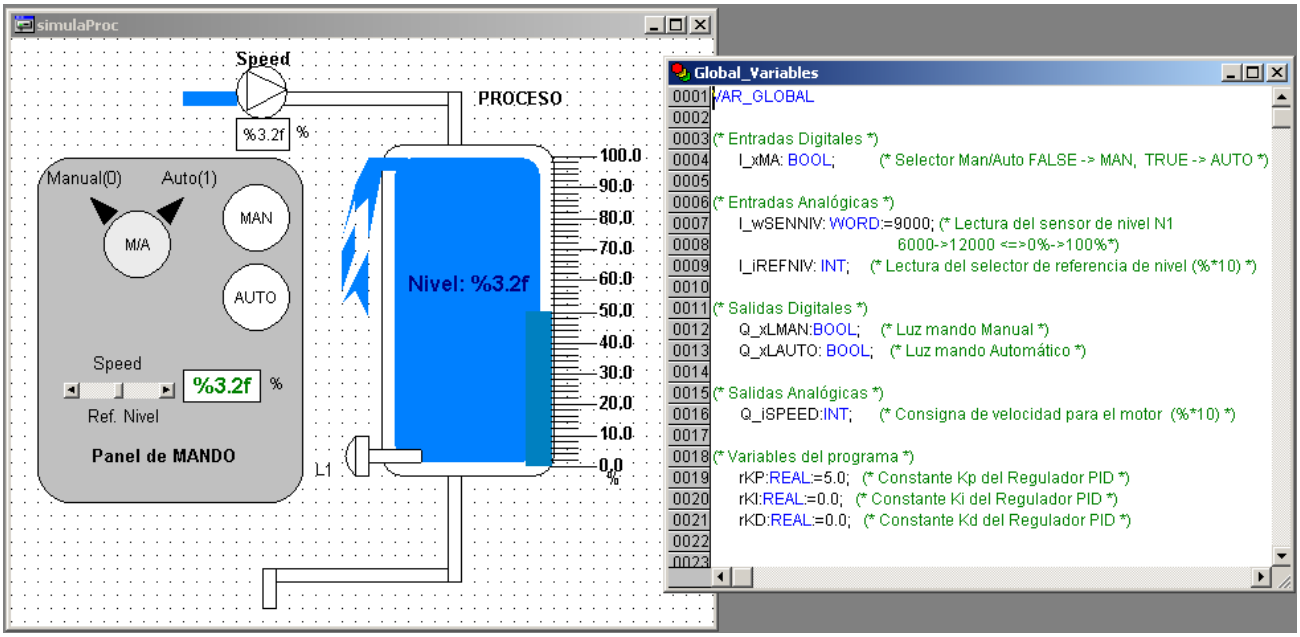
# IMPLEMENTACIÓN DE UN REGULADOR PID

**OBJETIVOS:**

- POUs: programa, función y acciones.
- Gestor de tareas. Tareas cíclicas.
- Entradas y salidas analógicas.
- Implementación de un PID.

**ESQUEMA DEL PROCESO:**

La figura representa un sistema utilizado suministrar líquido controlando su nivel en un tanque:




**ELEMENTOS Y SEÑALES DEL SISTEMA:**

El sistema consta de un ***Tanque Principal*** con una capacidad determinada de producto, que incluye los siguientes elementos sensores y actuadores:

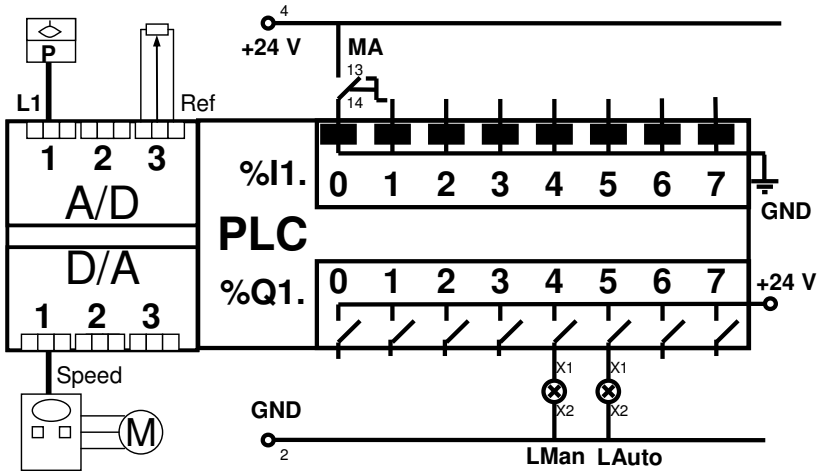
- Un sensor de nivel analógico (transmisor de presión) **L1,** para medir el porcentaje de nivel de líquido en el tanque.
- Una bomba que abastece de agua al tanque movida por un motor de velocidad variable (***Speed***).

**PANEL DE CONTROL:**

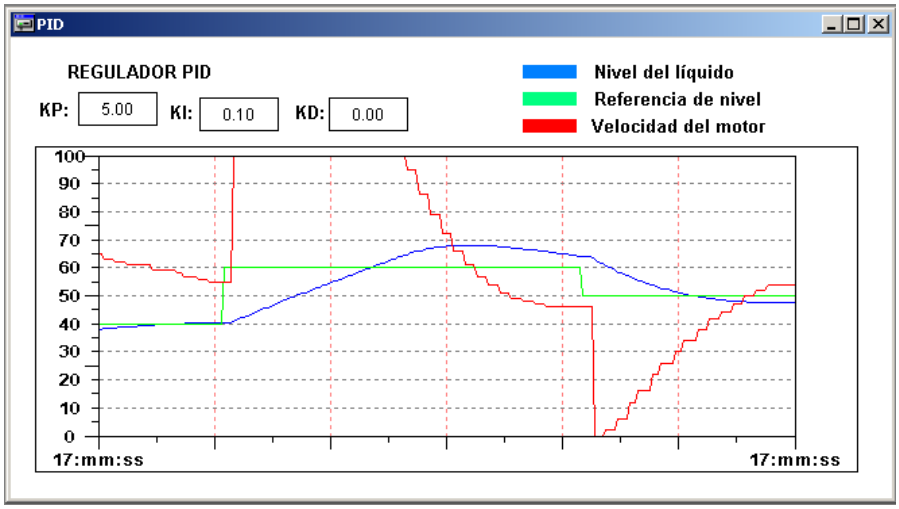
Dispone de los siguientes elementos agrupados en un panel de mando:

- Un interruptor ***M/A*** y dos lámparas ***LMAN*** y ***LAUTO***, que permiten reconocer el modo Manual o Automático de la instalación.
- Un control deslizante que en modo Manual permite controlar la velocidad del motor de la bomba (***Speed***, 0.0 -> 100.0 %) y en modo Automático indica el nivel deseado de líquido en el depósito (***Ref. Nivel***, 0.0 -> 100.0 %).

La figura siguiente ilustra un ejemplo de conexión de los sensores y actuadores a los módulos E/S del PLC. La salida analógica asociada a la consigna de velocidad para el motor (Speed, 0.0->100.0%) es un entero equivalente a 10 veces la consigna de velocidad, es decir, es un INT (16 bits) **Q_iSPEED** que está en el rango [Min=0,Max=1000].




**PANTALLA GRÁFICA DEL REGULADOR PID:**
Se dispone además de una pantalla en la que se representan gráficamente las variables relacionadas con el Regulador PID y desde la que se pueden modificar las constantes del regulador.




**DESCRIPCIÓN DE LAS OPERACIONES DEL SISTEMA:**

**Modo Manual**
La lámpara LMAN debe estar encendida. El operador puede controlar la velocidad del motor de la bomba con el control deslizante del Panel de Control.

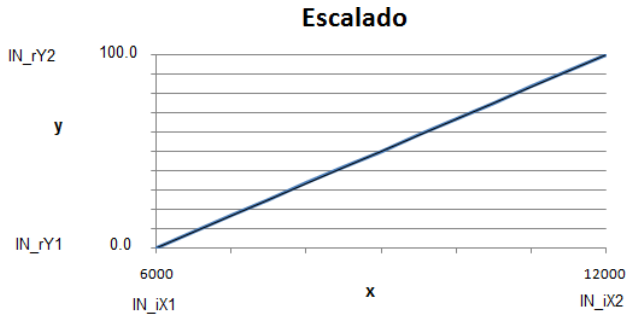
**Modo Automático**
La lámpara LAUTO debe estar encendida. El regulador PID del sistema deberá estabilizar el nivel del líquido en el tanque en el valor indicado por el control deslizante del Panel de Control actuando adecuadamente sobre la velocidad del motor que mueve la bomba.

Con el interruptor M/A el sistema cambia entre modo Manual y modo Automático.

**TAREAS A REALIZAR:**

1. Escribir una función que devuelva un valor real entre 0 y 100, correspondiente al valor indicado por el control deslizante del Panel de Control. El módulo de entradas analógico, conectado al control deslizante, proporciona un nº entero INT (16 bits) **I_iREFNIV** que está en el rango [Min=0,Max=1000]. La función ha de devolver un valor real en el rango [Min=0.0, Max=100.0].

2. Escribir la función FU_rLEENIVEL en lenguaje ST, que devuelva un valor real correspondiente al porcentaje de líquido en el tanque: 0% vacío, 100% lleno. El módulo de entradas analógico, conectado al sensor de nivel, proporciona un valor WORD (16 bits) **I_wSENNIV**. El valor entero correspondiente está en el rango [Min=6000,Max=12000]. La función ha de devolver un valor real en el rango [Min=0.0,Max=100.0].




3. Escribir un programa en que lleve a cabo las operaciones indicadas, controlando adecuadamente el "Gestor de Tareas" ("Freewheeling" y "Cíclicas"):
   - Tratar los datos de entrada (determinación del porcentaje de nivel y de la referencia).
   - Determinar el modo de funcionamiento del sistema (Manual o Automático).
   - Operaciones según el modo de funcionamiento del sistema.
   - Gestionar las variables de salida.

**OBSEVACIONES:**

1. No se deben realizar modificaciones en el programa de simulación proporcionado, en las pantallas de visualización o en las variables ya creadas.

2. Las nuevas variables que se creen (tanto globales como locales) para la realización del programa, deberán seguir el convenio de nomenclatura que permite conocer su tipo (x: boleanas, i: enteros, r: reales, etc.).

# Declaración de variables locales:

```
PROGRAM CONTROL_NIVEL
VAR
   rTM:REAL:=1.0;        (* Tiempo de muestreo en segundos *)
                         (* Debe coincidir con el "tiempo de ciclo" del programa *)
   rREF_NIVEL: REAL;     (* REFERENCIA de nivel *)
   rERROR:REAL:=0.0;     (* ERROR en este ciclo *)
   rERROR_OLD:REAL:=0.0; (* ERROR en el ciclo anterior *)
   rINTEGRAL:REAL:=0.0;  (* INTEGRAL del error *)
   rDERIVADA:REAL:=0.0;  (* DERIVADA del error *)
   rSPEED:REAL:=0.0;     (* CONSIGNA de velocidad para el motor de la bomba *)
END_VAR
```

# Versión básica del programa:

```
(* Programa ciclico que calcula cada 1 segundos el algoritmo del PID *)
rNIVEL:=FU_rLEENIVEL(I_wSENNIV);    (* Se actualiza la variable global rNIVEL *)
rREF_NIVEL:=FU_rLEEREF(I_iREFNIV);  (* Lectura del valor de REFERENCIA *)

rERROR:= rREF_NIVEL-rNIVEL;               (* Cálculo del error *)
rINTEGRAL:=rINTEGRAL+rERROR_OLD*rTM;      (* Cálculo de la integral del error *)
rDERIVADA:=(rERROR-rERROR_OLD)/rTM;       (* Cáculo de la derivada del error *)

rSPEED:=rKP*(rERROR+rKI*rINTEGRAL+rKD*rDERIVADA);  (* Consigna de velocidad *)

rERROR_OLD:=rERROR;     (* Se guarda el valor del error para el próximo ciclo *)
```

# Versión mejorada del programa:

```
(* Programa ciclico que calcula cada 1 segundos el algoritmo del PID *)
rNIVEL:=FU_rLEENIVEL(I_wSENNIV);    (* Se actualiza la variable global rNIVEL *)
rREF_NIVEL:=FU_rLEEREF(I_iREFNIV);  (* Lectura del valor de REFERENCIA *)
rERROR:= rREF_NIVEL-rNIVEL;         (* Cálculo del error *)
IF xPID_ACTIVO THEN     (* Se dan las condiciones para el funcionamietno del PID *)
   IF rKI>0.0 THEN
     rINTEGRAL:=rINTEGRAL+rERROR_OLD*rTM; (* Cálculo de la integral del error *)
   ELSE
     rINTEGRAL:=0.0;     (* Evita que la integral del error crezca cuando no se
                          usa la acción integral *)
   END_IF;
   rDERIVADA:=(rERROR-rERROR_OLD)/rTM;     (* Cáculo de la derivada del error *)
   rSPEED:=rKP*(rERROR+rKI*rINTEGRAL+rKD*rDERIVADA);  (* Consigna de velocidad *)
   IF rSPEED > 100.0 THEN rSPEED:=100.0; END_IF;(* Límites de velocidad 0->100 % *)
   IF rSPEED < 0.0 THEN rSPEED:=0.0; END_IF;
   Q_iSPEED:=10*REAL_TO_INT(rSPEED); (* Se envía la consigna a la salida *)
ELSE
   rINTEGRAL:=0.0; (* Asegura que la integral vuelve a cero si el PID no activo *)
   rSPEED:=0.0;
END_IF;

rERROR_OLD:=rERROR;     (* Se guarda el valor del error para el próximo ciclo *)
```