Some tips regarding using Profibus DP slaves connected to an S7 CPU via a CP342-5.

By Jesper M. Pedersen. May 2009.

1	Info: There will be 2 separate Proces Images (PI). One on the S7 CPU, the other on the CP342-5 Plan to use one of these 2 strategies for addressing the i/o that is connected via the CP342-5:	
	case a.	CP342-5 PI is to be addressed via an independent DB in the S7 program.
	case b.	CP342-5 PI is be addressed via the same CPU PI as the regular i/o.
2	Setup the HW Configuration of both S7 CPU and CP342-5. Attempt to assign the i/o of the CP342-5 to one contigous block of the PI. Start from byte 0. Avoid large gaps in the addressing. This because it has to transferred as one block between the S7 CPU and the CP342-5. (described in step 4.) This is also valid even if there are several DP slaves. Info: Byte addresses in the two PI's of the S7 CPU and the CP342-5 may overlap.	
3	Prepare the "target area" in the S7 CPU for the i/o addresses of the CP342-5.	
	case a.	Setup a shared DB (i.e. "DPIO") with a STRUCT ("INP") for the inputs and another STRUCT ("OUTP") for the outputs. Each STRUCT shall be formatted exactly as the real i/o in the distributed hardware. i.e.: digital inputs and outputs becomes BOOLs. Analog inputs and outputs becomes INTs. f.ex.: A digital ouput becomes "DPIO".OUTP.K023. An analog input becomes "DPIO".INP.H23B01. It is recommended that the symbols from the electrical documentation is used rather than the PLC hardware i/o addresses. This makes the program more readable.
	case b.	Reserve some unused PI area in the S7 CPU for the i/o addresses of the CP342-5. Update the Symbol list in of the S7 CPU with the i/o of the CP342-5 - but offset to match the difference between the PI of the CP342-5, and the reserved PI of the S7 CPU. i.e.: If PI bytes 0-19 are used in the S7 CPU, and PI bytes 0-9 are used by the CP342-5, then reserve bytes 20-29 for the CP342-5 i/o in the PI of the S7 CPU. Info: The size of the PI of the S7 CPU is limited depending on the exact type. If there is more DP i/o via the CP342-5 than what can fit into the PI of the S7 CPU, then it is not possible to use this method. Then use case a in stead.

4	Add calls to the blocks FC2 DP_RECV and FC1 DP_SEND in the program of the S7 CPU. Place the calls so that FC2 DP_RECV is in the beginning before other code is executed, and FC1 DP_SEND is at the end after other code is executed. Info: Notice that DP_RECV and DP_SEND always transfers a block of i/o starting from byte 0 of the PI on the CP342-5. Info: Only make ONE call of DP_RECV and DP_SEND respectively ! It is not possible to transfer fragmented addresses by multiple calls. This because the PI that is accessed in the CP342-5 always start from byte 0.	

	case a.	// sample FC2 call:
		// place BEFORE other code that uses the i/o of the CP342.5.
		call “DP_RECV”(
		
		// this is the start address of the CP342-5 module itself.
		// It is setup in the HW Config of the S7 CPU.
		// Notice: Hex format. 256 decimal becomes 100 hexadecimal.
		CPLADDR:=W#16#0100,
		
		// this symbolic assignment automatically specifies the correct number of bytes.
		RECV:= “DPIO”.INP,
		
		NDR:=M 99.1,
		ERROR:=M 99.0,
		STATUS:=MW 104,
		DPSTATUS:=MB 0);
		
		// sample FC1 call:
		// place AFTER other code that uses the i/o of the CP342.5.
		call “DP_SEND”(
		
		// this is the start address of the CP342-5 module itself.
		// It is setup in the HW Config of the S7 CPU.
		// Notice: Hex format. 256 decimal becomes 100 hexadecimal
		CPLADDR:=W#16#0100,
		
		// this symbolic assignment automatically specifies the correct number of bytes.
		SEND:=”DPIO”.OUTP,
		
		DONE:=M 99.1,
		ERROR:=M 99.0,
		STATUS:=MW 104);

	case b.	// sample FC2 call.
		// place BEFORE other code that uses the i/o of the CP342.5.
		call “DP_RECV”(
		
		// this is the start address of the CP342-5 module itself.
		// It is setup in the HW Config of the S7 CPU.
		// Notice: Hex format. 256 decimal becomes 100 hexadecimal.
		CPLADDR:=W#16#0100,
		
		// this moves input bytes 0-9 of the CP342-5 to input bytes 20-29 of the
		// S7 CPU
		RECV:= P#I20.0 BYTE 10,
		
		NDR:=M 99.1,
		ERROR:=M 99.0,
		STATUS:=MW 104,
		DPSTATUS:=MB 0);
		
		// sample FC1 call:
		// place AFTER other code that uses the i/o of the CP342.5.
		call “DP_SEND”(
		
		// this is the start address of the CP342-5 module itself.
		// It is setup in the HW Config of the S7 CPU.
		// Notice: Hex format. 256 decimal becomes 100 hexadecimal
		CPLADDR:=W#16#0100,
		
		// this moves output bytes 20-29 of the S7 CPU to output bytes 0-9 of
		// the CP342-5
		SEND:= P#Q20.0 BYTE 10
		
		DONE:=M 99.1,
		ERROR:=M 99.0,
		STATUS:=MW 104);