Unidad 6 - CONTROL DIGITAL:

6.A: Controladores basados en microprocesador , introducción:

Esta unidad tiene por objetivo familiarizar al alumno con los controladores PID basados en microprocesador (µP) y conocer algunas de las múltiples prestaciones de las que disponen.

Dado que las variantes son muchas, debido a las concepciones tanto teóricas: por ejemplo al considerar distintas posibles estructuras del controlador PID, que puede ser ideal o serie o paralelo, disponibilidad de lógica difusa como alternativa de control, etc. ; como prácticas: controlador unilazo, multilazo, con autosintonía, conexión a internet, tipo de interfase con el operador, etc.

Cabe mencionar también que el menor precio de los µP y mayor capacidad a desplazado a los controladores neumáticos y electrónicos basados en operacionales, predecesores analógicos del actual controlador digital.

Se ha pensado que un medio idóneo para alcanzar el objetivo de esta unidad es el desarrollo del algoritmo PID (se ha elegido la estructura ideal) en forma de diagrama de flujo computacional y explicar algunas de las prestaciones como subrutinas.

6.B: Elementos de hardware:

La arquitectura del controlador PID digital no difiere, básicamente, de la de cualquier computador, el Gráfico 1 muestra tal esquema.

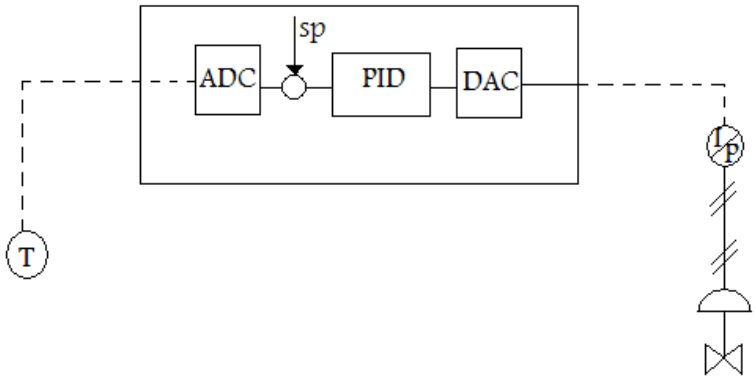


Figura 1

La entrada (o entradas para controladores multilazos) será una señal que en 4 – 20 mA llega desde un transmisor en campo y la salida (o salidas para controladores multilazos) también en 4 –20 mA, irá al convertidor de corriente a presión (I/P) si el elemento final de control es una válvula o directamente en corriente si es un variador de frecuencia.

La Figura 1 muestra la estructura básica de un controlador digital, la señal de entrada continua es convertida a un valor numérico digitalizado mediante un ADC(convertidor analógico-digital) , normalmente de 12 bits de resolución , que se suma algebraicamente con el valor deseado o set-point que también ha sido ingresado por el operador y digitalizado.

Veremos más adelante que el programa principal *usa una subrutina llamada ADC, como resultado de ello se obtiene el error digitalizado*. Así mismo, la salida digitalizada del programa principal o bloque PID es *entregada a una subrutina DAC que utilizando un conversor digital - analógico entrega una salida que en combinación con un "holder" o mantenedor de señal , colocan una señal continua en 4-20 mA para ir a un convertidor I/P o a la entrada de un variador de frecuencia,* como ya se dijo.

La Figura 2 amplía el detalle del hardware, aquí es importante destacar que la memoria ROM, además del firmware de arranque y del algoritmo PID, contiene una biblioteca de funciones , por ejemplo, para linealizado de termocuplas, según el tipo que se trate (J, K, R, S...), totalizador de caudales, álgebra para cálculos sencillos, etc.

Dichas funciones le dan características particulares a cada controlador , por lo que varían notablemente entre uno y otro, lo que hace necesario, ante una aplicación determinada, tener en claro los requerimientos del mismo.

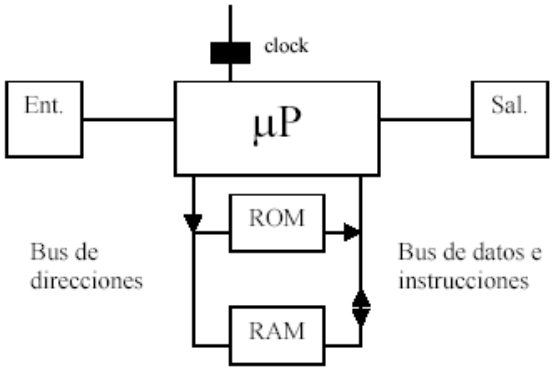


Figura 2

Este esquema puede completarse con otras prestaciones tales como salida de comunicaciones digitales RS-422 o RS-485 (que mediante una interfase a RS-232C posibilita el dialogo con PC´s), salidas USB, panel frontal de operación, etc.

Dependiendo de la velocidad y capacidad del µP , junto con el hardware asociado, pueden constituirse controladores unilazo o multilazo, esto es que más de un controlador PID puede implementarse con un solo µP, la Figura 3 muestra y la Figura 4 un detalle simplificado del multiplexado de entradas y desmultiplexado de salidas.

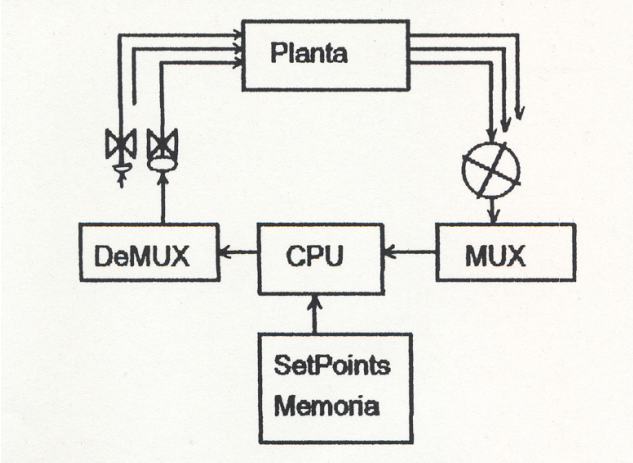


Figura 3

La Figura 3, indica que el multiplexador de entrada (a la izquierda del gráfico) está recibiendo señales continuas de más de un transmisor, a su vez el desmultiplexador y sostenedores de señales de salida (holder) entregan señales con destino a varios accionadores finales de control.

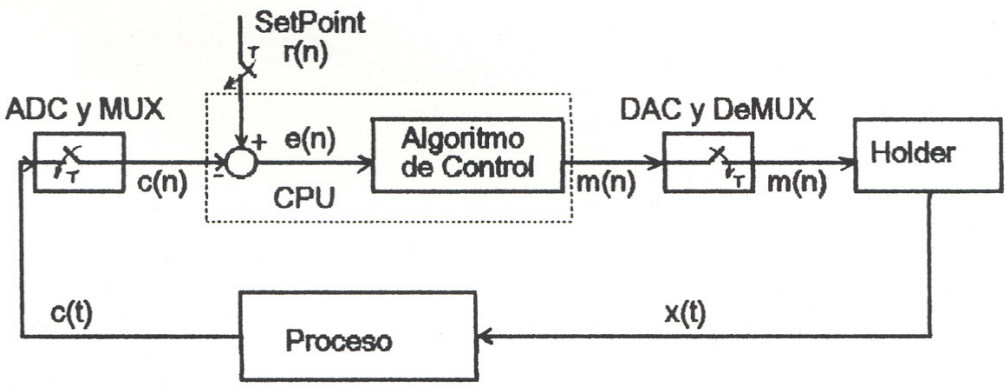


Figura 4

Dependiendo del fabricante y modelo los controladores multilazo van desde 2 hasta 128 controladores PID con un misma CPU, los PLC´s también pueden resolver controladores PID multilazo con su CPU, al tiempo que también procesan esquemas de control de contactos (álgebra de Boole – Unidad 7), para ello en general se deben adquirir los módulos de entradas y salidas analógicas.

6.C: Funciones de transferencias en lazos continuos y muestreados:

La Figura 5 muestra un lazo de control realimentado en el dominio de la variable s.

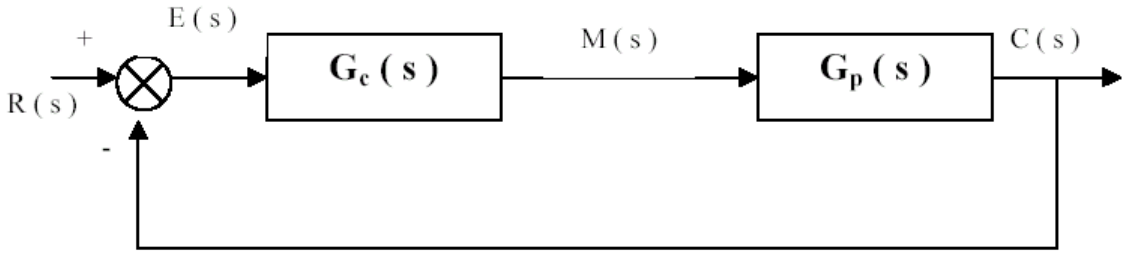


Figura 5

La función de transferencia del controlador $G_c(s)$ es:

$$G_c(s) = \frac{M(s)}{E(s)} = K_C \cdot (1 + \frac{1}{T_i \cdot s} + T_d \cdot s)$$

La FT corresponde a un controlador PID denominado ideal ya que los tres términos que la componen no interactúan entre si.

Se debe aclarar expresamente que el tratamiento interno en la CPU del algoritmo tipo PID se debe hacer en el campo de los sistemas muestreados, esto es el campo de la variable z, la Figura 6 indica tal tratamiento. No obstante no se hará un análisis en tal dominio, ya que lo que interesa es mostrar la prestaciones de un controlador tipo PID a través del software que lo realiza.

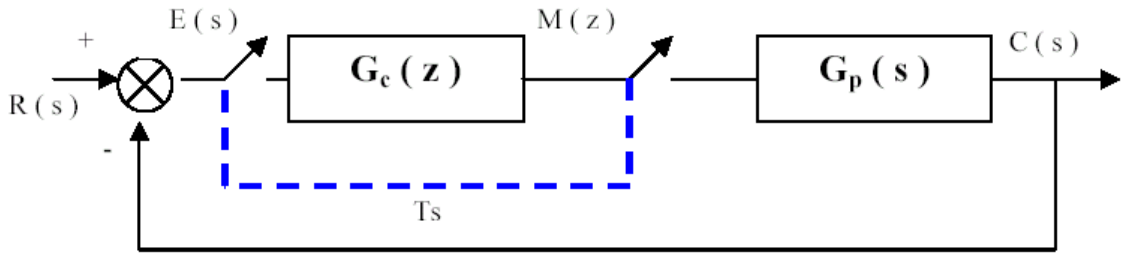


Figura 6

Si es relevante para nuestros propósitos notar el Ts o tiempo de muestreo (time sample) ya que el desempeño de los lazos de control están estrechamente ligados a la variación de la velocidad de los cambios de las variables a controlar, mientras más rápidos sean los procesos, menor debe ser el Ts, caso contrario no se podría hacer CONTROL en TIEMPO REAL.

6.D: Elementos de software:

A los efectos de analizar el software es conveniente recordar algunos métodos numéricos, a partir del mismo algoritmo PID en el **dominio del tiempo**, representado a continuación en orden de la función derivativa, proporcional e integral respectivamente:

$$m(t) = K_c \cdot Td \cdot \frac{d\,e(t)}{dt} + K_c \cdot e(t) + \frac{K_c}{T_i} \cdot \int e(t)\,dt$$

Para digitalizar tales formas, sabemos que la derivada de una función en un instante k está dada por la expresión siguiente:

$$\left.\frac{df}{dt}\right|_k = \frac{f_k - f_{k-1}}{\Delta t}$$

Donde el Δt es el tiempo entre muestreo y muestreo.

A su vez la función integral está representada por la próxima expresión:

$$\int f(t)\,dt\Big|_k = \sum_{k=0}^{n} f_k \cdot \Delta t$$

Si ahora consideramos un instante enésimo en el que tiene lugar una salida m_n del controlador PID, a la vez que llamamos e_n al error actual y e_{n-1} o $error_{old}$ por anterior al e_n, se tiene que la expresión temporal anterior pasa a ser la siguiente digitalizada:

$$m_n = K_c \cdot \left[Td \cdot \frac{(e_n - e_{n-1})}{\Delta t} + e_n + \frac{1}{T_i} \cdot \sum_{k=0}^{n} e_k \cdot \Delta t \right]$$

Si asumimos que Δt es igual al Ts y que la variable **s** es igual a sumatoria Σ de los errores de la función integral y si finalmente hacemos:

- **Kp=Kc**
- **Ki=Kc.Ts/Ti**
- **Kd=Kc.Td/Ts**

Donde:

- **T_s= ΔT**
- **S_n=S_{n-1}+e_n**

Se tiene que:

$$m_n = K_p \cdot e_n + K_i \cdot s_n + K_d \cdot (e_n - e_{n-1})$$

Siendo esta última expresión la digitalizada del algoritmo PID y que analizaremos mediante un diagrama de flujo, al que después le agregaremos subrutinas a fin de comprender las prestaciones más comunes de un controlador digital.

Los controladores digitales se configuran a través de un Menú de opciones, si por ejemplo el usuario decide usar sólo acción proporcional , inmediatamente aparece el bias b(t) o polarización y solicita se ingrese un valor para el mismo, usualmente el 50%. Por el contrario, si escoge acción P + I, desaparece el bias. Por lo explicado la salida para un controlador sólo proporcional será:

$$m_n = K_p \cdot e_n + b_n$$

Además y por razones que tienen que ver con la naturaleza de los procesos, es conveniente adelantar en este punto que la acción derivativa rara vez se emplea, tanto es así que hay autores que la consideran una acción de excepción, esto es así dado que la mencionada acción se aplica sólo a procesos capacitivos de forma tal, que ante un cambio de valor deseado r(t), por ejemplo en escalón, se adelante la respuesta de la variable controlada c(t) ; en general se aplica a lazos de temperatura donde es alta la inercia térmica y en niveles de gran capacitancia, su uso en otras variables, con alta velocidad de variación , como caudales, presión , etc. conduce a un mal desempeño del lazo ya que deriva frentes abruptos de c(t) que desencadenan impulsos en la salida m(t) , llevando al elemento de acción final de un extremo a otro, haciendo cuando menos, inútil la acción de control o perjudicial la mayor parte de las veces.

Con el fin de que la acción derivativa sea lo más eficiente posible, dentro del menú de opciones de los controladores digitales, está la posibilidad implementar un filtro pasa bajos por software y a veces , en otros , su realización es por hardware (filtro RC) como lo muestra la Figura 7, en ambos casos su misión es la de eliminar ruidos de alta frecuencia entrantes junto con la señal a procesar, normalmente la constante de tiempo del filtro mencionado es la mitad del tiempo de muestreo Ts.

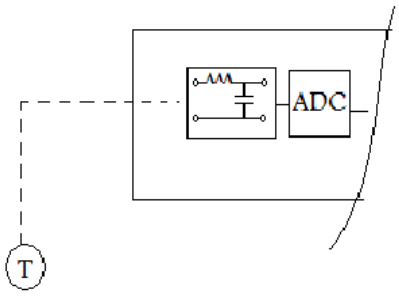


Figura 7

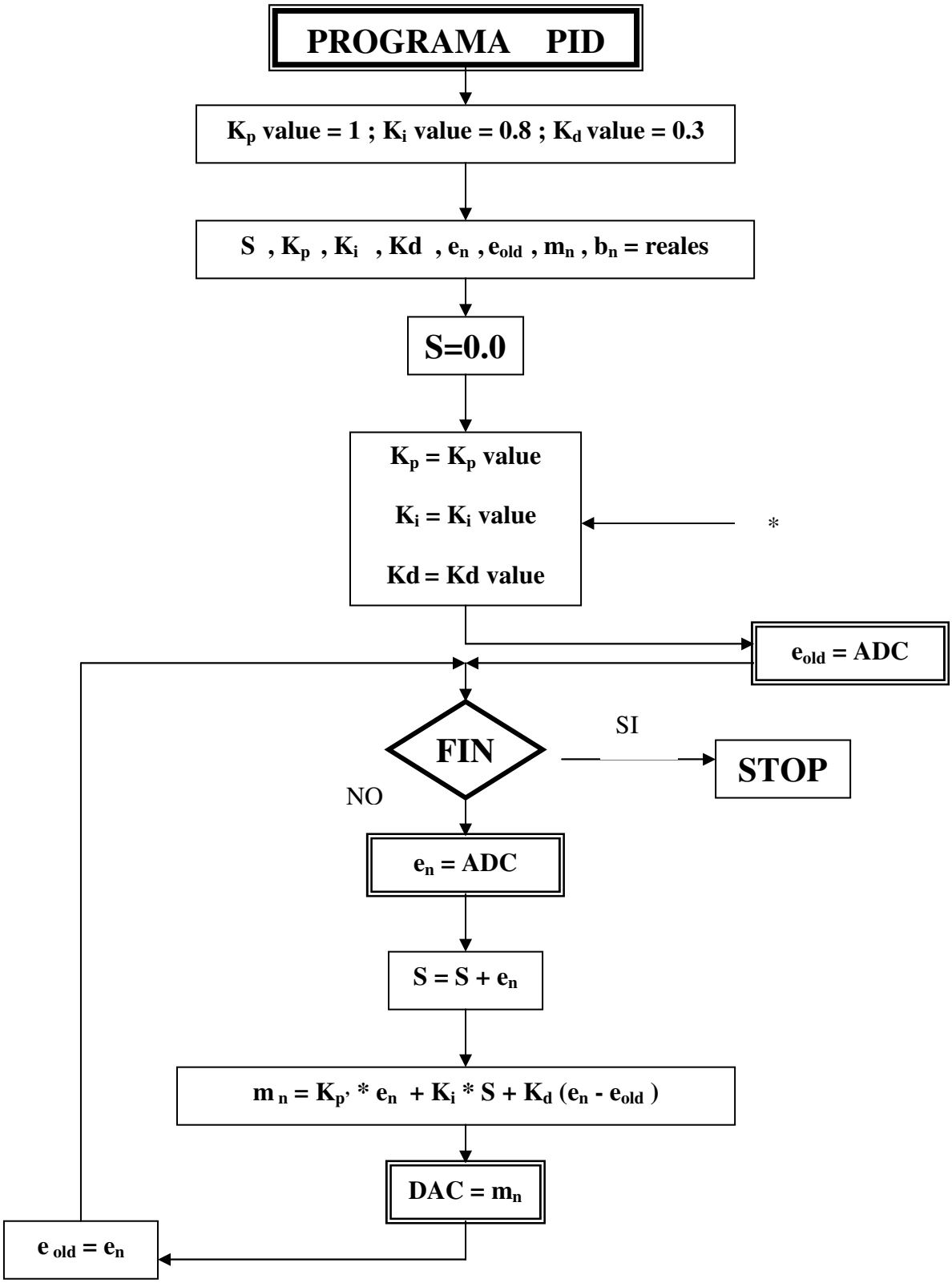
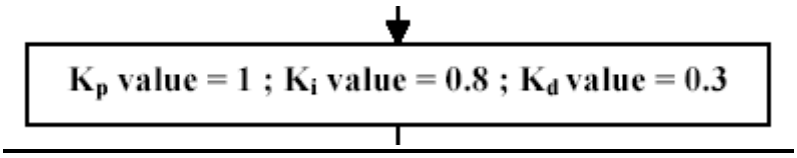

PROGRAMA PID
Kp value = 1 ; Ki value = 0.8 ; Kd value = 0.3
S , Kp , Ki , Kd , en , eold , mn , bn = reales
S=0.0
Kp = Kp value
Ki = Ki value
Kd = Kd value
*
eold = ADC
FIN
SI
STOP
NO
en = ADC
S = S + en
m n = Kp' * en + Ki * S + Kd (en - eold)
DAC = mn
e old = en

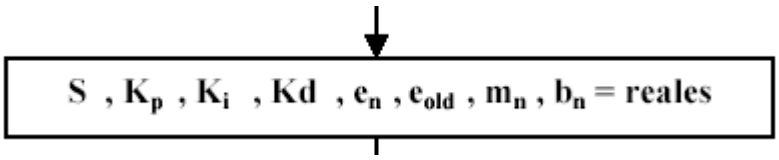
Figura 8

6.E: Análisis del algoritmo PID digitalizado: en la Figura 8, se observa el diagrama de flujo principal del algoritmo PID, que describiremos en sus etapas más significativas:

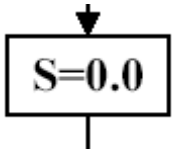
a) El fabricante suele brindar valores por default a los parámetros proporcional, integral y derivativo, el lazo de control tendrá un desempeño no aceptable pero arrancará, luego tendrá el usuario la posibilidad de aplicar los correctamente calculados por él (Unidad 2).



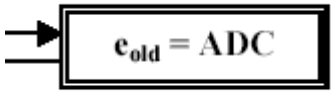
b) Esta declaración de variables reales es para significar que estos controladores trabajan con aritmética de punto flotante y por lo tanto pueden alcanzar valores muy grandes o muy pequeños, más adelante veremos que en cierto modo esta posibilidad acarrea un inconveniente llamado de saturación integral (windup).



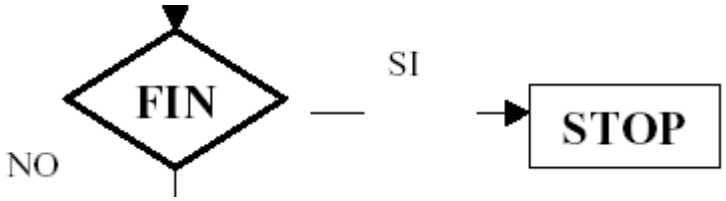
c) El acumulador integral S es colocado a cero , antes de comenzar a ejecutarse el algoritmo PID en el “loop” del programa.



d) Se carga el error anterior , aquí denominado e_{old} ó e_{n-1} a partir de llamar a la subrutina ADC, que como ya se dijo, entrega el error que en ese momento ha procesado.

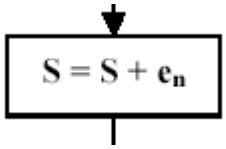


d) Aquí se pregunta por terminar o no, pero no el programa, sino que si se opta por SI, lo que se pretende es cambiar los valores de los parámetros de las acciones proporcional, integral y derivativa mediante una subrutina que veremos más adelante, si se opta por NO, se entra al "loop" del programa.

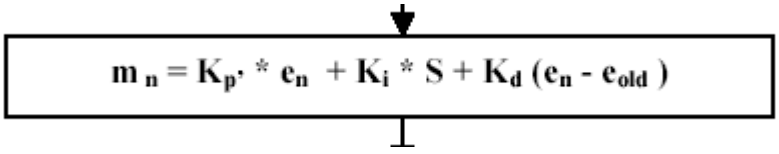


e) En esta etapa se llama nuevamente a la subrutina ADC, cuyo salida, el error, es asignado al error actual e_n , es obvio que al inicio del programa e_n es igual a e_{old} , pero es por única vez , ya que luego e_{old} se refrescará dentro del "loop" del programa a partir de e_n

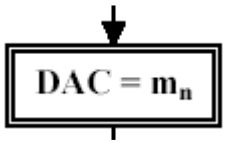
f) Si hay error, el valor de S empieza a tener un valor distinto de cero.



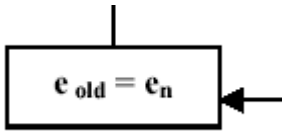
g) Si hay error, el programa calculará una salida m_n , si se ha aplicado acción derivativa, esto de K_d distinto de cero, únicamente en la primera corrida esta acción no aportará a la salida mencionada, en las corridas posteriores si, siempre y cuando los errores actuales y anteriores sean diferentes.



h) La subrutina DAC recibe un valor de la salida m_n , que se transforma en un valor continuo, que junto con el holder producen una salida en 4 a 20 mA, como ya se explicara.



i) Se actualiza el error anterior a partir del actual y comienza un nuevo ciclo del "loop" del programa, notar que a partir de este segundo ciclo, los valores de tales errores no serán necesariamente iguales.



6.F: Subrutinas complementarias: para que lo visto represente a un controlador PID practicable y útil debemos dotarlo de:

a) Sincronización de Ts con el tiempo real y no en función de la velocidad de procesamiento de la CPU.
b) Posibilitar la transferencia automática a manual y manual a automática sin salto (bumpless).
c) Posibilitad de evitar la saturación integral (integral windup).
d) Permitir al instrumentista los cambios de los parámetros Kp, Ki y Kd.

Desarrollaremos ahora estas importantes subrutinas complementarias del programa principal ya visto.

a) Sincronización de Ts con el tiempo real y no en función de la velocidad de procesamiento de la CPU: la Figura 9 muestra esta subrutina, toda computador industrial dispone de un reloj, que puede ser consultado mediante una subrutina llamada, por ejemplo, GET_TIME, cada vez que se la requiere, dará el tiempo real en horas, minutos, segundos, valor cronológico que puede ser asignado a cualquier variable como TIME, a su vez podemos poner en juego una segunda variable como NEXT SAMPLE TIME (NST abreviada) que es igual a TIME + Ts , esto es importante ya que de esa forma el algoritmo queda "suspendido" después de ejecutarse el siguiente bloque del "loop" del programa principal:

Para procesos como los de refinerías, petroquímicas e industrias similares, el Ts es del orden de una centésima de minuto, esto es 0,6 segundos, durante ese tiempo el algoritmo PID de un lazo en particular, además de sincronizarse con el tiempo real, permite al hardware atender otros lazos PID, si el controlador fuere multilazo.

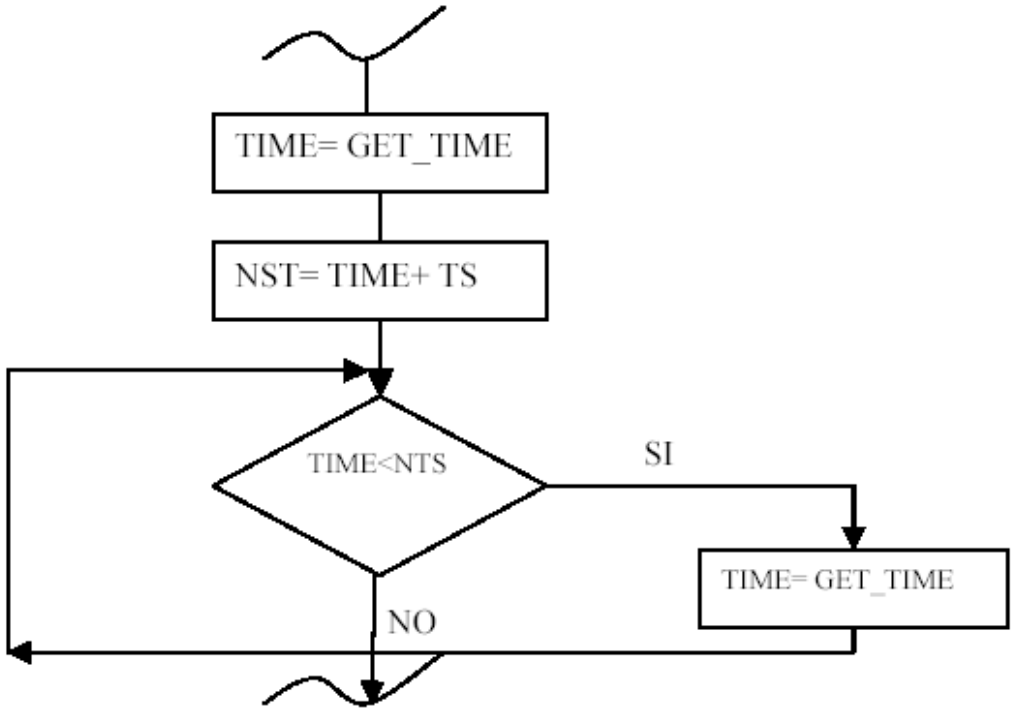
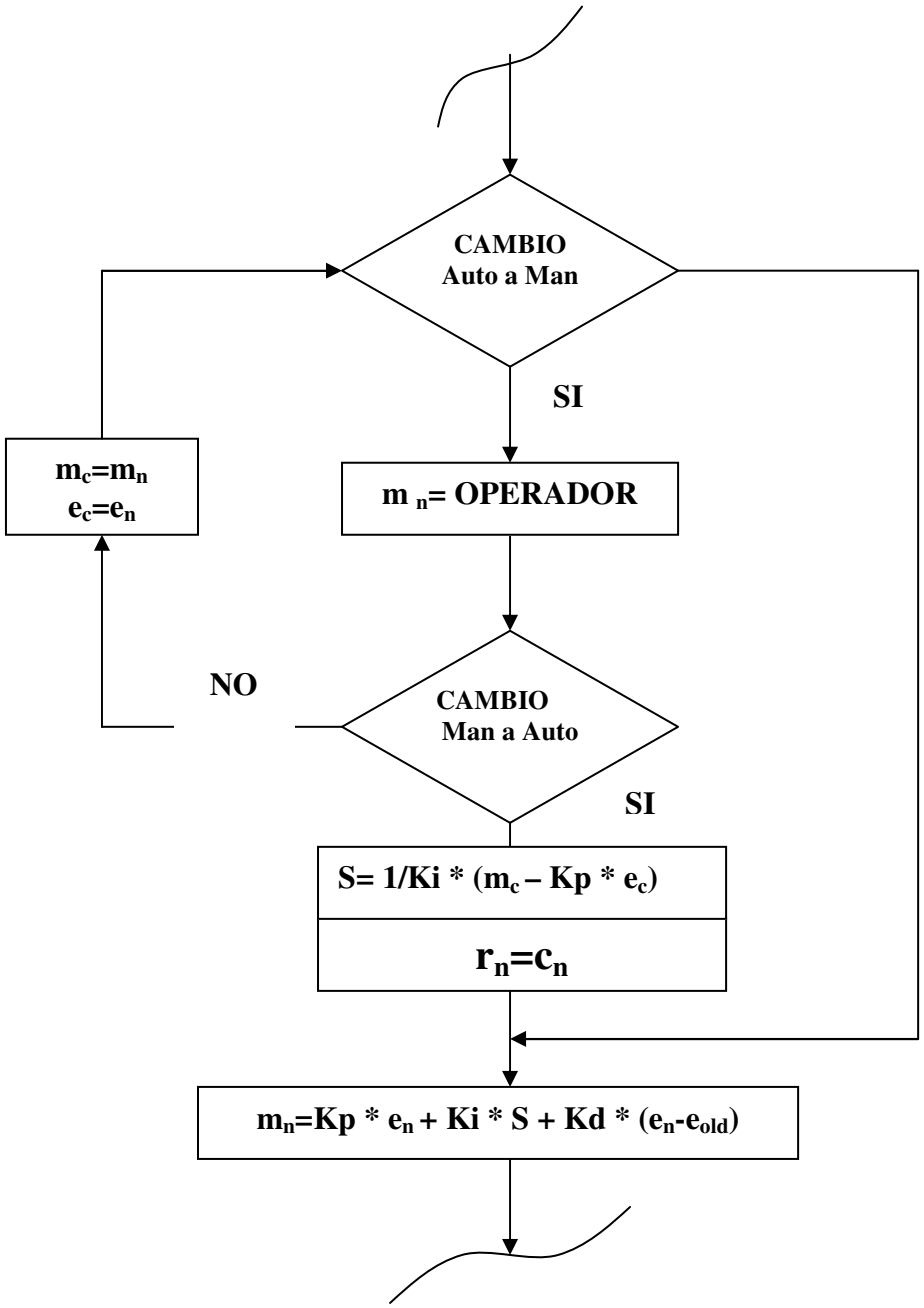


Figura 9

Ya dentro del "loop" de la subrutina, hasta que TIME no sea igual o mayor a NST, el programa de ese PID que interrumpido por un lapso igual a Ts, ver que TIME se actualiza hasta que eso sucede.

b) Posibilitar la transferencia automática a manual y manual a automática sin salto (bumpless) : esta subrutina va a continuación de la anterior, aquí el problema es que todo controlador tipo PID, puede pasar a manual , en ese caso la m_n queda en función del operador y no del algoritmo , puede

haber un salto muy grande, que desestabilizará el proceso controlado cuando se retorna de manual a automático ya que r(t) seguramente no coincide con c(t) y el error e(t) será significativo. Para evitar dicho salto en la salida, se emplea la siguiente subrutina (Figura 10):



Mientras el operador no opta por pasar de automático a manual , se ejecuta la salida normalmente, cuando elige pasar a automático es él quien maneja la salida, no obstante se genera un "loop" de seguimiento o de "tracking", esto es que cada cambio que hace el operador , la salida se asigna a una variable m_c y se mide el error en que es asignado a e_c, cuando el operador

decide pasar de manual a automático, para evitar el salto en la salida m_n, se calcula un valor de S en base a los valores del "tracking" y se hace que r_n sea igual a c_n, la salida m_n normal que se ejecuta a continuación no saltará por no haber error y contar con una valor de S que deja la salida igual a la que dejó el operador inmediatamente antes de pasar a automático, evitando así el salto, a partir de ese momento , S se actualizará según el "loop" principal del programa.

b) Posibilidad de evitar la saturación integral (integral windup): esta subrutina está a continuación de la anterior descripta, sucede que si se deja que el valor de S alcance valores muy altos o muy bajos, ante un cambio de set-point o r(t) en un sentido contrario a uno anterior, ese valor de S debe incrementar o decrementar según sea el caso y si alcanzó valores extremos tarda mucho tiempo en tomar el valor que para ese momento corresponda, es por ello que el usuario puede poner límites tanto inferiores como superiores, si bien la acción integral pierdo un mínimo de desempeño, por este medio e evita la saturación, caso contrario el lazo de control pierde desempeño en forma notable, la Figura 11.a muestra esta subrutina y la Figura 11.b, la respuesta de un sistema con y sin subrutina para evitar la mencionada saturación.

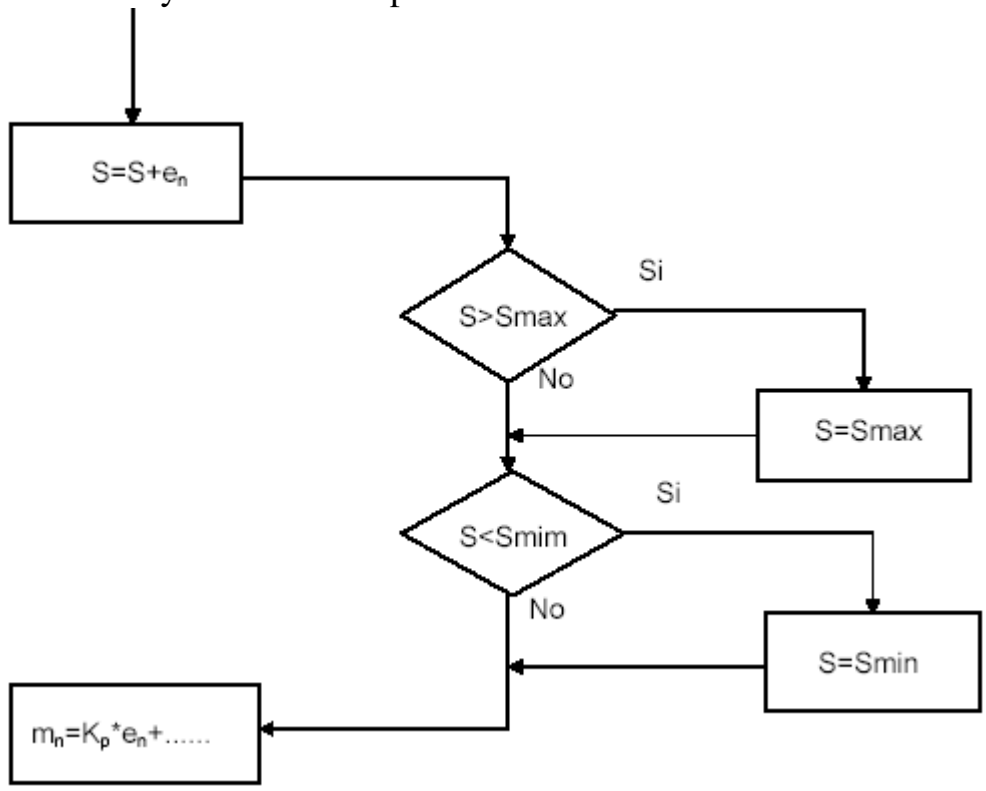
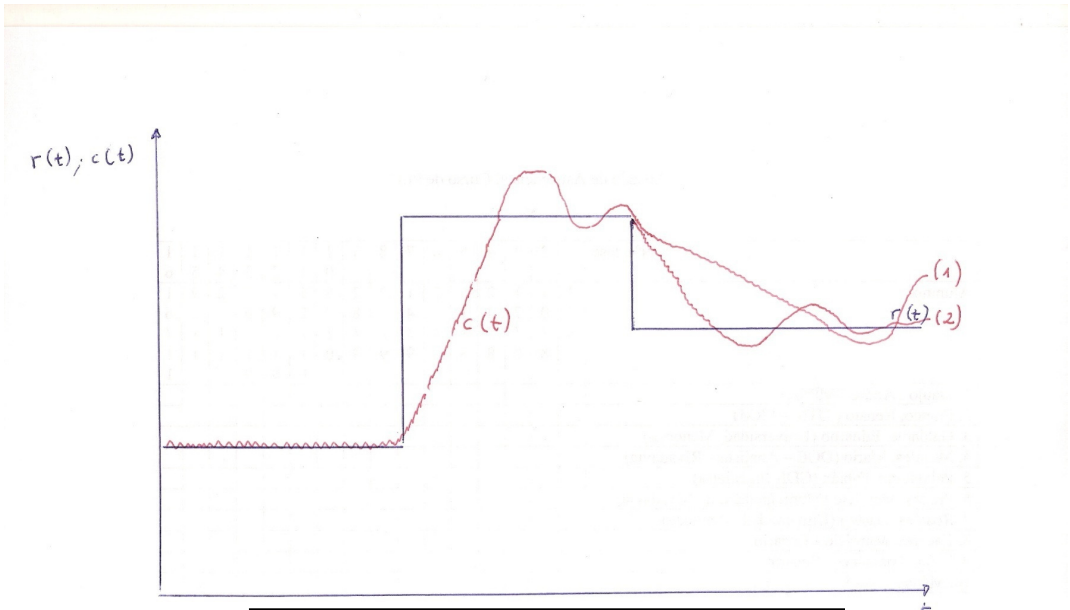


Figura 11.a



1) Sin limitación de la acción integral
2) Con limitación de la acción integral

Figura 11.b

d) Permitir al instrumentista los cambios de los parámetros Kp, Ki y Kd: cuando estaba la opción de FIN, se comentó que es para cambiar los parámetros mencionados, efectuados dichos cambios, se retorna al programa principal en el punto indicado con el asterisco, la Figura 12, muestra ésta última subrutina.

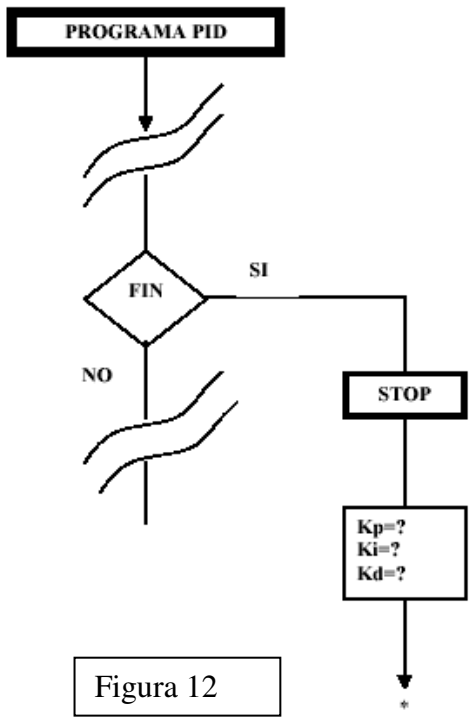


Figura 12

Un comentario sobre la acción derivativa: el propósito de esta acción, como ya se adelantara, es acelerar la respuesta del lazo cerrado en procesos altamente capacitivos, pero es normal que al introducir cambios en el set point o r(t) instantáneamente , separándose bruscamente de c(t) y por consiguiente generando un error e(t) considerable, la acción derivativa amplifique grandemente, provocando grandes fluctuaciones en la salida m(t) que a su vez las traslada a la variable controlada c(t), *por lo que se puede decir que la acción derivativa aplicada sobre el error es hipersensible a las variaciones de set-point*, una alternativa para superar este problema es modificar el algoritmo de control, que como ya sabemos es:

$$m_n = K_p \cdot e_n + K_i \cdot s_n + K_d \cdot (e_n - e_{n-1})$$

Ahora en vez de que acción derivativa sea el producto de K_d por la diferencia de los errores actual y anterior, sea el producto de K_d por la diferencia del valor de la variable medida o controlada actual y anterior, esto es:

$$m_n = K_p \cdot e_n + K_i \cdot s_n - K_d \cdot (c_n - c_{n-1})$$

Nota: el cambio de signo en la acción derivativa se visualizada mejor en el dominio del tiempo y se debe a que:

$$e(t) = r(t) - c(t), \quad si \quad r(t) = cte, \quad \frac{de(t)}{dt} = \frac{dr(t)}{dt} - \frac{dc(t)}{dt} = \frac{dcte}{dt} - \frac{dc(t)}{dt} = -\frac{dc(t)}{dt}$$

La Figura 13 muestra el algoritmo PID con acción derivativa sobre el error y la Figura 14 sobre la medición de la variable controlada c(t).

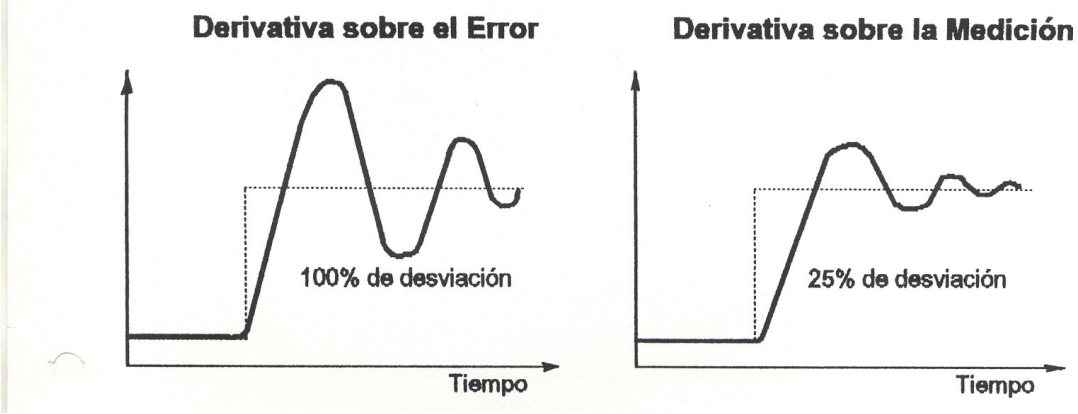


Figura 13

Figura 14

Finalmente la figura 15 muestra un diagrama en bloque con los componentes principales de un controlador digital y su vinculación con el proceso.

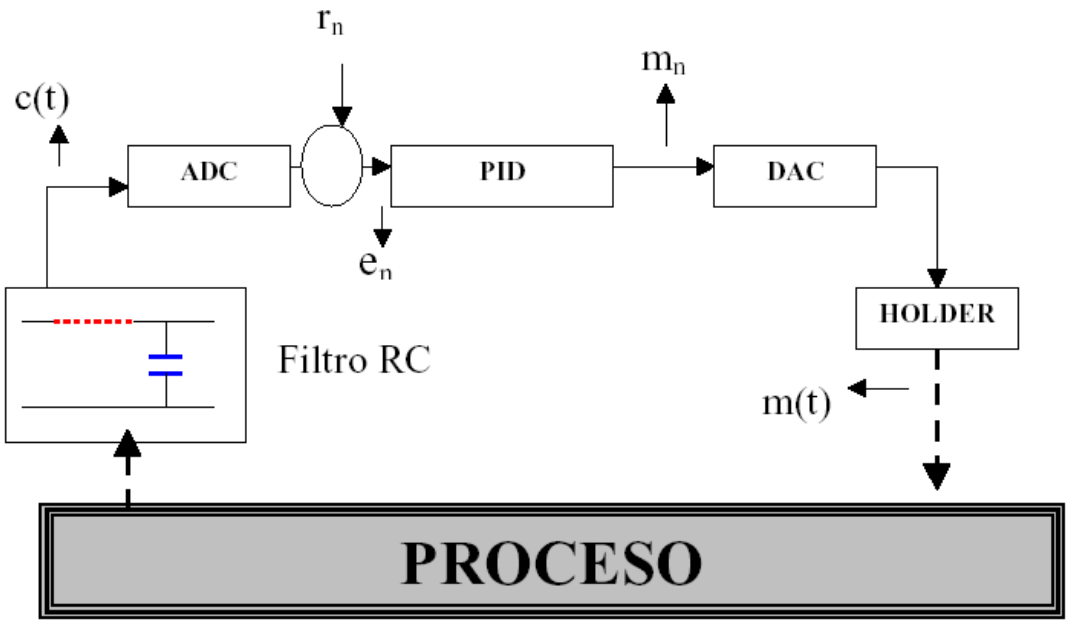


Figura 15

En la prácticas de Laboratorio de la Cátedra de Instrumentación y Control Automático, se utilizan controladores como los descriptos en esta Unidad : en la Planta de control de nivel de una columna mediante un controlador PID unilazo , mientras que la Planta para el control de caudal en una línea, utiliza también un controlador digital PI pero integrado en un variador de velocidad para motor trifásico de CA.

A continuación se muestra en la Figura 16 un controlador digital unilazo y la Figura 17 el panel del mismo (cortesía de la firma Foxboro).

Figura 16

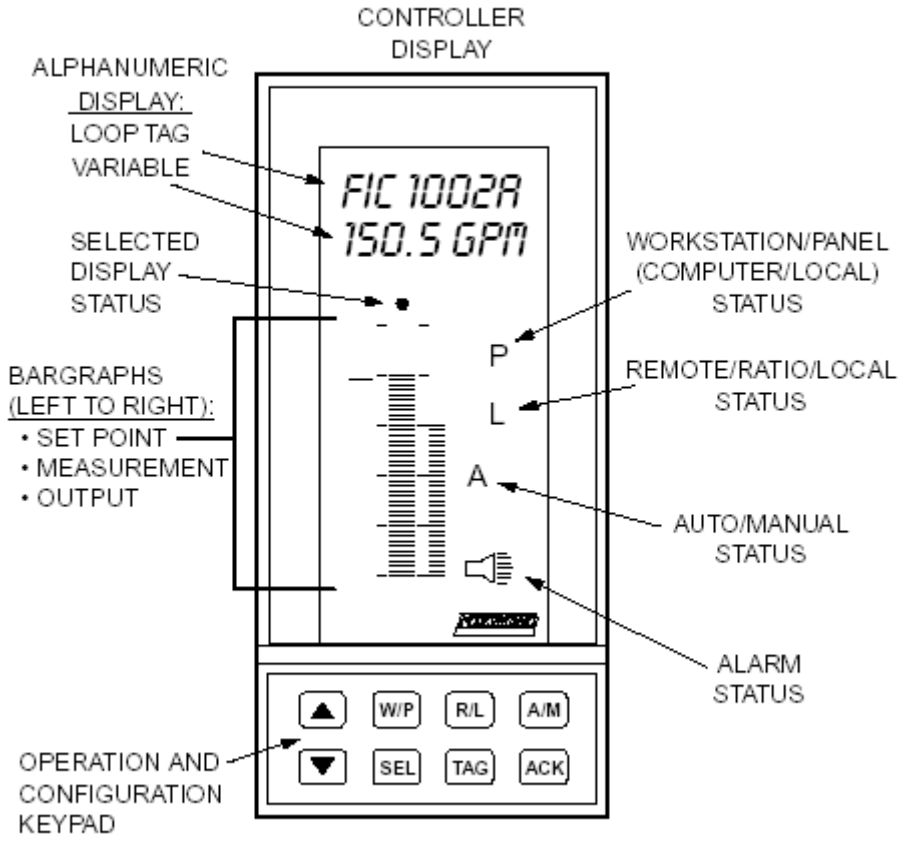


Figura 17

Profesor Titular: Ing. Alfredo Ernesto Puglesi
Profesor Adjunto: Ing. María Susana Bernasconi
JTP: Ing. Esther Bibiana Castiglione
Colaboró: Pablo Barbazza