APIMOU

Versión 2.6

Manual del usuario

ADVERTENCIA – EXENCIÓN DE RESPONSABILIDAD

Este software se proporciona “tal cual”, sin garantía de ningún tipo, expresa o implícita, incluyendo, entre otras, las garantías de comerciabilidad, adecuación a un propósito particular o ausencia de errores.

En ningún caso el autor será responsable de ningún daño, pérdida de datos, fallo de hardware u otros efectos adversos que resulten directa o indirectamente del uso de este software, incluso si se ha advertido al autor de la posibilidad de tales daños.

El uso de este software es bajo su propia responsabilidad.

Índice

1 Presentación ... 3
2 Información sobre licencias ... 3
3 Rápida puesta en marcha ... 4
3.1 Ejemplo de programa en lógica combinatoria ... 4
3.2 Ejemplo de programa en lógica secuencial ... 4
4 Variables ... 7
4.1 Cambio mnemonico ... 7
4.2 Variables de entrada ... 7
4.3 Variables de salida ... 7
4.4 Variables de etapa ... 8
4.5 Bits internos ... 8
4.6 Variables de temporizadores ... 8
4.7 Variables de contadores ... 9
5 Ecuaciones lógicas ... 9
6 Temporizadores y contadores ... 9
7 Grafcet ... 11
7.1 Etapas y acciones ... 11
7.2 Transiciones y receptividad ... 12
7.3 Uniones orientadas ... 13
7.4 Divergencias y convergencias en “Y” ... 13
7.5 Divergencias y convergencias en “O” ... 14
7.6 Grafcet múltiple ... 14
7.7 Edición multipágina ... 14
7.8 Órdenes de forzado ... 14
7.9 Varias funciones de edición ... 15
8 Funciones especiales ... 17
8.1 Señal de reloj o intermitente ... 17
8.2 Flanco de subida o bajada ... 17
8.3 Bit de inicialización ... 18
9 Funcionamiento en modo simulación ... 19
10 Funcionamiento en modo conectado ... 19
11 Generación de código Arduino ... 20
12 Transferencia directa de un programa a una placa Arduino ... 21
13 Generación de código ESP32 ... 22
13.1 información general ... 22
13.2 Requisitos previos : ... 23
13.3 Modos Station y Access Point ... 23
13.4 Procedimiento ... 24
14 Transferencia directa de un programa a una placa ESP32 ... 26
15 Gestión de archivos ... 27



1 Presentación

Apimou es un simulador de controlador lógico programable (PLC) que permite crear y probar programas sencillos que implementan funciones lógicas combinatorias y secuenciales. Puede funcionar en modo simulación o en modo conectado, conectado a una parte operativa software o real. Dispone de 32 entradas binarias y 32 salidas binarias, así como de variables internas (variables de paso, variables de temporización, variables de recuento, bits internos).

Las salidas pueden ser funciones directas de las entradas, y/o depender de una secuencia materializada por un grafcet.

El archivo Apimou.zip incluye :

-> Apimou.exe, el programa principal (que puede ser autosuficiente en modo simulación),

-> Apimou_es.pdf, este documento,

-> Bees.exe, Afficheur.exe, Feux_tri.exe, Convoyeur&Verins.exe, Ascenseur.exe, Chariots.exe (y quizás otros) que son las partes operativas de software disponibles actualmente. Apimou sólo podrá comunicarse con una parte operativa a la vez,

-> una carpeta Sources que contiene el código fuente (C++Builder 10.3.3) de la parte operativa Bees. Este código se proporciona para entender el protocolo de comunicación entre Apimou y las partes operativas. Es perfectamente posible prescindir de abrir esta carpeta.

-> Una carpeta Fichiers api que contiene, en las subcarpetas correspondientes, algunos programas de ejemplo para las partes operativas propuestas. Las subcarpetas "pour Afficheur", "pour Ascenseur", "pour Feux tricolores", "pour Convoyeur et vérins" contienen un único fichero "config_(nombre del PO).api". Cuando este archivo se abre en Apimou, simplemente encontramos mnemónicos (para entradas/salidas) correspondientes a lo que vemos en la parte operativa correspondiente (etiquetas de botones, sensores, etc.). Queda por crear el programa. En Internet se pueden encontrar ejercicios relativos a estas partes operativas, y Apimou permite probar en la práctica las soluciones de estos ejercicios.

2 Información sobre licencias

Apimou es software libre.

El código fuente de Apimou está disponible en el sitio web del autor, o bajo petición.

Apimou utiliza el componente DOSCommand de TurboPack, licenciado bajo MPL 1.1.

Apimou usa la librería Boost, licenciada bajo Boost Software License 1.0.

Apimou utiliza DelphiZXingQRCode de Foxit Software, licenciado bajo Apache-2.0.

3 Rápida puesta en marcha

Los dos ejemplos siguientes se probarán en modo de simulación.

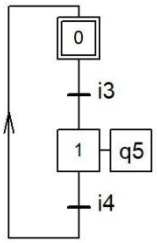
3.1 Ejemplo de programa en lógica combinatoria

1. Lanzamiento de Apimou
2. Compruebe que está seleccionado el modo Simulación (Modo -> Simulación)
3. Poner el PLC en marcha (Autómata -> Run)
4. Haga clic en las casillas %I1.0, %I1.1, etc. (mnemónicos correspondientes i0, i1, etc.): verá que se encienden los leds de entrada en el PLC (y eso es todo: de momento el PLC no está programado)
5. Poner el PLC en Stop (Autómata -> Stop)
6. Haga clic en la pestaña Salidas, y en la columna Ecuación de la salida %Q2.3 (mnemónico q3 escriba i2+i7
7. Poner el PLC en marcha (Autómata -> Run)
8. Haga clic en la pestaña Entradas y pruebe el programa: haga clic en las casillas %I1.0, %I1.1, etc. Puede ver que la salida 3 (dirección %Q2.3, mnemónico q3) se activa (el LED de salida correspondiente se enciende) si hace clic en %I1.2 (mnemónico i2) o %I1.7 (mnemónico i7).

El PLC se programó con un ejemplo puramente combinatorio (q3 = i2 + i7) y se probó en modo simulación.

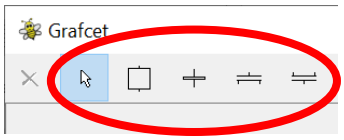
3.2 Ejemplo de programa en lógica secuencial

O para programar el siguiente grafcet: :

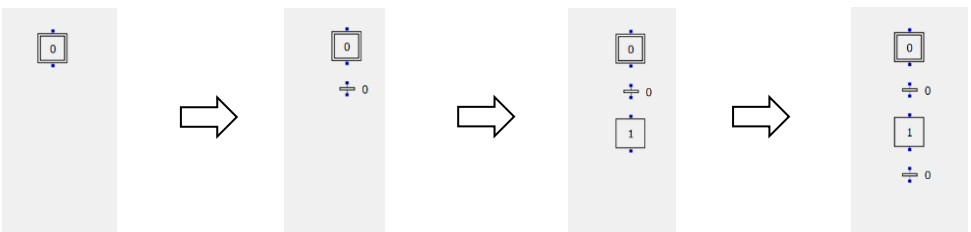


(hecho con Editsab)

1. Autómata -> Stop.
2. Archivo -> Nuevo (borramos todo).
3. Ver -> Grafcet. En la barra de herramientas del formulario que aparece se encuentran los botones correspondientes a los elementos necesarios para la constitución de un grafcet: etapa, transición, divergencia y convergencia en Y.

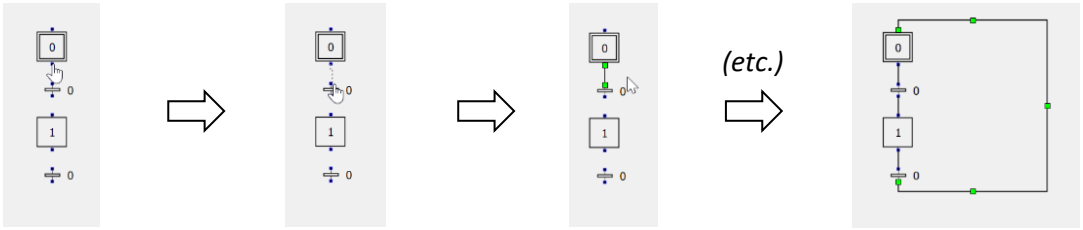


4. Elige una etapa y colócalo en la tarjeta, luego una transición y colócala debajo. Repite hasta que tengas todos los elementos que quieras.
   La primera etapa insertado en el formulario es por defecto una etapa inicial (doble cuadrado). Los números de la etapa se incrementan automáticamente. Las transiciones tienen por defecto receptividades siempre falsas (valor 0).

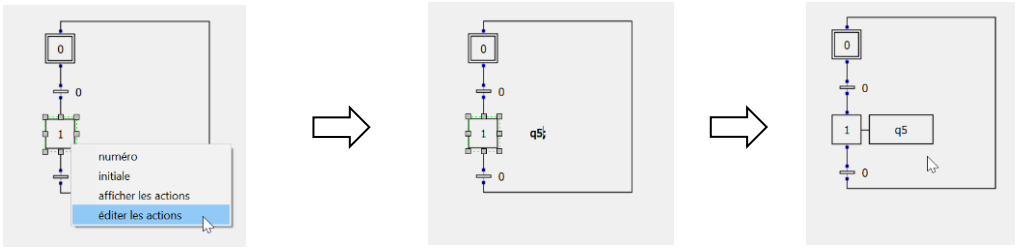


5. Enlaza las etapas con las transiciones y las transiciones con las etapas haciendo clic en un punto de enlace (pequeño cuadrado azul), tirando y soltando en el segundo punto de enlace. Para el

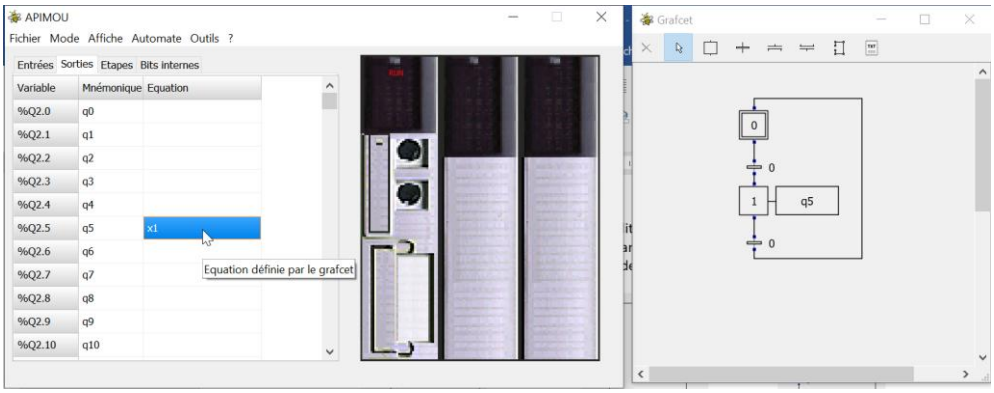
último enlace (última transición a paso inicial), haz clic en el tirador derecho (pequeño cuadrado verde) y tira de él hacia la derecha para dejar espacio a las acciones.



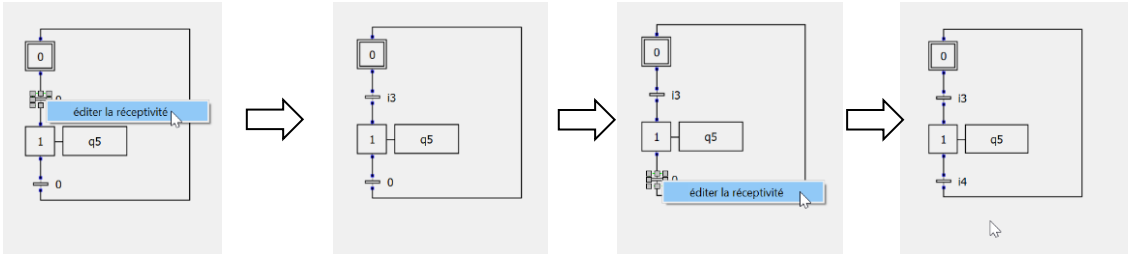
6. Haga clic con el botón derecho en el etapa 1 y elija editar acciones. Introduzca la acción elegida (aquí q5). El punto y coma ";" sirve para separar las acciones. Como aquí sólo hay una, puede borrarla (no es obligatorio). Valide haciendo clic en un espacio libre del formulario o con la combinación de teclas Ctrl+Enter.



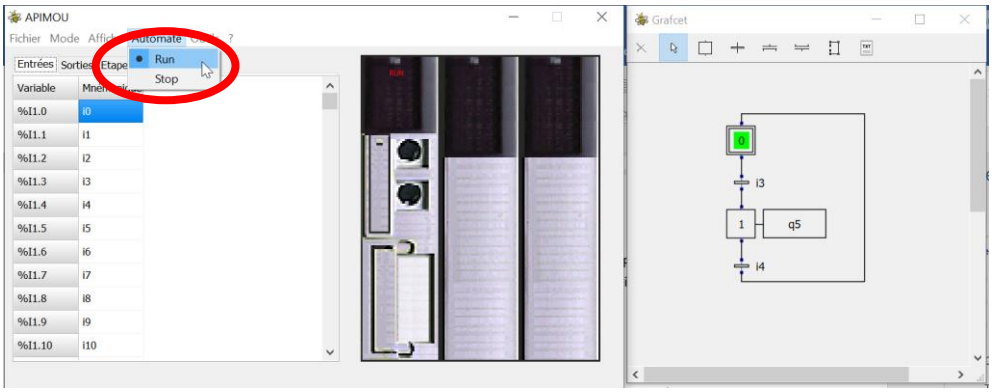
Puede ver en la hoja principal, haciendo clic en la pestaña salidas, que la ecuación de la salida q5 se ha introducido automáticamente (q5 = x1).



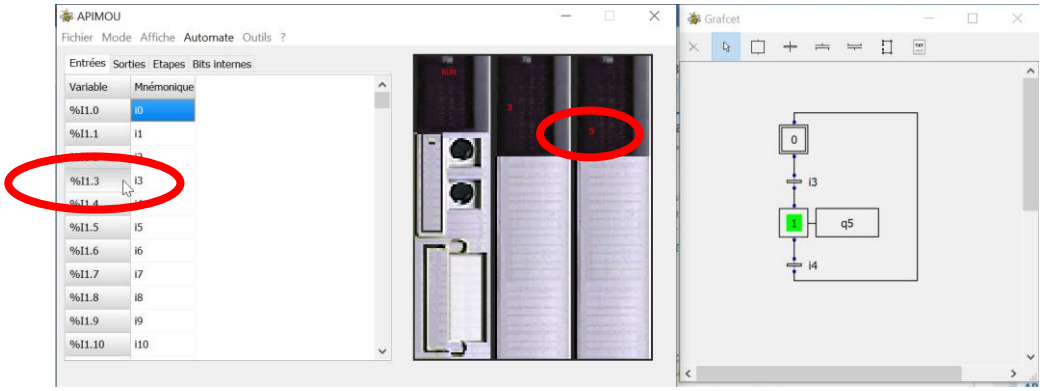
7. Haga clic con el botón derecho en la primera transición y elija editar receptividad. Introduzca la receptividad elegida (aquí i3). Valide haciendo clic en un espacio libre de la tarjeta o con la combinación de teclas Ctrl+Enter. Haga lo mismo para la segunda transición, de receptividad i4. La introducción del grafcet (del programa) ha terminado.



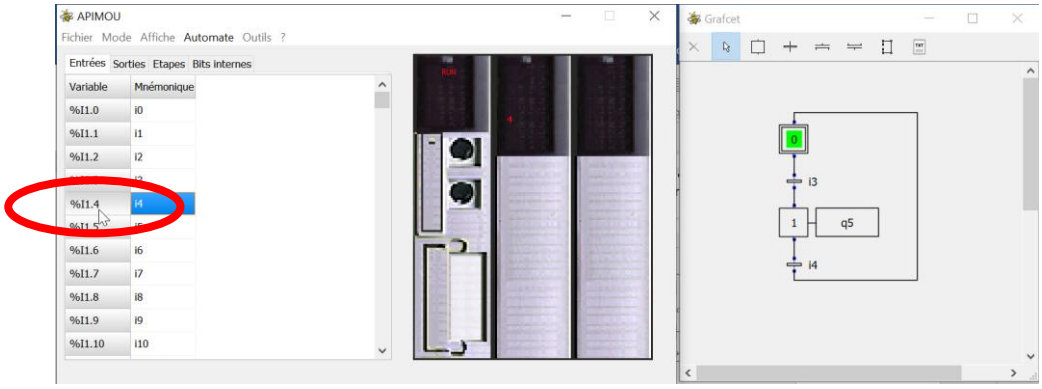
8. Prueba del programa: en el archivo principal, menú Autómata -> Run. Un cuadrado verde (el "token") indica en el grafcet que la etapa 0 (inicial) está activo.



9. Haga clic en la pestaña Entradas y luego en la casilla %I1.3 (mnemónico i3): se ve en el grafcet que la etapa 0 se desactivó y que la etapa 1 está ahora activa. Se efectúa la acción correspondiente, así la salida q5 pasa a 1 (el led de la salida 5 se enciende).



10. Haga clic en la casilla %I1.4 (mnemónico i4): se ve en el grafcet que la etapa 1 se desactivó y que la etapa 0 está ahora activa, se volvió a la etapa inicial. La salida q5 vuelve a 0 (el LED de salida correspondiente se apaga).



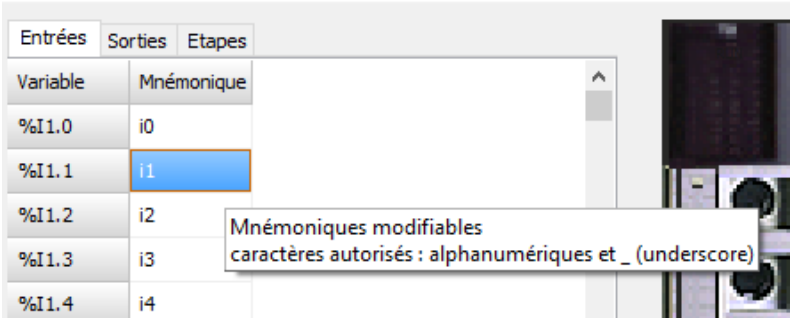
4 Variables

Las variables de entrada, salida, etapa y bits internos se identifican por un par (dirección, mnemónico). Las variables de temporizador y contador se identifican sólo por un mnemónico (t0 a t7 para temporizadores, ct0 a ct7 para contadores).
Las direcciones sólo se utilizan aquí a efectos de identificación, no pueden utilizarse en expresiones lógicas.
Los mnemónicos son esenciales y se utilizan en las expresiones lógicas. Cada mnemónico es único.

4.1 Cambio mnemonico

Los mnemónicos por defecto se pueden cambiar desde la ventana principal, siempre que el PLC esté en STOP.

Los caracteres permitidos para los mnemónicos son todos los caracteres alfanuméricos (sin acento) y el guión bajo (_). Un mnemónico sólo puede constar de un carácter, pero las mnemotecnias 0 y 1 están reservadas (0 corresponde a una condición siempre falsa y 1 a una condición siempre verdadera). Los mnemónicos t0 a t7 y ct0 a ct7 también están reservados (variables de temporización y recuento).
Los mnemónicos distinguen entre mayúsculas y minúsculas: A1 y a1 son dos mnemónicos diferentes.
Para modificar un mnemónico, selecciónelo y haga clic para acceder al modo de edición de la tabla:



4.2 Variables de entrada

Corresponden a los estados de los sensores on-off conectados a la tarjeta de entrada del PLC. Sus valores son leídos por el PLC. En modo simulación, estas variables son forzadas: se les asigna un valor pulsando sobre las casillas %I.x.

4.3 Variables de salida

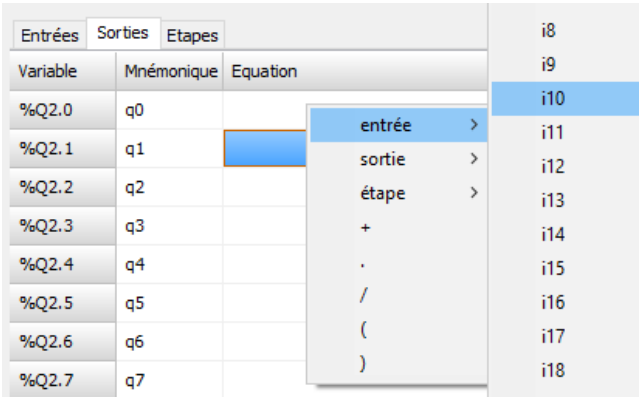
Corresponden a los estados de los (pre, micro) actuadores de conmutación conectados a la tarjeta de salida del PLC.
Sus valores son calculados por el autómata.
Estos cálculos son el resultado de funciones lógicas que se definen aquí, en la pestaña Salidas de la hoja principal, en la columna Ecuación.

Si se ha creado uno (o varios) grafcet, las ecuaciones de las salidas correspondientes a las acciones de este (estos) grafcet se definen automáticamente y no es posible modificarlas. También se puede no programar ninguna acción en el grafcet e introducir manualmente las ecuaciones de las salidas, lo que deja más libertad.

Hay dos formas de introducir una ecuación lógica para una variable de salida:

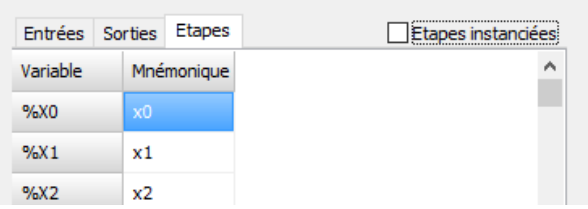
- o seleccionando la celda y haciendo clic en ella para entrar en el modo de edición. La ecuación se introduce entonces mediante el teclado.

- o seleccionando la celda y haciendo clic con el botón derecho del ratón, sin entrar en el modo de edición, para seleccionar las variables y operadores necesarios :



4.4 Variables de etapa

Corresponden a los estados de las etapas del grafcet(s). Son variables internas cuyos valores calcula el autómata.
Por defecto, se pueden ver todas las variables de paso, aunque no se utilicen todas.
Para ver sólo las etapas utilizadas (instanciadas), marque la casilla Etapas instanciadas en la pestaña Etapas de la ventana principal:



4.5 Bits internos

Los bits internos se utilizan para lo que queramos, están definidos, como las variables de salida, por una ecuación lógica.

Por ejemplo, pueden utilizarse para expresar :

- una condición lógica "compleja", que se resumirá así en una variable,
- un flanco de subida o bajada,
- un bit de inicialización.

También se pueden utilizar, si a uno le gustan las dificultades, para programar un grafcet en ecuaciones lógicas.

4.6 Variables de temporizadores

Las variables de temporizadores (t0 a t7) son variables internas que cambian a 1 cuando el temporizador correspondiente (T0 a T7) ha expirado (véase § Temporizadores y contadores). Pueden utilizarse según sea necesario para expresar una salida, un bit interno, una receptividad o cualquier otra expresión lógica.

4.7 Variables de contadores

Las variables de contadores (ct0 a ct7) son variables internas que cambian a 1 cuando se completa el contador correspondiente (CT0 a CT7) (véase § Temporizadores y contadores). Pueden utilizarse según sea necesario para expresar una salida, un bit interno, una receptividad o cualquier otra expresión lógica.

5 Ecuaciones lógicas

Las ecuaciones lógicas permiten definir las salidas, las receptividades de un grafcet, los bits internos. También permiten declarar las condiciones de activación o de reinicio de los bloques de funciones (temporizadores, contadores). Las reglas siguientes se aplican en todos los casos::

Los operandos son los mnemónicos de la entrada, etapa, temporizador, contador, posiblemente variables de salida y bits internos..

Los tres únicos operadores utilizados son Y simbolizado por un punto (.), O simbolizado por un signo más (+) y NO simbolizado por una barra oblicua (/).

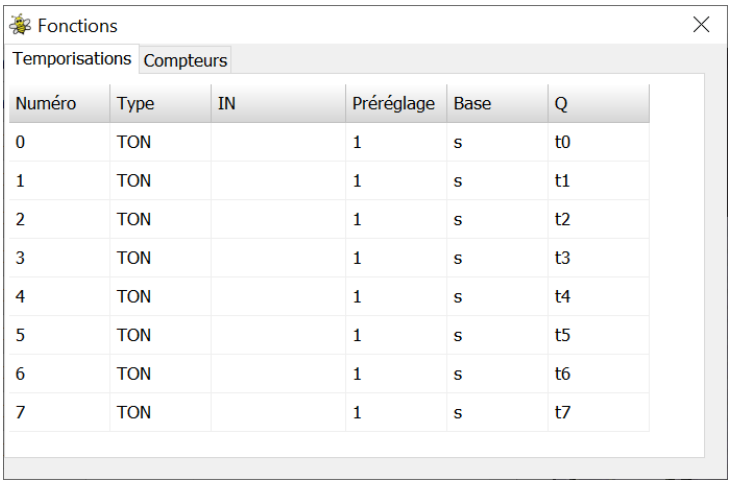
Ejemplo : I1 O i2 Y i3 Y NO i4 se escribirá como i1+i2.i3.i4/

Nota :

- el operador / se aplica a la variable o expresión precedente,
- una ecuación se declara directamente en el espacio que la recibe, no comienza con un signo de igualdad,
- Los corchetes se utilizan para priorizar las operaciones si es necesario: i1+i2.i3 corresponde a i1+(i2.i3), distinto de (i1+i2).i3,
- el operador . no está implícito: i1.(i2+i3) no puede escribirse i1(i2+i3),
- es posible separar las variables del operador mediante espacios,
- se pueden utilizar los valores 0 y 1 (falso y verdadero) (1 permite, por ejemplo, declarar una receptividad siempre verdadera).

6 Temporizadores y contadores

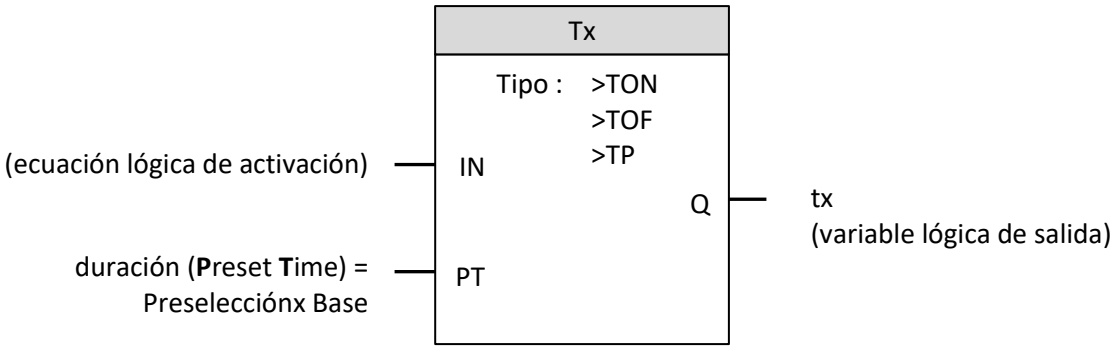
Apimou permite utilizar 8 temporizadores (numerados de 0 a 7) configurables como "TON", "TOF", "TP" y 8 contadores (también numerados de 0 a 7) de tipo "CTU". Se puede acceder a estas funciones a través del menú Ver -> Funciones.



Numéro	Type	IN	Préréglage	Base	Q
0	TON		1	s	t0
1	TON		1	s	t1
2	TON		1	s	t2
3	TON		1	s	t3
4	TON		1	s	t4
5	TON		1	s	t5
6	TON		1	s	t6
7	TON		1	s	t7

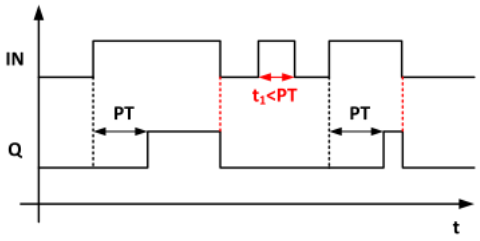
Cada temporizador o contador tiene asociada una variable lógica de salida Q impuesta (por ejemplo, t4 para el temporizador 4 y ct3 para el contador 3).

Los temporizadores pueden representarse mediante el siguiente bloque de funciones: :



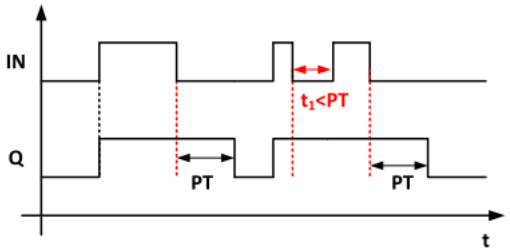
La ecuación lógica de activación (IN) puede introducirse desde el teclado o desde el menú contextual que aparece al hacer clic con el botón derecho del ratón sobre la celda. Puede, como cualquier otra ecuación, constar de una sola variable.

Cronograma del temporizador TON (retardo de conexión) :



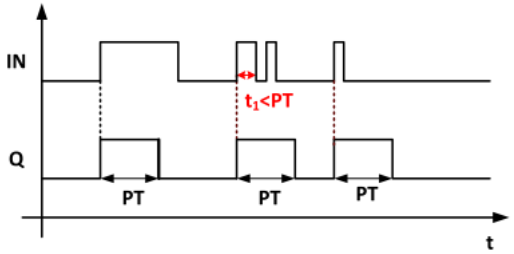
fuente : International Journal of Mechatronics and Applied Mechanics, 2018, Issue 4

Cronograma del temporizador TOF :



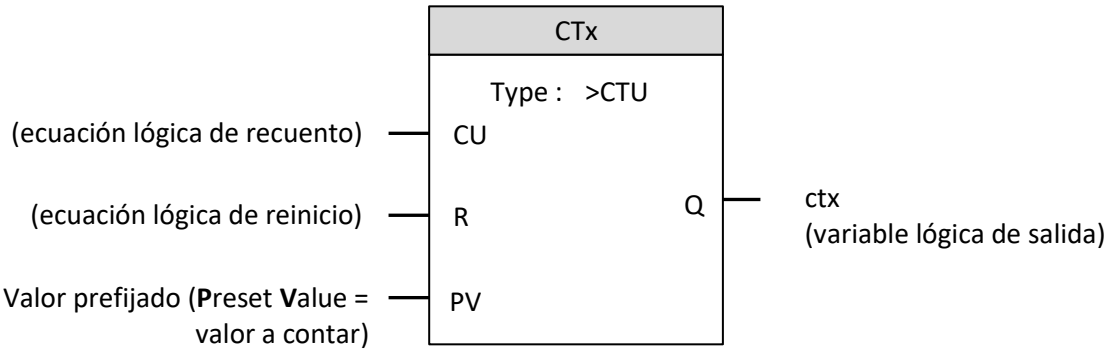
fuente : International Journal of Mechatronics and Applied Mechanics, 2018, Issue 4

Cronograma del temporizador TP (impulso de duración fija):



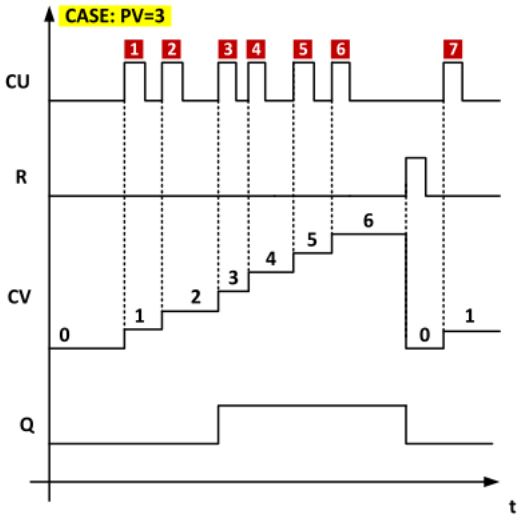
fuente : International Journal of Mechatronics and Applied Mechanics, 2018, Issue 4

Los contadores pueden representarse mediante el siguiente bloque de funciones:



Las ecuaciones lógicas de recuento (CU) o de reinicio (R) pueden introducirse mediante el teclado o el menú contextual que aparece al hacer clic con el botón derecho del ratón sobre la celda. Pueden, como cualquier otra ecuación, constar de una sola variable.

Cronograma de un contador CTU (para PV = 3):



CV : Current Value = valor actual

fuente : International Journal of Mechatronics and Applied Mechanics, 2018, Issue 4

7 Grafcet

Lo esencial de lo que hay que saber para apoderarse de un grafcet se describe en el apartado correspondiente de rápida pueste en marcha.

7.1 Etapas y acciones

Las etapas pueden ser simples o iniciales (activados al paso en RUN del autómata). La elección se realiza mediante un menú contextual (clic derecho sobre la etapa -> inicial).

Cada etapa tiene un número único. Es posible modificarlo, siempre por menú contextual (clic derecho sobre la etapa -> número, introducir el valor deseado, validar pulsando sobre un espacio libre de la ficha o con la combinación de teclas Ctrl+Enter).

La definición de una acción asociada a una etapa se hace por menú contextual (clic derecho sobre la etapa -> editar acciones, introducir la expresión deseada, validar haciendo clic en un espacio libre del formulario o con la combinación de teclas Ctrl+Enter).

Si se asocian varias acciones a la misma etapa, deben ir separadas por un punto y coma (;).

Una acción continua debe ser declarada por el mnemónico de la salida correspondiente. No es posible asignar un bit interno a una acción (siempre se puede hacer fuera del grafcet definiendo la ecuación del bit deseado, m2=x5 por ejemplo si se quiere activar el bit m2 con la etapa 5).

Una acción condicional debe declararse utilizando la sintaxis condición ? salida.

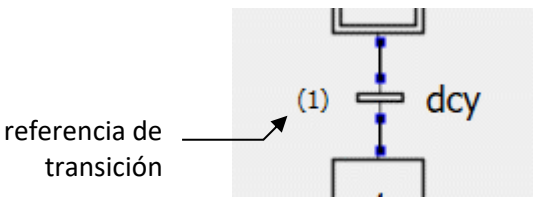
Las ecuaciones de las salidas correspondientes a las acciones del grafcet se definen *automáticamente* y no es posible modificarlas. Tampoco se puede programar ninguna acción en el grafcet e introducir manualmente las ecuaciones de las salidas.

Ejemplos de acciones:

Grafcet	Grafcet Apimou	Ecuaciones de salida (definidas automáticamente)
3 — q7	3 — q7	q7 = x3
3 — q7 9 — q7	3 — q7 9 — q7	q7 = x3 + x9
3 — q7 | q10	3 — q7 ; q10	q7 = x3 q10 = x3
I0.(i1/+i4) 3 — q7 | q10	3 — i0.(i1/+i4)?q7 ; q10	q7 = x3.(i0.(i1/+i4)) q10 = x3
I0.(i1/+i4) 3 — q7 | q10 9 — q7	3 — i0.(i1/+i4)?q7 ; q10 9 — q7	q7 = x3.(i0.(i1/+i4)) + x9 q10 = x3

7.2 Transiciones y receptividad

Las transiciones se marcan con un número que las identifica unívocamente. Estos marcadores pueden mostrarse u ocultarse haciendo clic con el botón derecho en cualquier transición y eligiendo "ver referencias".



Una transición tiene por defecto una receptividad asociada que es siempre falsa (0), lo que implica que aunque la transición esté validada (etapas ascendentes activas), nunca se franquea. La modificación de una receptividad se hace por menú contextual (clic derecho sobre la transición -> editar la receptividad, introducir la expresión deseada, validar haciendo clic en un espacio libre de la ficha o con la combinación de teclas Ctrl+Enter).

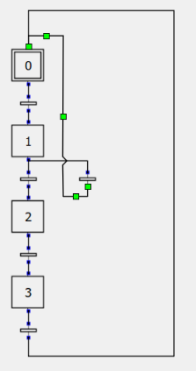
Una receptividad siempre verdadera se define por el valor 1. Una receptividad no definida equivale a una receptividad siempre falsa.

7.3 Uniones orientadas

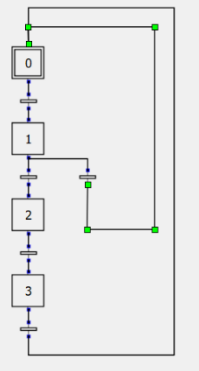
Las uniones etapa <-> transición sólo pueden trazarse en el sentido de su lectura:

- ✓ Para una unión etapa->transición, partimos del punto de enlace descendente de la etapa hasta el punto de enlace ascendente de la transición, lo contrario es imposible.
- ✓ Para una unión transición->etapa, partimos del punto de enlace aguas abajo de la transición hasta el punto de enlace aguas arriba de la etapa, lo contrario es imposible.

Si el modo de trazado por defecto de la unión (automático) no resulta visualmente satisfactorio, debido a las líneas que pueden cruzarse entre sí, puede seleccionarse el modo " unión manual" en la barra de herramientas, que permite trazar líneas de varios segmentos y disponer estos segmentos como se desee.



unión automática

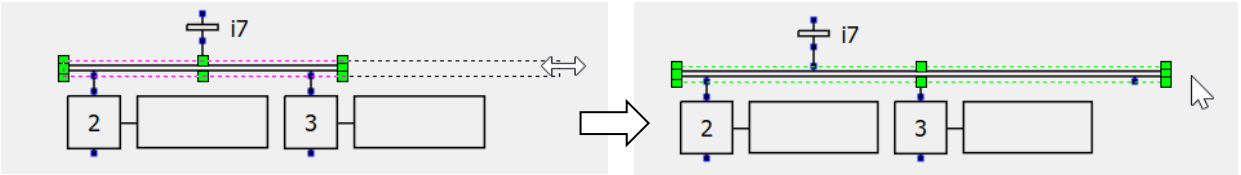


unión manual

No se permiten construcciones que no se correspondan con las reglas del grafcet, en particular la regla de alternancia etapa/transición. Por lo tanto, todo lo que se puede dibujar es conforme.

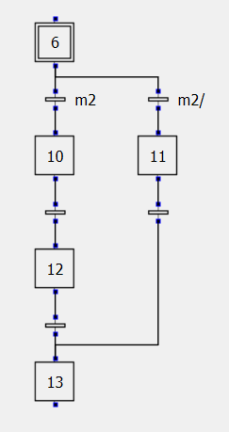
7.4 Divergencias y convergencias en "Y"

Estas construcciones se utilizan para secuencias simultáneas.
Los divergentes tienen una entrada y, por defecto, dos salidas. Si desea iniciar tres (o más) secuencias simultáneas, basta con redimensionar el bloque arrastrándolo hacia la derecha:



El principio es el mismo para las convergencias.

7.5 Divergencias y convergencias en “O”



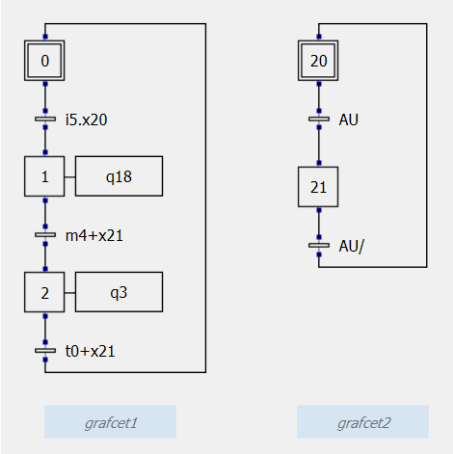
Estas construcciones se utilizan para selecciones de secuencia (o saltos).

No hay nada especial que señalar aquí, simplemente es necesario asegurarse de que las receptividades de las transiciones sobre las que se realiza la selección son (lógica o físicamente) excluyentes, para evitar la proliferación de tokens.

7.6 Grafcet múltiple

Es posible, para una misma aplicación, introducir varios grafcet, sincronizados o independientes.

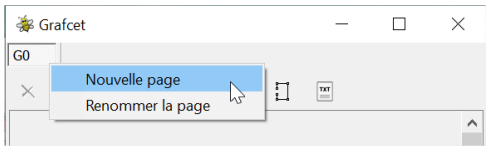
Ejemplo de grafcet sincronizado:



7.7 Edición multipágina

Si hay que introducir varios grafcet, cada uno de ellos puede introducirse en una página diferente. Si desea estructurar el programa jerarquizando los grafcet y utilizando ordenes de forzado, este método es obligatorio: cada grafcet se denomina de esta forma y, a continuación, puede activar un comando de forzado que designe un grafcet específico.

Por defecto, sólo hay una página en el espacio de edición, denominada "G0". Puede añadir una nueva página haciendo clic con el botón derecho del ratón en la pestaña y seleccionando "nueva página" en el menú emergente.



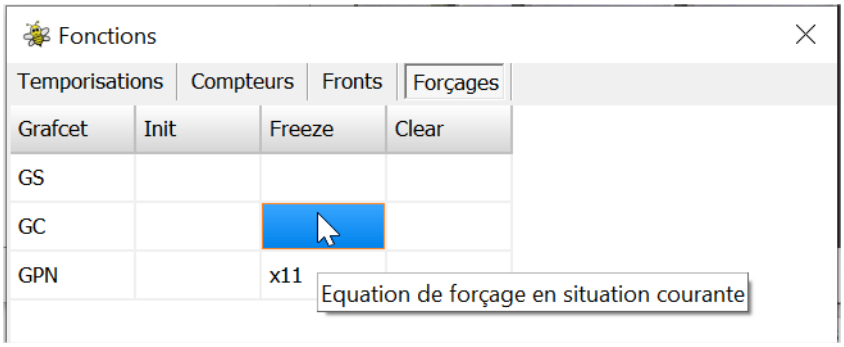
7.8 Órdenes de forzado

A menudo, un sistema lógico de control completo debe estructurarse de forma que puedan procesarse y coordinarse varios modos de funcionamiento, teniendo en cuenta todas las situaciones posibles (véase GEMMA y los grafcets jerárquicos derivados de él). En este caso, puede ser útil utilizar órdenes de forzado para poner inmediatamente un grafcet en una situación determinada.

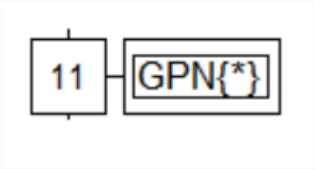
Apimou permite tres acciones de forzado para cada grafcet: INIT, FREEZE y CLEAR. Estas acciones están disponibles en el menú Ver -> Funciones, pestaña Órdenes de forzado.

INIT	Reinicia el grafcet: se activa la etapa inicial (las etapas iniciales si hay varios), los demás se desactivan.
FREEZE	Fija el grafcet en la situación actual: ya no evoluciona, y no se puede hacer ninguna transición sean cuales sean los valores de receptividades.
CLEAR	Pone el grafcet en estado vacío: todos las etapas, iniciales o no, se desactivan.

Cada una de estas acciones se desencadena mediante una ecuación lógica. Obviamente, puede resumirse en una simple variable de etapa.



La configuración anterior, por ejemplo, corresponde en representación grafcet estándar a :



(réalisé avec Editsab)

Nota: si dos o más de estos tres comandos estuvieran activos simultáneamente para el mismo grafcet (lo que no debería ocurrir normalmente), CLEAR tiene prioridad sobre INIT, que a su vez tiene prioridad sobre FREEZE.

7.9 Varias funciones de edición

Para insertar varios bloques (etapa o transición o comentario) sin hacer clic cada vez en la barra de herramientas : pulse la tecla *Mayús (Shift)* del teclado y el *botón Etapa o el botón Transición o el botón Comentario de la barra de herramientas* y, a continuación, suelte la tecla Mayús. Haga clic en el área de edición tantas veces como sea necesario, moviendo el ratón para distribuir los bloques. Haga clic en el botón *Cancel inserting* de la barra de herramientas para detener la inserción**.**

Para alinear varios elementos verticalmente : seleccione los elementos (haga clic y arrastre) y, a continuación, haga clic con el *botón derecho del ratón* y seleccione *Alinear verticalmente* en el menú contextual.

Para dividir varios elementos verticalmente : seleccione los elementos (haga clic y arrastre), luego haga clic con el *botón derecho del ratón* y elija *Dividir verticalmente* en el menú emergente.

Para aumentar o disminuir el espacio entre varios elementos alineados y distribuidos verticalmente : pulse la tecla *Alt* del teclado y, manteniendo pulsada la tecla, gire la *rueda del ratón* hacia arriba o hacia abajo.

Para insertar una secuencia lineal de 1 a 9 etapas : sitúa el ratón en la zona de edición donde se va a insertar la secuencia y, a continuación, pulsa la tecla *Ctrl* del teclado y cualquiera de las *teclas 1 a 9* del teclado (no las teclas del teclado numérico).

Para aumentar o reducir el zoom : pulsa la tecla *Ctrl* del teclado y, manteniéndola pulsada, gira la *rueda del ratón* hacia arriba o hacia abajo.

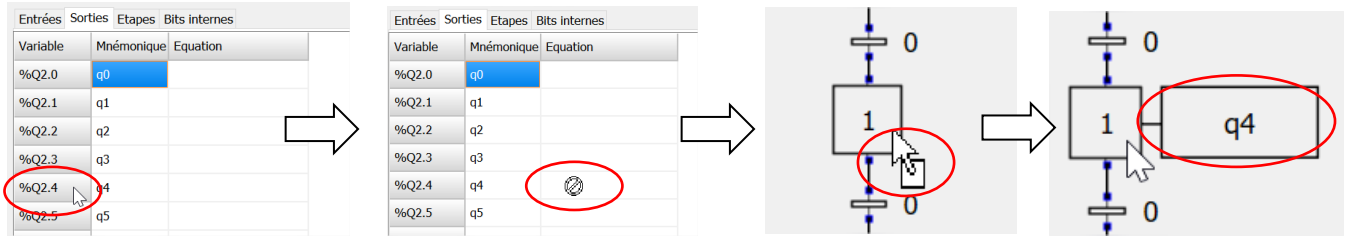
Para volver al nivel de zoom por defecto (100%): pulse el *botón central* del ratón.

Para introducir acciones y receptividades mediante arrastrar y soltar (drag & drop) :

Las *entradas*, *salidas*, *etapas* y *bits internos* pueden arrastrarse y soltarse en el grafcet haciendo clic con el botón izquierdo del ratón en la casilla *Variable* correspondiente del formulario principal, moviendo después el ratón hacia el grafcet y soltando el botón en un paso o transición. Para los *temporizadores* y *contadores*, puede hacer lo mismo haciendo clic en la casilla de salida (Q) de la hoja de funciones correspondiente al temporizador o contador deseado.

Las *etapas* sólo aceptan variables de *salida*, las *transiciones* aceptan *todas* las variables.

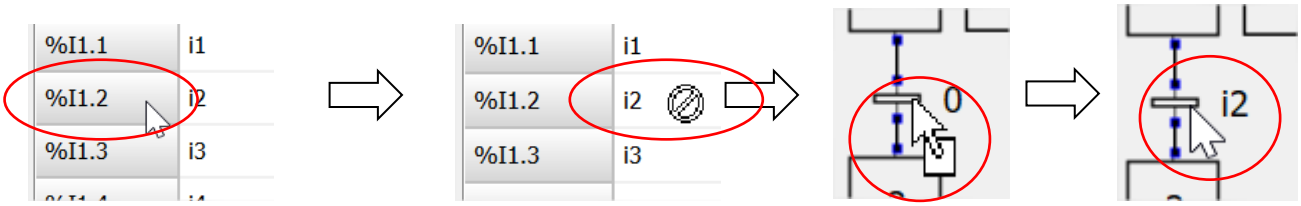
→ Caso de una variable de salida de arrastrar y soltar en una etapa:



Si la salida se coloca en una etapa que ya tiene una o más acciones, se añade a las acciones existentes. Las acciones condicionales no pueden declararse de este modo.

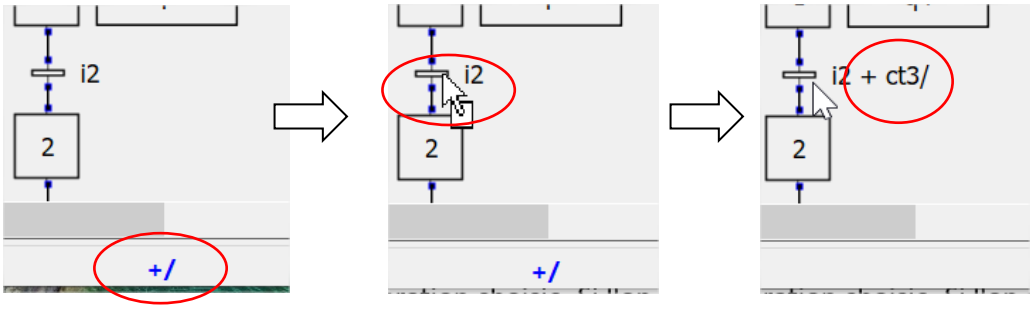
→ Caso de cualquier variable arrastrada y soltada en una transición:

Sin acción previa, una receptividad definida o indefinida (no vacía, vacía o con valor por defecto 0) será sustituida por la variable depositada :



Si desea introducir el complemento de la variable seleccionada (por ejemplo, NO i2 o i2/), debe pulsar primero la tecla "/" (Slash) del teclado. Este carácter ("/") aparecerá entonces en la barra de estado del fichero grafcet. A continuación, puede arrastrar y soltar.

Para completar una receptividad con una secuencia de operaciones lógicas, primero debe pulsar la tecla correspondiente a la operación deseada ("+" para OR o "." para AND). El carácter correspondiente ("+" o ".") se introducirá en la barra de estado del fichero grafcet. La variable arrastrada se añadirá entonces a la receptividad con la operación elegida. Si desea añadir el complemento de una variable del mismo modo, deberá pulsar la operación elegida ("+" o ".") y después la tecla "/" antes de arrastrar y soltar la variable:



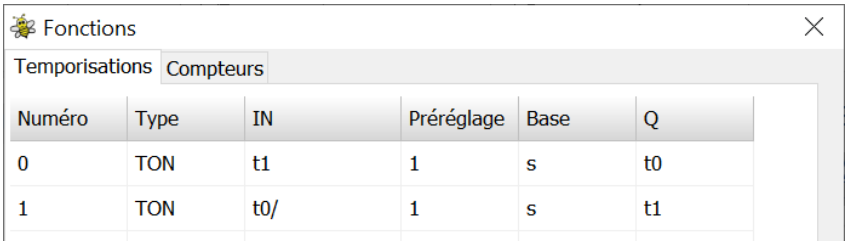
No es posible priorizar las operaciones de este modo.

8 Funciones especiales

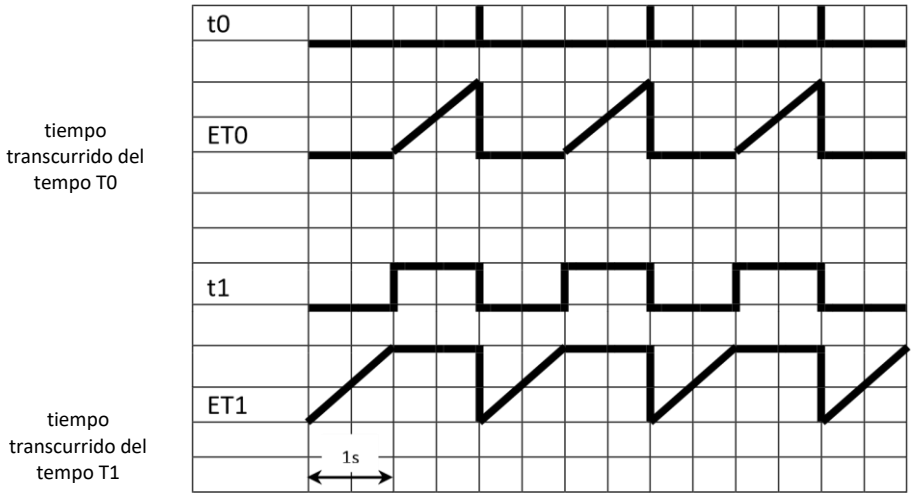
Las siguientes funciones no están disponibles directamente en Apimou, pero es posible crear equivalentes.

8.1 Señal de reloj o intermitente

Para obtener un bit intermitente, se pueden utilizar dos temporizaciones Ton :



La operación corresponde al siguiente cronograma :

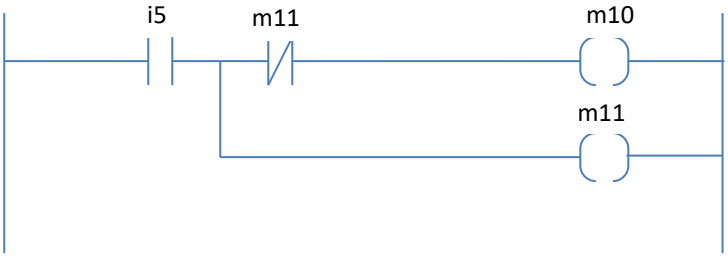


La frecuencia de la señal se varía variando el tiempo prefijado (PT) de los dos temporizadores. Con un PLC "real" este tipo de señal suele estar disponible en un bit del sistema.

8.2 Flanco de subida o bajada

Para crear un flanco de subida en una variable, por ejemplo i5, hay que utilizar dos bits internos, por ejemplo m10 y m11.

El esquema equivalente en lenguaje de escalera sería el siguiente:



El PLC escanea el programa de izquierda a derecha y de arriba a abajo. Cuando i5 pasa a 1, con el bit m11 a 0, el bit m10 pasa a 1 (1ère línea). m11 inmediatamente después (en el mismo ciclo PLC) pasa a 1 ya que i5 es 1. En el siguiente ciclo (PLC), i5 sigue siendo 1, m11 es 1, y m10 vuelve a caer a 0. Esto da m10 = i5↑, por lo que m10 puede utilizarse como flanco ascendente para i5.

Apimou no se programa en escalera sino en ecuaciones, lo que da:

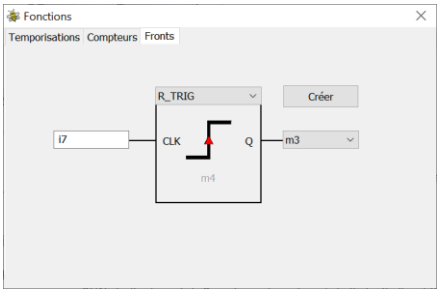
(i5↑ =) m10 = i5.m11/ y m11 = i5

un flanco de bajada, tendríamos:

(i5↓ =) m10 = i5/.m11/ y m11 = i5/

Nota : El bit que sirve de flanco de subida (m10 para el ejemplo) debe imperativamente tener un número inferior al que sirve de variable intermedia (aquí m11), ya que Apimou calcula los bits internos en orden ascendente de sus números.

Para crear rápidamente un bit de flanco, según el principio descrito anteriormente, se puede utilizar el asistente accesible desde el menú Ver->Funciones, pestaña Flancos :



A continuación, basta con definir la variable (o expresión) cuyo flanco se desea detectar, el tipo de flanco (de subida o bajada) y la imagen de bits interna del flanco. Al pulsar el botón Crear se generan automáticamente las ecuaciones necesarias.

Los mnemónicos de bits internos aceptan los caracteres "↑" (Alt+24) y "↓" (Alt+25). Si, por ejemplo, se crea un flanco ascendente de %I1.7 (mnemónico por defecto "i7") con el bit interno %M3 (mnemónico por defecto "m3"), el mnemónico de %M3 puede cambiarse a "i7↑".

Nota : "↑" y "↓" son sólo caracteres, no funciones. La creación de un frente pasa necesariamente mediante el procedimiento detallado anteriormente.

8.3 Bit de inicialización

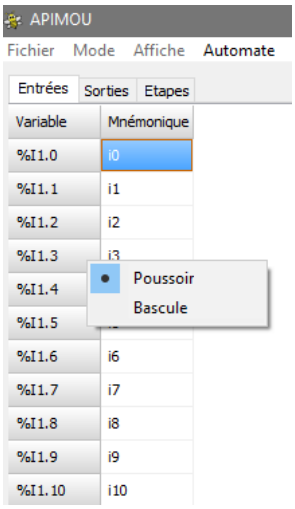
Un bit de inicialización es un bit que se pone a 1 en el primer ciclo del PLC y a 0 a partir de entonces (es decir, un flanco de subida en el RUN del PLC). En los PLCs reales, es un bit de sistema. Como Apimou no tiene uno, podemos hacer un equivalente de la siguiente manera:

Entrées | Sorties | Etapes | Bits internes

Variable	Mnémonique	Equation
%M0	init	1.m1/
%M1	m1	1

La explicación de cómo funciona se desprende de lo escrito en el párrafo anterior.

9 Funcionamiento en modo simulación



Menú Modo -> Simulación.

Como el PLC está en modo RUN (es decir, ejecutando el programa), el estado de las entradas puede cambiarse pulsando sobre las variables correspondientes en la pestaña Entradas de la ventana principa. Los LED de la tarjeta de entrada se encenderán. Para cada variable, es posible elegir una operación de pulsador (modo por defecto) o una operación de conmutación haciendo clic con el botón derecho del ratón en la variable elegida.

La visualización del estado de las etapas se realiza en el grafcet (etapa(s) activa(s) en verde)

El estado de los bits internos se muestra en la tabla correspondiente (celda mnemotécnica en verde = bit activado).

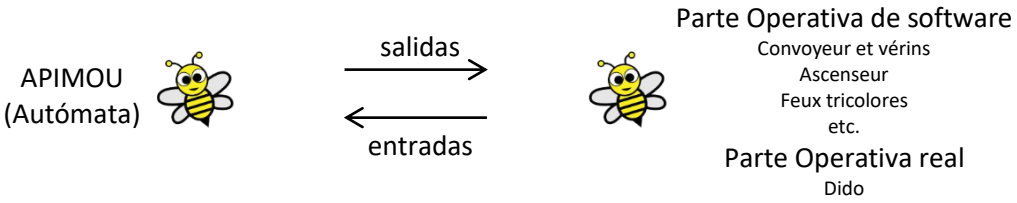
El estado de las salidas se visualiza directamente mediante los LED de la tarjeta de salida.

10 Funcionamiento en modo conectado

Menú Modo -> Conectado.

El funcionamiento de Apimou es el mismo que en modo simulación, cuando el PLC está en modo RUN, se puede ver el estado de las entradas/salidas en las tarjetas, el estado de los bits internos en la tabla y el estado de las etapas en el grafcet.

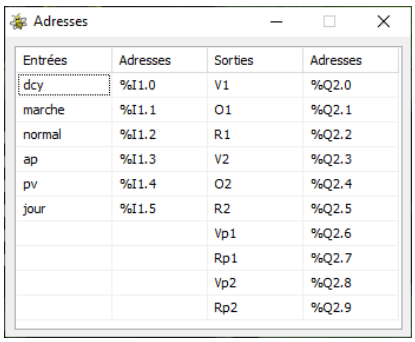
En este modo, Apimou intercambia con una parte operativa de software o real :



Para ello es necesario que la parte operativa esté conectada a Apimou, lo que puede hacerse a través del menú Apimou -> Conectar.

También es necesario conocer el esquema eléctrico del PLC (Apimou) para esta parte operativa Se dispone de la siguiente información: a qué entradas están conectados los sensores, a qué salidas están conectados los actuadores, es decir, una tabla de entradas/salidas. Se puede obtener en el menú Ver -> Direcciones.

Ejemplo de *feux tricolores* :



El pulsador ap (llamada peatonal), por ejemplo, está conectado a la cuarta entrada del PLC (Apimou), dirección %I1.3.

En Apimou, el mnemotécnico por defecto para la entrada %I1.3 es i3. Si lo sustituimos por ap, podremos utilizar ap en el programa para referirnos a la entrada %I1.3.

Los archivos "config_(nombre OP).api", una vez cargados en Apimou, evitan la tediosa tarea de cambiar todos los mnemónicos para que coincidan con las etiquetas de la parte operativa.

Para una parte operativa real (ocho entradas/ocho salidas como máximo), Dido puede utilizarse para controlar un módulo de E/S externo en un enlace RS-485 en Modbus RTU (no es necesario entender qué es el enlace RS-485 o el protocolo Modbus RTU).

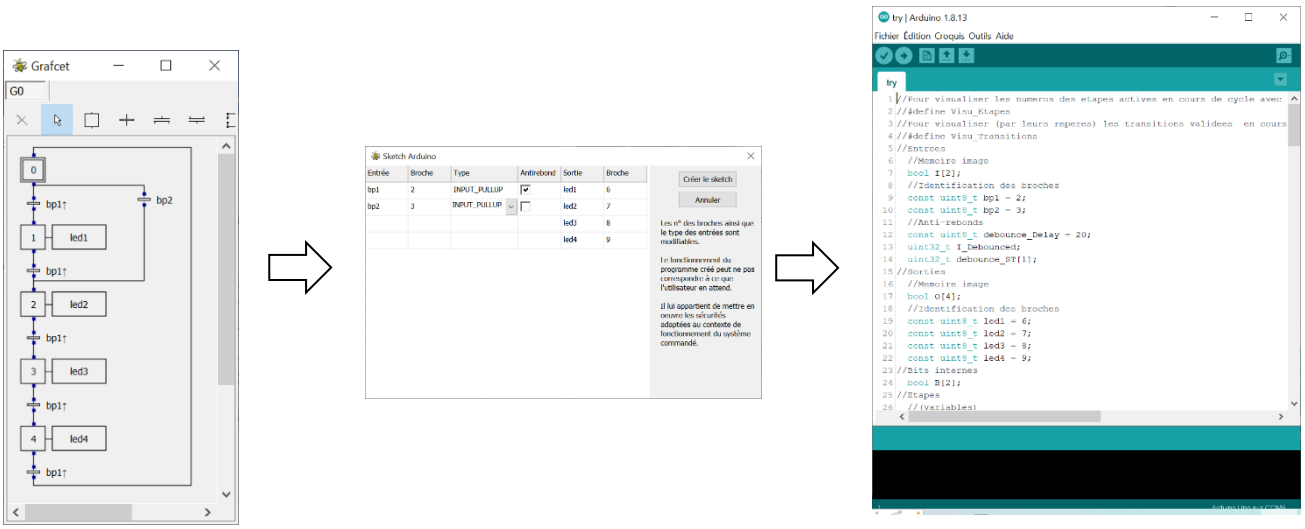
Nota : Los elementos de mando suministrados con las versiones 1.x.x de Apimou (incluido Dido) no son compatibles con las versiones 2.x.x.

11 Generación de código Arduino

Nota : Esta función sólo genera código C++ compatible con Arduino. No hay comunicación entre Apimou y Arduino.

El menú *Archivo -> Crear un sketch Arduino* permite transformar el programa en un sketch Arduino (fichero ".ino"). Es posible modificar los pines de las entradas/salidas así como elegir para cada entrada el tipo de operación (INPUT_PULLUP, PULLUP, PULLDOWN).

También se puede asignar a cada entrada una función antirrebote. Esta opción puede ser útil para sensores mecánicos cuya señal puede rebotar brevemente al cambiar de estado. Si, por ejemplo, las pulsaciones sucesivas de un pulsador deben comprobarse mediante un flanco ascendente, o un contador debe incrementarse del mismo modo, los rebotes de la señal pueden provocar un funcionamiento errático. Para solucionar este problema se utiliza un dispositivo antirrebote. El anti-rebote se consigue aquí mediante un temporizador que congela el valor de la entrada durante un periodo determinado en cuanto cambia de estado. El tiempo por defecto es de 20 ms, que puede cambiarse en el menú *Herramientas->Opciones*.



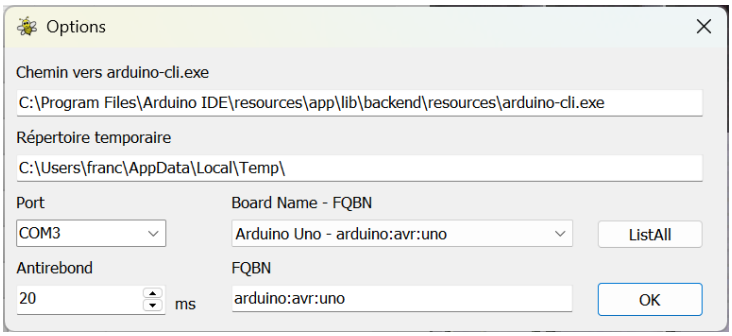
12 Transferencia directa de un programa a una placa Arduino

Un programa creado con Apimou se puede transferir (cargar) directamente a una placa Arduino, sin tener que pasar por el IDE de Arduino. Esta característica requiere que el programa **arduino-cli.exe** esté instalado en el ordenador. Ya está presente en su ordenador si ha instalado la **versión 2.x del IDE Arduino**, de lo contrario, con un IDE Arduino 1.x, tendrá que instalar este programa de forma independiente.

Lo único que hace aquí Apimou es recuperar la información necesaria para que arduino-cli compile y transfiera el programa, usando el comando :

arduino-cli compile --upload --verbose --no-color (sketch.ino) --port (nombre del puerto) --fqbn (identificador FQBN)

Antes de realizar una primera transferencia directa, debe introducir los parámetros necesarios pulsando en el menú Herramientas->Opciones (estos parámetros se guardarán para su uso posterior en un archivo llamado Apimou.ini) :



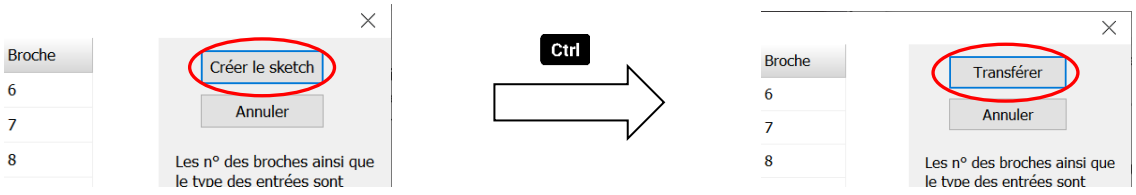
Para los usuarios del IDE Arduino 2.x, la ruta a arduino-cli.exe debería en principio rellenarse automáticamente. Si arduino-cli.exe se ha instalado de forma independiente, o si Apimou no lo encuentra, tendrás que definirla.

Por defecto, el directorio temporal es el del usuario. Es posible cambiarlo. Apimou creará en este directorio un subdirectorio apimou_tmp en el que guardará un fichero apimou_tmp.ino. Este es el archivo que será compilado y transferido por arduino-cli. Es borrado y reescrito cada vez que se realiza una nueva transferencia.

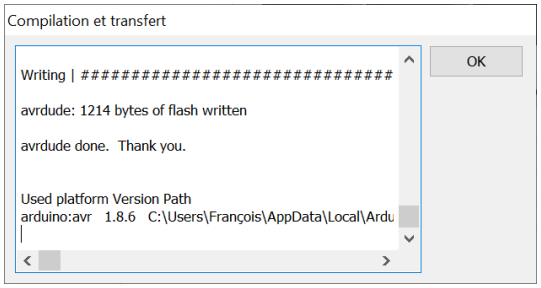
El puerto, a seleccionar de la lista desplegable, debe ser aquel al que está conectada la placa Arduino. Si tienes varios dispositivos USB conectados, puede que el puerto seleccionado por defecto no sea el correcto. Si la placa no está conectada, el puerto no aparecerá. Tendrá que conectar la tarjeta y, a continuación, cerrar y volver a abrir la ventana de opciones.

El FQBN (Fully Qualified Board Name) es el de la tarjeta utilizada. Si la tarjeta está instalada y la ruta a arduino-cli es válida, puedes encontrar el FQBN haciendo clic en el botón ListAll y luego eligiendo la tarjeta de la lista desplegable.

El procedimiento para transferir el programa directamente a la placa es entonces a través del menú Archivo -> Crear sketch Arduino, tal y como se ha explicado en el párrafo anterior. En el formulario que aparece, manteniendo pulsada la tecla Ctrl del teclado, el botón "Crear sketch" pasa a llamarse "Transferir":



Al hacer clic en este botón se inicia el procedimiento de compilación y transferencia. Una ventana muestra los resultados de la operación, proporcionados por arduino-cli :



Si el procedimiento falla, puede que necesite :

- Detectar la placa conectada con el comando *arduino-cli board list*
- Listar las placas instaladas y obtener su FQBN con el comando *arduino-cli board listall*
- Listar las familias de placas instaladas con el comando *arduino-cli core list*
- Lista las familias de placas que pueden ser instaladas con el comando *arduino-cli core search*
- Instala una familia de placas con el comando *arduino-cli core install (...)*
- y así sucesivamente. La documentación oficial se puede encontrar en https://github.com/arduino/arduino-cli

Si necesitas ejecutar un comando, puedes, por ejemplo

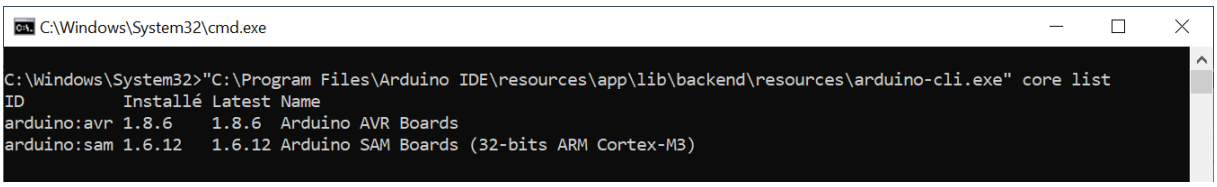
- Copiar/pegar la ruta a arduino-cli en un editor de texto (bloc de notas, por ejemplo)
  C:\Program Files\Arduino IDE\resources\(…)\arduino-cli.exe

- Rodéalo con comillas (para evitar malas interpretaciones debido a los espacios)
  "C:\Program Files\Arduino IDE\resources\(...)\arduino-cli.exe"

- Añade el comando deseado
  "C:\Program Files\Arduino IDE\resources\(…)\arduino-cli.exe" core list

- Abra el intérprete de comandos de Windows cmd.exe (escriba cmd.exe en el campo de búsqueda del menú Inicio)

- Copie la línea del editor de texto y péguela en cmd.exe :



13 Generación de código ESP32

13.1 información general

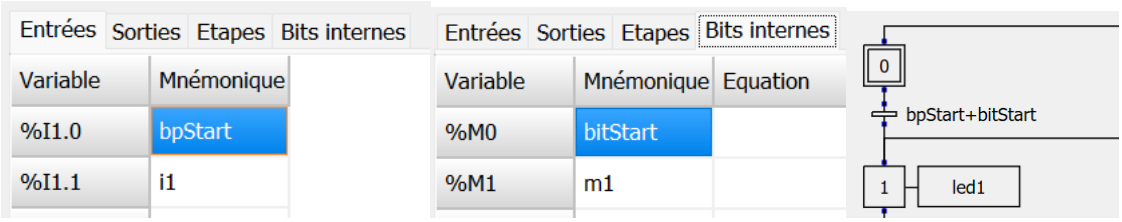
Con un ESP32 es posible generar código C++ (archivo .ino) y un conjunto de archivos (.html, .png) que se pueden utilizar con cualquier navegador web para ver el progreso del grafcet o grafcets que componen el programa. La comunicación es vía WiFi, por lo que se puede utilizar cualquier terminal (PC, smartphone, tablet) conectado a la misma red que el ESP32.

Además de esta pantalla, es posible que también pueda acceder a los ajustes a través de una página web específica :

- ✓ cambios en los bits, denominados *bits HMI* (si se utilizan),
- ✓ cambios en los valores predefinidos del temporizador (si se utiliza),
- ✓ cambios en los valores predefinidos del contador (si se utiliza).

Los *bits HMI* son, en Apimou, bits internos que *no están definidos por una ecuación* y que *intervienen en cualquier ecuación* (salida o temporizador o contador o forzado o transición).

Ejemplo :



bpStart es una entrada (un pulsador, por ejemplo), conectada físicamente a un pin del ESP32. bitStart es un bit interno que no tiene ecuación. bitStart interviene en la receptividad de la transición paso 0 -> paso 1 (ecuación «bpStart + bitStart», que significa que el ciclo se inicia si se pulsa el pulsador bpStart o si bitStart cambia a 1). Cuando el programa se traduzca en un sketch, bitStart se considerará un bit HMI y será posible forzarlo desde la página de ajustes.

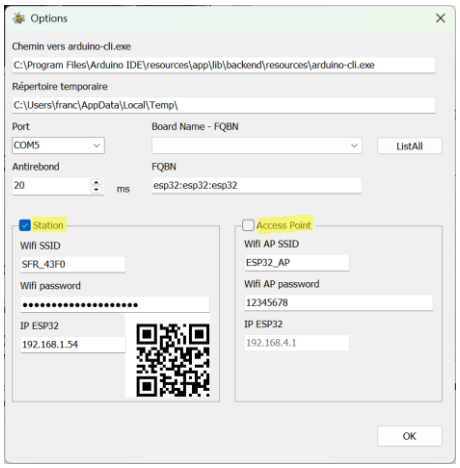
Para *validar* un ajuste, debe pulsarse el botón de validación correspondiente. Este ajuste seguirá siendo válido hasta que se reinicie el ESP32. Para *guardar* todos los ajustes habilitados (incluidos los bits HMI), haga clic en el botón guardar. La copia de seguridad se realiza en la memoria Flash (NVS) y se carga cada vez que se reinicia el ESP32.

13.2 Requisitos previos :

- IDE Arduino 2.x
- Tarjeta ESP32 instalada
- Plugin LittleFS Uploader (ver https://randomnerdtutorials.com/arduino-ide-2-install-esp32-littlefs/) instalado
- Biblioteca TCP Async de ESP32ASync instalada
- Biblioteca ESP Async WebServer de ESP32Async instalada

13.3 Modos Station y Access Point

Antes de generar el sketch, seleccione en el menú *Herramientas ->Opciones* el modo Wifi a utilizar: **Station** y/o **AccessPoint**.



En modo **Station**, esp32 se conecta a una red Wifi existente. Para ello, es necesario conocer el nombre (SSID) de la red y la contraseña que autoriza el acceso a la misma. El router de la red (la "box") asigna automáticamente una dirección IP al ESP32. Siempre que haya señal suficiente, el modo Station es el más eficaz

En modo **Access Point**, el esp32 genera su propia red WiFi y actúa como router. Se definen el nombre de la red (SSID) y la contraseña. Se impone la dirección IP del esp32, igual a 192.168.4.1. El modo Access Point está limitado a 4 clientes, en la práctica 1 o 2 para un refresco fluido y siempre que la frecuencia no supere los 100 a 200 ms.

Es posible elegir ambos modos simultáneamente, **Station** + **Access Point**. En este caso, el modo Access Point puede utilizarse como reserva, en caso de que la red WiFi existente se caiga o esté fuera de cobertura. Esta elección sigue siendo menos eficaz que el modo Station solo.



Para que un dispositivo (PC, smartphone, tableta) pueda acceder al servidor ESP32 en modo **Access Point**, primero debe conectarse a la red Wi-Fi creada por el ESP32. Si este dispositivo estaba conectado al servidor (box) de la red Wi-Fi local existente, perderá esta conexión y, por tanto, entre otras cosas, el acceso a Internet.

Los **códigos QR** que aparecen en la hoja de opciones se refieren directamente a las direcciones IP del ESP32 (dirección en la red local y/o dirección del punto de acceso), siempre que el dispositivo esté conectado a la red Wi-Fi correspondiente.

13.4 Procedimiento

Para el modo **Station**, lo mejor es definir primero el nombre (Wifi SSID) de la red Wifi y la contraseña para esta misma red (Wifi password). Estos dos parámetros también se pueden configurar posteriormente.

Para el modo **Access Point**, primero debe definir el nombre (Wifi AP SSID) de la red Wifi AP y la contraseña para esta red (Wifi AP password). Se sugieren valores por defecto. La contraseña puede estar vacía (en cuyo caso la red estará abierta).

Después de seleccionar el menú ***Archivo -> Crear un sketch -> ESP32***, el procedimiento es idéntico al descrito en la sección Generar código Arduino. A diferencia de una placa Arduino, aquí se puede configurar una entrada como INPUT_PULLDOWN.

Tras hacer clic en ***Crear sketch***, Apimou creará un archivo .ino (el sketch) y una subcarpeta con los archivos .html y .png necesarios. Es importante que el archivo .ino se cree en una carpeta con el mismo nombre (por ejemplo, abajo, project.ino está en una carpeta llamada project).

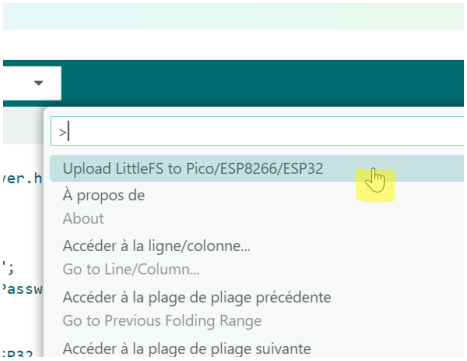
- projet\
  - projet.ino
  - (projet.api)
  - data\
    - home.html
    - vue.html
    - grafcet0.html
    - grafcet0.png
    - grafcet1.html
    - grafcet1.png
    - (…)

A continuación, abra el archivo .ino creado en el **IDE de Arduino**.

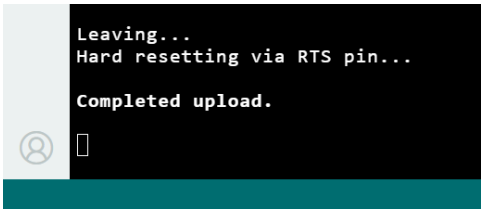
Si el SSID y la contraseña de la red WiFi en modo Station no se han introducido ya en el menú Opciones de Apimou, tendrás que hacerlo en el IDE de Arduino (manteniendo las comillas).



Tras comprobar que se ha seleccionado la tarjeta correcta, que el puerto COM es correcto y que el **monitor serie está cerrado**, pulse **[Ctrl] + [Mayús] + [P]** para abrir el menú de comandos y seleccione *Upload LittleFS to Pico/ESP8266/ESP32*.



El plugin LittleFS Uploader subirá todos los archivos de la carpeta \data del proyecto al ESP32. Después de unos segundos, debería aparecer el mensaje *Completed upload*.

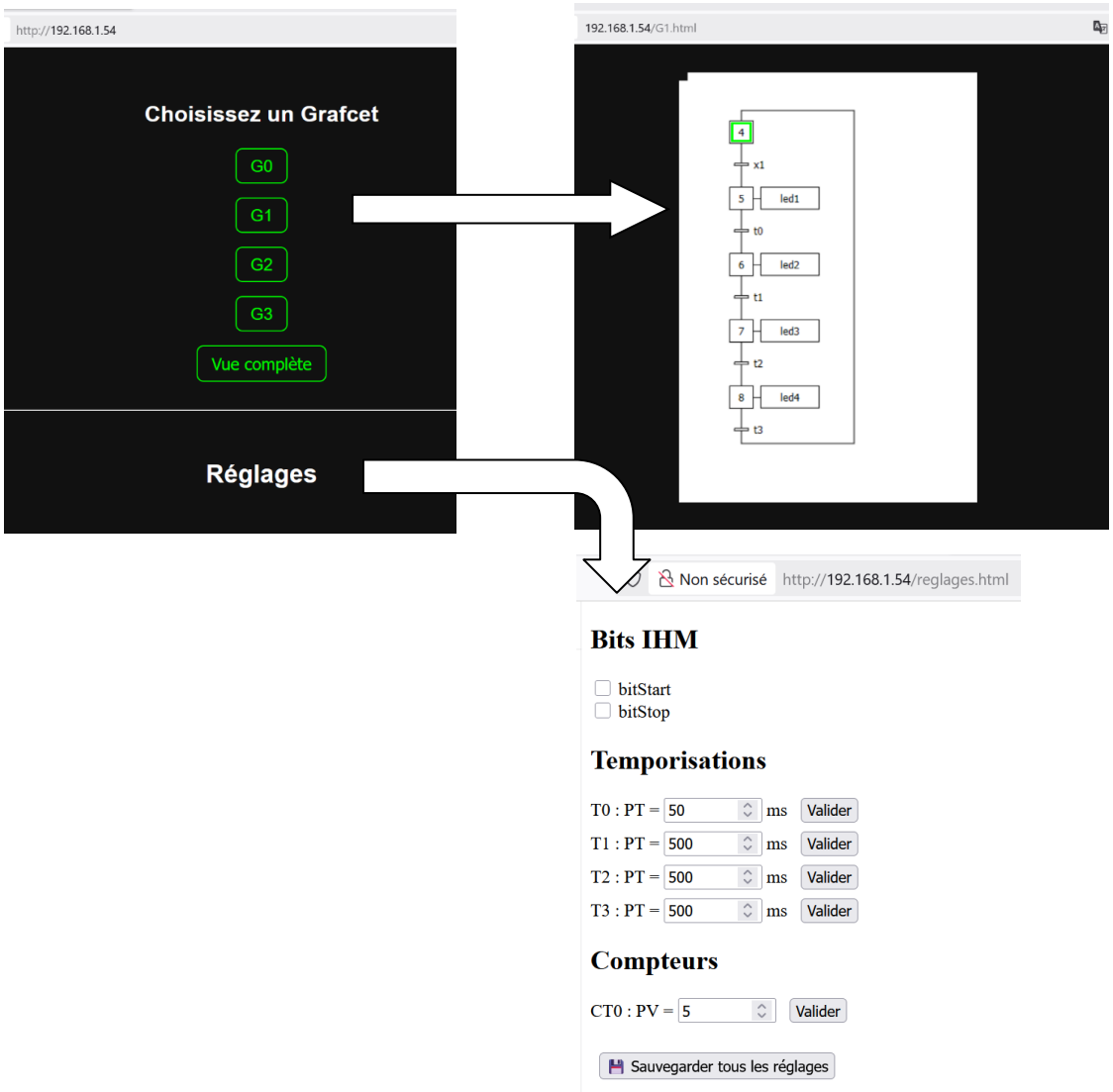


A continuación, **abre el monitor serie** a 115200 baudios y **sube el sketch** (Sketch -> Upload o [Ctrl]+[U]).

Al final de la carga, el monitor serie debería mostrar un mensaje confirmando la conexión a la red WiFi y la dirección IP del ESP32 para el modo Station y/o el modo Access Point.



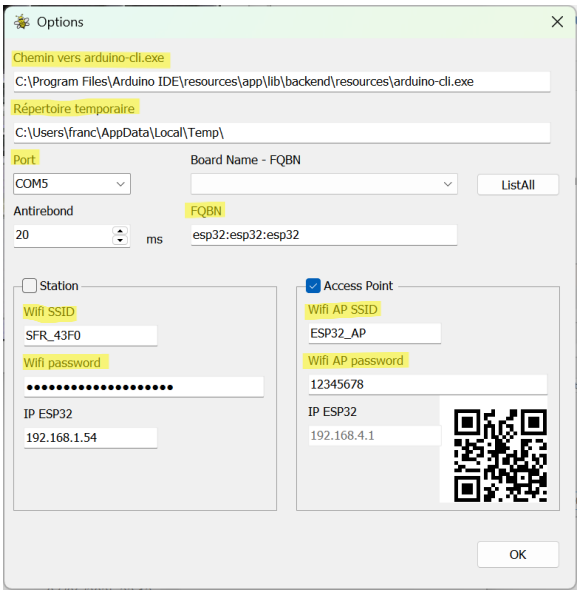
Basta con conectarse a la red deseada (red local existente en modo Station o red creada por el ESP32 en modo Access Point), copiar una de estas direcciones y pegarla en la barra de direcciones de un navegador web (o escanear uno u otro de los códigos QR) para acceder a la página de inicio de la interfaz de visualización. Desde aquí puedes seleccionar el grafcet que desees (puede que sólo haya uno) y ver cómo se anima en tiempo real. También puedes abrir la página de configuración y modificar los bits de la HMI o los preajustes de temporizadores y contadores.



14 Transferencia directa de un programa a una placa ESP32

Antes de llevar a cabo este procedimiento, es necesario haber generado y transferido un programa al menos una vez con el IDE de Arduino, tal y como se explica en el párrafo anterior, para asegurarse de que todo está en su sitio y funciona.

A continuación, tienes que introducir todos los parámetros esenciales en la ventana de opciones (Herramientas->Opciones): **ruta a arduino-cli**, **directorio temporal**, **puerto**, **FQBN**, SSID y **contraseña** Wifi para el modo(s) elegido(s) (Station y/o Access Point)..

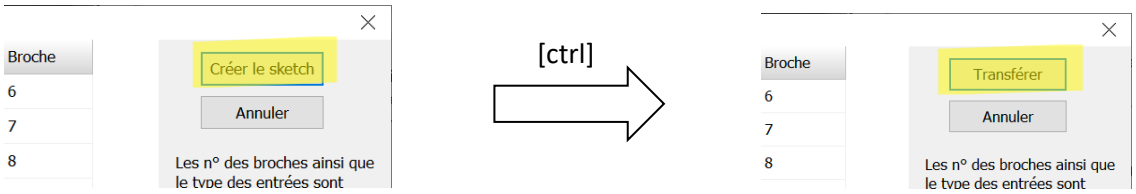


Los cuatro primeros parámetros se explican detalladamente en la sección Transferencia directa de un programa a una placa Arduino.

La dirección IP del ESP32 en modo **Station** debería encontrarse automáticamente; el campo «IP ESP32» puede dejarse en blanco. Si una

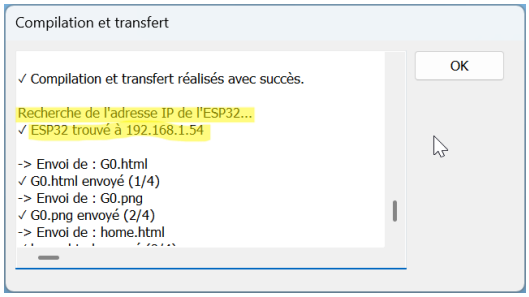
transferencia inicial falla porque no se encuentra esta dirección, puede ser necesario rellenarla.

Después de seleccionar el menú ***Archivo -> Crear sketch -> ESP32*** y configurar las entradas/salidas, basta con mantener pulsada la tecla [ctrl] del teclado y el botón «Crear sketch» pasará a llamarse «Transferir»:

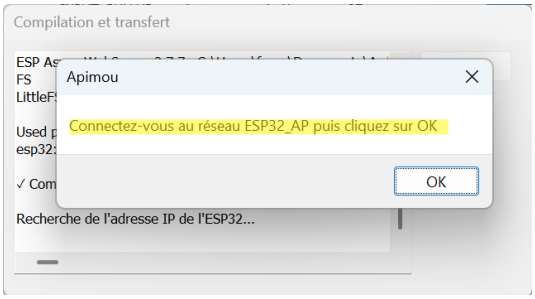


Pulsando este botón se inicia el procedimiento de compilación y transferencia. Primero, arduino-cli compila y luego transfiere el sketch. A continuación, Apimou envía los archivos .png y .html necesarios a través de WiFi (aquí no se utiliza el plugin LittleFS Uploader).

Cuando se selecciona el modo **Station** o **Station + Access Point**, los archivos html y png se transfieren en modo Station, por lo que el PC debe estar conectado a la red local existente. Una vez compilado y transferido el sketch, se busca la dirección IP del ESP 32 y se transfieren los archivos html y png.



Cuando se selecciona el modo **Access Point** solamente, el procedimiento se detiene después de la transferencia del sketch. A continuación, aparece un mensaje que le indica que se conecte a la red creada por esp32 (por defecto ESP32_AP con la contraseña 12345678), a continuación, haga clic en Aceptar para que los archivos html y png pueden ser transferidos..



Al final del procedimiento, la dirección IP del ESP32 se copia en el portapapeles. Basta con pegarla ([ctrl]+[V]) en la barra de direcciones del navegador, o volver al menú Opciones para escanear uno u otro de los códigos QR.

15 Gestión de archivos

El menú Archivo -> Nuevo borra todo (mnemónicos, ecuaciones, grafcet) sin previo aviso.

Para guardar un archivo (un programa), seleccione el menú Archivo -> Guardar como... La extensión de los archivos Apimou es ".api" (extensión añadida por defecto).

Para abrir un archivo, utilice el menú Archivo -> Abrir... También es posible, si Apimou no se está ejecutando, abrir un archivo (y de hecho Apimou) haciendo doble clic desde la carpeta en la que se encuentra. El sistema operativo le pedirá entonces que asocie al ejecutable la extensión ".api".