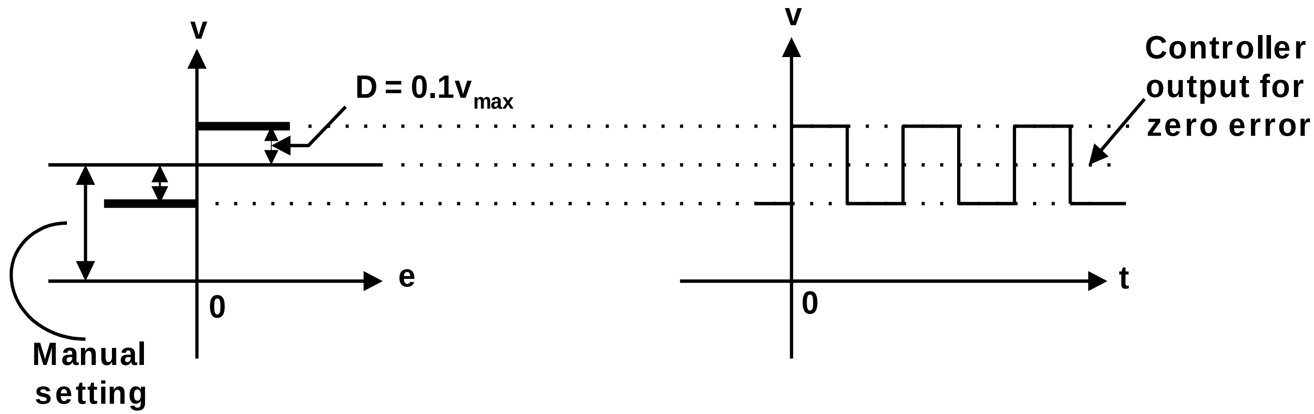
At first, the process is brought to equilibrium state (≈ zero error), by setting a constant control signal in manual mode.

The tuning is then activated by pushing the mode switch to tune position.

Modified relay characteristic for a non-zero set-point

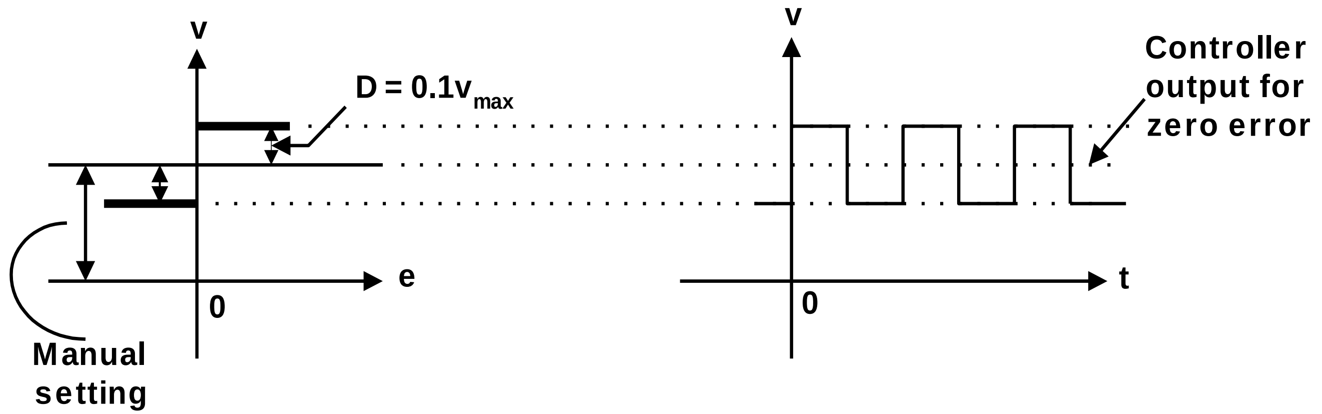


Modified relay characteristic for a non-zero set-point



➢ **The relay amplitude ‘D’ is initially set to 10% of the controller output-range.**

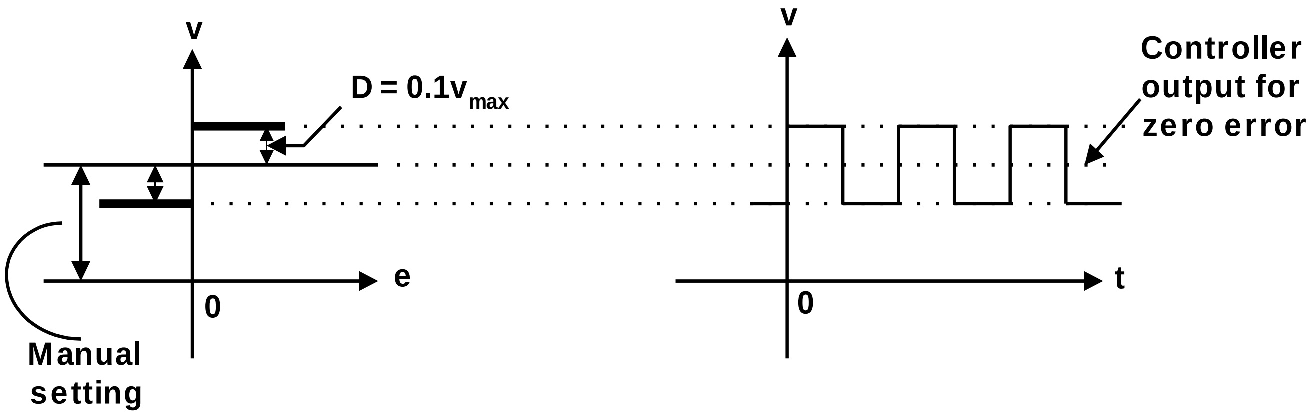
Modified relay characteristic for a non-zero set-point



➢The relay amplitude 'D' is initially set to 10% of the controller output-range.

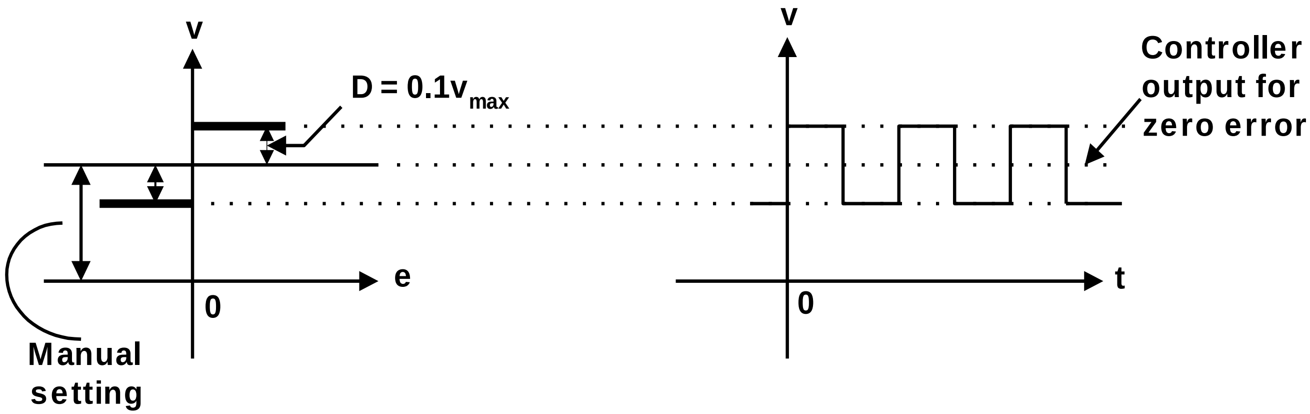
➢This amplitude is adjusted after one and a half period to give oscillation of 2% of the mean output.

Modified relay characteristic for a non-zero set-point



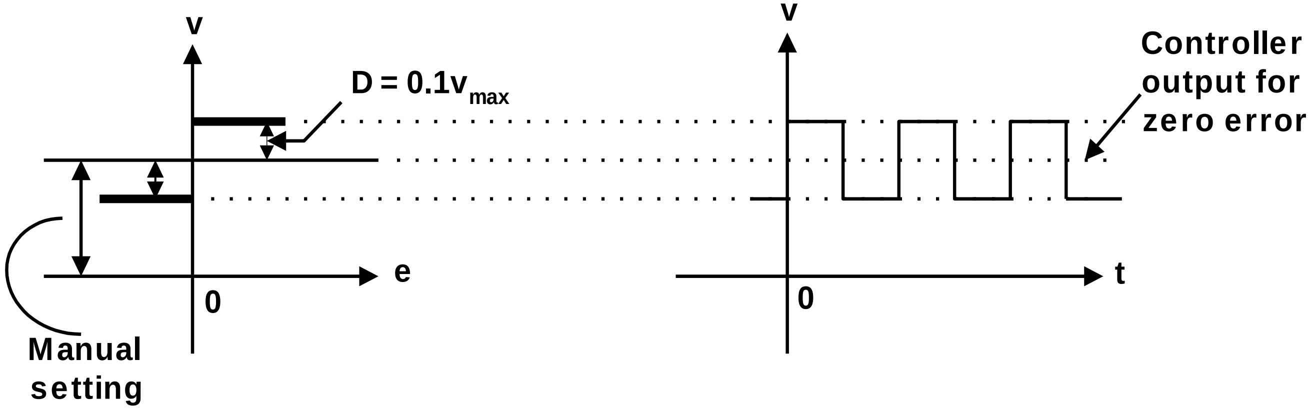
- ➢The relay amplitude ‘D’ is initially set to 10% of the controller output-range.
- ➢This amplitude is adjusted after one and a half period to give oscillation of 2% of the mean output.
- ➢This ensures minimum disturbance at the process output due to tuning.

Modified relay characteristic for a non-zero set-point



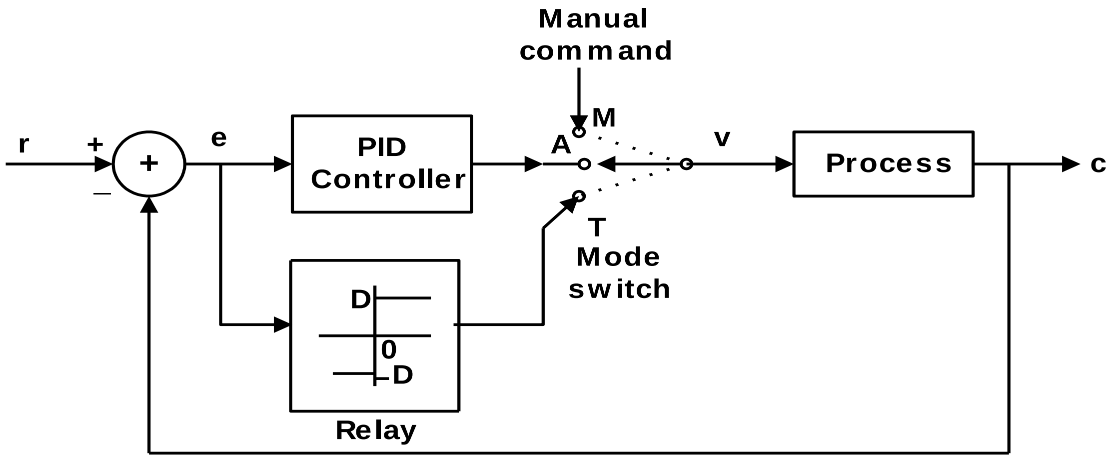
- The relay amplitude 'D' is initially set to 10% of the controller output-range.
- This amplitude is adjusted after one and a half period to give oscillation of 2% of the mean output.
- This ensures minimum disturbance at the process output due to tuning.
- This adjustment is done by measuring the change in output during the first one and a half period.

Modified relay characteristic for a non-zero set-point



- The relay amplitude 'D' is initially set to 10% of the controller output-range.
- This amplitude is adjusted after one and a half period to give oscillation of 2% of the mean output.
- This ensures minimum disturbance at the process output due to tuning.
- This adjustment is done by measuring the change in output during the first one and a half period.
- The modified relay amplitude is stored for the next tuning operation.

Satt Control Autotuner



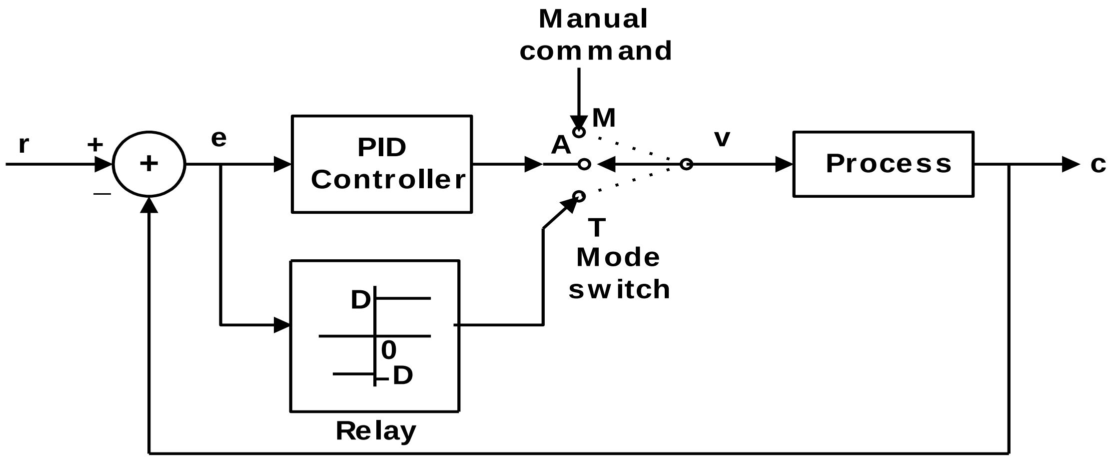
M → Manual position

A → Auto position

T → Tune position

The system is automatically switched to Auto mode after estimating the critical gain K_c and critical time period T_c during first 5½ period of oscillation.

Satt Control Autotuner



M → Manual position
A → Auto position
T → Tune position

The system is automatically switched to Auto mode after estimating the critical gain K_c and critical time period T_c during first 5½ period of oscillation.

The parameters of PID controller (viz. K_p, T_i and T_d) are determined from K_c and T_c according to Z-N rule.

References

1. **Process Control Systems** *by* Shinskey
2. **Automatic Process Control** *by* Eckman
3. **Principles of Process Control** *by* Patranabis
4. **Process Control** *by* Harriott
5. **Process Systems Analysis and Control** *by* Coughanowr and Koppel
6. **Process Control** *by* Pollard
7. **Chemical Process Control** *by* Stephanopoulos
8. **Modern Control Engineering** *by* Ogata
9. **Applied Process Control** *by* Chidambaram

Thank You