



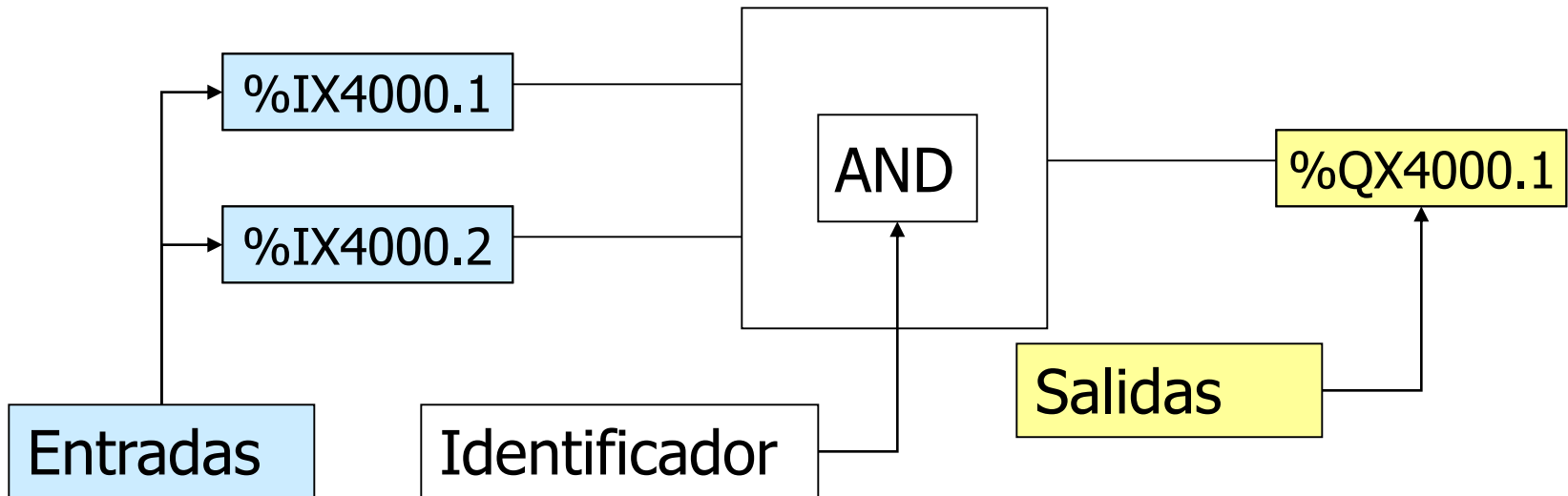
PLC

LENGUAJE FBD

(Function Block Diagram)

El bloque funcional

Bloque funcional: rectángulo con entradas, salidas e identificador



Los tipos de datos dependen del bloque



Estructura de programa FBD

Programa FBD: consiste en conexiones entre bloques funcionales y datos, por líneas de conexión

El programa se ejecuta de arriba abajo, y de izquierda a derecha

Se permite conectar la salida de un bloque a la entrada de otro



Bloques Funcionales

- Se definen sobre la base de “templates” de bloques
- Template de bloque: programa que define el bloque
- Para utilizar un template de bloque, la tarea declara una instancia del template
- Bibliotecas propias de usuario (re-uso)



Funciones

- Aceptan múltiples parámetros de entrada
- Salida única



Función vs Bloque Funcional

- Diferencia entre bloque funcional y función:
 - Número de salidas:
 - Función permite sólo una salida
 - Bloque funcional permite más de una
 - Variables persistentes (conservan valor entre ejecuciones):
 - Función no
 - Bloque funcional sí



Grupos de instrucciones

Clasificación:

Funciones binarias

Entradas y salidas de tipo binario: AND, OR, XOR

Número de entradas variable; posibilidad de negación de entradas

Funciones de timers y contadores

Detalladas en la clase sobre lenguaje LD

Funciones de palabras/reales

Funciones de comparación: el resultado es un bit

Funciones aritméticas: el resultado es una palabra/real (+, -, *, /)

Funciones lógicas bit a bit: AND, OR, XOR, Shift, Rotate

Funciones de control de programa

JUMP; lectura o escritura inmediata de E/S

Funciones de control

PI, PID, etc. *

Funciones de comunicación

Comunicación por protocolos, por ejemplo MODBUS

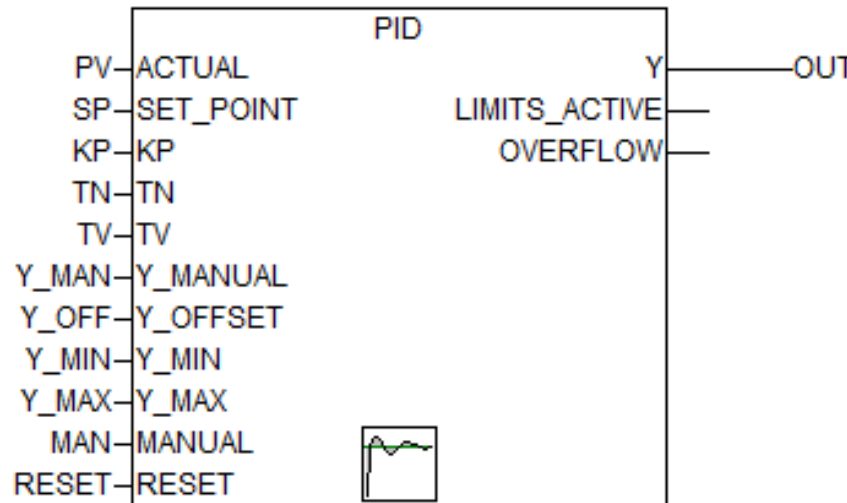
Funciones de conversión de formato

PACK y UNPACK

El 90% de los programas se resuelve con el 20% de las instrucciones

Grupos de instrucciones

- Funciones de control: PI, PID, etc.



$$OUT = Y = KP \left(e + \frac{1}{TN} \int e \cdot dt + TV \frac{\partial e}{\partial t} \right) + Y_OFFSET; e = SP - PV$$

$$PID(s) = KP \left(1 + \frac{1}{TN \cdot s} + TV \cdot s \right)$$



LD o FBD

Algunas diferencias entre un programa FBD y un programa LD:

La implementación de las funciones lógicas de bits

La concatenación de bloques funcionales:

Permitida en FBD

No permitida en LD



Ejemplo Bomba

Se desea escribir un programa que controle el encendido - apagado de una bomba.

La bomba será encendida si:

- 1) Se pulsa el botón de arranque.
- 2) La protección térmica está deshabilitada.
- 3) Está abierto el botón de alarma.
- 4) Está abierto el botón de parada.

Desde un tiempo T después del encendido, no puede haber ni sobre corriente ni baja corriente. Expresado de otra forma, desde un tiempo T después del arranque, la corriente I debe cumplir $I_{\text{MIN}} < I < I_{\text{MAX}}$, siendo I_{MIN} e I_{MAX} límites prefijados.

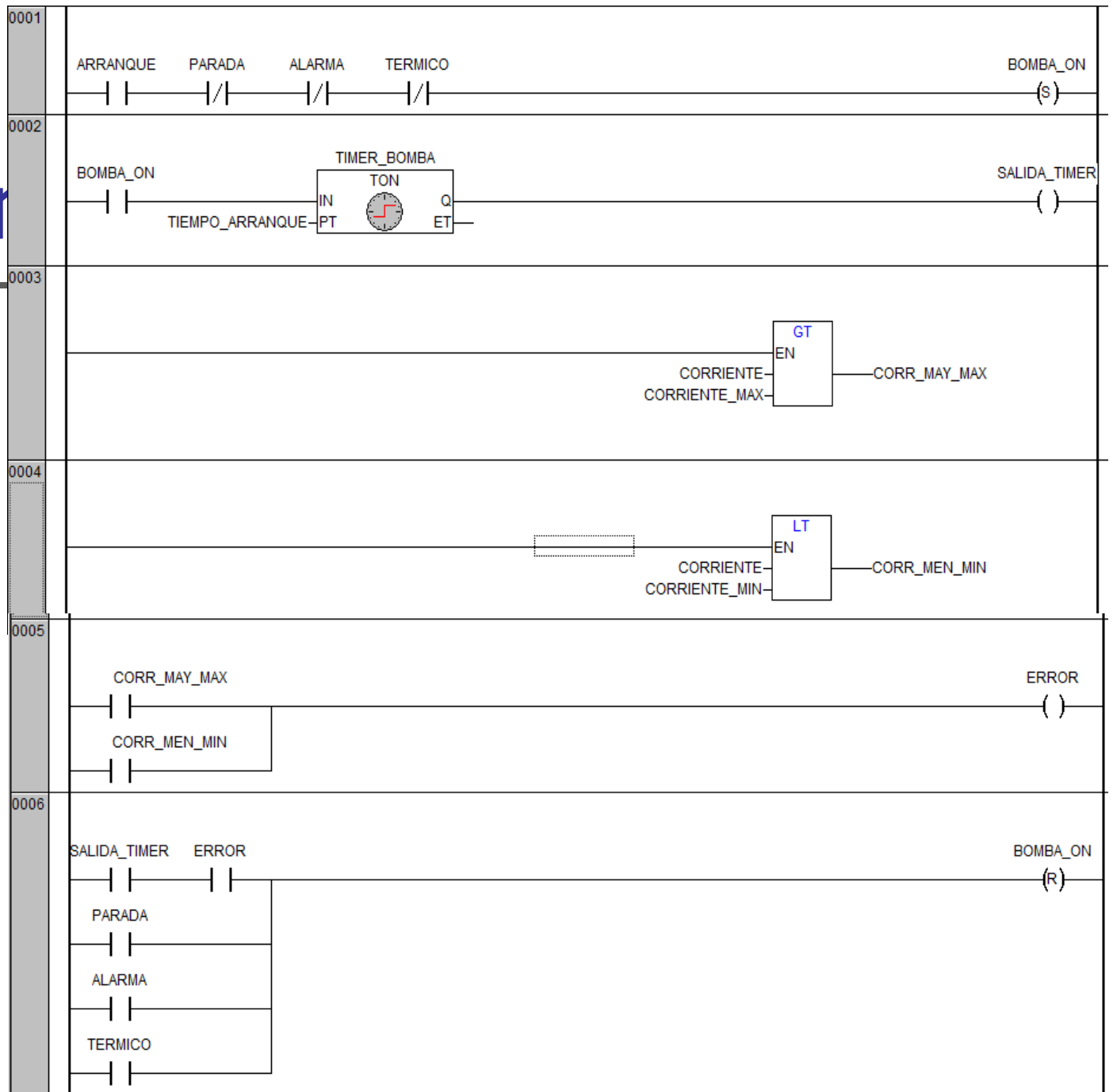


Ejemplo Bomba

El motor de la bomba se apagará si:

- 1) Se pulsa el botón de parada.
- 2) Se cierra la protección térmica.
- 3) Se pulsa el botón de alarma.
- 4) Los límites de corriente no son los correctos.

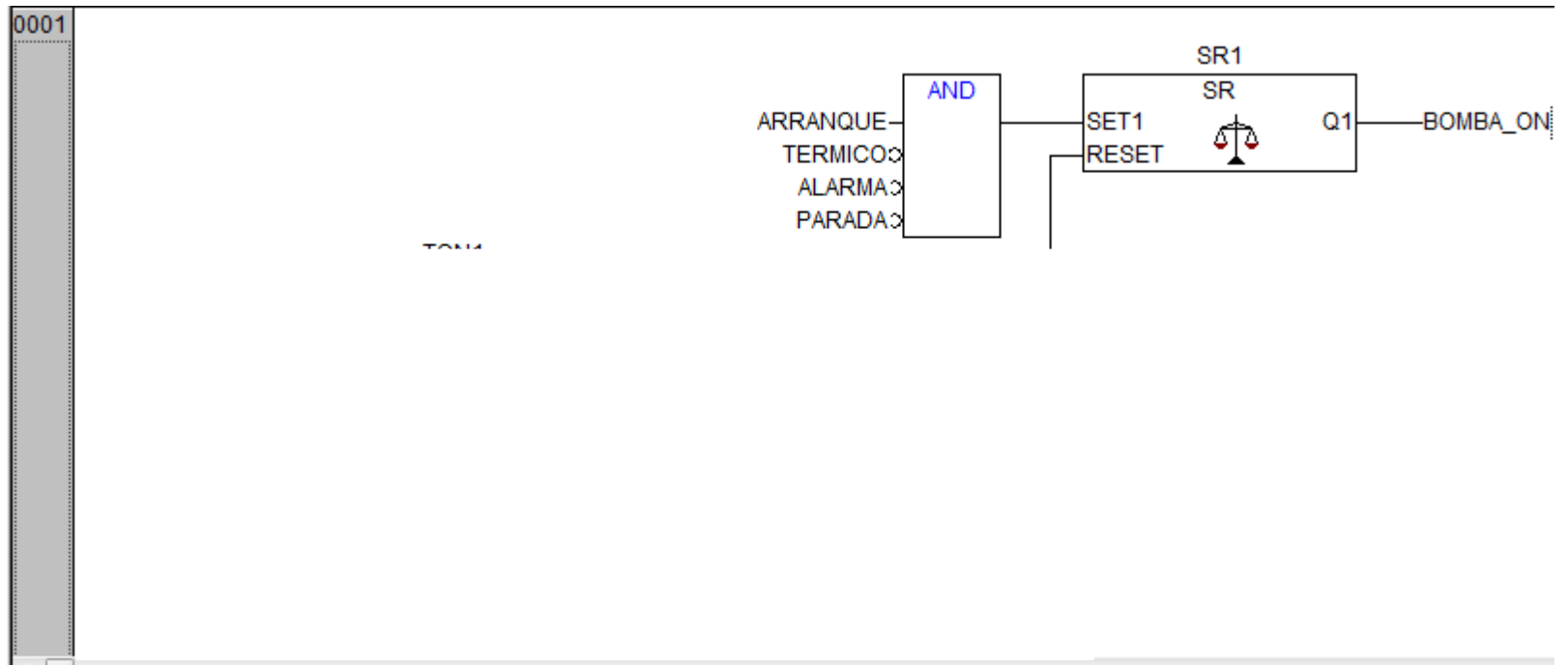
Ejer



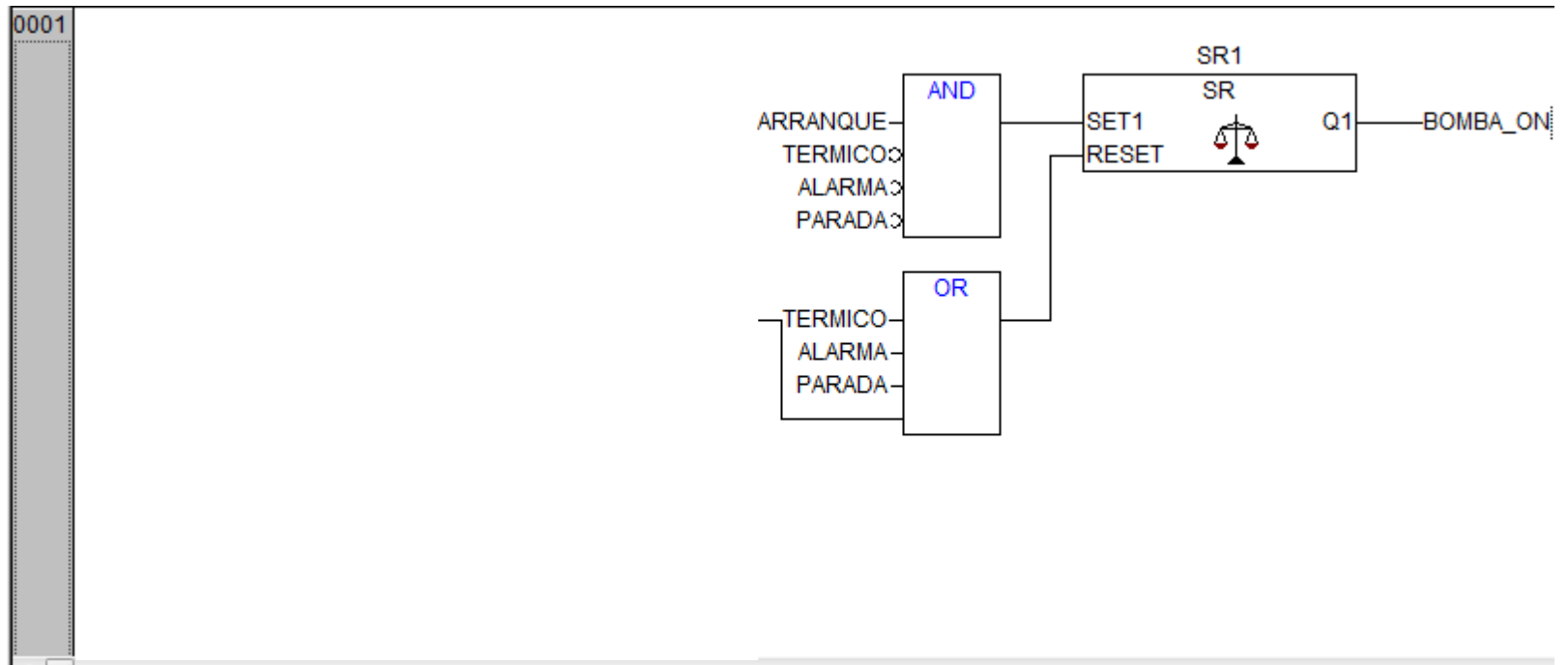
Ejemplo Bomba en FBD



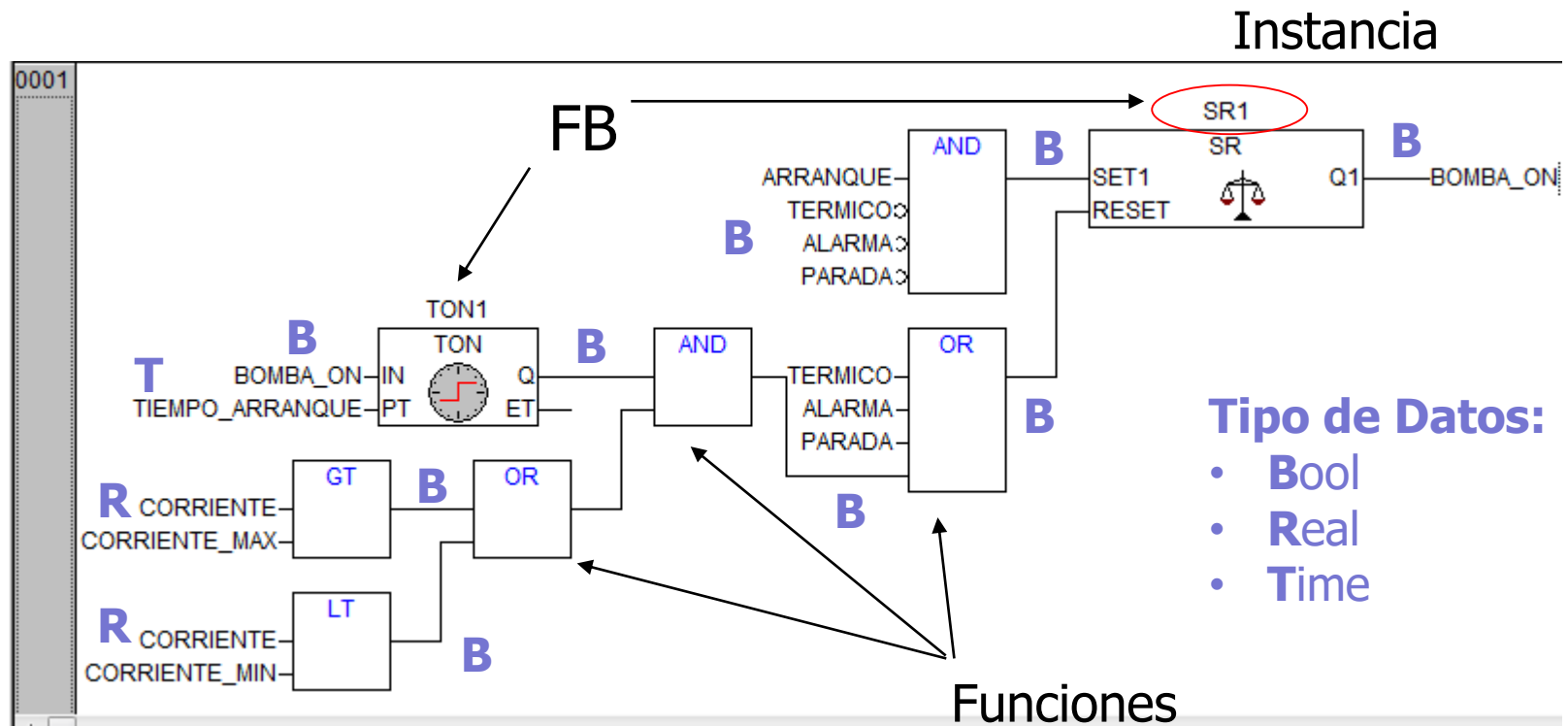
Ejemplo Bomba en FBD



Ejemplo Bomba en FBD



Ejemplo Bomba en FBD



Ambiente en FBD

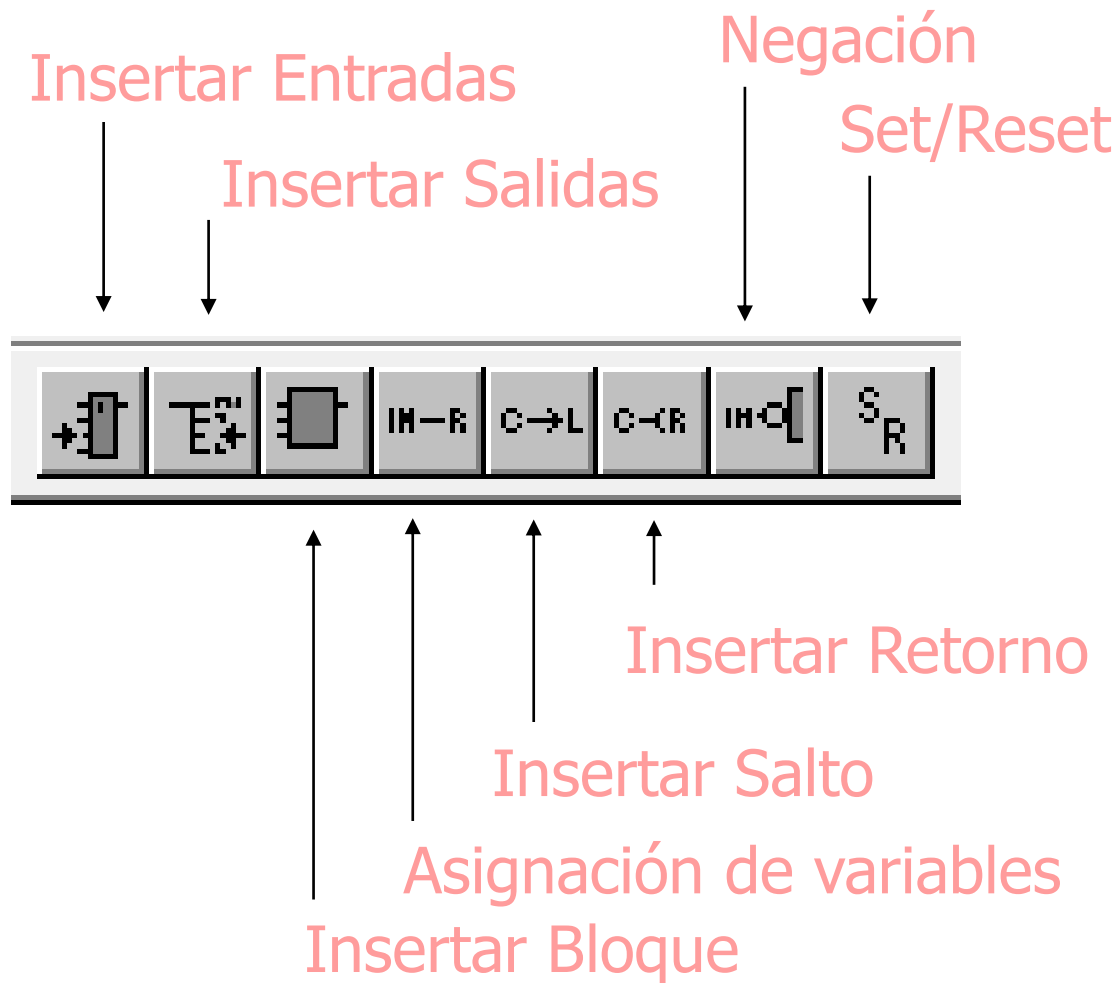
The screenshot displays a software interface for PLC programming. The top menu bar includes File, Edit, Project, Insert, Extras, Online, Window, and Help. Below the menu is a toolbar with various icons for file operations and editing. A red circle highlights a specific section of the toolbar containing icons for undo, redo, and other editing functions. The main workspace is divided into two panes. The left pane, titled 'POUs', shows a project tree with 'PLC_PRG (PRG)' and 'TestFBD (PRG)' listed. The right pane shows the ladder logic editor for 'TestFBD'. The code in the editor is as follows:

```
0001 PROGRAM TestFBD
0002 VAR
0003 END_VAR
0004
0005
```

Below the code, there is a ladder logic diagram with a single step labeled '0001' containing a coil with '???' inside. At the bottom of the interface, there is a status bar with tabs for 'POUs', 'Data types', 'Visualizations', and 'Resources'. A message box at the bottom right displays the following text:

```
Loading library 'C:\Program Files (x86)\Common Files\CAA-Targets\VAB_B_AC500\AC500_V12\Library\SysInt_AC500_V10.lib'
Loading library 'C:\Program Files (x86)\Common Files\CAA-Targets\VAB_B_AC500\AC500_V12\Library\SysExt_AC500_V10.lib'
Loading library 'C:\Program Files (x86)\Common Files\CAA-Targets\VAB_B_AC500\AC500_V12\Library\OnBoardIO_AC500_V13.lib'
```

Herramientas FBD



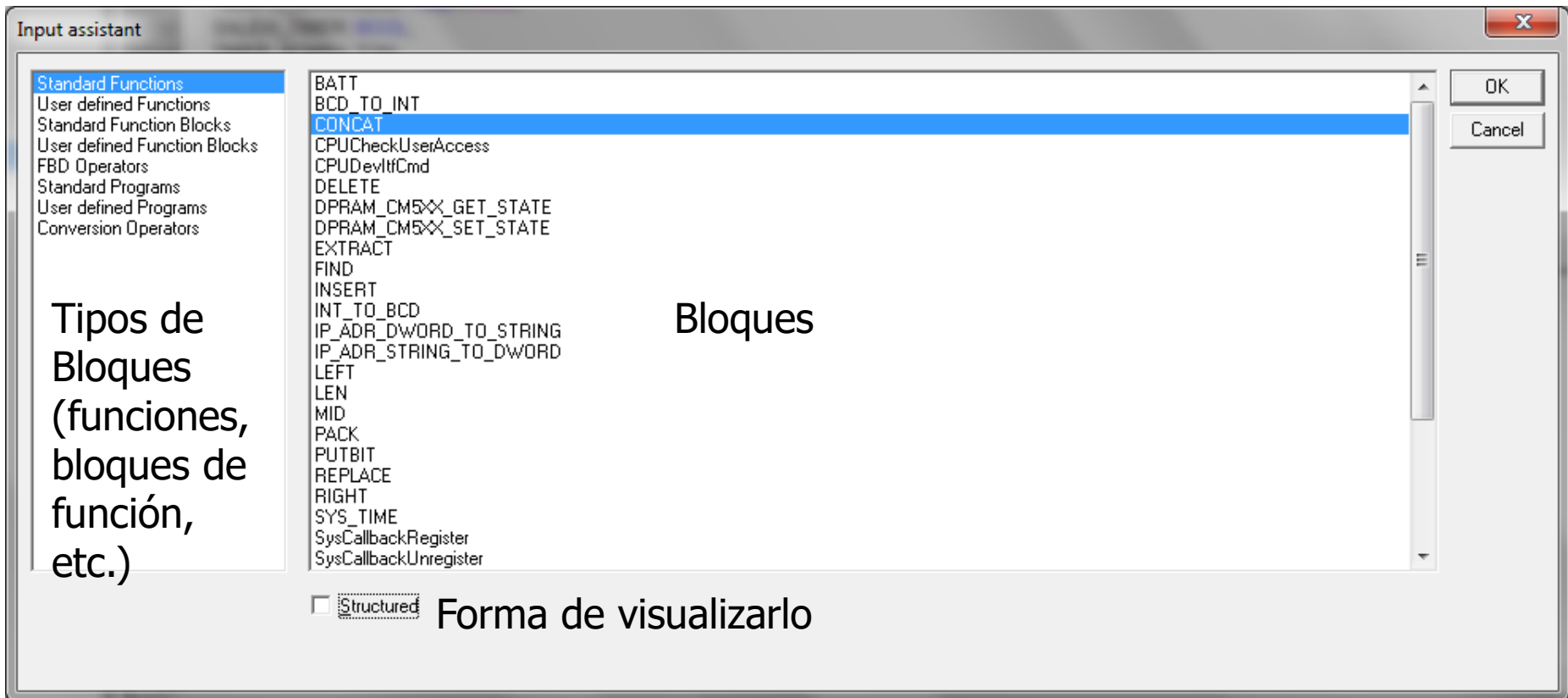
Ambiente en FBD

The screenshot displays a software interface for PLC programming. The top menu bar includes 'File', 'Edit', 'Project', 'Insert', 'Extras', 'Online', 'Window', and 'Help'. Below the menu is a toolbar with various icons for file operations and editing, along with a '100%' zoom level indicator. The main workspace is divided into two panes. The left pane, titled 'POUs', shows a project tree with 'PLC_PRG (PRG)' and 'TestFBD (PRG)' listed. The right pane shows the ladder logic editor for 'TestFBD (PRG)'. The editor displays the following code:

```
0001 PROGRAM TestFBD
0002 VAR
0003 END_VAR
0004
0005
```

Below the code, a ladder logic diagram is shown for step 0001. It features a blue 'AND' gate with two red '???' labels on its inputs. To the right of the diagram, the text 'Identificador Asistente -> F2' is displayed.

Bibliotecas de Bloques



Library Manager

- Menú "Window" -> "Library Manager"

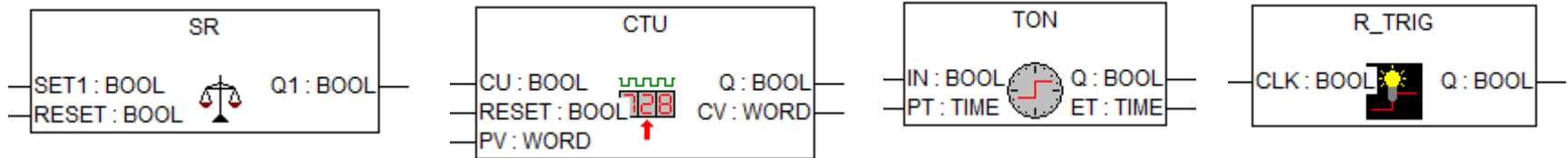
The screenshot displays the Library Manager interface with three main sections:

- Bibliotecas:** A list of system libraries including BusDiag.lib, SysInt_AC500_V10.lib, SysExt_AC500_V10.lib, OnBoardIO_AC500_V13.lib, standard.lib (highlighted), Iecsf.lib, Util.lib, SysLibTime.lib, SysTaskInfo.lib, SysLibMem.lib, SysLibInitLibrary.lib, SYSLIBCALLBACK.LIB, and Ethernet_AC500_V10.lib.
- Bloques:** A tree view of function blocks under POU's, including Bistable Function Blocks (RS (FB), SEMA (FB), SR (FB) - highlighted), Counter (CTD (FB), CTU (FB), CTUD (FB)), and String Functions (CONCAT (FUN)).
- Declaración:** The declaration code for the SR block:

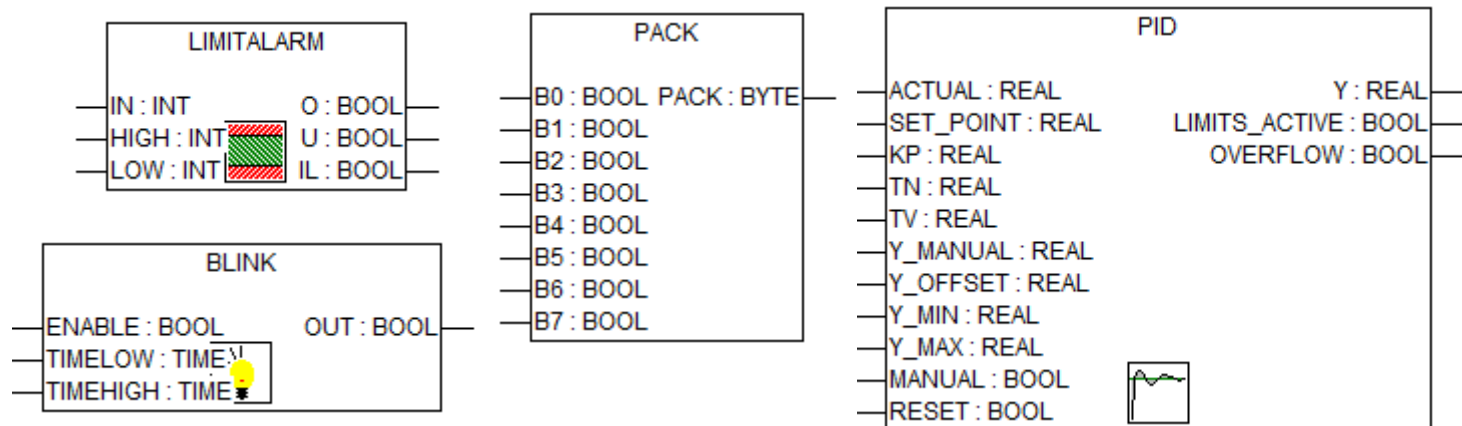
```
FUNCTION_BLOCK SR
(*
  Bistable function, set dominant
  Q1 = SET1 OR (NOT RESET AND Q1)
*)
VAR_INPUT
  SET1: BOOL;
  RESET: BOOL;
END_VAR
VAR_OUTPUT
  Q1: BOOL;
END_VAR
```
- Representación:** A graphical representation of the SR block as a rectangle with a balance scale icon. Inputs are SET1: BOOL and RESET: BOOL. The output is Q1: BOOL.

Library Manager

■ Standard.lib:



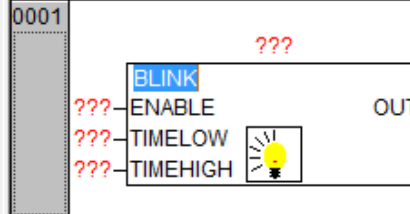
■ Util.lib:



Ayuda para Bloques – F1

POUs
PLC_PRG (PRG)
TestFBD (PRG)

```
0001 PROGRAM TestFBD
0002 VAR
0003 END_VAR
0004
0005
```



Identificador
F1 -> Ayuda

IMPORTANTE

BLINK

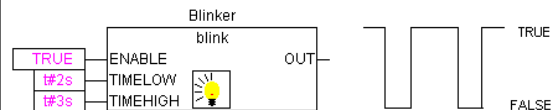
Provided by [util.lib](#).

The function block BLINK generates a pulsating signal. The input consists of ENABLE of the type BOOL, as well as TIMELOW and TIMEHIGH of the type TIME. The output OUT is of the type BOOL.

If ENABLE is set to TRUE, BLINK begins to set the output for the time period TIMEHIGH to TRUE and afterwards to set it for the time period TIMELOW to FALSE.

When ENABLE is reset to FALSE, output OUT will not be changed, i.e. no further pulse will be generated. If you explicitly also want to get OUT FALSE when ENABLE is reset to FALSE, you might use "OUT AND ENABLE" (i.e. adding an AND box with parameter ENABLE) at the output.

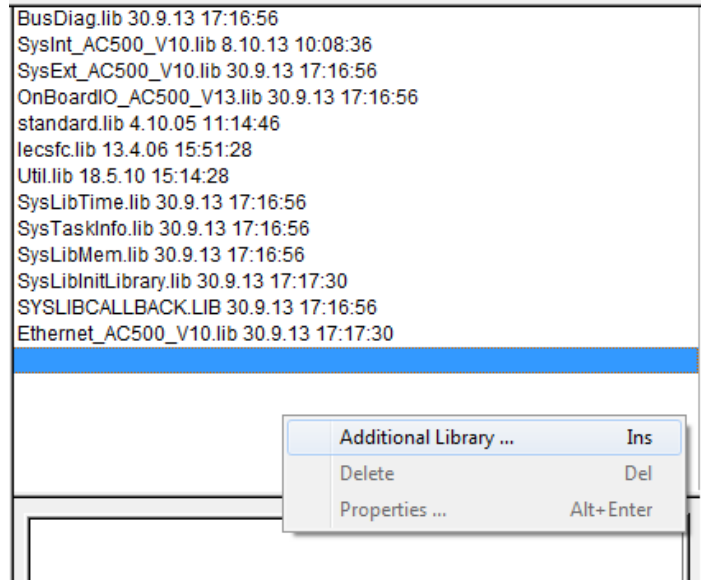
Example in CFC:





Library Manager

- Agregar bibliotecas:



Bloques de Usuario

Nuevo "Function Block"

The screenshot shows a software interface for creating a new function block. On the left, a project tree under 'POUs' shows 'Mis Bloques' containing 'Bomba (FB)', 'PLC_PRG (PRG)', and 'ProgFBD (PRG)'. An arrow points to 'Bomba (FB)' with the text 'Nuevo "Function Block"'. On the right, a table lists the block's structure:

0001	FUNCTION_BLOCK	Bomba
0002	VAR_INPUT	
0003		ARRANQUE: BOOL;
0004		TERMICO: BOOL;
0005		PARADA: BOOL;
0006	END_VAR	
0007	VAR_OUTPUT	
0008		BOMBA_ON:BOOL;
0009	END_VAR	
0010	VAR	
0011		SR1: SR;
0012	END_VAR	
0013		

Below the table, a ladder logic diagram labeled 'Lógica interna' shows the internal logic. It features an AND gate with inputs 'ARRANQUE', 'TERMICO', and 'PARADA'. An OR gate has inputs 'TERMICO' and 'PARADA'. The outputs of both gates connect to the 'SET1' input of an SR flip-flop (SR1). The flip-flop's 'RESET' input is also connected to the OR gate. The flip-flop's output 'Q1' is connected to the 'BOMBA_ON' output variable.

Bloques de Usuario

The screenshot displays a PLC programming environment. On the left, a project tree shows a folder named 'POUs' containing 'Mis Bloques', which includes 'Bomba (FB)', 'PLC_PRG (PRG)', and 'ProgFBD (PRG)'. The 'ProgFBD (PRG)' is selected. The main editor area shows a ladder logic program with the following code:

```
0001 PROGRAM ProgFBD
0002 VAR
0003   BBA1_ARR: BOOL;
0004   BBA1_TER: BOOL;
0005   BBA1_PAR: BOOL;
0006   BBA1_ON: BOOL;
0007 END_VAR
0008
0009
```

An 'Input assistant' dialog box is open, showing a list of function blocks. The 'User defined Function Blocks' category is selected, and 'Bomba (FB)' is highlighted. The 'Structured' checkbox is checked. The dialog has 'OK' and 'Cancel' buttons.

At the bottom of the software window, the status bar indicates: 1 Error(s), 0 Warning(s).

Bloques de Usuario

The screenshot displays a software interface for PLC programming. On the left, a project tree shows the following structure:

- POUs
 - Mis Bloques
 - Bomba (FB)
 - PLC_PRG (PRG)
 - ProgFBD (PRG)**

The main editor area shows the following code:

```
0001 PROGRAM ProgFBD
0002 VAR
0003   BBA1_ARR: BOOL;
0004   BBA1_TER: BOOL;
0005   BBA1_PAR: BOOL;
0006   BBA1_ON: BOOL;
0007   BBA1: Bomba;
0008 END_VAR
```

The code is annotated with the word "Declaración" (Declaration) in large text.

Below the code, an instance diagram is shown, labeled "Instancia" (Instance). It features a box representing the "Bomba" block, with the instance name "BBA1" above it. The connections are as follows:

- BBA1_ARR is connected to the "ARRANQUE" input of the "Bomba" block.
- BBA1_TER is connected to the "TERMICO" input of the "Bomba" block.
- BBA1_PAR is connected to the "PARADA" input of the "Bomba" block.
- The "BOMBA_ON" output of the "Bomba" block is connected to the "BBA1_ON" variable.

Bloques de Usuario

¿Agregar un entrada más a cada bomba?

The screenshot displays a software interface for a PLC project. On the left, a project tree shows a folder named 'POUs' containing a sub-folder 'Mis Bloques'. Inside 'Mis Bloques', there are three items: 'Bomba (FB)', 'PLC_PRG (PRG)', and 'ProgFBD (PRG)'. The 'ProgFBD (PRG)' item is selected and highlighted in blue.

On the right, the main editor shows the ladder logic for the 'ProgFBD' program. The code is as follows:

```
0001 PROGRAM ProgFBD
0002 VAR
0003   BBA1_ARR: BOOL;
0004   BBA1_TER: BOOL;
0005   BBA1_PAR: BOOL;
0006   BBA1_ON: BOOL;
0007   BBA1: Bomba;
0008   BBA2: Bomba;
0009   BBA2_ARR: BOOL;
0010   BBA2_TER: BOOL;
0011   BBA2_PAR: BOOL;
0012   BBA2_ON: BOOL;
0013 END_VAR
```

Below the code, the ladder logic is visualized. It consists of two rungs, 0001 and 0002. Rung 0001 contains a function block named 'BBA1' (Bomba). The inputs are 'BBA1_ARR' (ARRANQUE BOMBA_ON), 'BBA1_TER' (TERMICO), and 'BBA1_PAR' (PARADA). The output is 'BBA1_ON'. Rung 0002 contains a function block named 'BBA2' (Bomba). The inputs are 'BBA2_ARR' (ARRANQUE BOMBA_ON), 'BBA2_TER' (TERMICO), and 'BBA2_PAR' (PARADA). The output is 'BBA2_ON', which is shown as a dashed box, indicating it is not yet connected to a coil or other logic.

Bloques de Usuario

The screenshot displays a software interface for a PLC project. On the left, a tree view shows the project structure under 'POUs', with 'Mis Bloques' containing 'Bomba (FB)', 'PLC_PRG (PRG)', and 'ProgFBD (PRG)'. The 'Bomba (FB)' block is selected. The main area shows the declaration of the function block:

```
0001 FUNCTION_BLOCK Bomba
0002 VAR_INPUT
0003     ARRANQUE: BOOL;
0004     TERMICO:  BOOL;
0005     PARADA:   BOOL;
0006 END_VAR
0007 VAR_OUTPUT
0008     BOMBA_ON:BOOL;
0009 END_VAR
0010 VAR
0011     SR1: SR;
0012 END_VAR
0013
```

Below the declaration, the ladder logic for the function block is shown. It consists of two parallel paths leading to an SR (Set-Reset) flip-flop:

- The top path uses an AND gate with inputs 'ARRANQUE', 'TERMICO', and 'PARADA'. Its output is connected to the 'SET1' input of the SR block.
- The bottom path uses an OR gate with inputs 'TERMICO' and 'PARADA'. Its output is connected to the 'RESET' input of the SR block.

The SR block has a 'Q1' output connected to a coil labeled 'BOMBA_ON'.

Agregar un
entrada más a
cada bomba



Uso de bloques de funciones

- Re-uso de soluciones probadas
- Fácil modificación general
- Cajas negras con comportamiento conocido
- Simplifica el programa general

Elemento clave de la norma IEC 61131 para una alta calidad de software