

# PROGRAMACIÓN

En el documento de programación vamos a exponer nuestro trabajo realizado en la parte del programa del PLC de Omron escogido para este proceso, CJ2M-CPU31. Se ha utilizado 2 tipos de lenguajes de programación, el texto estructurado y el diagrama de bloques de función, ambos explicados en el documento principal de nuestro trabajo.

A continuación, exponemos en la Figura 1 el entorno de programación del software de Omron CX-Programmer para el PLC utilizado.

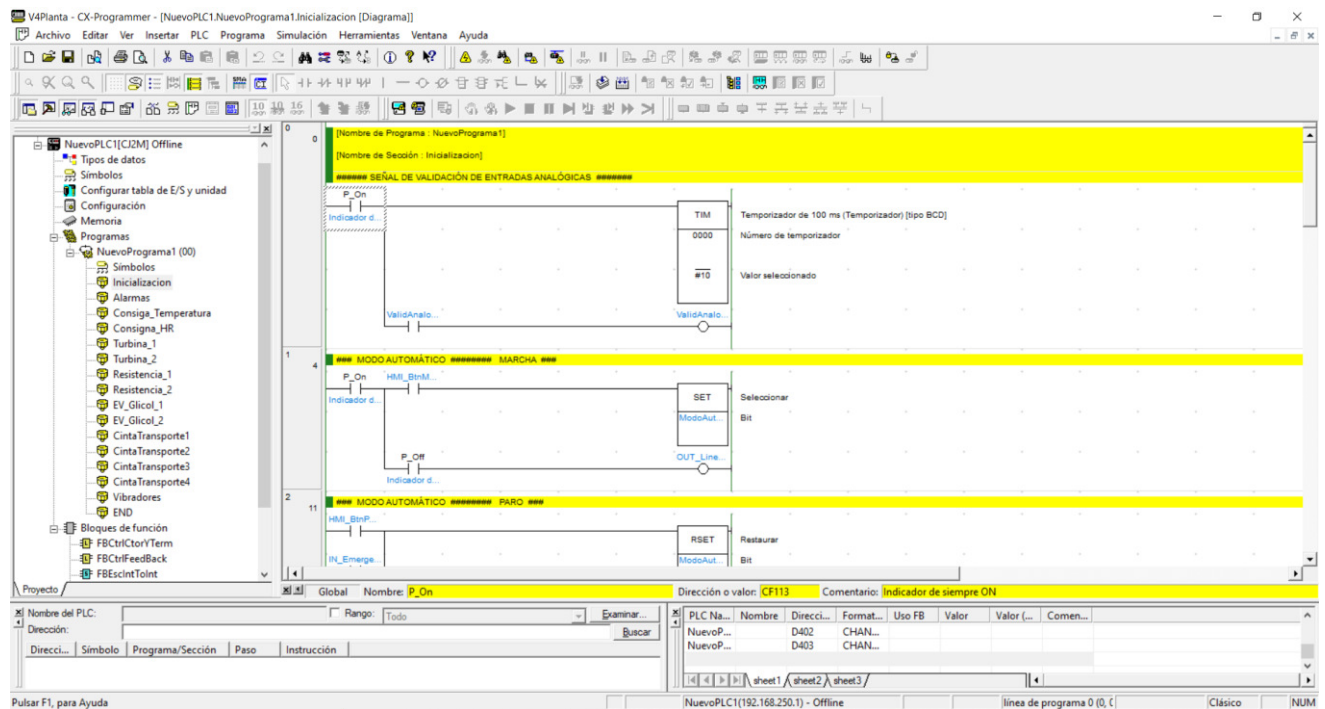


Figura 1 Entorno de programación

En la parte superior tenemos las barras de herramientas, se pueden modificar los iconos que queremos tener visibles y acceso directo a ellos. En el cuadro de la izquierda encontramos el menú de configuración principal del proyecto, según la pestaña seleccionada se nos abrirá una ventana emergente de configuración del PLC hasta llegar a la parte del programa, que dividiremos entre secciones y bloques de función. Conforme navegamos por las secciones que contienen estos 2 últimos se nos abrirá en el cuadro de la derecha la parte del programa seleccionada para modificar o revisar. Por último, en la parte inferior se indica toda la información del elemento seleccionado en el programa.

Seguidamente detallamos el programa que hemos creado, con sus los bloques de función utilizados y cada sección en la que hemos dividido el programa

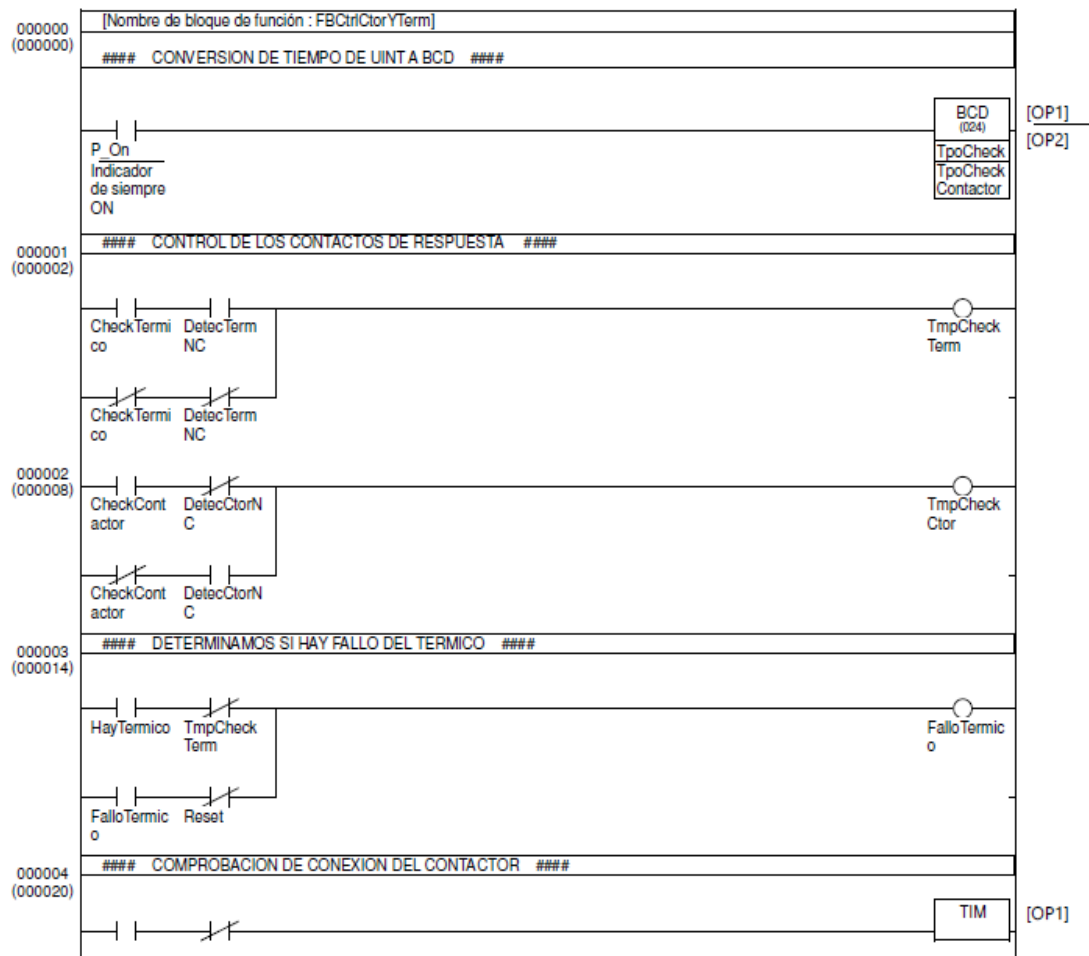
# ÍNDICE

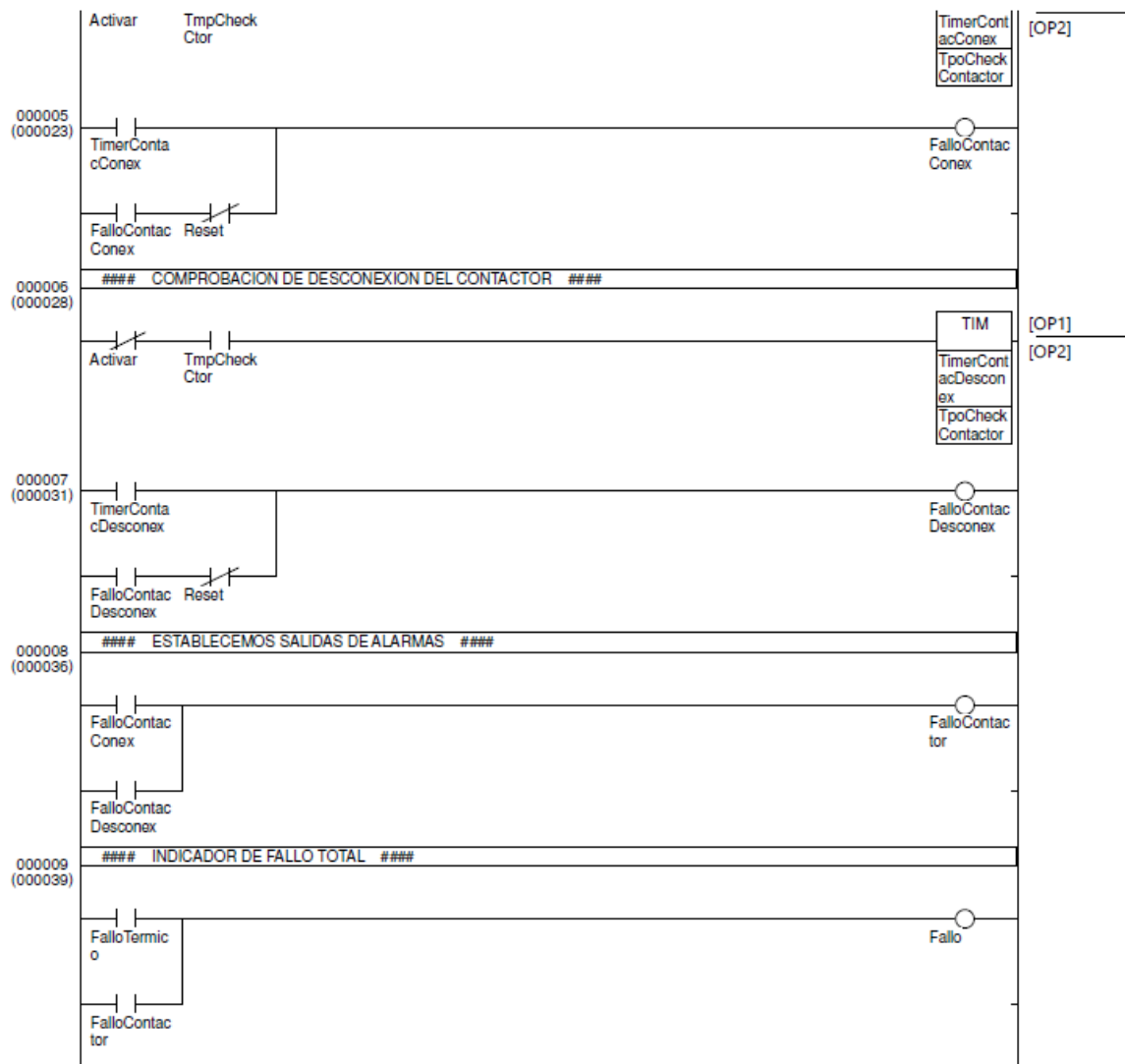
1.	BLOQUES DE FUNCION .....	3
1.1	Control contactores y térmicos. ....	3
1.2	Feedback. ....	5
1.3	Escalar Int a Int. ....	7
1.4	Escalar Int to Real. ....	8
1.5	Escalar Real to Int. ....	9
1.6	Histéresis de máximos a Real. ....	10
1.7	Histéresis de mínimos a Real.....	10
1.8	Gestión colores baliza. ....	11
2.	SECCIONES PROGRAMA.....	12
2.1	Inicialización. ....	12
2.2	Alarmas .....	14
2.3	Consigna de temperatura.....	17
2.4	Consigna humedad .....	23
2.5	Turbina 1 .....	25
2.6	Turbina 2 .....	26
2.7	Resistencia 1.....	27
2.8	Resistencia 2.....	28
2.9	Electroválvula 1 .....	29
2.10	Electroválvula 2 .....	31
2.11	Cinta transporte 1.....	33
2.12	Cinta transporte 2.....	35
2.13	Cinta transporte 3.....	37
2.14	Cinta transporte 4.....	39
2.15	Vibradores .....	41
2.16	Fin .....	42

# 1. BLOQUES DE FUNCION

## 1.1 Control contactores y térmicos.

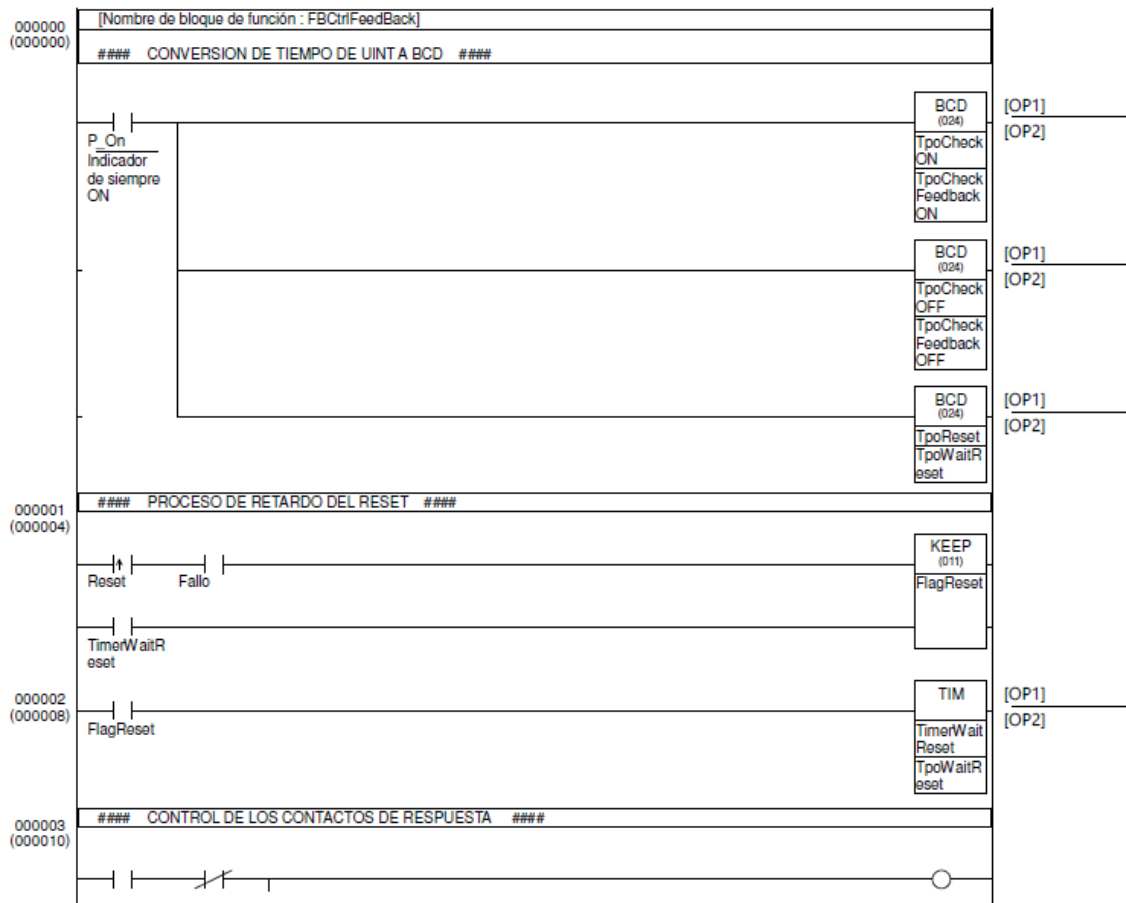
Tipo variable	Nombre	Tipo de dato	Retenido	AT	Valor inicial	Comentario
Entradas	EN	BOOL	No		FALSE	Controla ejecución del bloque de función.
Entradas	Activar	BOOL	No		FALSE	
Entradas	HayTermico	BOOL	No		FALSE	
Entradas	CheckTermico	BOOL	No		FALSE	
Entradas	CheckContactor	BOOL	No		FALSE	
Entradas	Reset	BOOL	No		FALSE	
Entradas	DetecTermNC	BOOL	No		FALSE	
Entradas	DetecCtorNC	BOOL	No		FALSE	
Entradas	TpoCheck	UINT	No		0	
Salidas	ENO	BOOL	No		FALSE	Indica éxito de ejecución del bloque de función.
Salidas	FalloTermico	BOOL	No		FALSE	
Salidas	FalloContactor	BOOL	No		FALSE	
Salidas	Fallo	BOOL	No		FALSE	
internos	TpoCheckContactor	WORD	No		10	
internos	TimerContacConex	TIMER	No			
internos	TimerContacDesconex	TIMER	No			
internos	FalloContacConex	BOOL	No		FALSE	
internos	FalloContacDesconex	BOOL	No		FALSE	
internos	TmpCheckTerm	BOOL	No		FALSE	
internos	TmpCheckCtor	BOOL	No		FALSE	

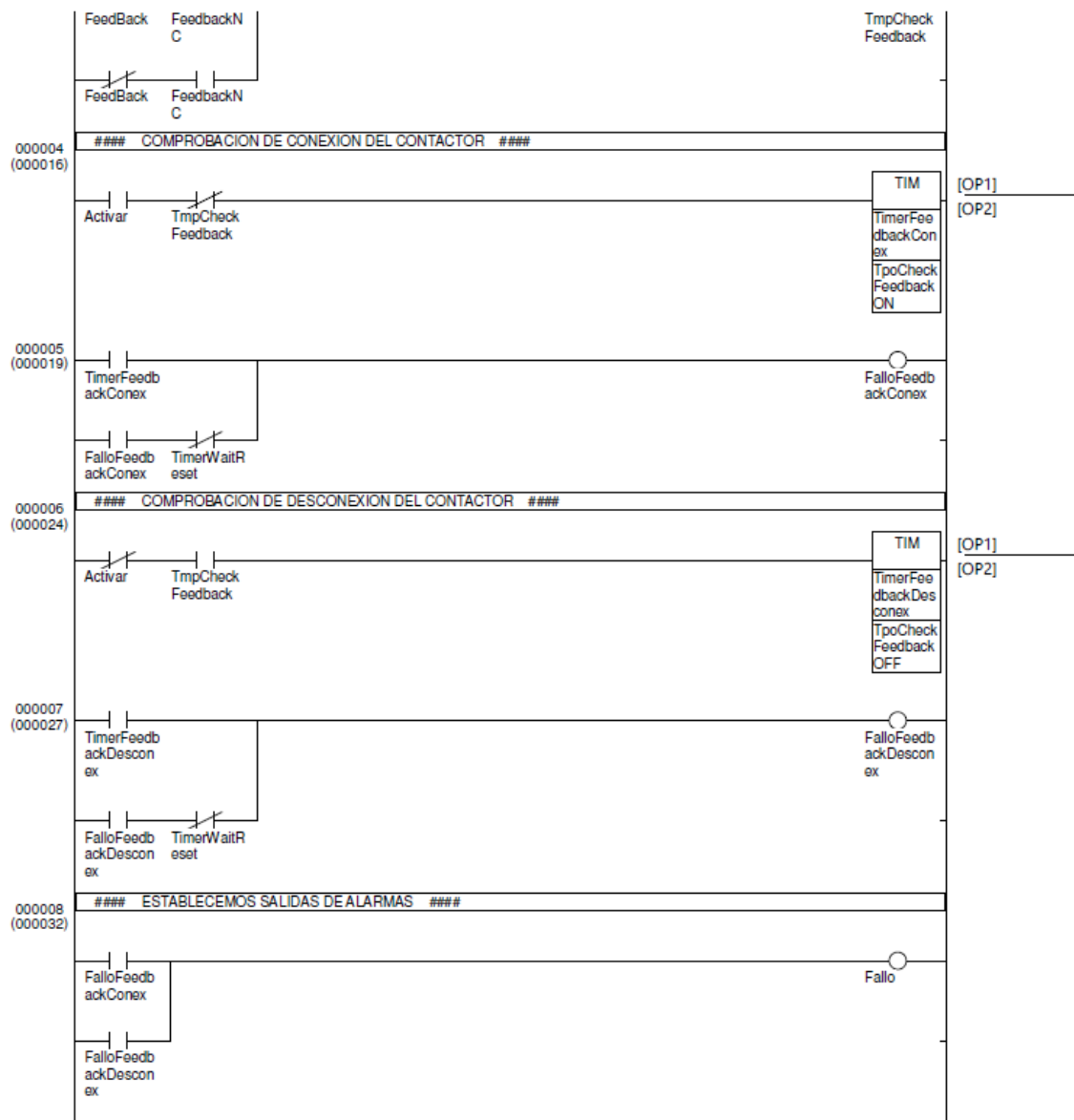




## 1.2 Feedback.

Tipo variable	Nombre	Tipo de dato	Retenido	AT	Valor inicial	Comentario
Entradas	EN	BOOL	No		FALSE	Controla ejecución del bloque de función.
Entradas	Activar	BOOL	No		FALSE	
Entradas	FeedBack	BOOL	No		FALSE	
Entradas	Reset	BOOL	No		FALSE	
Entradas	TpoCheckON	UINT	No		0	
Entradas	TpoCheckOFF	UINT	No		0	
Entradas	TpoReset	UINT	No		0	
Entradas	FeedbackNC	BOOL	No		FALSE	
Salidas	ENO	BOOL	No		FALSE	Indica éxito de ejecución del bloque de función.
Salidas	Fallo	BOOL	No		FALSE	
internos	TimerFeedbackConex	TIMER	No			
internos	TimerFeedbackDesconex	TIMER	No			
internos	TimerWaitReset	TIMER	No			
internos	FalloFeedbackConex	BOOL	No		FALSE	
internos	FalloFeedbackDesconex	BOOL	No		FALSE	
internos	TmpCheckFeedback	BOOL	No		FALSE	
internos	TpoCheckFeedbackON	WORD	No		10	
internos	TpoCheckFeedbackOFF	WORD	No		0	
internos	TpoWaitReset	WORD	No		0	
internos	FlagReset	BOOL	No		FALSE	





### 1.3 Escalar Int a Int.

Tipo variable	Nombre	Tipo de dato	Retenido	AT	Valor inicial	Comentario
Entradas	EN	BOOL	No		FALSE	Controla ejecución del bloque de función.
Entradas	ValorX	INT	No		0	
Entradas	X0	INT	No		0	
Entradas	X1	INT	No		0	
Entradas	Y0	INT	No		0	
Entradas	Y1	INT	No		0	
Salidas	ENO	BOOL	No		FALSE	Indica éxito de ejecución del bloque de función.
Salidas	ValorY	INT	No		0	
internos	Recta_m	REAL	No		0	0,
internos	Recta_n	REAL	No		0	0,
internos	Y0Real	REAL	No		0	0,
internos	Y1Real	REAL	No		0	0,
internos	X0Real	REAL	No		0	0,
internos	X1Real	REAL	No		0	0,
internos	ValorXReal	REAL	No		0	0,
internos	ValorYReal	REAL	No		0	0,

[Nombre de bloque de función : FBESclntToInt]
<pre> (* CONVERTIMOS VALORES *) X0Real := INT_TO_REAL(X0); X1Real := INT_TO_REAL(X1); Y0Real := INT_TO_REAL(Y0); Y1Real := INT_TO_REAL(Y1); ValorXReal := INT_TO_REAL(ValorX);  (* OBTENEMOS RECTA DE CONVERSION DE VALOR ANALÓGICO A DIGITAL *) RECTA_m := (Y1Real - Y0Real) / (X1Real - X0Real); RECTA_n := Y0Real - (RECTA_m * X0Real);  (* OBTENEMOS SALIDA *) ValorYReal := (RECTA_m * ValorXReal) + RECTA_n; ValorY := REAL_TO_INT(ValorYReal); </pre>

## 1.4 Escalar Int to Real.

Tipo variable	Nombre	Tipo de dato	Retenido	AT	Valor inicial	Comentario
Entradas	EN	BOOL	No		FALSE	Controla ejecución del bloque de función.
Entradas	ValorX	INT	No		0	
Entradas	X0	INT	No		0	
Entradas	X1	INT	No		0	
Entradas	Y0	REAL	No		0	0,
Entradas	Y1	REAL	No		0	0,
Entradas	ExtFault	BOOL	No		FALSE	
Entradas	Reset	BOOL	No		FALSE	
Salidas	ENO	BOOL	No		FALSE	Indica éxito de ejecución del bloque de función.
Salidas	ValorY	REAL	No		0	0,
Salidas	Fault	BOOL	No		FALSE	
internos	X0Real	REAL	No		0	0,
internos	X1Real	REAL	No		0	0,
internos	Y0Real	REAL	No		0	0,
internos	Y1Real	REAL	No		0	0,
internos	ValorXReal	REAL	No		0	0,
internos	ValorYReal	REAL	No		0	0,
internos	Recta_m	REAL	No		0	0,
internos	Recta_n	REAL	No		0	0,

[Nombre de bloque de función : FBESclntToReal]

```

(* CONVERTIMOS VALORES *)
X0Real := INT_TO_REAL(X0);
X1Real := INT_TO_REAL(X1);
Y0Real := Y0;
Y1Real := Y1;
ValorXReal := INT_TO_REAL(ValorX);

(* OBTENEMOS RECTA DE CONVERSION DE VALOR ANALÓGICO A DIGITAL *)
RECTA_m := (Y1Real - Y0Real) / (X1Real - X0Real);
RECTA_n := Y0Real - (RECTA_m * X0Real);

(* OBTENEMOS SALIDA *)
ValorYReal := (RECTA_m * ValorXReal) + RECTA_n;
ValorY := ValorYReal;

(* OBTENEMOS SALIDA DE FALLO *)
if (Reset) THEN Fault := FALSE; END_IF;
IF (ExtFault OR (ValorX < X0) OR (ValorX > X1)) THEN Fault := TRUE; END_IF;

```



## 1.5 Escalar Real to Int.

Tipo variable	Nombre	Tipo de dato	Retenido	AT	Valor inicial	Comentario
Entradas	EN	BOOL	No		FALSE	Controla ejecución del bloque de función.
Entradas	ValorX	REAL	No		0	0,
Entradas	X0	REAL	No		0	0,
Entradas	X1	REAL	No		0	0,
Entradas	Y0	INT	No		0	
Entradas	Y1	INT	No		0	
Salidas	ENO	BOOL	No		FALSE	Indica éxito de ejecución del bloque de función.
Salidas	ValorY	INT	No		0	
internos	Recta_m	REAL	No		0	0,
internos	Recta_n	REAL	No		0	0,
internos	Y0Real	REAL	No		0	0,
internos	Y1Real	REAL	No		0	0,
internos	X0Real	REAL	No		0	0,
internos	X1Real	REAL	No		0	0,
internos	ValorXReal	REAL	No		0	0,
internos	ValorYReal	REAL	No		0	0,

[Nombre de bloque de función : FBEscRealToInt]

```

(* CONVERTIMOS VALORES *)
X0Real := X0;
X1Real := X1;
Y0Real := INT_TO_REAL(Y0);
Y1Real := INT_TO_REAL(Y1);
ValorXReal := ValorX;

(* OBTENEMOS RECTA DE CONVERSION DE VALOR ANALÓGICO A DIGITAL *)
RECTA_m := (Y1Real - Y0Real) / (X1Real - X0Real);
RECTA_n := Y0Real - (RECTA_m * X0Real);

(* OBTENEMOS SALIDA *)
ValorYReal := (RECTA_m * ValorXReal) + RECTA_n;
ValorY := REAL_TO_INT(ValorYReal);

```

## 1.6 Histéresis de máximos a Real.

Tipo variable	Nombre	Tipo de dato	Retenido	AT	Valor inicial	Comentario
Entradas	EN	BOOL	No		FALSE	Controla ejecución del bloque de función.
Entradas	ValorProc	REAL	No		0	0,
Entradas	ValorMax	REAL	No		0	0,
Entradas	ValorHister	REAL	No		0	0,
Salidas	ENO	BOOL	No		FALSE	Indica éxito de ejecución del bloque de función.
Salidas	EnHister	BOOL	No		FALSE	
internos	iValorMin	REAL	No		0	0,

[Nombre de bloque de función : FBHisterDeMaxREAL]

```
(* OBTENEMOS VALOR MÁXIMO *)
iValorMin := ValorMax - ValorHister;

(* MOMENTO DE ENTRADA EN PROCESO DE HISTÉRESIS *)
IF (ValorProc >= ValorMax) THEN EnHister := TRUE; END_IF;

(* RESETEAMOS PROCESO DE HISTÉRESIS *)
IF (ValorProc < iValorMin) THEN EnHister := FALSE; END_IF;
```

## 1.7 Histéresis de mínimos a Real.

Tipo variable	Nombre	Tipo de dato	Retenido	AT	Valor inicial	Comentario
Entradas	EN	BOOL	No		FALSE	Controla ejecución del bloque de función.
Entradas	ValorProc	REAL	No		0	0,
Entradas	ValorMin	REAL	No		0	0,
Entradas	ValorHister	REAL	No		0	0,
Salidas	ENO	BOOL	No		FALSE	Indica éxito de ejecución del bloque de función.
Salidas	EnHister	BOOL	No		FALSE	
internos	iValorMax	REAL	No		0	0,

[Nombre de bloque de función : FBHisterDeMinREAL]

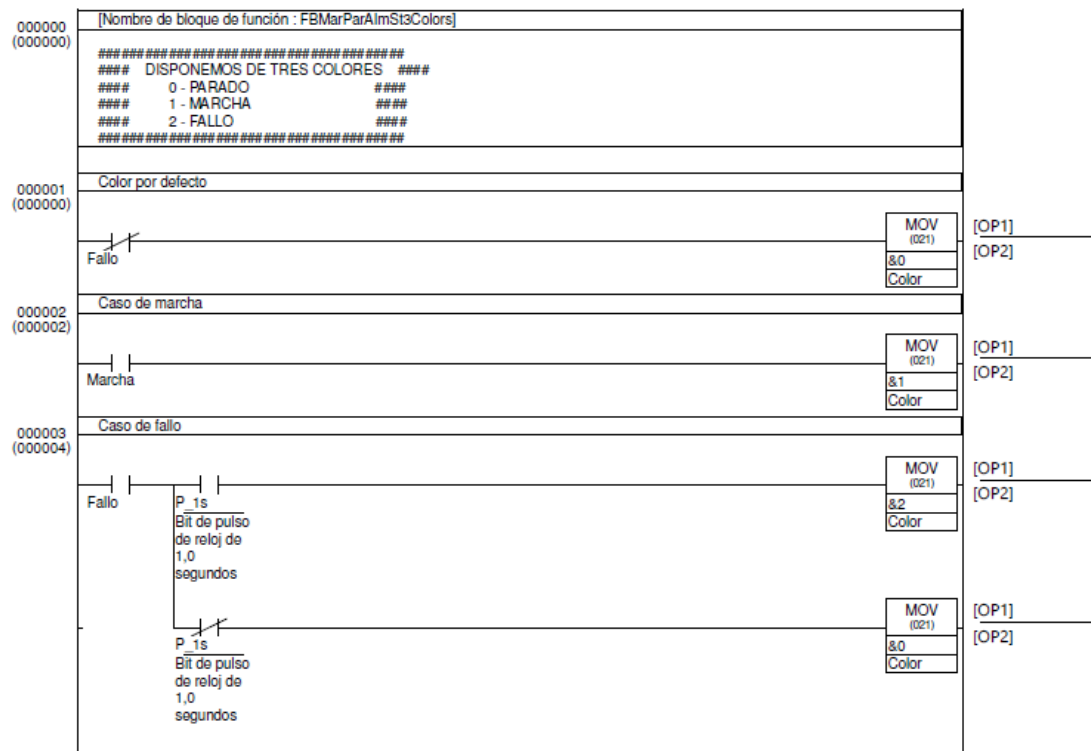
```
(* OBTENEMOS VALOR MÁXIMO *)
iValorMax := ValorMin + ValorHister;

(* MOMENTO DE ENTRADA EN PROCESO DE HISTÉRESIS *)
IF (ValorProc <= ValorMin) THEN EnHister := TRUE; END_IF;

(* RESETEAMOS PROCESO DE HISTÉRESIS *)
IF (ValorProc > iValorMax) THEN EnHister := FALSE; END_IF;
```

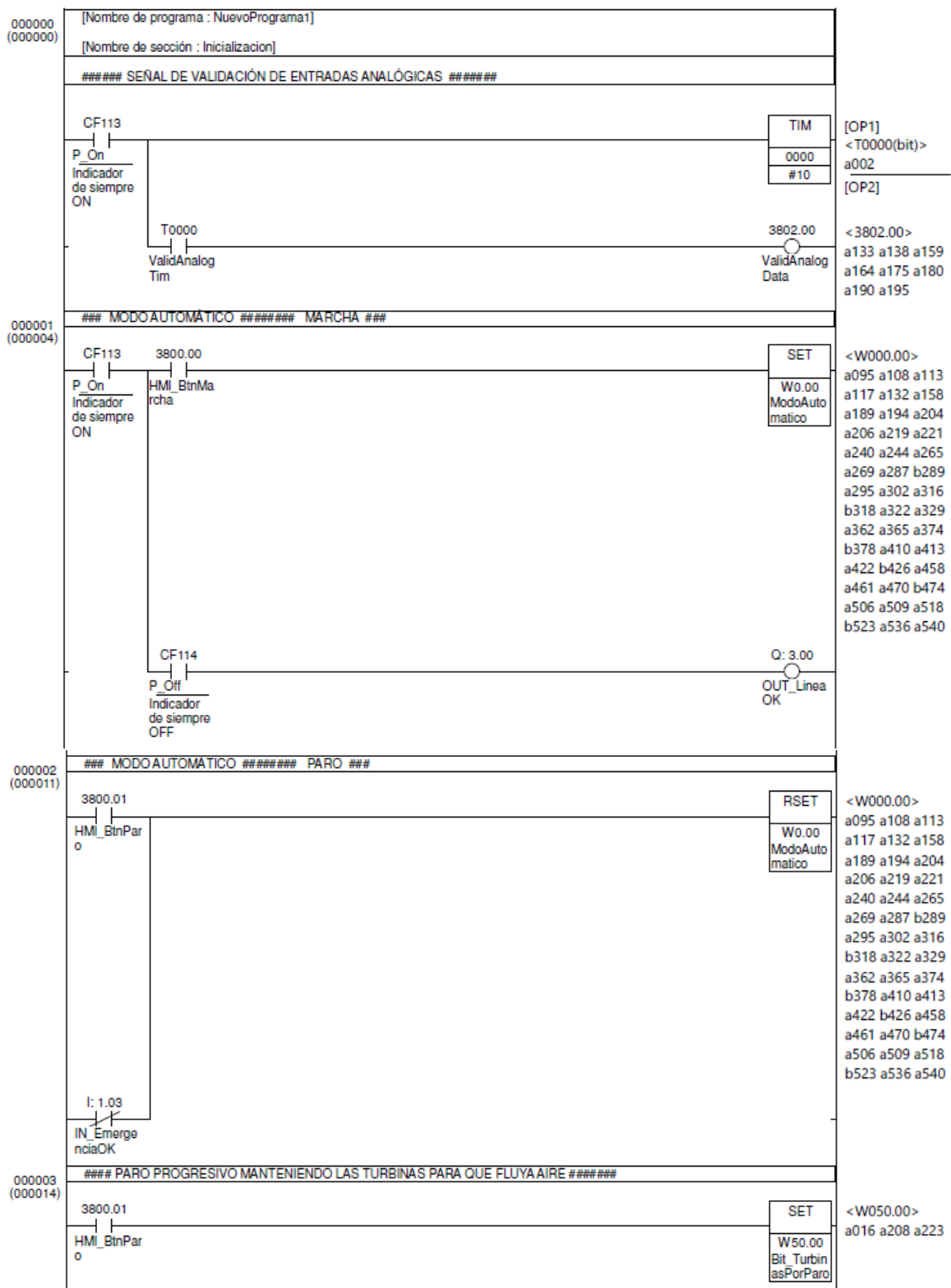
## 1.8 Gestión colores baliza.

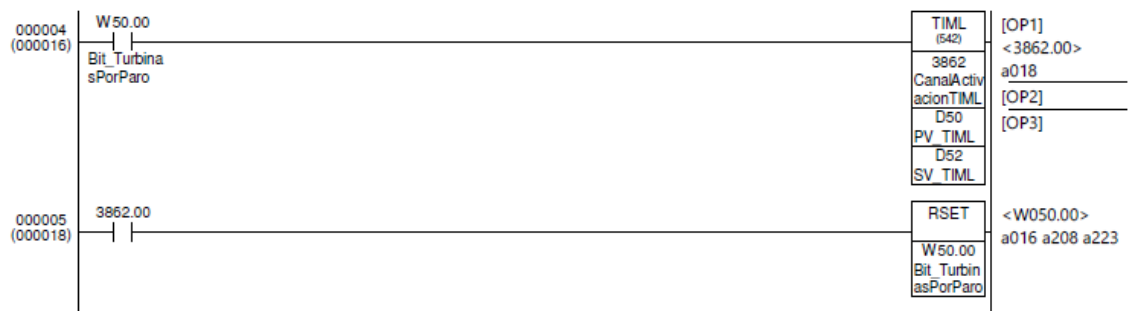
Tipo variable	Nombre	Tipo de dato	Retenido	AT	Valor inicial	Comentario
Entradas	EN	BOOL	No		FALSE	Controla ejecución del bloque de función.
Entradas	Marcha	BOOL	No		FALSE	
Entradas	Fallo	BOOL	No		FALSE	
Salidas	ENO	BOOL	No		FALSE	Indica éxito de ejecución del bloque de función.
Salidas	Color	UINT	No		0	



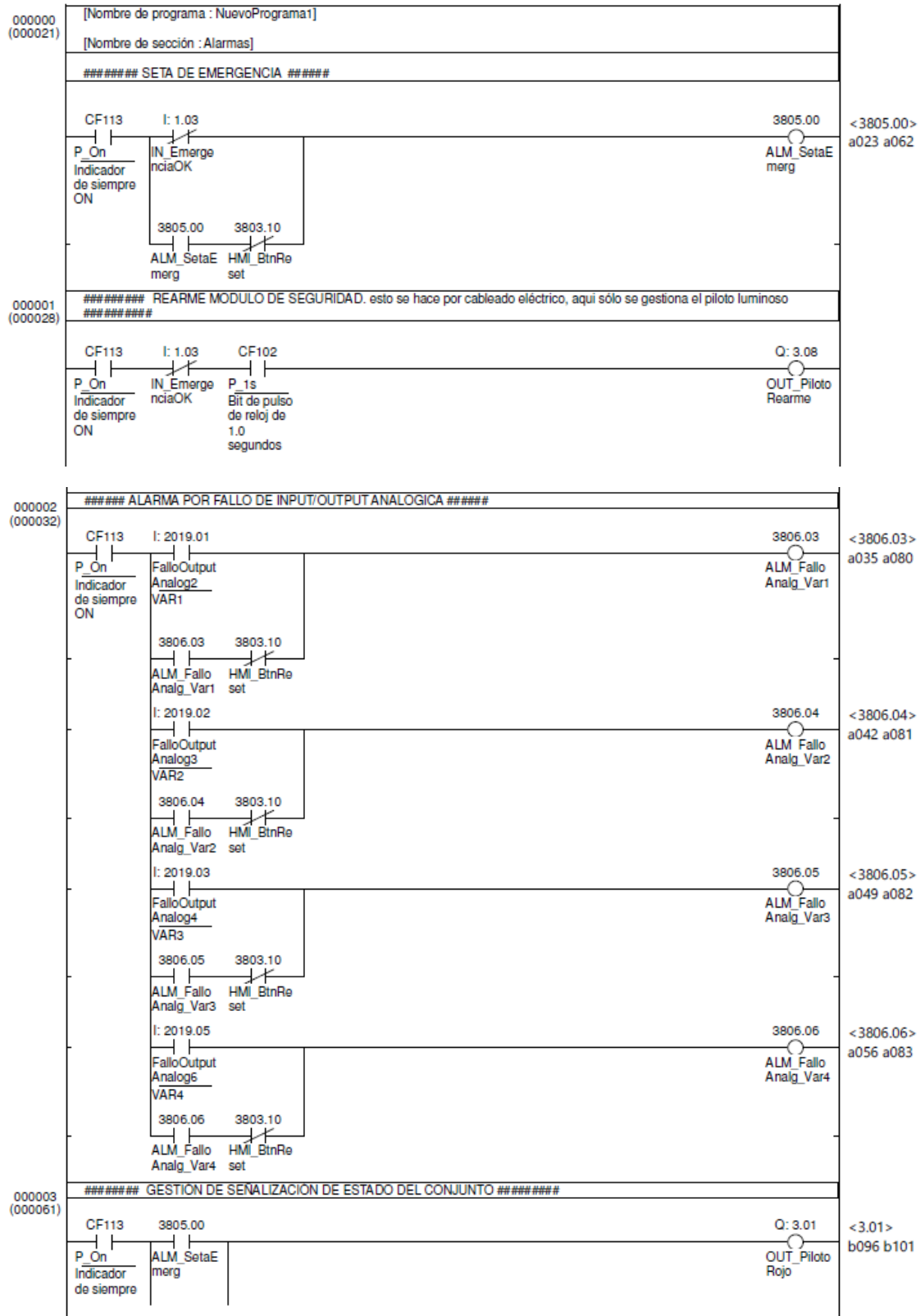
## 2. SECCIONES PROGRAMA

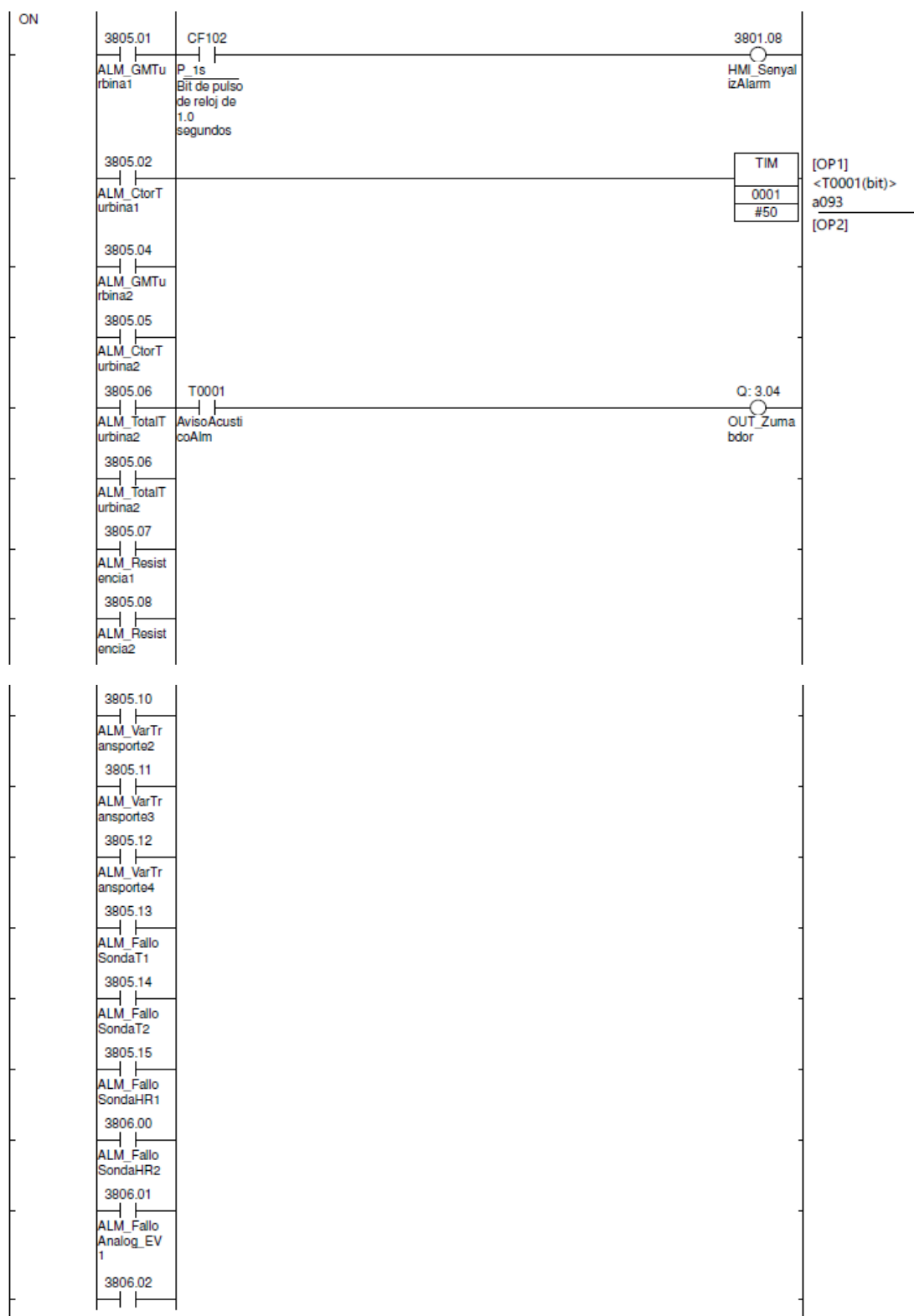
### 2.1 Inicialización.

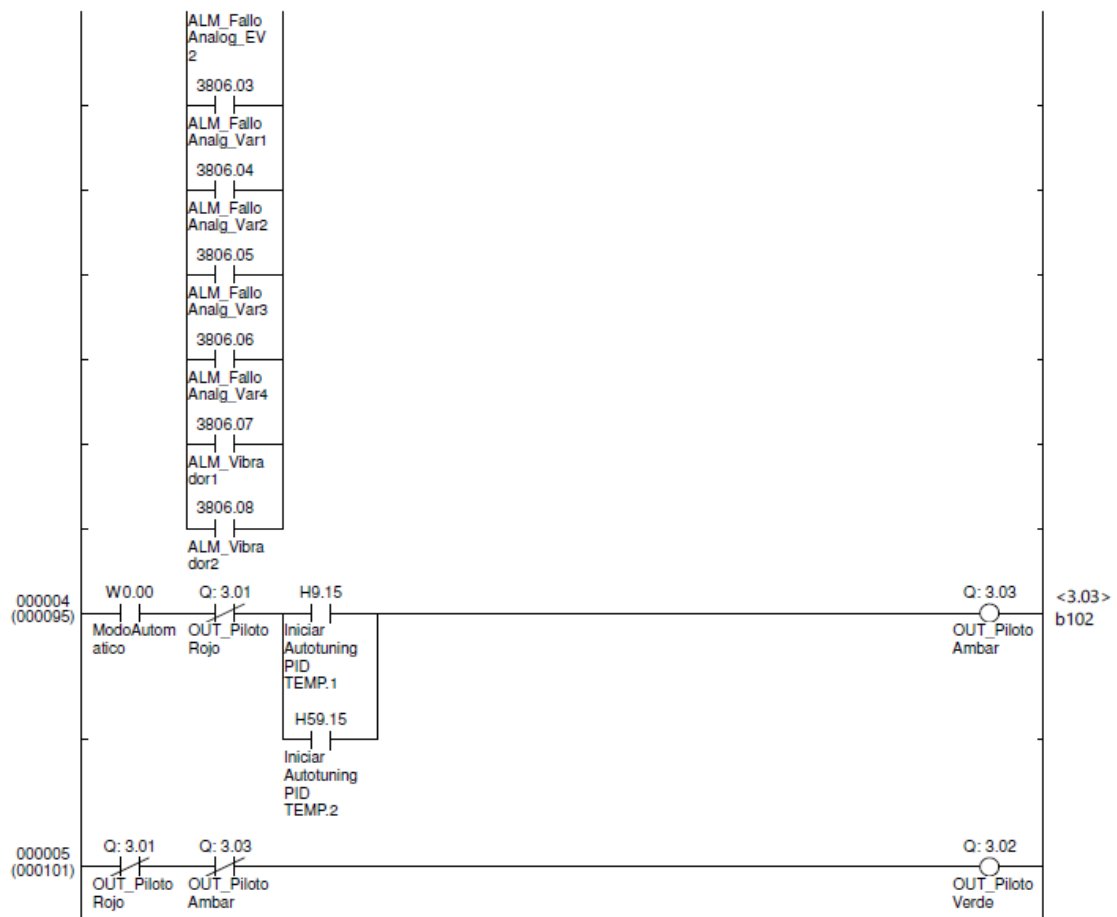




## 2.2 Alarmas

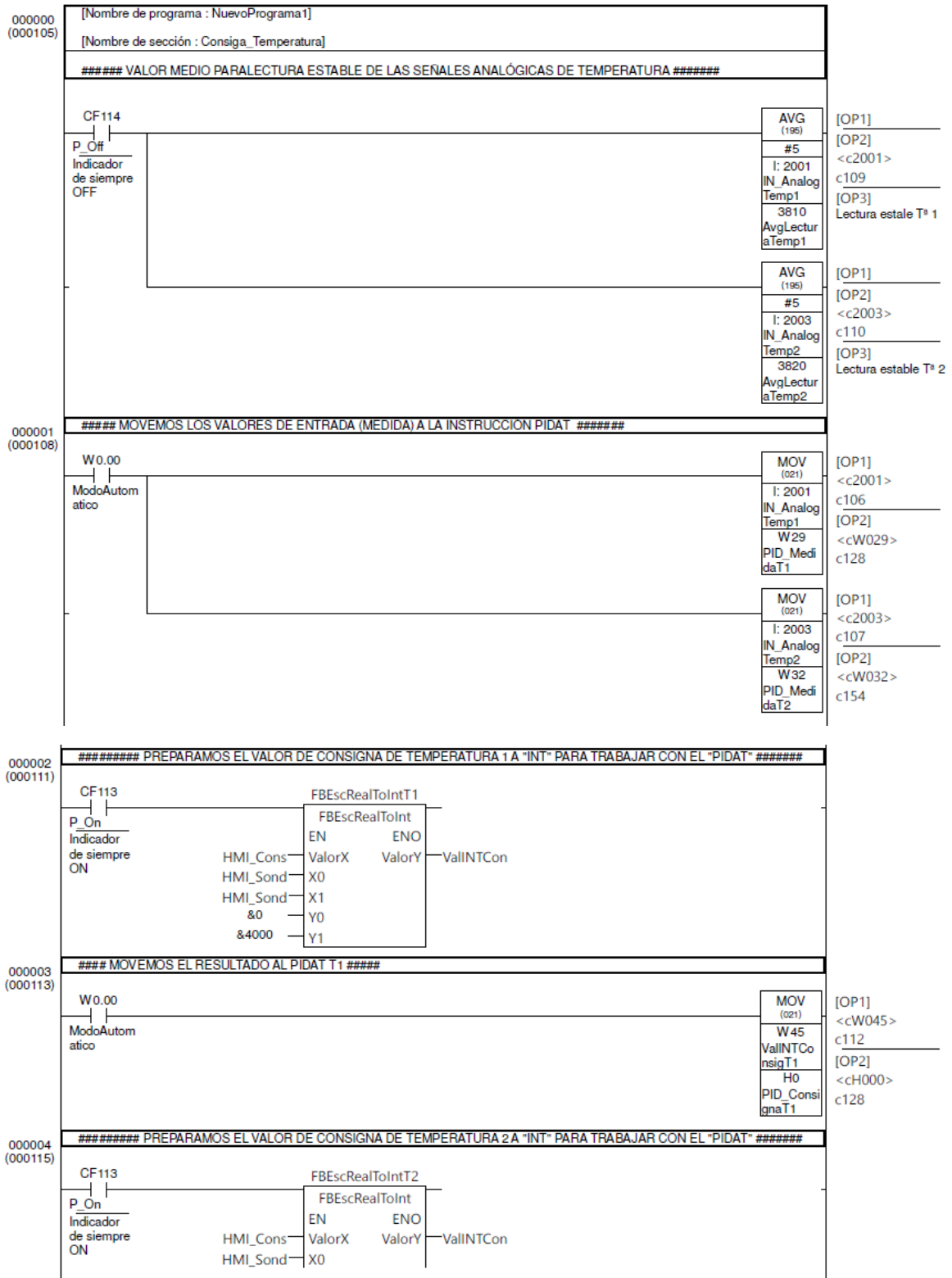


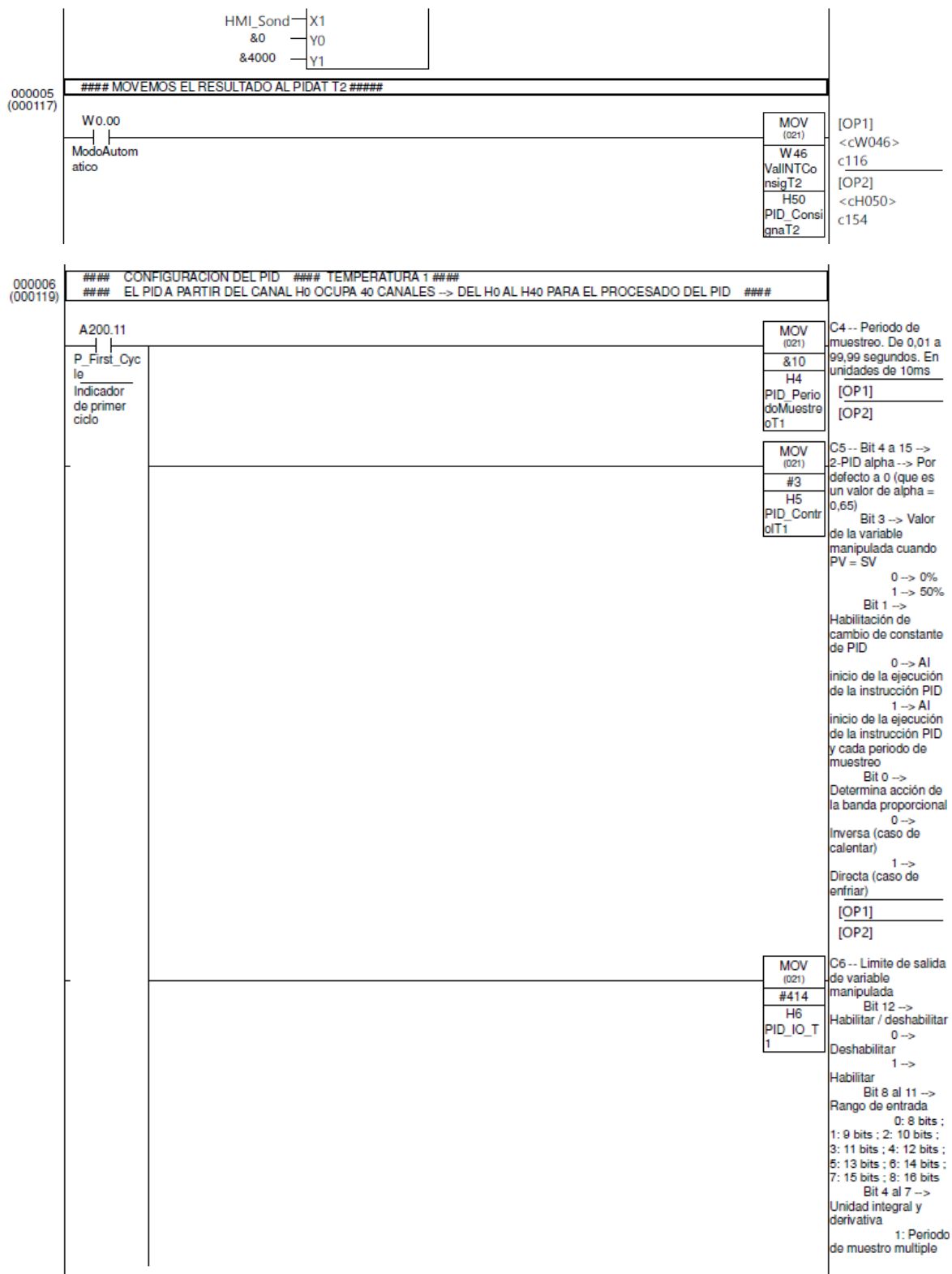


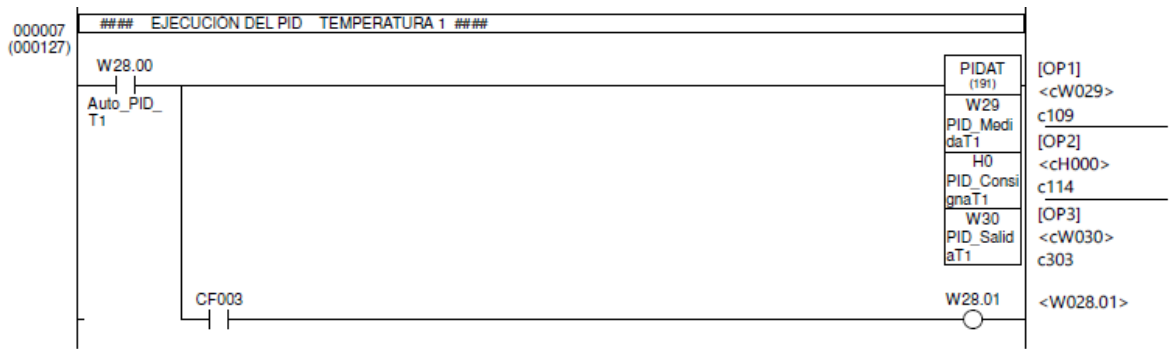
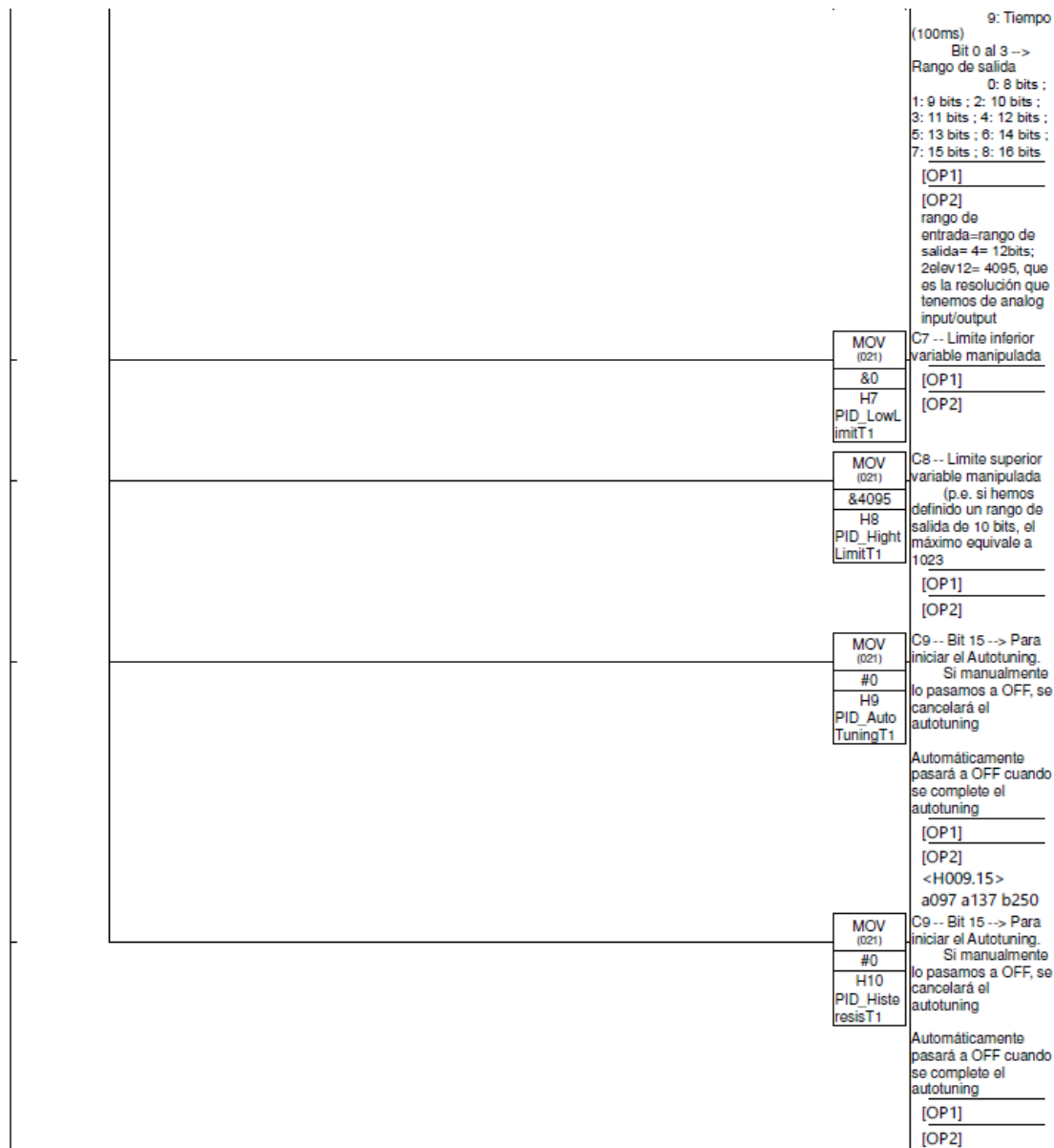


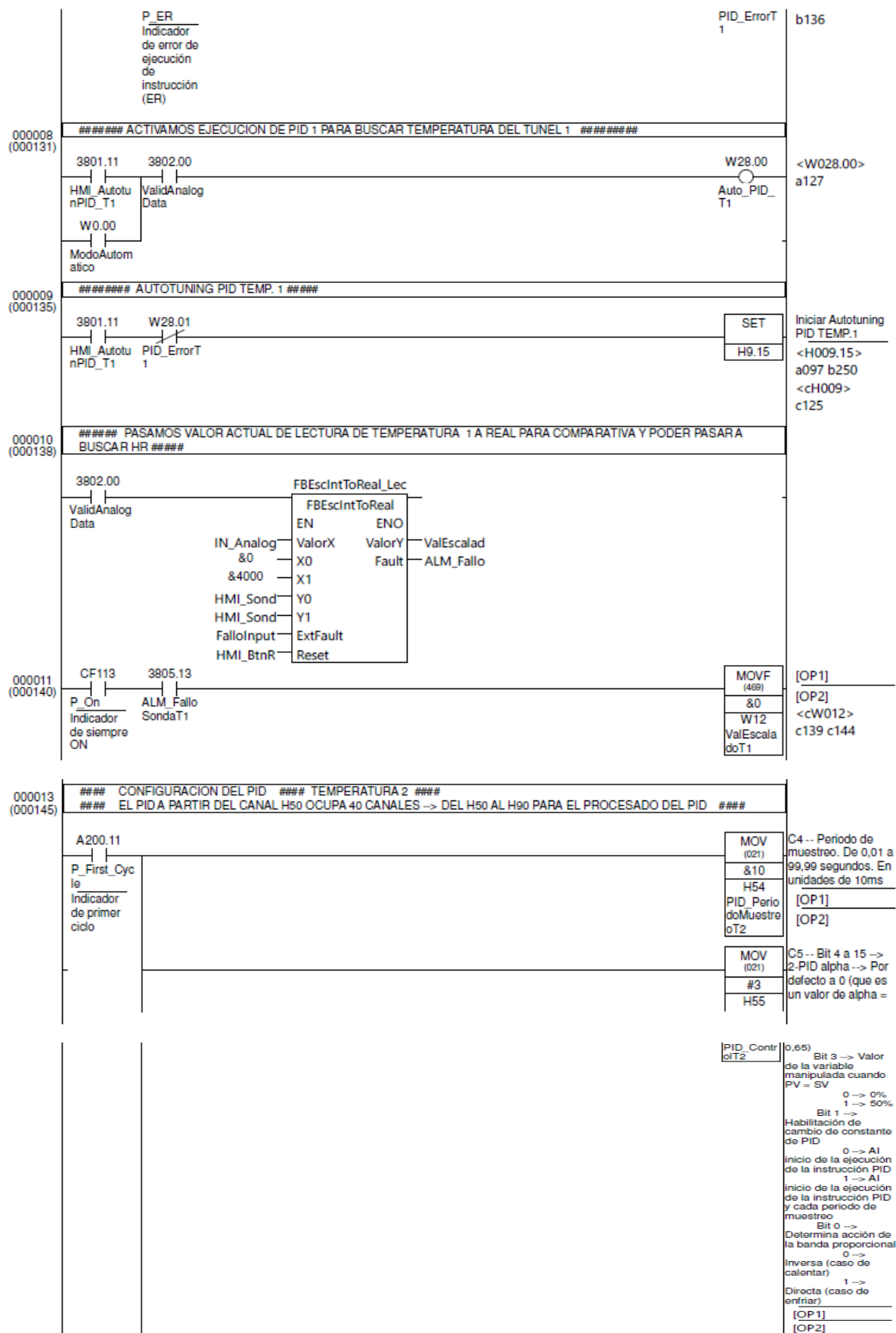


## 2.3 Consigna de temperatura

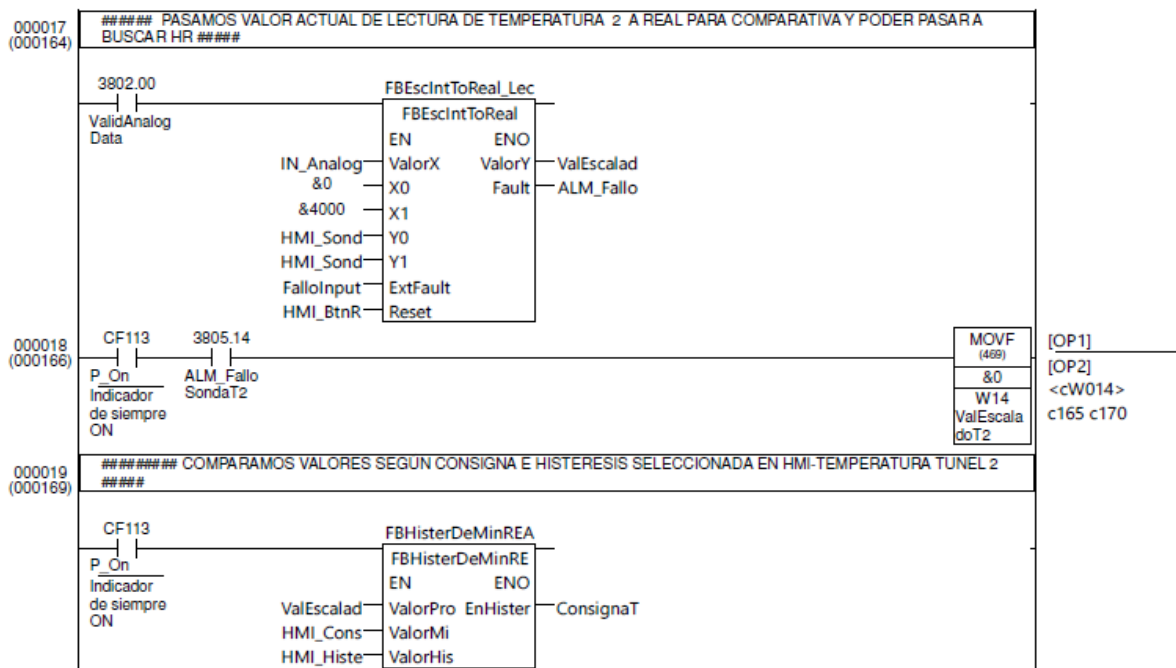
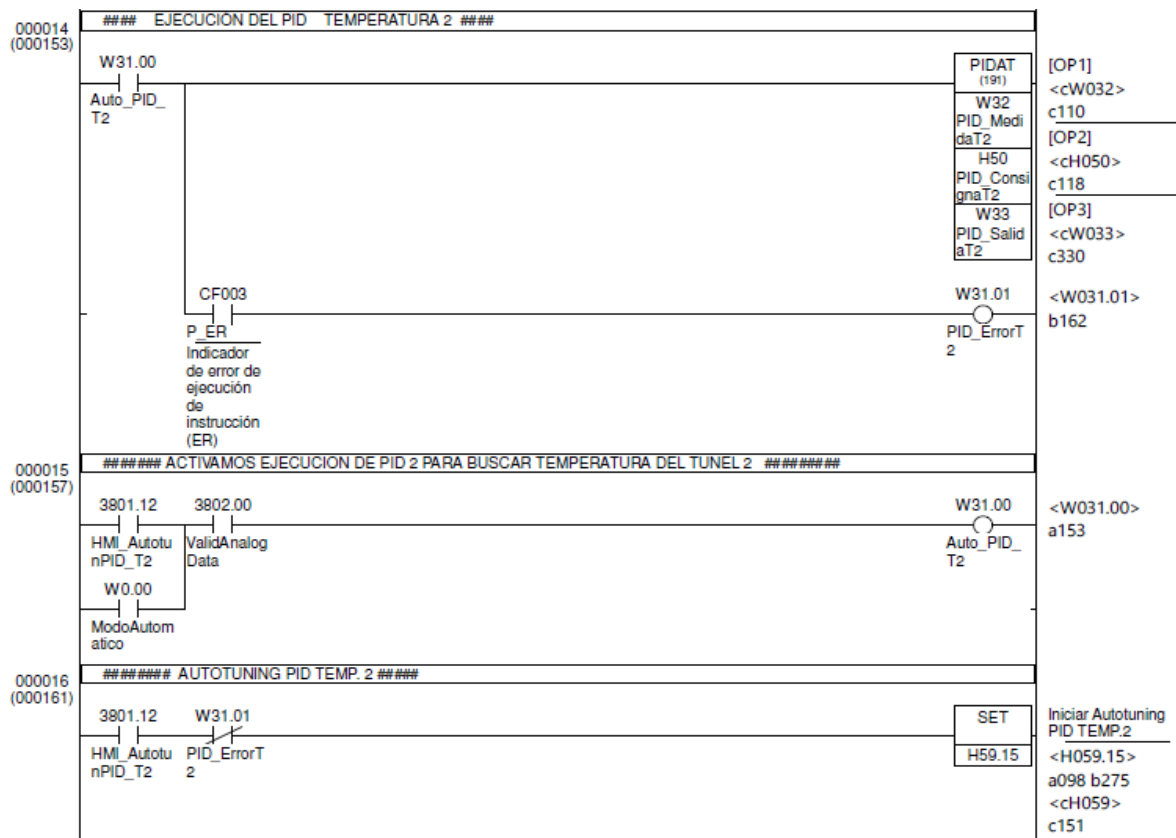




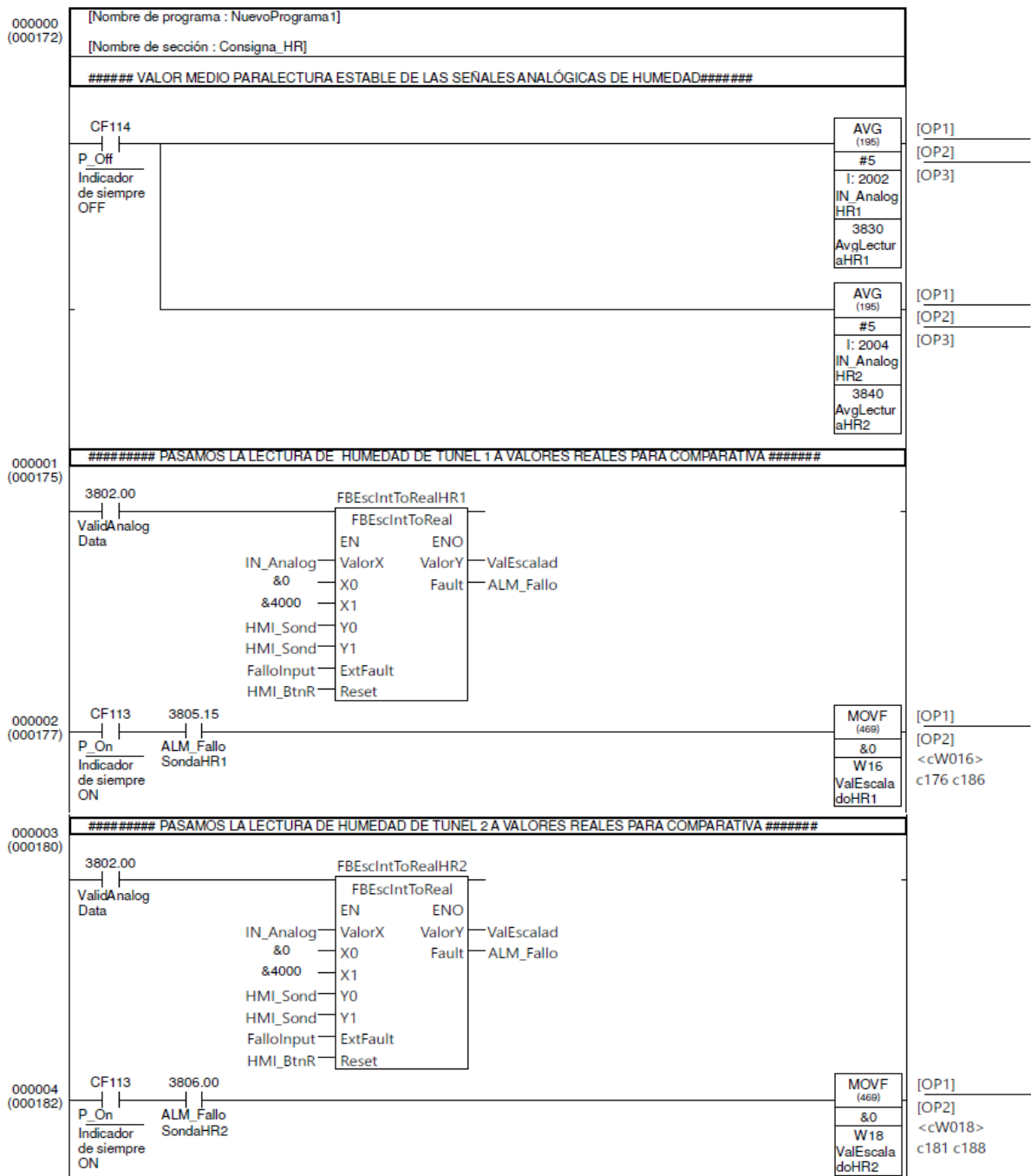


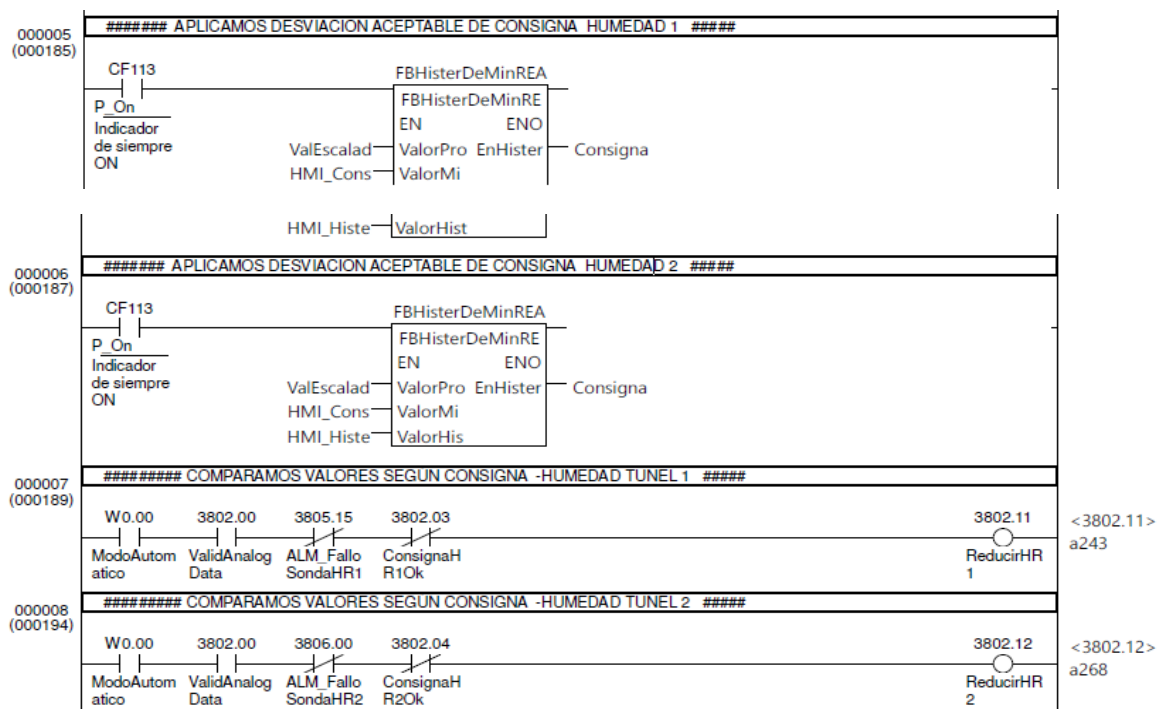


	<div>MOV (021)</div> <div>#414</div> <div>H56</div> <div>PID_IO_T</div> <div>2</div>	C6 -- Limite de salida de variable manipulada Bit 12 --> Habilitar / deshabilitar 0 --> Deshabilitar 1 --> Habilitar Bit 8 al 11 --> Rango de entrada 0: 8 bits ; 1: 9 bits ; 2: 10 bits ; 3: 11 bits ; 4: 12 bits ; 5: 13 bits ; 6: 14 bits ; 7: 15 bits ; 8: 16 bits Bit 4 al 7 --> Unidad integral y derivativa 1: Periodo de muestro multiple 9: Tiempo (100ms) Bit 0 al 3 --> Rango de salida 0: 8 bits ; 1: 9 bits ; 2: 10 bits ; 3: 11 bits ; 4: 12 bits ; 5: 13 bits ; 6: 14 bits ; 7: 15 bits ; 8: 16 bits [OP1] [OP2] rango de entrada=rango de salida= 4= 12bits; 2elev12= 4095, que es la resolución que tenemos de analog input/output
	<div>MOV (021)</div> <div>&amp;0</div> <div>H57</div> <div>PID_LowLimitT2</div>	C7 -- Limite inferior variable manipulada [OP1] [OP2]
	<div>MOV (021)</div> <div>&amp;4095</div> <div>H58</div> <div>PID_HightLimitT2</div>	C8 -- Limite superior variable manipulada (p.e. si hemos definido un rango de salida de 10 bits, el máximo equivale a 1023 [OP1] [OP2]
	<div>MOV (021)</div> <div>#0</div> <div>H59</div> <div>PID_AutoTuningT2</div>	C9 -- Bit 15 --> Para iniciar el Autotuning. Si manualmente lo pasamos a OFF, se cancelará el autotuning Automáticamente pasará a OFF cuando se complete el autotuning [OP1] [OP2] <H059.15> a098 a163 b275
	<div>MOV (021)</div> <div>#0</div> <div>H60</div> <div>PID_Histe resistT2</div>	C9 -- Bit 15 --> Para iniciar el Autotuning. Si manualmente lo pasamos a OFF, se cancelará el autotuning Automáticamente pasará a OFF cuando se complete el autotuning [OP1] [OP2]



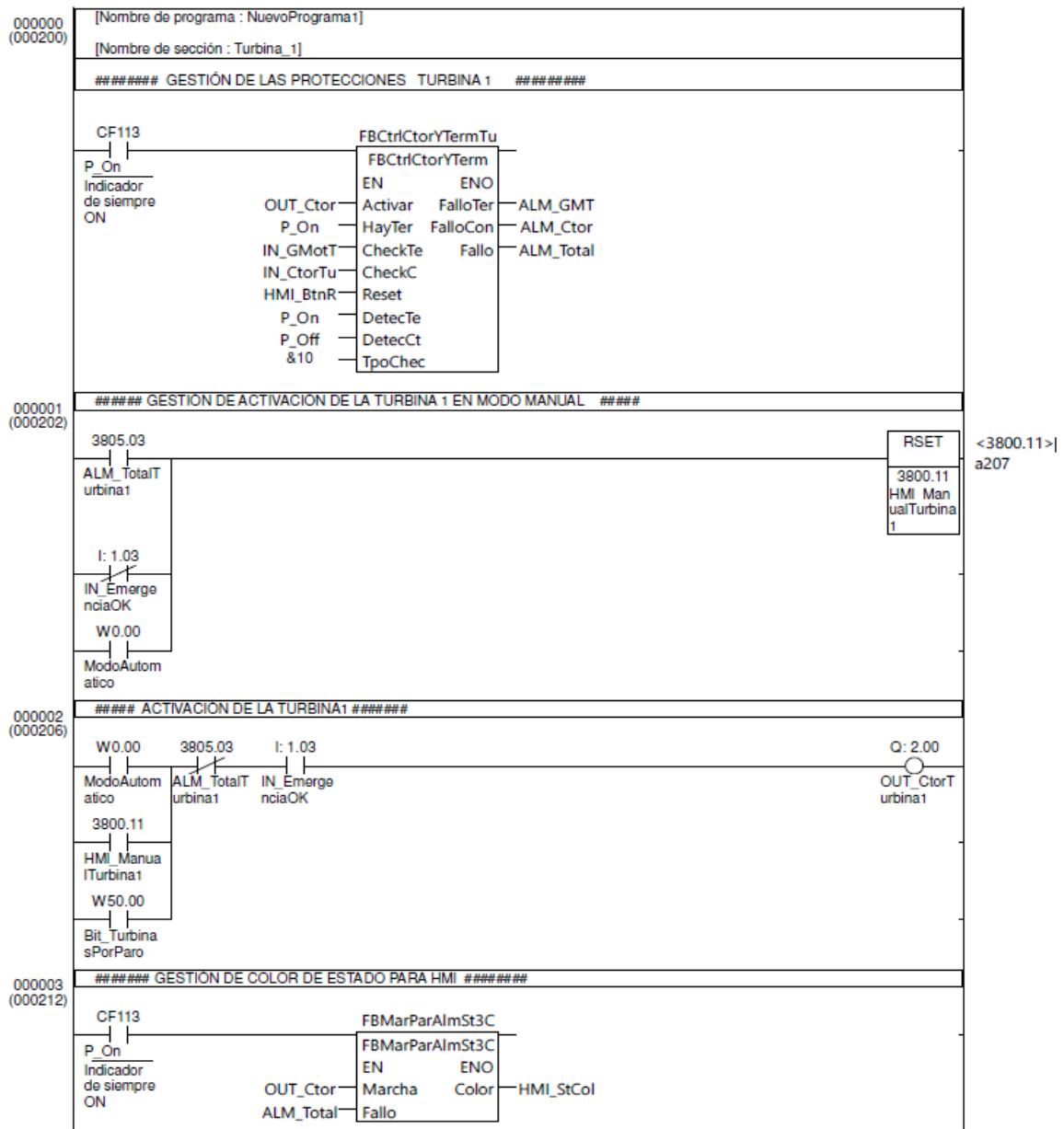
## 2.4 Consigna humedad



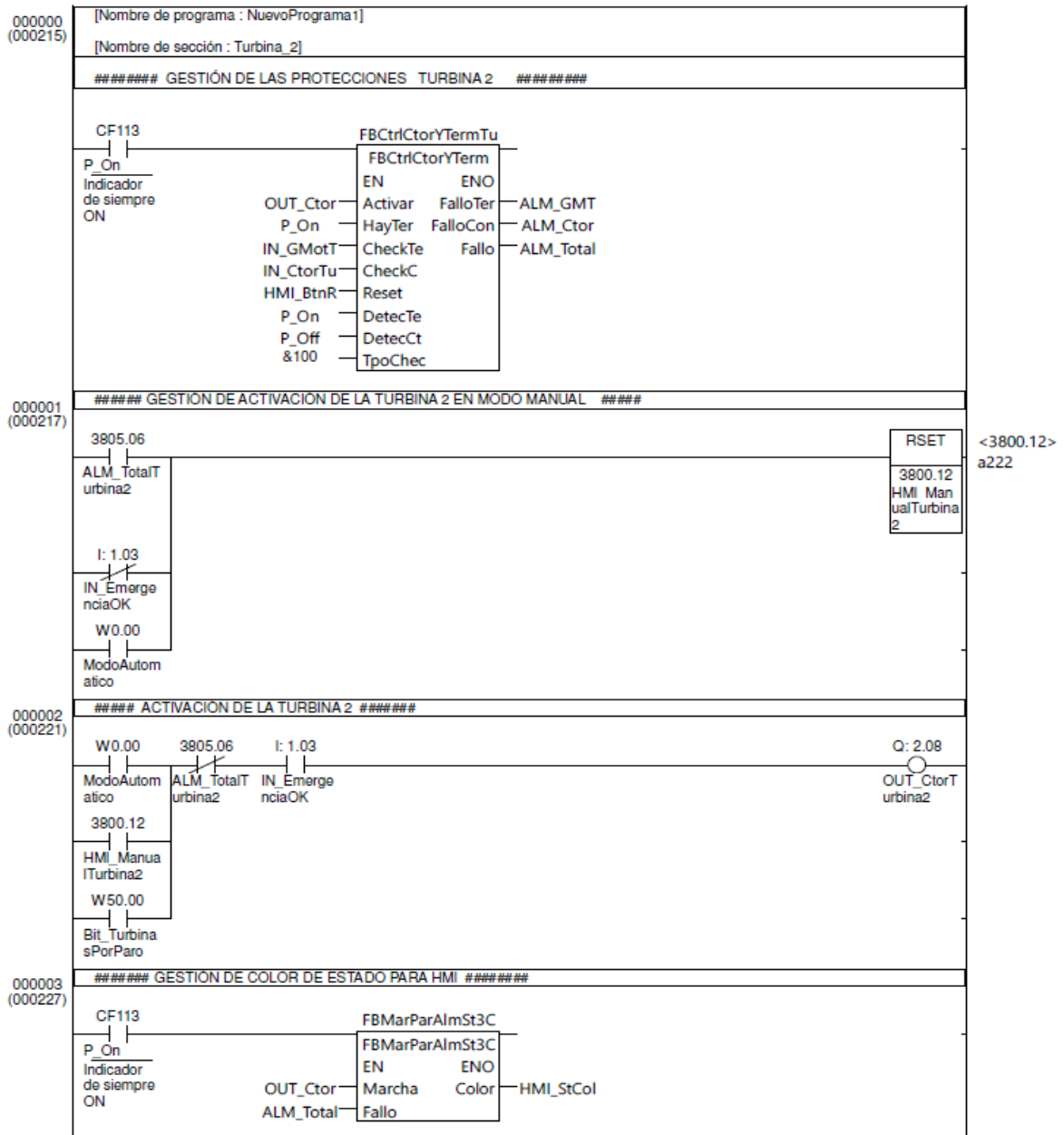




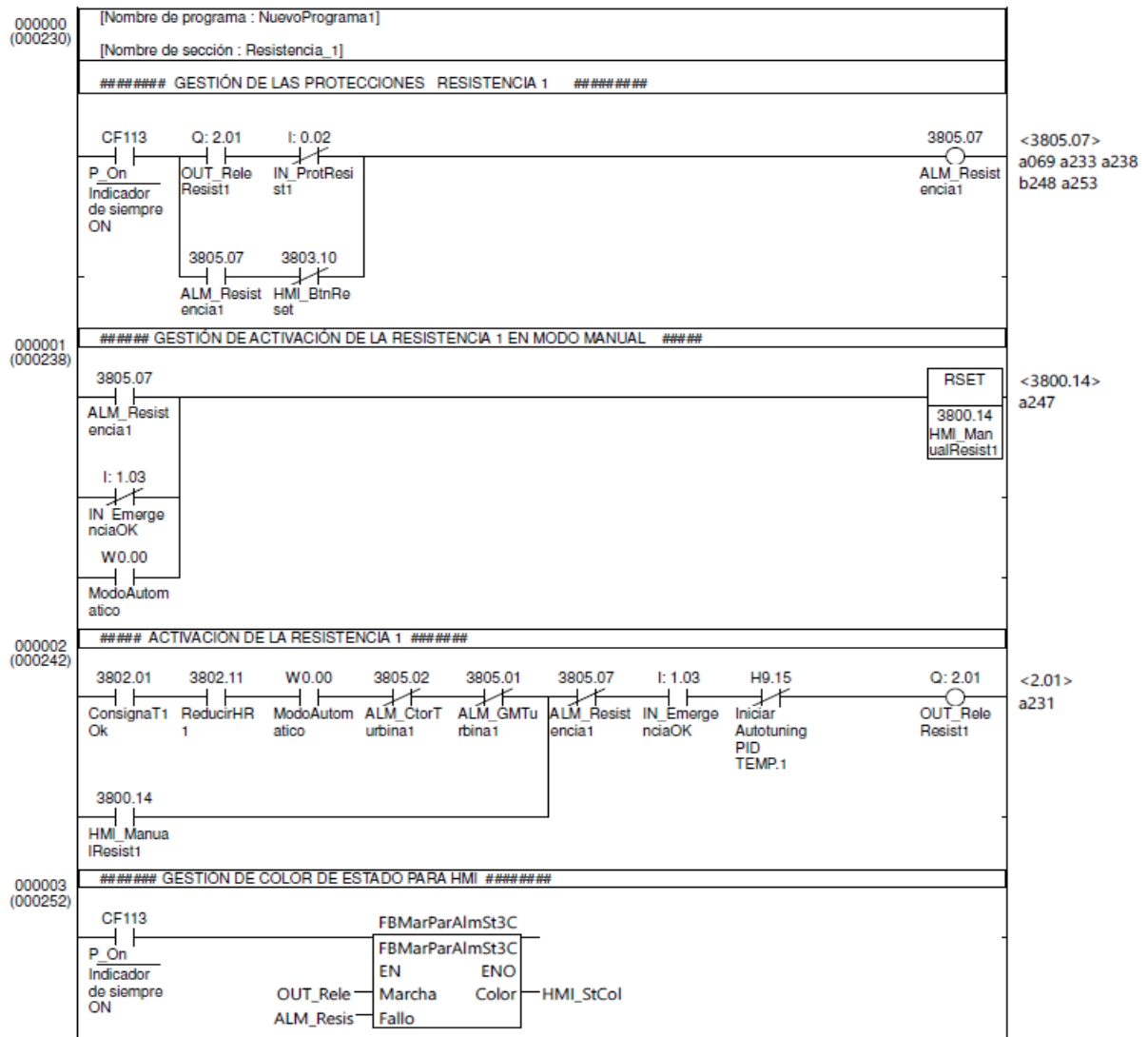
## 2.5 Turbina 1



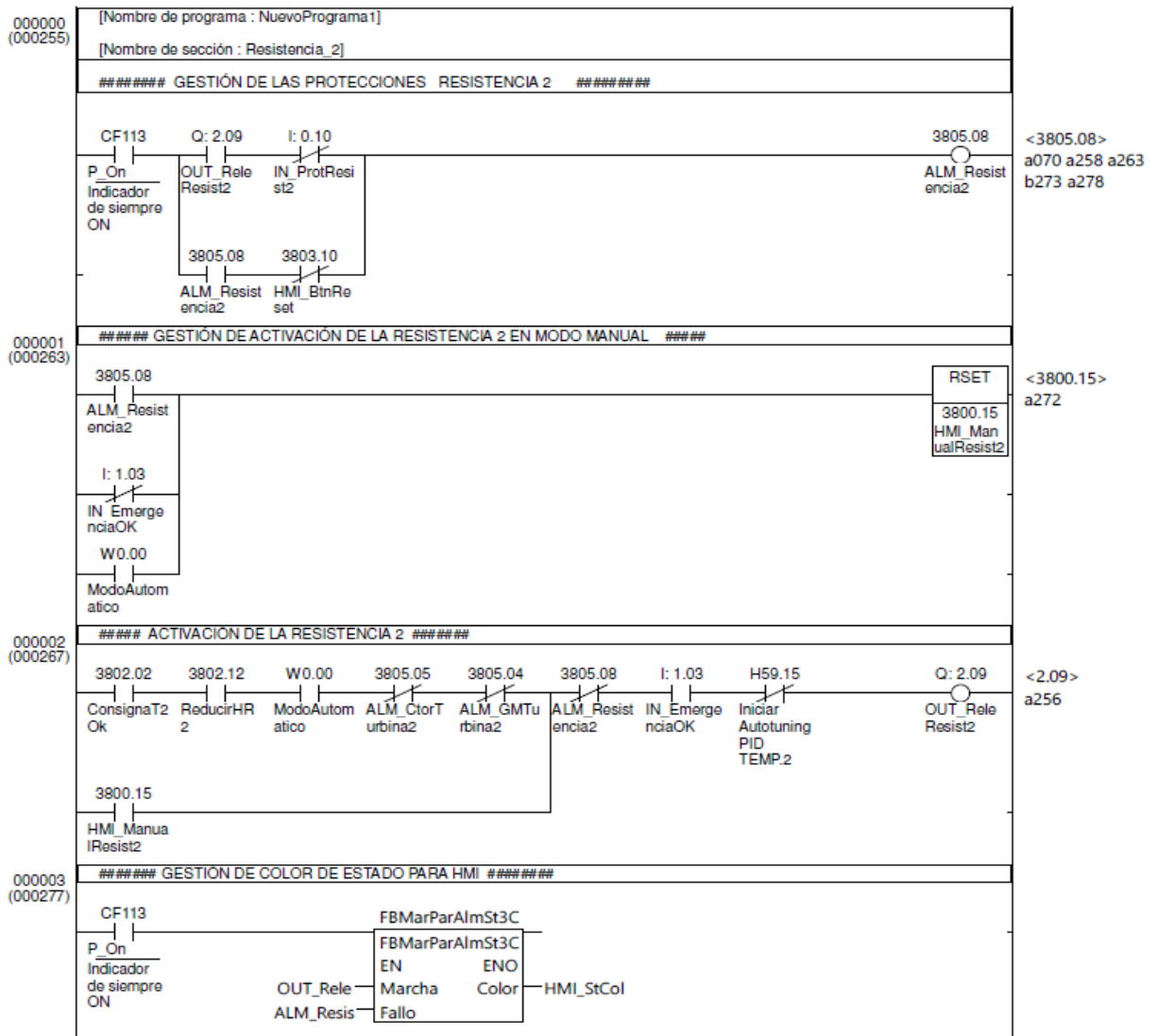
## 2.6 Turbina 2



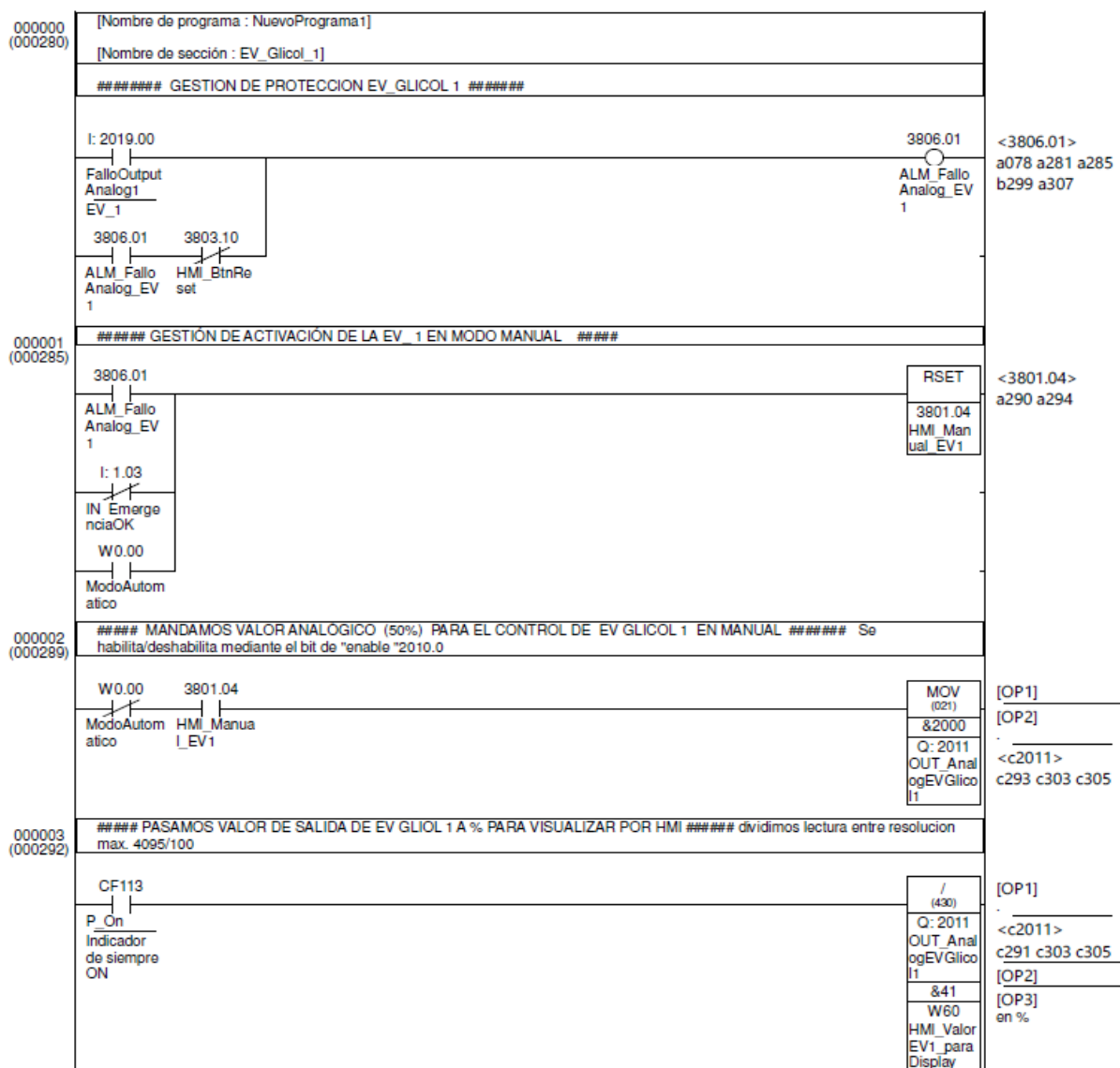
## 2.7 Resistencia 1

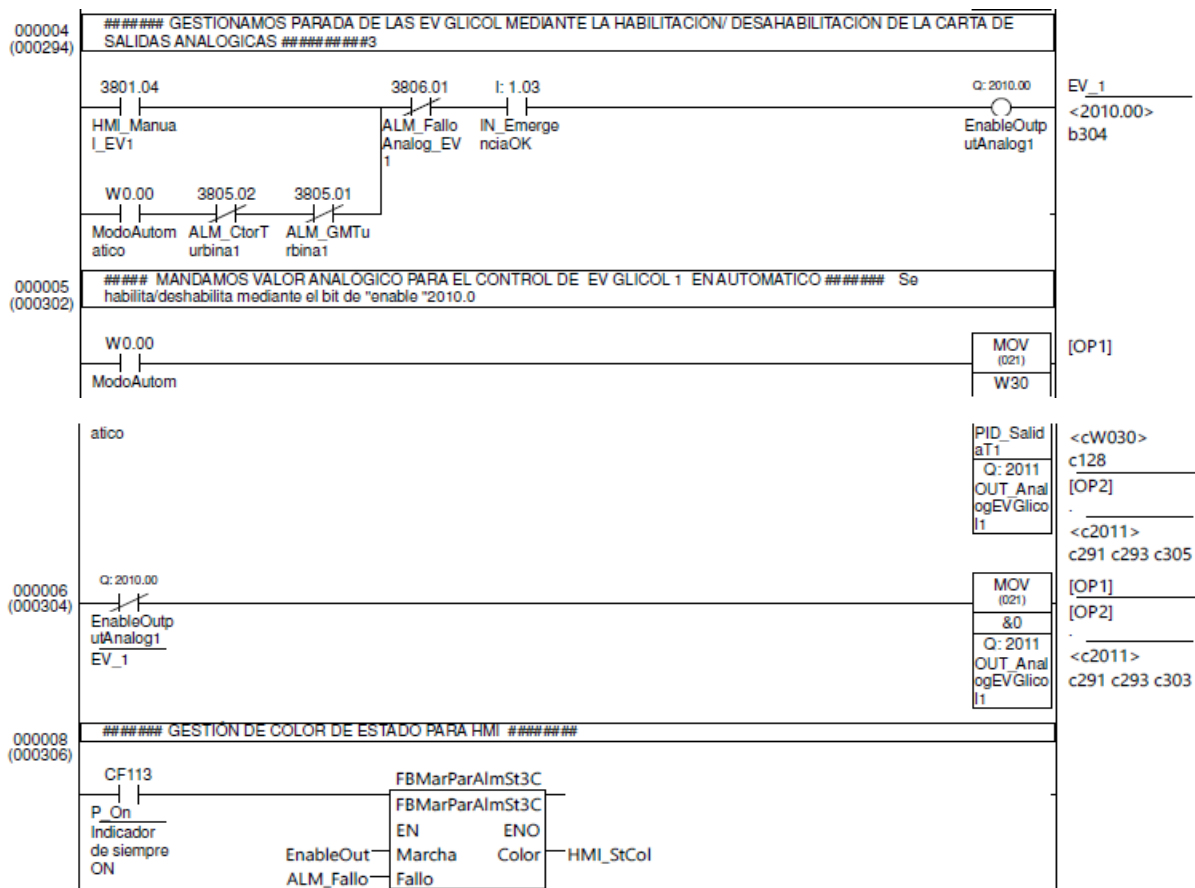


## 2.8 Resistencia 2

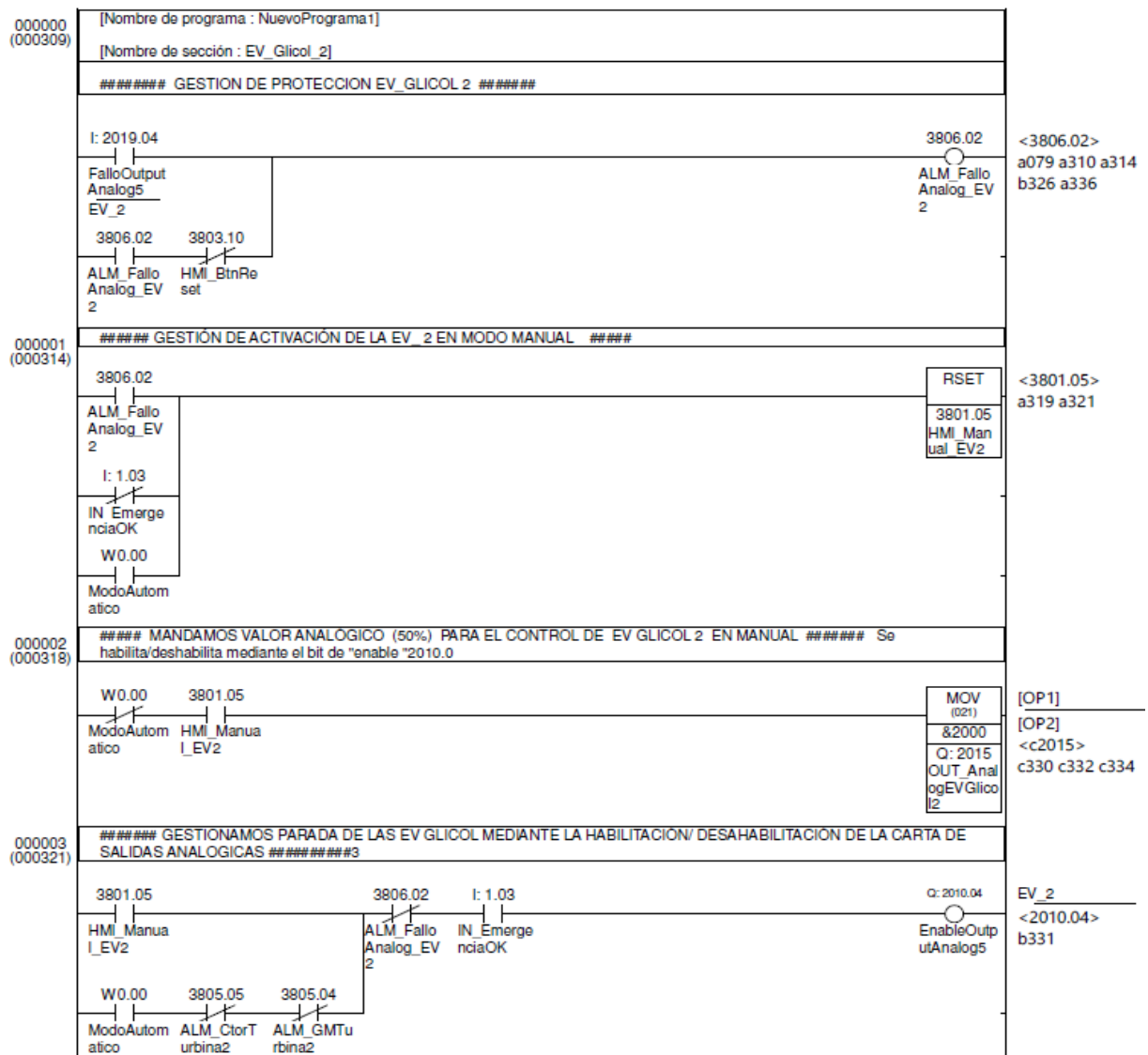


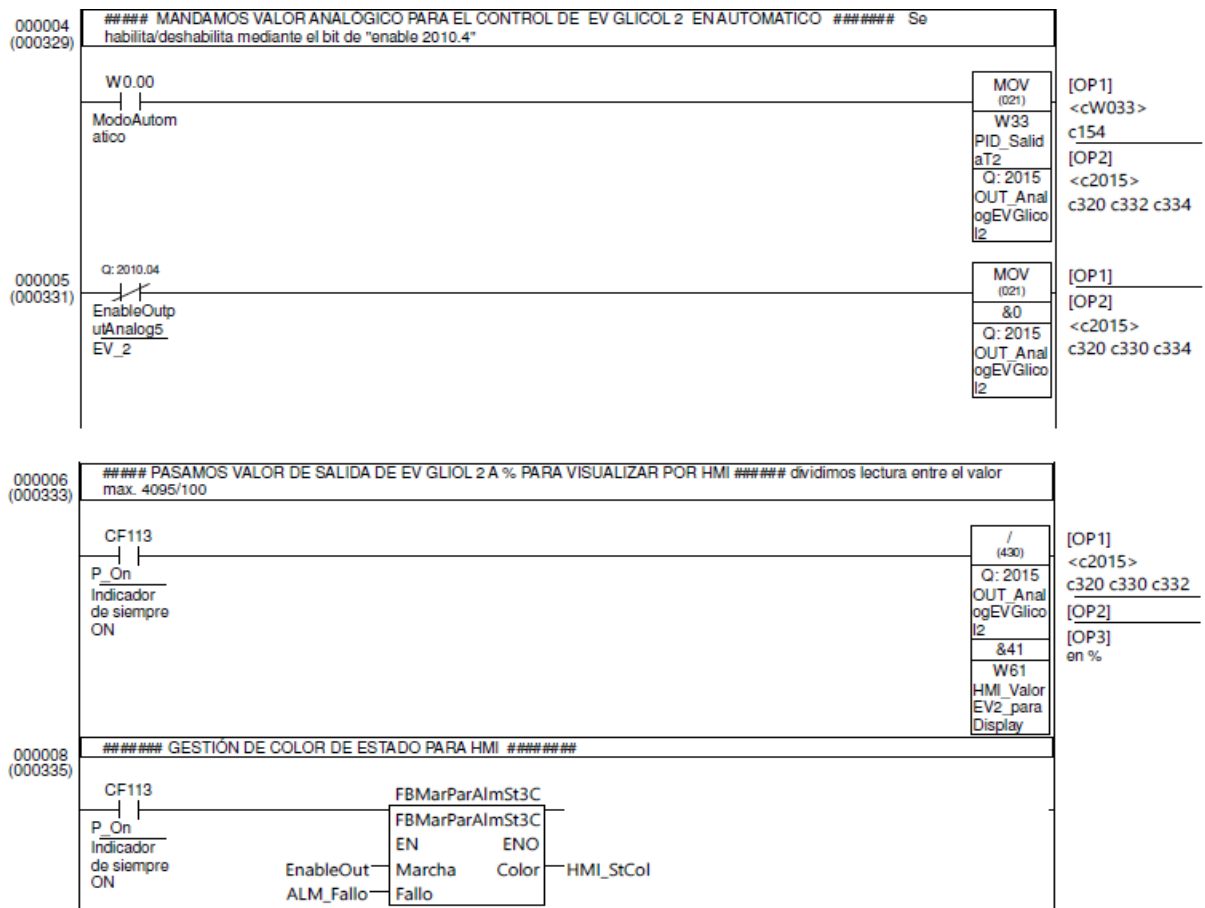
## 2.9 Electroválvula 1





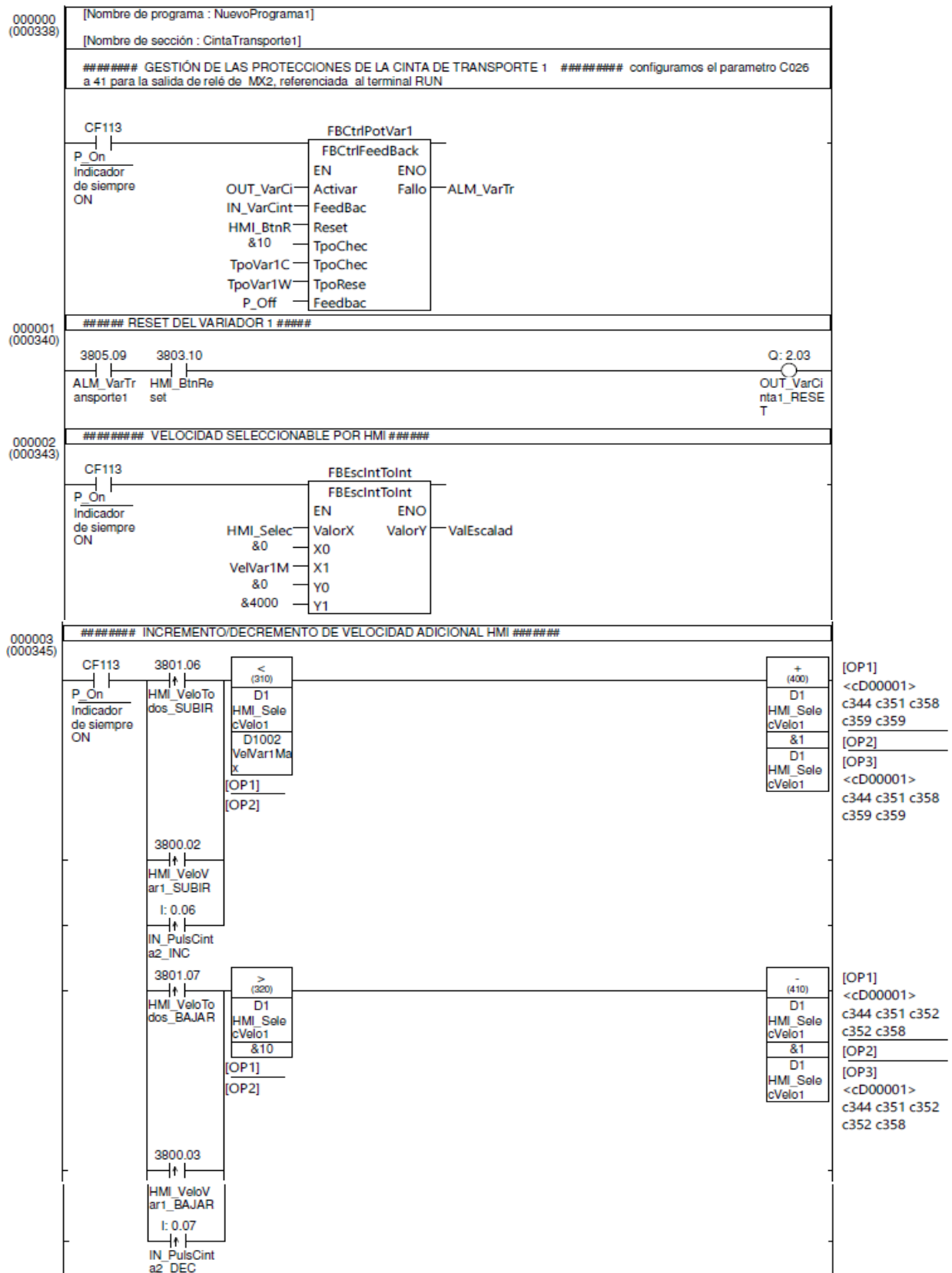
## 2.10 Electroválvula 2

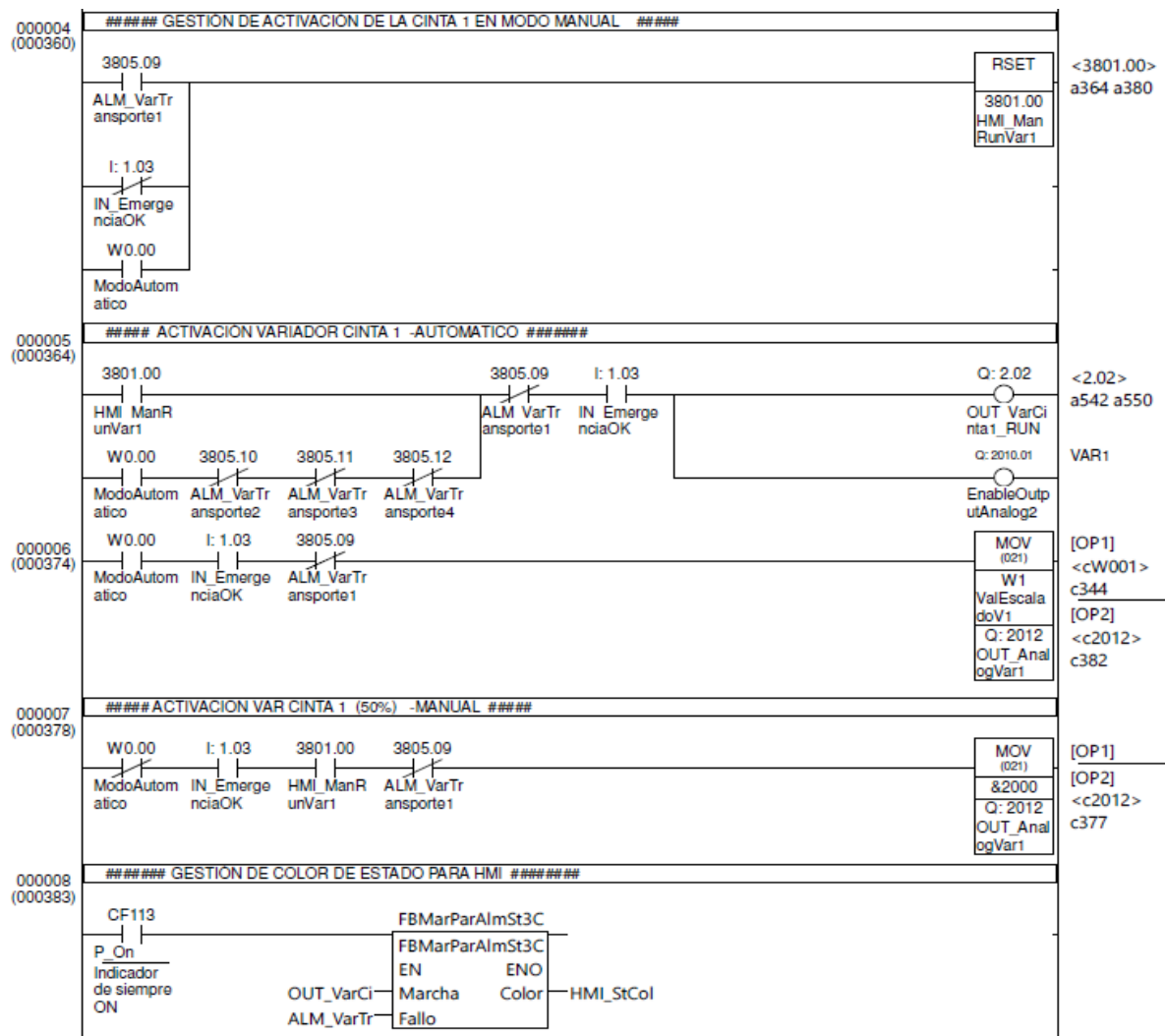




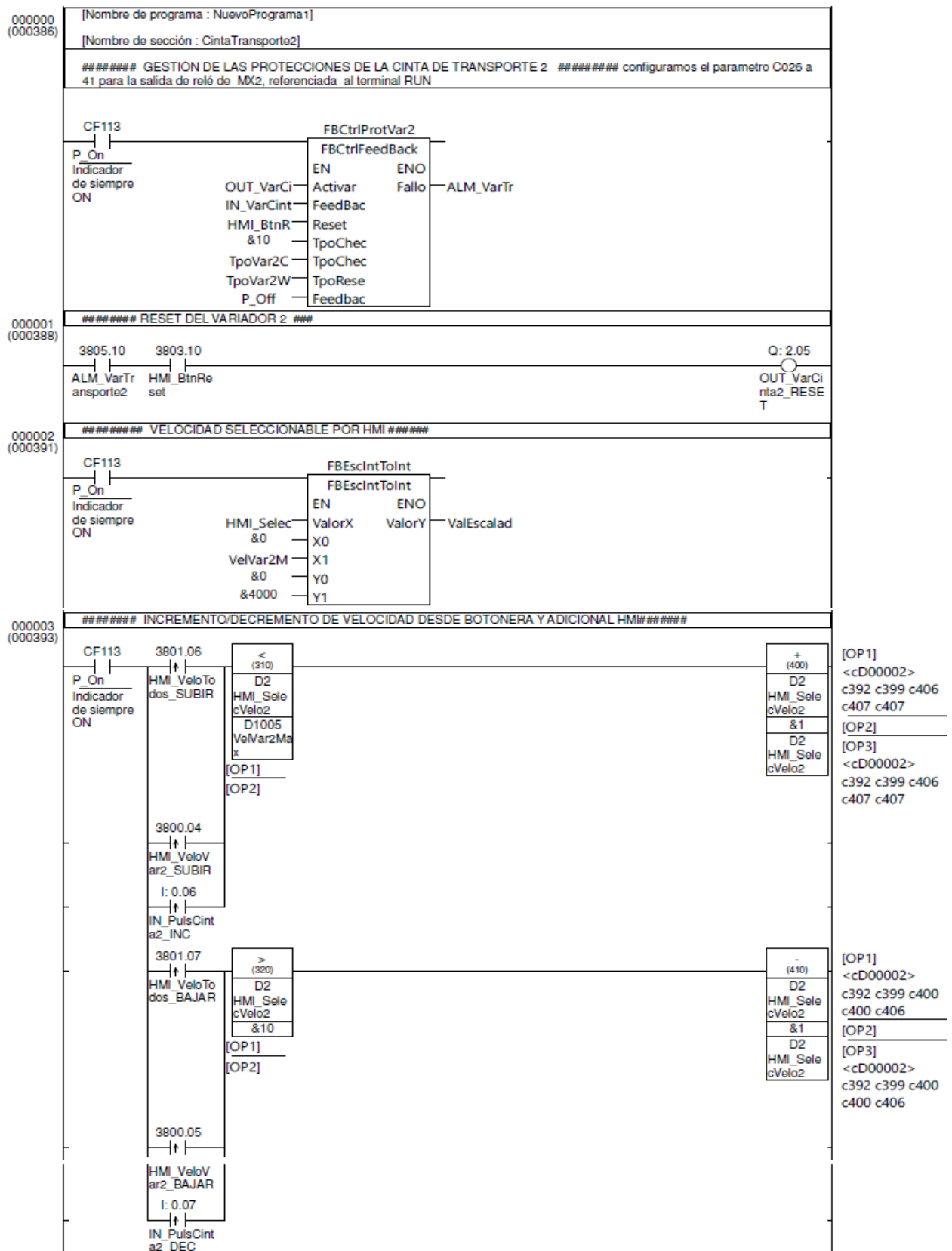


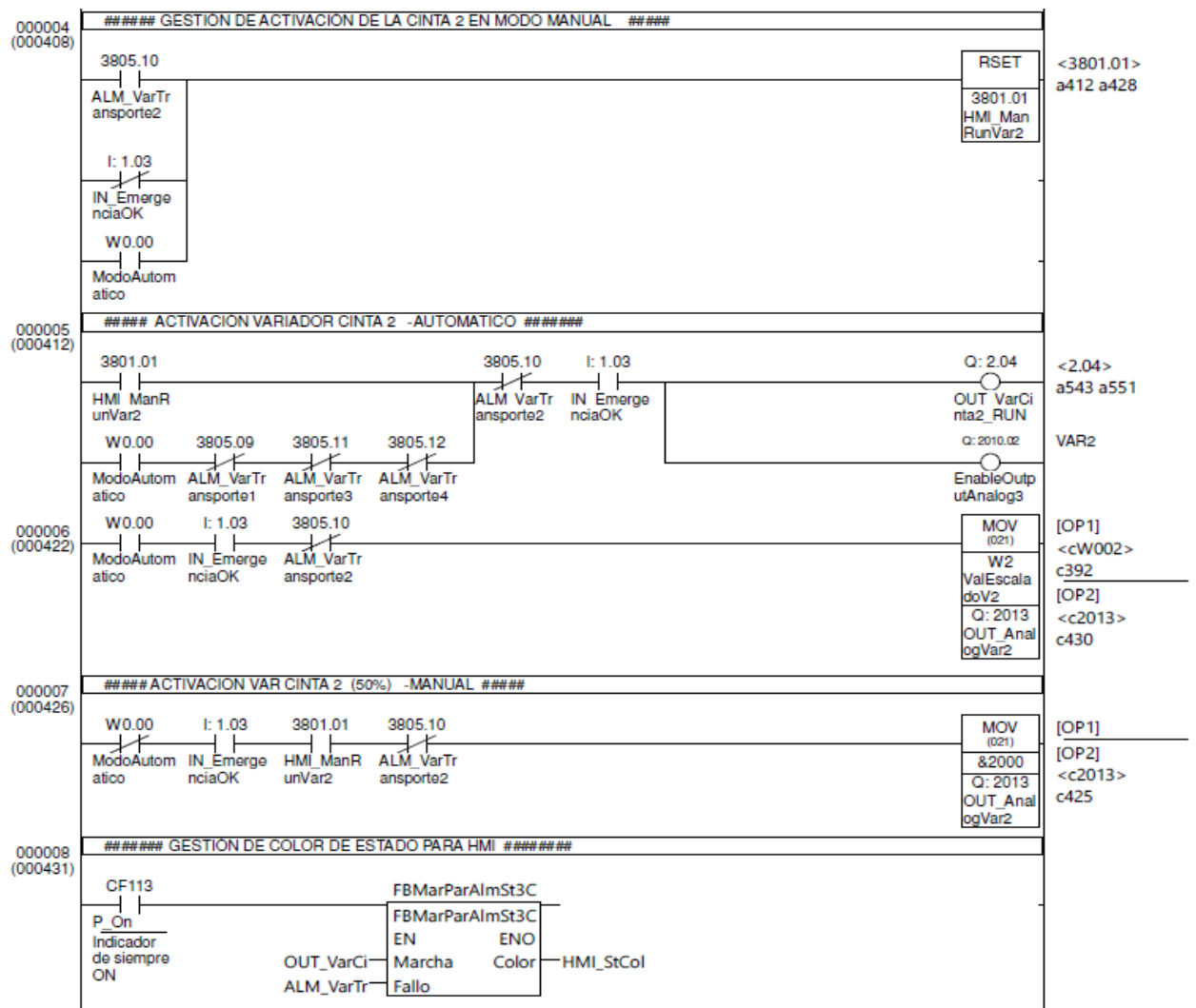
## 2.11 Cinta transporte 1



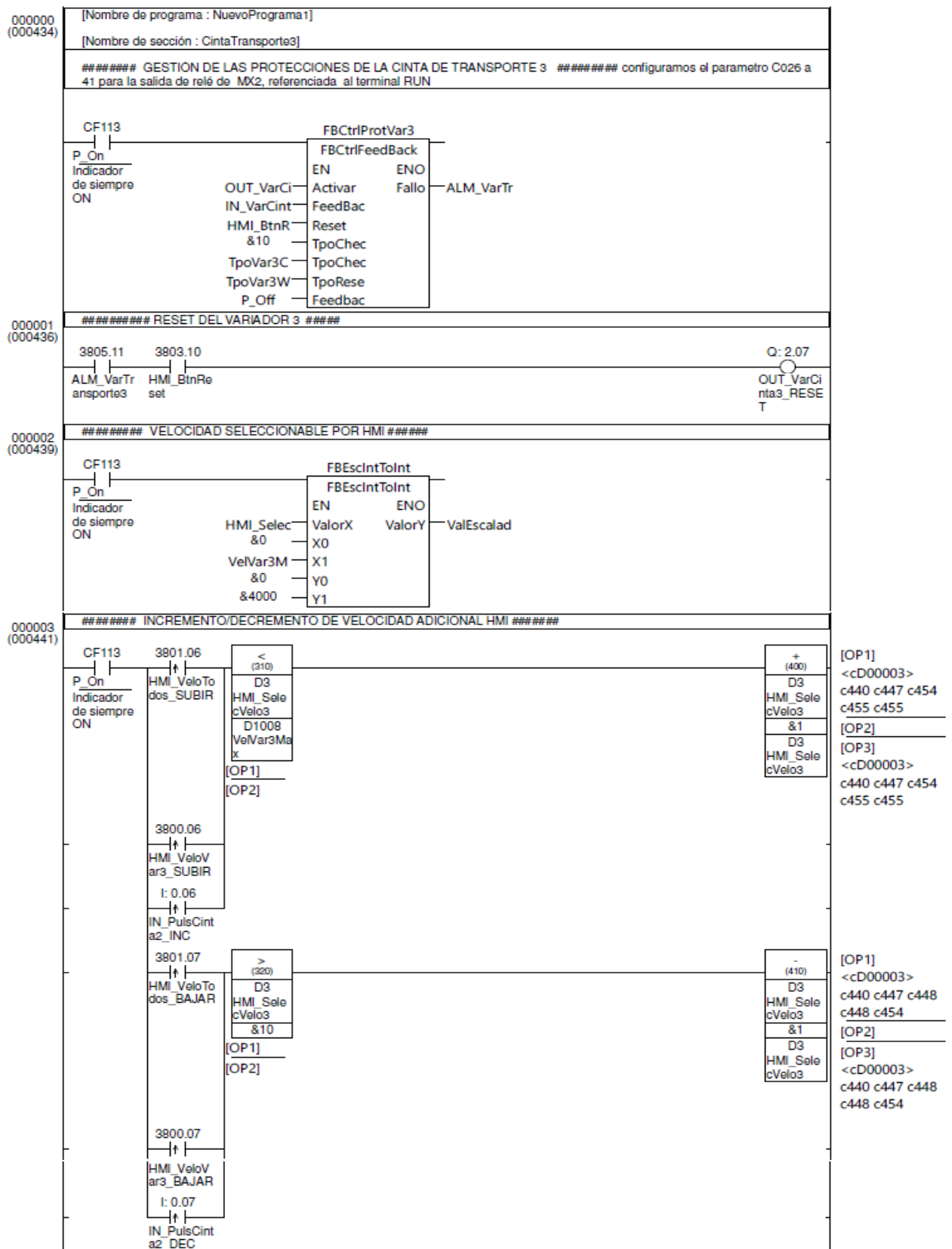


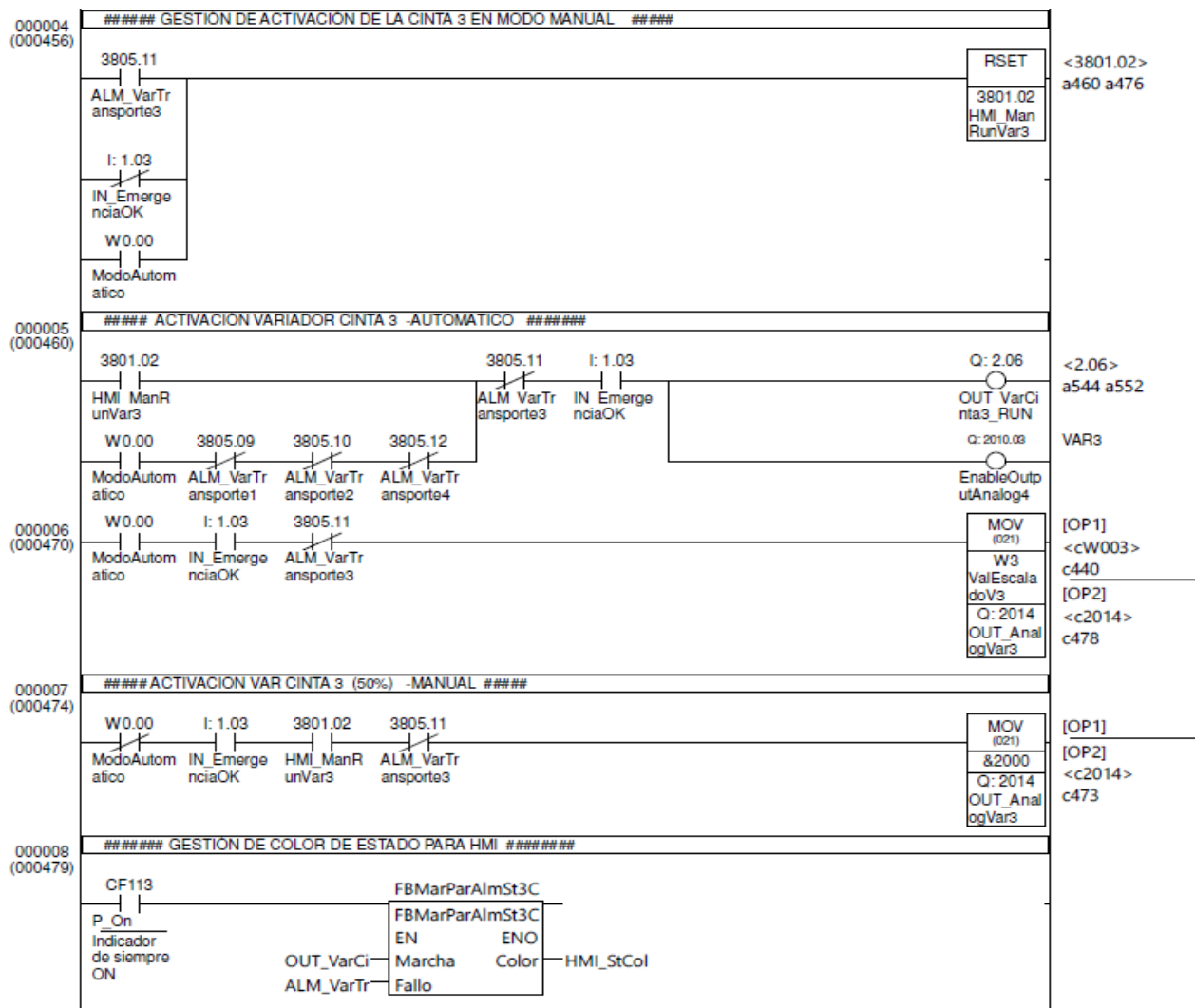
## 2.12 Cinta transporte 2



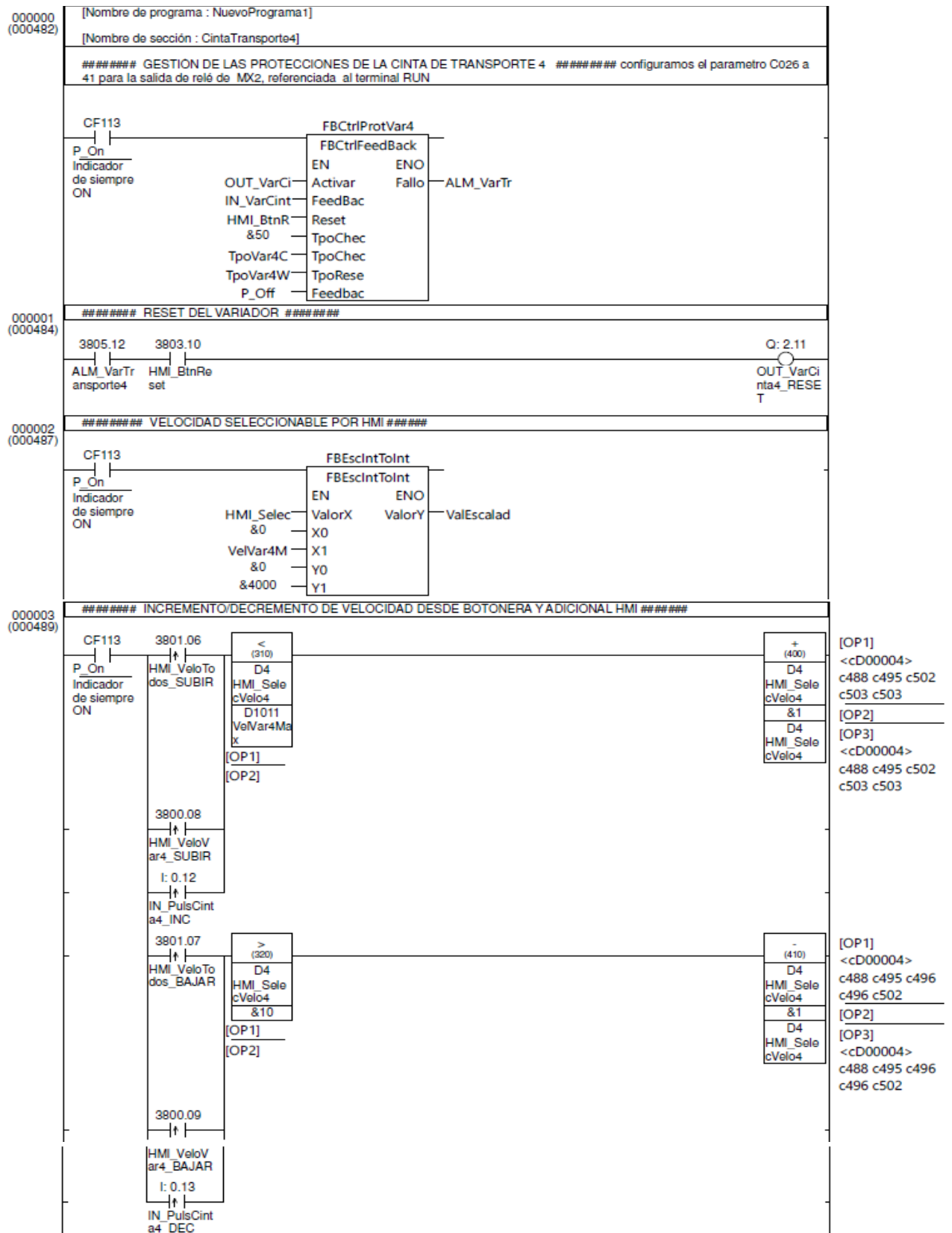


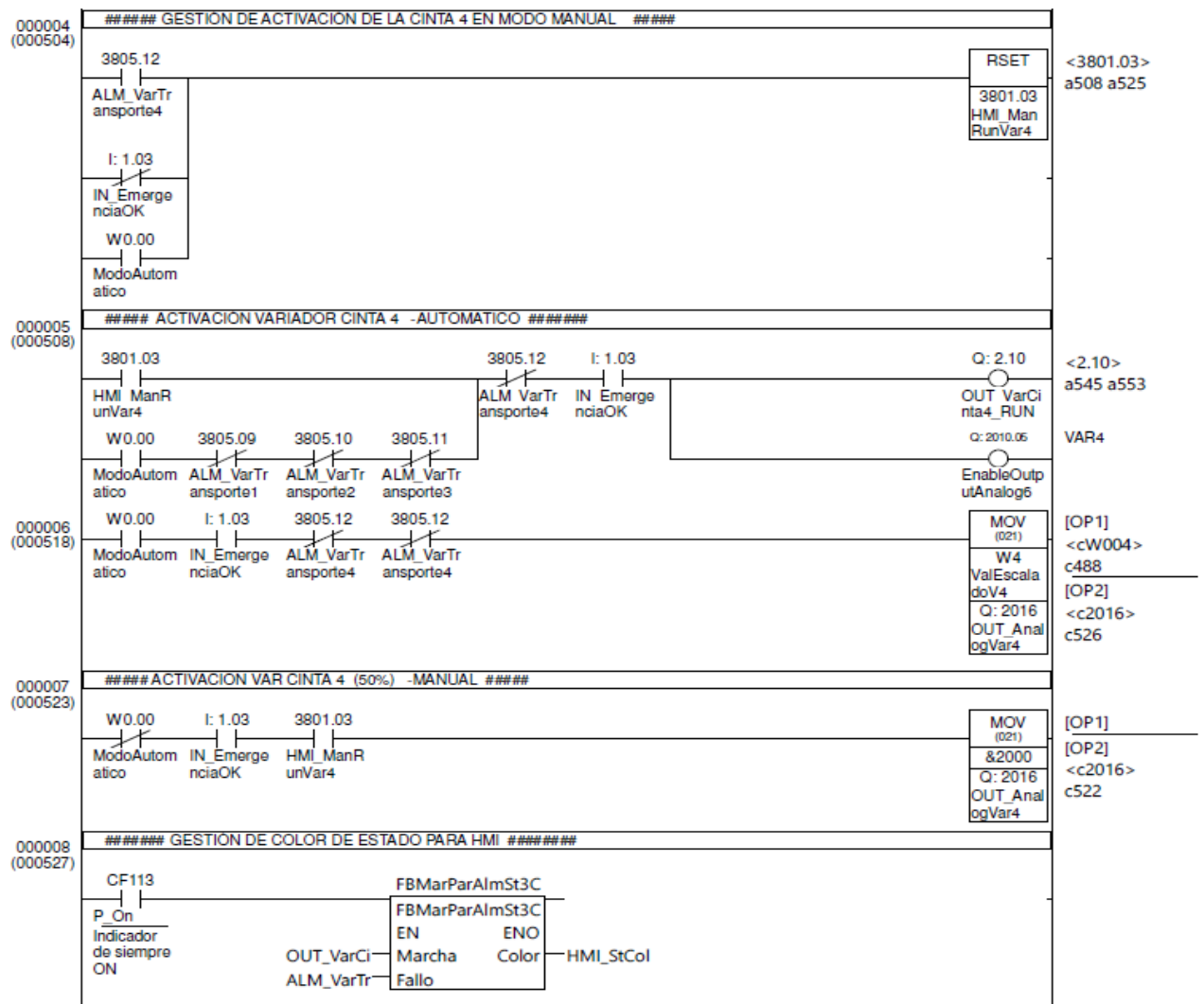
## 2.13 Cinta transporte 3





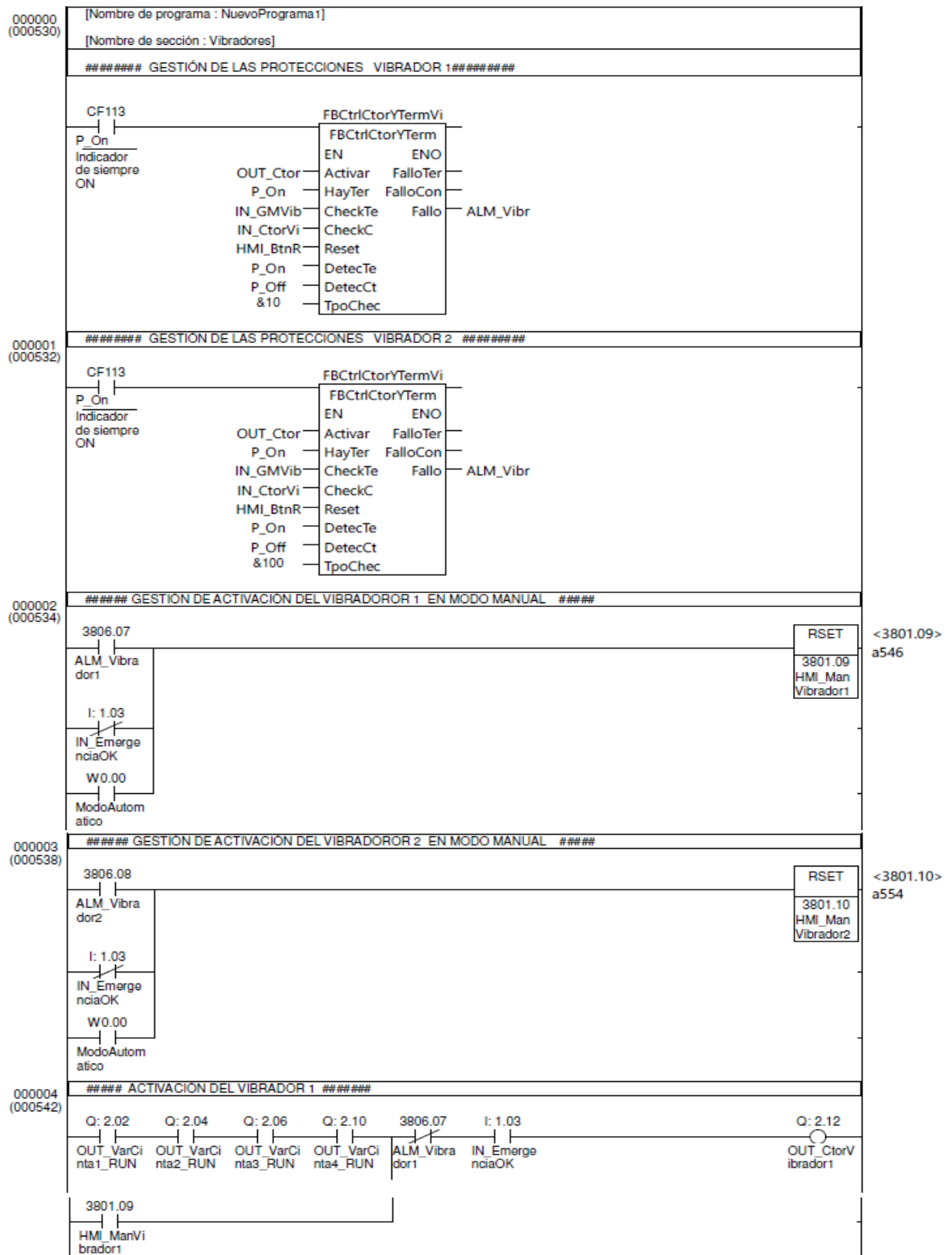
## 2.14 Cinta transporte 4

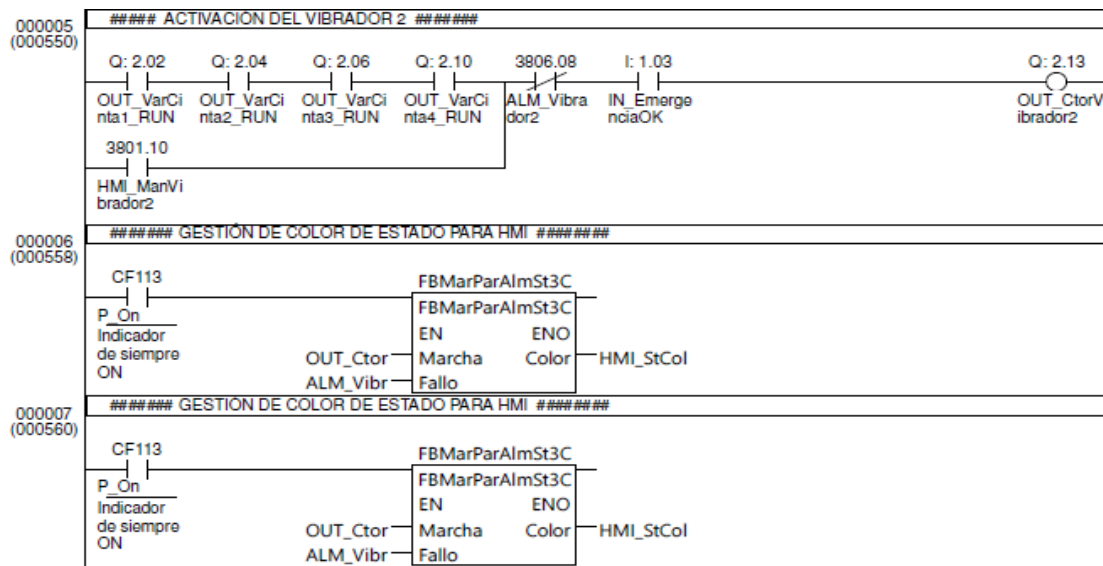






## 2.15 Vibradores





2.16 Fin

