V-PLM, SOFTWARE ENTRENADOR DE CONTROL LÓGICO

Antonio Salvá C.
UNAM, Facultad de Ingeniería, División de Ingeniería Eléctrica
CP 04510
Tel: (55) 5622 – 3108, Fax: (55) 5616 – 1855
E mail: salva@dctrl.fi-b.unam.mx y asalva@prodigy.net.mx
Maricarmen Hernández R., Erick A. Dehesa C., Luis A. Altamirano Y.

INTRODUCCIÓN

El V-PLM es un sistema que implementa virtualmente diversas funciones de control lógico realizables por controladores lógicos programables (PLC). Las funciones realizables por el V-PLM están basadas en las propias de un PLC básico desarrollado en la Facultad de Ingeniería de la UNAM. Este PLC se denomina programador lógico modular (PLM) y cuenta con su propio lenguaje de programación, llamado SIIL1, mediante el cual el usuario puede configurar al PLM para una determinada aplicación de control de sistemas de eventos discretos.

1. ANTECEDENTES

El PLM es un PLC prototipo que cuenta con 32 entradas optoacopladas y 16 salidas asociadas a relevadores de baja potencia. También cuenta con 168 variables internas que no requieren estar asociadas a entradas o salidas físicas y su función principal es la de enlazar módulos lógicos cuando el número de entradas y/o salidas existentes no es suficiente dentro de una aplicación de control binario. El PLM realiza de manera virtual bloques funcionales que típicamente se encuentran en una aplicación de control lógico. Estos bloques funcionales van desde las conocidas compuertas lógicas, hasta elementos más elaborados como temporizadores, flip-flops, contadores de eventos o secuenciadores de estados, además de otros bloques de naturaleza auxiliar. Para el caso del PLM a esos bloques se les denomina como *módulos lógicos (ML).*

Tanto en la semántica del PLM como en la correspondiente al V-PLM se hace referencia a los ML, los cuales son bloques funcionales que realizan funciones lógicas elementales para el diseño de aplicaciones de control lógico. Éstos pueden ser representados a nivel de caja negra como se observa en la figura 1, en la que se muestra un ML que contiene *n* entradas y *m* salidas; dichas literales *n* y *m* varían de acuerdo a la función que el ML lleva a cabo, de esta manera, una compuerta OR de 4 entradas se representaría como un bloque en el que *n* tiene un valor de 4, mientras que *m* vale 1.

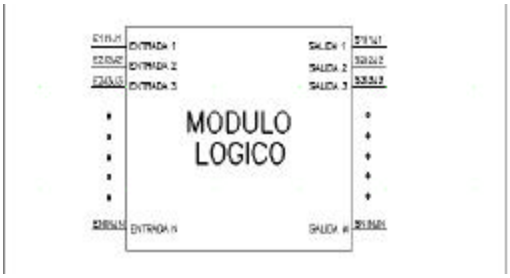


Fig 1. Representación genérica de módulo lógico.

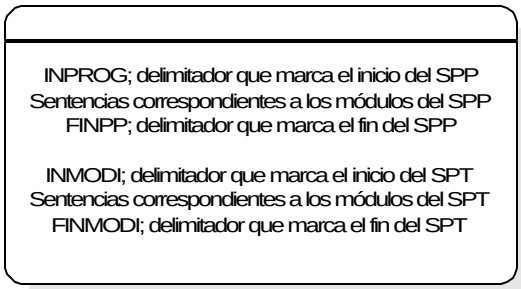
Aspectos generales acerca de hardware y software del PLM pueden verse en [1], para una descripción más detallada puede consultarse [2].

1.1 Características básicas del Lenguaje SIIL1

Para desarrollar una aplicación con el PLM, cada ML debe ser declarado por medio de uno o varios renglones de texto que definen el tipo de módulo y sus características, de tal forma que al conjunto de módulos requeridos le corresponderá una serie de renglones contenidos en un archivo de texto, el cual es procesado por una PC mediante software que genera el código objeto que deberá ejecutar el procesador central del PLM, cuyo prototipo piloto emplea el microcontrolador 68HC11F1 operando en modo expandido; además de las declaraciones asociadas con los módulos lógicos, se requiere de otras instrucciones que no están propiamente relacionadas con un determinado módulo, pero son necesarias para delimitaciones de ejecución del programa objeto en el procesador del dispositivo.

Al conjunto de declaraciones mencionadas en el párrafo anterior se le denomina programa fuente en lenguaje SIIL1, asociado con la aplicación que se desea realice el PLM. De acuerdo con lo que se ha explicado anteriormente, un programa escrito en código SIIL1 contendrá una serie de sentencias que en conjunto representarán la implementación de la solución elegida al problema de control

lógico planteado. Dichas sentencias podrán ser aquellas que se asocian de manera directa a los bloques funcionales de una aplicación, o palabras reservadas que actúan como delimitadores de código perteneciente a los subprogramas principal (SPP) y temporizado (SPT) en los que se divide una aplicación SIIL1; una forma sencilla de presentar esta separación entre los dos subprogramas es la que se ve aquí:



Existen algunos módulos, como ciertos tipos de temporizadores y los secuenciadores de estados, cuya declaración en el programa SIIL1 requiere de escribir más de una línea de texto; sin embargo, todas las sentencias mencionadas se apegan a formas sintácticas que pueden generalizarse como sigue:

CODM#N E1,…,En,…,S1,…,Sm, D1,…,Dq, CADBI;

en donde:

CODM es una cadena de caracteres que representa a la función que se desea efectúe un módulo.

N es un número que el usuario asigna a un ML en particular; esto se hace necesario porque todos los módulos de un mismo tipo deben de respetar una numeración, por lo cual no podrá haber módulos de la misma clase con un número de asignación repetido dentro de una aplicación.

E1,…,En es una lista de n variables binarias para designar las entradas que el módulo lógico requiere.

S1,…,Sm es una lista de m variables binarias para designar las salidas que el módulo lógico puede contener.

D1,…,Dq es una lista de datos auxiliares que son requeridos por algunas clases de módulos, estos datos pueden representar tiempos asociados a la duración de pulsos o el número de estados dentro de un secuenciador, entre otros casos. Para los módulos que requieren de datos auxiliares, q es un número comprendido entre 0 y 3; otros módulos, como las compuertas lógicas, no requieren de estas especificaciones.

CADBI es una cadena de datos binarios (unos y ceros) en la que se configuran características de funcionamiento de los módulos; dichas características pueden indicar el tipo de flanco que activa la respuesta de un módulo o cuáles de las entradas de una compuerta tendrán una negación implícita durante la ejecución.

Es importante aclarar que existen ciertas consideraciones que deben observarse a fin de evitar la generación de errores por parte del software traductor de código SIIL1, por ejemplo, el primer caracter de la instrucción nunca deberá estar en la primera columna y al final de la misma siempre ha de colocarse el caracter ";". Todo el texto a la derecha del caracter ";" no se toma en cuenta por el software generador del código objeto, de esta manera el usuario podrá colocar comentarios en el programa fuente; si se desea tener un renglón completo como comentario, simplemente se coloca en la primera columna del mismo ya sea el caracter ";" o bien el caracter "*".

Para fines ilustrativos, a continuación se presenta la declaración en SIIL1 correspondiente a una compuerta NAND de cuatro entradas que se desea sean las variables físicas E01, E02, E13 y E14; se requiere que las entradas no tengan preinversión y que la salida sea la variable intermediaria I101, además a esta compuerta se le asigna el número siete; la forma sintáctica asociada es:

NAND4#7 E01, E02, E13, E14, I101, 1111;

Para detalles acerca de las formas sintácticas para declarar cada uno de los módulos lógicos realizables con el PLM pueden consultarse [2] y [4].

2. ASPECTOS DEL V-PLM

El V-PLM, que se presenta en este trabajo como una opción didáctica orientada a reproducir el funcionamiento del PLM, cuenta con una serie de pantallas o interfaces gráficas para permitir al usuario la simulación de un programa fuente en código SIIL1. Inicialmente, la pantalla principal (figura 2) aparece frente al usuario, indicando con esto que el V-PLM está preparado para iniciar los subprocesos que preceden a la simulación.

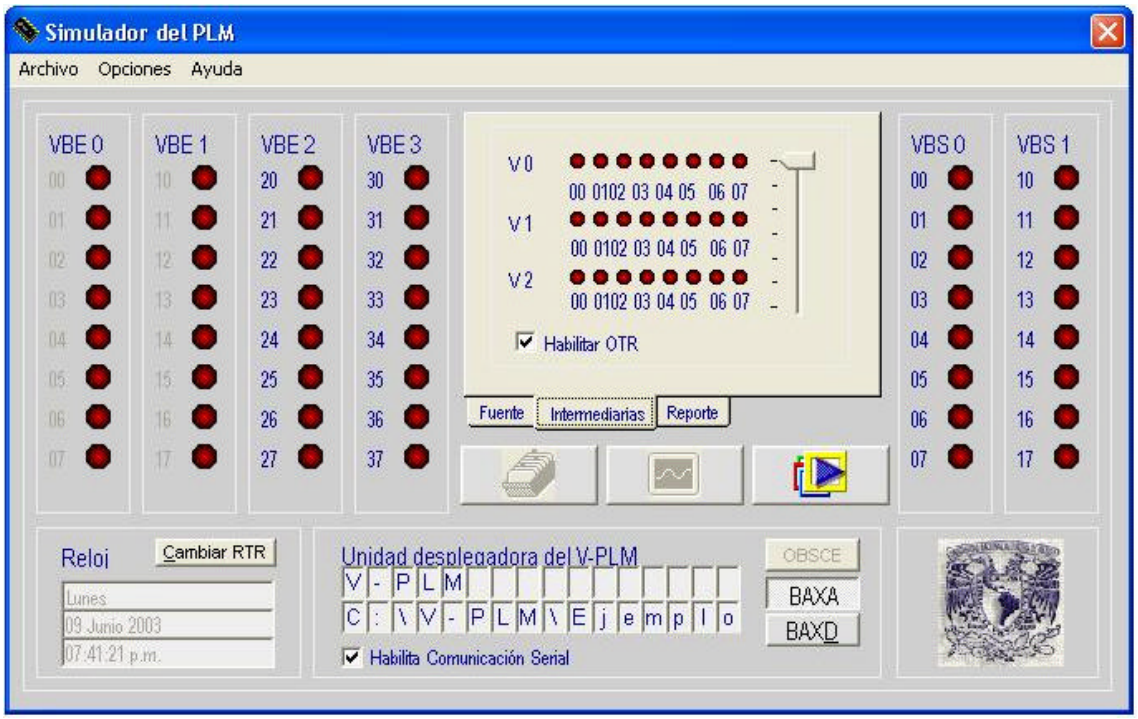


Fig 2. Pantalla principal del V-PLM.

El simulador contiene 32 variables de entrada, las cuales están agrupadas en conjuntos de 8 elementos por grupo. En ellas se registran los valores de entrada al sistema, y se representan por pequeños botones a través de los cuales el usuario es capaz de proporcionar el estado de las variables que requiere, ya sea nivel alto o bajo (uno o cero lógicos, respectivamente); cada uno de estos botones cuenta con un indicador correspondiente a la entrada especificada, en donde se puede testificar visualmente el estado de la misma; dicho estado cambia encendiendo o apagando esa entrada (haciendo click sobre el número de la entrada correspondiente), lo que ocasionará un cambio de color en el indicador (verde indica nivel alto, rojo indica nivel bajo).

Cuando se requiera hacer una simulación habilitando la característica de comunicación serial que se incluye en el V-PLM, los bytes 0 y 1 cambiarán de valor en el Panel principal de acuerdo a la manipulación que el usuario haga en dichos estados mediante la interfaz física diseñada con ese propósito. El cambio de estado de dichas variables se verá reflejado en los mismos indicadores dentro del Panel de la pantalla principal, pero no podrán modificarse sus estados generando eventos de click sobre éstos hasta que el usuario desverifique la casilla que activa la comunicación serial, localizada justo debajo de la Unidad Desplegadora (UD) del V-PLM, en la parte inferior de la pantalla. La interfaz física mencionada anteriormente se encarga de la recepción de los valores de entrada y el envío de valores de salida en y desde el Panel principal.

El simulador cuenta también con un bloque de 16 salidas, las cuales están representadas por 2 grupos de 8 elementos cada uno; en este caso sólo existen indicadores visuales para testificar cuál es el estado de estas variables, ya sea nivel alto o bajo, de la misma manera como se hace con las entradas, sin embargo, sus estados no pueden ser modificados directamente por el usuario, lo que significa que los valores almacenados en estas variables sólo se verán alterados durante la ejecución de las aplicaciones de control lógico.

2.1 Botones de ejecución

La interfaz gráfica cuenta con 4 botones principales con los que se lleva a cabo, paso a paso, la labor de simulación, éstos son: *Abre archivo SIL, Depurar archivo SIL, Iniciar la simulación* y *Detener la simulación.* Al presionarse este último botón, la ejecución de los ML se detiene, y el botón *Iniciar la simulación* vuelve a ser visible, repitiéndose este proceso

cada vez que el usuario detenga o reinicie una simulación en el V-PLM.

2.2 Bloque de visualización

Existen tres pantallas de visualización contenidas en el simulador: *Fuente*, *Intermediarias* y *Reporte*, en las que se presenta información relacionada con el programa fuente que se ejecutará en el V-PLM. Dichas pantallas se describen brevemente a continuación

Fuente

Mediante esta pantalla, el usuario puede visualizar el código fuente del archivo SIL que se está simulando o que está próximo a evaluarse.

Intermediarias

La pestaña *Intermediarias* contiene 7 bloques de 3 grupos cada uno, representando a las variables internas del V-PLM; análogamente a los grupos de variables de salida, sus valores no pueden ser modificados directamente por el usuario. Un total de 168 variables intermediarias se organiza en 21 grupos de 8 elementos cada uno, y se pueden visualizar en bloques de 3 grupos variando la posición de la barra deslizable que con ese fin se ha colocado a la derecha de la ventana. Mediante la inspección de este bloque se puede testificar cómo se va modificando el valor de las variables intermediarias, de acuerdo a los resultados de la ejecución de los módulos lógicos dentro del programa SIL activo en el simulador.

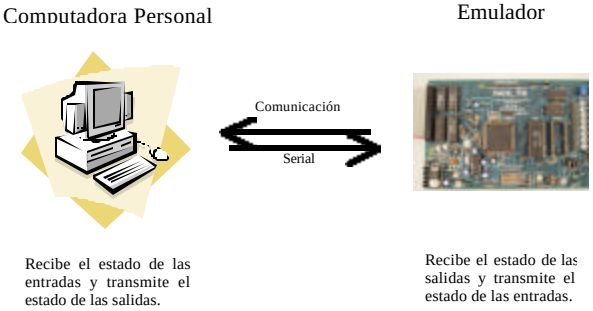
Debajo del bloque que contiene a las variables intermediarias, existe una casilla de verificación denominada *Habilitar OTR*, que habilita o deshabilita la observación de las variables en tiempo real (OTR); esto permite al usuario ser testigo del cambio en los valores de este tipo de variables en plena ejecución.

La sección de *Reporte de Simulación* contiene un resumen de la información generada por el V-PLM sobre los tiempos registrados en cada ciclo de ejecución de los módulos lógicos contenidos en la aplicación SIIL1 al término de cada simulación.

Es posible conocer más detalles del proceso por medio de la generación de un reporte que muestra tanto los datos contenidos en el visor de la pantalla *Reporte*, como el estado final de las variables del V-PLM, además de algunas referencias sobre las opciones de simulación que el usuario está empleando. Al reporte señalado puede accederse pulsando el botón *Imprimir Reporte*, localizado en la parte inferior del visor aquí descrito, con el cual el usuario podrá imprimir los datos mostrados o simplemente inspeccionar los datos generados.

2.3 Interfaz física

Una vez que se tiene la herramienta de simulación, se proporciona la interfaz con la que el usuario puede experimentar físicamente con 16 variables de entrada, 16 de salida y continuar empleando las variables intermediarias, así como seguir operando las restantes 16 variables de entrada por medio del Panel. Para esto se incluye un elemento de hardware (emulador), que se comunica de forma serial con la PC y que en tiempo real (menos de 10 [ms]) cambia el estado de las entradas físicas asociadas, así como también establece en sus salidas valores de voltaje que representan el estado de las 16 variables de salida asociadas con la interfaz física.



El emulador fue desarrollado sobre la misma base tecnológica del PLM y se constituye fundamentalmente de la tarjeta FACIL_11B, diseñada por el primer autor de este documento, cuyo elemento central es el microcontrolador 68HC11F1 de la compañía Motorola.

La transmisión de las entradas se hace “por poleo”, es decir, que constantemente se están enviando los valores de las entradas en la interfaz física hacia la PC, mientras que las salidas se reciben por servicio de interrupción. Cada vez que la PC envía el valor de la salida hacia el microcontrolador, el flujo de ejecución “salta” a la rutina que discrimina entre los dos grupos de salidas disponibles y las muestra. El código en lenguaje ensamblador se encuentra almacenado en la memoria del microcontrolador, para permitir la ejecución autónoma de la aplicación dedicada al intercambio de entradas y salidas entre la interfaz física y su contraparte ejecutable en la PC.

3. EJEMPLO DE APLICACIÓN

El presente ejemplo está dirigido a ilustrar el uso del V-PLM en una línea de producción industrial de procesos en serie que se efectúan sobre un producto en particular. Supóngase que se desea administrar el flujo de dichos procesos de ensamble en una fábrica automotriz, los cuales ocurren a través de una línea de producción que requiere ser detenida cada vez que el vehículo que se está armando se sitúa en un lugar determinado, para que algún operario o maquinaria especializados ejecuten una tarea en particular en el automóvil. Cuando el vehículo pase por cada sensor, la línea se detendrá cierto tiempo a fin de poder cumplir con las tareas especificadas en la tabla siguiente:

Sensor	Proceso	Retraso [s]
1	Pintado de carrocería	30
2	Secado de carrocería	25
3	Colocación del tablero de instrumentos	15

Tabla 1. Procesos de la línea de ensamble.

Además de lo anterior, el sistema debe ser capaz de tomar en cuenta el estado de un bus de comunicaciones formado por un par de cables provenientes desde un PLC anterior, e informar por medio del mismo bus a un PLC posterior el estado en que se encuentra, a fin de monitorear el tránsito de los vehículos de una línea de producción a otra. La siguiente tabla contiene el significado de los mensajes que componen este código de señalización entre las líneas de ensamble, así como la acción a tomar:

Estado	Significado	Acción
00	No habilitado	Parar banda y mostrar mensaje
01	No hay insumo	Parar banda y emitir alarma
10	Atasco	Parar banda y emitir alarma
11	Producción normal	Continuar y mostrar mensaje

Tabla 2. Sistema de señalización en la línea de ensamble.

Para implantar el control lógico requerido se emplearon cinco variables de entrada del V-PLM, dos asociadas con un bus de retroaviso (E06 y E07), y las tres restantes asociadas con sendos sensores de presencia. La aplicación requiere el empleo de tres salidas físicas, una asociada con la activación de la línea (S00) y las otras dos asociadas a lámparas testigo (S06 y S07).
De acuerdo a las condiciones planteadas, en el siguiente diagrama se muestra una posible solución al circuito lógico de control que llevará a cabo las tareas solicitadas:

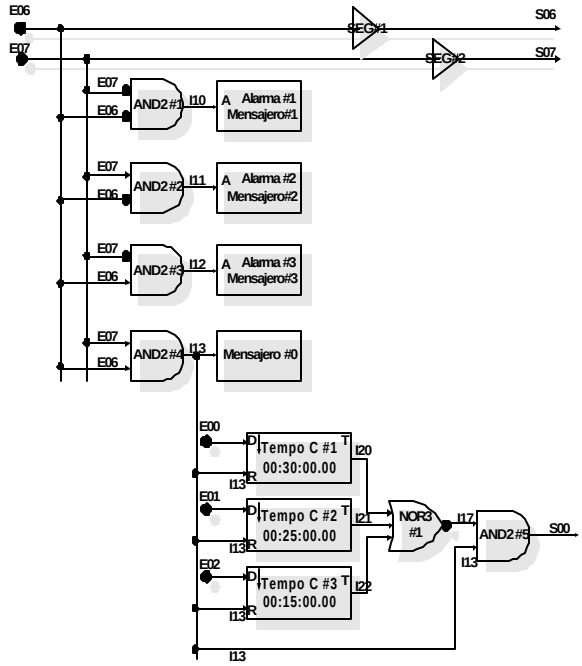


Fig. 3. Diagrama lógico planteado para la línea de ensamble.

Según se observa en la figura 3, los seguidores a las entradas E06 y E07 se emplean como señales de retroaviso para comunicar al PLC actual el estado de otro PLC anterior dentro del proceso de producción, dichas entradas se relacionan con el bus de señalización descrito en la tabla 2, y se conectan a un bloque de compuertas AND de 2 entradas que se encargan de discriminar los estados presentes en el bus. Las salidas de estas compuertas se conectan a su vez a módulos de Alarma que activan Mensajeros para desplegar mensajes que alerten al operador sobre algún problema en la línea de ensamble. Cuando la producción es normal, la compuerta AND2 número 4 activa la salida I13 en nivel alto, y la enlaza a la compuerta AND2 número 5 como una condición para mantener al motor en marcha. Dentro del orden de la línea, se asignan las variables de entrada E00, E01 y E02 al primer, segundo y tercer sensor de presencia, respectivamente. Cada variable se conecta a la entrada de disparo del temporizador One Shot (TempoC) que le corresponde, mientras que la salida I13 mencionada anteriormente se conecta a la entrada de RESET de los 3 temporizadores empleados y sus salidas se introducen a una compuerta NOR de 3 entradas; esta compuerta enviará al motor la señal de paro durante el periodo indicado por el temporizador que se

activó. Si la salida I17 de la compuerta NOR se verifica, la compuerta AND2 número 5 colocará un nivel bajo en su salida para detener el motor; por otro lado, si I17 no se verifica e I13 continúa verificado (es decir, no se presentan los problemas descritos en la tabla 2), el motor de la banda electromecánica en la línea continuará en funcionamiento, puesto que su salida asociada S00 quedará verificada.

Para implementar esta solución empleando al V-PLM, es necesario hacer las declaraciones apropiadas de los módulos lógicos ilustrados en la figura 3; dichas declaraciones pueden agruparse como se ve en el listado 1 en los subprogramas principal y temporizado de un programa fuente en código SIIL1.

Ensamble.SIL

```
CONFIG1;                          *CONFIGURACIÓN DE FUNCIONAMIENTO DEL PLM
INPROG;                           *INICIO DE SUBPROGRAMA PRINCIPAL
DESP;                             *HABILITACIÓN DE LA UD DEL V-PLM O DEL PLM

SEG#1 E06, S06;                   *SEGUIDORES A LAS VARIABLES DE ENTRADA ASOCIADAS
SEG#2 E07, S07;                   *A LOS SENSORES DE LA LÍNEA DE PRODUCCIÓN

AND2#1 E06, E07, I10, 00;         *BLOQUE DE COMPUERTAS LÓGICAS PARA DISCRIMINAR LOS
AND2#2 E06, E07, I11, 10;         *ESTADOS PRESENTES EN LAS ENTRADAS DE LOS SENSORES
AND2#3 E06, E07, I12, 01;
AND2#4 E06, E07, I13, 11;
AND2#5 I17, I13, S00, 11;

NOR3#1 I20, I21, I22, I17, 111;   *COMPUERTA LÓGICA PARA HABILITAR EL FUNCIONAMIENTO DE LA
                                  *LÍNEA DE PRODUCCIÓN
ALARMA#1 I10, 1, 1;               *ALARMA ASOCIADA AL MENSAJE DE ESPERA DEL VEHÍCULO,
                                  *PROVENIENTE DE OTRA LÍNEA DE PRODUCCIÓN ANÁLOGA (NO HABILITADO)
ALARMA#2 I11, 2, 1;               *ALARMA ASOCIADA AL MENSAJE DE ESPERA DE INSUMO PARA
                                  *CONTINUAR CON LA PRODUCCIÓN, DETIENE EL MOVIMIENTO DE LA
                                  *BANDA (SIN INSUMO)
ALARMA#3 I12, 3, 1;               *ALARMA ASOCIADA AL MENSAJE DE ATASCO EN LA LÍNEA DE PRODUCCIÓN,
                                  *QUE NECESARIAMENTE DETIENE EL MOVIMIENTO DE LA BANDA (ATASCO)

FINPP;                                    *FIN DE SUBPROGRAMA PRINCIPAL

INMODI;                                   *INICIO DE SUBPROGRAMA TEMPORIZADO
MANDESP 4;
TEMPOC#1 E00, I13, I20, 00:00:06.00, 001; *TEMPORIZADOR QUE DETIENE 6[s] EL TRÁNSITO DEL PRODUCTO
                                          *PARA EL PROCESO DE PINTADO DE CARROCERÍA
TEMPOC#2 E01, I13, I21, 00:00:10.00, 001; *TEMPORIZADOR QUE DETIENE 10[s] EL TRÁNSITO DEL PRODUCTO
                                          *PARA EL PROCESO DE SECADO DE CARROCERÍA
TEMPOC#3 E02, I13, I22, 00:00:05.00, 001; *TEMPORIZADOR QUE DETIENE 6[s] EL TRÁNSITO DEL PRODUCTO
                                          *PARA LA COLOCACIÓN DEL TABLERO DE INSTRUMENTOS

   MENSAJERO#1 "NO HABILITADO", 01, 16, 15, 1001; *MENSAJE ASOCIADO A LA ALARMA # 1 (NO HABILITADO)
# sistema no habilitado, esperando la señal de línea de|
## producción anterior.

   MENSAJERO#2 "SIN INSUMO", 01, 16, 015, 1001;    *MENSAJE ASOCIADO A LA ALARMA #2 (SIN INSUMO)
# Insumo de producción no adecuado o en cantidad diferente a|
## la requerida, favor de revisar paso 1 de producción.

   MENSAJERO#3 "ATASCO", 01, 16, 015, 1001;        *MENSAJE ASOCIADO A LA ALARMA #3 (ATASCO)
# Obstrucción en la línea.|
## Favor de revisar.

   MENSAJERO#0 "OP. NORMAL", 01, 16, 015, 1001;    *MENSAJE TESTIGO DE OPERACIÓN NORMAL
## Producción Normal.

FINMODI;                                  *FIN DE SUBPROGRAMA TEMPORIZADO
```

Listado 1. Código SIIL1 para el ejemplo.

El código del listado anterior se analiza, depura y simula con los elementos de las interfaces dispuestas en el V-PLM [4], con lo cual se ejemplifica el uso de este software en el control de los procesos de producción en serie que se utilizan en diversas industrias. Es importante aclarar que por simplicidad se consideró que la línea de ensamble se encuentra ocupada por un solo vehículo a la vez, desde el inicio del recorrido hasta su fin, liberando la línea después de que se ha completado el tiempo total de duración de los tres procesos requeridos, además de los retrasos por la existencia de problemas, si éstos se presentan; una vez transcurrido dicho tiempo, un nuevo producto puede ser colocado dentro de la línea de ensamble para ser atendido.

La ventaja que presenta la solución implementada por medio del V-PLM se refleja en que, a pesar de las modificaciones en los requerimientos del problema, el mantenimiento y la adecuación del código en el programa fuente a las nuevas condiciones se hace rápida y fácilmente, admitiendo impactos mínimos en el desempeño final, como podría ser debido a la presencia de módulos adicionales, lo que no representa mayor problema para el usuario gracias al diseño del lenguaje SIIL1 y a la funcionalidad en la estructura del software emulador del PLM.

Para mayores detalles sobre el manejo y características adicionales del V-PLM puede consultarse [4].

4. CONCLUSIONES

Con el V-PLM se pueden generar reportes de apoyo y lograr resultados confiables, cuando se le utiliza en la resolución de problemas en el campo académico, brindando al mismo tiempo una aproximación al ámbito de la industria en la cual se presentan comúnmente estas situaciones; por lo anterior, el producto mencionado puede servir como auxiliar didáctico de entrenamiento para los profesores y alumnos en los laboratorios y prácticas de las asignaturas relacionadas a este campo del control lógico. Tiene además posibilidades de crecimiento, ya que su estructura es flexible a cambios y adecuaciones, pudiendo actualizar sus características según se requiera para extender el periodo de vida útil del sistema.

5. REFERENCIAS

[1] L. M. Ortega y A. Salvá, “PROGRAMADOR LÓGICO MODULAR”, Chihuahua, Chih, Memoria de ELECTRO 98, octubre de 1998.

[2] A. Salvá “PROGRAMADOR LÓGICO MODULAR”, México D. F., Tesis de maestría, División de Estudios de Posgrado, Facultad de Ingeniería, UNAM, febrero de 1999.

[3] J. Webb, PROGRAMMABLE LOGIC CONTROLLERS, Principles and Applications, Merril, 1992.

[4] L. Altamirano, M. Hernández, E. Dehesa, “DESARROLLO DE SOFTWARE DE SIMULACIÓN PARA EL PROGRAMADOR LÓGICO MODULAR (PLM)”, México, D. F., Tesis de licenciatura, Facultad de Ingeniería, UNAM, Mayo de 2003.