

## Actividades y ejercicios prácticos de programación con CPU S7-300

### EJERCICIO Nº 1. Descarga de Manuales Siemens. Conexión y Revisión del cableado de las CPUs.

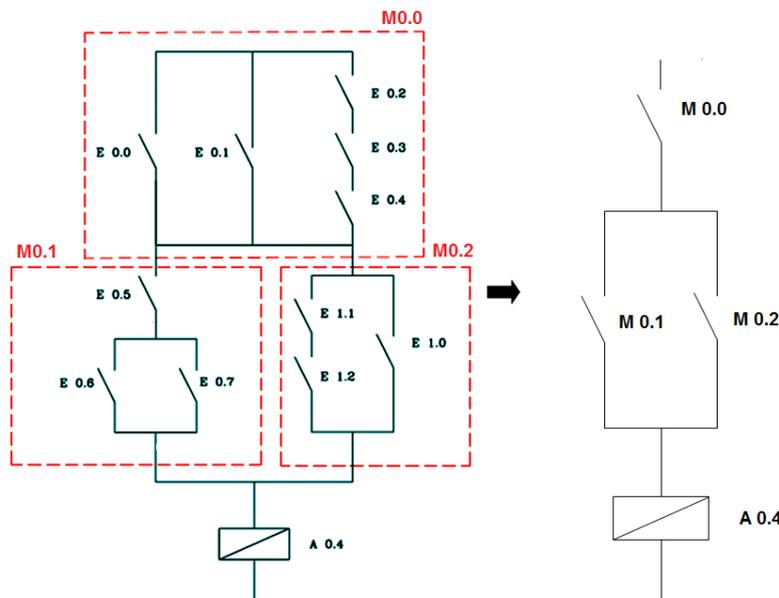
1. Identifica las CPU S7 de Siemens disponibles en el aula: Tipo y Modelo de CPU, módulos de E/S, tarjetas, etc.
2. Visita la página web de Siemens y descarga los manuales de cada CPU y Módulos de E/S disponibles. Descarga también los manuales de Step7 y Tia Portal. Enlace a la web: <https://www.siemens.com/es/es/home.html>
3. Realiza y/o comprueba el conexionado del cableado de las CPUs y del entrenador de prácticas. Prepara y/o supervisa las conexiones RJ45 necesarias para la comunicación con las CPU S7-1200 y S7-1500, así como las del Switch de comunicaciones.
4. Alimenta el entrenador de prácticas y comprueba el funcionamiento de las CPUs.
5. Haz un esquema eléctrico completo del conexionado de cada una de las CPU del entrenador.

### EJERCICIO Nº 2. Configuración de una CPU S7-300 y Simulación con PLCSIM con STEP7

1. Configura en el Administrador de Simatic Step7 un proyecto que incluya una Fuente de alimentación PS 307 2A y una CPU 313C-2DP o una CPU 314C-2DP disponibles en el aula. (dirección CPU: 2)
2. Abre el editor de símbolos (*HWConfig -> Herramientas-> Tabla de símbolos*) y crea una tabla con los nombres indicados en el punto 3.
3. Programa el bloque OB1 de forma que nos permita leer las 5 primeras entradas digitales (*símbolos: entrada 1, etc.*) y las muestre en sus salidas (*símbolos: salida 1, etc.*) cuando se active la entrada E0.6 (*símbolo: marcha*). Las salidas han de borrarse todas tras pulsar la entrada E0.7 (*símbolo: borrado*)
4. Inicia el simulador **PLCSIM**, carga el proyecto en él y configúralo para visualizar el estado de las variables de E/S del programa.
5. Desde la ventana de programación del OB1, crea una tabla de variables (*sistema de destino -> observar/forzar variable*), que nos permita conocer en tiempo real el estado de las variables del programa.
6. Activa el icono **observar o "gafas"**  tanto en la ventana de tabla de variables, como en el programador del OB1 para conocer en tiempo real el estado de las variables y comprobar la activación de los contactos.

### EJERCICIO Nº 3. Programación en lenguajes KOP, AWL y FUP. Trabajo con Marcas. Con STEP7

1. Configura en el Administrador de Simatic Step7 un proyecto que incluya una Fuente de alimentación PS 307 2A y una CPU 313C-2DP o una CPU 314C-2DP disponibles en el aula. (dirección CPU: 2)
2. Programa el bloque OB1 en los lenguajes AWL, KOP y FUP para resolver el circuito de la figura utilizando las marcas indicadas.
3. Utiliza el simulador PLCSIM para comprobar el funcionamiento del programa.



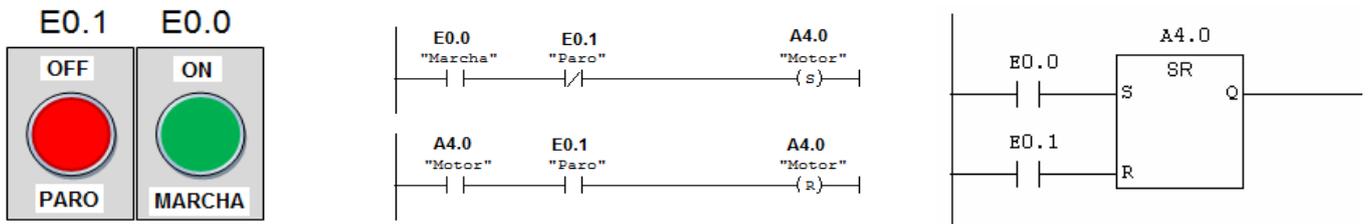
### EJERCICIO Nº 4. Programación del Bloque de Organización OB100 con STEP 7

1. Configura en el Administrador de Simatic Step7 un proyecto que incluya una Fuente de alimentación PS 307 2A y una CPU 313C-2DP o una CPU 314C-2DP disponibles en el aula. (dirección CPU: 2).
2. Programa el bloque de organización OB100 de manera que al pasar el PLC de modo STOP a RUN se deben poner activas las salidas A124.1 y A124.2. Así mismo, la marca M0 debe fijarse al valor 231 (valor decimal).
3. Programa ahora el bloque de organización OB1 de manera que se muestre el valor del byte de marcas M0 en el byte de salidas AB125 cuando la entrada E124.1 se active. El contenido del byte AB125 debe borrarse al activar la entrada E124.2.
4. Conecta el cable MPI de programación, Ajusta y configura la "interface PG-PC" y realiza los ajustes necesarios para comunicar el PLC con el PC.
5. Carga el proyecto en la CPU y comprueba su funcionamiento.

### EJERCICIO Nº 5. Programación con instrucciones SET, RESET y SR con S7-300 en STEP 7

Se desea hacer un enclavamiento eléctrico, de manera que con dos pulsadores, uno de marcha (E0.0) y otro de paro (E0.1), se active o se desactive un motor (A4.0).

1. Programa el OB1 utilizando instrucciones SET, RESET y SR en el Administrador STEP 7.
2. Comprueba el funcionamiento con el software de simulación PLCSIM y en la CPU.



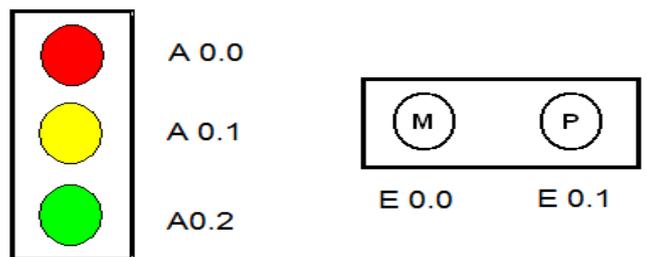
### EJERCICIO Nº 6. Trabajo con temporizadores.

Configura un proyecto que permita leer las 8 primeras entradas de la CPU y las muestre en sus salidas cuando al activar la entrada E1.0 transcurra un tiempo de 5 segundos. Si accionamos E1.1 las salidas deben permanecer a 0.

SOLUCIÓN EN KOP	SOLUCIÓN EN AWL	SOLUCIÓN EN FUP
	<pre> OB1 : "Main Program Sweep (Cycle)" Segm. 1 : Título: U   E   1.0 L   SST#5S SS  T   1 U   E   1.1 R   T   1 NOP 0 NOP 0 U   T   1 =   M   0.0  Segm. 2 : Título: U   M   0.0 SPBNB _001 L   EB  0 T   AB  0 _001: NOP 0  Segm. 3 : Título: U   E   1.1 SPBNB _002 L   0 T   AB  0 _002: NOP 0                     </pre>	

### EJERCICIO Nº 7. Control de un semáforo.

Tenemos un semáforo con las tres luces: verde, ámbar y rojo. Disponemos también de dos pulsadores de mando: un pulsador de marcha y un pulsador de paro. El ciclo comienza tras pulsar el pulsador de marcha, siguiendo la siguiente secuencia indicada. El ciclo es repetitivo hasta que se pulse el pulsador de paro. En ese momento se apaga todo. Siempre que se active el pulsador de marcha queremos que el sistema empiece por el verde.



**SECUENCIA:** 1º/ Verde durante 5 seg.      2º/ Verde + Amarillo durante 2 seg.      3º/ Rojo durante 6 seg

1. Resuelve el ejercicio programándolo en KOP.
2. Simula el proceso con **PLCSIM** y **SIMIT SCE**.
3. Carga el programa en la CPU física y trata de comunicar con el software **SIMIT SCE** para visualizar y comprobar el funcionamiento del programa.

## Solución en AWL

```

U      E      0.0    //Al activar el pulsador de marcha
S      A      0.2    //Encender el verde
U      A      0.2    //Si se ha encendido el verde

L      S5T#5S      //Cuenta 5 segundos
SE     T      1     //Con el temporizador 1
U      T      1     //Y cuando acabes de contar
S      A      0.1   //Enciende el amarillo
U      A      0.1   //Si se ha encendido el amarillo

L      S5T#2S      //Cuenta 2 segundos
SE     T      2     //Con el temporizador 2
U      T      2     //Y cuando acabes de contar
S      A      0.0   //Enciende el rojo
R      A      0.1   //Apaga el amarillo
R      A      0.2   //Y apaga el verde
U      A      0.0   //Si se ha encendido el rojo

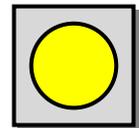
L      S5T#6S      //Cuenta 6 segundos
SE     T      3     //Con el temporizador 3
U      T      3     //Cuando acabes de contar
S      A      0.2   //Enciende el verde
R      A      0.0   //Y apaga el rojo
U      E      0.1   //Si se activa el pulsador de paro
R      A      0.0   //Apaga el rojo
R      A      0.1   //Apaga el amarillo
R      A      0.2   //Apaga el verde

```

## EJERCICIO Nº 8. Intermitente de 1 segundo.

Queremos programar un intermitente utilizando un solo temporizador de 1 segundo. Queremos que la lámpara esté activa un segundo y no activa otro segundo. Queremos que haga esto sin ninguna condición previa.

### Solución en KOP:

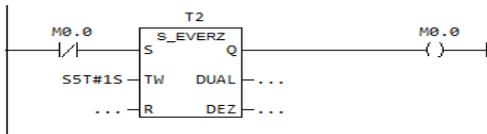


OB1 : LÁMPARA INTERMITENTE 1 SEGUNDO

Comentario:

Segm. 1 : Título:

Programamos T1 a un segundo



Segm. 2 : Título:

Mientras M0.0 esté desactivada salta al segmento 4. Esto provoca un bucle continuado de 1 segundo.



Segm. 3 : Título:

Activa la lámpara si esta apagada.



Segm. 4 : Título:

Este final no sirve para nada. Solo para el sarto del programa.



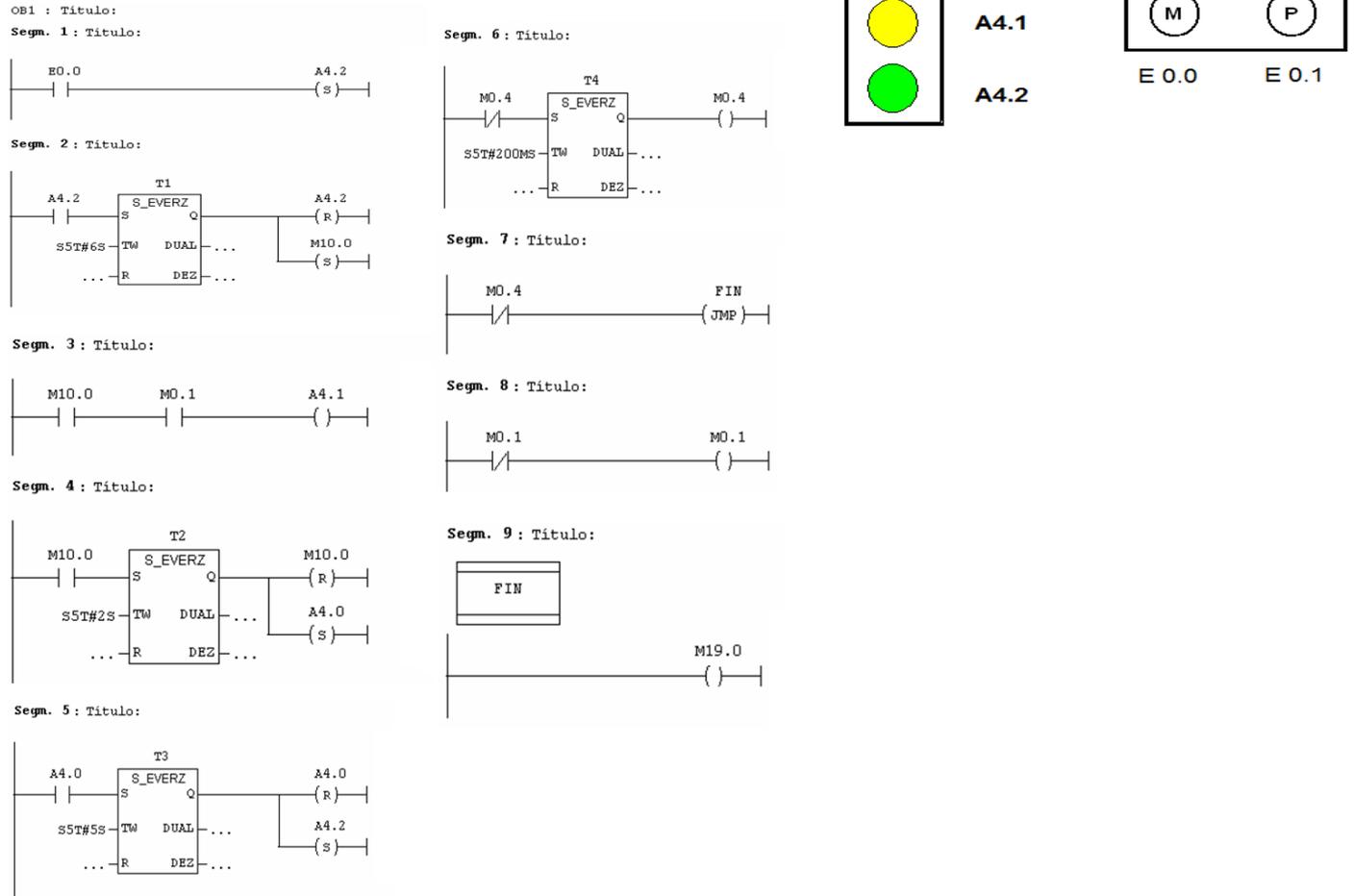
## EJERCICIO 9. Control de un semáforo con luz amarilla intermitente (Temporizadores y Saltos)

Tenemos un semáforo con las tres luces verde, ámbar y rojo. Disponemos también de dos pulsadores de mando: un pulsador de marcha y un pulsador de paro. El ciclo comienza tras pulsar el pulsador de marcha, siguiendo la siguiente secuencia indicada. El ciclo es repetitivo hasta que se pulse el pulsador de paro. En ese momento se apaga todo. Siempre que le dé al pulsador de marcha quiero que empiece por el verde.

**SECUENCIA:** 1º/ Verde durante 5 seg. 2º/ Amarillo intermitente durante 2 seg. 3º/ Rojo durante 6 seg.

Como puede verse, el funcionamiento es idéntico al del ejercicio nº 10, pero con el ciclo modificado. Aquí, la luz amarilla queda intermitente.

### Solución en KOP



## EJERCICIO 10. Control de un Parking. (Contadores y Comparadores)

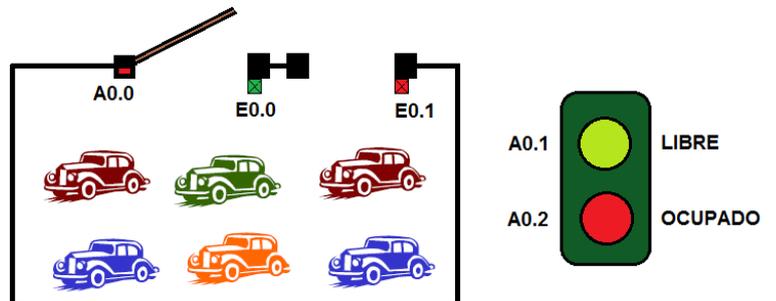
El funcionamiento de un parking queremos que sea el siguiente:

Cuando llega un coche y el parking esté libre, queremos que se abra la barrera. A la salida no tenemos barrera. Cuando sale un coche simplemente sabemos que ha salido.

En el parking caben 10 coches. Cuando el parking tenga menos de 10 coches queremos que esté encendida la luz de libre. Cuando en el parking haya 10 coches queremos que esté encendida la luz de ocupado.

Además, queremos que si el parking está ocupado y llega un coche que no se abra la barrera.

Carga el programa en la CPU física y comunica con el software **SPS VISU** para visualizar y comprobar el funcionamiento del programa.



OBI : Parking de Coches

Comentario:

Segm. 1 : Título:

Si llega un coche y el parking está libre, abrimos la barrera

```

U "Sensor Entrada" E0.0
U "Libre" A0.1
= "Barrera" A0.0
    
```

Segm. 2 : Título:

Contamos y descontamos el nº de coches

```

U "Barrera" A0.0
ZV Z 1
U "Sensor Salida" E0.1
ZR Z 1
NOP 0
NOP 0
NOP 0
NOP 0
L Z 1
T MW 10
NOP 0
NOP 0
    
```

Segm. 3 : Título:

Si el nº de coches es inferior a 10 encendemos la luz verde de "Libre".

```

L MW 10
L 10
<I
= "Libre" A0.1
    
```

Segm. 4 : Título:

Si no está libre, activamos la luz de ocupado.

```

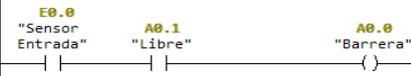
UN "Libre" A0.1
= "Ocupado" A0.2
    
```

OBI : Parking de Coches

Comentario:

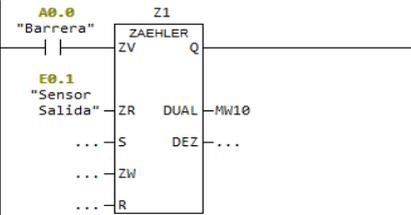
Segm. 1 : Título:

Si llega un coche y el parking está libre, abrimos la barrera



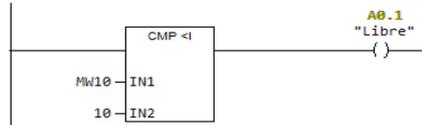
Segm. 2 : Título:

Contamos y descontamos el nº de coches



Segm. 3 : Título:

Si el nº de coches es inferior a 10 encendemos la luz verde de "Libre".



Segm. 4 : Título:

Si no está libre, activamos la luz de ocupado.

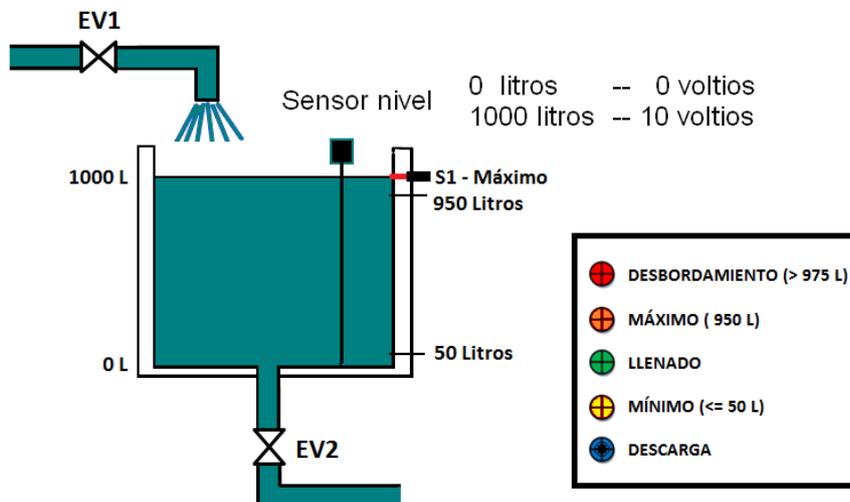


**EJERCICIO Nº 11. Trabajo con E/S analógicas y Comparadores.**

Programa una CPU S7-300 de forma que permita la lectura de dos entradas analógicas entre 0 y 10 V. El sistema activará un motor de un ventilador cuando la temperatura registrada en ambos sensores sea igual o superior a 40 ºC.

**EJERCICIO Nº 12. Escalado y desescalado de valores analógicos. Funciones especiales.**

Configura un proyecto que permita controlar el siguiente sistema:



Se desea supervisar y controlar el llenado y vaciado de un depósito de manera que la electroválvula EV1 de llenado se active cuando a éste le queden tan sólo 50 litros, y que se desactive cuando tenga 950 litros. Para la medida de nivel líquido se dispone de un sensor analógico que entrega una señal entre 0 y 10 V para los niveles mínimo (0 Litros) y máximo (1000 Litros) respectivamente.

El sistema dispone de varias luces de señalización que indicaran en todo momento la situación del sistema:

L1 – Desbordamiento (> 975 litros).

L2 – Nivel máximo (950 L).

L3 – Llenado.

L4 – Nivel Mínimo ( $\leq$  50 litros)

L5 - Descarga

Para evitar un posible desbordamiento del depósito y como medida de seguridad, la electroválvula EV1 deberá cerrarse si el depósito alcanza un nivel igual o superior a 975 litros, hecho que será indicado mediante la luz de desbordamiento (L1). El sensor S1 nos servirá como apoyo al sensor de medida de nivel en caso de que este falle.

La electroválvula EV2 de vaciado, tiene un caudal máximo de 5 litros/segundo y es regulable entre 0-10 V. Ésta se abrirá para suministrar 50 litros cada vez que activemos el pulsador de descarga. Con objeto de aseguramos una descarga eficaz, la electroválvula EV2 no podrá abrirse si el depósito tiene menos de 75 litros de líquido.

Para realizar el programa se deberán utilizar obligatoriamente las funciones de escalado y desescalado de valores analógicos necesarias (funciones FC105 y FC106).

Carga el programa en la CPU física y comunica con el software **SPS VISU** para visualizar y comprobar el funcionamiento del programa.