Labview OPC UA Pol

Created	@March 29, 2024 10:18 AM
Tags	Electrolizadores S7-1200
Prerequisits	OPC UA

Introducción y objetivos

Desarrollo de un código estandarizado en SubVIs para lograr la lectura de datos vía comunicación OPC UA con el autómata S7-1200.

🔗 **S7-1200 Endpoint:** opc.tcp://10.4.210.54:4841

A continuación se lista la tabla de variables disponibles en el dispositivo:

▼ Tabla variables S7-1200

Variable		Tipo	Namespace	Node ID
CONTACTO PRAL		BOOL	4	3
TERMICO EXTRACTORES		BOOL	4	4
RELE SEGURIDAD		BOOL	4	5
SOBRETENSIONES		BOOL	4	6
ARRANQUE COMPRESSOR		BOOL	4	7
ESTADO COMPRESOR		BOOL	4	8
EV_1		BOOL	4	9
EV_2		BOOL	4	10
EV_3		BOOL	4	11
EV_4		BOOL	4	12
EV_5		BOOL	4	13
EV_6		BOOL	4	14
EXTRACTOR 1		BOOL	4	21
ECTRACTOR 2		BOOL	4	22
BALIZA VERDE		BOOL	4	25
BALIZA AMARILLA		BOOL	4	26
BALIZA ROJA		BOOL	4	27
ARRAY DE DATOS		ARRAY(UINT16)	4	28
	[0]	UINT16	4	29
	[1]	UINT16	4	30
	[2]	UINT16	4	31

	[3]	UINT16	4	32
	[4]	UINT16	4	33
	[5]	UINT16	4	34
	[6]	UINT16	4	35
	[7]	UINT16	4	36
	[8]	UINT16	4	37

Además, a través del S7-1200 también podemos conseguir datos de los electrolizadores. A continuación encontramos la tabla de variables de estos dispositivos enlazados:

▼ ELY_1.1

Variable			Tipo	Namespace	Node ID
ELY_1.1			----------"DATOS_ENVIO"."ELY_1.1"----------		
	WARNING		ARRAY UINT 16	4	41
		1	UINT 16	4	42
	ERROR		ARRRAY DE UNIT16	4	43
		1	UINT 16	4	44
	ESTADO		ARRAY DE UINT16	4	45
		1	UINT16	4	46
	PRODUCCION		ARRAY DE FLOAT	4	47
		1	FLOAT	4	48
		2	FLOAT	4	49
	MEDIDAS		ARRAY DE FLOAT	4	50
		1	FLOAT	4	51
		2	FLOAT	4	52
		3	FLOAT	4	53
		4	FLOAT	4	54
		5	FLOAT	4	55
		6	FLOAT	4	56
		7	FLOAT	4	57
		8	FLOAT	4	58
		9	FLOAT	4	59
		10	FLOAT	4	60
		11	FLOAT	4	61
	STAR/STOP		ARRAY DE UINT 16	4	62
		1	UINT 16	4	63

▼ ELY_1.2

ELY_1.2			---------- "DATOS_ENVIO"."ELY_1.2"- ----------		
	WARNING		ARRAY UINT 16	4	67
		1	UINT 16	4	68
	ERROR		ARRRAY DE UNIT16	4	69
		1	UINT 16	4	70
	ESTADO		ARRAY DE UINT16	4	71
		1	UINT16	4	72
	PRODUCCION		ARRAY DE FLOAT	4	73
		1	FLOAT	4	74
		2	FLOAT	4	75
	MEDIDAS		ARRAY DE FLOAT	4	76
		1	FLOAT	4	77
		2	FLOAT	4	78
		3	FLOAT	4	79
		4	FLOAT	4	80
		5	FLOAT	4	81
		6	FLOAT	4	82
		7	FLOAT	4	83
		8	FLOAT	4	84
		9	FLOAT	4	85
		10	FLOAT	4	86
		11	FLOAT	4	87
	STAR/STOP		ARRAY DE UINT 16	4	88
		1	UINT 16	4	89

▼ ELY_1.3

ELY_1.3			---------- "DATOS_ENVIO"."ELY_1.3"- ----------		
	WARNING		ARRAY DE UINT16	4	93
		1	UINT 16	4	94
	ERROR		ARRRAY DE UNIT16	4	95
		1	UINT 16	4	96
	ESTADO		ARRAY DE UINT16	4	97
		1	UINT16	4	98
	PRODUCCION		ARRAY DE FLOAT	4	99
		1	FLOAT	4	100
		2	FLOAT	4	101

	MEDIDAS		ARRAY DE FLOAT	4	102
		1	FLOAT	4	103
		2	FLOAT	4	104
		3	FLOAT	4	105
		4	FLOAT	4	106
		5	FLOAT	4	107
		6	FLOAT	4	108
		7	FLOAT	4	109
		8	FLOAT	4	110
		9	FLOAT	4	111
		10	FLOAT	4	112
		11	FLOAT	4	113
	STAR/STOP		ARRAY DE UINT 16	4	114
		1	UINT 16	4	115

▼ ELY_2.1

ELY_2.1			---------- "DATOS_ENVIO"."ELY_2.1"- ----------		
	WARNING		ARRAY UINT 16	4	119
		1	UINT 16	4	120
	ERROR		ARRRAY DE UNIT16	4	121
		1	UINT 16	4	122
	ESTADO		ARRAY DE UINT16	4	123
		1	UINT16	4	124
	PRODUCCION		ARRAY DE FLOAT	4	125
		1	FLOAT	4	126
		2	FLOAT	4	127
	MEDIDAS		ARRAY DE FLOAT	4	128
		1	FLOAT	4	129
		2	FLOAT	4	130
		3	FLOAT	4	131
		4	FLOAT	4	132
		5	FLOAT	4	133
		6	FLOAT	4	134
		7	FLOAT	4	135
		8	FLOAT	4	136
		9	FLOAT	4	137
		10	FLOAT	4	138
		11	FLOAT	4	139

	STAR/STOP		ARRAY DE UINT 16	4	140
		1	UINT 16	4	141

▼ ELY_2.2

ELY_2.2			----------"DATOS_ENVIO"."ELY_2.2"----------		
	WARNING			4	145
		1	UINT 16	4	146
	ERROR		ARRRAY DE UNIT16	4	147
		1	UINT 16	4	148
	ESTADO		ARRAY DE UINT16	4	149
		1	UINT16	4	150
	PRODUCCION		ARRAY DE FLOAT	4	151
		1	FLOAT	4	152
		2	FLOAT	4	153
	MEDIDAS		ARRAY DE FLOAT	4	154
		1	FLOAT	4	155
		2	FLOAT	4	156
		3	FLOAT	4	157
		4	FLOAT	4	158
		5	FLOAT	4	159
		6	FLOAT	4	160
		7	FLOAT	4	161
		8	FLOAT	4	162
		9	FLOAT	4	163
		10	FLOAT	4	164
		11	FLOAT	4	165
	STAR/STOP		ARRAY DE UINT 16	4	166
		1	UINT 16	4	167

▼ ELY_2.3

ELY_2.3			----------"DATOS_ENVIO"."ELY_2.3"----------		
	WARNING		ARRAY UINT 16	4	171
		1	UINT 16	4	172
	ERROR		ARRRAY DE UNIT16	4	173
		1	UINT 16	4	174
	ESTADO		ARRAY DE UINT16	4	175
		1	UINT16	4	176

	PRODUCCION		ARRAY DE FLOAT	4	177
		1	FLOAT	4	178
		2	FLOAT	4	179
	MEDIDAS		ARRAY DE FLOAT	4	180
		1	FLOAT	4	181
		2	FLOAT	4	182
		3	FLOAT	4	183
		4	FLOAT	4	184
		5	FLOAT	4	185
		6	FLOAT	4	186
		7	FLOAT	4	187
		8	FLOAT	4	188
		9	FLOAT	4	189
		10	FLOAT	4	190
		11	FLOAT	4	191
	STAR/STOP		ARRAY DE UINT 16	4	192
		1	UINT 16	4	193

Conociendo estas direcciones, cabe destacar que hemos decidido usar el OPC UA Toolkit de Labview, debido a que NI OPC Servers requiere una licencia extra que no poseemos.

Librerías custom

Para poder crear un dashboard o Scada más adelante, hemos desarrollado una serie de librerías en formato SubVIs para realizar la lectura de los datos de forma encapsulada.

Cabe destacar que todas estas SubVIs son de **una sola ejecución** y que, por lo tanto, deberán estar en un **bucle externo** en el codigo **main** para poder **leer los datos de manera recurrente**.

Empezaremos entendiendo el flujo de programa del código main de ejemplo y, a partir de aquí, iremos desglosando las SubVIs para entender su funcionamiento y su correcta forma de empleo.

Ejemplo de codigo main

Aunque la figura 1 parezca compleja, el flujo de código es simple. Empezamos estableciendo conexión con el servidor S7-1200 a través de OPC UA sin seguridad, y procedemos a entrar en el bucle. Una vez dentro del bucle, leemos los datos de cada dispositivo de manera secuencial: S7-1200, ELY1.1, ELY1.2, ELY1.3...ELY2.3. Finalmente, si pulsamos el botón de Stop, el bucle termina y el se cierra la conexión OPC UA con el autómata.

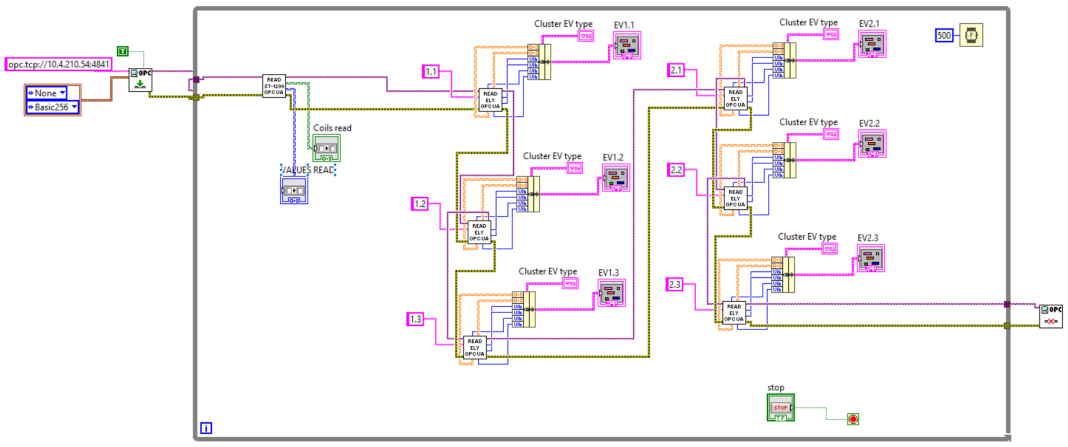

Figura 1 - Diagrama de Bloques del código main de ejemplo

En la figura 2 podemos ver parte del Front View. Como se trata de un código de ejemplo, simplemente tenemos una visualización de todos los datos que podemos leer agrupada por dispositivos. En un dashboard o Scada final, es posible que muchos de estos datos se visualicen de manera distinta, en gráficas o después de ser tratados. Es por eso que simplemente nos hemos centrado en mostrar la capacidad de lectura en este ejemplo.

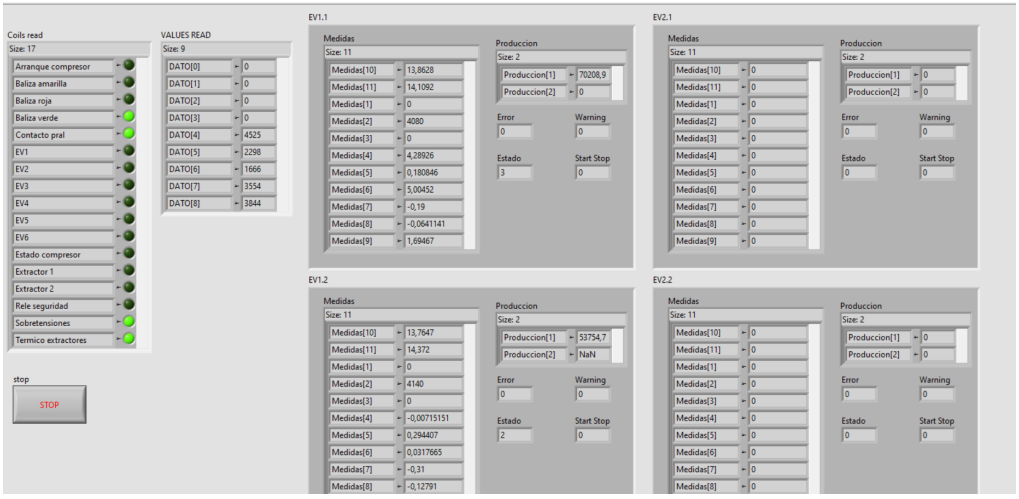

Figura 2 - Front View del código main de ejemplo

Lectura Datos S7-1200

Hemos encapsulado la lectura de datos del autómata en una SubVI (*"read_S7_1200 (SubVI).vi"*). Podemos ver el código en la figura 3, y el panel frontal en la figura 4.

Para empezar a entender más en profundidad esta SubVI, vamos a describir sus INPUTS y OUTPUS:

- **INPUTS:**

- Refnum in: enlace a la conexión OPC UA
- Error in: conexión al error out del bloque anterior del toolkit.

- **OUTPUTS:**
  - Coils read: mapa (diccionario) con los valores booleanos de lectura. En la figura 3 podemos observar el array de Keys (”Contacto pral”, “Termico extractores”, ...) cogido del nombre que toma la variable en el directorio del servidor OPC UA.
  - Values read: mapa (diccionario) con los valores UINT16 de lectura. En la figura 3 podemos ver que los Keys son genéricos por el momento (”Dato[0]”, “Dato[1]”, ...) porque aún falta por identificar a qué pertenece cada lectura.
  - Refnum out: enlace a conexión OPC UA.
  - Error out: errores de salida, en caso que haya.

Además, podemos ver que hacemos uso de dos subVIs más:

- READ TO MAP BOOL (*”read_to_map(SubVI).vi”*)
- READ TO MAP UINT16 (*”read_to_map_2(SubVI).vi”*)

Estas las detallaremos más adelante. Resumidamente, cogen el array de objetos cluster de lectura, el array de Keys y los sincronizan en un map (diccionario) para facilitar el acceso a las variables.

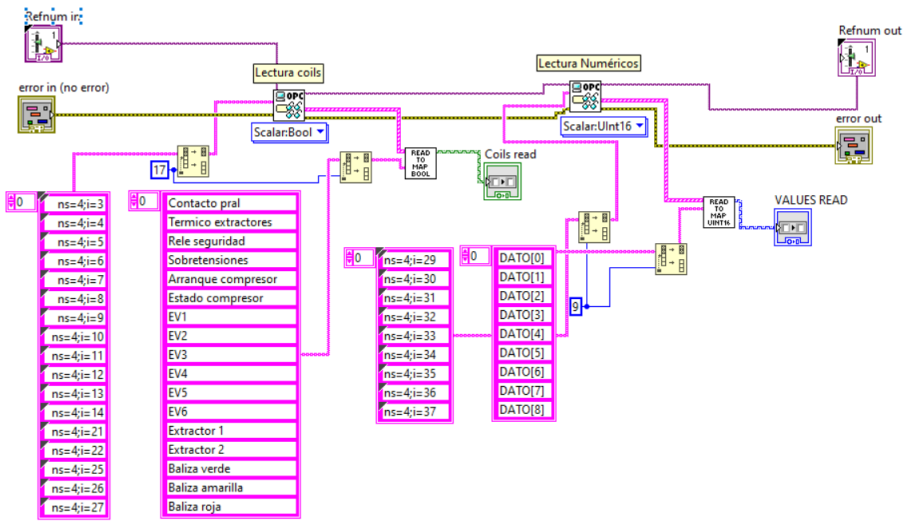

Figura 3 - Diagrama de bloques SubVI (*”read_S7_1200 (SubVI).vi”*)

> **Detalle**: Nótese que el array de strings está conectado a un split con la longitud deseada del array. Esto es debido a que es muy fácil confundirse y añadir un string vacío al array, hecho que desincronizaría los valores de lectura con las keys.

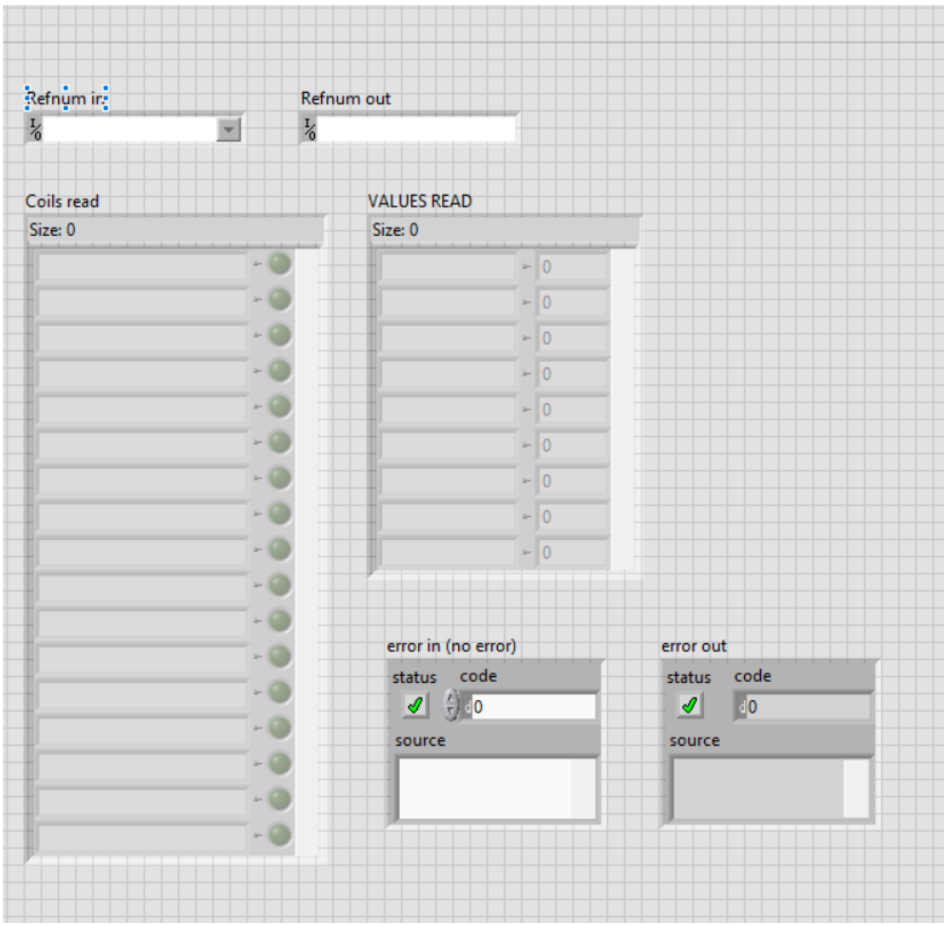

Figura 4 - Panel Frontal SubVI (*"read_S7_1200 (SubVI).vi"*)

Lectura Datos Electrolizadores

Hemos encapsulado la lectura de datos de los electrolizadores en una SubVI (*"ELY (SubVI).vi"*). Podemos ver el código en la figura 5, y el panel frontal en la figura 6.

Para empezar a entender más en profundidad esta SubVI, vamos a describir sus INPUTS y OUTPUS:

- **INPUTS:**
  - Electrolizador: Número de electrolizador "X.X" en formato string (Ej: "1.1"). Selecciona el caso en el Switch case para introducir las constantes de direcciones de los registros en el servidor OPC UA S7-1200.
  - Refnum in: enlace a la conexión OPC UA
  - Error in: conexión al error out del bloque anterior del toolkit.
- **OUTPUTS:**
  - Estado: entero que designa en qué estado se encuentra el electrolizador.
  - Error: entero que designa si existe caso de error en el electrolizador.
  - Warning: entero que designa si existe caso de warning en el electrolizador.
  - StartStop: entero que dice si el electrolizador está encendido o en pausa.

- Medidas: Mapa de medidas relacionado con keys genéricas por el momento. *Falta por relacionar el valor de las medidas con su significado.*
- Producción: Mapa de medidas de producción con keys genéricas por el momento. *Falta por relacionar el valor de las medidas con su significado.*
- Refnum out: enlace a conexión OPC UA.
- Error out: errores de salida, en caso que haya.

Además, podemos ver que hacemos una subVI más:

- READ TO MAP FLOAT (*"read_to_map_3(SubVI).vi"*)

Esta la detallaremos más adelante. Resumidamente, coge el array de objetos cluster de lectura, el array de Keys y los sincroniza en un map (diccionario) para facilitar el acceso a las variables.

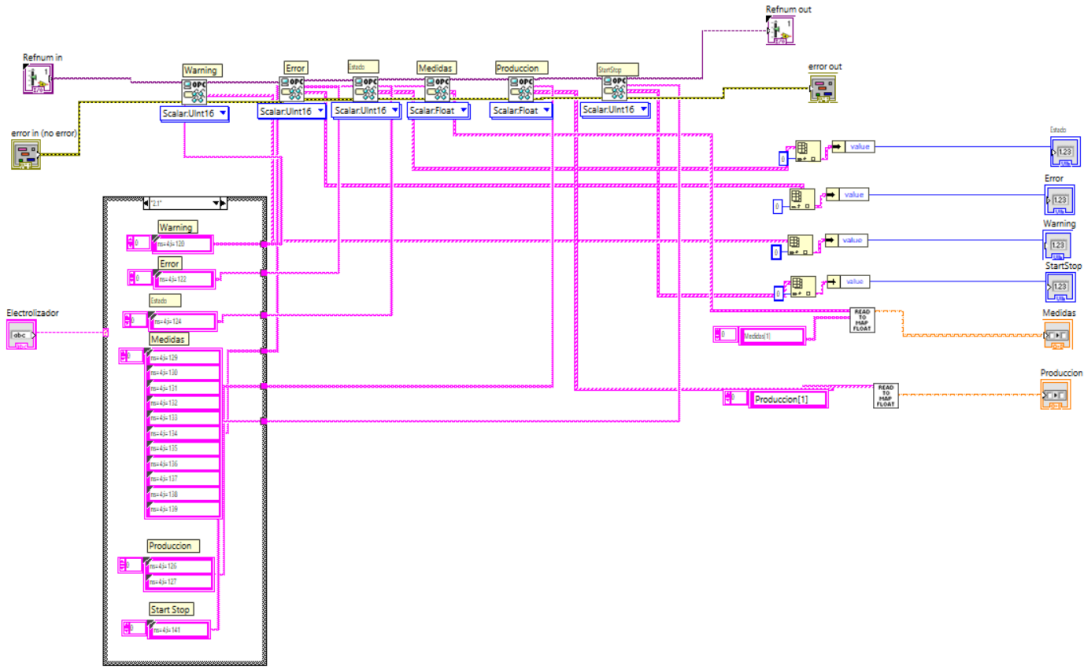

Figura 5 - Diagrama de Bloques SubVI (*"ELY (SubVI).vi"*)

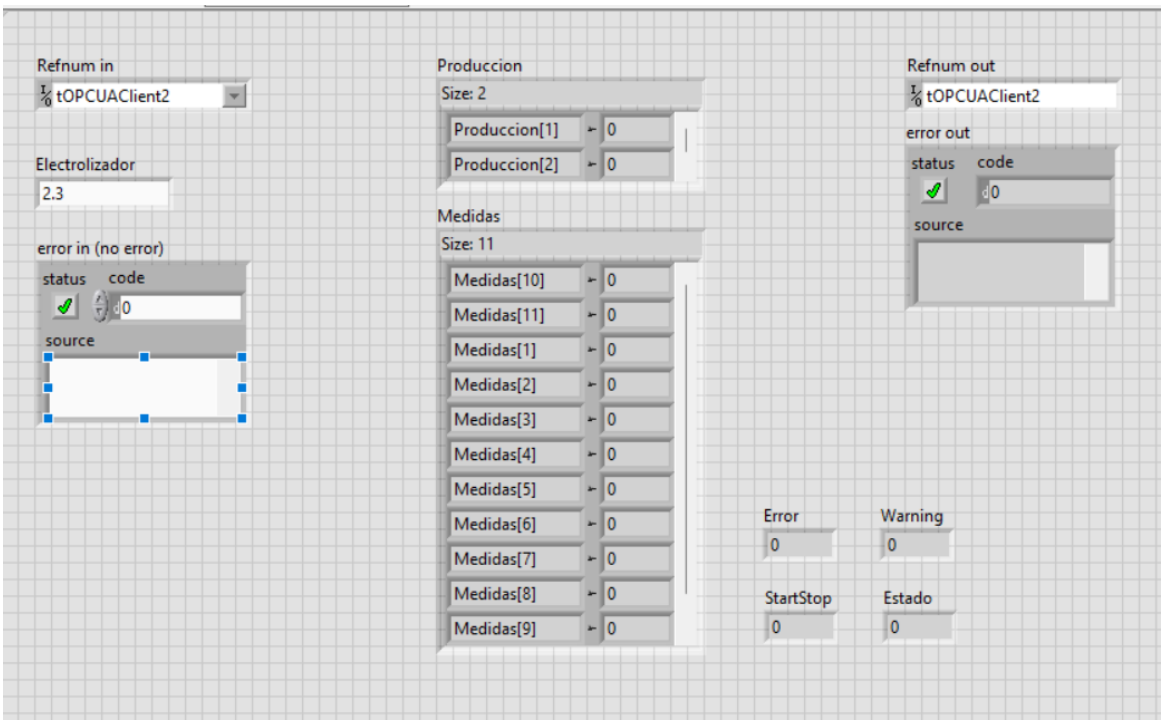

Figura 6 - Panel Frontal SubVI (*"ELY (SubVI).vi"*)

Podemos observar los mapas Producción y Medidas con sus keys en la figura 6

READ TO MAP

Se trata de un conjunto de 3 SubVIs que siguen el mismo principio, con la diferencia de que cada una de ellas está implemetada para un tipo de dato distinto. Idealmente, deberíamos refactorizar estas 3 SubVIs en una de sola que pueda seleccionar el tipo de variable a tratar. Además, podríamos implementar tratamiento de errores. Por el momento, el funcionamiento es básico, simple y funcional.

Estas SubVIs están disenyadas para tratar el objeto que devuelven las SubVIs de múltiple lectura del OPC UA Toolkit. Estas devuelven un array de clusters con las lecturas realizadas, dentro de cada cluster hay una serie de varibales (Node id, Value, etc). "Read to map" coge el value de cada objeto cluster del array y lo enlaza con una key de tipo string, con el mismo indice en el array que el objeto de lectura. De esta manera, podemos tratar los valores de medidas en un formato mucho más cómodo, pues se encuentran enlazadas en un map (diccionario).

- **INPUTS:**
  - Read Input Cluster: array de clusters de lectura.
  - Key Array: array de strings.

Los dos inputs deben tener una misma longitud

- **OUTPUTS:**
  - Output map: map(diccionario) con los valores de lectura y las keys del array enlazados según su índice (posición) en el array.

A continuación podemos ver el front panel y el diagrama de bloques de cada una de las variantes.

READ TO MAP BOOL (read_to_map(SubVI).vi)

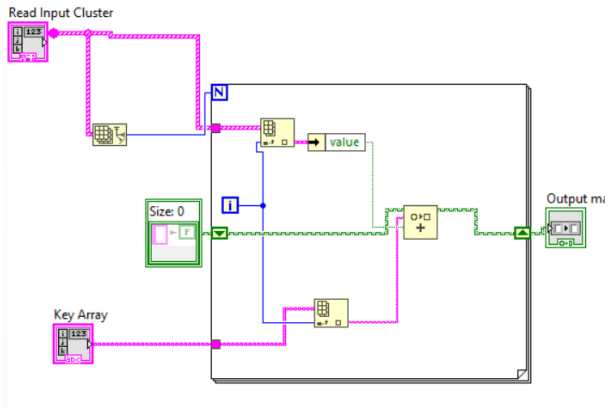

Figura 7 - Diagrama de bloques SubVI (read_to_map(SubVI).vi)

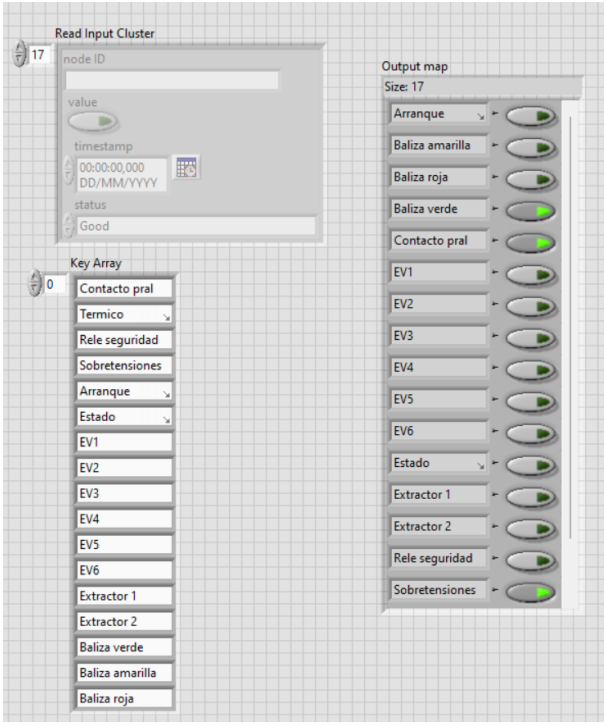

Figura 8 - Front Panel SubVI (read_to_map(SubVI).vi)

READ TO MAP UINT16 (read_to_map_2(SubVI).vi)

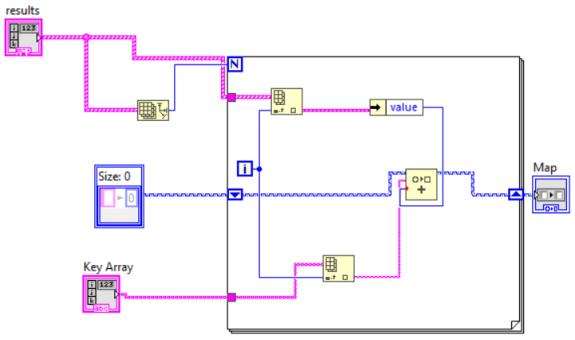

Figura 9 - Diagrama de bloques SubVI (read_to_map_2(SubVI).vi)

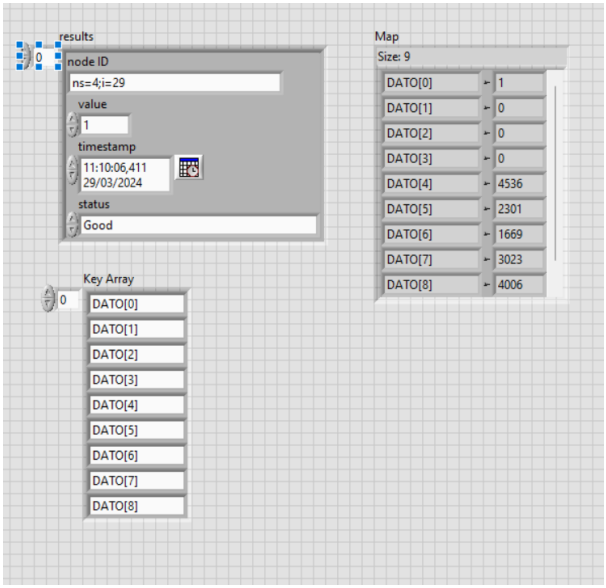

Figura 10 - Front Panel SubVI (read_to_map_2(SubVI).vi)

READ TO MAP FLOAT (read_to_map_3(SubVI).vi)

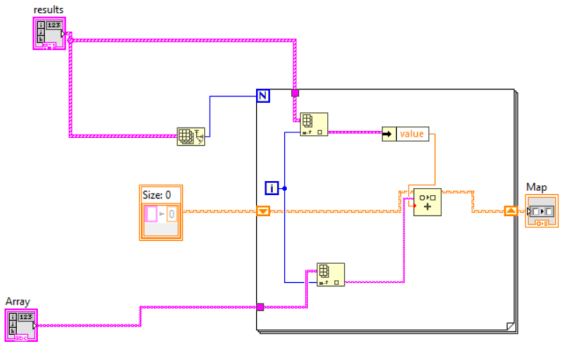

Figura 11 - Diagrama de bloques SubVI (read_to_map_3(SubVI).vi)

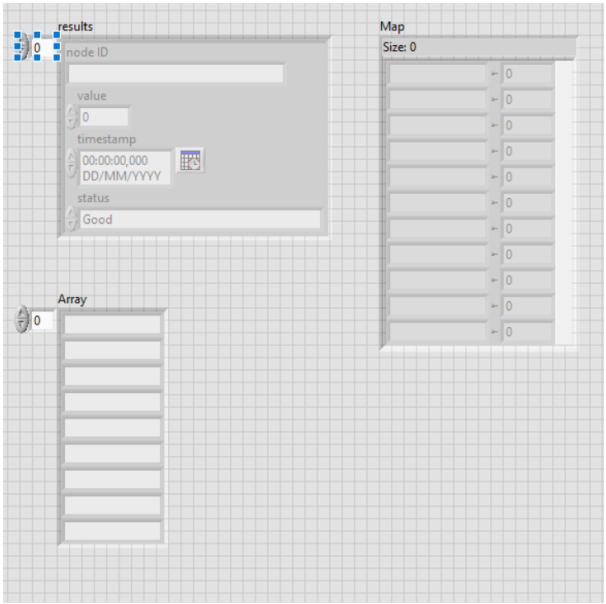

Figura 12 - Front Panel SubVI (read_to_map_3(SubVI).vi)

Conclusiones

Hemos logrado implementar una serie de librerías para leer los siguientes dispositivos al completo mediante OPC UA:

- S7-1200

- Conjunto electrolizadores (ELY1.1, ELY1.2, ..., ELY2.3)

Aún falta por implementar diversos features:

- Agregar capa de seguridad al server y client OPC UA
- Relacionar algunos de los datos obtenidos con su significado
- Implementar este código en un Dashboard o Scada.