EPICS using SLS s7plc driver

PLC configurations



National Synchrotron Radiation Research Center Radio Frequency Group Yu-Han Lin

In this documents, the following items are used:
Software1) EPICS Base 3.14.8.2 (http://www.aps.anl.gov/epics/base/R3-14/index.php)
2) Step 7 – SIMATIC Manager V5.2
3) SLS s7 driver (http://epics.web.psi.ch/software/s7plc/)
Hardware1) Siemens S7-300 PLC
2) Siemens CP343 Communication module

Configuration	of Siemens S7 PLC
<pre>Step 1: Create an empty project. Execute the "SIMATIC Manager" and create a new project called "EPICS_PLC" as shown in Figure 1.</pre>	SIMATIC Manager Pic Yiew Option: Window Help Die State State
	Figure 1

SIMATIC Manager - [EPICS_PL	C C:\Program Files\Siemens\Step7\s7pri	ui	Step 2: Insert a PLC station in project.
SIMATIC Manager - [EPICS PL File Edit Insert PLC View Or EPICS PLC Cut Copy Paste Delete Insert New Object PLC Rename Object Properties	C CAProgram Files/Siemens/Siep7/s7pro ptions Window Help P	F	Step 2: Insert a PLC station in project. On the "EPICS_PLC" project, click the right button to insert a SIMATIC Station as shown in Figure 2. The result is shown in Figure 3.
	SIMATIC S5 PG/PC MPI PROFIBUS Industrial Ethernet PTP S7 Program M7 Program	E	SIMATIC Manager - [EPICS_PLC C:Program File File Edit Insert PLC View Options Window Help C C R R R R R R R R R R R R R R R R R R

Figure 2

Figure 3



Step 3: Configure the hardware configuration of the PLC.

In the SIMATIC Station as shown in Figure 3, doubleclick "Hardware" : this operation leads you into the hardware configuration tool (figure 4). According to your PLC hardware, configure the modules at the right position. As all configurations are finished as shown in Figure 4, press button 1 to compile the configuration code, and then press button 2 to download the configurations into the PLC.



Figure 4



Step 4: Configure the connection profile of the PLC.

When the hardware is properly set up, the sections "CPU 318-2" and "CP 343-1" are automatically created in the SIMATIC Station as shown in Figure 5. In the "CPU 318-2" section, double-click the "Connection" to configure the PLC connections.

Figure 5







The network configuration is similar to the hardware configuration. You need only to drag a station or subnet into the project, and the entry becomes automatically created.

Here we seek to create an Industrial Ethernet subnet: so we drag the "Industrial Ethernet" into the project as shown in Figure 6. Figure 7 shows the result.

Figure 6



Figure 8

National Synchrotron Radiation Research Center, RF Group, Yu-Han Lin







Figure 11



Step 5: Create a communication channel for EPICS.

There is no communication channel that can serve for data transmission / receiving. Select the CPU module; the communication channel becomes displayed at the bottom of the Netpro window.

To create a new communication channel, click the right button on the CPU module, and select "Insert New Connection". As illustrated in Figure 12, a configuration window pops up.



6/1 EPICS using SLS s7plc driver

Insert New Connection	
Connection Partner	Use the "Connection Partner" group to specify the system or device with which PLC is going to communicate. For PLC to communicate with EPICS IOC, choose "Unspecified". The "Connection" group is used to specify the communication protocol of PLC. Choose "TCP connection" for EPICS IOC communication, then press the "OK" button. A notification window will appear as shown in Figure 13; press "OK" again to ignore the message.
Project: <u>Station:</u> (Unspecified) <u>M</u>odule: <u>M</u>odule: 	Insert New Connection (2443:172)
Connection Type: TCP connection Image: Display properties before inserting	Connections outside subnets are possible. Please check any router addresses, if required.
OK <u>Apply</u> Cancel Help	Help



Figure 13

The configuration program leads you to configure the property of the "TCP connection". As demonstrated in Figure 14, the "General Information" tab shows the ID and LADDR parameters. These two parameters are the identification number of this communication channel, and are used later.

Properties - TCP	connection	×
Options Gener	Overview al Information	Status Information Addresses
-Local Endpoint		-Block Parameters
ID (hex):	0001 A050 💌	1 — ID }
<u>N</u> ame:	ICP connection1	W#16#0100 - LADDR
Via <u>C</u> P:	CP 343-1 - (R0/\$\$4)	(more of
	<u>R</u> oute	
☐ <u>A</u> ctive com	ection establishment	
Use ETP pr	otocol	
OK		Cancel Help

Figure 14



The "Addresses" tab is used to specify the ip and port parameters of the local (PLC) and the partner (IOC). The local ip address has been set at step 4 (Figure 8). The default local port is 2000.

Do NOT specify the ip address and port of partner. If these two parameters were specified, the communication between PLC and IOC might fail.

Properties - TCP c	onnection		X
Options Genera	Overview I Information	Status Informa Addresses	tion
Ports from 1025 th (For further ports,	ırough 65535 are avail: refer to online help)	able.	
IP (dec): <u>P</u> ORT (dec):	Local 140.110.205.61 2000	Partner	
OK		Cancel	Help

Figure 15

Broperties - ICP connection X General Information Addresses Options Overview Local Mode: Send/Recv Image: Send/Recv OK Cancel Help	The "Options" tab is used to specify the communication mode of PLC. Set the mode to be "Send/Recv". Press "OK": the configuration of TCP connection is then finished.
--	---

Figure 16







Figure 17

Figure 18

Step 6: Insert data blocks for transmission and receiving.

Back to the SIMATIC Manager. select the "Blocks" section in the tree. There is only one organization block "OB1" in the "Block" section as illustrated in Figure 19.

We need to create two data blocks: one serves for data transmission. and the other for data receiving.









Figure 20

In the Data Block properties	Properties - Data Block	×
configuration window (Figure 21), rename the data block "DB10", and set the type to be "Shared DB".	General - Part 1 General - Part 2 Calls Attributes Name and type: DB10 Shared DB Image: Calls Symbolic Name: Calls Shared DB Image: Calls	<u> </u>
Create another data block, and rename it "DB11" with type "Shared DB". We will later use "DB10" for data transmission and "DB11" for data receiving.	Symbol Comment: Created in Language: DB Project path: Storage location of project: C:\Program Files\Siemens\Step7\s7proj\EPICS_PL Code Interface Date created: 16/03/2007 11:47:43 Last modified: 16/03/2007 11:47:43 Comment: Image: Comment:	2:43
	OK	Help

Figure 21



As shown in Figure 22, two data blocks were created, but the contents of these two data blocks are not yet defined. We therefore need to define the contents of these two data blocks.

Double-click on "DB10"; a configuration window as Figure 23 pops up. Modify the contents of DB10, as shown in Figure 24.



Figure 22

🗮 LAD/STL/FED - [DB10	EPICS_PLCS	SIMATIC 30)(1)/CPU 318-2]	Open "DB11", and modify
🖬 <u>F</u> ile <u>E</u> dit Insert PLC I	<u>D</u> ebug <u>V</u> iew Opt	tions <u>W</u> indow	Help _ 🗗 🗙	the content of DB11 to be
		68 🚵		the same as DB10.
Address Name	Туре	Initial va	Comment	Save these modifications,
0.0	STRUCT			and close the configura-
+0.0 DB_VAR	INT	0	Temporary placeholder variable	tion window.
=2.0	END_STRUCT			
Press F1 to get Help.	} 2∷Info ∧	3: Cross-refe	rences λ 4: Address info. λ	

Note that you can define the contents of DB10 and DB11 to meet your data format that you seek to transmit and to receive. Figure 24 is just illustrated as an example.

Figure 23

Address	N	lame	Туре	Initial	Vč	Comment
*0.0			STRUCT			
+0.0		Contents	ARRAY[0119]			Temporary placeholder vari
*2.0			WORD			
=240.0			END_STRUCT			

Figure 24





Figure 27



12/1 EPICS using SLS s7plc driver

🗱 LAD/STL/FBD - [FC1 EPIC	PLC/SIMATIC 300(1)/CPU 318-2]		
🖬 File Edit Insert PLC Debug	<u>V</u> iew <u>O</u> ptions <u>W</u> indow <u>H</u> elp	-	a ×
		┣■ № ┼┼┼-О┌─└→└─ №	
		Contents Of: EnvironmentUnterface'	
	- Interface	Name	
🖅 FB6 WRITE CP_300 🔼	IN IN		
FB8 USEND CP300PBK			
FB12 BSEND CP300PBC			
FB13 BRCV CP300PBK	FCl : Title:	`	
EB14 GET CP300PBK	Comment:		
FB15 PUT CP300PBK	Network 1		
FB55 IP_CONFIG CP_3	Network 1 . Title.		
FC2 DP RECV CP 300			_
FC3 DP_DIAG CP_300	Lomment:		
FC4 DP CTRL CP 300			
FC5 AG_SEND CP_300	AN T 1		
FC7 AG LOCK CP 300	SD T 0		
FC8 AG_UNLOCK CP_	A T O		
E FC40 FTP_CONNECT C	L S5T#100MS		
FC41 FTP_STORE CP_1	= M 100.0		
FC42 FIF_REIRIEVE (- 11
FC44 FTP_QUIT CP_30	Network 2: Title:		
	Comment:		
AG_SEND/CP_300 T			
	Nei	twork 2	
B Program e B=Callstr			
			2
x			
		Harrison & F.M. Her, & C.Di., C.	2.0.
	INIO A 3: Cross-references A 4: A	aaress muo. A 5: Moarry A 6: Diagnostics A	7:00
Press F1 to get Help.	G	I offline Abs < 5.2 Nw 2 Ln 1 Insert	Chg //
	E: 20		

Figure 28

Figure 28 shows the programming editor of FC1. In network 1, the memory address M100.0 is turned on/off per 100 ms. We can use this memory address as the switch to turn on/off the data transmission.

In network 2, we need to call the sub-program "AG_SEND", which can be found in the "Program elements" tree. Note that the "AG_SEND" function depends on the module.

This function is used to send data. Double-click the "AG_SEND"; it becomes automatically added to the selected network.

Table 1

Net	work	1:	
AN	Т	1	
L	S5T	`#100MS	
SD	Т	0	
А	Т	0	
L	S5T	`#100MS	
SD	Т	1	
=	М	100.0	



National Synchrotron Radiation Research Center, RF Group, Yu-Han Lin





The editor will ask you to input the required parameters. Fill these rows with proper parameters, then save and quit.

Note:

- 1) The "ID" and "LADDR" are used to identify the connection channel; they must match the values in Figure 14.
- 2) For CP300 series, the maximum length of "AG_SEND" is bounded by 240 bytes.
- 3) In network 2, we use memory address M100.0 to turn on/off the transmission.
- 4) Data block (DB10) of length 240 bytes is sent.

Table 2

Network	2:
CALL	"AG_SEND"
ACT	:=M100.0
ID	:=1
LADDR	:=W#16#100
SEND	:=DB10.Contents
LEN	:=240
DONE	:=MO. 0
ERROR	:=MO. 1
STATUS	:=MW10





Figure 30



K LAD/STL/#BD - [OB1 EPICS_PLC/SIMATIC 300(1)/CPU 318-2]	
🖶 <u>File Edit Insert PLC D</u> ebug <u>V</u> iew Options <u>W</u> indow <u>H</u> elp	- 8 ×
	N?
Contents Of: 'Environment'Interface'	
Interface Name	
SIMATIC_NET_CP	
	I I
FB3 READ CP 300	
FB4 REPORT CP_300	
FB5 STATUS CP_300	
FB6 WRITE CP_300	
FB9 URCY CP300PBK OB1 · "Main Brogram Sween (Cycle)"	
FB12 BSEND CP300PBK	
FB13 BRCV CP300PBK	
FB55 IP_CONFIG CP_30	
FC1 DP_SEND CP_300 Comment:	
FC2 DP_RECV CP_300	
CALL FC 1	
FC5 AG SEND CP 300	
FC6 AG_RECV CP_300	
FC7 AG_LOCK CP_300	
Comment: Network 2	
AG RECV/CT 300 E.	
×	
I: Error λ 2: Info Λ 3: Cross-references λ 4: Address info. λ 5: Modify λ 6: Diagnost	cs <u>}</u> 7:Co
Press F1 to get Help. © offline Abs < 5.2 Nw 2 Ln 1	Insert Chg

Figure 31

The editor of OB1 is shown in Figure 31.

The "FC1", which is created to send data, is called in network 1. To receive data from IOC, we need an "AG_RECV" function to listen to the messages from Ethernet.

Again, move the mouse to network 2, and double-click the "AG_RECV" function. The function becomes automatically inserted into "OB1" and the project.



16/1 EPICS using SLS s7plc driver

K LAD/STL/PBD - [OB1 EPICS_PLC/SIMATIC 300(1)/CPU 318-2]			
🖙 File Edit Insert PLC Debug Yiew Options Window Help 🗧 🖻			
	▶?		
Contents Of: Environment'Interface'			
Interface Name			
FB6 WRITE CP_300			
BIS USEND CP300PBK	1		
FB12 BSEND CP300PBK			
FB14 GET CP300PBK			
FB15 PUT CP300PBK			
B55 IP_CONFIG CP_300			
THE FCI DE SEND CE 300			
TC2 DI_KLEV CI_000 CAND TE I			
FC4 DP_CTRL CP_300			
FC5 AG_SEND CP_300	_		
Hetwork 2: Title:			
FC8 AG INLOCK CP 30 Comment:			
EC40 FTP_CONNECT CP			
CALL "AG RECV"			
Determined and the second seco			
LADDR :=			
TC50 AG LSEND CP 300			
AG RECV/CP 300 E			
Program elem			
FC6 / AG_RECV / AG RECEIVE			
	<u>λ</u> 7: Co		
Press F1 to get Help. So offline Abs < 5.2 Nw 2 Ln 1 I	nsert Chg //		



Arein the editor cale way to fill in the required no		Table 3
rameters. Fill these rows with proper parameters, then save	Network	2:
	CALL	"AG_RECV"
Note:	ID	:=1
1) The "ID" and "LADDR" must also match the values in	LADDR	:=W#16#100
Figure 14.	RECV	:=P#DB11.DBX 0.0 BYTE 240
2) For the CP300 series, the maximum length of "AG_RECV"	NDR	:=M1.2
is also bounded by 240 bytes.	ERROR	:=M1.3
3) The received data will be written into DB11, and the	STATUS	:=MW20
length of the received message will be written to MW30.	LEN	:=MW30



(NSRRC)

SIMATIC Manager - [EPICS_]	PLC C:\Program Files\Siemens\Step7\s7proj\EPICS_PL] 🔳 🗖 🔀
🎒 <u>F</u> ile <u>E</u> dit Insert PLC <u>V</u> iew	
Dr Barkar	🚵 🗣 🚰 🛅 🔚 🏗 🔁 < No Filter > 💽 🍸
EPICS_PLC SIMATIC 300(1) CPU 318-2 Sources	System data FC5 G DB10 DB11
Blocks	1. Select all
Press F1 to get Help.	

Figure 33

Select all components in SIMATIC Manager, and press the download button to download all components into PLC. The general-purpose communication configuration is then finished.

Communication Example

- In this example, we will
- 1) confirm the connection between PLC and EPICS IOC,
- 2) put an analog value into PLC from IOC,
- 3) multiply this analog value by 2 in PLC,
- 4) read this value,
- 5) read a bit (switch) status, and
- 6) modify this bit status (turn on/off the switch).



🗱 LAD/STL/FED - [OB1 EPICS_PI	CVSIMATIC 300(1)/CPU 318-2]	
💶 File Edit Insert PLC Debug View	<u>Options Window H</u> elp	_ 8 ×
	-이빱나파노 💦	
FB blocks	⊡ - ⊕ Interface ⊕ TEMP	Contents Of: EnvironmentUnt
 ☑ SFB blocks ☑ SFC blocks ☑ Multiple instances ☑ Libraries 	Network 3: Title: Comment:	
	L DB11.DB0 L 2 *I	1 0
Function blocks of the project $\underline{\tau_{\leq}}$	T DB10.DBW NOP O	
Program elem 🛐 Call struct		
Press F1 to get Help.		🛛 offline 🛛 Abs · 🦯

Figure 34

Network 4: Title:			
Comment:			
Α	DB11.DBX	2.0	
=	0 36.0		
=	DB10.DBX	2.0	
A	DB11.DBX	2.1	
=	Q 36.1		
=	DB10.DBX	2.1	
A	DB11.DBX	2.2	
=	Q 36.2		
=	DB10.DBX	2.2	
A	DB11.DBX	2.3	
=	Q 36.3		
=	DB10.DBX	2.3	

Figure 35

- In the PLC terminal, insert a new network (Network 3) in Organization block 1 (OB1) to
- 1) load the value from DB11.DBW0,
- 2) multiply this value by 2, and
- 3) save the value to DB10. DBW0.

Insert another new network (Network 4) to read the bit status of DB11 and turn on/off the output device. The program is shown in Figure 35.

Remember to download the modified OB1 into PLC by pressing the download button. Then switch the PLC to RUN mode.



```
Table 4 shows the modified contents of data
base "example.db".
The record "s7-status" is used to verify
the PLC connection status. The record
"ail6-1" is an analog input record that
reads the value of the first word of DB10
(i.e. DB10.DBW0). The record "ao16-1" is an
analog output record that writes the value to
the first word of DB11 (i.e. DB11.DBWO). The
records "bi-x/bo-x" are the binary in-
put/output records; they read/write the
value from/to the x-th bit of the third byte
of DB10/DB11.
                 Table 4
record(bi, s7-status) {
  field(DTYP, "S7plc stat")
  field(INP, "@ Station:0")
  field(SCAN, "I/O Intr")
  field(ZNAM, "disconnected")
  field(ONAM, "connected")
  field(ZSV, "MAJOR")
  field(FLNK, "s7-status-counter")
}
record(calc, s7-status-counter) {
  field(INPA, "s7-status-counter")
  field(CALC, "A+1")
  field(FLNK, "s7-disconnect-counter")
}
record(calc, s7-disconnect-counter) {
  field(INPA, "s7-status")
  field(INPB, "s7-disconnect-counter.LA")
  field(INPC, "s7-disconnect-counter")
  field(CALC, "(A=0&&B=1)?C+1:C")
}
record(ai, ai16-1) {
  field(SCAN, " I/O Intr")
  field(DTYP, "S7plc")
  field(INP, "@ Station:0/0 T=WORD")
}
record(ao, ao16-1) {
  field(DTYP, "S7plc")
  field(OUT, "@ Station:0/0 T=WORD")
```

```
record(bi, bi-1) {
  field(SCAN, "I/O Intr")
  field(DTYP, "S7plc")
  field(INP, "@ Station:0/2 B=0 T=BYTE")
record(bi, bi-2) {
  field(SCAN, "I/O Intr")
  field(DTYP, "S7plc")
  field(INP, "@ Station:0/2 B=1 T=BYTE")
}
record(bi, bi-3) {
  field(SCAN, "I/O Intr")
  field(DTYP, "S7plc")
  field(INP, "@ Station:0/2 B=2 T=BYTE")
}
record(bi, bi-4) {
  field(SCAN, "I/O Intr")
  field(DTYP, "S7plc")
  field(INP, "@ Station:0/2 B=3 T=BYTE")
record(bo, bo-1) {
  field(DTYP, "S7plc")
  field(OUT, "@ Station:0/2 B=0 T=BYTE")
  field(PINI, "YES")
}
record(bo, bo-2) {
  field(DTYP, "S7plc")
  field(OUT, "@ Station:0/2 B=1 T=BYTE")
  field(PINI, "YES")
record(bo, bo-3) {
  field(DTYP, "S7p1c")
  field(OUT, "@ Station:0/2 B=2 T=BYTE")
  field(PINI, "YES")
record(bo, bo-4) {
  field(DTYP, "S7p1c")
  field(OUT, "@ Station:0/2 B=3 T=BYTE")
  field(PINI, "YES")
```



Table 5 shows the startup script of the IOC. Note that the <IP address> and the <port> field must exactly match the PLC connection configurations (see Figure 15). Set the input size and output size to 240 bytes, which is the maximum limit of a CP343 module.

```
Table 5
dbLoadDatabase .../../dbd/s7plcApp.dbd
s7plcApp_registerRecordDeviceDriver
var s7plcDebug 1
#s7plcConfigure name, IPaddr, port, inSize, outSize, bigEndian, recvTimeout, sendIntervall
#connects to PLC <name> on address <IPaddr> port <port>
                 : size of data bock PLC \rightarrow IOC [bytes]
#<inSize>
#<outSize>
                  : size of data bock IOC \rightarrow PLC [bytes]
#<br/>bigEndian>=1 : Motorola format data (MSB first)
#<bigEndian>=0 : Intel format data (LSB first)
#<recvTimeout> : time to wait for input before disconnecting [ms]
#<sendIntervall> : time to wait before sending new data to PLC [ms]
s7plcConfigure Station:0, 140. 110. 205. 61, 2000, 240, 240, 1, 2000, 500
dbLoadRecords example.db
iocInit
```



📤 Applications Actions 🝺 🎅 ≪ 🍣 🖉 🌍				
✓			root@localhost:/usr/lo	cal/epics/base-3.14.8.2/s7plc/src
<u>F</u> ile <u>E</u> dit <u>V</u> iew <u>T</u> ermin	nal Ta <u>b</u> s <u>H</u> elp			
s7plcMain: main thre	ead started			
s7plcMain Station:0:	: Connect to 140.	110.205.63	1:2000 on socke	et 5
s7plcEstablishConneo	ction Station:0:	fd=5, IP=3	140.110.205.61	port=2000
s7plcMain Station:0:	: starting send t	hread Stat	tion:0S	
s7plcMain Station:0:	: starting recv t	hread Stat	tion:0R	
s7plcSendThread Stat	tion:0: started			
s7plcReceiveThread S	Station:0: starte	ed		
iocInit: All initial	lization complete	•		
epics> <u>dbpr s7-stat</u> ı	15	check cor	nnection status	
ASG:	DESC:	DISA:	0	DISP: 0
DISV: 1	NAME: s7-status	RVAL:	1	SEVR: NO_ALARM
STAT: NO_ALARM	SVAL: 0	TPRO:	0	VAL: 1
epics> dbpr ai16-1	/	N		
ASG:	DESC:	DISA:	0	DISP: 0
DISV: 1	NAME: ai16-1	RVAL:	0	SEVR: NO_ALARM
STAT: NO_ALARM	SVAL: 0	TPRO:	0	VAL: 0
epics> dbpf ao16-1 2	234	Analog int	out/output	
DBR_DOUBLE:	234		pat/output	
epics> dbpr ai16-1				
ASG:	DESC:	DISA:	0	DISP: 0
DISV: 1	NAME: ai16-1	RVAL:	468	SEVR: NO_ALARM
STAT: NO_ALARM	SVAL: 0	<pre>/ TPRO:</pre>	0	VAL: 468
epics> <u>dbpr_bi-1</u>		N		
ASG:	DESC:	DISA:	0	DISP: 0
DISV: 1	NAME: bi-1	RVAL:	0	SEVR: NO_ALARM
STAT: NO_ALARM	SVAL: 0	TPRO:	0	VAL: 0
epics> dbpf bo-1 1 Pinery input (output				
DBR_STRING:				
epics> dbpr bi-1				
ASG:	DESC:	DISA:	0	DISP: 0
DISV: 1	NAME: bi-1	RVAL:	1	SEVR: NO_ALARM
STAT: NO_ALARM SVAL: 0 VAL: 1				
剩 🛛 🖾 root@localhost:/usr/local/epi 🧶 [RPM resource gcin - Mozilla 🕽				

Figure 36

Execute the start up script. The result of the execution is illustrated in Figure 36. It is easy to read/write an analog value from/to the PLC and turn on/off a switch of a device.

Enjoy EPICS!

