دورة برنامج (STEP7)

اعداد المهندس
حارث اكرم ناصر
العراق بغداد ٢٠١٠

مقدمة: ان برنامج (Step7) هو البرنامج الاكثر انتشارا في انظمة التحكم الآلي ولايكاد اي مكان يحتوي على منظومة سيطرة يخلو من انظمة (Simatic7) وبعد ان كان يستخدم في الانظمة المساعدة لمحطات التوليد كمعالجات الوقود والمراجل البخاريه اصبح ألآن يستخدم بانظمة السيطرة على المحطات الكهربائية من نوع سيمنز وهذا الذي دفعني لكتابة هذه الدورة المبتدئة فبعد ان اطلعت على اغلب ملفات سيمنز الخاصة ب (Step7) وجدتها رغم تميزها لكنها مغلقة ومشتتة بالنسبة للمبتدئين وانا احب ان اكتب الدورات للمبتدئين لانهم الاكثر احتياجا وايضا المحترفين لايحتاجون الى الدورات في اغلب الاحيان.
فأسئل ممن يستفاد من هذه الدورة الدعاء لكي اتمكن من كتابة الجزء الثاني وربما اكثر حسب الحاجة فبعد الاطلاع على بعض التطبيقات الصناعية لبرنامج (Step7) وجدت ان الدورة التي قدمتها لاتغطي ٣٠-٤٠ % من البرامج الكبيرة فانشاء الله في الجزء الثاني سنذكر بعض النقاط المهمة والمستخدمة في اكثر التطبيقات

حارث اكرم ناصر
مواليد ١٩٨١ العراق بغداد
بكلوريوس هندسة كهرباء الجامعة المستنصرية ٢٠٠٣
ماجستير الكترونيك واتصالات الجامعة المستنصرية ٢٠٠٦
اعمل حاليا في بغداد محطة كهرباء القدس قسم السيطرة الذاتية
مجال العمل: العمل على متحسسات السيطرة على المحطات من نوع (Frame9) والمنظومات المساعدة و(PLC) الخاص بها (GE Speedtronic MK5,MK6) وبرامج السيطرة على المحطات (Cimplicity HMI ,IDOS, GE toolbox)
وحديثا على برامج الانظمة المساعدة لمحطات التوليد مثل (Rslogix5000,Rslogix500,Step7)
وايضا اجيد صيانة الحاسبات المكتبية وبرامج فجول بيسك وماتلاب
قدمت بعض الدورات المبتدئة كالتالي:
١- المنهاج التدريبي الشامل في السيطرة الذاتية الجزء الاول والثاني حول (MK5) وهو جزء من سلسلة للاسف لم استطع اكمالها بسبب بعض الظروف الخاصة وقتها وموجودة على موقع الكتب العربية (www.kutub.info)
٢- دورة في شرح برنامج (Matrixvb) وهو برنامج من خلاله يمكن ادراج ٦٠٠ دالة مختلفة من الماتلاب داخل فجول بيسك ليقوم بتسهيل اكثر العمليات الحسابية في فجول بيسك والموجود على الموقع (www.4shared.com) بكتابة اسم البرنامج بنافذة البحث
٣- دورتين عن (Rslogix5000) ودورتين (Rslogix500) ودورة (Rstest lite) ضمن منتدى التحكم الآلي الرائع للاستاذ حسن الشحات
ذكرت هذه الفقرات لغرض تقديم المساعدة في هذه المجالات قربة الى الله تعالى
يمكن مراسلتي على البريد الالكتروني (harithnassir2007@yahoo.com)

تتكون هذه الدورة من عشرة ايام كالتالي:

اليوم الاول:
١ـ تنصيب البرنامج
٢ـ تفعيل البرنامج
٣ـ التعرف على المكونات المادية لل (Simatic 300)

اليوم الثاني:
١ـ نافذة (Simatic Manager)
٢ـ تعريف المكونات المادية (Hardware) داخل البرنامج
٣ـ اضافة (Rack) اخر
٤ـ تغيير عناوين الكارتات

اليوم الثالث
١ـ البرمجة مع الرموز (Symbol)
٢ـ كتابة البرنامج المنطقي داخل الوحدة التنظيمية (OB1)
٣ـ جدول تعريف المتغيرات

اليوم الرابع
١ـ العنونة
٢ـ انواع البيانات داخل (Step7)

اليوم الخامس
١ـ برمجة الدوال الفرعية
٢ـ تكوين الوحدات الوظيفية وملفات البيانات
٣ـ برمجة ملفات البيانات المشتركة

اليوم السادس
١ـ تكوين الارتباط (Online) مع (CPU)
٢ـ استخدام (Simulator)
٣ـ تحميل البرنامج المنطقي

اليوم السابع
١ـ فحص البرنامج اثناء العمل

اليوم الثامن
١ـ الايعازات المنطقية الجزءالاول

اليوم التاسع
١ـ الايعازات المنطقية الجزءالثاني

اليوم العاشر
١ـ الايعازات المنطقية الجزءالثالث

اليوم الاول:

١ـ تنصيب البرنامج

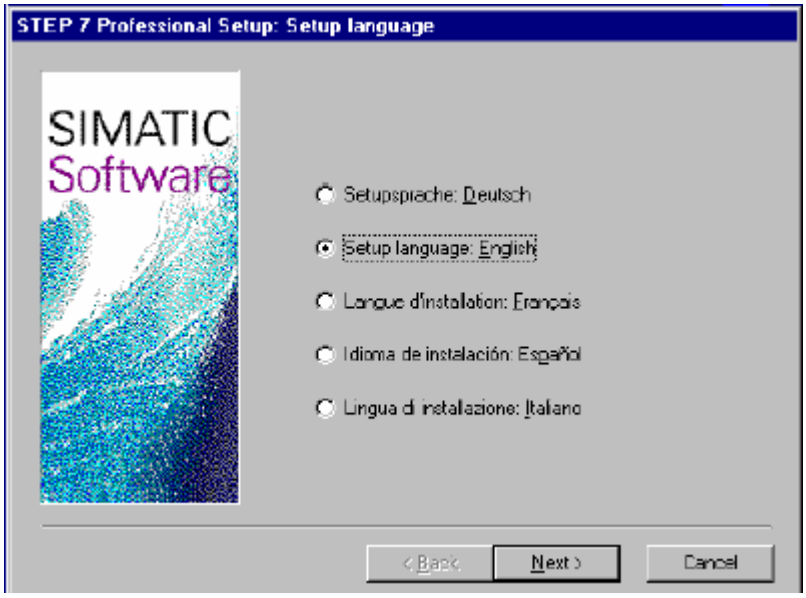
٢ـ تفعيل البرنامج

٣ـ التعرف على المكونات المادية لل (Simatic 300)

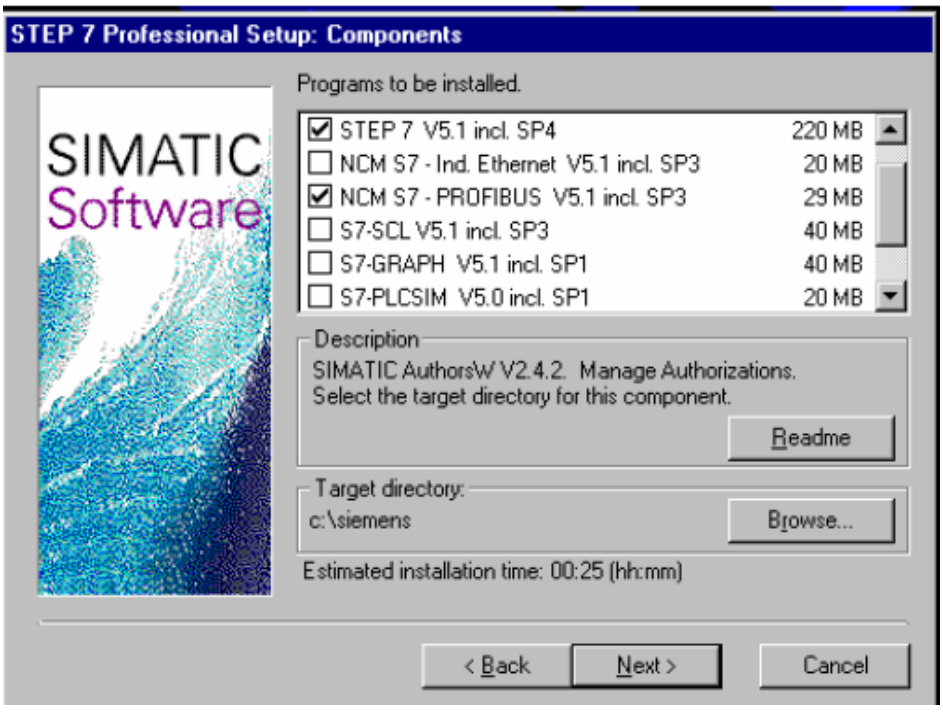
١ـ تنصيب البرنامج

خطوات تنصيب البرنامج:

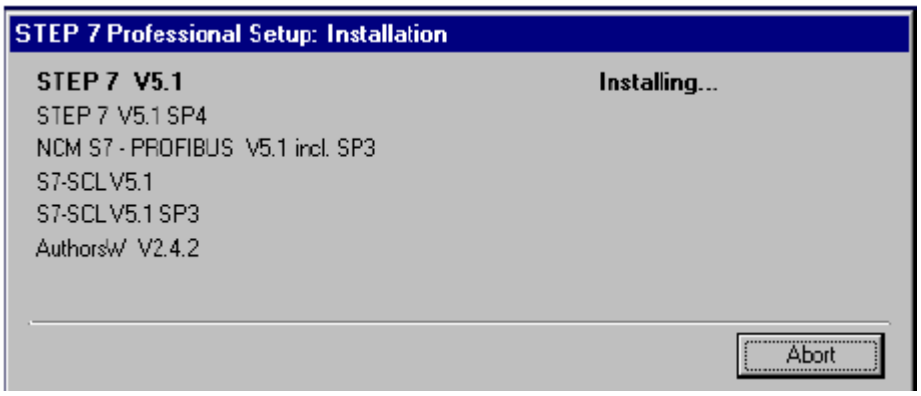
١ـ نفتح قرص (Step7) ونضغط على (setup.exe) ستظهر النافذة التالية:



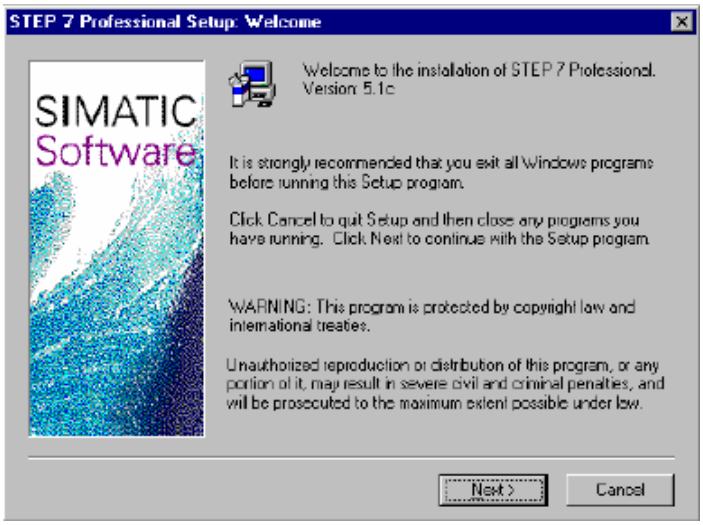
٢ـنختار اللغة ونضغط (Next)



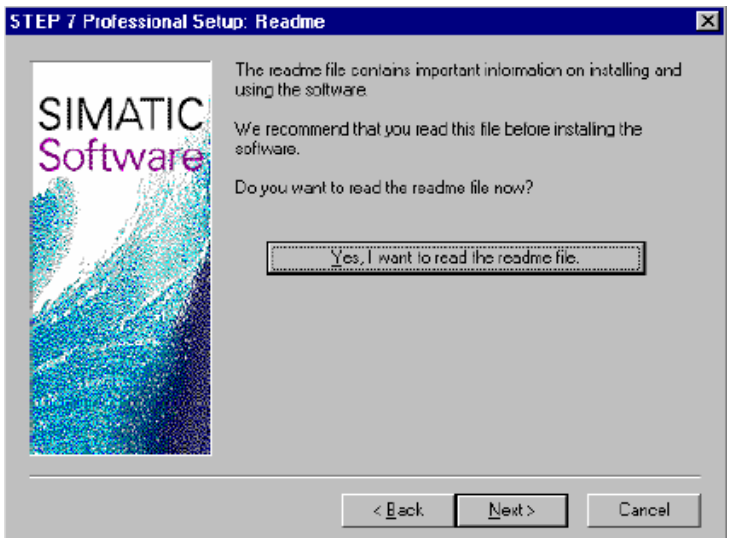
٣-نختار الاجزاء التي نحتاجها والافضل اختيار جميع الخيارات اعلاه ونضغط (Next) ستظهر النافذة التالية:



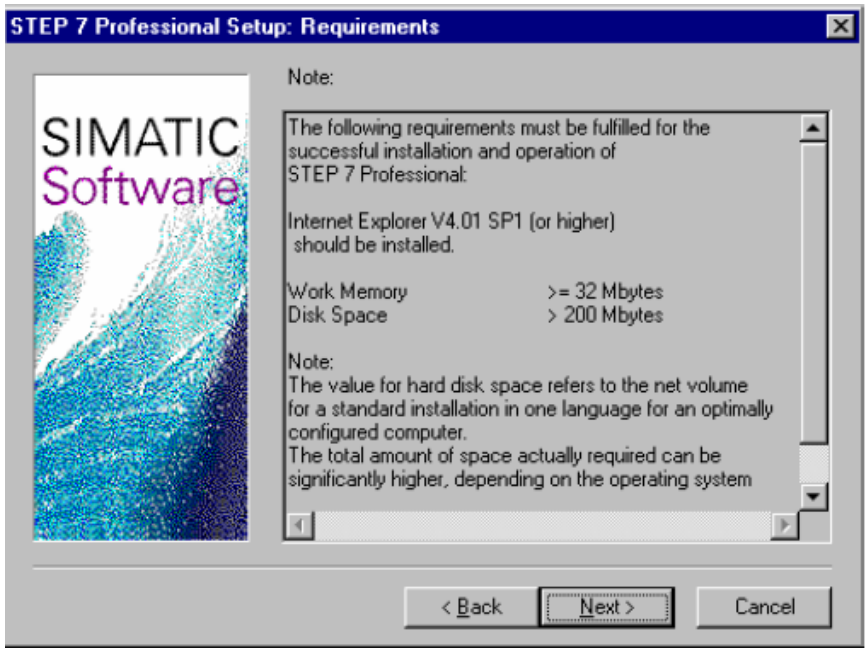
٤- ستظهر النافذة التالية نضغط على (Next)



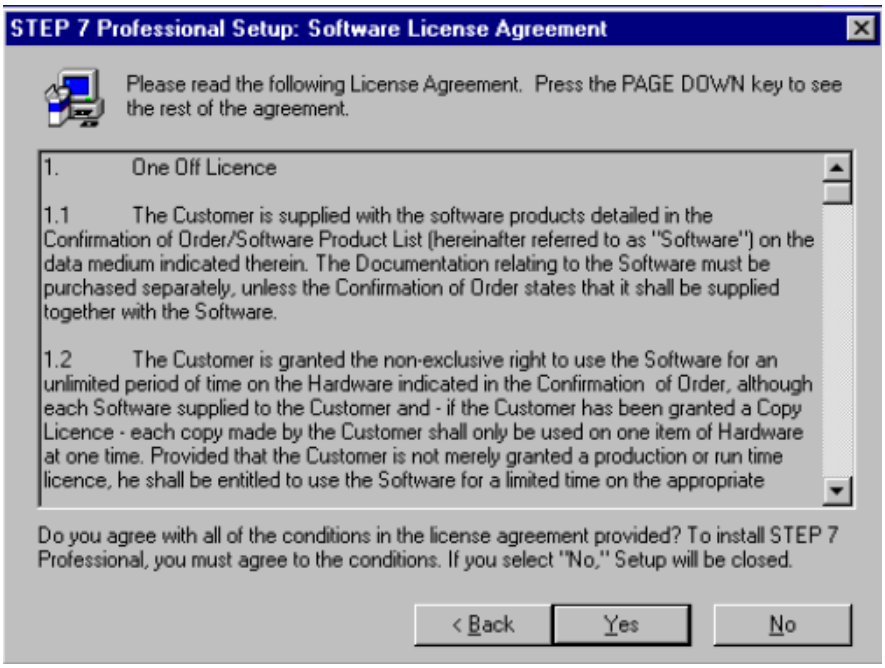
٥- نضغط على (Next)



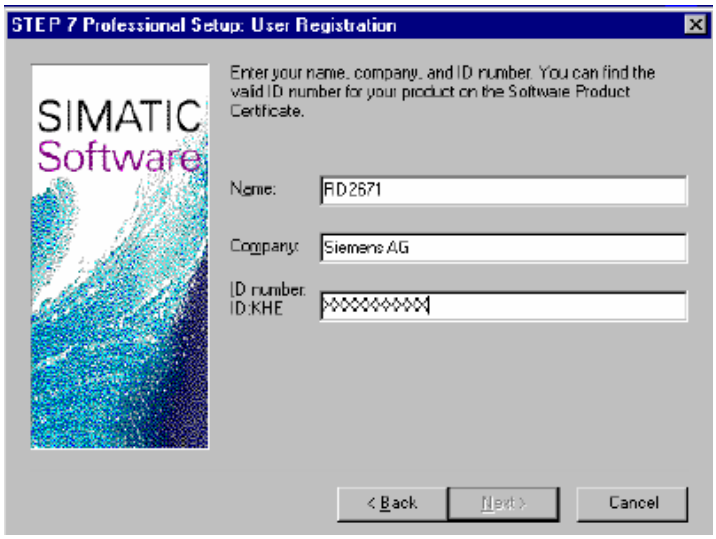
٦- نضغط على (Next)



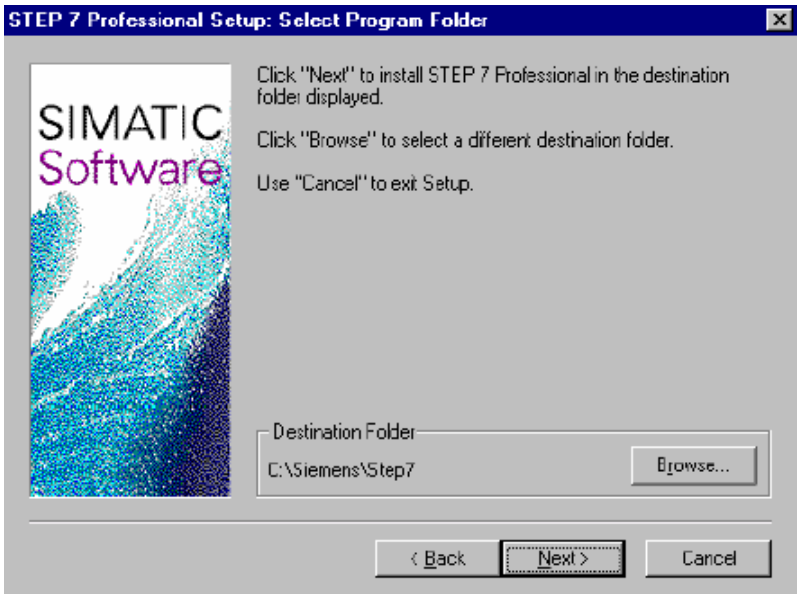
٧- نضغط على (Next)



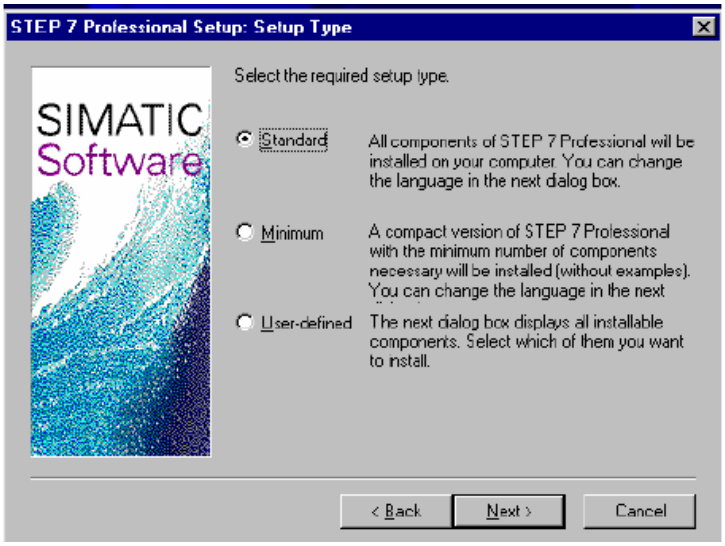
٨- نضغط على (Yes)



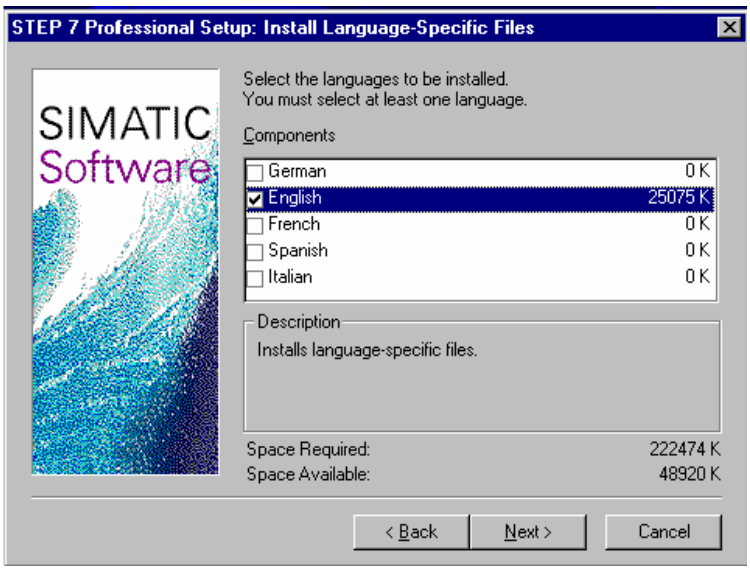
٩- نكتب (Serial) ونضغط على (Next)



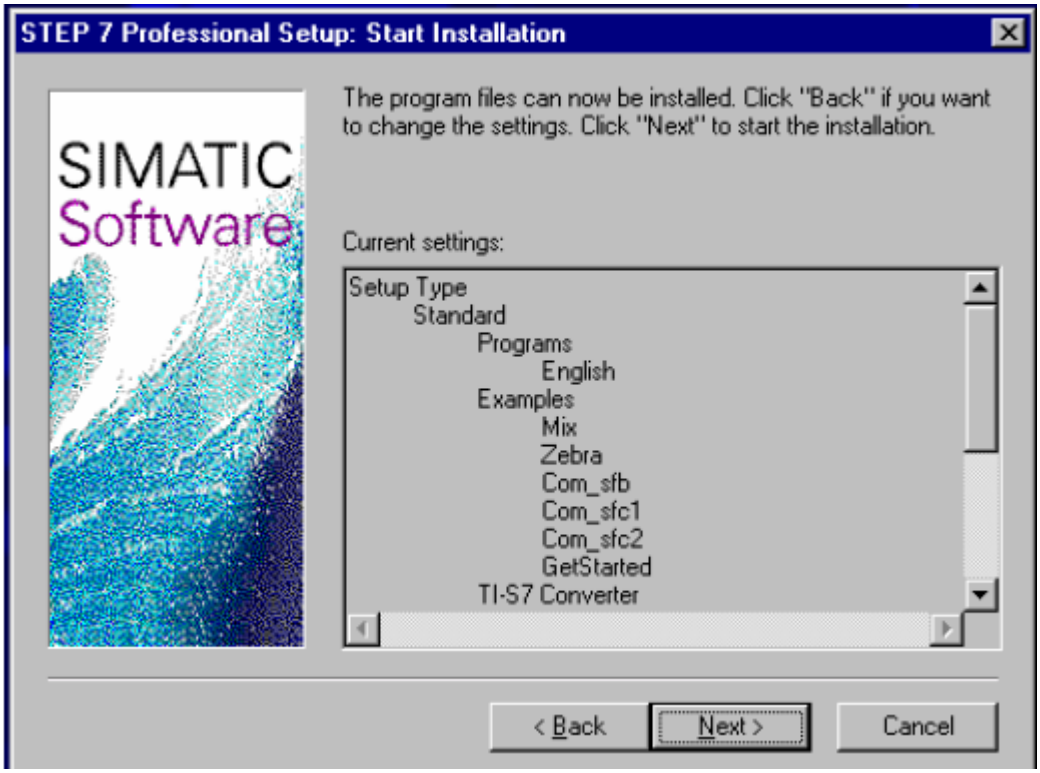
١٠-نختار مكان التنصيب ونضغط على (Next)



١١ـ نضغط على (Next)



١٢ـ نضغط على (Next)



١٣- نضغط على (Next) لبدأ التنصيب

٢-تفعيل البرنامج

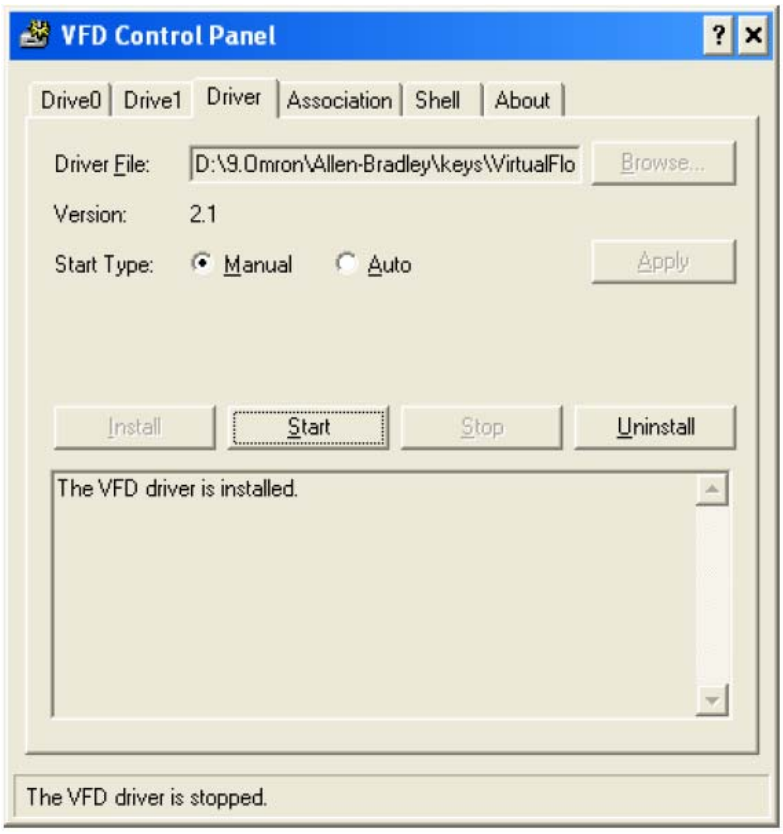
خطوات تفعيل البرنامج:

والان سنتعلم كيفية فتح البرامج بصورة مجانية

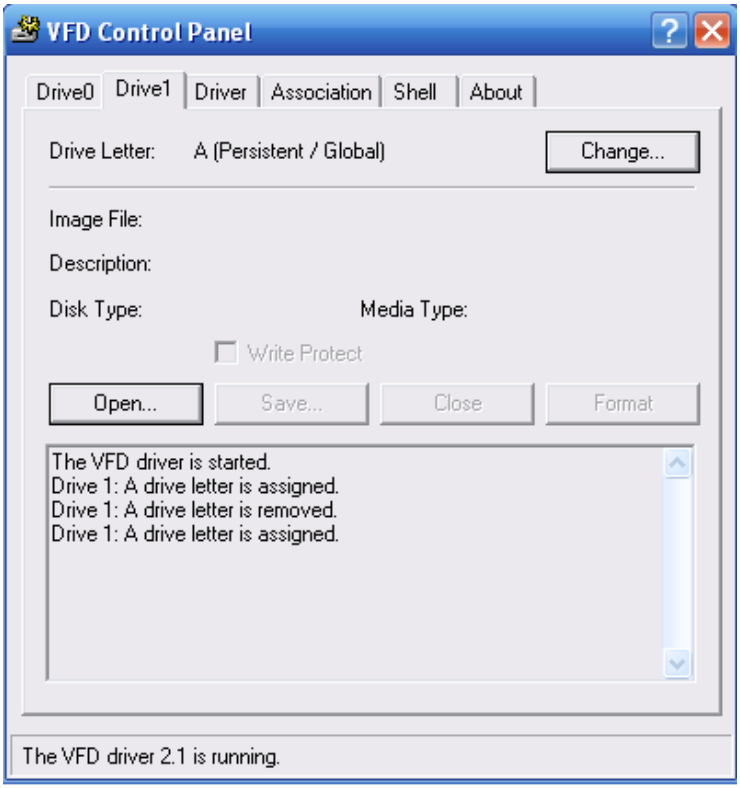
١- ندخل على موقع (www.4shared.com) ونبحث عن مجلد اسمه (Rockwell_keys_upload_by_ejbg) حيث يحتوي هذا المجلد على برنامج (virtual floppy drive)

٢- ندخل على موقع (www.4shared.com) ونبحث عن مجلد اسمه (Siemens SIMATIC KEYS AUTORIZATIONS YELLOW DISK STEP7 PCS7 WINCC WINCCFLEXIBLE DOCPRO SMARTLABEL 02-2008 updated-fixed) و بعد فتح ضغط الملفات نقوم بالبحث عن ملف اسمه (Simatic_all_key.IMA) بالنسبة للنسخة (V5.3) او (Step7 V5.4 Pro Yellow Disk.IMA) بالنسبة للنسخة (V5.4)

٣- نفتح البرنامج (vfdwin) بالنقر المزدوج عليه ستظهر النافذة كالتالي



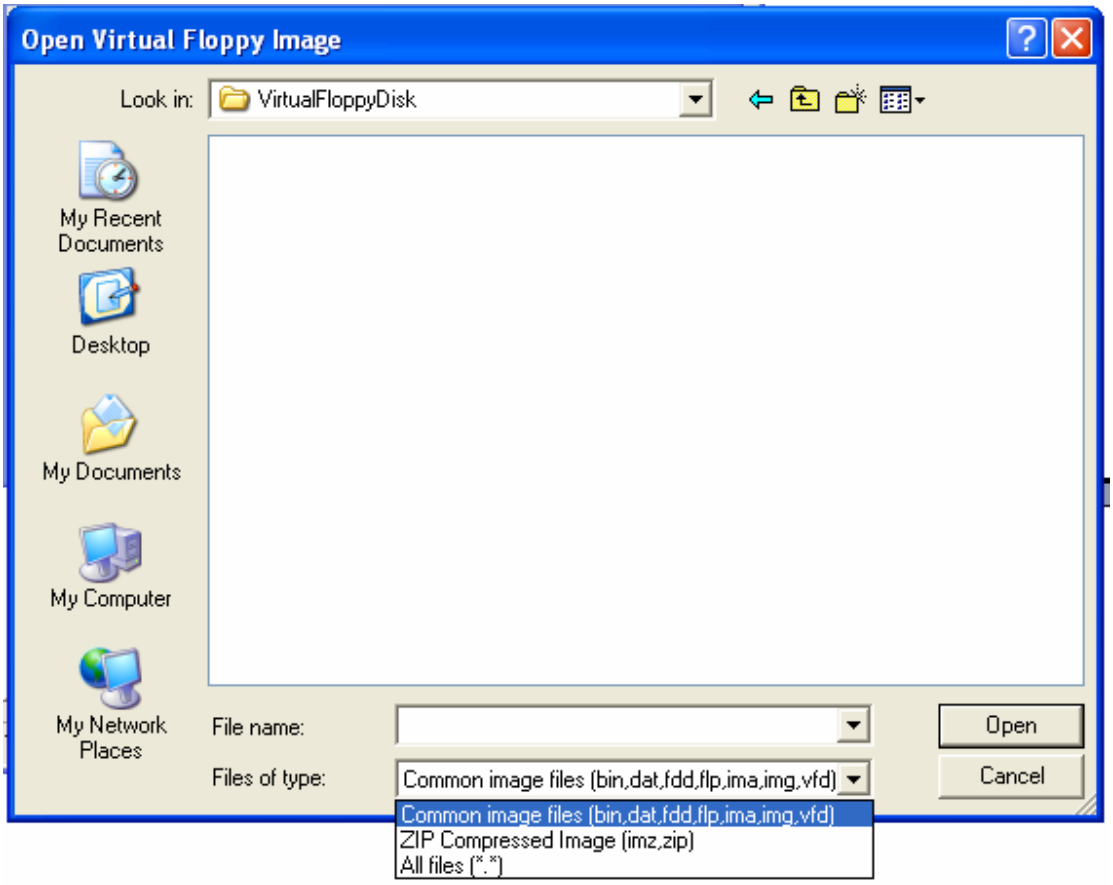
٤- نقـوم بالـضغط علـى مفتـاح (start) ثـم ننتقـل الـى الخانــة (Drive1) او (Drive0) ستظهر النافذة ادناه



٥- نضغط على مفتاح (Change) ونقوم بوضع حرف للقرص المرن الوهمي وليكن (A)

٦- نضغط على مفتاح (open) سيظهر مربع كالتالي

٧- نضغط على زر (Browse) فيظهر مربع حوار كالتالي:

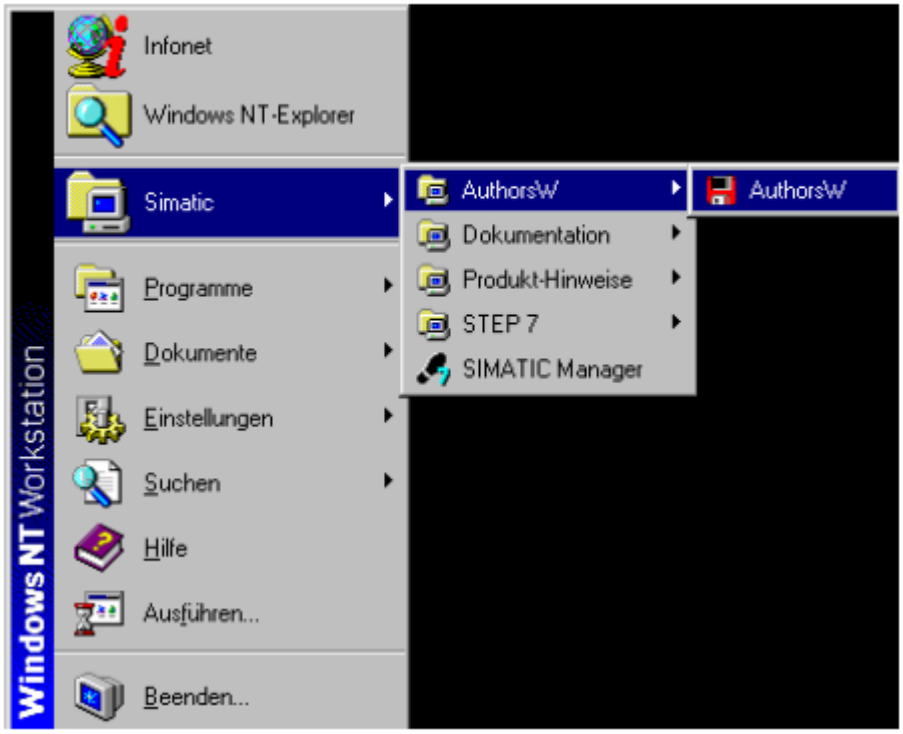


٨- نذهب الى المجلد الذي يحتوي الملف (Simatic_all_key.IMA) او (Step7 V5.4 Pro Yellow Disk.IMA) ونضغط (open)

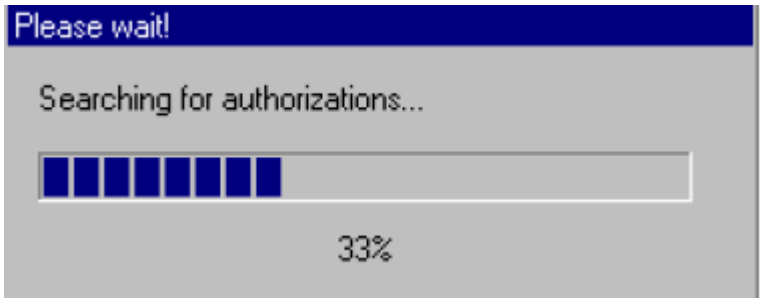
سيتكون (Floppy A) بحتوي على ملف التفعيل

٩-نفتح نافذة التفعيل الخاصة ببرنامج (Step7)

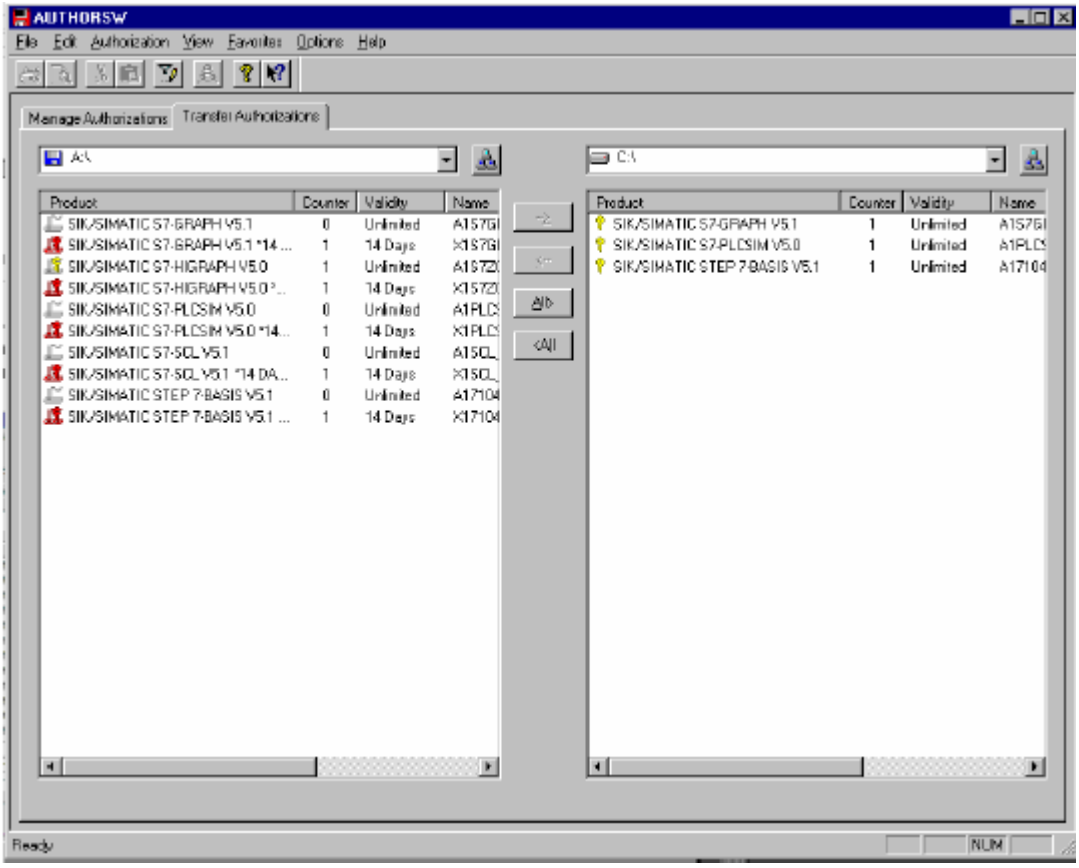
(→ START → Simatic → AuthorsW → AuthorsW).



١٠- سيتم البحث عن ال (Key)



ستظهر النافذة التالية نحدد مكان (Key) ومكان اخر لنقله الى مجلد اخر عن طريق (<-)



انتهت عملية التفعيل

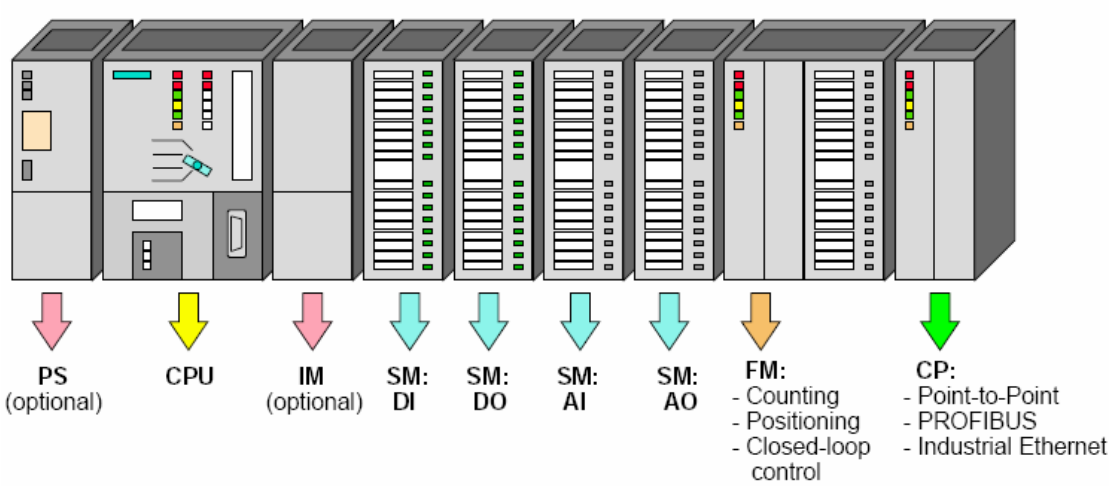
ملاحظة: يجب تفعيل البرامج التالية:
١-(Simatic Step7 5.x)
٢-(S7plcsim)

٣-التعرف على المكونات المادية لل (Simatic 300)

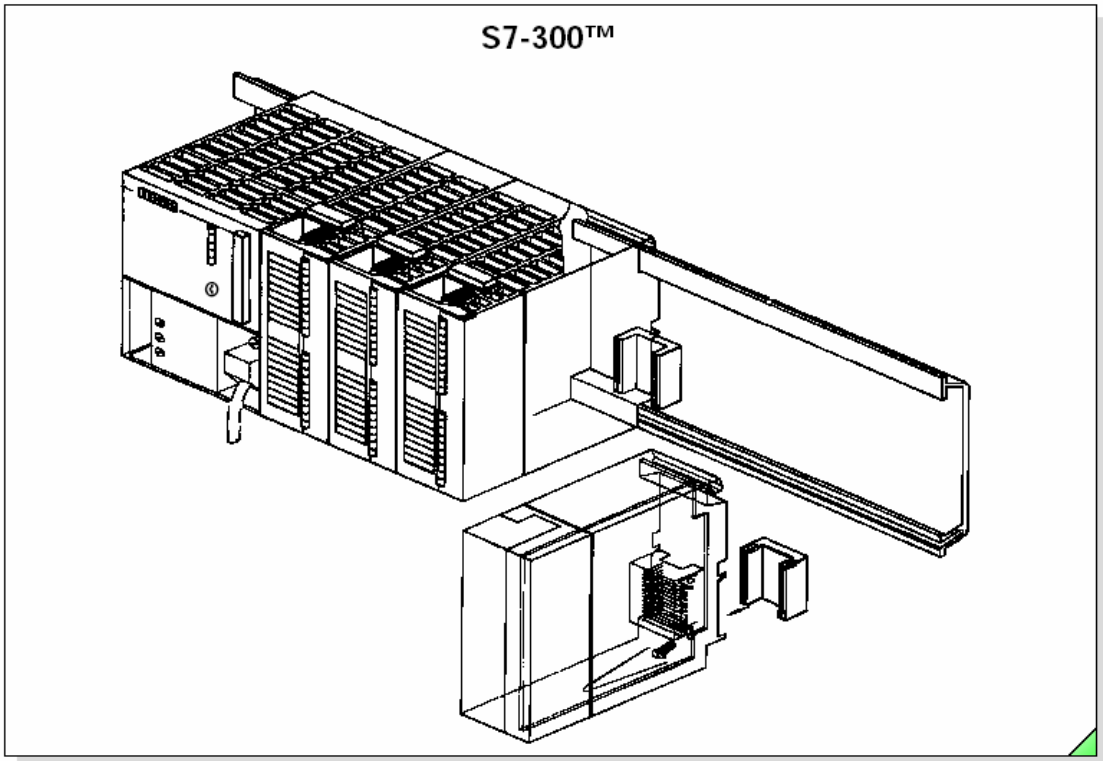
المكونات المادية لل (Simatic 300) حسب الجدول التالي:

Table 1-1. S7-300 and S7-400 Basic Components.

Component	Abbrev.	Brief Description
Racks	-	Mounting base for installing various user-selected modules.
Power Supply	PS	Supplies internal operating voltages to racks and modules.
Central Processor	CPU	Stores and processes the user control program and data.
Signal Modules	SM	Digital/Analog I/O interfaces to field sensors and actuators.
Function Modules	FM	Intelligent modules that execute control tasks independent of the CPU (e.g., PID, stepper-positioning, servo-positioning).
Communication Processors	CP	Used to establish networking among S7 PLCs and other stations or point-to-point serial links to other devices.
Interface Modules	IM	Used to make various local and remote interconnections between the S7-300 and S7-400 central and expansion racks.
Programming Device	PG/PC	PG is a PC pre-configured with facilities for developing and managing STEP 7/STEP 5 programs; PC is a user-configured PC programming system.
Multi-Point Interface	MPI	Low-performance network and multi-point programming interface to components with MPI port [e.g., CPUs, CPs, FMs, operator panels (OPs)].
Distributed I/O	DP	I/O subsystems (or DP-slaves), connected to a DP-master according to Profibus DP standard EN 50170 Volume 2.



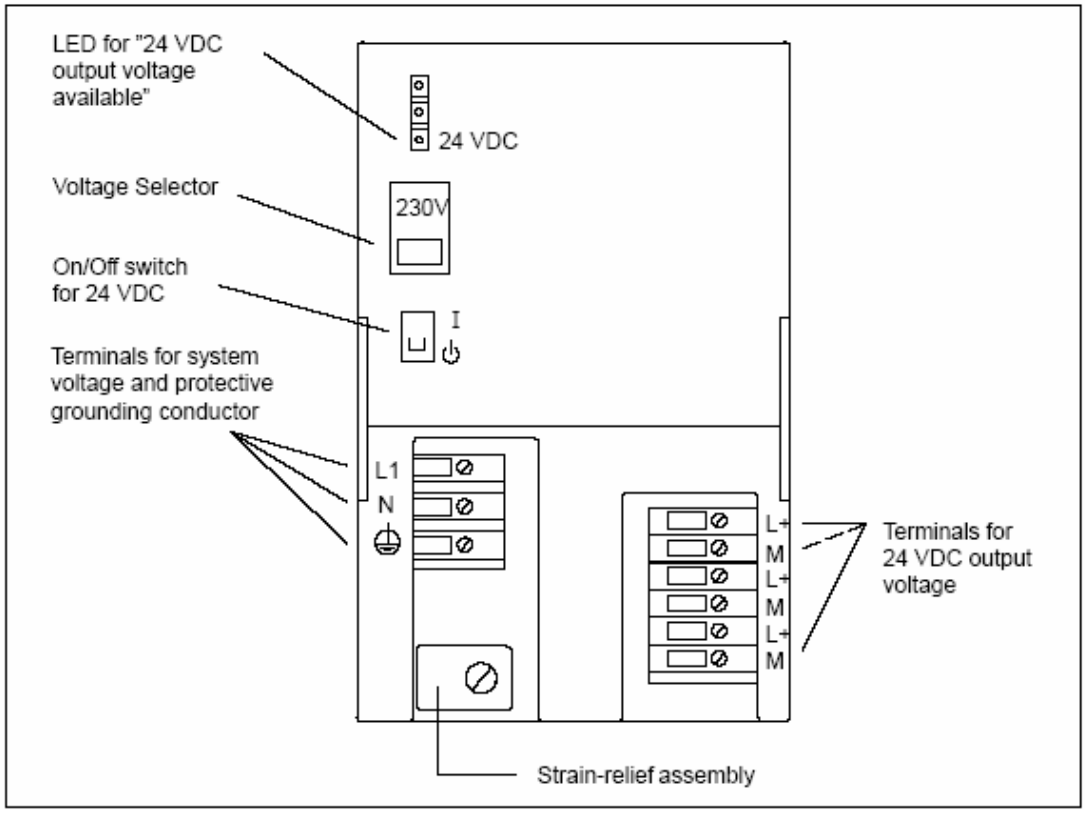
١- (Racks):يكــون ال (Rack) الخــاص ب (Simatic300) بــشكل الــسكة (Rail) ويكون باطوال مختلفة . وقبل وضع الكارت بداخله يجب تثبيت حلقة بشكل حرف (U) داخل ال (Rack) ليتم تثبيت الكارت بها.



٢-مجهز القدرة (PS): يجهز الفولتية المطلوبة لتشغيل الكارتات داخل ال (Rack) ويحمل المواصفات التالية :

Module	Family	Input	Outputs
PS 307	S7-300	120/230 VAC	24 VDC/2 A
PS 307	S7-300	120/230 VAC	24 VDC/5 A
PS 307	S7-300	120/230 VAC	24 VDC/10 A

Wiring schematic of the PS 307; 5 A



٣-(CPU):يقوم بخزن وتنفيذ البرنامج المنطقي ويحمل المواصفات التالية:

Short Name	Brief Description
CPU 31x-x	S7-300 Standard CPUs
CPU 31x-x IFM	CPUs with integrated I/O Functions (e.g., digital/analog, HS counters)
CPU 31x-x DP	S7-300; CPUs with integrated Profibus DP (DP master/DP slave port)
CPU 31xC	S7-300; Compact CPUs with integrated I/O Functions
CPU 31xF	S7-300; Fault Tolerant CPUs with integrated I/O Functions

يتكون (CPU) الخاص ب (Simatic300) على الاجزاء التالية:

Mode Selector	MRES	=	Memory reset function (Module Reset)
	STOP	=	Stop mode, the program is not executed.
	RUN	=	Program execution, read-only access possible from PG.
	RUN-P	=	Program execution, read/write access possible from PG.
Status Indicators (LEDs)	SF	=	Group error; internal CPU fault or fault in module with diagnostics capability.
	BATF	=	Battery fault; battery empty or non-existent.
	DC5V	=	Internal 5 VDC voltage indicator.
	FRCE	=	FORCE; indicates that at least one input or output is forced.
	RUN	=	Flashes when the CPU is starting up, then a steady light in Run mode.
	STOP	=	Shows a steady light in Stop mode. Flashes slowly for a memory reset request, Flashes quickly when a memory reset is being carried out, Flashes slowly when a memory reset is necessary because a memory card has been inserted.
Memory Card	A slot is provided for a memory card. The memory card saves the program contents in the event of a power outage without the need for a battery.		
Battery Compartment	There is a receptacle for a lithium battery under the cover. The battery provides backup power to save the contents of the RAM in the event of a power outage.		
MPI Connection	Connection for a programming device or other device with an MPI interface.		
DP Interface	Interface for direct connection of distributed I/Os to the CPU.		

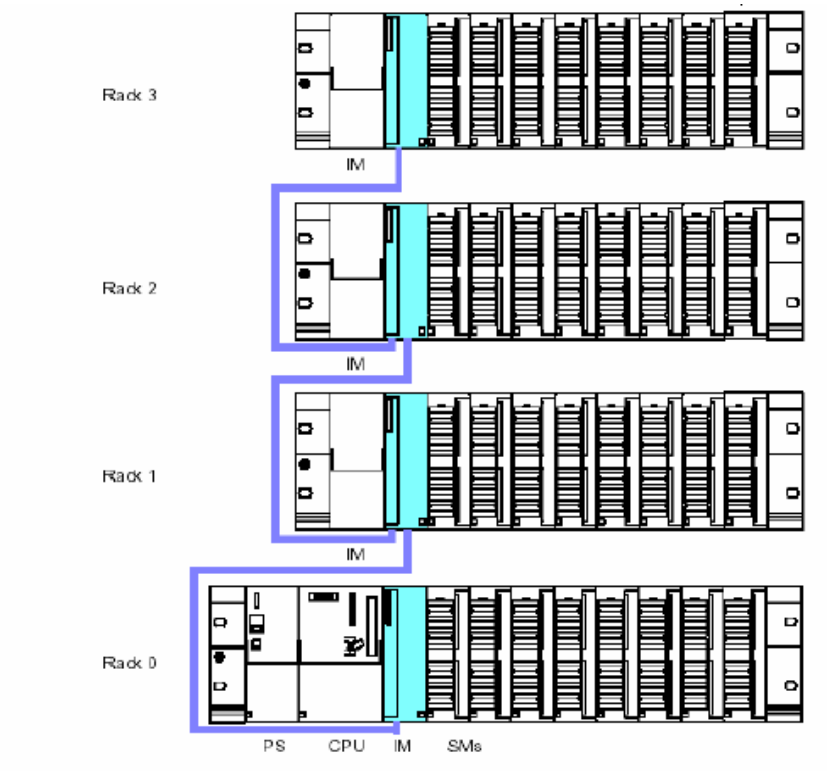
٤- كارتات المداخل والمخارج (SM):وهي وسائل الربط بين المتحسسات الخارجية وال (CPU) وبالنسبة لل (Simatic300) يحوي الانواع التالية:
أ- كارتات المداخل الرقمية (DI-300)
ب- كارتات المخارج الرقمية (DO-300)
ج-كارتات المداخل التماثلية (AI-300)
د- كارتات المخارج التماثلية (AO-300)
ه-كارتات مداخل ومخارج رقمية مشتركة (DI/DO-300)
ي- كارتات مداخل ومخارج تماثلية مشتركة (AI/AO-300)

٥- كارتات الوظائف الخاصة (FM): تستخدم في التطبيقات المعقدة مثل ال (SERVO) و (High Speed Counter) وتحتوي الكارتات بداخلها على (PID Controller) وتأخذ الرمز (FM-300)

٦-كارتات الاتصال (CP): تستخدم هذه الكارتات للاتصال بين (Simatic300) و (PLC) آخر او الاتصال عن طريق (Serial) مع الطابعة وغيرها

٧-كارتات الارتباط (IM): تستخدم للربط بين (Rack) وآخر وتحمل المواصفات التالية:

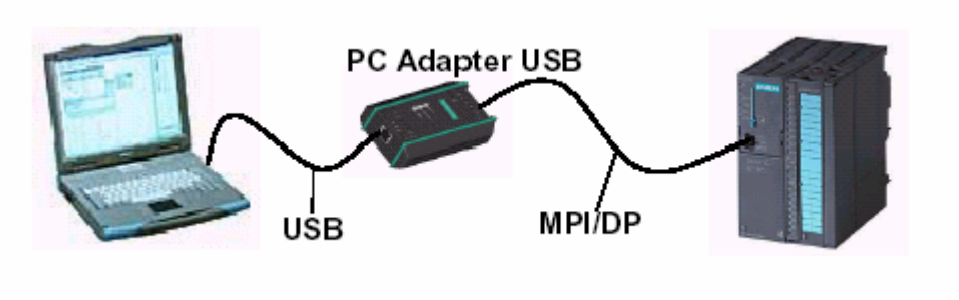
Interface Module Pair		Brief Description	Distance
CPU IM	Exp. Rack IM		
IM 365	IM 365	S7-300 Local Expansion (1-Tier Max. Expansion)	1 m
IM 360	IM 361	S7-300 Local Expansion (3-Tier Max. Expansion)	10 m



٨-(PG/PC):تمثل اعدادات الاتصال بين الحاسبة وال (PLC) فمن جهة (PLC) يحتوي ال (CPU) على منافذ (MPI) و(DP) ومن جهة الحاسبة اما نستخدم حاسبات خاصة مثل (Field PG) او (Power PG) او حاسبة اعتيادية تحتوي على كارتات (MPI) بعدة اشكال:

Power PG

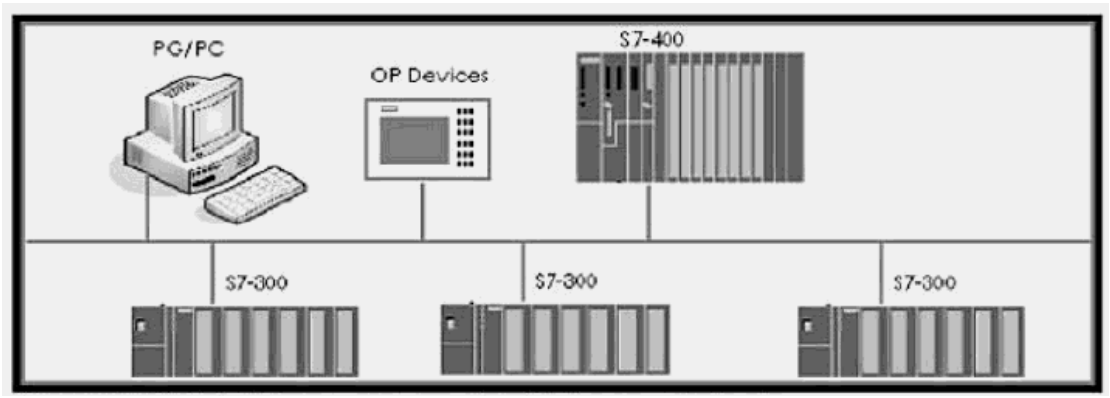
Field PG



ISA/PCI MPI Card

MPI Port

٩-منفذ (MPI): يستخدم للربط بين الحاسبة و (CPU) لاغراض تحميل البرنامج وايضا للاتصال مع شاشات سيمنز المبرمجة (OPx) او مع كارتات (CP) او (FM)



١٠-منفذ (DP): وهو يشبه شكل المنفذ (MPI) الا انه ابطأ بكثير ويستخدم لربط (CPU) مع (CPU) آخر لتكوين الاتصال من نوع (Master/Slave) عن طريق (Profibus) والشكل ادناه يوضح (CPU315) يحوي منفذين من جهة اليمين (DP) ومن جهة اليسار (MPI) ويكتب اسم المنفذ فوق المنفذ

اليوم الثاني:

١ـ نافذة (Simatic Manager)

٢ـ تعريف المكونات المادية (Hardware) داخل البرنامج

٣ـ اضافة (Rack) اخر

٤ـ تغيير عناوين الكارتات

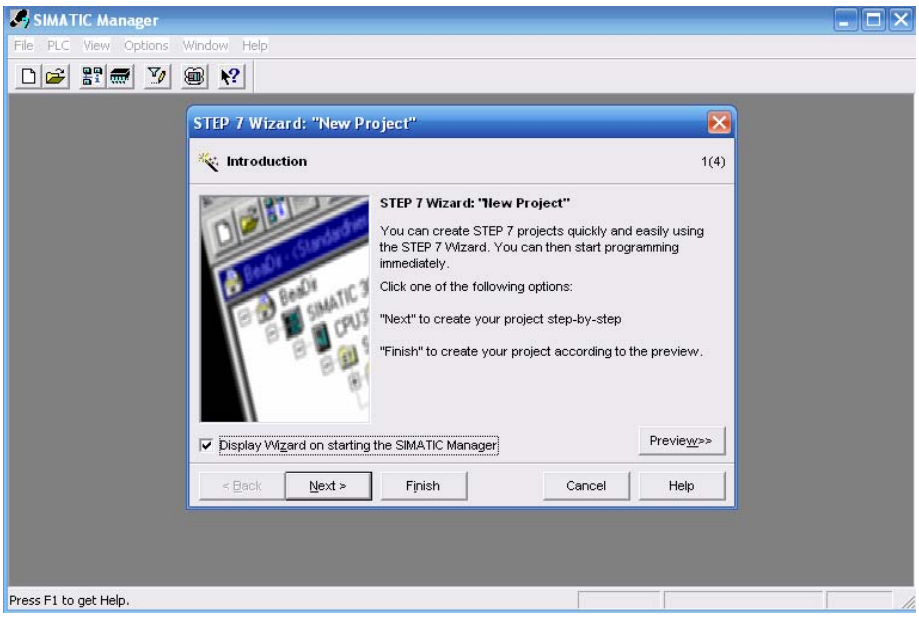
١ـ نافذة (Simatic Manager): عبارة عن طريقتين:

الطريقة الاولى: (Use The Project Wizard)

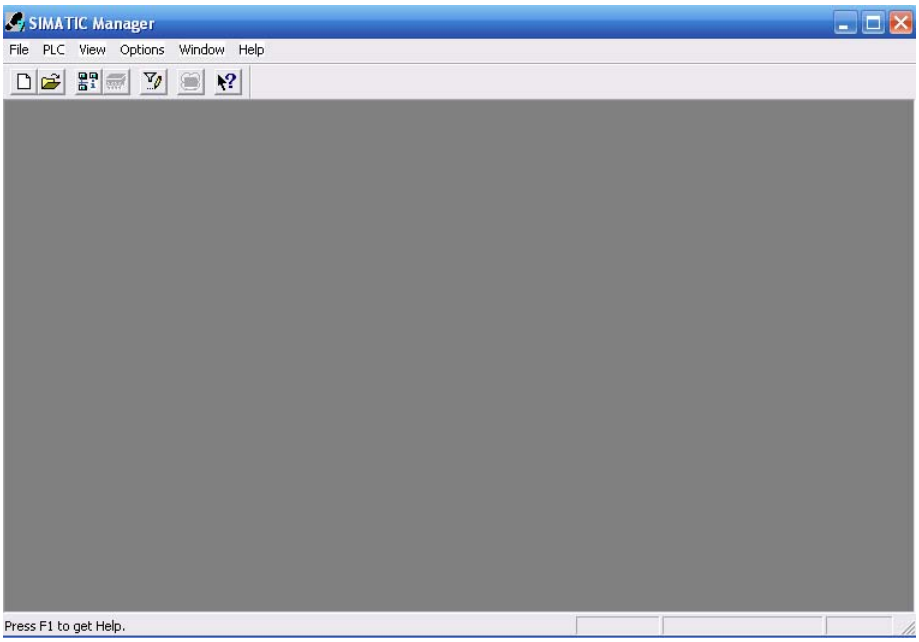
وهذه الطريقة يمكن من خلالها تكوين ملفات المشروع بصورة مباشرة

عند فتح البرنامج من المسار التالي :

ستظهر النافذة التالية:

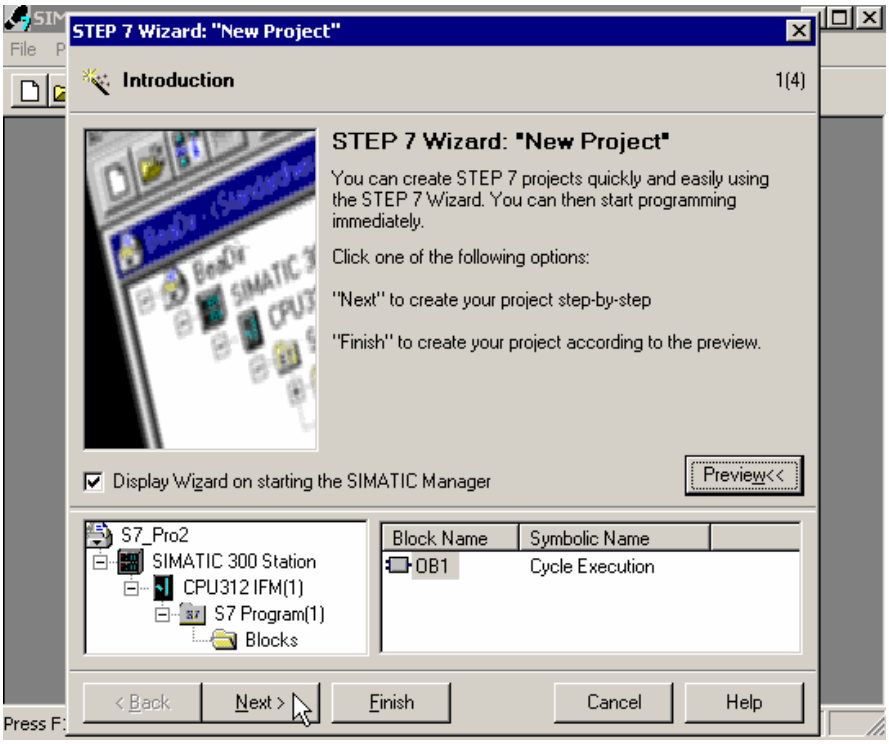


نضغط على (Cancel) سيكون شكل النافذة كالتالي:

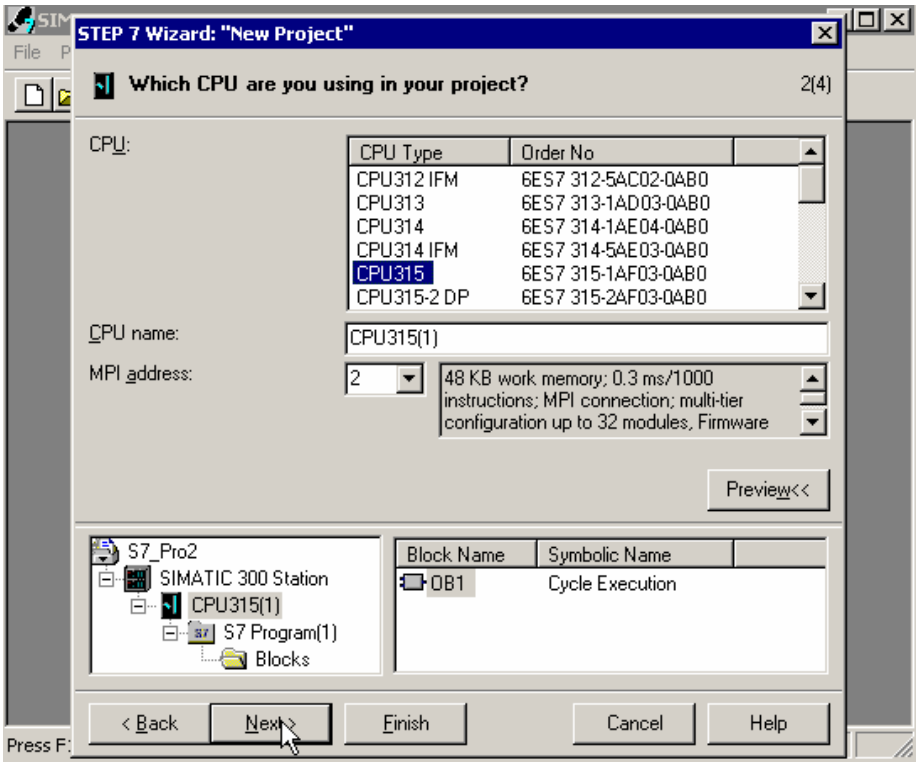


من قائمة (File) نختار (New Project Wizard)
ستظهر النافذة التالية:

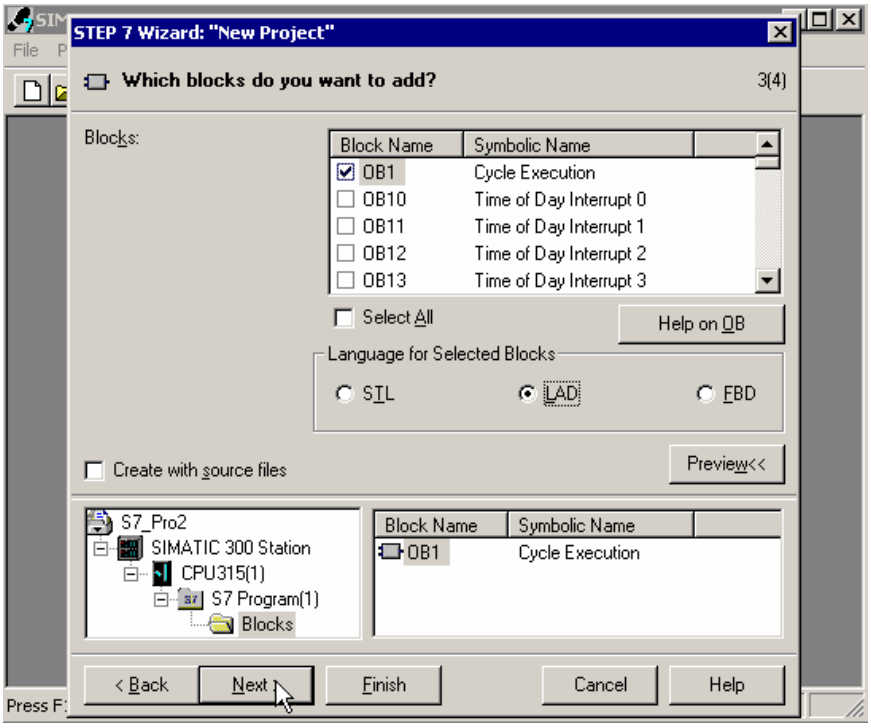
نضغط على (Preview) ستظهر النافذة التالية:



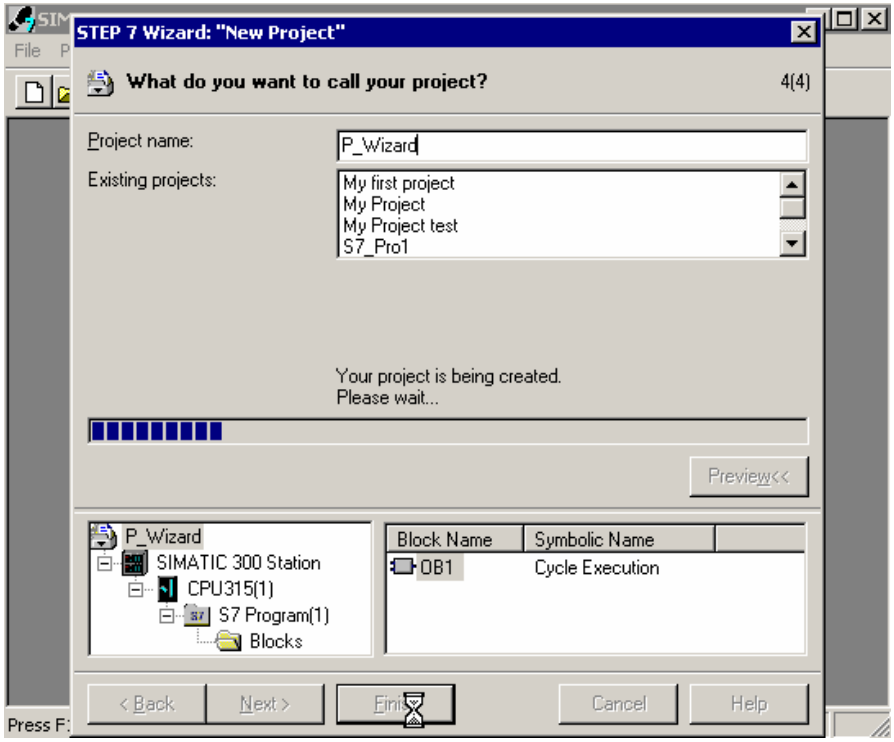
نضغط على (Next)



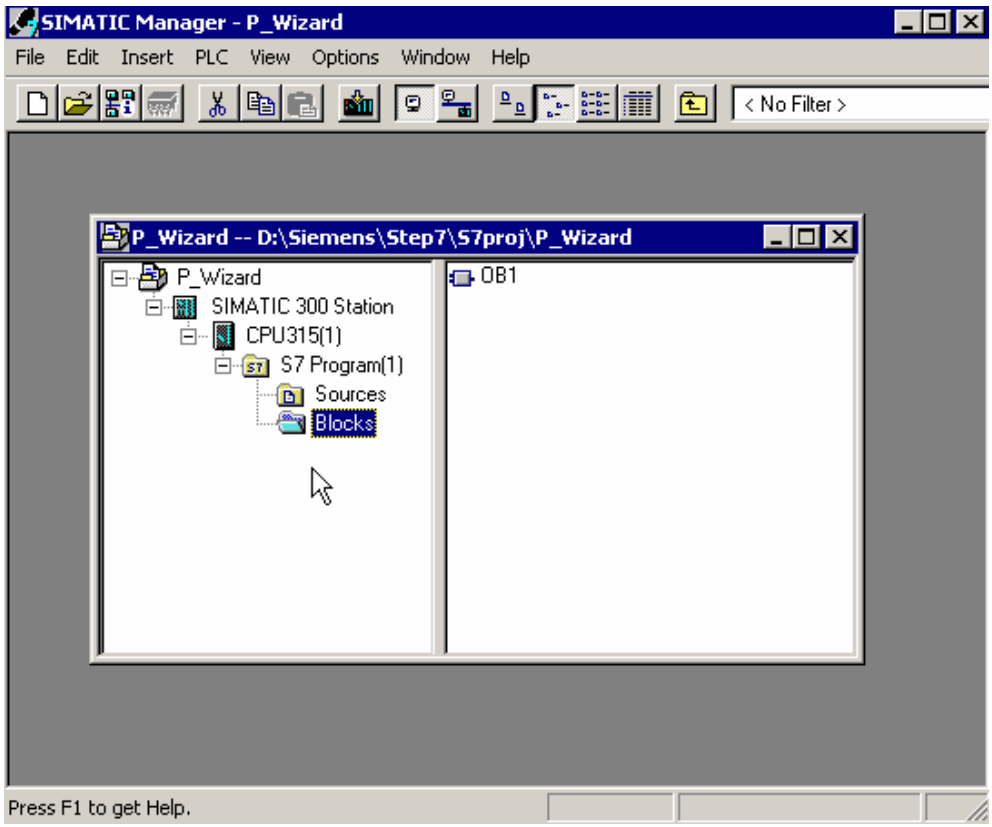
نختار نوع ال (CPU) ونضغط على (Next) ستظهر النافذة التالية:



نختار الوحدة التنظيمية (OB) ونوع لغة البرمجة وليكن (LAD) ثم نضغط (Next) ستظهر النافذة التالية:



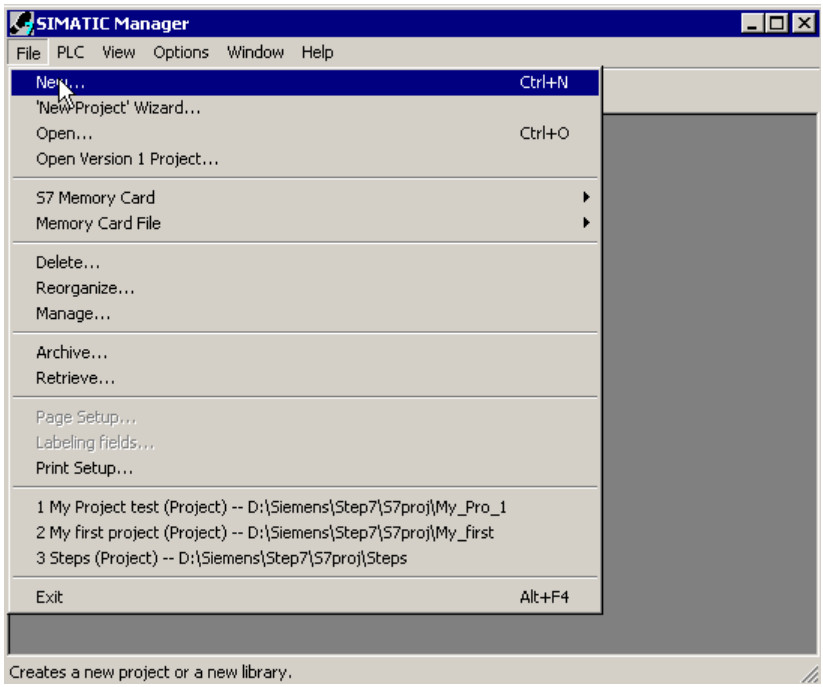
نقوم بكتابة اسم المشروع في حقل (Project name) ونضغط على (Finish)
ليتم تكوين المشروع وتظهر النافذة التالية:



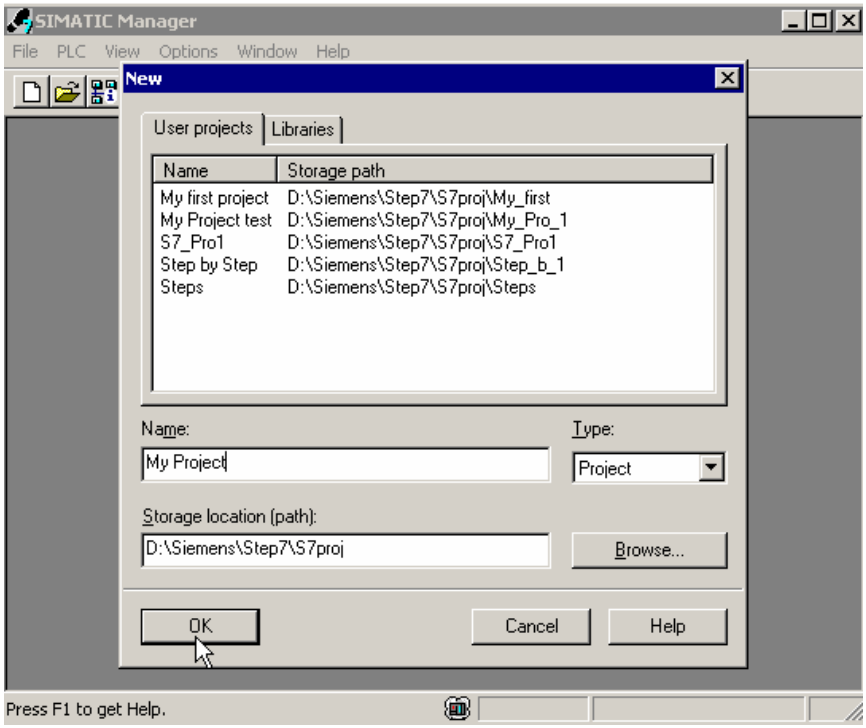
وهذه النافذة تبين جميع اجزاء المشروع

الطريقة الثانية:

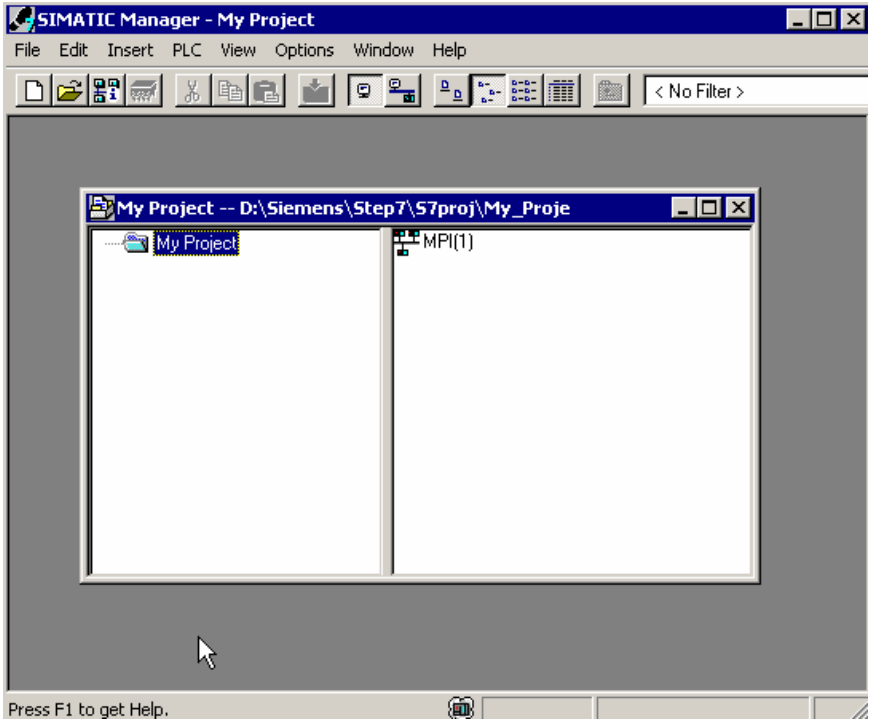
من قائمة (File) نختار (New):



ستظهر النافذة التالية نقوم بكتابة اسم المشروع ونضغط على (Ok)



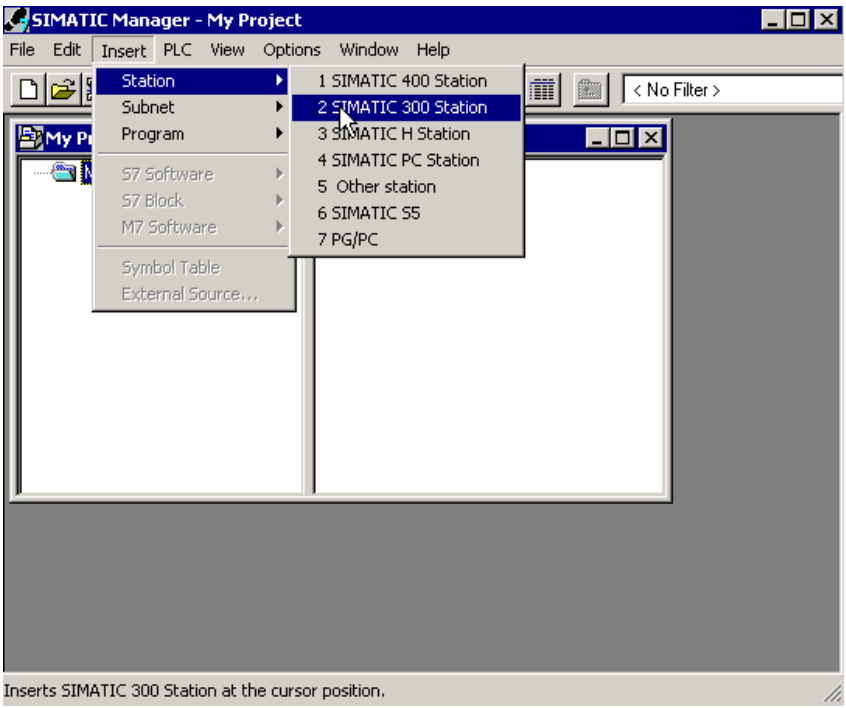
ستظهر النافذة التالية:

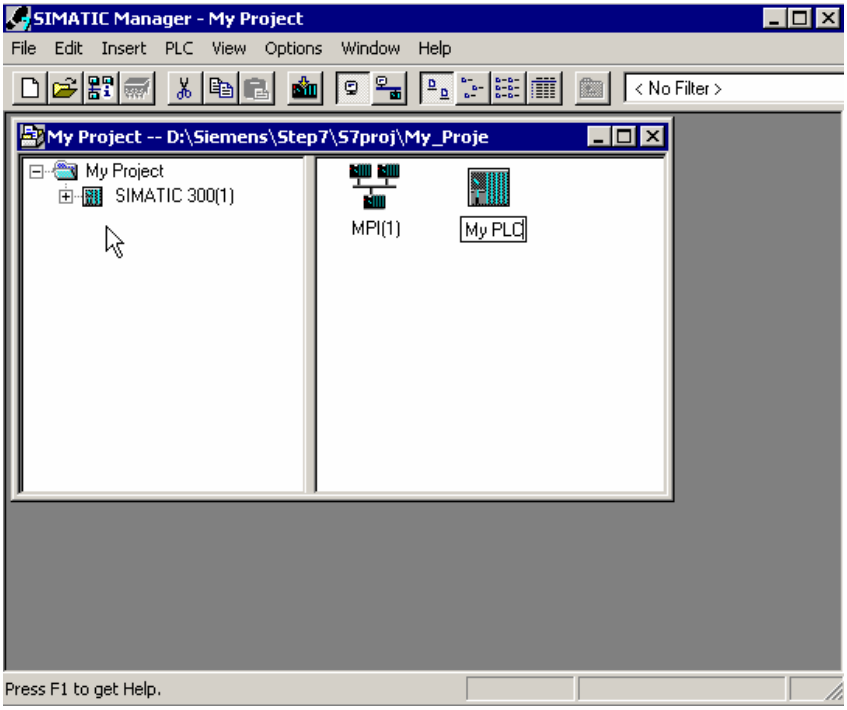


وبعدها نقوم بتعريف المكونات المادية (Hardware) بصورة تدريجية

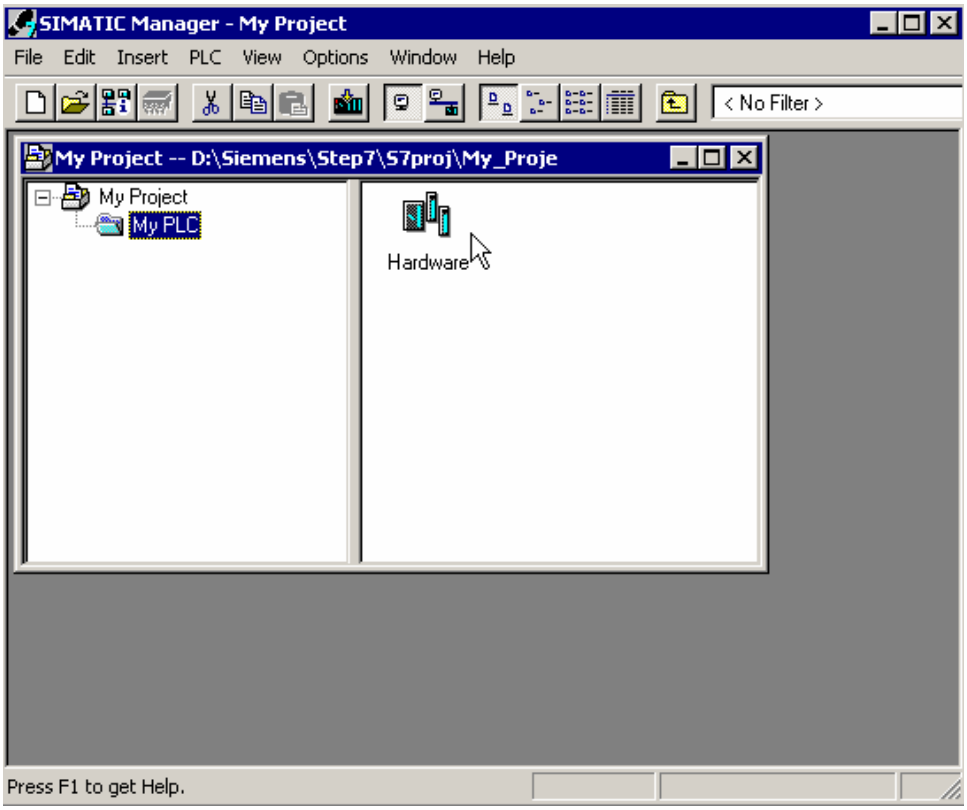
٢ـتعريف المكونات المادية (Hardware) داخل البرنامج

لتعريف ال (Hardware) نقوم باضافة (Station) من قائمة (Insert) نختار (Station) ثم (SIMATIC 300 Station)

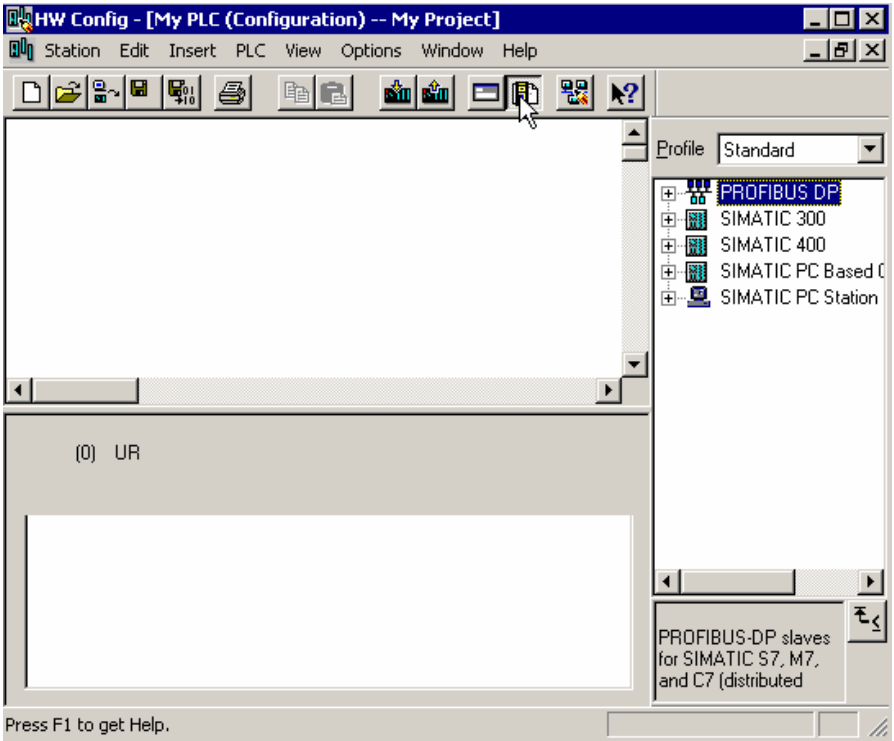




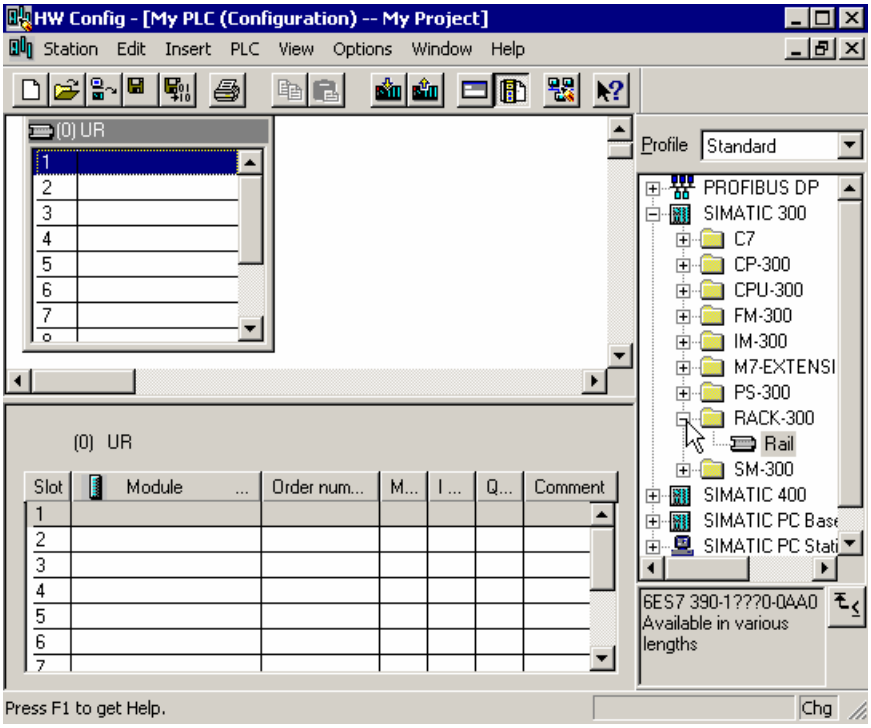
نقوم بوضع اسم لل (Station) وليكن (My PLC) ثم نضغط (enter) ستظهر النافذة التالية:



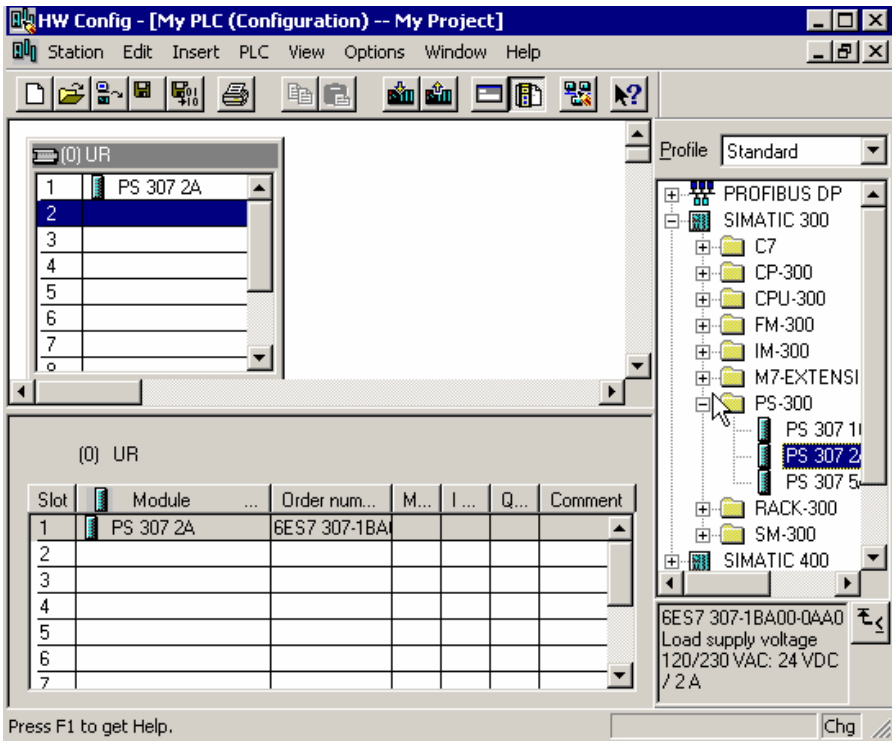
عن اختيار (My PLC Station) تظهر ايقونة (Hardware) داخل النافذة على اليمين نقوم بالنقر المزدوج على ايقونة (Hardware) تظهر النافذة التالية نضغط على ايقونة (Catalog) او (CTRL+k) لعرض اسماء الكارتات:



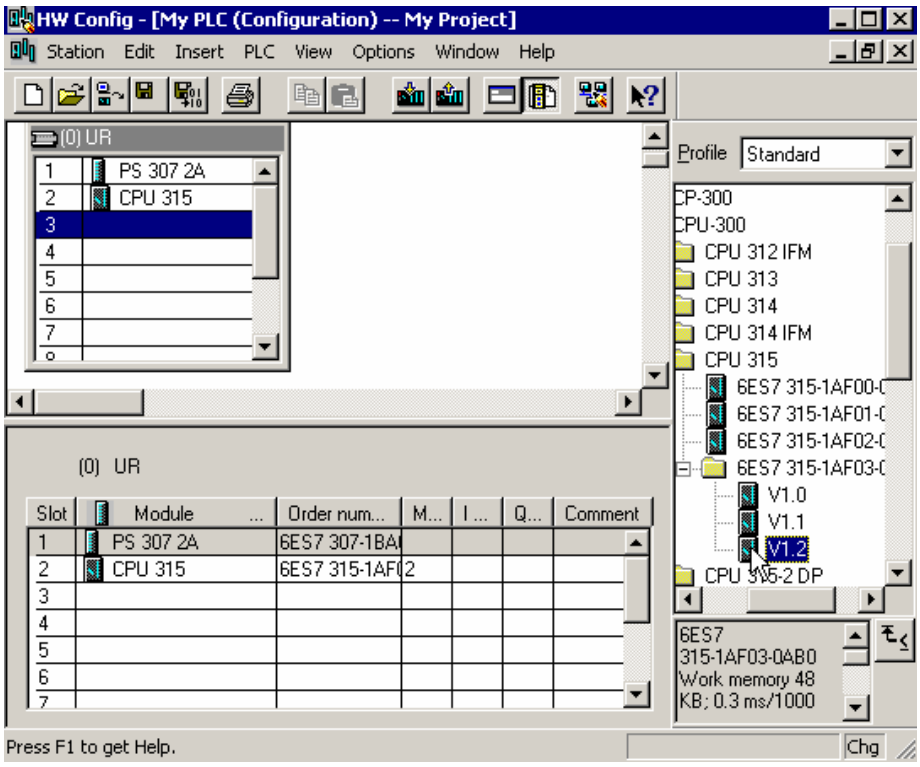
نختار من نافذة (Catalog) على اليمين التفرع (SIMATIC 300) ثم (RACK-300) ثمم ننقر نقرا مزدوجاً على (Rail) سيكون شكل النافذة كالتالي:



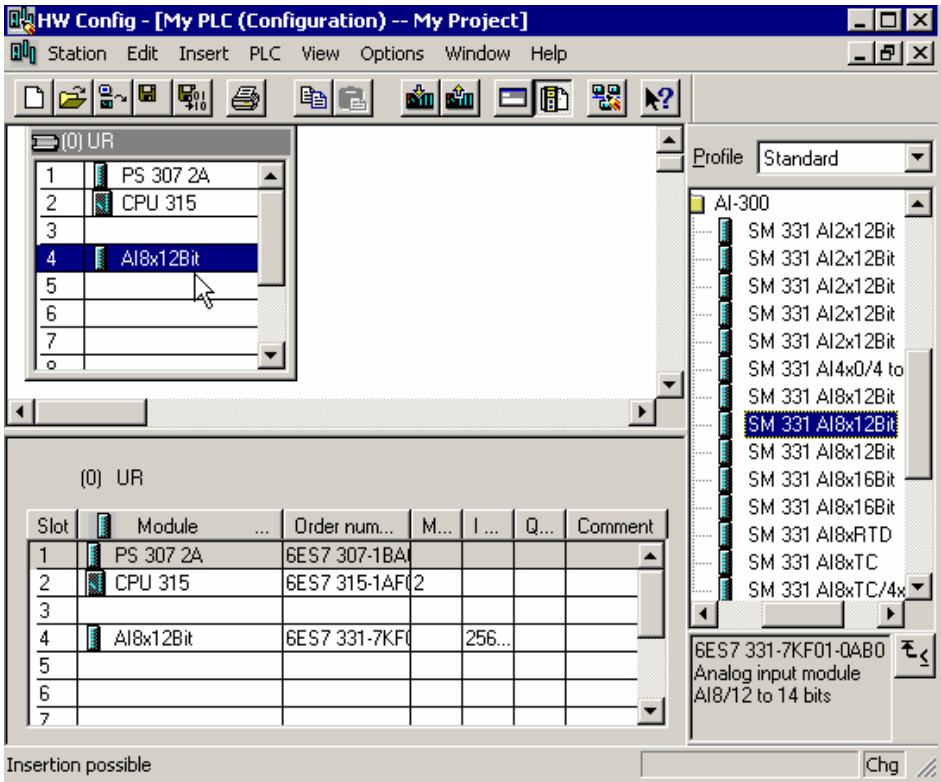
وبنفس الطريقة نختار (Power Supply) كالتالي:



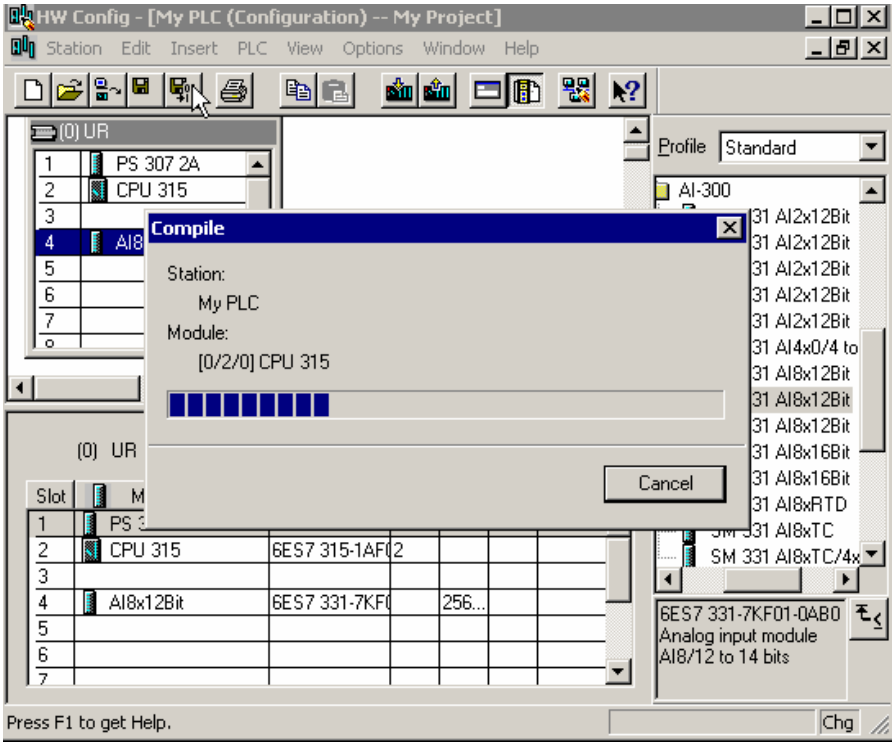
وبعدها نختار ال (CPU) كالتالي:



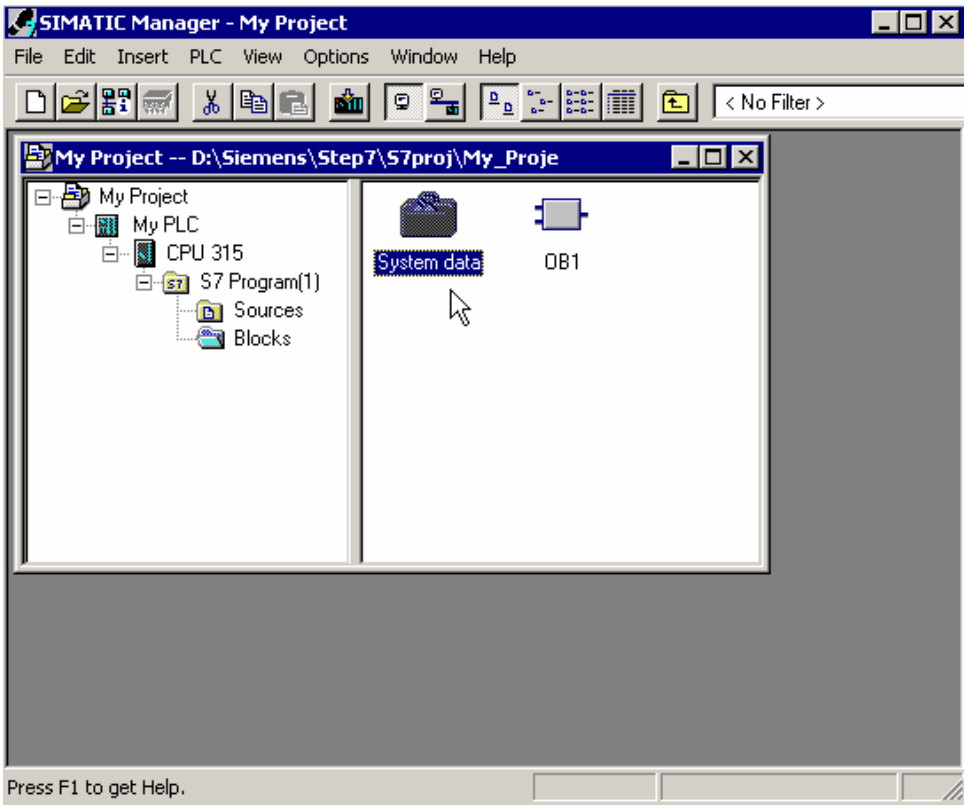
وبعدها نقوم باضافة الكارتات ابتداءا من (Slot) رقم (4) وذلك لان ال (Slot 3) مخصص لكارتات من نوع (IM) اي كارتات اتصال في حالة ربط (Rack) اضافي مع ال (Rack) الحالي:



نقوم بالضغط على الايقونة الخامسة من شريط الادوات (Save and Compile) كالتالي:



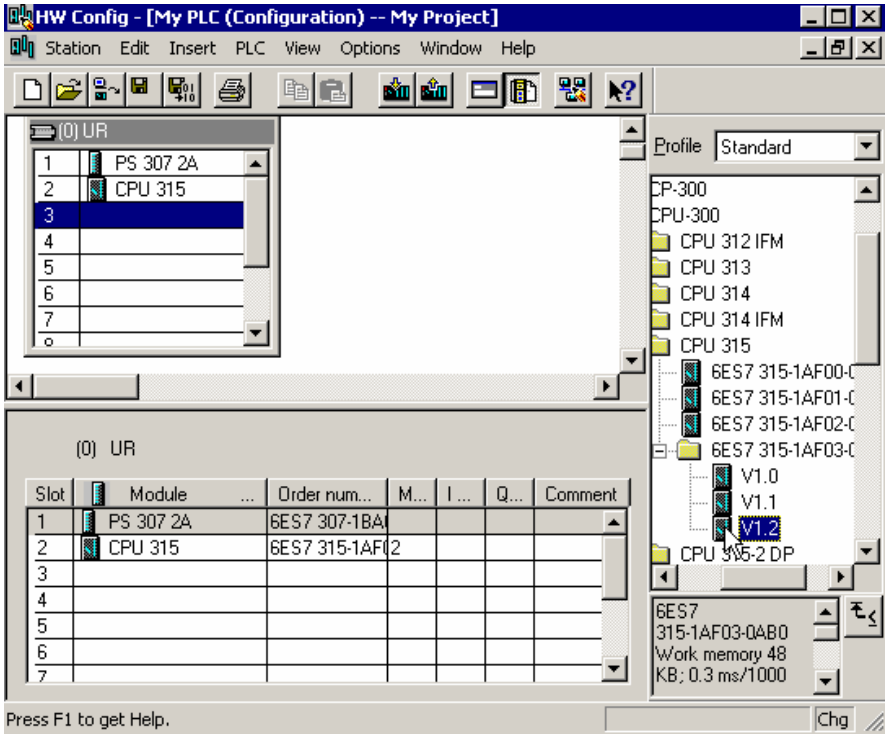
عند العودة الى نافذة ملفات المشروع نلاحظ تم اضافة الملف (System data)



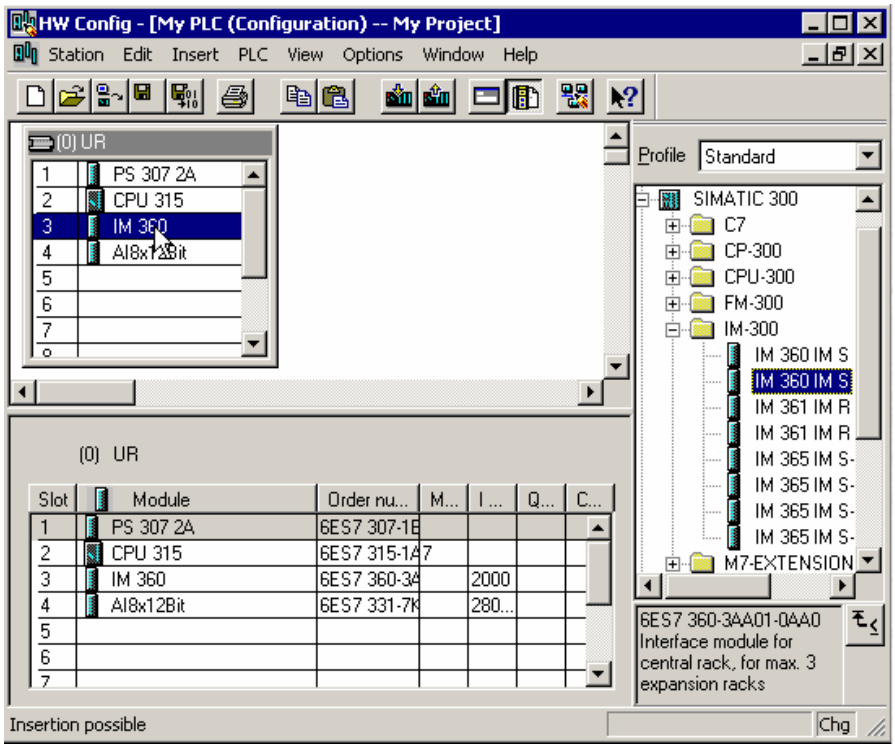
٣- اضافة (Rack) اخر

خطوات اضافة (Rack) آخر

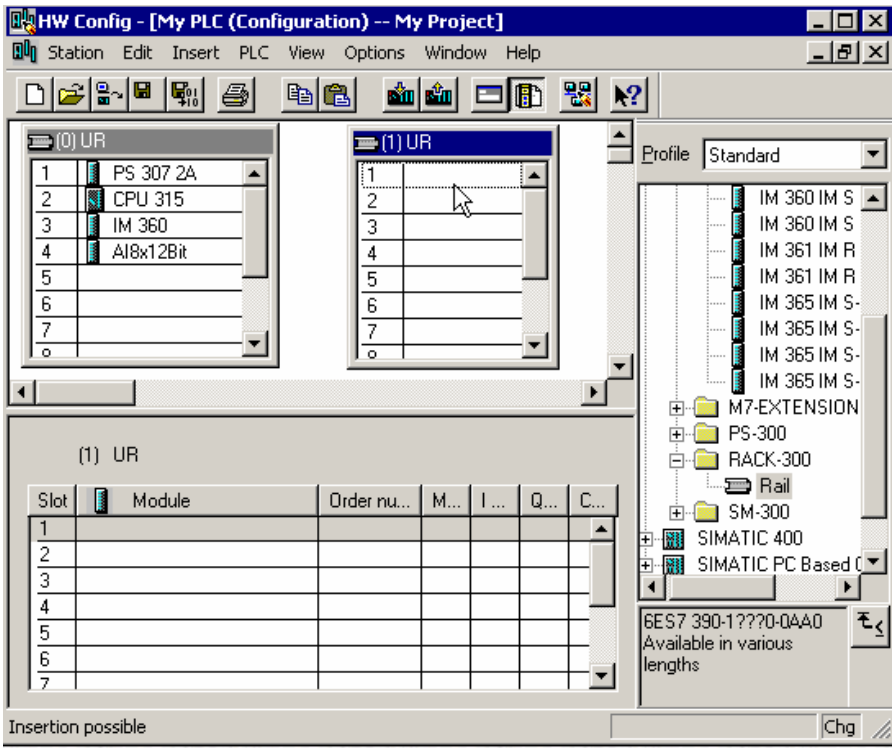
١- نقوم بتكوين نافذة الهاردوير كما تعلمنا سابقا كالتالي ونختار (Slot3) المخصص لكارتات الاتصال من نوع (IM)



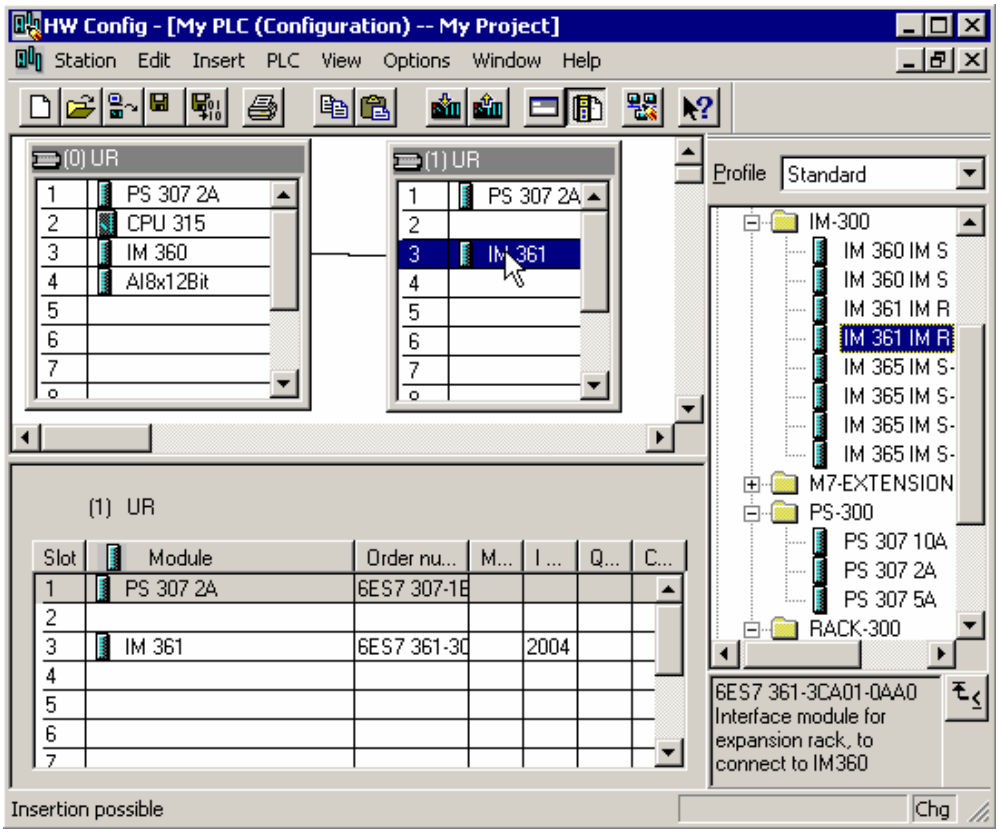
٢-نقوم باضافة الكارت (IM360) الى (Slot3) حيث يوضع هذا الكارت بال (Rack) الاول فقط



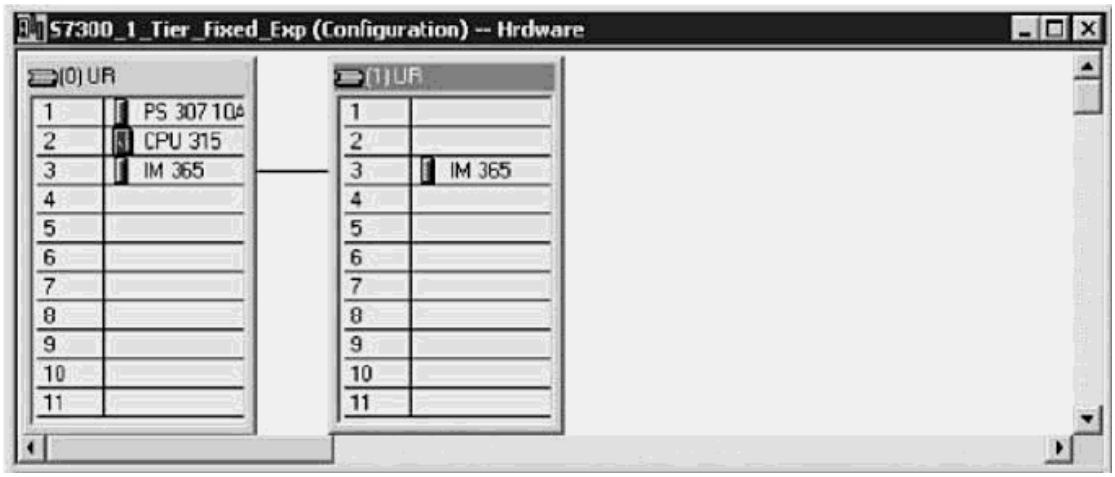
٣- نقوم باضافة (Rack) آخر الى نافذة الهاردوير:



٤- نقوم باضافة الكارتات التالية الى ال (Rack) الثاني:



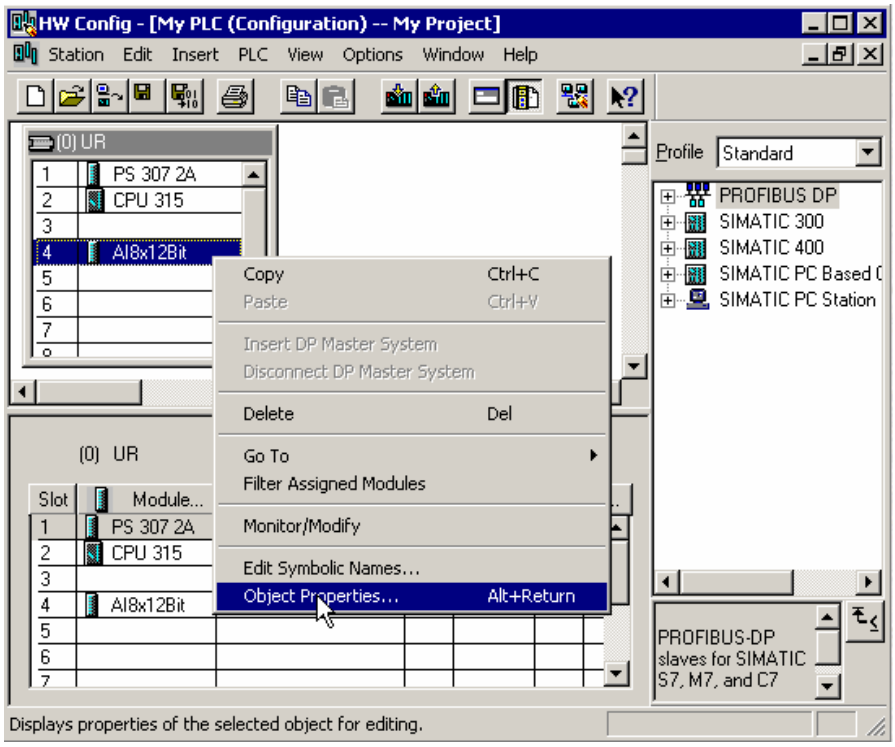
٥- سيتكون الارتباط بصورة تلقائية ويأخذ الارتباط عدة اشكال كالتالي:



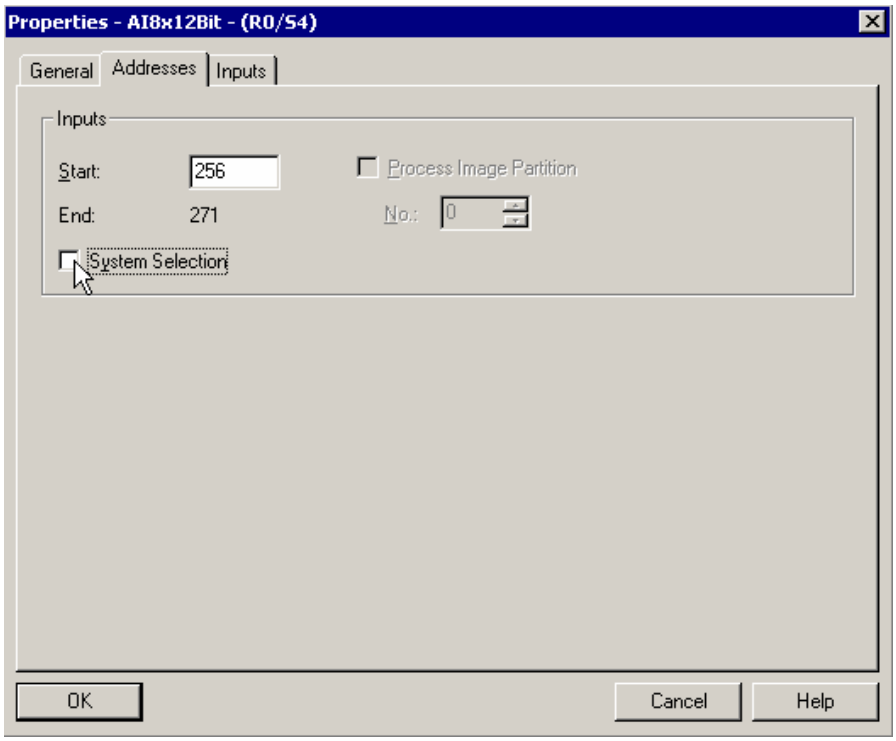
راجع جدول خصائص (IM Card) في دروس اليوم الاول

٤ـتغيير عناوين الكارتات

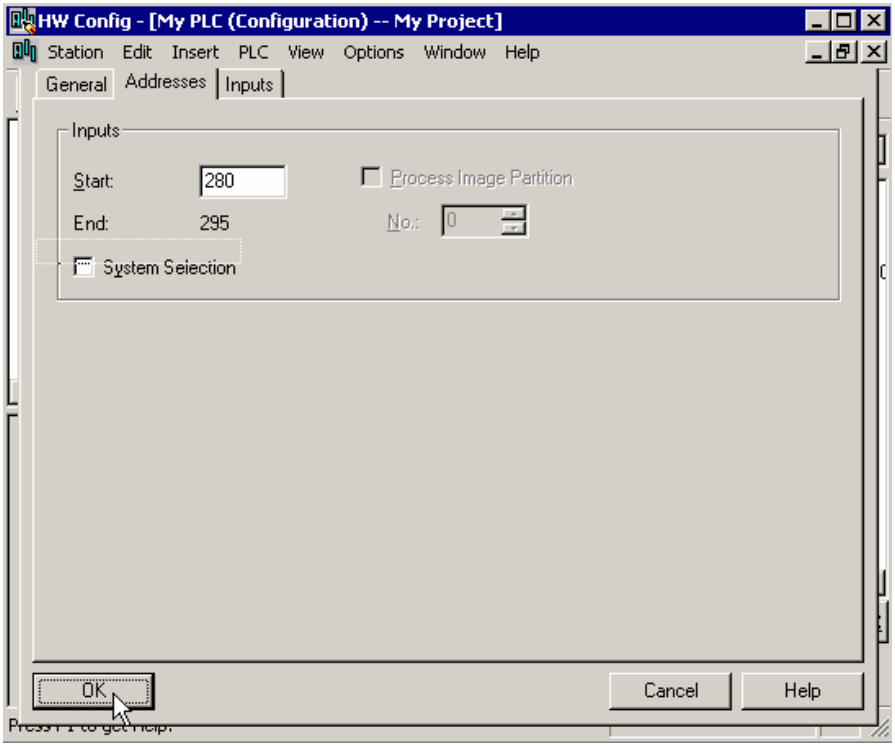
١ـمن نافذة الهاردوير نختار احد كارتات المداخل اوالمخارج و نضغط بالماوس الايمن ونختار (Object Properties)



٢ـ ستظهر النافذة التالية من خانة (Addresses) نقوم برفع علامة صح من مربع الاختيار (System Selection)

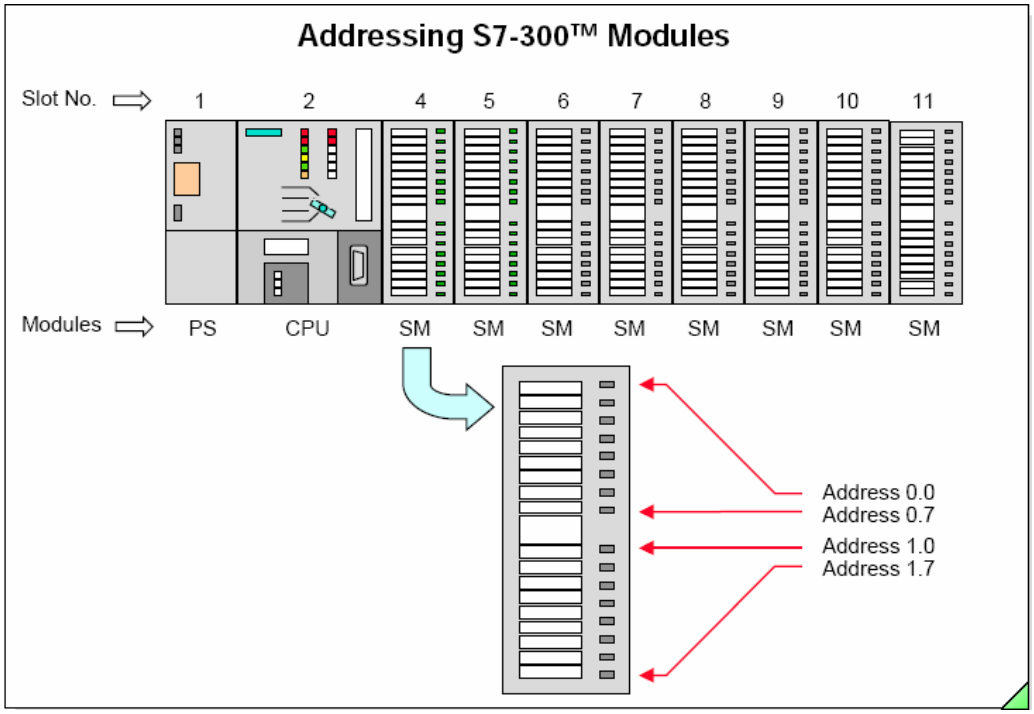


٣- نقوم بكتابة العنوان الجديد داخل مربع (Start) ونضغط (OK)

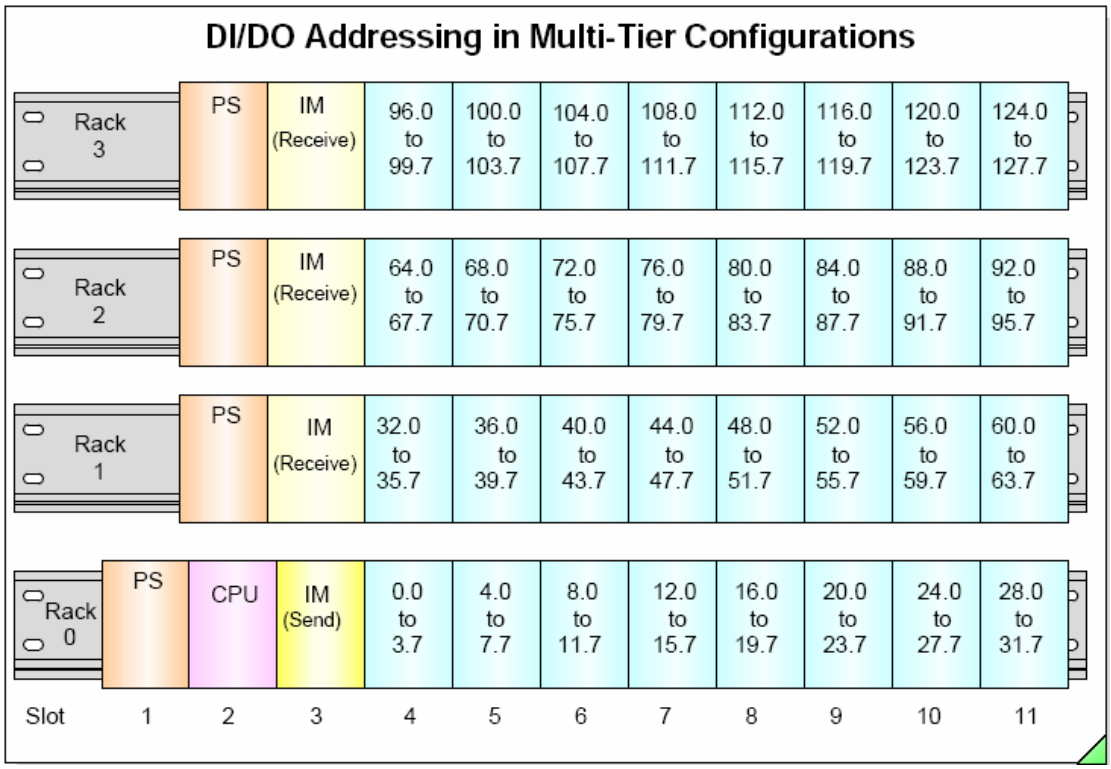


٤-ثم نقوم بحفظ التغييرات

الشكل التالي يبين العنوان لكارتات (16 bit)

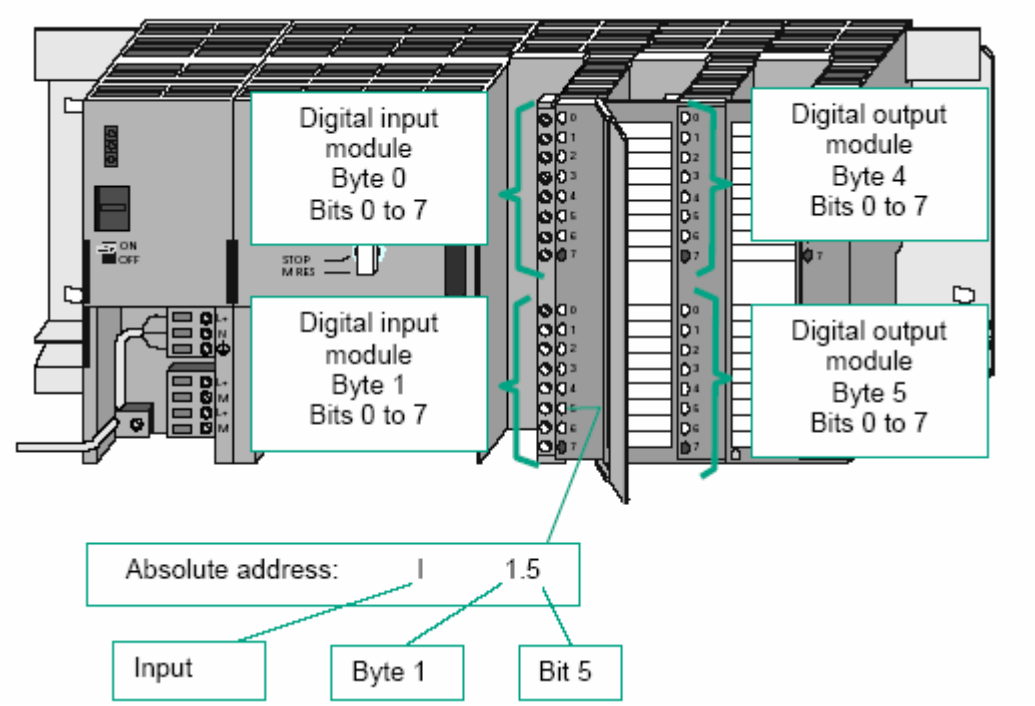


والشكل التالي يبين العناوين لكارتات (32 bit) لاكثر من (Rack)

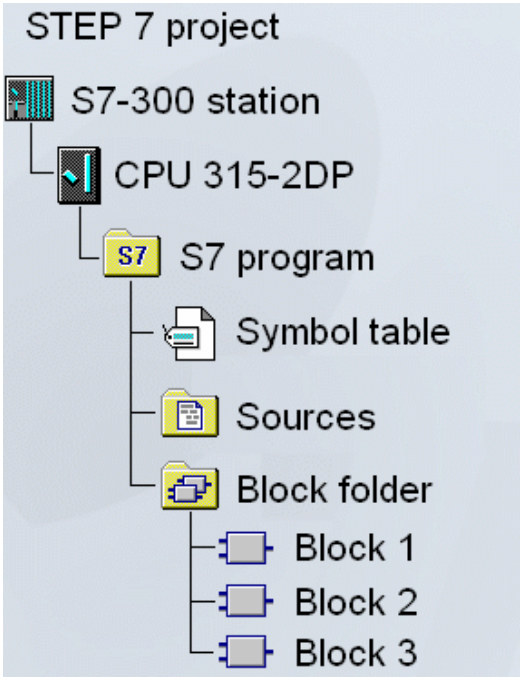


اليوم الثالث
١- البرمجة مع الرموز (Symbol)
٢- كتابة البرنامج المنطقي داخل الوحدة التنظيمية (OB1)
٣- جدول تعريف المتغيرات

١- تعريف الرموز (Symbol)
وهي عبارة عن اسماء تعريفية توضع للعناوين من اجل فهم البرنامج وسنتعرف على العناوين بشكل تفصيلي في درس لاحق ولكن الان سنأخذ فقط عناوين المداخل (I) والمخارج (Q) والشكل التالي يوضح كيفية عنونة الهاردوير

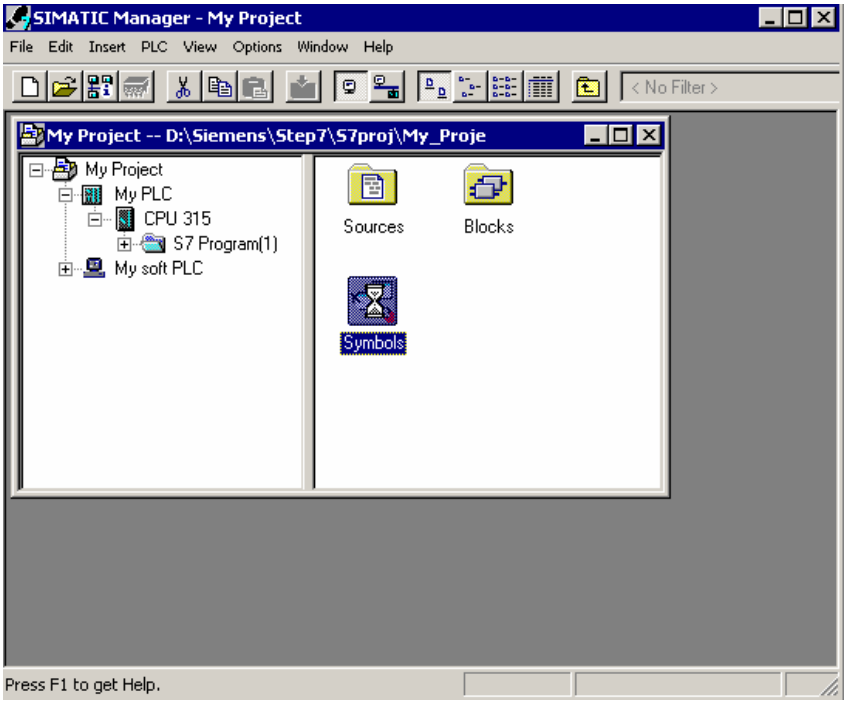


يتكون مشروع (Step7) من الاجزاء التالية:

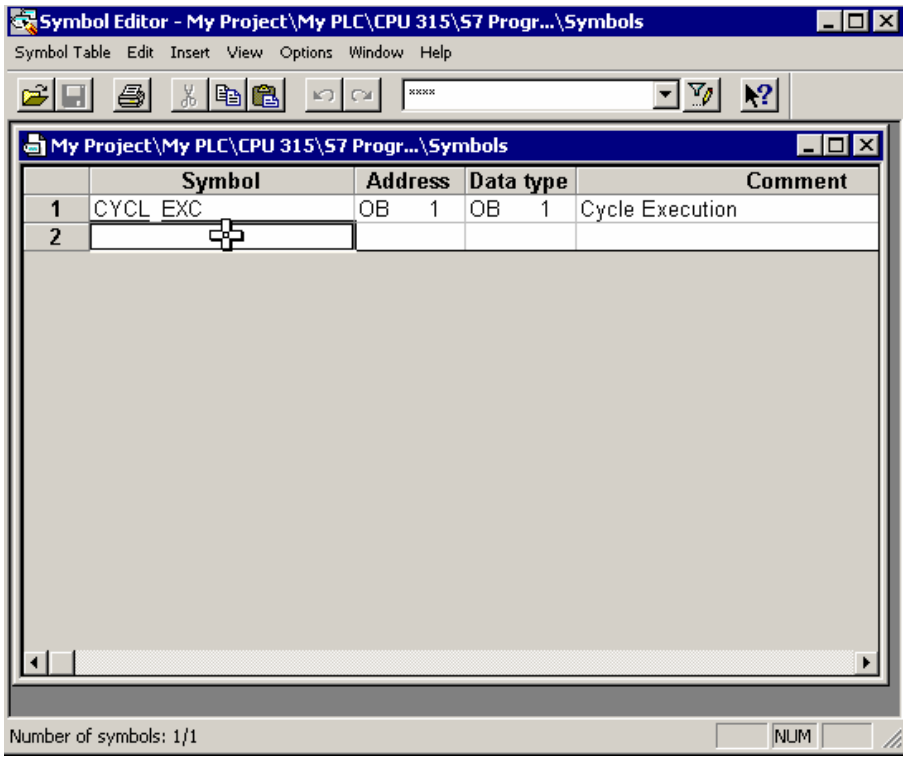


خطوات اضافة الرموز الى البرنامج:

أ- نضغط على المجلد (S7 program) في نافذة المشروع في جهة اليسار سيظهر الملف (Symbols) في النافذة اليمنى



ب- بالنقر المزدوج على الملف (Symbols) ستظهر النافذة التالية:



ج- نقوم بتغير (CYCL_EXC) الى (Main Program) مثلاً

	Symbol	Address	Data Type
1	Cycle Execution	OB 1	OB 1
2			

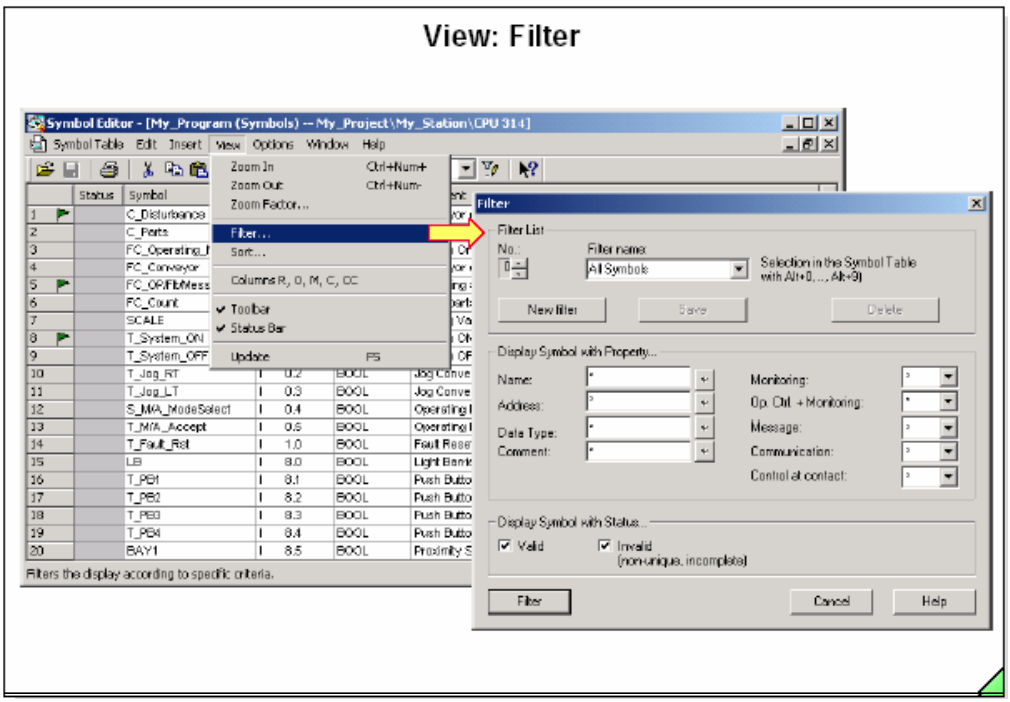
	Symbol	Address	Data Type
1	Main Program	OB 1	OB 1
2	Green Light	Q 4.0	BOOL

	Symbol	Address	Data Type
1	Main Program	OB 1	OB 1
2	Green Light	Q 4.0	BOOL
3	Red Light	Q 4.1	BOOL

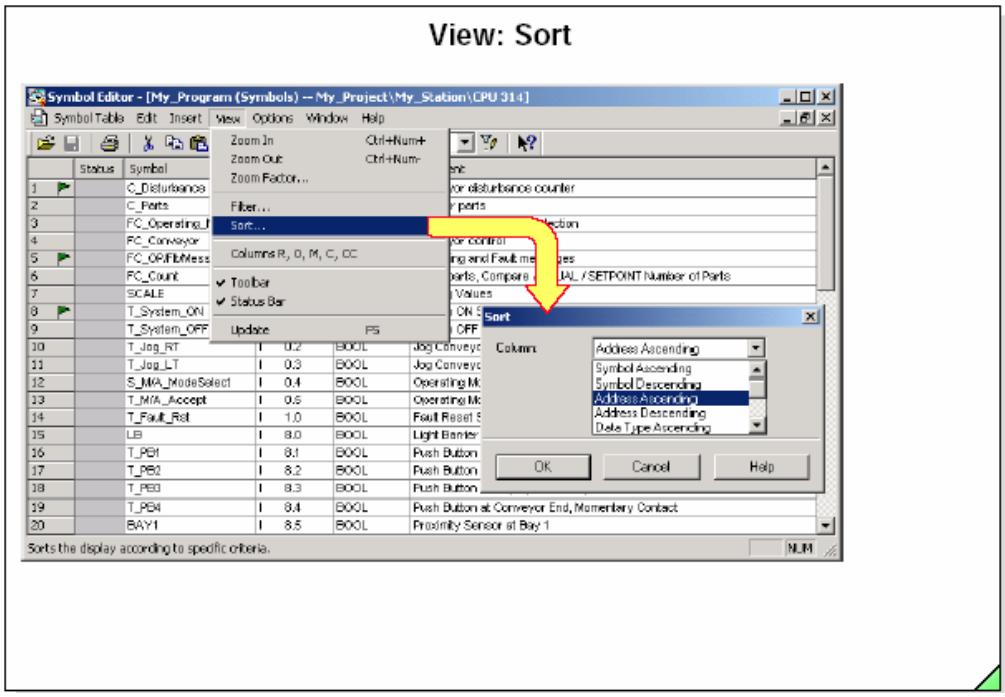
د- نقوم بكتابة الرمز (Green Light) واعطائه العنوان (Q4.0) وهو البت الاول من كارت الاخراج الذي يحوي البايت الرابع ويختلف تسلسل البايتات باختلاف سعة كل كارت وسنتعلم لاحقا كيفية تنسيق العناوين حسب الرغبة

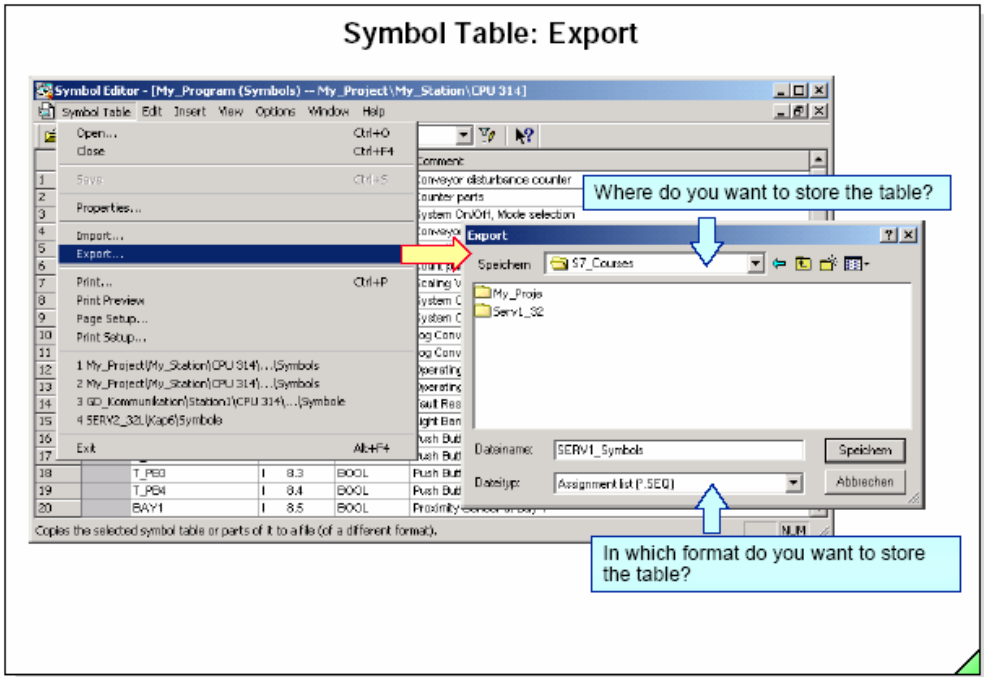
ه- نقوم بحفظ التغييرات من قائمة (File-Save)

و- يمكن اجراء عمليات البحث والتعديل والترشيح والفرز على ملف (Symbols) كالتالي:



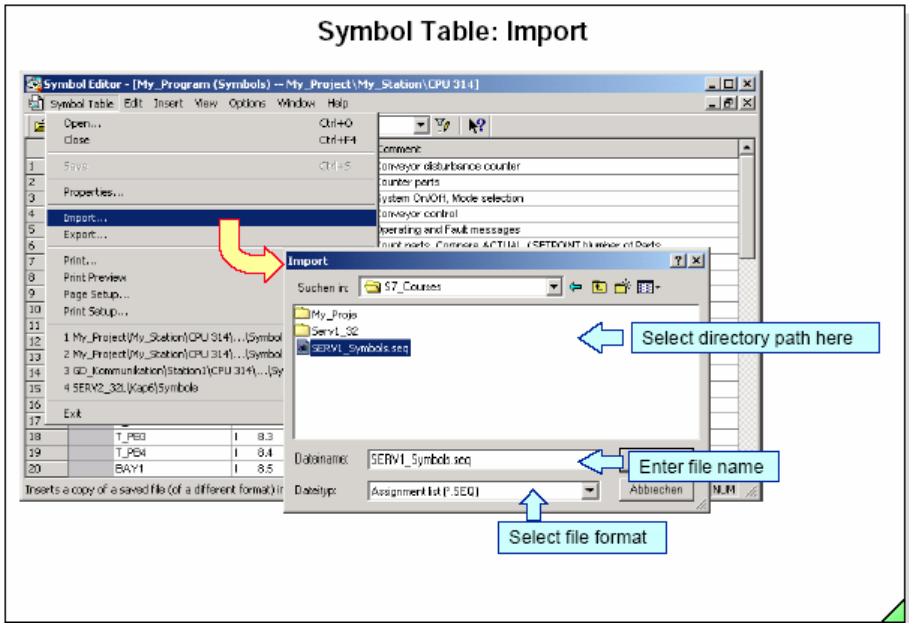
لعرض رموزمعينة نكتب اول حرف ثم علامة نجمة مثلا لعرض كلمة (Sensor) نكتب في حقل (Name) العبارة (S*) او لعرض المداخل فقط نكتب (I*) في حقل (Address) وهكذا





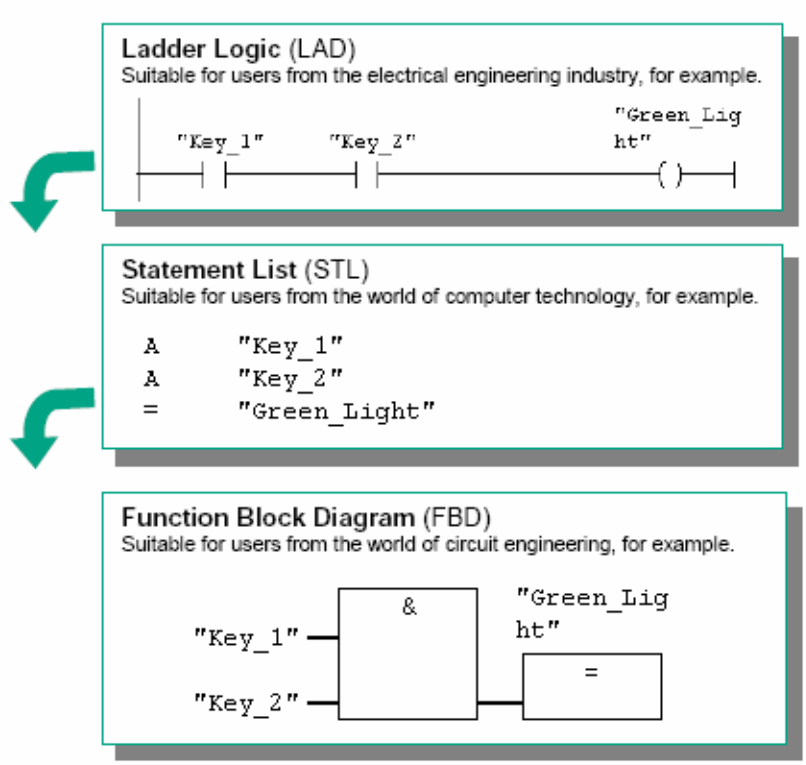
يمكن تصدير او استيراد ملف الرموز للبرامج التالية:

- ASCII Format (*.ASC)
  - Notepad
  - Word
- Data Interchange Format (*.DIF)
  - EXCEL
- System Data Format (*.SDF)
  - ACCESS
- Assignment List (*.SEQ)
  - STEP 5 assignment list



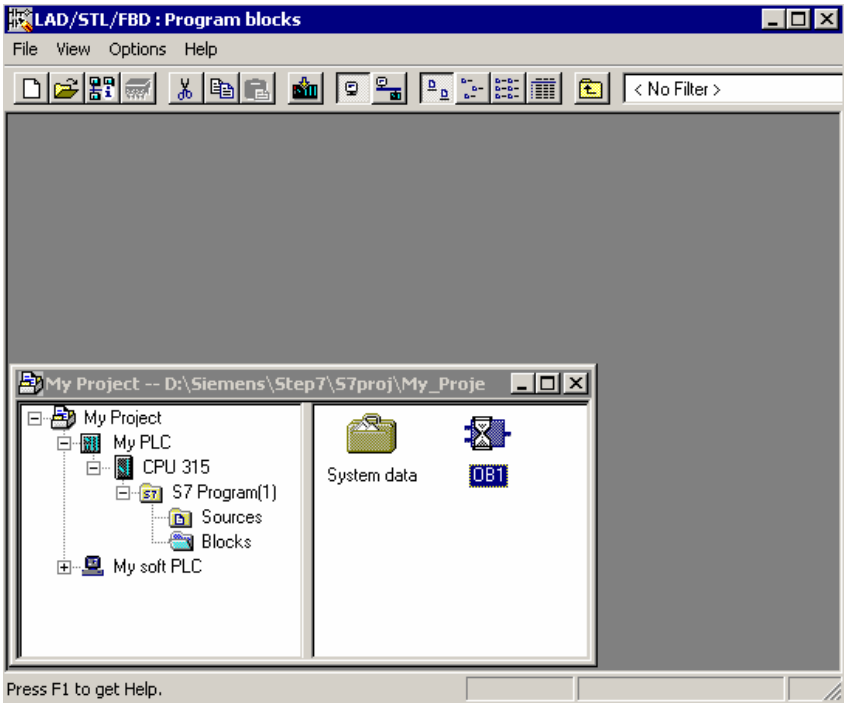
٢ـ كتابة البرنامج المنطقي داخل الوحدة التنظيمية (OB1)

يمكن كتابة البرنامج المنطقي بثلاثة لغات كالتالي:

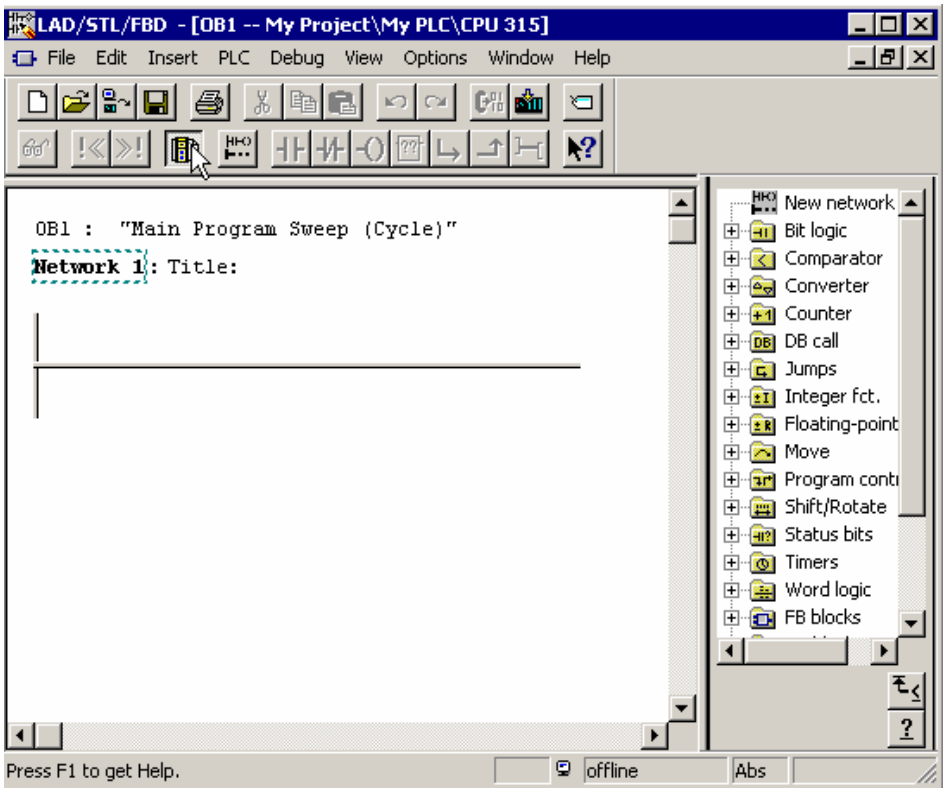


خطوات كتابة البرنامج المنطقي داخل (OB1)

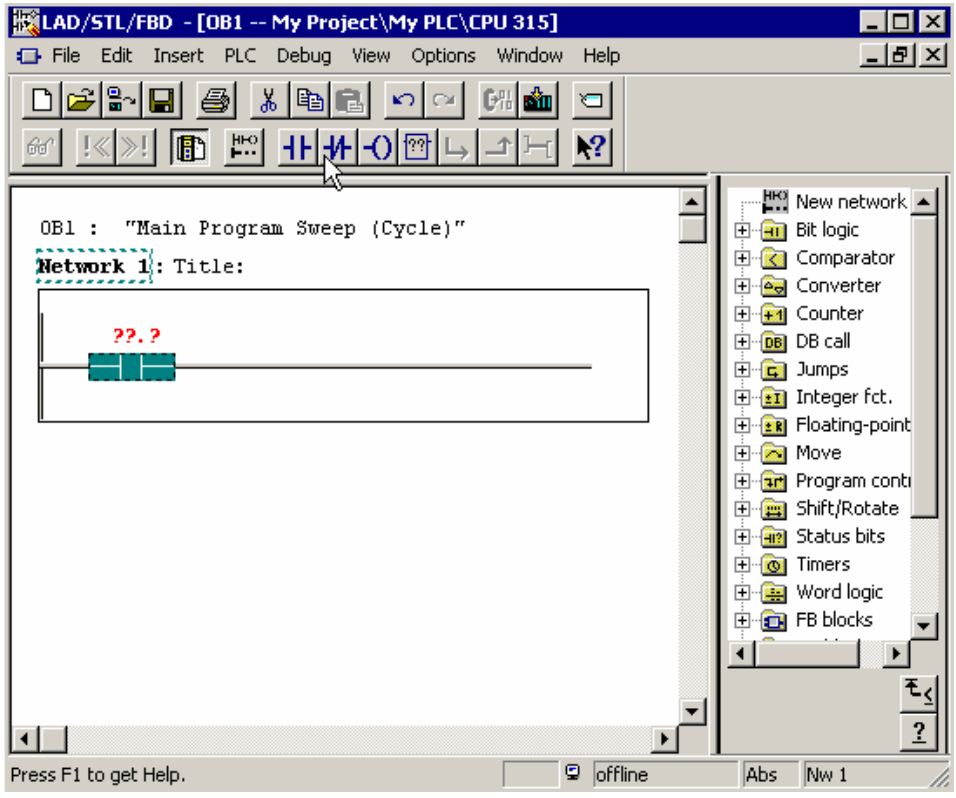
١ـ ننقر نقرا مزدوجا على الوحدة التنظيمية (OB1)



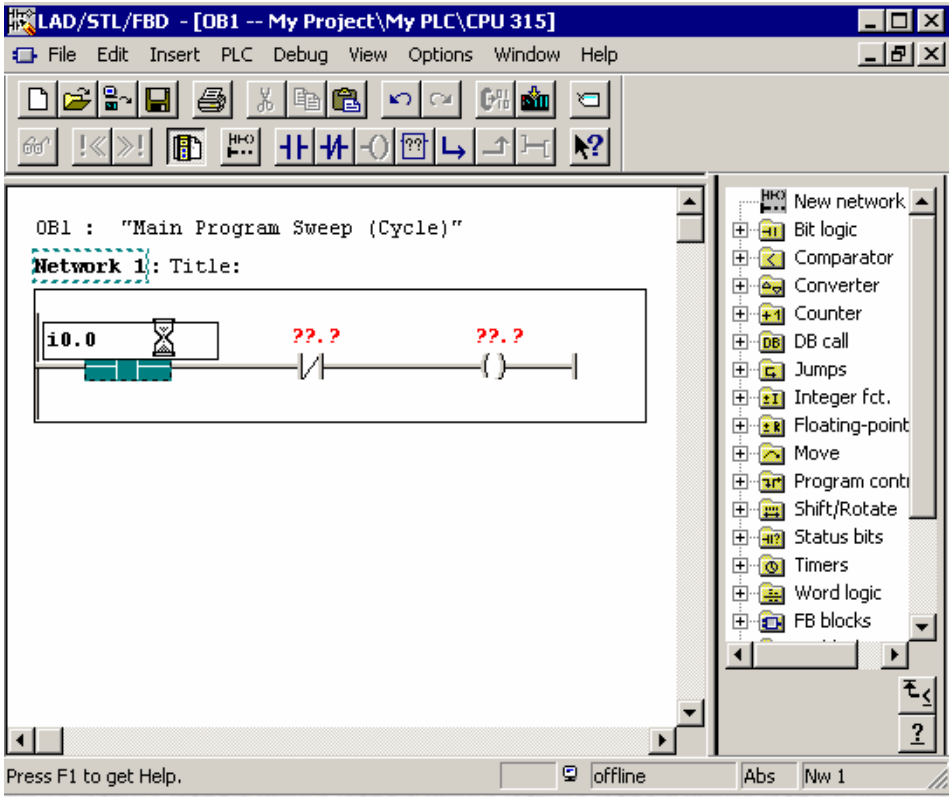
٢ـ ستظهر النافذة التالية نضغط على الايقونة الخاصة لاظهار العناصر البرمجية او من قائمة (Insert) نختار (Program Elements):



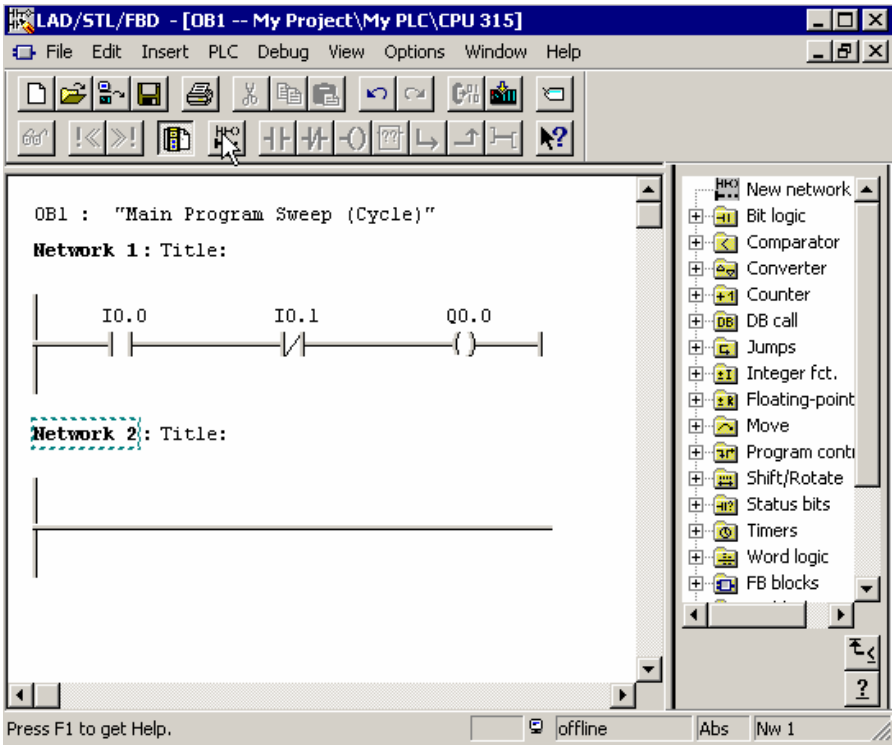
٣ـ نقوم باضافة العناصر من شريط الادوات وذلك بالضغط على الخط السلمي اسفل (Network1) ثم الضغط مرة واحدة على العناصر لاضافتها الى الخط السلمي:



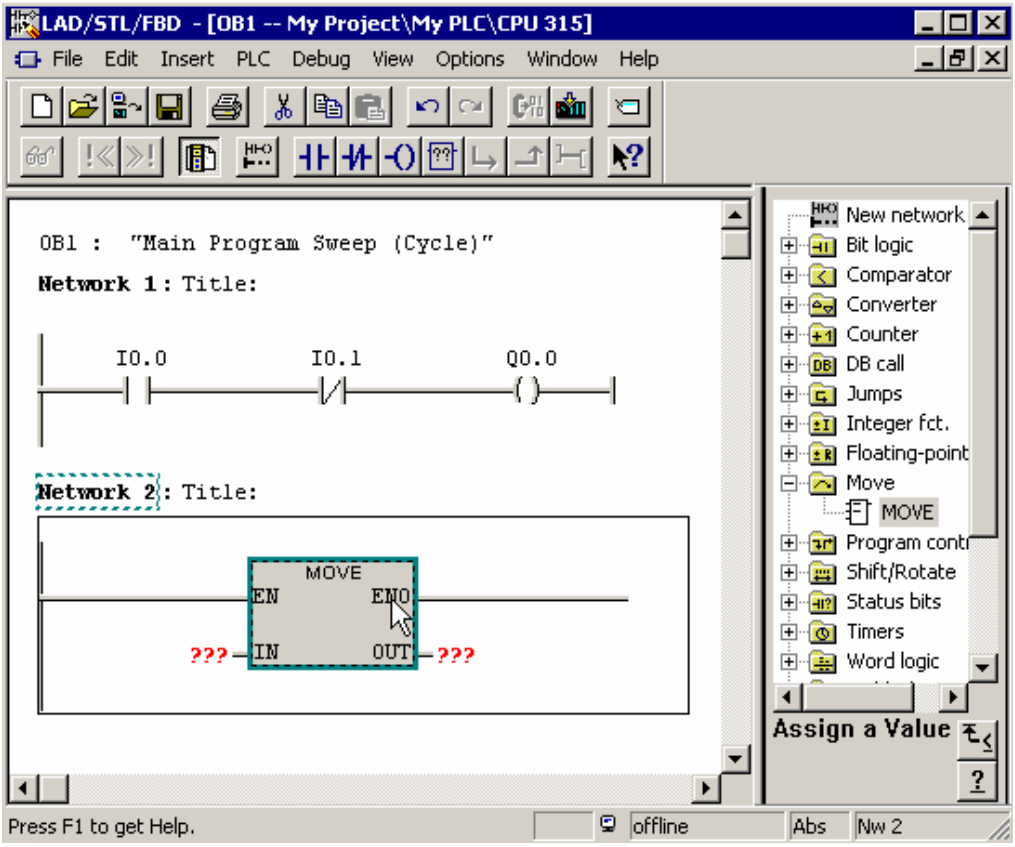
٤- بعد اضافة العناصر تقوم بوضع العناوين فوقها بدل من علامة الاستفهام الحمراء



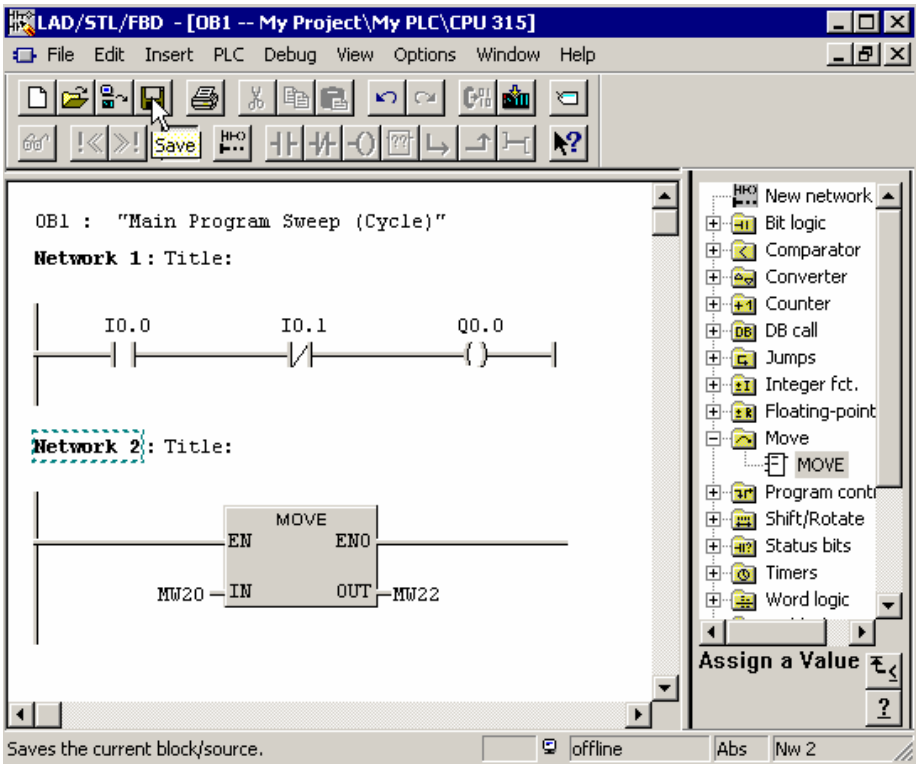
٥- بعد اكمال العناوين للمداخل (I) والمخارج (Q) نقوم باضافة سطر آخر من الايقونة (New Network) في شريط الادوات او من قائمة (Insert) نختار (Network)



٦ـ نقوم بالضغط على السطر اسفل (Network2) ثم نسحب من نافذة العناصر العنصر (Move) ونضعه فوق السطر الثاني

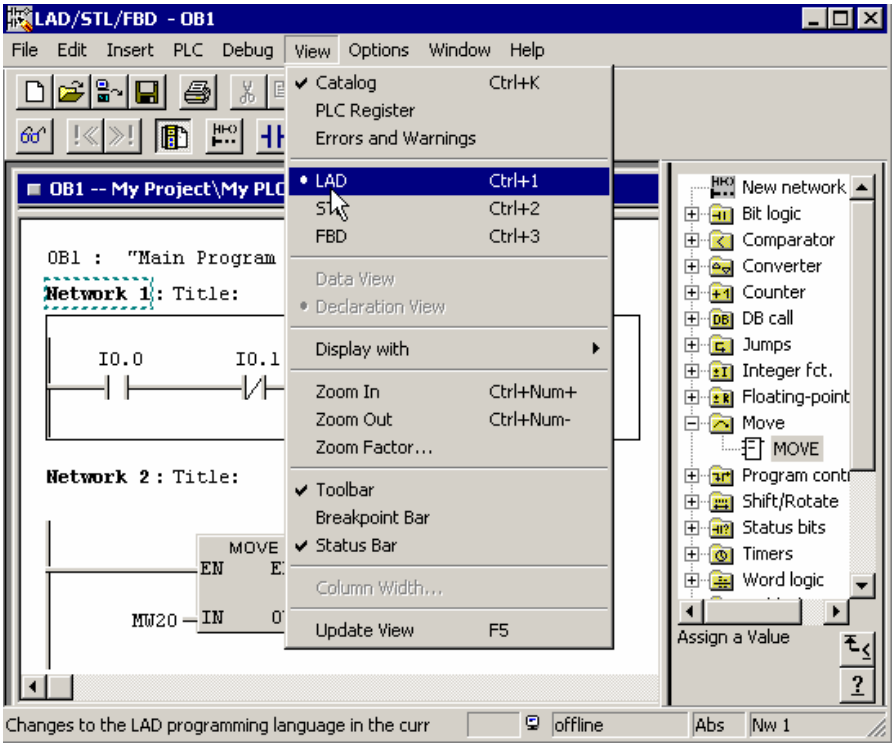


٧ـ نقوم بوضع العناوين بدل علامة الاستفهام ونحفظ التغييرات

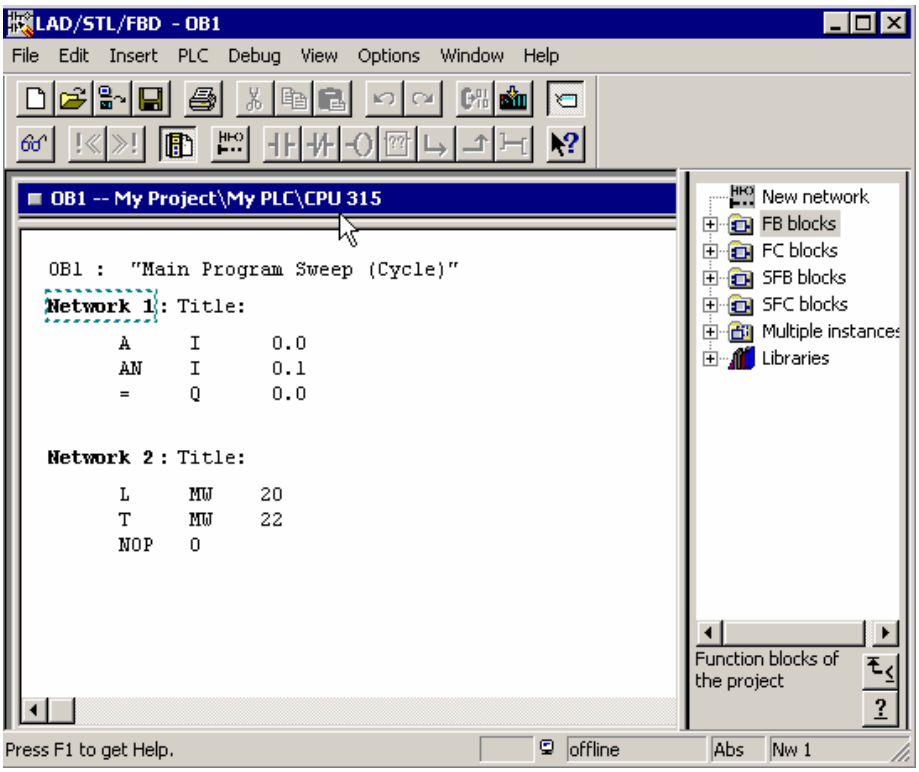


العناوين من نوع (M) هي عناوين تحتفظ بالقيم داخل الذاكرة بصورة مؤقتة وسيتم شرحها في فصل العنونة

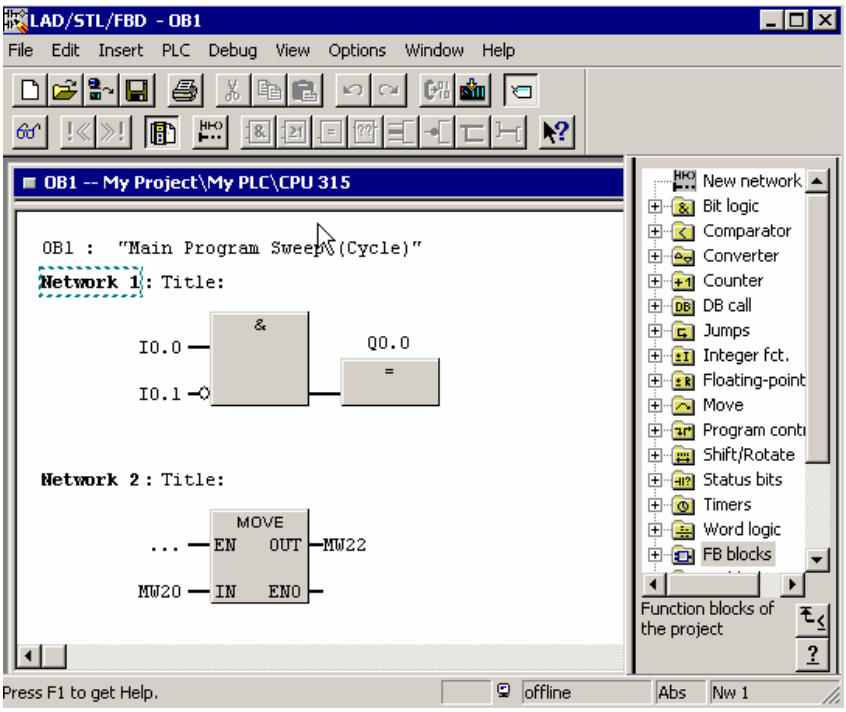
٨- يمكن تغيير لغة البرمجة وذلك من قائمة (View) نختار اللغة وسيتحول البرنامج المكتوب الى لغة آخرى



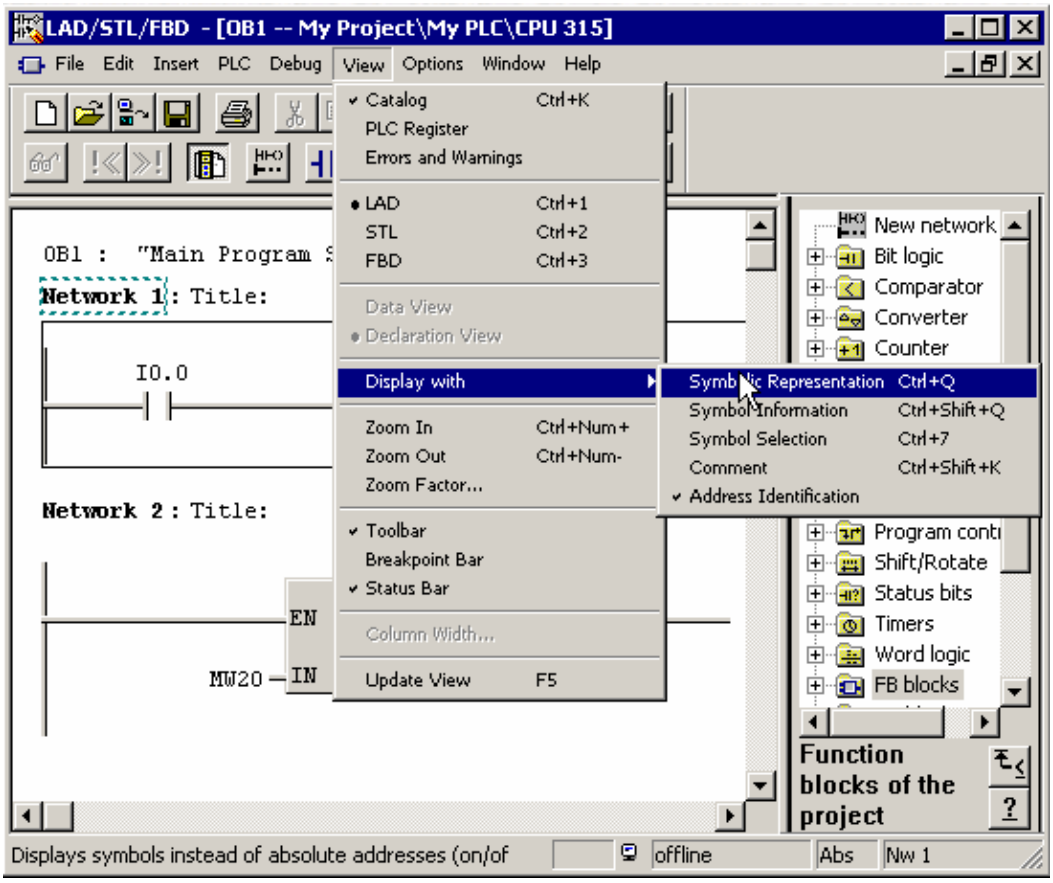
٩- عند اختيار (STL) يكون شكل البرنامج كالتالي:



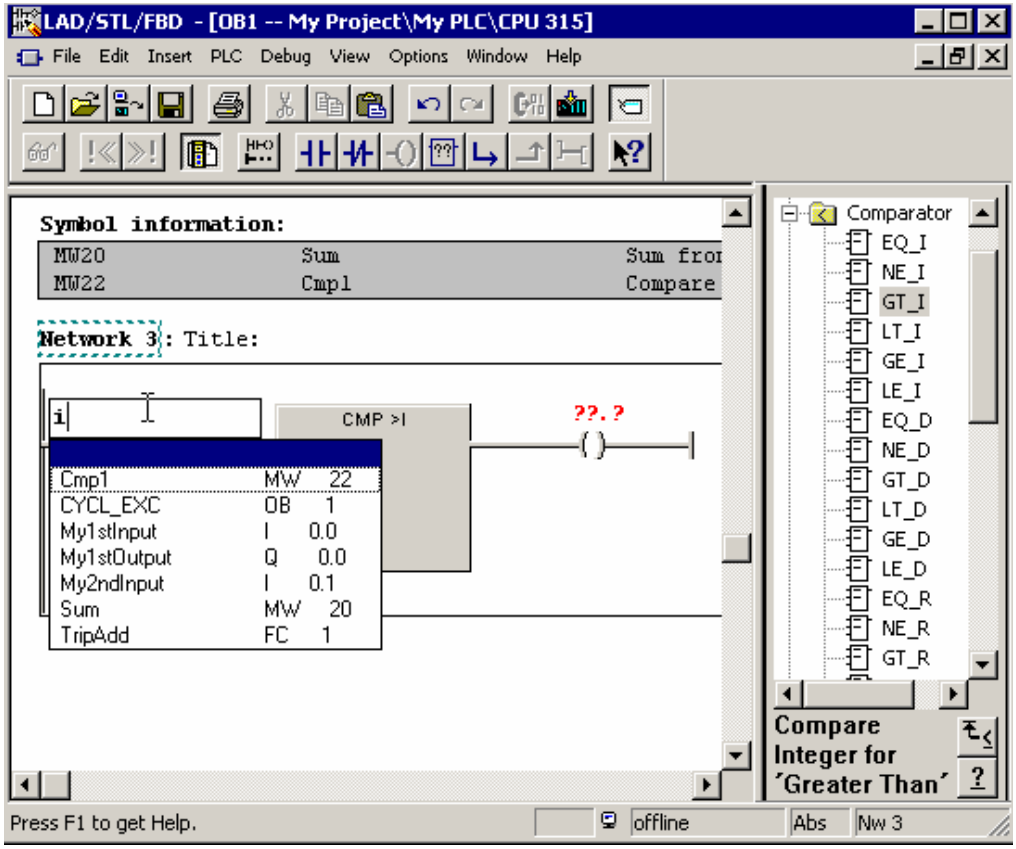
١٠- وعند اختيار (FBD) يكون شكل البرنامج كالتالي:



١١-يمكن عرض الرموز على العناوين كالتالي:



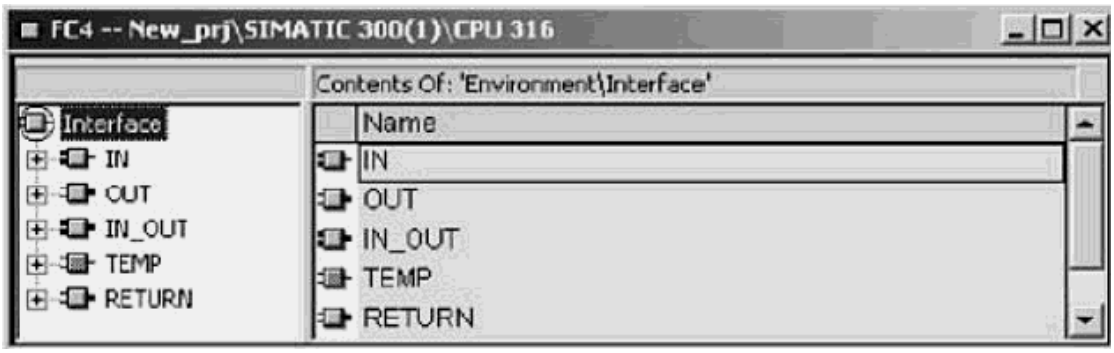
١٢ـ ويمكن كتابة الرموز مباشرة بدل العناوين وذلك بكتابة اي حرف ستظهر قائمة الرموز الملائمة للايعازفنختار واحدة منها



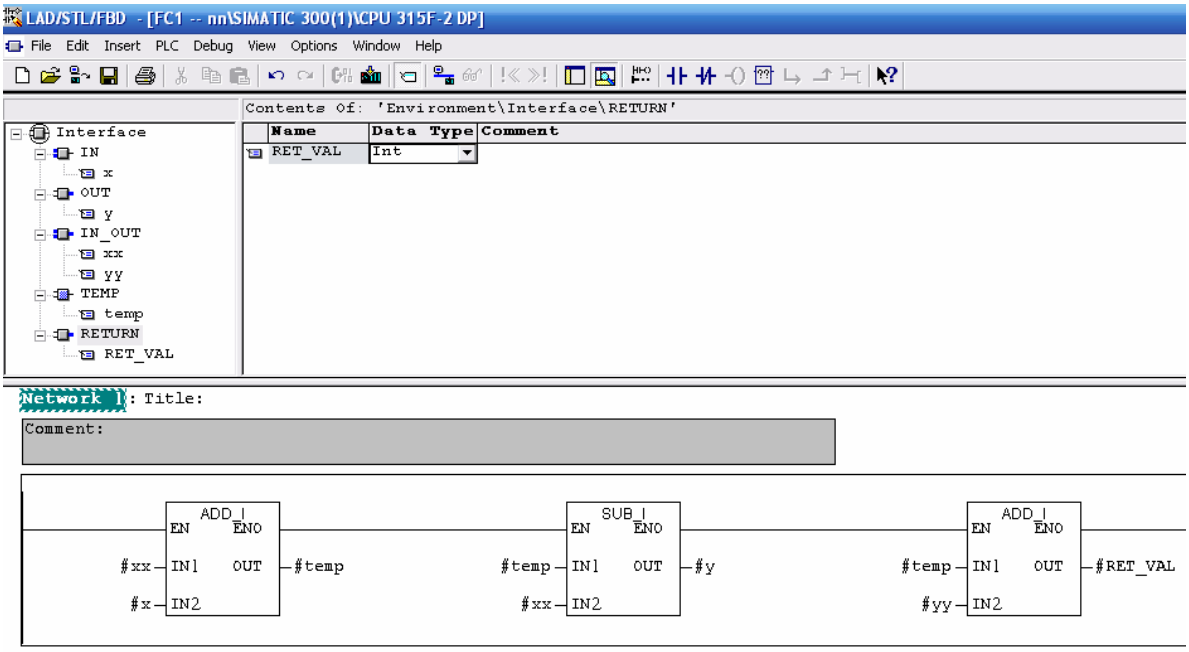
٣ـجدول تعريف المتغيرات : ويسمى ايضا جدول التصاريح (Declaration Table) وهو عبارة عن جدول موجود ضمن ملفات كتابة البرامج المنطقية مثل (OB,FC,FB) يستخدم لتعريف متغيرات يمكن استخدامها داخل البرنامج كما في لغات البرمج الاعتيادية فمثلا في لغة فجول بيسك يتم تعريف المتغير كالتالي:

Dim Var1 AS Integer

ويكون جدول المتغيرات اعلى ملفات كتابة البرامج وشكله كالتالي:



ويحتوي على الخانات التالية:
(In): يستخدم لتعريف المتغيرات الداخلة الى الدوال
(Out): يستخدم لتعريف المتغيرات الخارجة الى الدوال
(In_Out): يستخدم لكلا الموضعين
(Temp): يستخدم لخزن القيم بصورة مؤقتة ولا يدخل ضمن متغيرات الدالة
(Return):متغير محجوز يمكن استعماله في اي مكان من الدالة ويكون جزء من متغيراتها
(Stat):يكون فقط داخل جدول تصاريح (FB) ويخزن قيم ثابتة سنستخدمه لاحقا



في الشكل التالي قمنا بتعريف المتغيرات كالتالي:

x IN,
y Out,
xx,yy IN_OUT
Temp Temp

لاحظ استخدامها داخل العناصر
يتكون جدول التصاريح من الخانات التي ذكرنا انواعها وكل خانة تتكون من الحقول التالية:
(Name): يكتب به اسم المتغير
(Data Type): يوضع به حدود قيم المتغير مثل (byte,int) سنتعلمها لاحقاً
(Comment): وصف للمتغير

اليوم الرابع
١ـ العنونة
٢ـ انواع البيانات داخل (Step7)

١ـ العنونة : وهي عبارة عن اماكن محجوزة بالذاكرة تستخدم لاارسال او استلام البيانات من والى كارتات المداخل والمخارج مع ذاكرة (CPU) وانواعها كالتالي:

Address Area	Access via Units of Following Size	S7 Notation (IEC)	Description
Process image input table	Input (bit)	I	At the beginning of the scan cycle, the CPU reads the inputs from the input modules and records the values in this area.
	Input byte	IB	
	Input word	IW	
	Input double word	ID	
Process image output table	Output (bit)	Q	During the scan cycle, the program calculates output values and places them in this area. At the end of the scan cycle, the CPU sends the calculated output values to the output modules.
	Output byte	QB	
	Output word	QW	
	Output double word	QD	
Bit memory	Memory (bit)	M	This area provides storage for interim results calculated in the program.
	Memory byte	MB	
	Memory word	MW	
	Memory double word	MD	
Timers	Timer (T)	T	This area provides storage for timers.
Counters	Counter (C)	C	This area provides storage for counters.
Data block	Data block, opened with "OPN DB":	DB	Data blocks contain information for the program. They can be defined for general use by all logic blocks (shared DBs) or they are assigned to a specific FB or SFB (instance DB).
	Data bit	DBX	
	Data byte	DBB	
	Data word	DBW	
	Data double word	DBD	
	Data block, opened with "OPN DI":	DI	
	Data bit	DIX	
	Data byte	DIB	
	Data word	DIW	
	Data double word	DID	

Local data	Local data bit	L	This area contains the temporary data of a block while the block is being executed. The L stack also provides memory for transferring block parameters and for recording interim results from Ladder Logic networks.
	Local data byte	LB	
	Local data word	LW	
	Local data double word	LD	
Peripheral (I/O) area: inputs	Peripheral input byte	PIB	The peripheral input and output areas allow direct access to central and distributed input and output modules (DP).
	Peripheral input word	PIW	
	Peripheral input double word	PID	
Peripheral (I/O) area: outputs	Peripheral output byte	PQB	
	Peripheral output word	PQW	
	Peripheral output double word	PQD	

الجدول اعلاه يبين انواع العناوين داخل الذاكرة وهي صورة للبيانات المستلمة او المرسلة كالتالي:

١- عناوين الادخال (I): وهذه العناوين تقرا البيانات من كارتات الادخال الى الذاكرة وتاخذ عدة صور حسب حجم البيانات التي يتم قراءتها فمثلا (I) اذا كانت البيانات ب (bit) وهو المستعمل في كارتات (Digital input) او (IB) اذا كانت البيانات ب (Byte= 8 bit) او(IW) اذا كانت البيانات ب (Word= 16 bit) او (ID) اذا كانت البيانات ب (Double Word= 32 bit)

٢- عناوين الاخرج (Q): وهذه العناوين ترسلا البيانات الى كارتات الاخراج من الذاكرة وتاخذ عدة صور حسب حجم البيانات التي يتم قراءتها فمثلا (Q) اذا كانت البيانات ب (bit) وهو المستعمل في كارتات (Digital Output) او (QB) اذا كانت البيانات ب (Byte= 8 bit) او(QW) اذا كانت البيانات ب (Word= 16 bit) او (QD) اذا كانت البيانات ب (Double Word= 32 bit)

٣- العناوين الداخلية (M): وهذه العناوين هي عناوين وسطية مخزونة داخل الذاكرة تستخدم لخزن البيانات الناتجة من العمليات الرياضية والمنطقية اي وظيفتها الاستلام من عناوين المداخل والارسال الى عناويين المخارج بعد دخولها بالعمليات الحسابية وتاخذ عدة صور حسب حجم البيانات التي يتم قراءتها فمثلا (M) اذا كانت البيانات ب (bit) او (MB) اذا كانت البيانات ب (Byte= 8 bit) او(MW) اذا كانت البيانات ب (Word= 16 bit) او (MD) اذا كانت البيانات ب (Double Word= 32 bit)

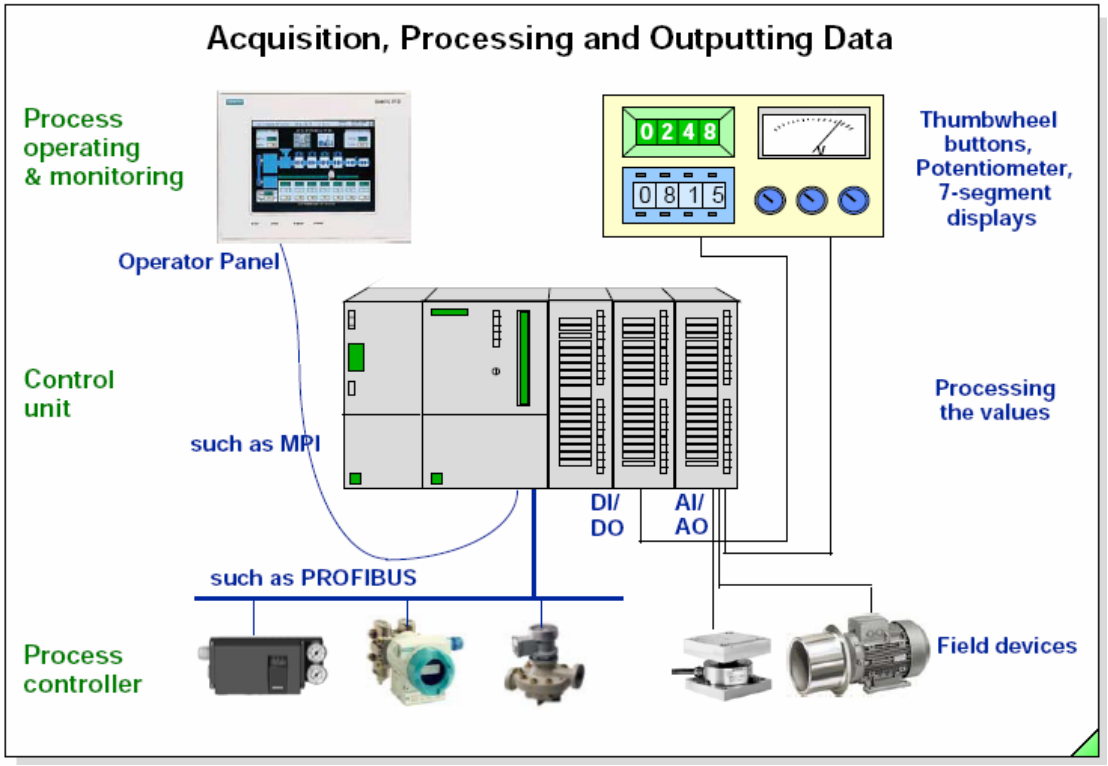
٤- عناوين المؤقتات (T): تستخدم كعناويين لايعازات المؤقتات سيتم شرحها لاحقاً

٥-- عناوين العدادات (C): تستخدم كعناويين لايعازات العدادات سيتم شرحها لاحقاً

٦-عناوين ملفات البيانات (DB): عناوين لملفات تشبه ملفات قواعد البيانات يتم قراءة وكتابة البيانات من واليها عند الحاجة وسيتم شرحها بالتفصيل في فصل لاحق

٧- العناوين المحلية (L): تقرأ البيانات من الذواكر المجوزة لملفات كتابة البرنامج وانشاء الل\ه سيتم التطرق عليها في الجزء الثاني لانها لاتلائم مستوى الدورة المبتدئ جداً

٨- العناوين الخارجية (PI/PQ):وهي عناوين تاخذ قيمها من ذواكر خارجية مثل ذواكرالشاشات المبرمجة (OPx) او ذواكر (Variable Speed Drive) او ذواكر (CPU) آخر عن طريق الربط (Master/Slave) الذي سيتم شرح برمجته في الجزء الثاني انشاء الله وتأخذ الاحجام التالية (PIB,PIW,PID) للمداخل و(PQB,PQW,PQD) والعناوين (PIW,PQW) الاكثر استخداما لارتباطها بقيم او كارتات (Analog) مثل (AI,AO)



سنأخذ بعض الامثلة عند تمثيل العناوين كالتالي:

Area	Identifier	Access Units	Example	Description of Example
Input Image (PII)	I	I = Bit	I 30.7	Input Byte 30, bit 7
		IB = Byte	IB 30	Input Byte 30
		IW = Word	IW 30	Input Word 30; bytes 30-31
		ID = Double word	ID 30	Input Double word 30; bytes 30-33
Output Image (PIQ)	Q	Q = Bit	Q 44.7	Output Byte 44, bit 7
		QB = Byte	QB 44	Output Byte 44
		QW = Word	QW 44	Output Word 44; bytes 44-45
		QD = Double word	QD 44	Output Double word 44; bytes 44-47
Bit Memory	M	M = Bit	M 23.6	Memory Byte 23, bit 6
		MB = Byte	MB 23	Memory Byte 23
		MW = Word	MW 24	Memory Word 24; bytes 24-25
		MD = Double word	MD 8	Memory Double word 8; bytes 8-11
Counter	C	C = Counter	C 64	Counter 64
Timer	T	T = Timer	T 12	Timer 12
Local Stack	L	L = Bit	L 2.7	Local Memory Byte 2, bit 7
		LB = Byte	LB 2	Local Memory Byte 2
		LW = Word	LW 2	Local Memory Word 2; bytes 2-3
		LD = Double word	LD 2	Local Memory Double word 2
Peripheral Inputs & Peripheral Outputs	PI	PIB = Byte	PIB 44	Peripheral Input Byte 44
		PIW = Word	PIW 66	Peripheral Input word 66
		PID = Double word	PID 82	Peripheral Input Double word 82
	PQ	PQB = Byte	PQB 44	Peripheral Output Byte 44
		PQW = Word	PQW 66	Peripheral Output word 66
		PQD = Double word	PQD 82	Peripheral Output Double word 82

٢-انواع البيانات داخل (Step7)

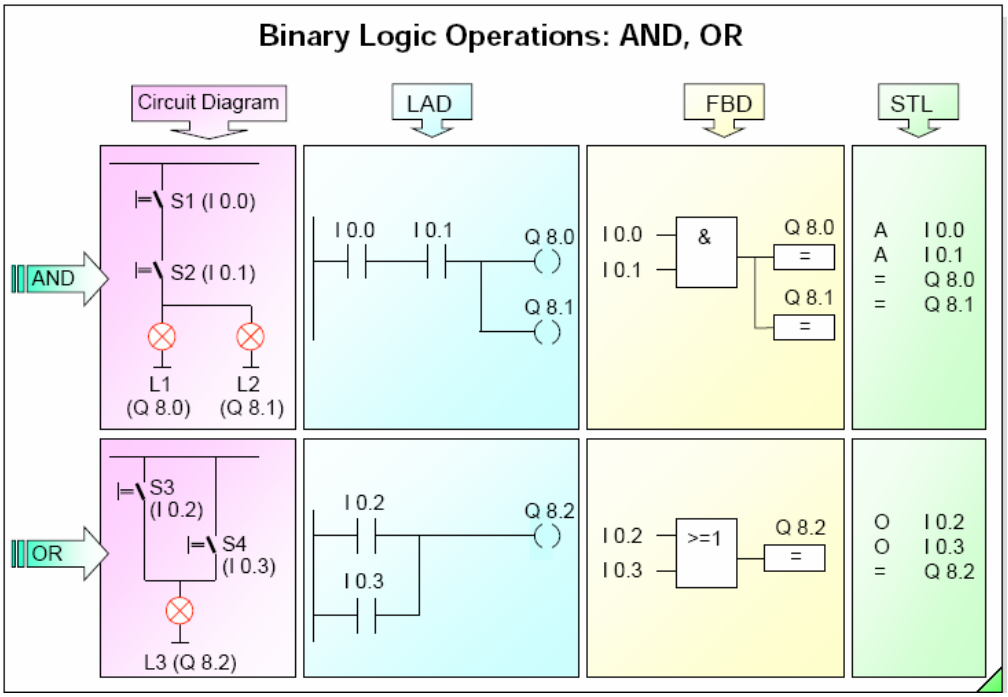
Type and Description	Size in Bits	Format Options	Range and Number Notation (lowest to highest value)_	Example
BOOL(Bit)	1.	Boolean text	TRUE/FALSE	TRUE
BYTE (Byte)	8	Hexadecimal number	B#16#0 to B#16#FF	L B#16#10 L byte#16#10
WORD (Word)	16	Binary number	2. 0 to	L 2#0001_0000_0000_0000
		Hexadecimal number	2#1111_1111_1111_1111 W#16#0 to W#16#FFFF	L W#16#1000 L word#16#1000
		BCD		L C#998
		Decimal number unsigned	C#0 to C#999 B#(0.0) to B#(255.255)	L B#(10,20) L byte#(10,20)
DWORD (Double word)	32	Binary number	2#0 to 2#1111_1111_1111_1111 1111_1111_1111_1111	2#1000_0001_0001_1000_ 1011_1011_0111_1111
		Hexadecimal number	DW#16#0000_0000 to DW#16#FFFF_FFFF	L DW#16#00A2_1234 L dword#16#00A2_1234
		Decimal number unsigned	B#(0,0,0,0) to B#(255,255,255,255)	L B#(1, 14, 100, 120) L byte#(1,14,100,120)
INT (Integer)	16	Decimal number signed	-32768 to 32767	L 1
DINT (Integer, 32 bits)	32	Decimal number signed	L#-2147483648 to L#2147483647	L L#1
REAL (Floating-point number)	32	IEEE Floating-point number	Upper limit: ±3.402823e+38 Lower limit: ±1.175 495e-38	L 1.234567e+13
S5TIME (SIMATIC time)	16	S7 time in steps of 10 ms (default)	S5T#0H_0M_0S_10MS to S5T#2H_46M_30S_0MS and S5T#0H_0M_0S_0MS	L S5T#0H_1M_0S_0MS L S5TIME#0H_1H_1M_0S_0MS
TIME (IEC time)	32	IEC time in steps of 1 ms, integer signed	-T#24D_20H_31M_23S_648MS to T#24D_20H_31M_23S_647MS	L T#0D_1H_1M_0S_0MS L TIME#0D_1H_1M_0S_0MS
DATE (IEC date)	16	IEC date in steps of 1 day	D#1990-1-1 to D#2168-12-31	L D#1996-3-15 L DATE#1996-3-15
TIME_OF_DAY (Time)	32	Time in steps of 1 ms	TOD#0:0:0.0 to TOD#23:59:59.999	L TOD#1:10:3.3 L TIME_OF_DAY#1:10:3.3
CHAR (Character)	8	ASCII characters	'A','B' etc.	L 'E'

في الجدول اعلاه يبين قيم البيانات التي سيتم استعمالها في كتابة البرامج داخل (Step7)

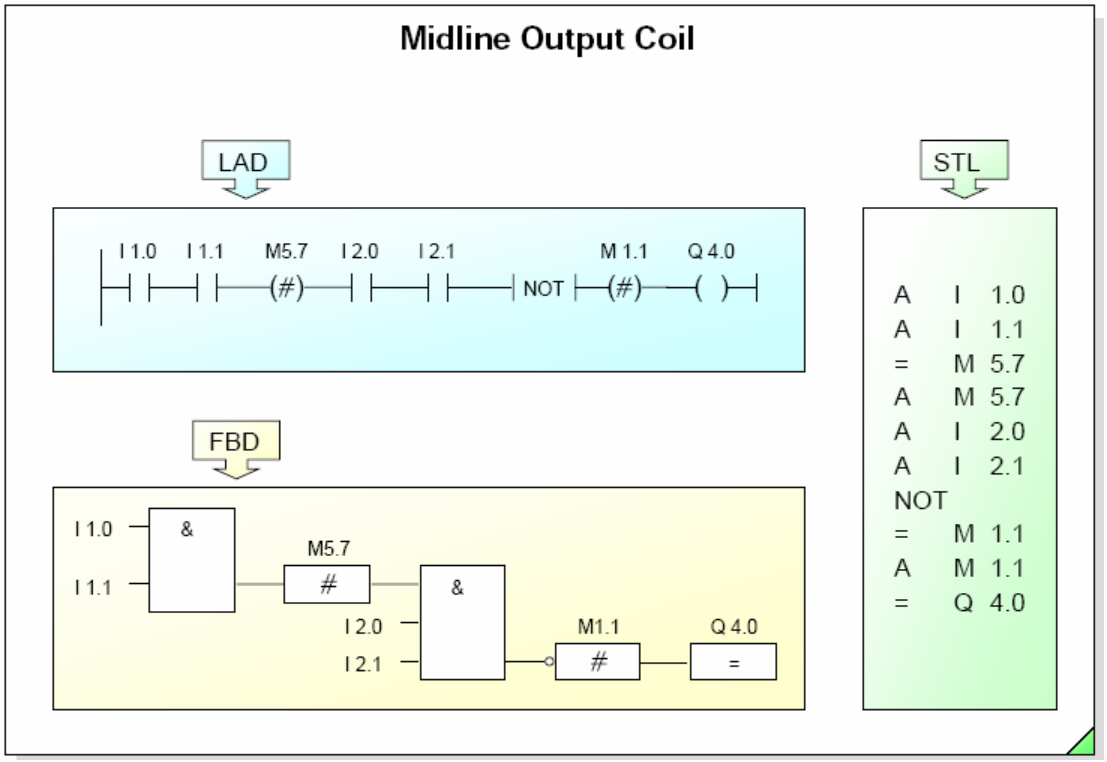
وايضا الجدول ادناه يبين التحويل بين الانظمة الرقمية

	Decimal		Binary					BCD								Hexadecimal	
	$*10^1$ =10	$*10^0$ =1	$*2^4$ =16	$*2^3$ =8	$*2^2$ =4	$*2^1$ =2	$*2^0$ =1	Tens Tetrad				Ones Tetrad				$*16^1$ = 16	$*16^0$ = 1
								8	4	2	1	8	4	2	1		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	0	1
2	0	2	0	0	0	1	0	0	0	0	0	0	0	1	0	0	2
3	0	3	0	0	0	1	1	0	0	0	0	0	0	1	1	0	3
4	0	4	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
5	0	5	0	0	1	0	1	0	0	0	0	0	1	0	1	0	5
6	0	6	0	0	1	1	0	0	0	0	0	0	1	1	0	0	6
7	0	7	0	0	1	1	1	0	0	0	0	0	1	1	1	0	7
8	0	8	0	1	0	0	0	0	0	0	0	1	0	0	0	0	8
9	0	9	0	1	0	0	1	0	0	0	0	1	0	0	1	0	9
10	1	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0	A
11	1	1	0	1	0	1	1	0	0	0	1	0	0	0	1	0	B
12	1	2	0	1	1	0	0	0	0	0	1	0	0	1	0	0	C
13	1	3	0	1	1	0	1	0	0	0	1	0	0	1	1	0	D
14	1	4	0	1	1	1	0	0	0	0	1	0	1	0	0	0	E
15	1	5	0	1	1	1	1	0	0	0	1	0	1	0	1	0	F
16	1	6	1	0	0	0	0	0	0	0	1	0	1	1	0	1	0
17	1	7	1	0	0	0	1	0	0	0	1	0	1	1	1	1	1
18	1	8	1	0	0	1	0	0	0	0	1	1	0	0	0	1	2
19	1	9	1	0	0	1	1	0	0	0	1	1	0	0	1	1	3

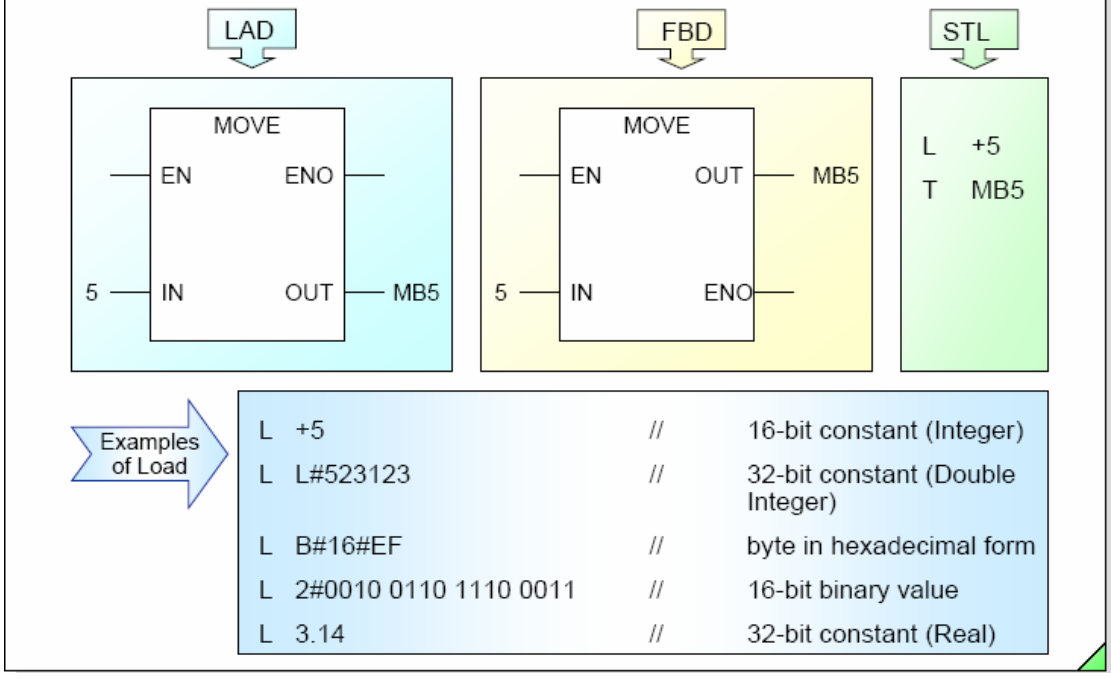
والان سنستعرض بعض الامثلة عن كيفية استخدام العناوين وانواع البيانات في البرامج المنطقية



Midline Output Coil

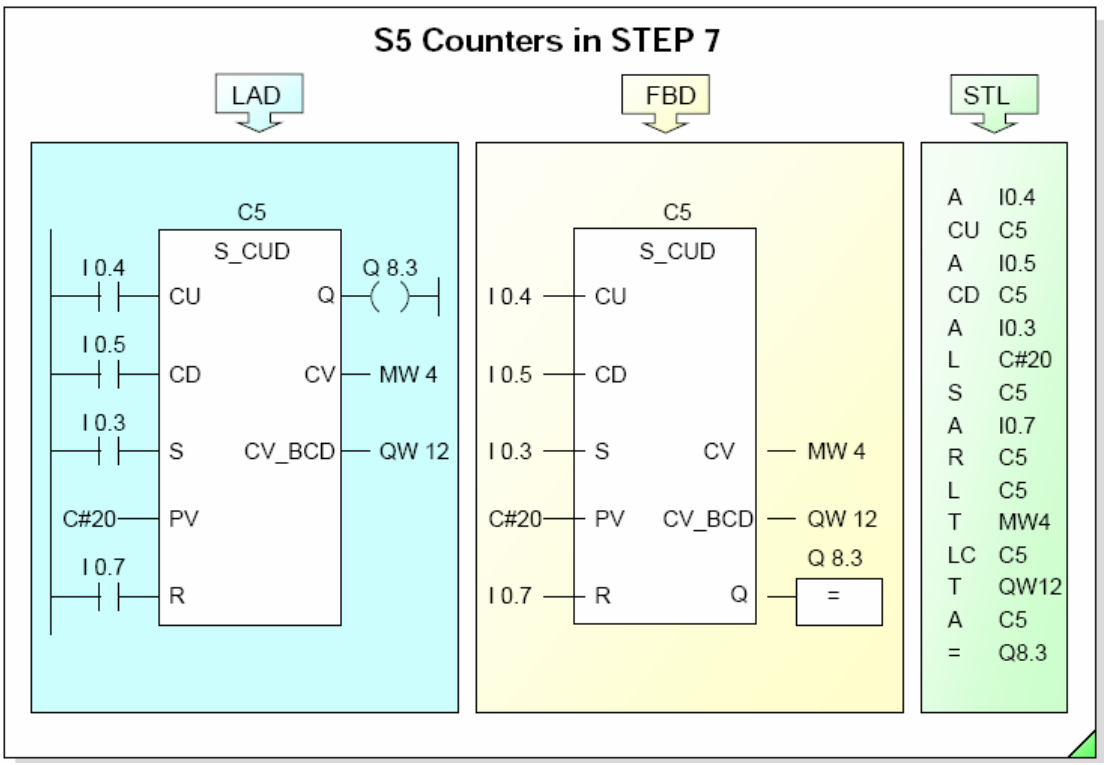


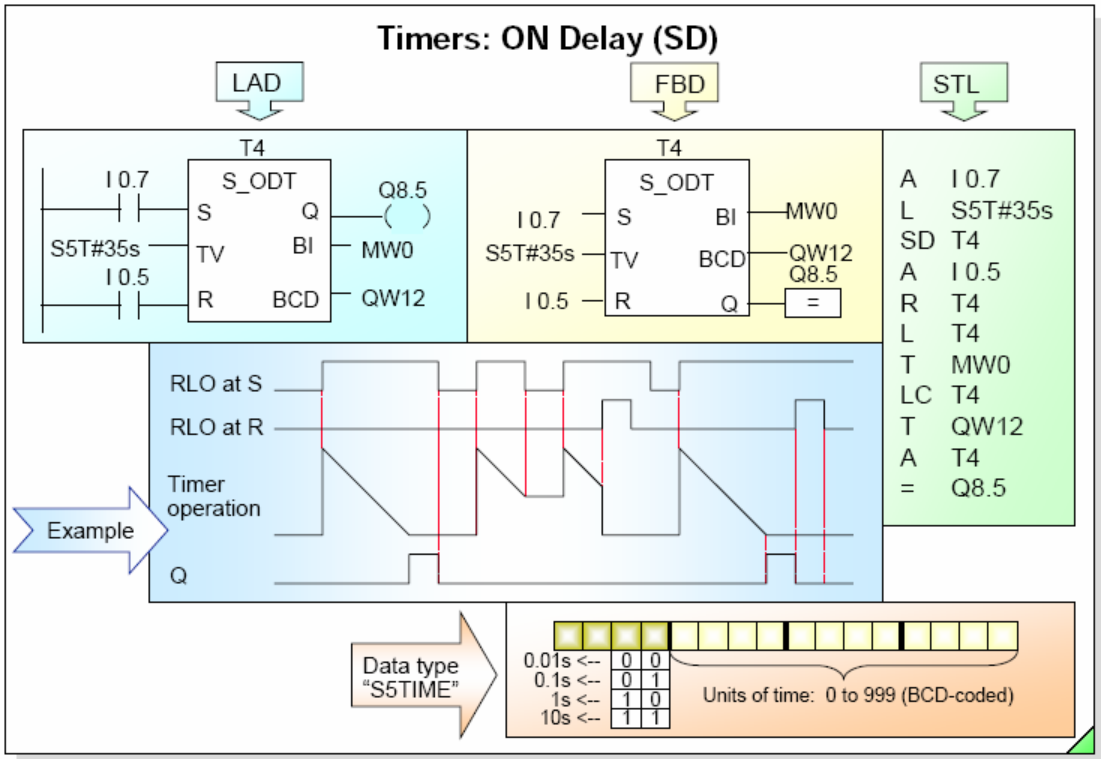
Loading and Transferring Data (1)



Examples of Load

```
L   +5                    //   16-bit constant (Integer)
L   L#523123              //   32-bit constant (Double Integer)
L   B#16#EF               //   byte in hexadecimal form
L   2#0010 0110 1110 0011 //   16-bit binary value
L   3.14                  //   32-bit constant (Real)
```


S5 Counters in STEP 7
LAD
FBD
STL
C5
S_CUD
I 0.4
CU
Q
Q 8.3
I 0.5
CD
CV
MW 4
I 0.3
S
CV_BCD
QW 12
C#20
PV
I 0.7
R
A I0.4
CU C5
A I0.5
CD C5
A I0.3
L C#20
S C5
A I0.7
R C5
L C5
T MW4
LC C5
T QW12
A C5
= Q8.3


Timers: ON Delay (SD)
LAD
FBD
STL
T4
S_ODT
I 0.7
S
Q
Q8.5
S5T#35s
TV
BI
MW0
I 0.5
R
BCD
QW12
A I 0.7
L S5T#35s
SD T4
A I 0.5
R T4
L T4
T MW0
LC T4
T QW12
A T4
= Q8.5
RLO at S
RLO at R
Timer operation
Q
Example
Data type "S5TIME"
0.01s <--
0.1s <--
1s <--
10s <--
Units of time: 0 to 999 (BCD-coded)

اليوم الخامس

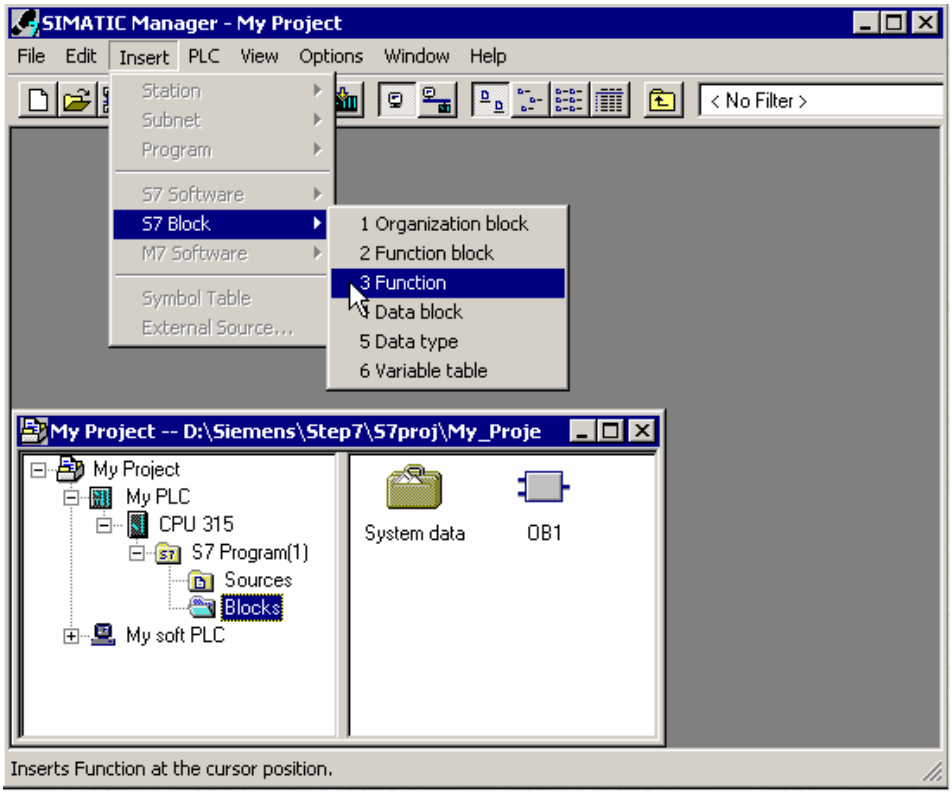
١ـ برمجة الدوال الفرعية
٢ـتكوين الوحدات الوظيفية وملفات البيانات
٣ـ برمجة ملفات البيانات المشتركة

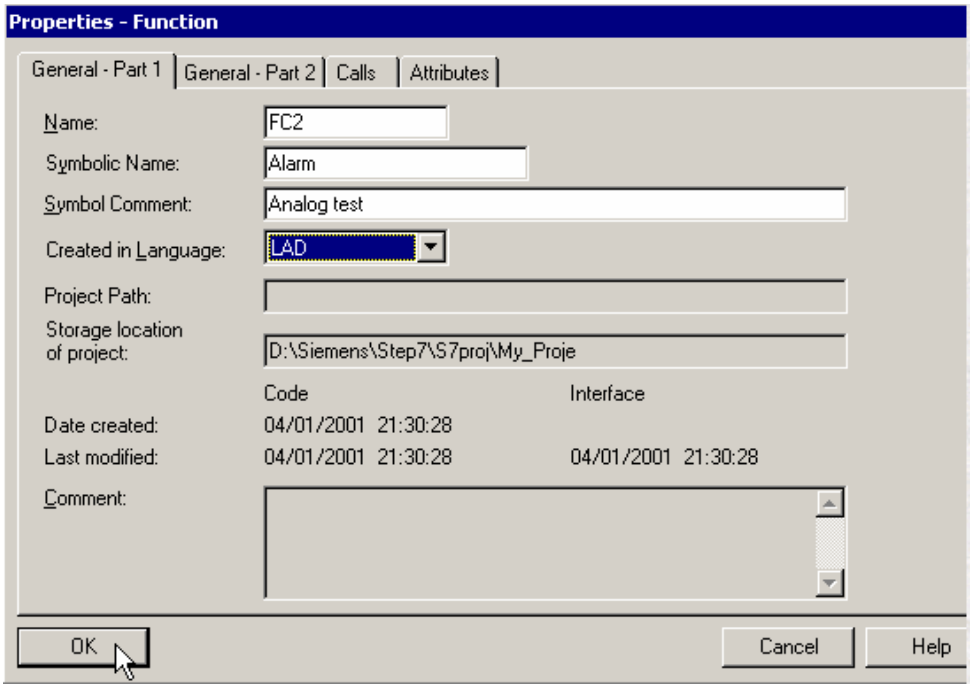
١ـ برمجة الدوال الفرعية

وهي عبارة عن ملفات يكتب بداخلها البرنامج المنطقي وتستخدم لسببين
١ـ يمكن استخدامها اكثر من مرة فلا يحتاج الى تكرار كتابة البرنامج
٢ـ لتجزئة البرامج الكبيرة الى اجزاء صغيرة
لا يمكن تنفيذ البرنامج داخل الدالة الفرعية ولكن يمكن تنفيذه من خلال استدعاءه داخل الوحدة التنظيمية (OB1)

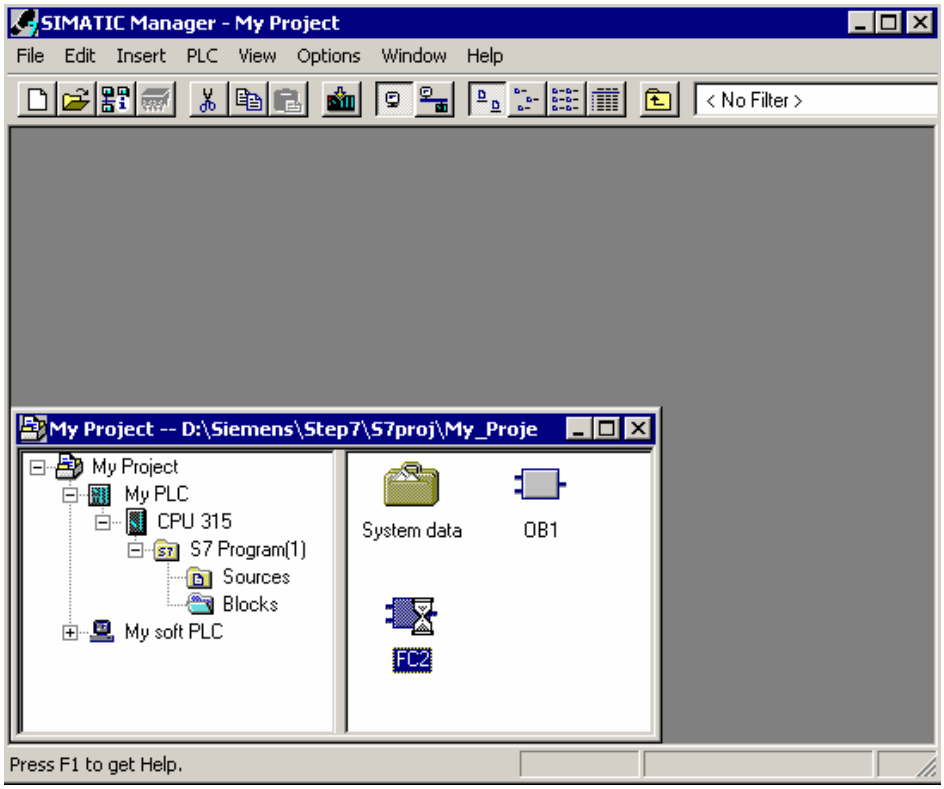
خطوات اضافة دالة فرعية:

١ـ نضغط على مجلد (Blocks) داخل نافذة المشروع
٢ـ من قائمة (Insert) نختار (S7Block) ثم (Function)

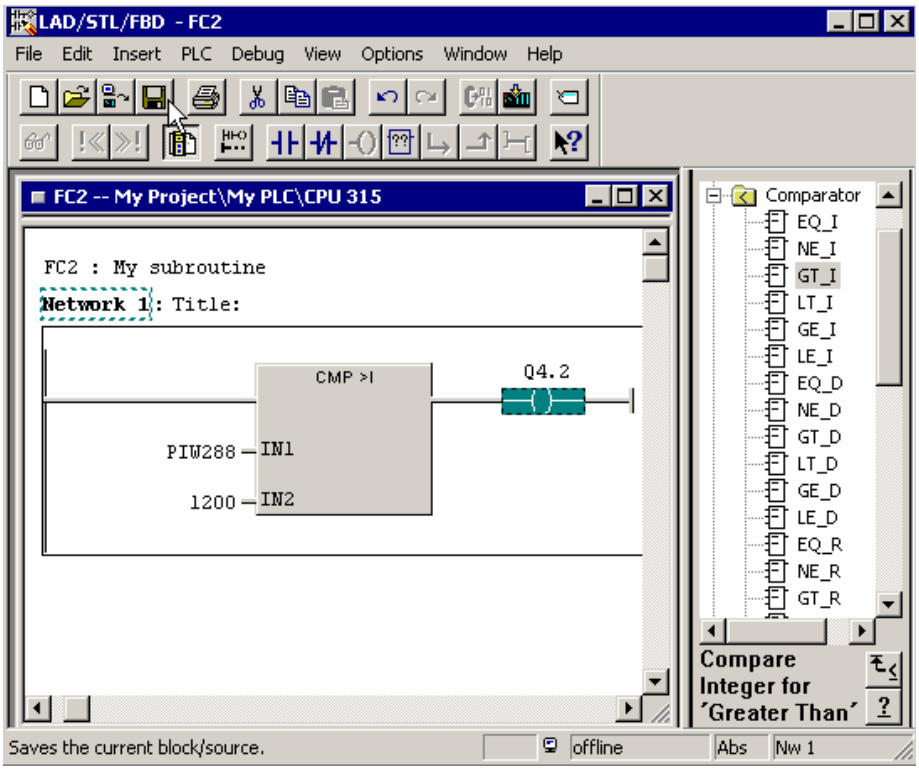




٣ـ ستظهر النافذة التالية نقوم بكتابة اسم وتعليق للدالة الفرعية ثم نضغط (OK) سيتكون الملف (FC2)

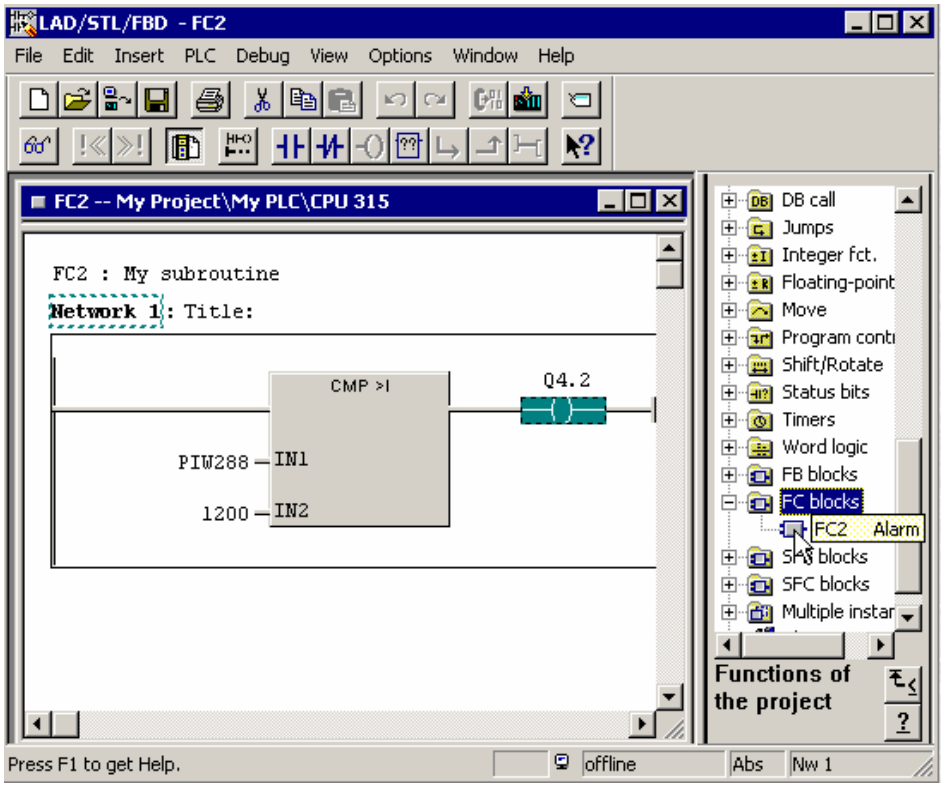


٥- ننقر نقرا مزدوجا على (FC2) ستظهر نافذة شبيهة بنافذة (OB1) نقوم باضافة العناصر والعناوين كما تعلمنا سابقا



٦-نقوم بحفظ التغييرات

٧-سنلاحظ انه تم اضافة الدالة الفرعية في نافذة العناصر حيث يمكن سحبها كعنصر

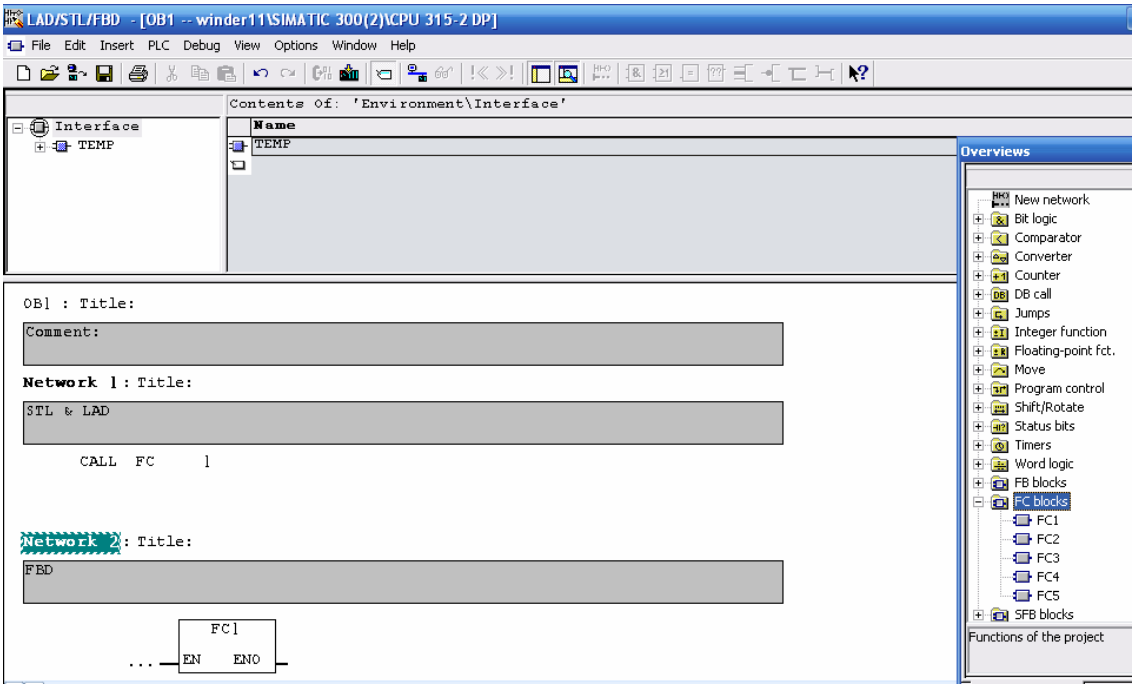


٨- يمكن استدعاء الدالة الفرعية من خلال نافذة (OB1) بالطرق التالية:

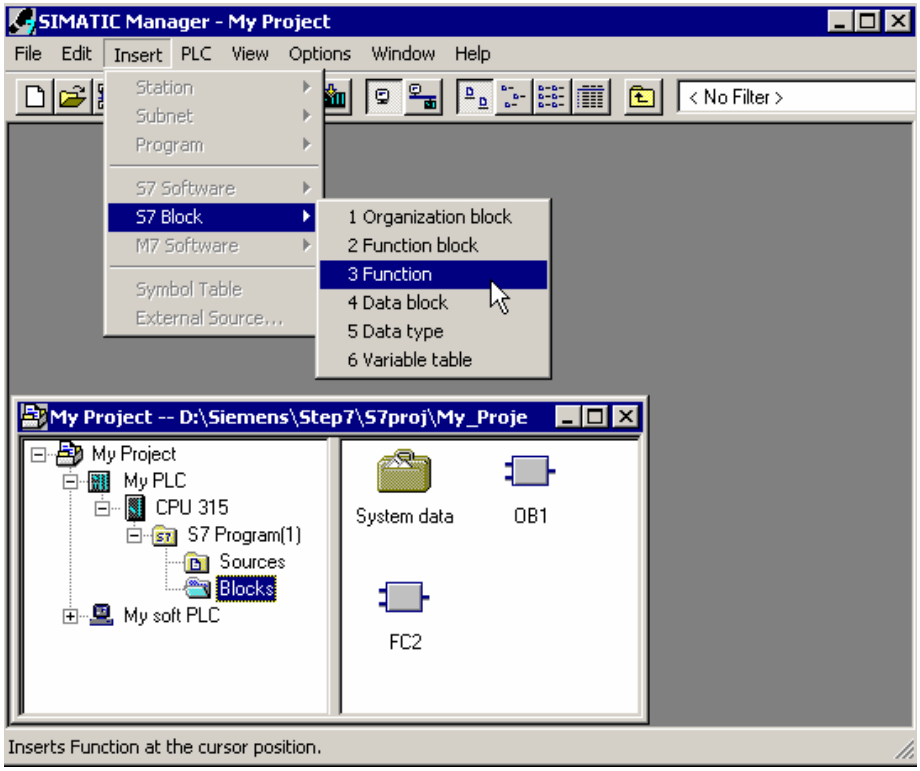
١- في لغة (STL) نكتب (Call Function name) مثلا

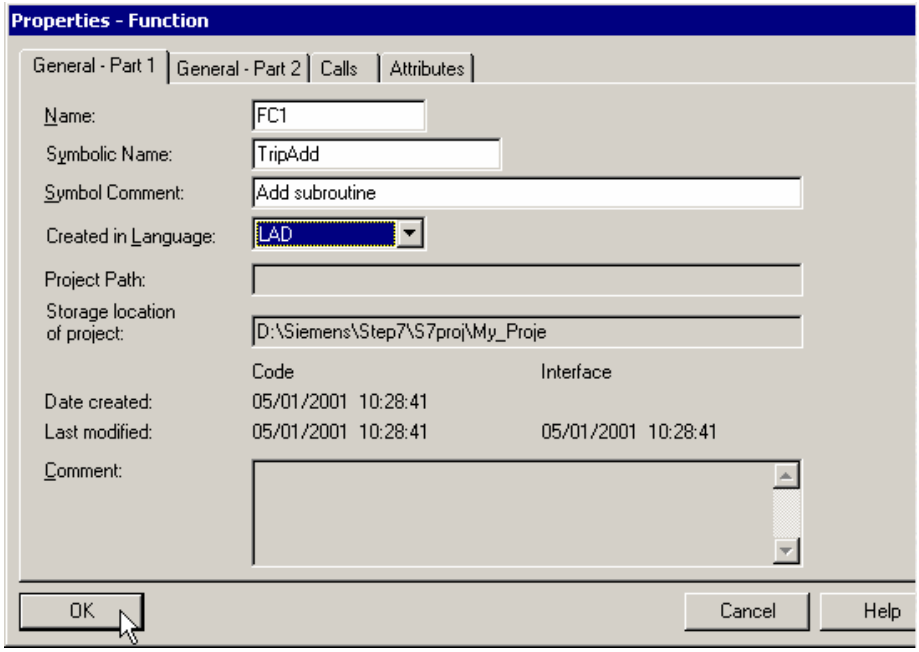
(Call FC 1) او (Call FC 2) الخ

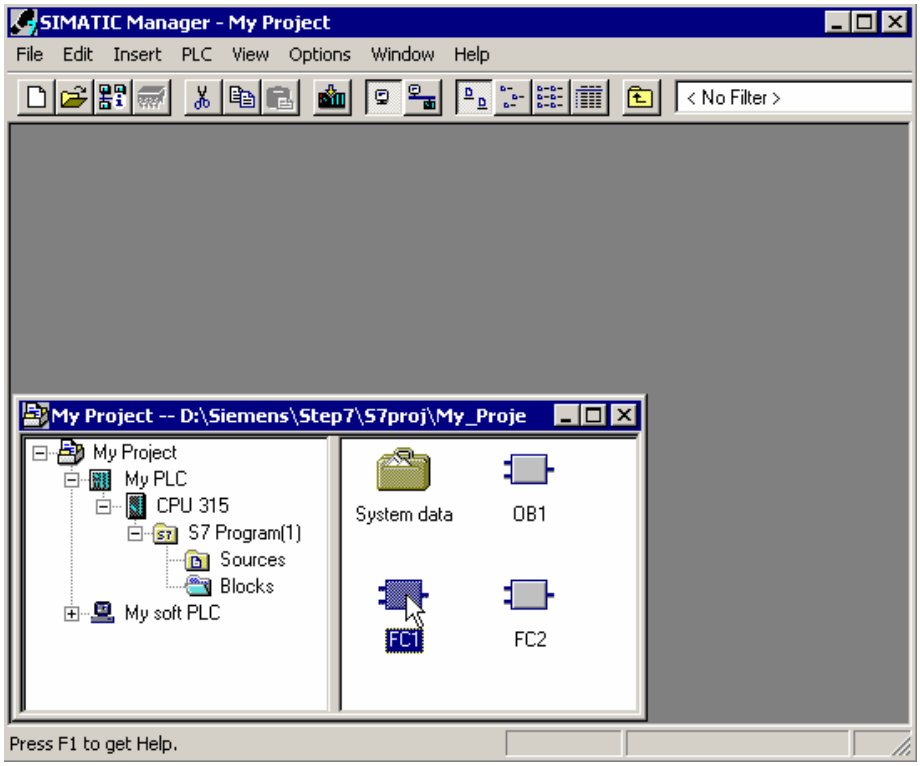
٢- في لغة (FBD) نقوم بسحبها كاحد العناصر

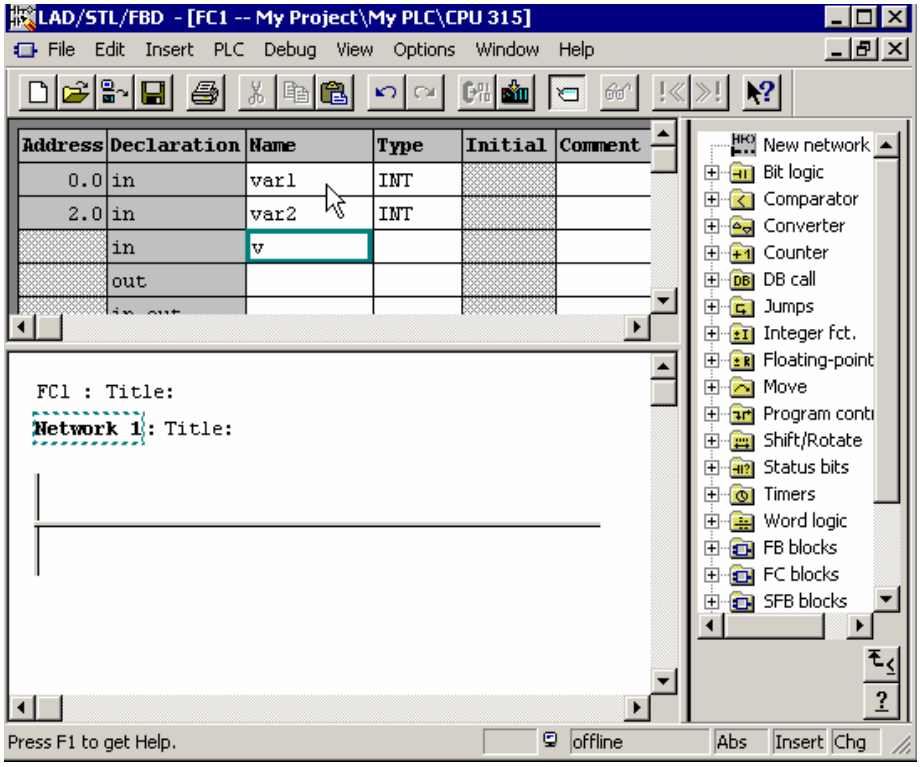


والآن سنقوم باضافة دالة فرعية آخرى بنسق آخر

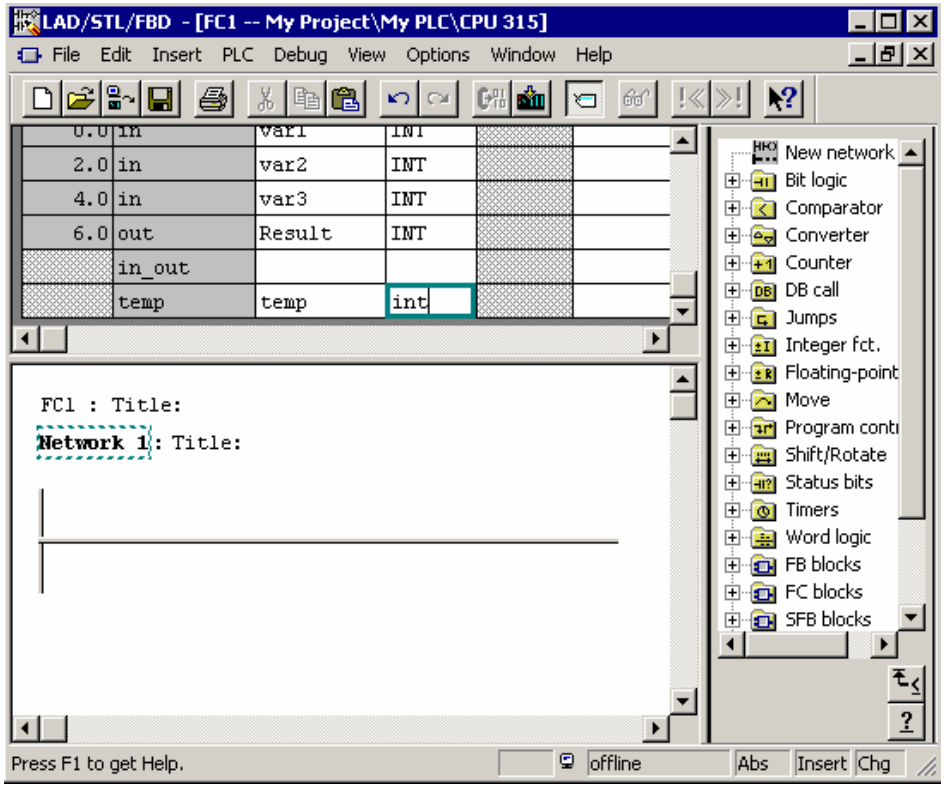



Properties - Function
General - Part 1
General - Part 2
Calls
Attributes
Name:
FC1
Symbolic Name:
TripAdd
Symbol Comment:
Add subroutine
Created in Language:
LAD
Project Path:
Storage location of project:
D:\Siemens\Step7\S7proj\My_Proje
Code
Interface
Date created:
05/01/2001 10:28:41
Last modified:
05/01/2001 10:28:41
05/01/2001 10:28:41
Comment:
OK
Cancel
Help

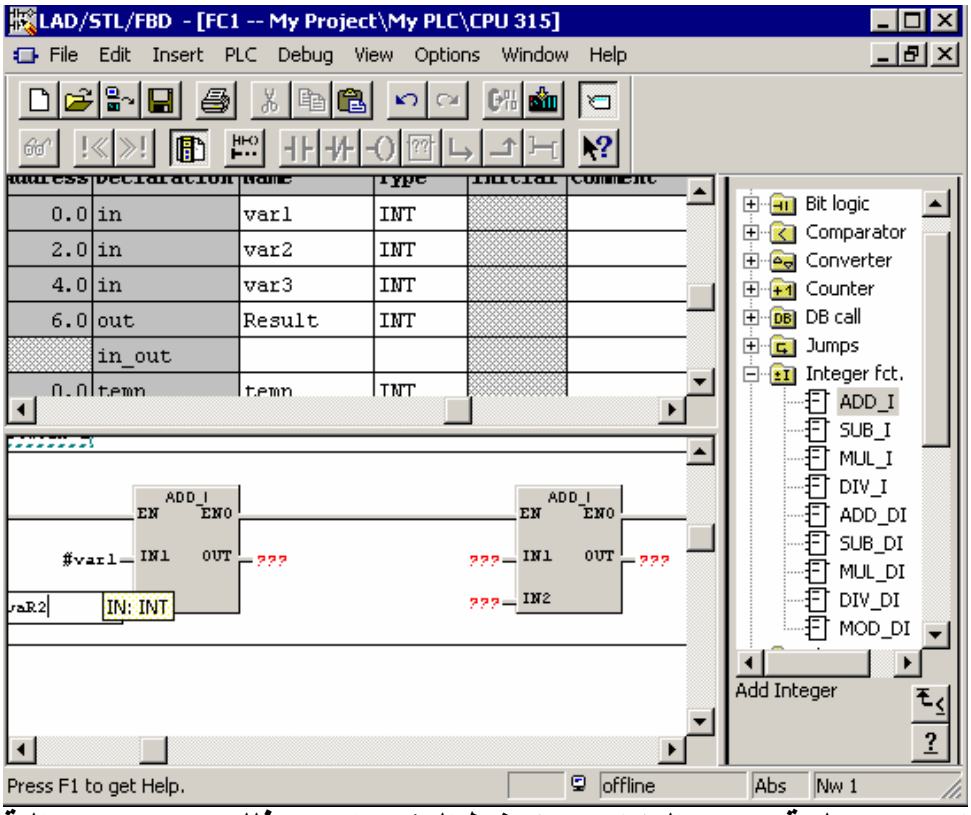

SIMATIC Manager - My Project
File
Edit
Insert
PLC
View
Options
Window
Help
< No Filter >
My Project -- D:\Siemens\Step7\S7proj\My_Proje
My Project
My PLC
CPU 315
S7 Program(1)
Sources
Blocks
My soft PLC
System data
OB1
FC1
FC2
Press F1 to get Help.



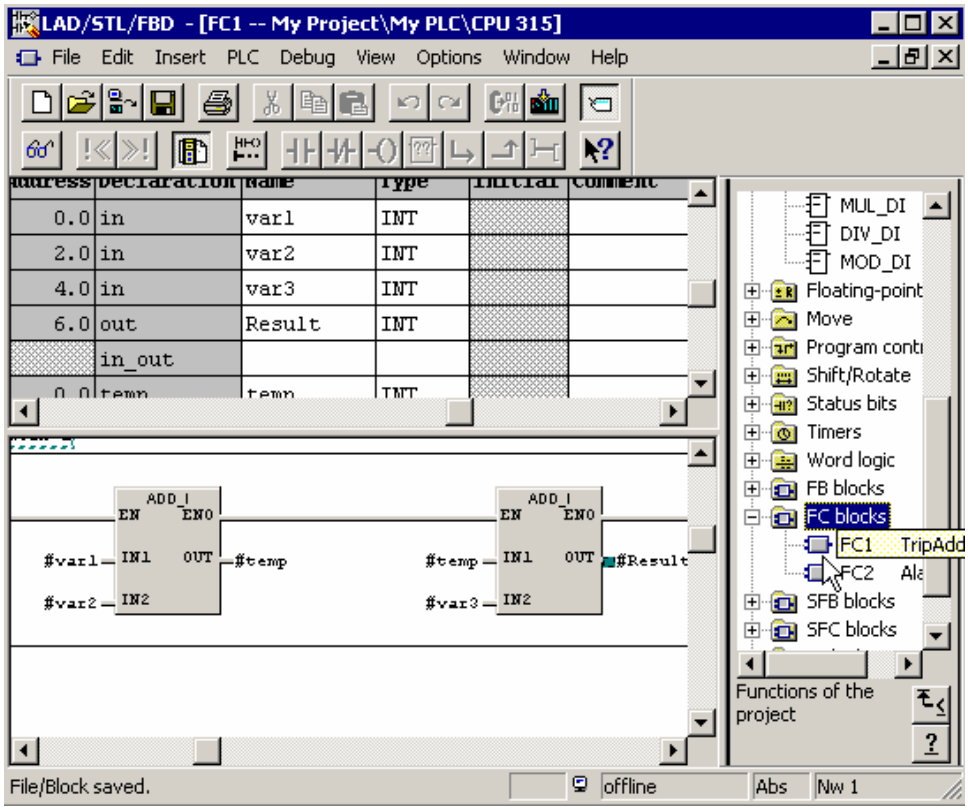
نقوم بملأ الجدول في النافذة اعلاه وتضع متغيرات بدل الرموز واعتبارها عناوين ثانوية



بعد اكمال الجدول نقوم باضافة العناصر ووضع العناويين الثانوية بدل الرموز



بعد الانتهاء من عملية وضع العناويين نحفظ التغييرات وبذلك تم تكوين دالة فرعية من دون استخدام الرموز مما يساعد في تقليل عدد الرموز المعرفة وهذه العناويين الثانوية لا تتم اضافتها في جدول الرموز



وعند الاستدعاء داخل (OB1) ستظهر مداخل الدالة الفرعية باللون الاحمر اي تحتاج الى قيم سواء كانت رقمية او قيم مأخوذة من العناوين

```
OB1 : Title:
Comment:

Network 1: Title:
Comment:

      CALL  FC     1
       Var1  :=
       Var2  :=
       Var3  :=
       Result:=
```

نقوم بادخال العناوين او المتغيرات الرقمية

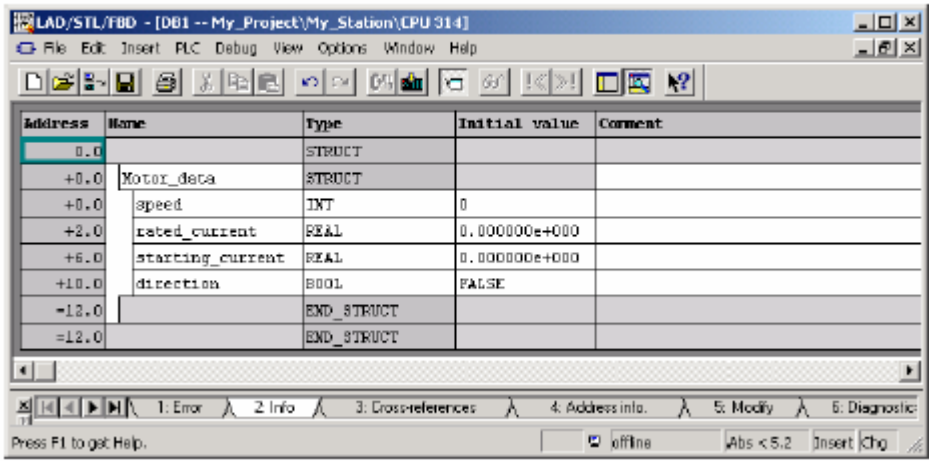
```
OB1 : Title:
Comment:

Network 1: Title:
Comment:

      CALL  FC     1
       Var1  :=IW0
       Var2  :=5
       Var3  :=3
       Result:=QW3
```

٢ـ تكوين الوحدات الوظيفية وملفات البيانات

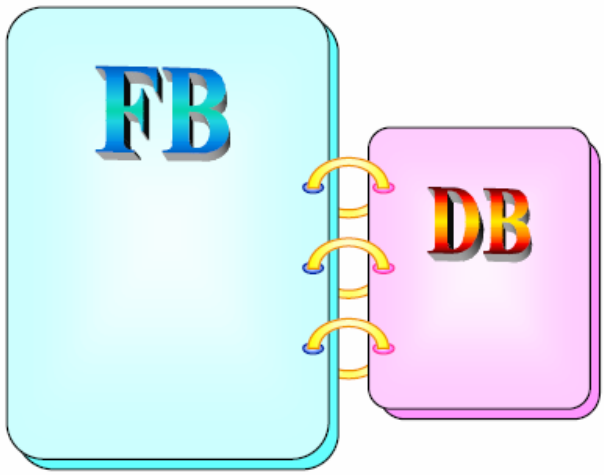
أـ الوحدات الوظيفية (Function Block) او (FB): وهي تشبه الدوال الفرعية التي تعلمنا برمجتها سابقا الى انها تحتاج الى ذاكرة لخزن المتغيرات التي بداخلها وهذه الذاكرة عبارة عن ملف بيانات (Data Block) او (DB) ولايمكن استدعاء (FB) داخل الوحدة التنظيمية (OB1) الا اذا تم تخصيص ملف بيانات (DB) لكل (FB)

بـ ملفات البيانات (DB): وهي عبارة عن قاعدة بيانات يتم خزن المتغيرات وقيمها بداخلها ويتم استدعاء قيم التغيرات من ملف البيانات عند الحاجة كما يمكن التعديل على المتغيرات اثناء العمل (Online) ويكون شكل ملف البيانات كالتالي:

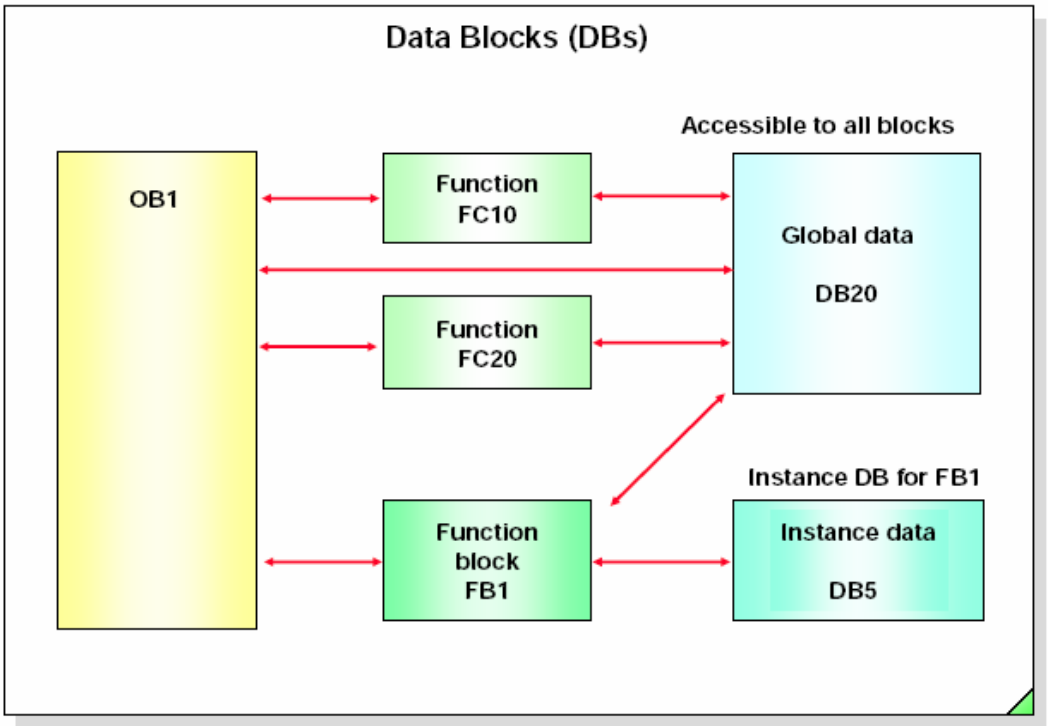
Display in the Program Editor (Data block DB 1):



تنقسم ملفات البيانات الى نوعين:
النوع الاول والذي يطلق عليه (Instance Data Block) والذي يكون مخصص فقط للتعامل مع (FB) ويمكن قراءة النتغيرات او تغييرها من خلال (FB) المتعلق بها فقط

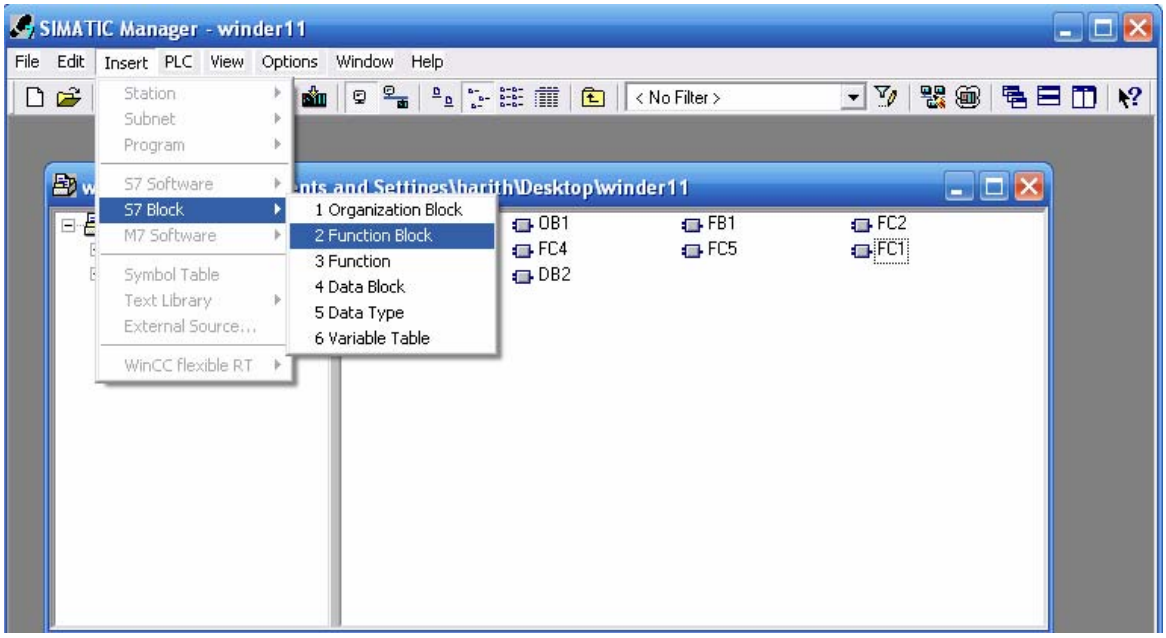


النوع الثاني من ملفات البيانات والتي تسمى (Shared Data Block) او (Global Data Block): وهي عبارة عن بيانات يمكن الاستفادة منها في اي مكان من البرنامج

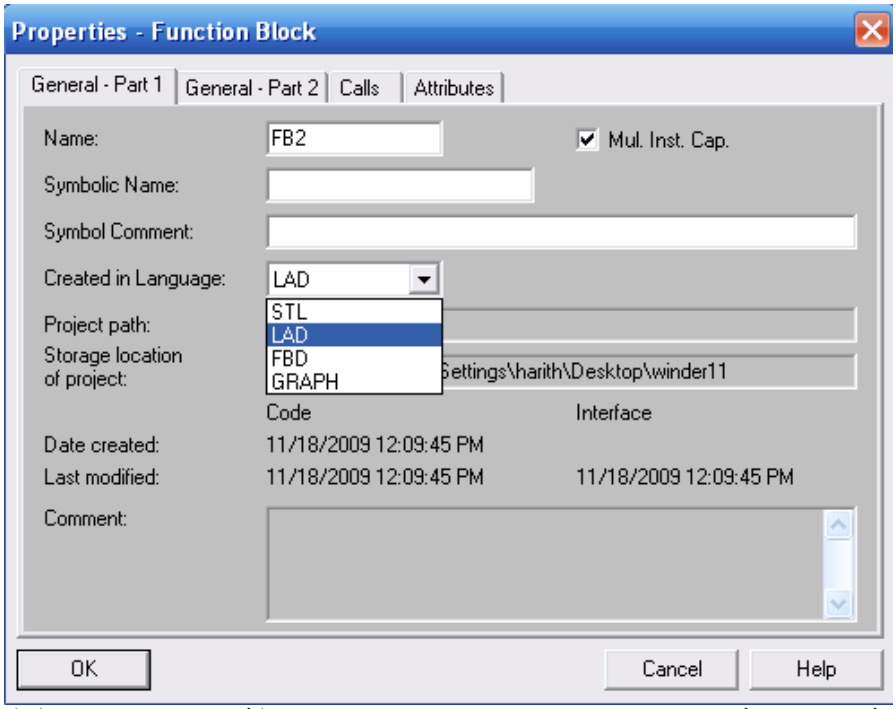


خطوات برمجة (FB):

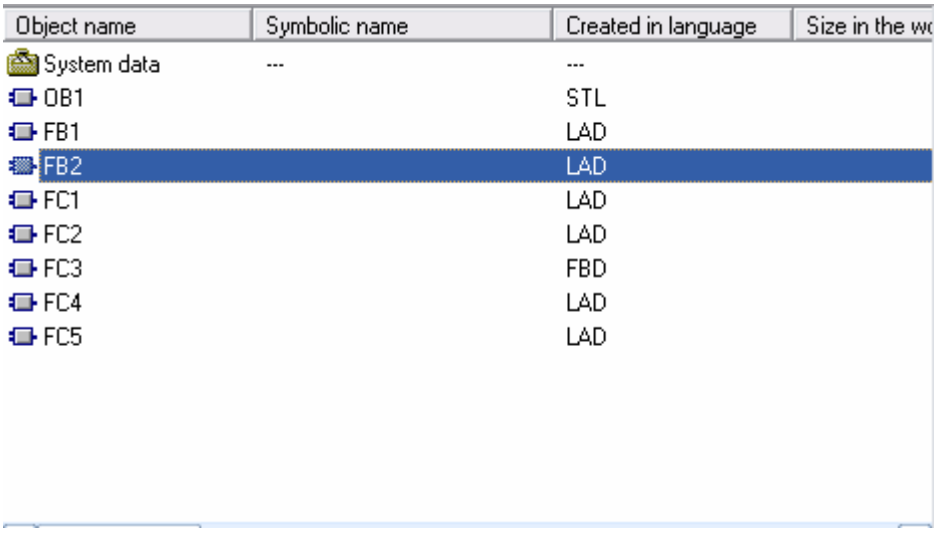
١ـ نقوم باضافة (FB) الى نافذة (Block)



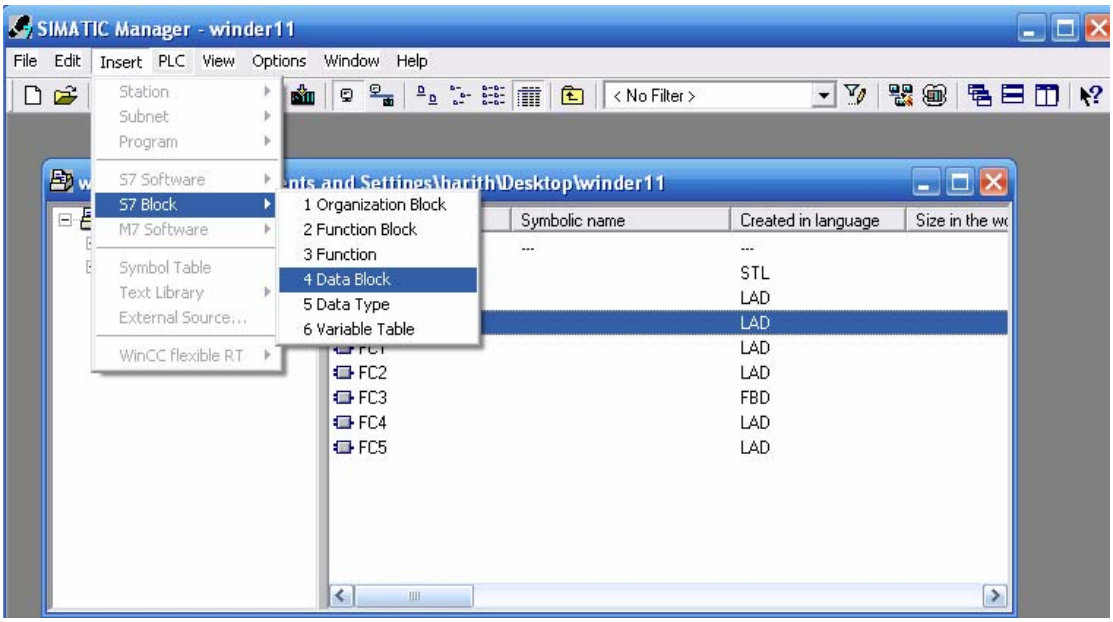
٢ـ ستظهر النافذة التالية:



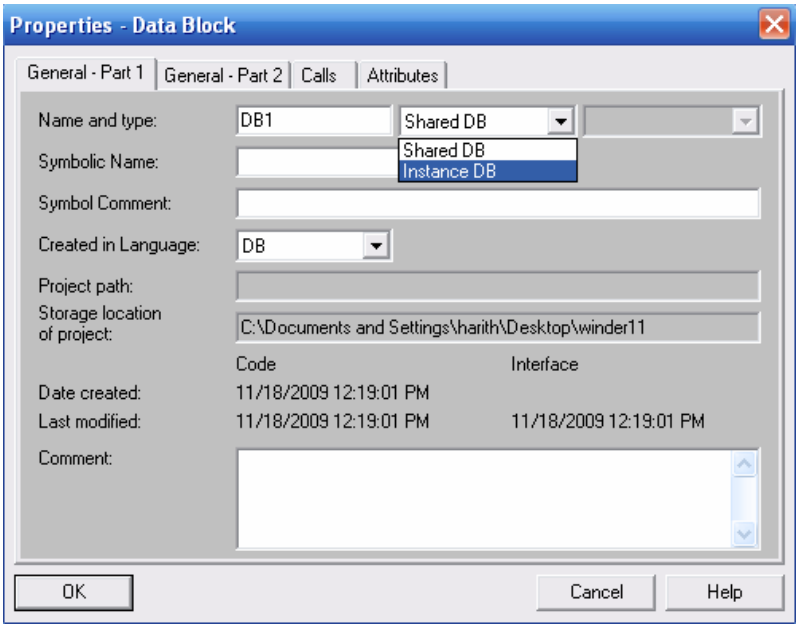
٣ـ نختار لغة البرمجة ونضغط على (OK) سيتم اضافة (FB) الى نافذة ال (Block)



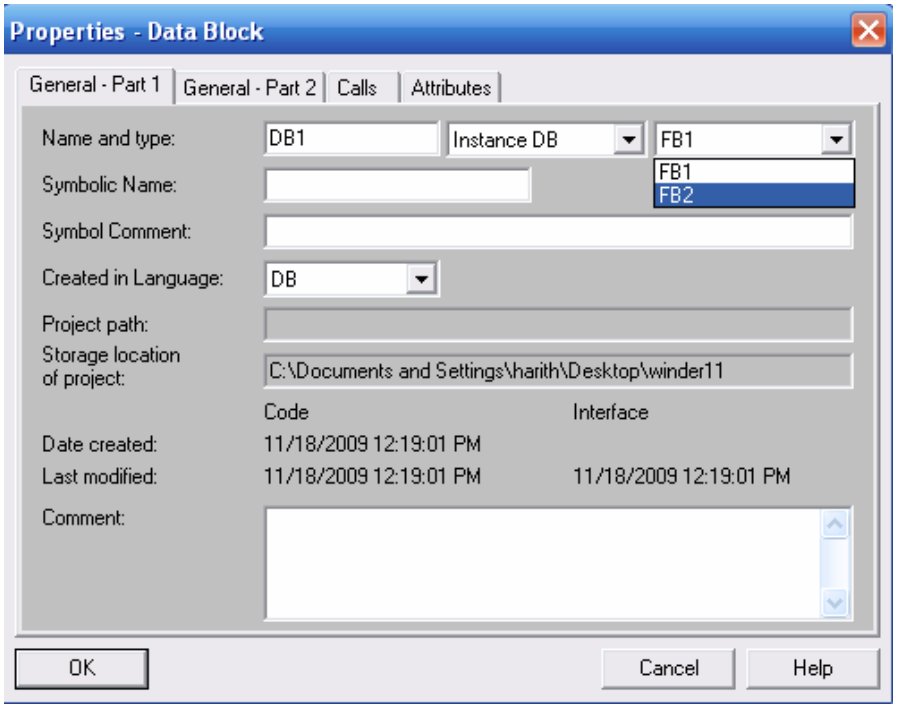
٤ـ نقوم باضافة ملف بيانات (DB)



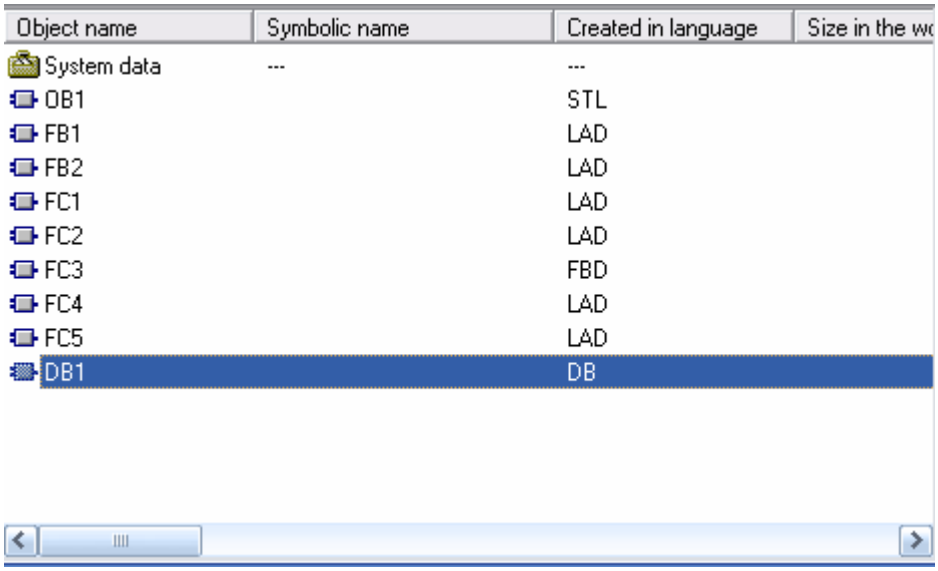
٥ـ ستظهر النافذة التالية:



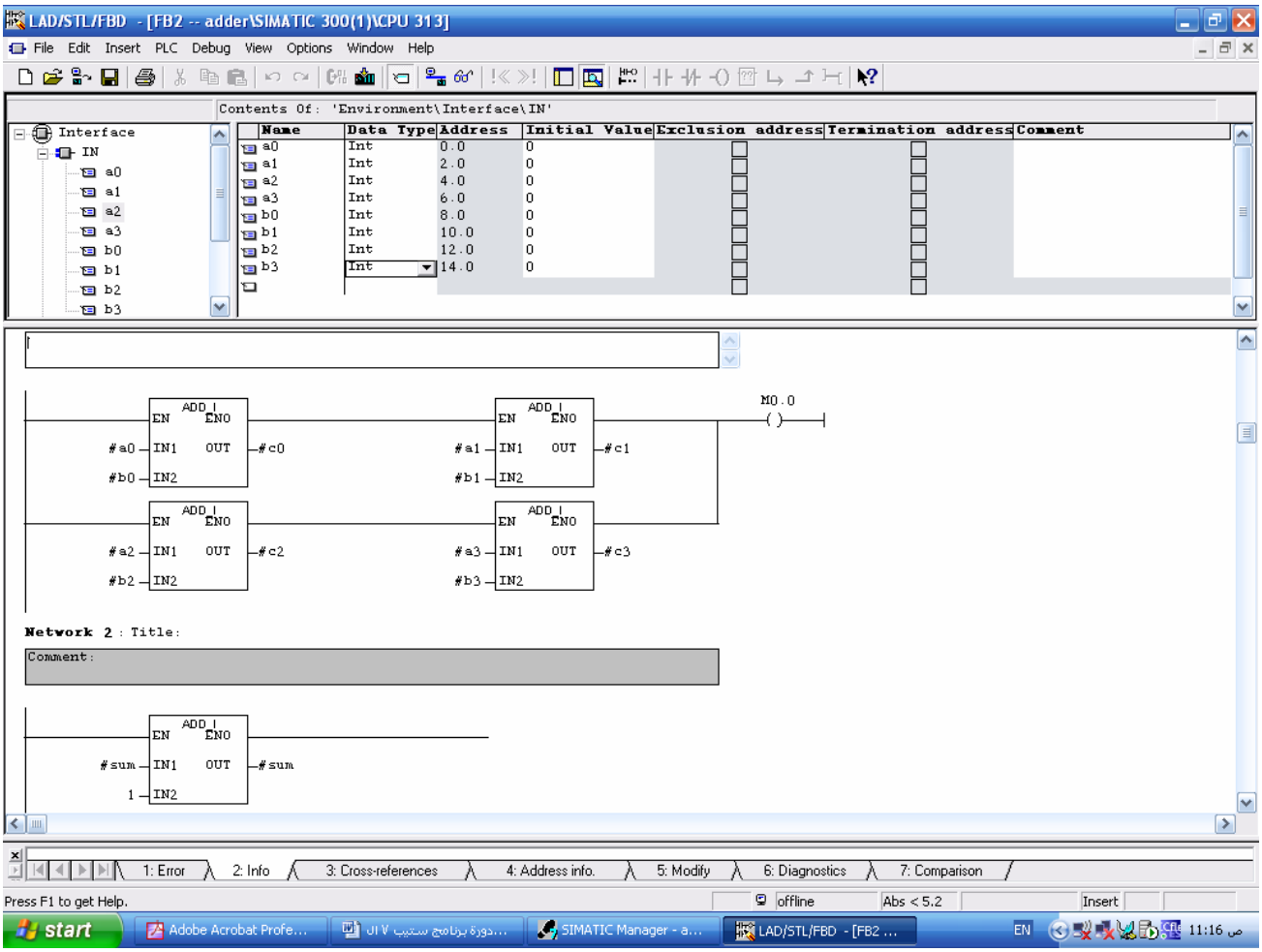
٦- نختار النوع (Instance DB) ثم نختار (FB) المرتبط معها ثم نضغط (OK)



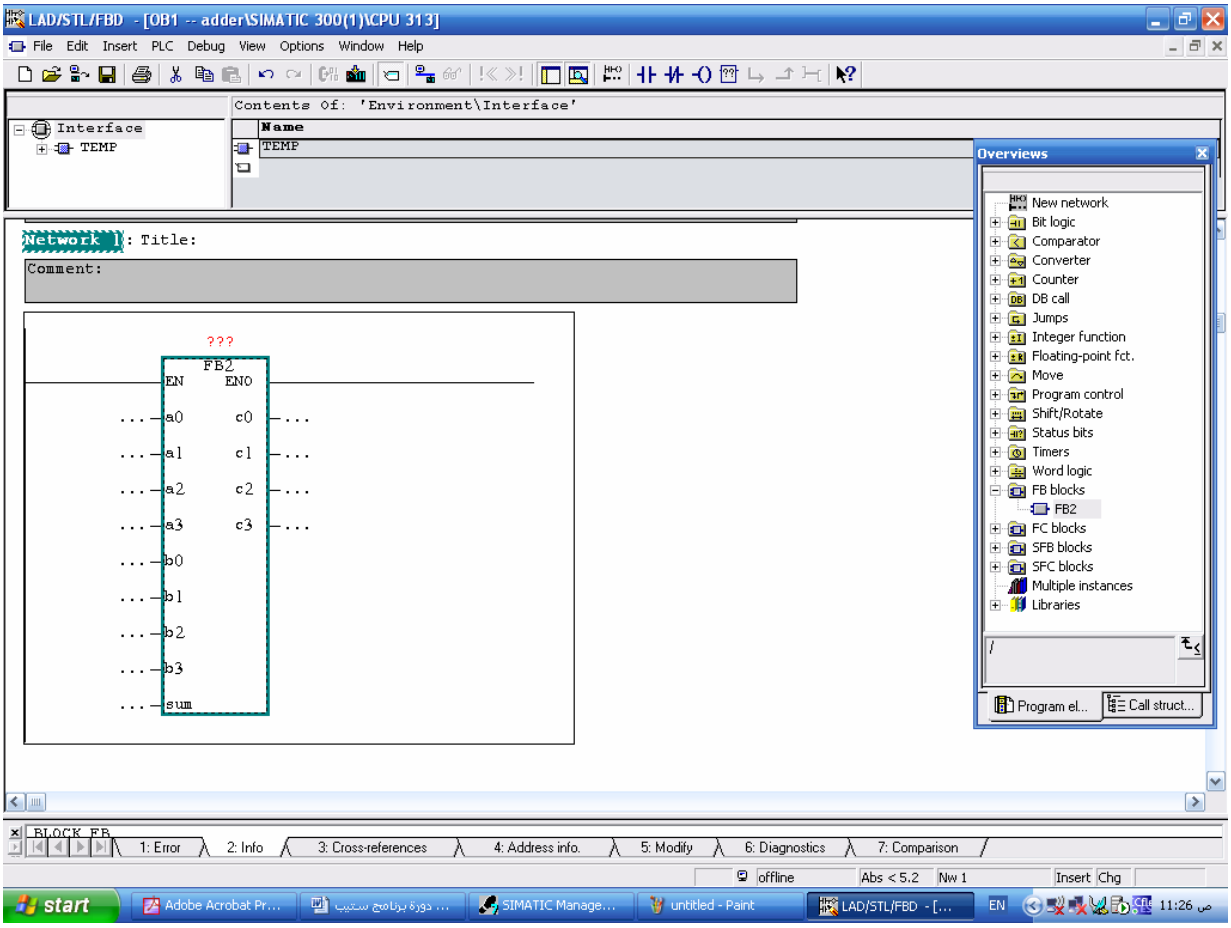
٧- سيتم اضافة ملف البيانات الى ناقذة (Block)

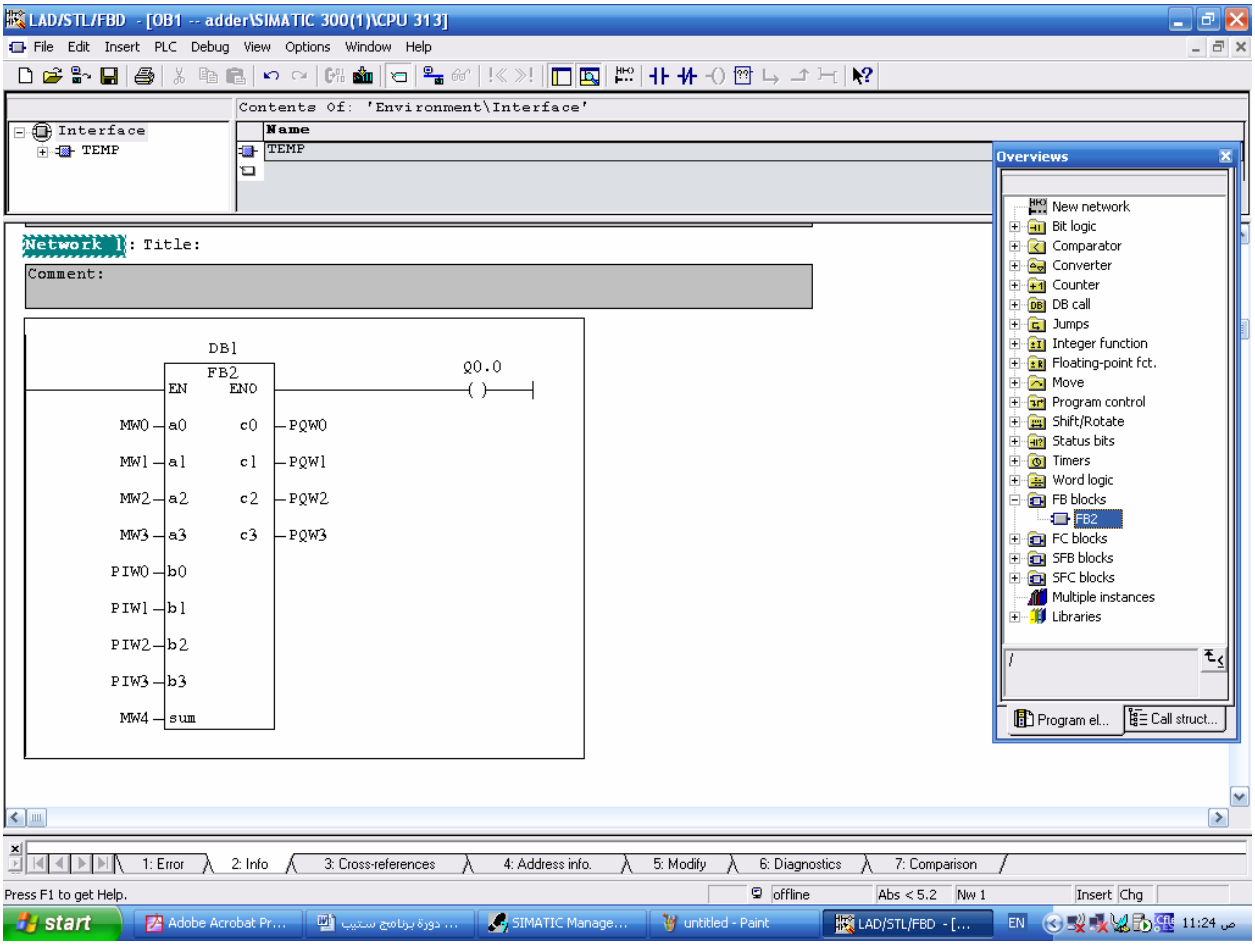


٩- ننقر نقرا مزدوجا على (FB2) ستظهر نافذة تشبه نافذة (FC) نقوم بكتابة البرنامج التالي بعد ان قمنا بتعريف المتغيرات في جدول التصاريح



١٠- لاستدعاء (FB) الذي قمنا بتكوينه اعلاه مع تعريف المتغيرات في جدول التصاريح نقوم بسحبه من نافذة العناصر داخل (OB1) او (FC) و يجب تعريف مداخل ومخارج (FB)





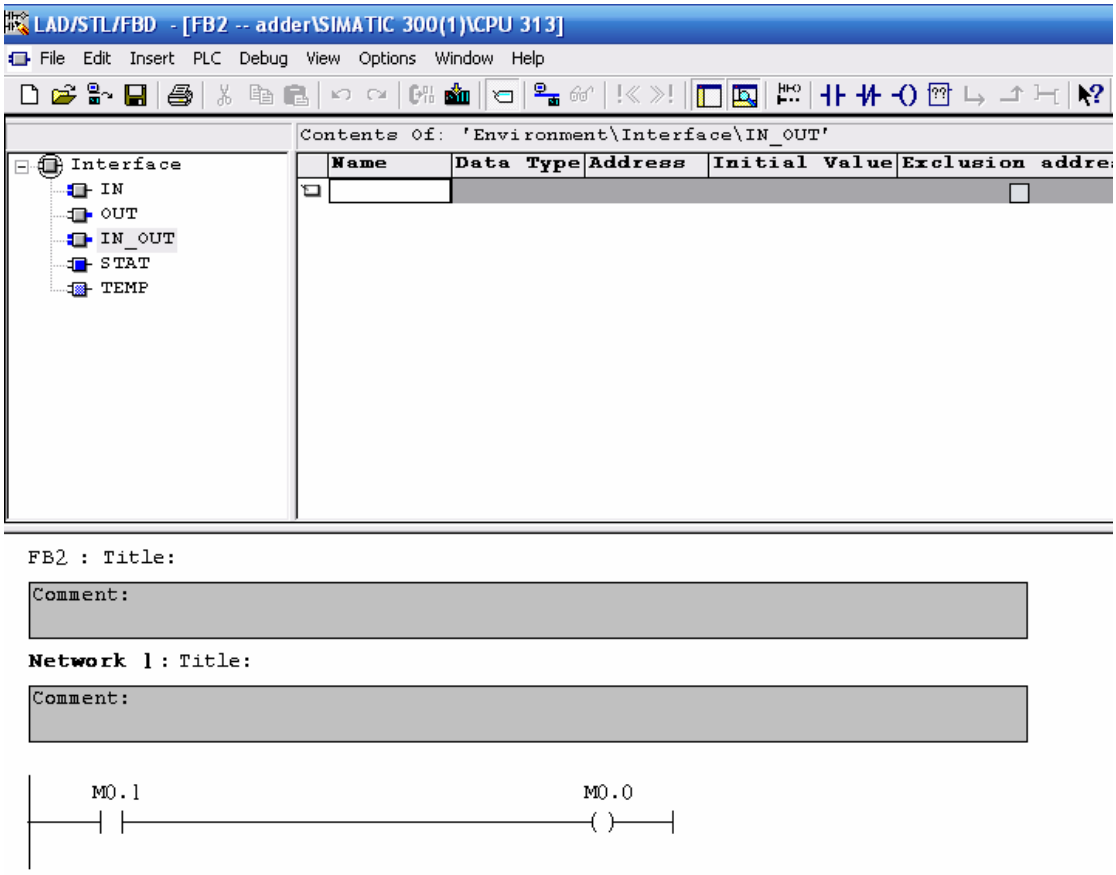
١١- بعد ان عرفنا متغيرات المداخل والمخارج ووضعنا عنوان لل (FB2) هو (DB1) اي ملف البيانات المرتبط به نفتح (DB1) سنلاحظ كالتالي:

DB Param - [DB1 -- adder\SIMATIC 300(1)\CPU 313]

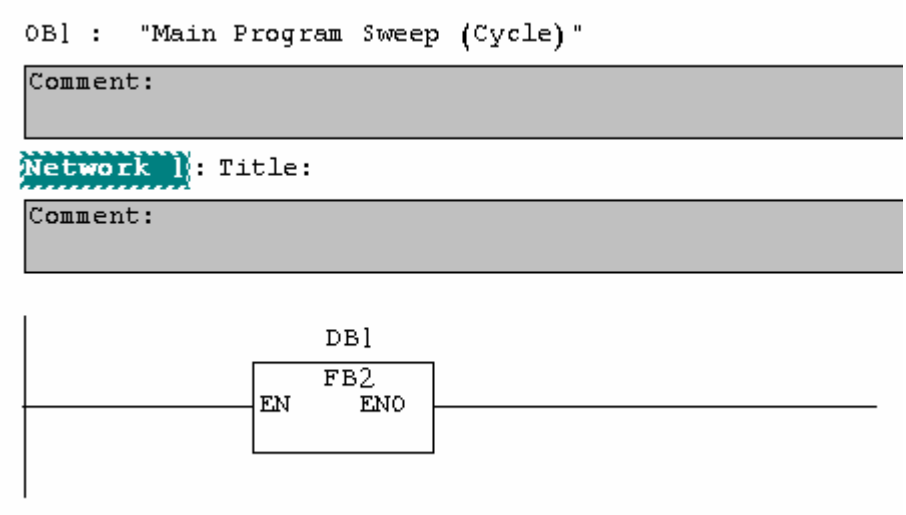
Data block Edit PLC Debug View Window Help

	Address	Declaration	Name	Type	Initial valu	Actual valu	Comment
1	0.0	in	a0	INT	0	0	
2	2.0	in	a1	INT	0	0	
3	4.0	in	a2	INT	0	0	
4	6.0	in	a3	INT	0	0	
5	8.0	in	b0	INT	0	0	
6	10.0	in	b1	INT	0	0	
7	12.0	in	b2	INT	0	0	
8	14.0	in	b3	INT	0	0	
9	16.0	out	c0	INT	0	0	
10	18.0	out	c1	INT	0	0	
11	20.0	out	c2	INT	0	0	
12	22.0	out	c3	INT	0	0	
13	24.0	in_out	sum	INT	0	0	

١٢- ويمكن برمجة (FB) بدون استخدام المتغيرات وبهذا لا يحتاج الى ملفات البيانات كالتالي:



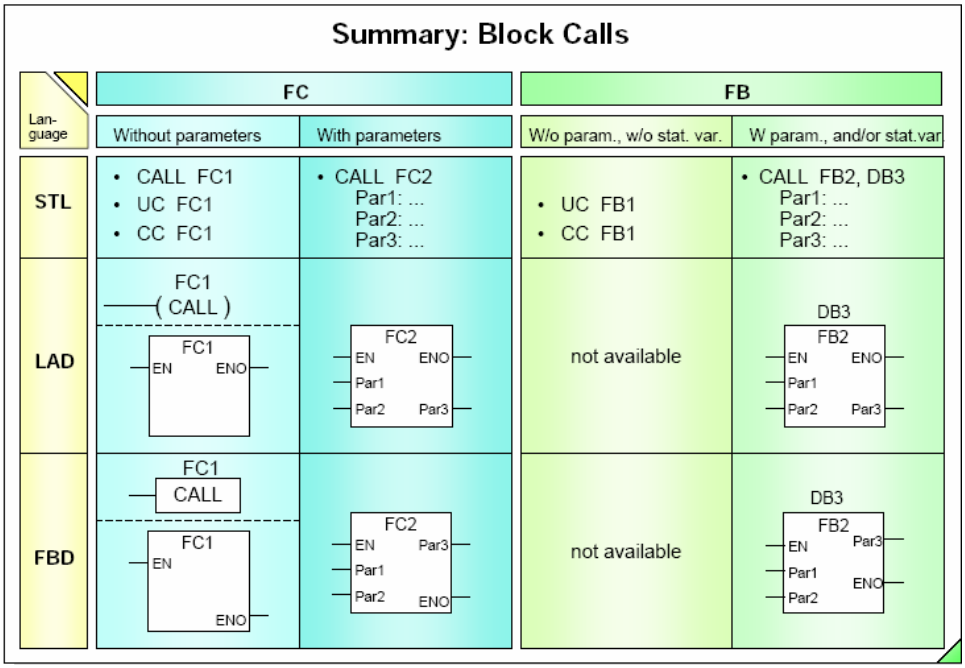
١٣- وللاستدعاء نقوم بسحبه من نافذة العناصر او استدعائه باستخدام الايعاز (Call) بلغة (STL)



```
OB1 :   "Main Program Sweep (Cycle)"
Comment:

Network 1: Title:
Comment:

      CALL  FB     2 , DB1
      NOP   0
```

١٤-وعند فتح ملف البيانات (DB1) سنجده كالتالي:

١٥- يوجد عدة اشكال للاستدعاء ولاداعي للتطرق اليها لان الذي ذكرناه يفي بالغرض



٣ـ برمجة ملفات البيانات المشتركة

وهي عبارة عن جداول تكتب بها المتغيرات وقيمها ويمكن الاستفادة من في اي مكان من البرنامج من دون الحاجةالى كتابة الارقام كل مرة

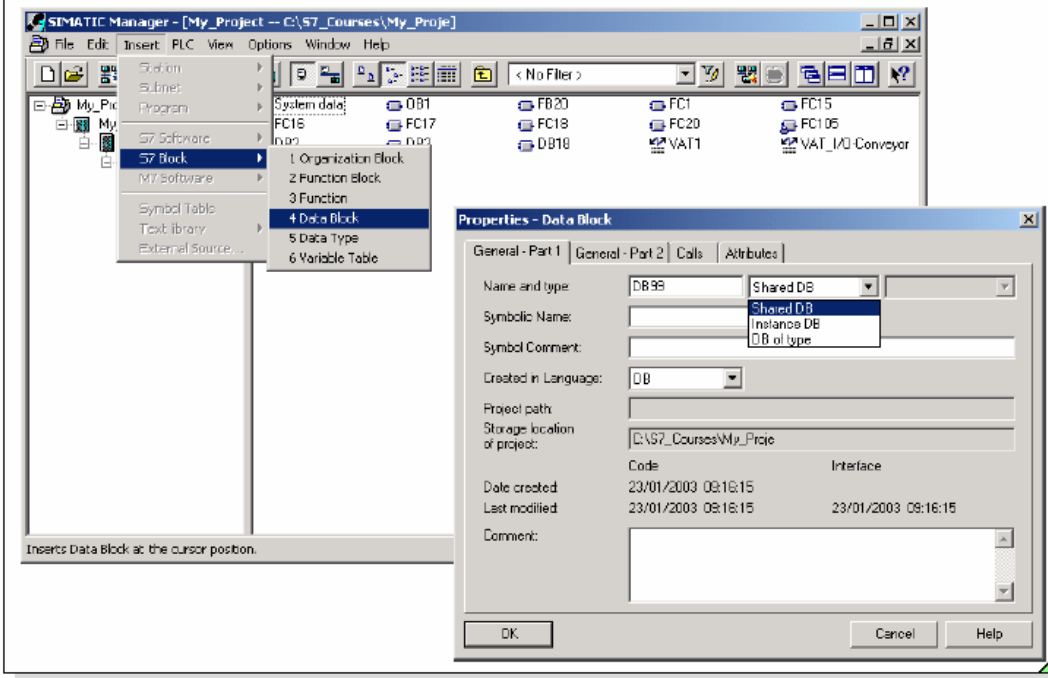
للتعامل مع الملفات المشتركة (DB Shared) يجب تذكر انواع البيانات كالتالي:

Elementary Data Types in STEP 7

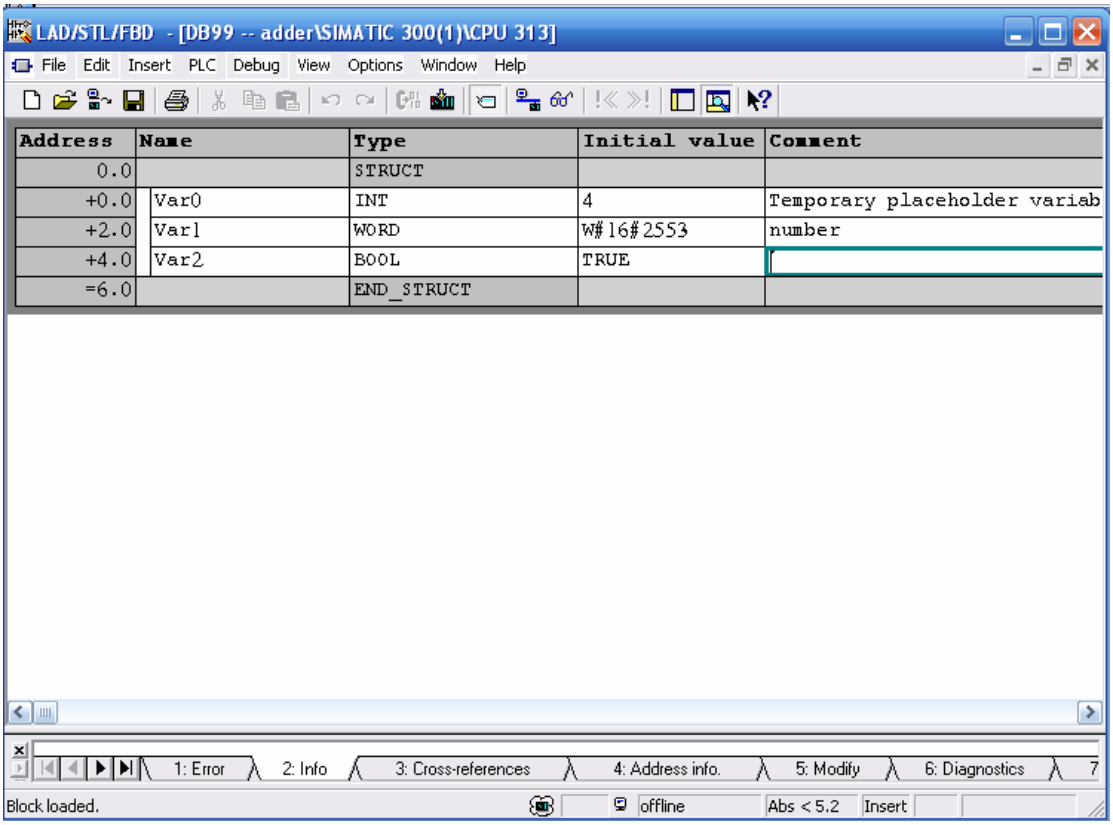
Keyword	Length (in bits)	Example of a constant of this type
BOOL	1	1 or 0
BYTE	8	B#16#A9
WORD	16	W#16#12AF
DWORD	32	DW#16#ADAC1EF5
CHAR	8	' w '
S5TIME	16	S5T#5s_200ms
INT	16	123
DINT	32	65539
REAL	32	1.2 or 34.5E-12
TIME	32	T#2D_1H_3M_45S_12MS
DATE	16	D#1993-01-20
TIME_OF_DAY	32	TOD#12:23:45.12

خطوات تكوين ملفات البيانات المشتركة:

Creating a New Data Block

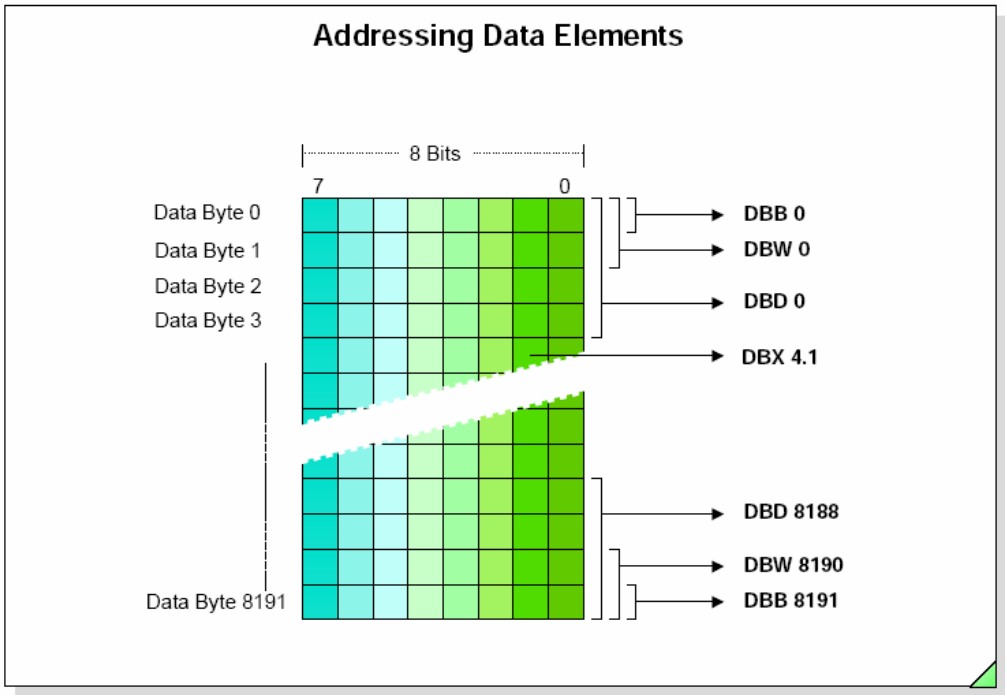


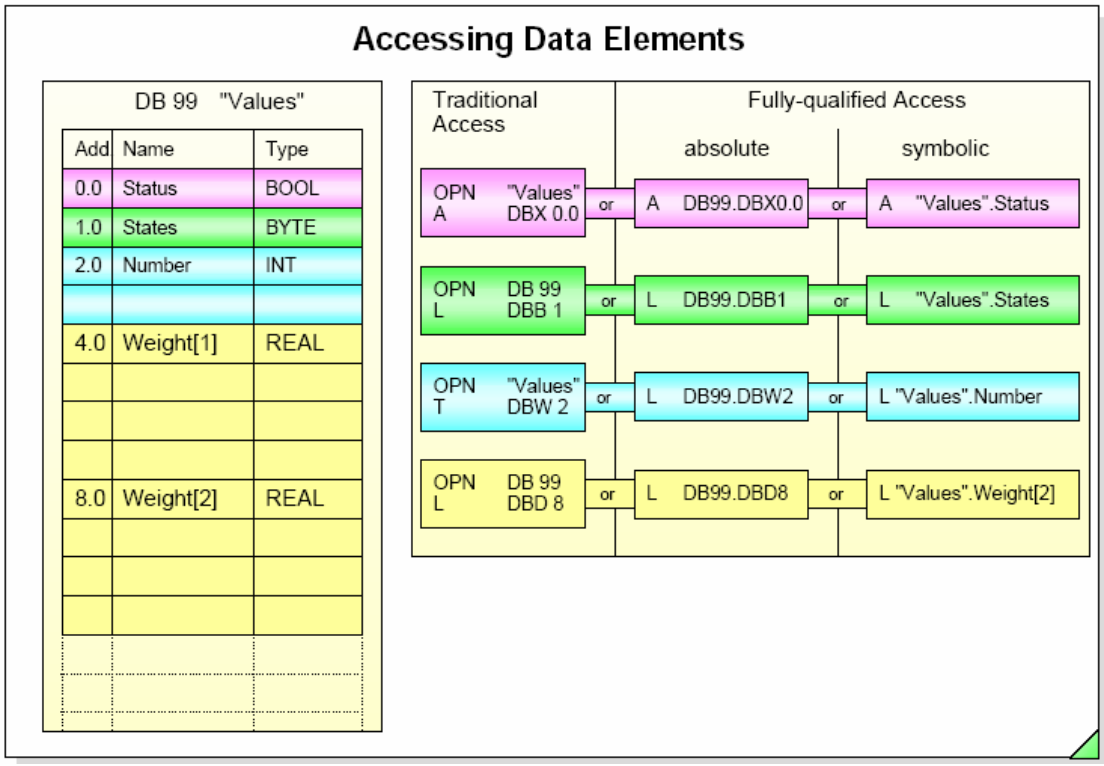
٢ـ نقوم بفتح الملف (DB99) الذي كوناه ونقوم باملاء البيانات كالتالي:



٣ـ نقوم بحفظ التغيرات وايضا عمل (Download) لملف البيانات (DB99) لانه محجوز في ذاكرة (CPU) وسنتعلم بالدرس القادم كيفية عمل (Download)

٤ـ استدعاء البيانات داخل البرنامج المنطقي يأخذ عتدة اشكال كالتالي:





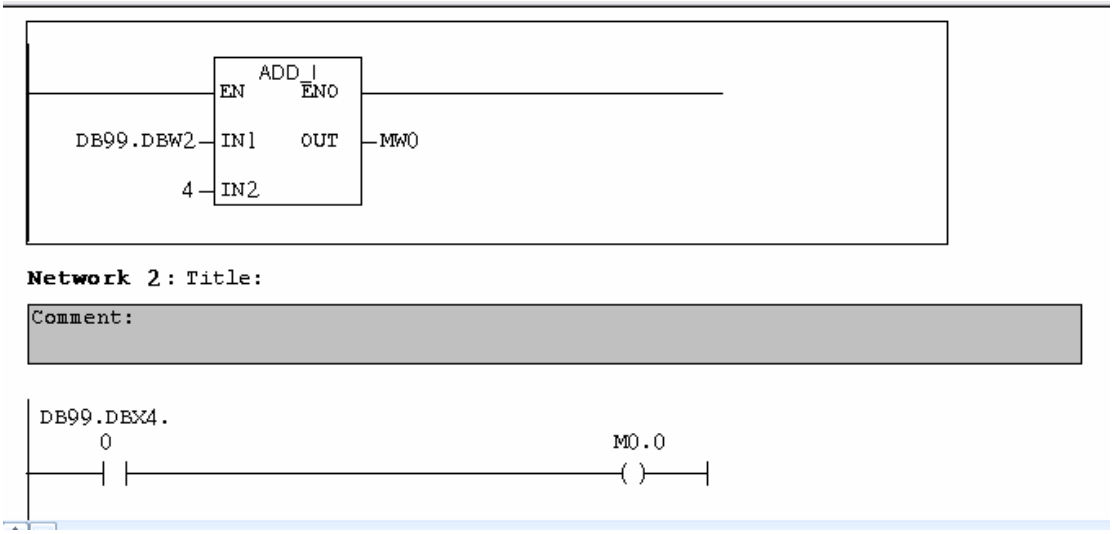
٥ـصيغة الاستدعاء كالتالي:

DB(Number).(DB Data Type)(Data Address)

DB(Number) , such as DB1,DB2,DB100…

DB Data Type Such as
(DBX) for Bool Data
(DBB) for Byte Data
(DBW) for Word Data
(DBD) for Double Word Data
Data Address Such as
0.0, 2.0, 4.0, for Bool
1,2,3,4 For other Types

٦- نفوم بكتابة البرنامج التالي داخل (OB1) لتوضيح عملية الاستدعاء



٧- تم استدعاء قيم (Var1) و(Var2) من ملف البيانات (DB99) باستخدام عنوان المتغير

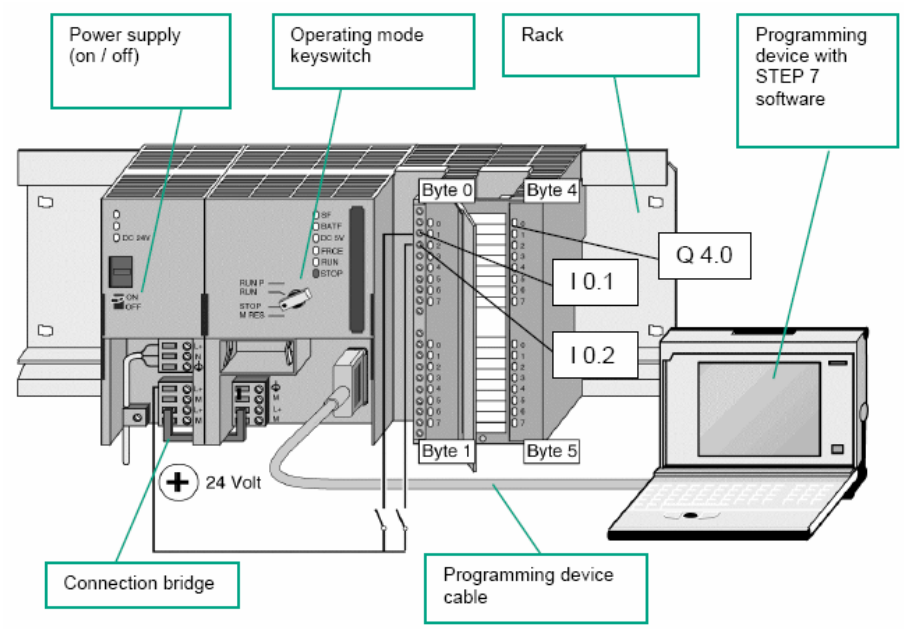
	BOOL	CHAR BYTE	DATE S5 TIME INT WORD	TIME OF DAY IEC TIME DINT REAL DWORD
	Bit	Byte	Word	Double Word
Input	I0.0	IB0	IW0	ID0
Output	Q0.0	QB0	QW0	QD0
Memory	M0.0	MB0	MW0	MD0
Data	DB1.DBX0.0	DB1.DBB0	DB1.DBW0	DB1.DBD0
Local	L0.0	LB0	LW0	LD0

اليوم السادس

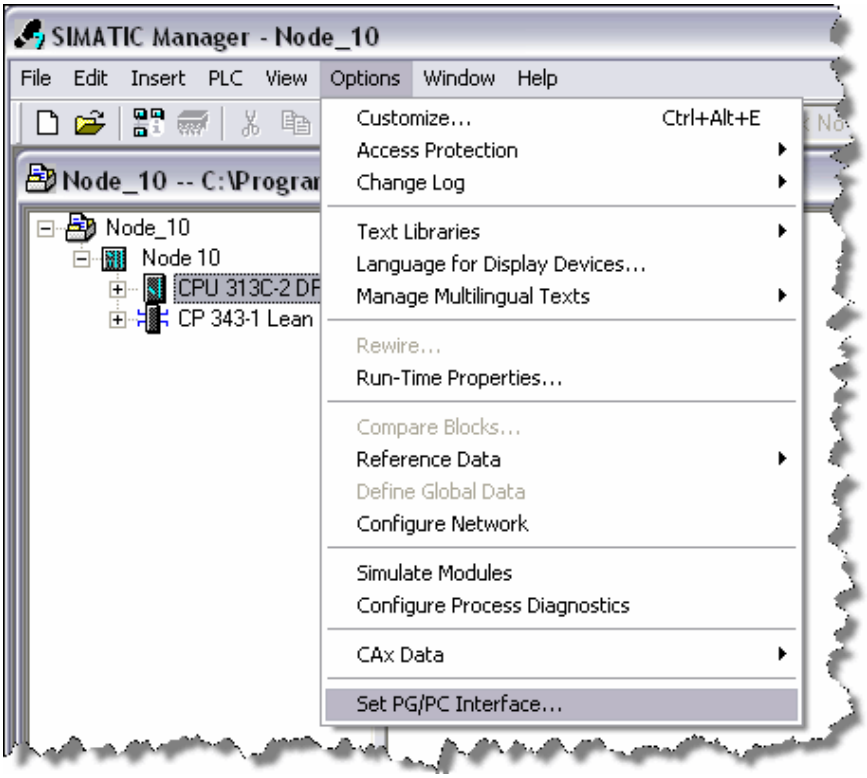
١ـ تكوين الارتباط (Online) مع (CPU)
٢ـ تحميل البرنامج المنطقي
٣ـاستخدام (Simulator)

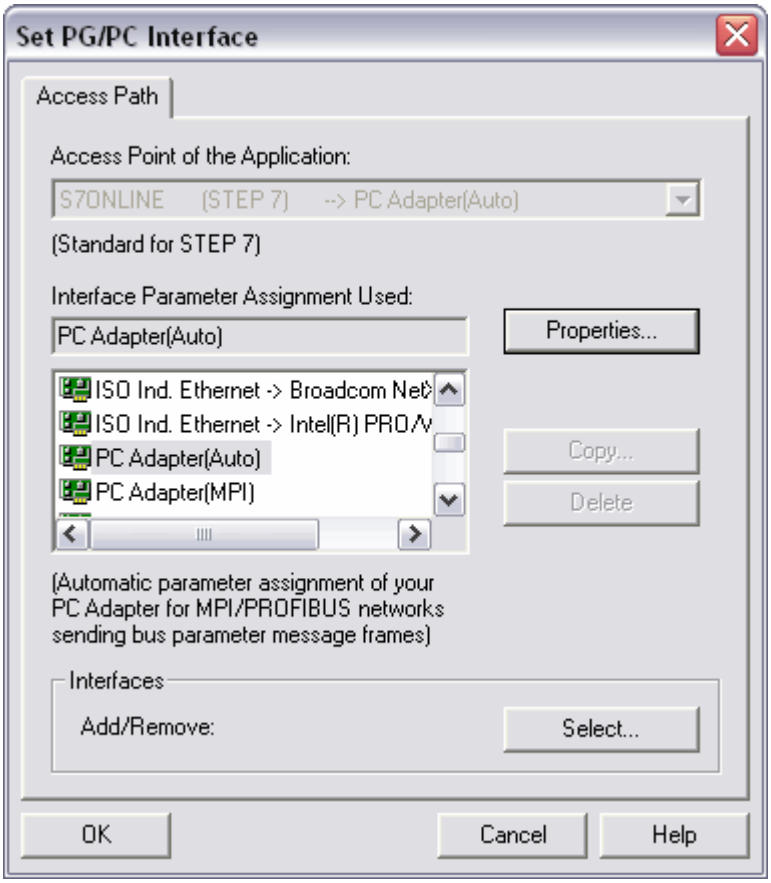
١ـ تكوين الارتباط (Online) مع (CPU)

أـ نقوم بربط الحاسبة مع (CPU) عن طريق (MPI)



بـ ضبط اعدادات الاتصال عن طريق برنامج (Step7) كالتالي:

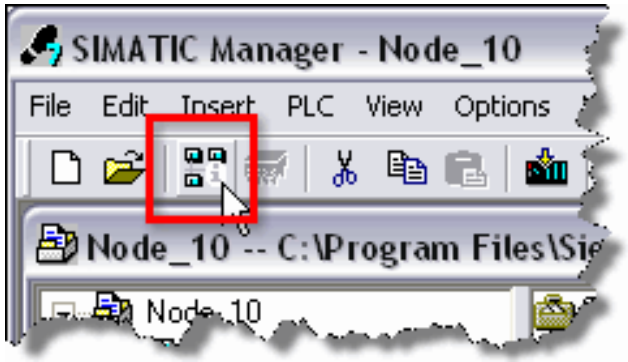




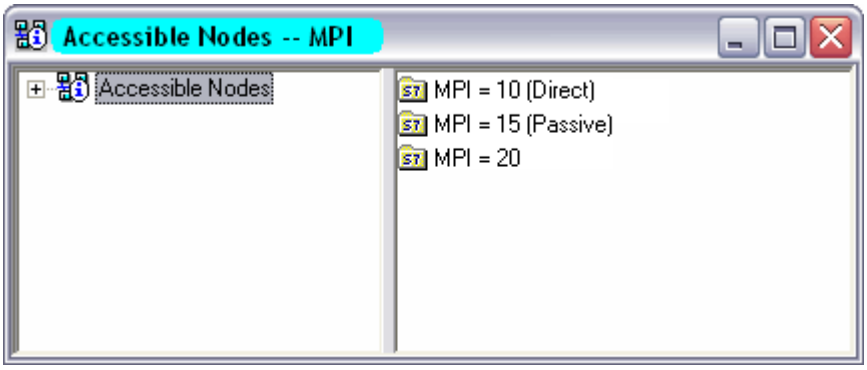
نقوم باختيار ((PC Adapter (Auto اذا كانت حاسبة اعتيادية موصولة باحد توصيلات (MPI) الخارجية

نقوم باختيار ((PC Adapter (MPI اذا كانت حاسبة اعتيادية موصولة باحد توصيلات (MPI) الخارجية في بعض الاحيان او حاسبة (PG) تحتوي على (MPI)

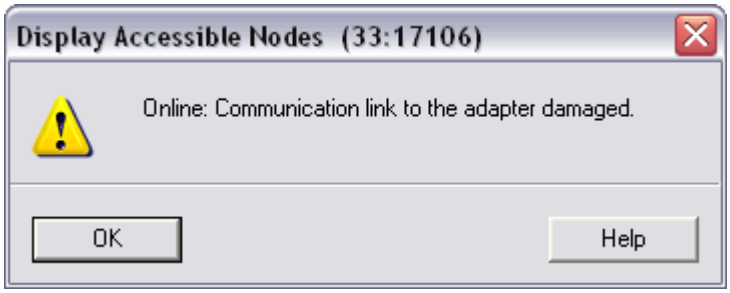
د- التأكد من نجاح الاتصال كالتالي:



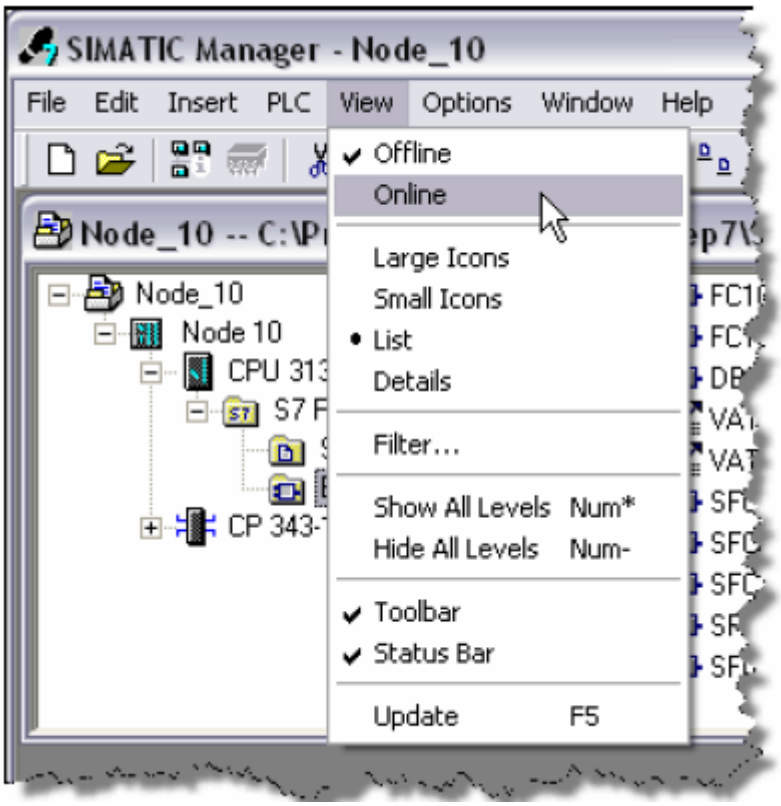
في حالة نجاح الاتصال تظهر النافذة التالية:



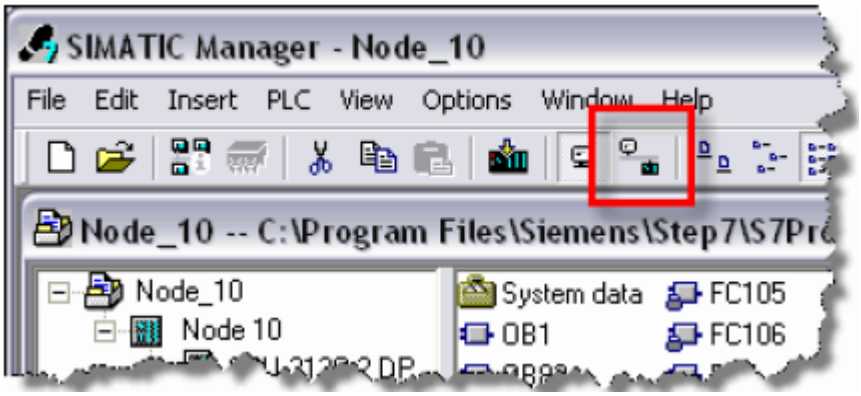
وفي حالة فشل الاتصال تظهر النافذة التالية فنقوم بالتأكد من اعدادات الاتصال



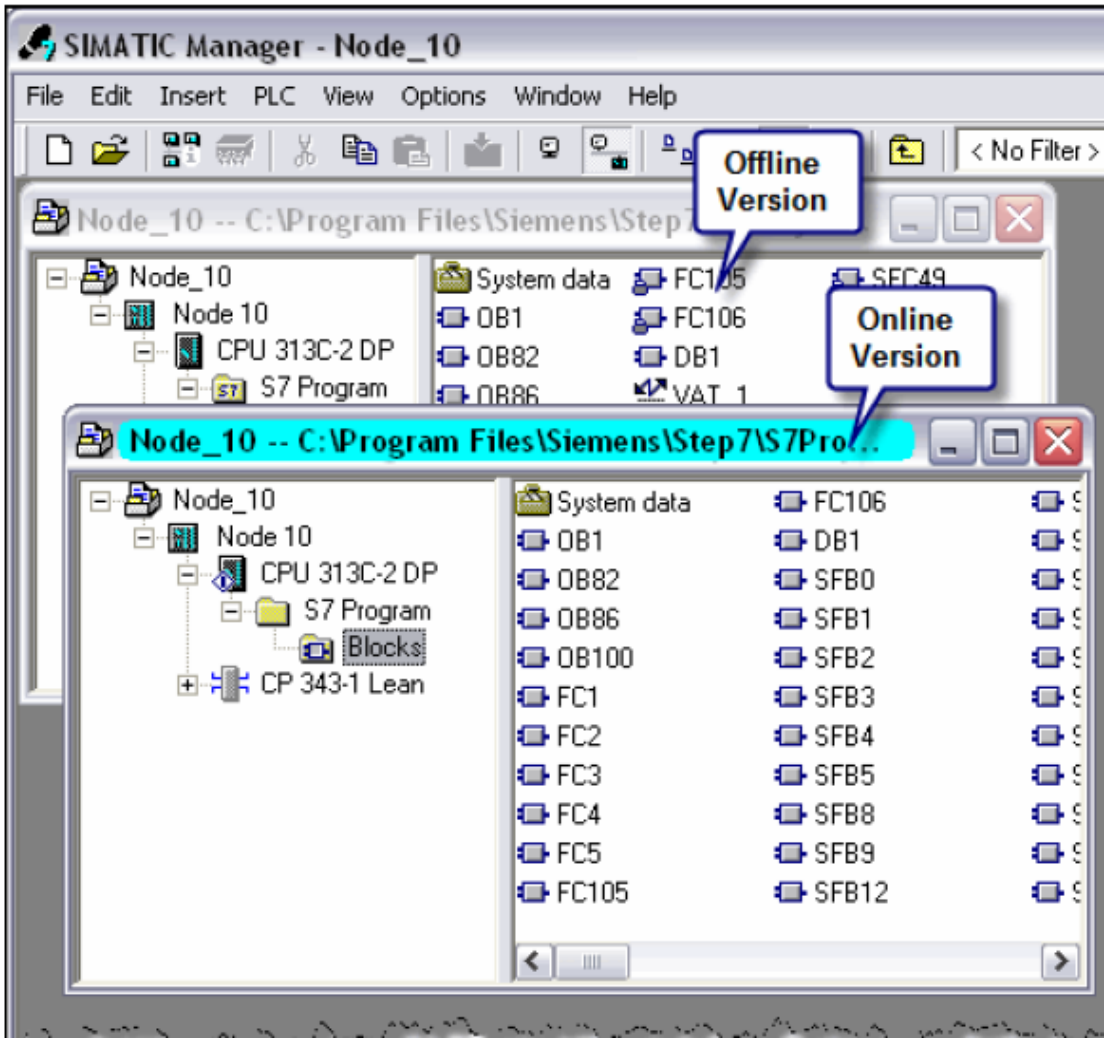
ه- ننتقل الى حالة (Online) كالتالي:



او من شريط الادوات



ستظهر النافذة التالية:



٢-تحميل البرنامج المنطقي

يكون تحميل البرنامج على نوعين

النوع الاول: تحميل البرنامج من الحاسبة الى ال (CPU) ويسمى (Download)

النوع الثاني: تحميل البرنامج من ال (CPU) الى الحاسبة ويسمى (Upload)

أ-تهيئة (CPU) لغرض عمل (Download)

يتم تنفيذ ذلك بالخطوات الثلاث الآتية :

الخطوة	المنفذ	النتيجة
١	أدر المفتاح إلى وضعية STOP	يضيء مؤشر STOP
٢	أدر المفتاح إلى وضعية MRES وأبقه في هذه الوضعية (حوالي ٣ ثوان) حتى يظهر مؤشر STOP من جديد	ينطفئ مؤشر STOP وبعد حوالي ٣ ثوان يعود مجدداً. من أجل وحدات المعالجة الحديثة انتظر حتى يضيء مؤشر STOP للمرة الثانية.
٣	أعد المفتاح إلى وضعية STOP وخلال الثانيتين التاليتين أعد الإطلاق في وضعية MRES.	يومض مؤشر STOP لمدة حوالي ٣ ثوان ثم يضيء مرةً أخرى بشكلٍ عادي، عندئذٍ يكون كل شيء جاهزاً ويكون قد تمت إعادة تصفير وحدة المعالجة

Switch on the power supply using the ON/OFF switch. The diode "DC 5V" will light up on the CPU.

Turn the operating mode switch to the STOP position (if not already in STOP). The red "STOP" LED will light up.

Resetting the CPU and Switching it to RUN

Turn the operating mode switch to the **MRES** position and hold it there for at least 3 seconds until the red "STOP" LED starts flashing slowly.

Release the switch and, after a maximum of 3 seconds, turn it to the **MRES** position again. When the "STOP" LED flashes quickly, the CPU has been reset.

If the "STOP" LED does not start flashing quickly, repeat the procedure.

A memory reset deletes all the data on the CPU. The CPU is then in the initial state.

Downloading the Program to the CPU

Now turn the operating mode switch to "STOP" again to download the program.

ب- يكون ال (Download) ام جزئي لبعض ملفات (Blocks) او كلي لكل اجزاء المشروع في بعض الاحيان يحتاج تحويل مفتاح (CPU) الى وضع (Stop) اثناء عملية ال (Downlod) للاجزاء التالية:

١- تحميل المشروع باكمله

٢- تحميل الوحدة التنظيمية (OB1)

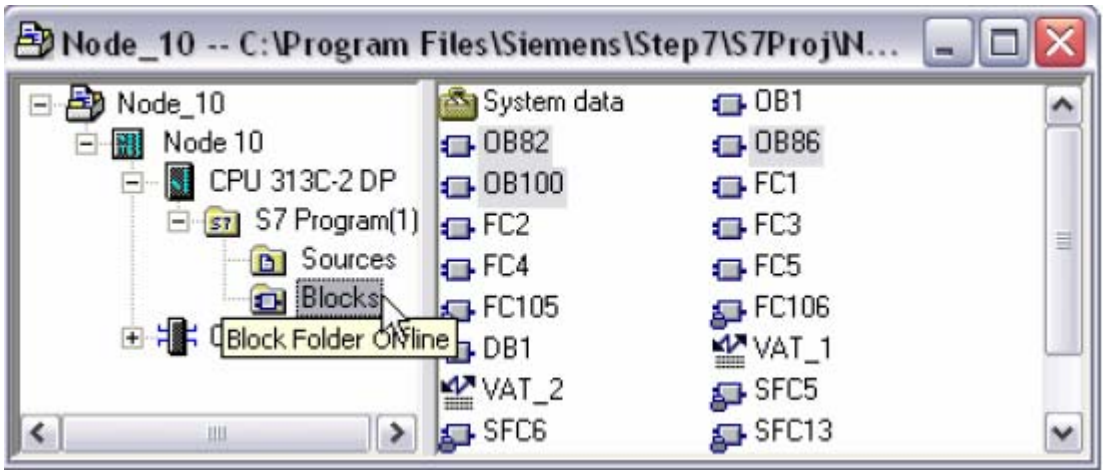
٣- تحميل الوحدات الوظيفية التابعة للنظام (SFB) سيتم شرحها بالجزء الثاني انشاء الله

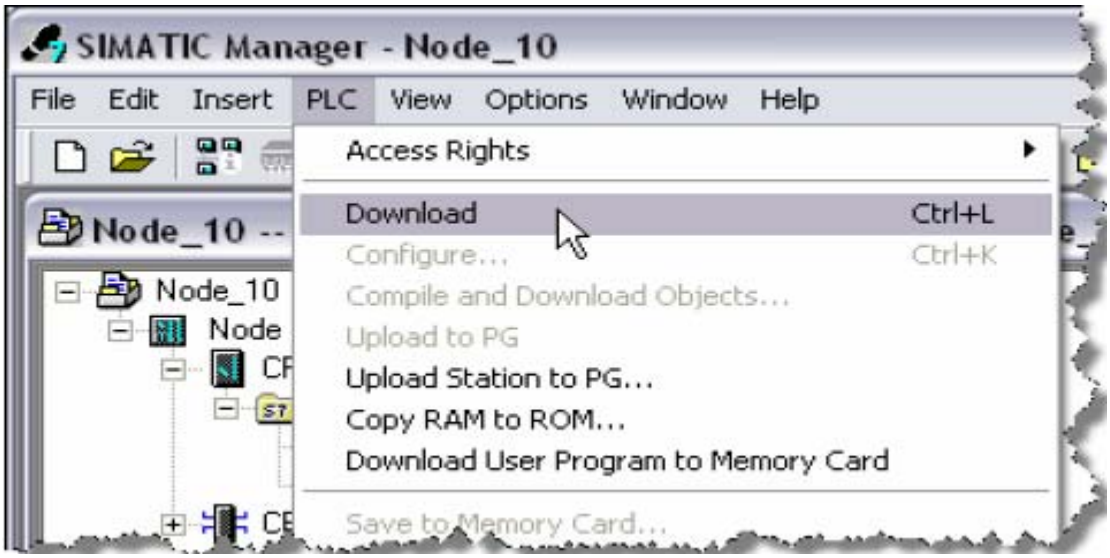
٤-تحميل ملفات البيانات (DB)

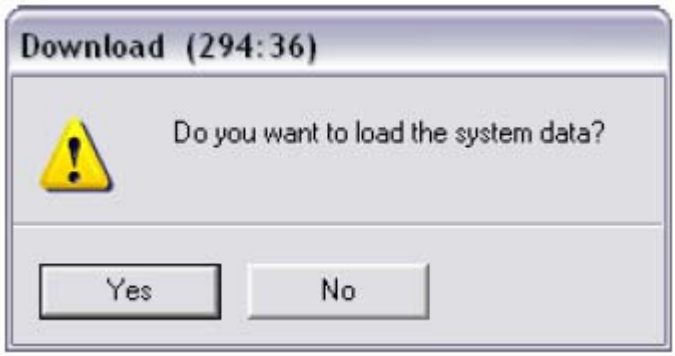
٥- تحميل نافذة (Hardware)

وعموما البرنامج يخبرك في حالة احتياجه الى تغيير وضع المفتاح اما باقي الاجزاء فيمكن تحميلها والمفتاح على وضع (Run-P)

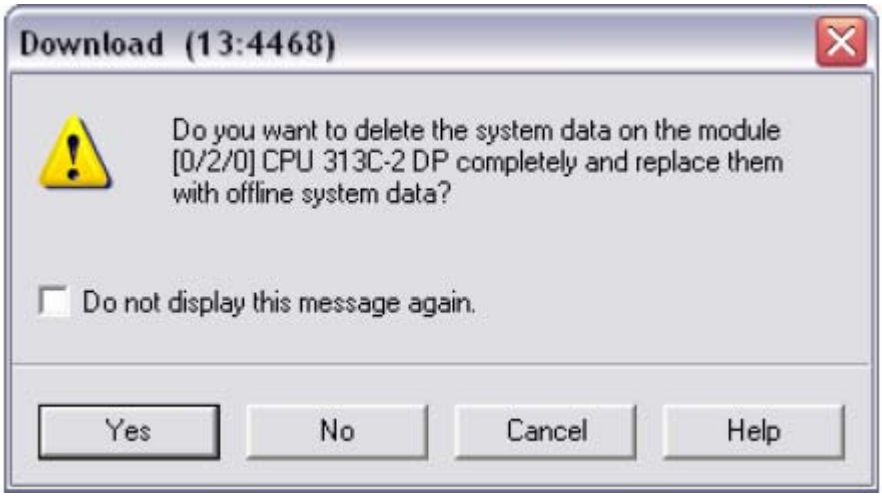
خطوات التحميل الجزئي الى (CPU)

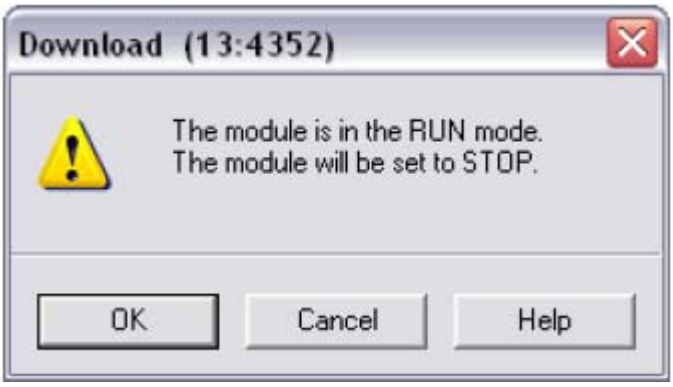


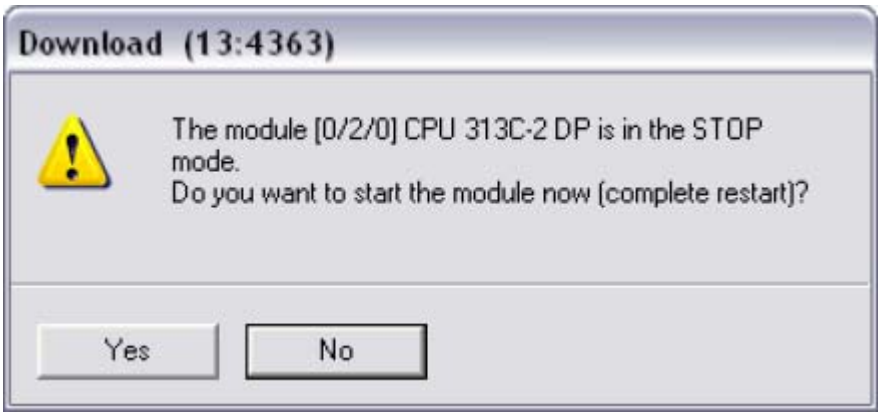




اضغط على (Yes)

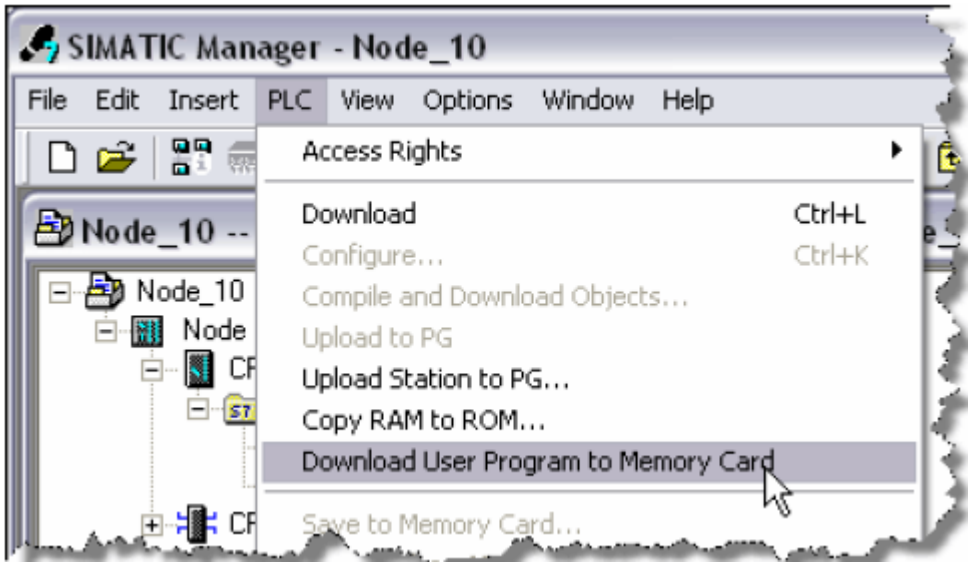






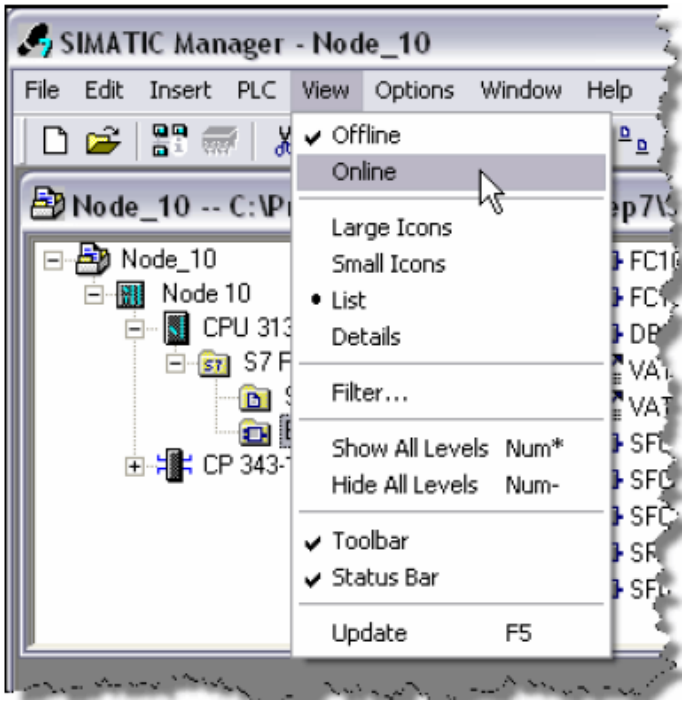
اضغط على (Yes) وحول المفتاح الى وضع (RUN)

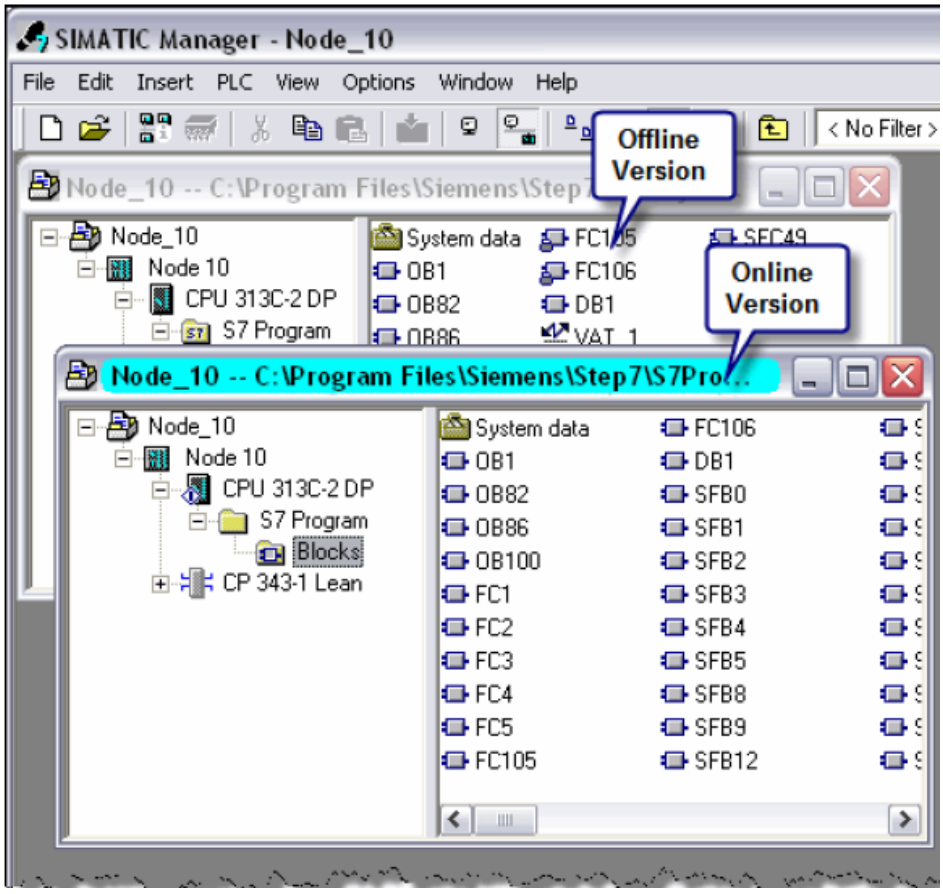
خطوات التحميل الكلي الى (CPU)

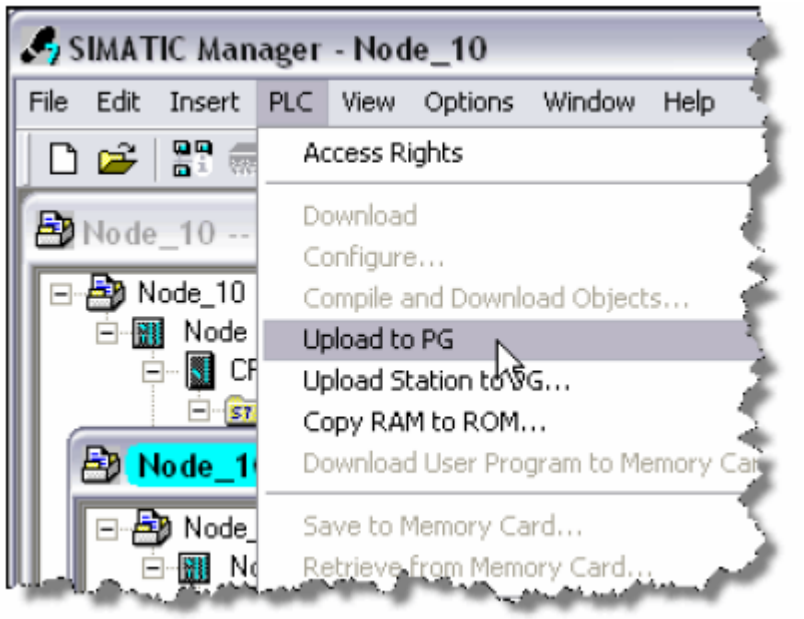


نضغط على (yes) ليتم مسح ذاكرة (CPU) وعمل تحميل لكل المشروع

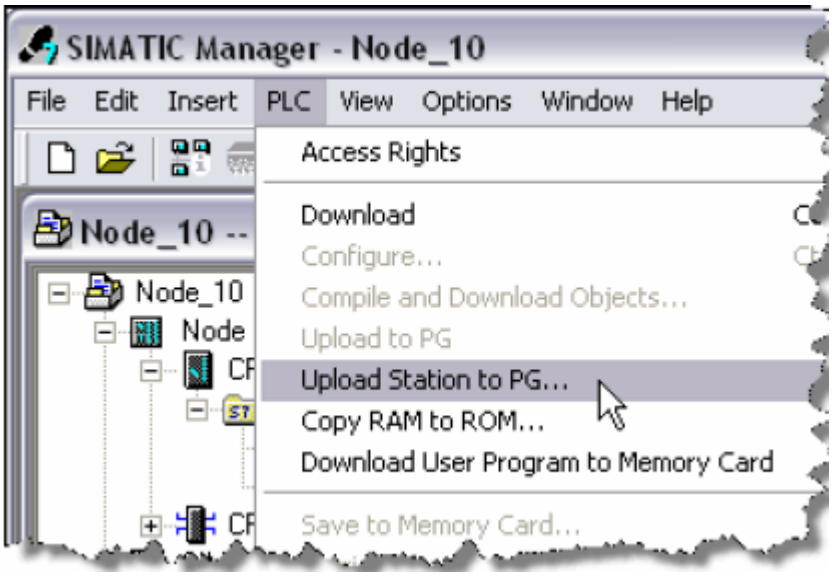
تحميل البرنامج من ال (CPU) الى الحاسبة ويسمى (Upload)
ويكون على نوعين:
ان يكون لدينا المشروع مخزون بالحاسبة فنقوم بفتحه وبعد ضبط اعدادات الاتصال نتبع الخطوات التالية:

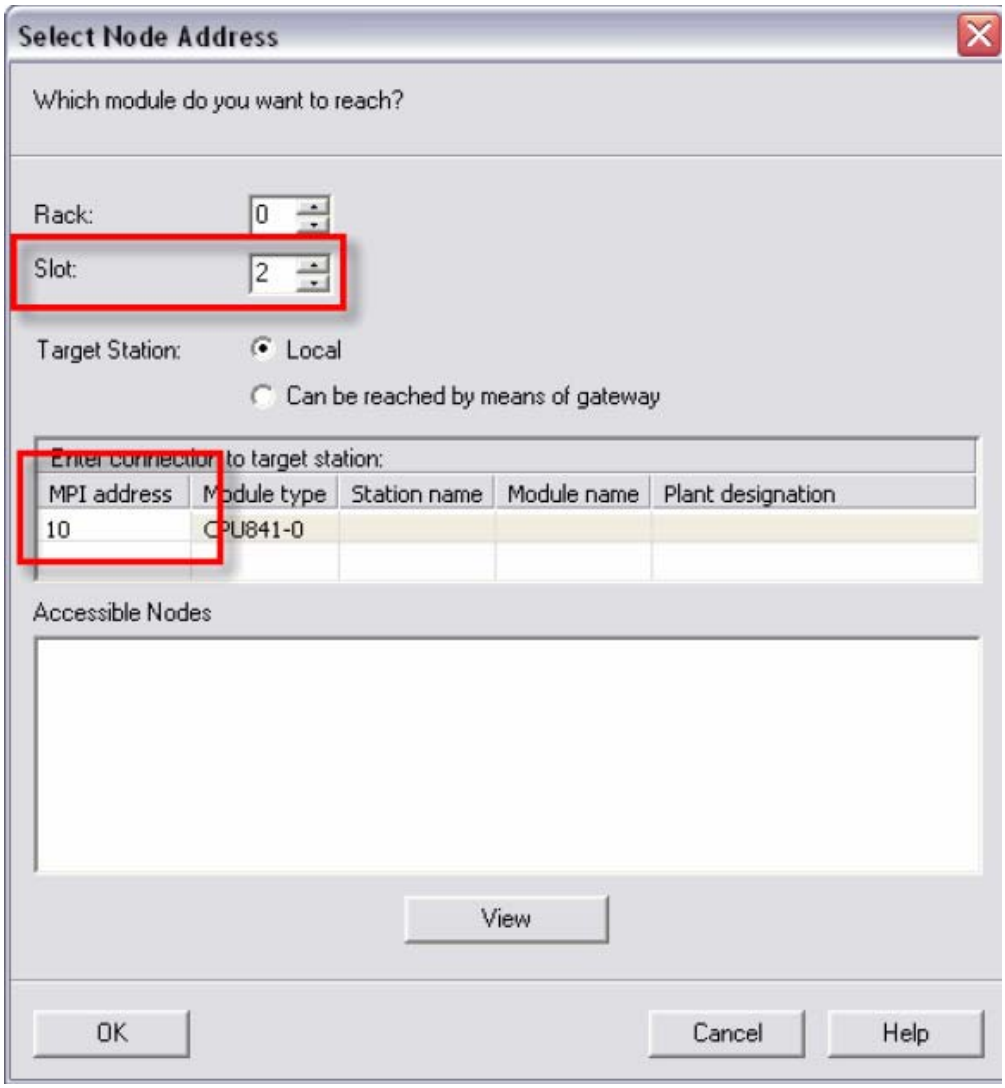




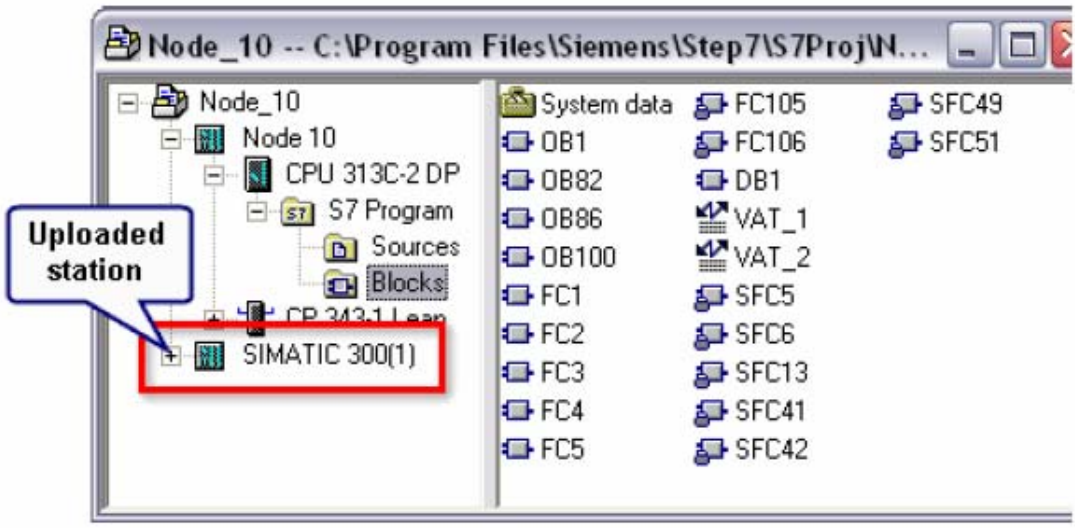


الحالة الثانية عندما لايكون لدينا المشروع مخزون داخل الحاسبة فنقوم بتكوين مشروع جديد او فتح اي مشرع المهم اعدادات الاتصال مع ال (CPU) صحيحة فنتبع الخطوات التالية:





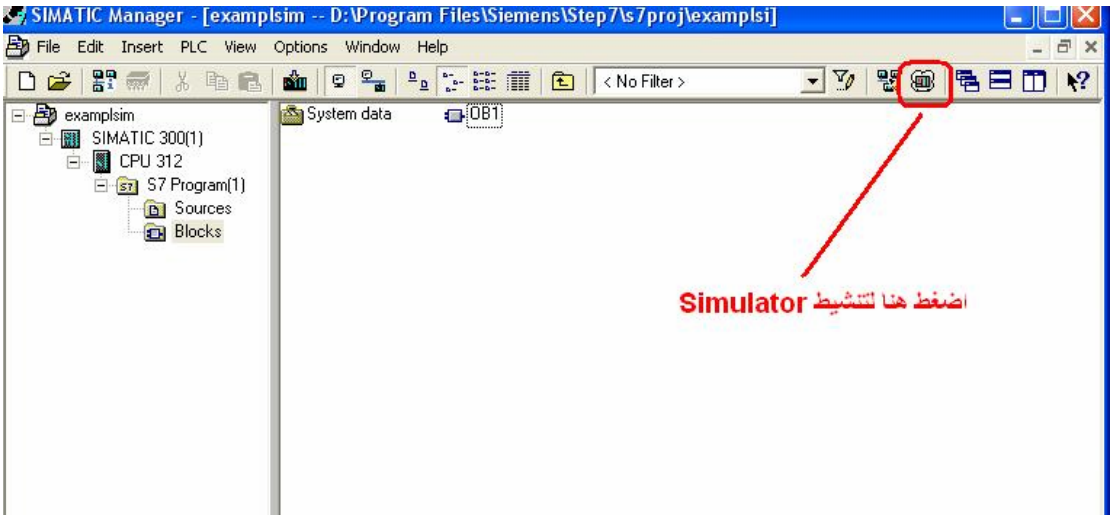
اذا لم يظهر لنا (MPI Address) نضغط على (View) ثم نختار ال (MPI Address) الصحيح في حالة وجود اكثر من واحد نتيجة ربط ال (MPI) مع جهاز خارجي ثم نضغط على (OK) ليتم تحميل البرنامج الى الحاسبة

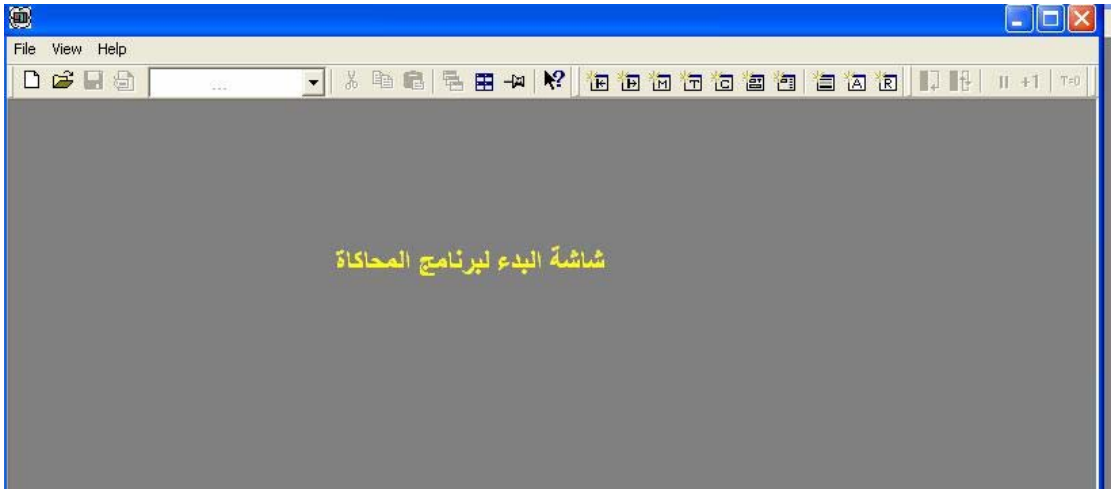


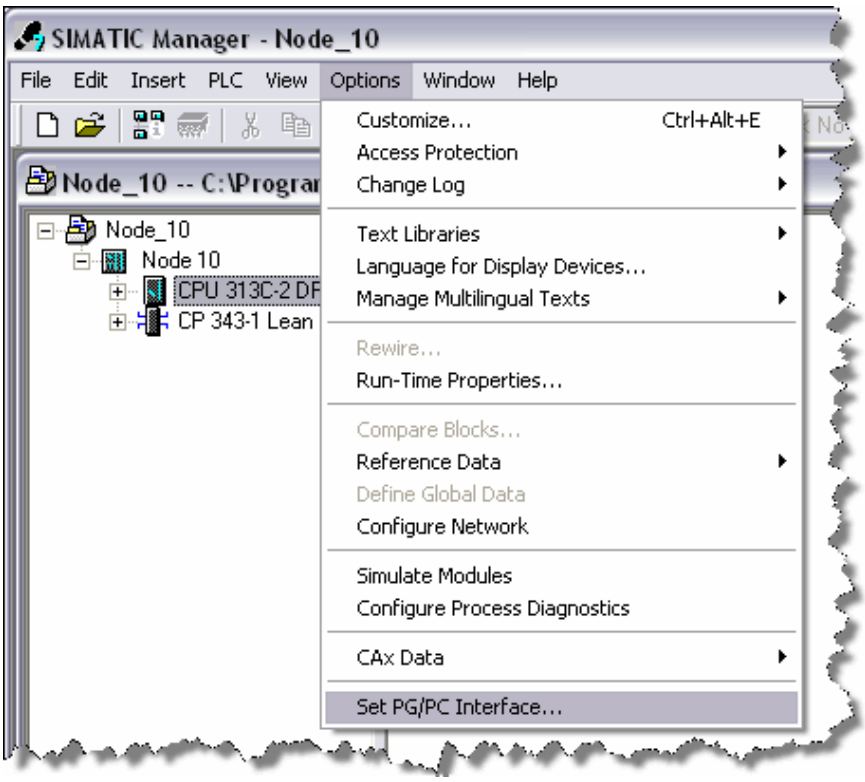
٣-استخدام (Simulator)

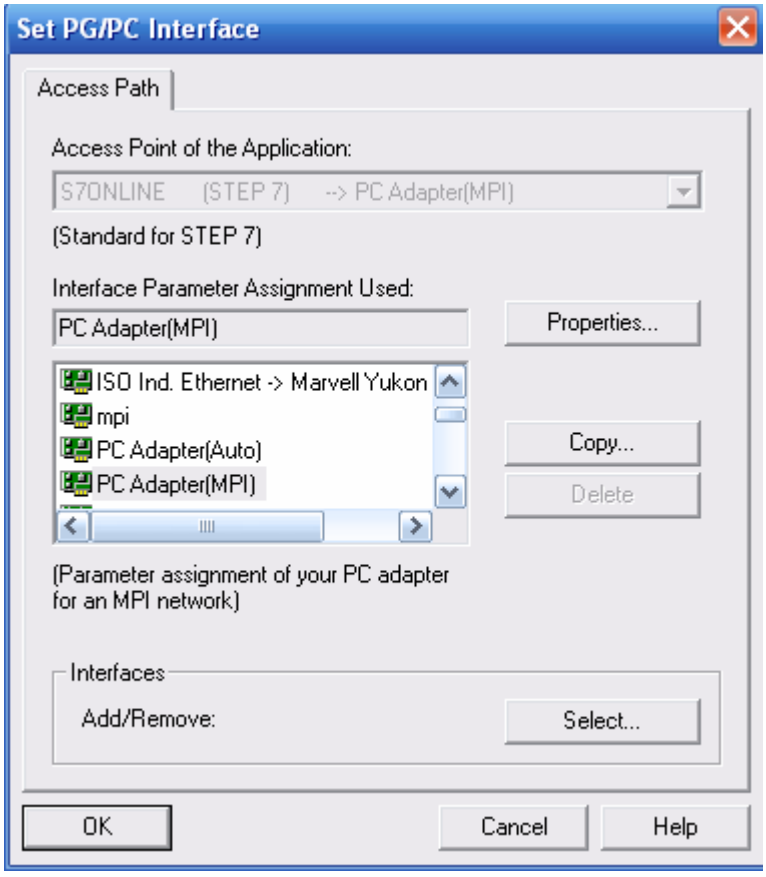
ان برنامج (Simulator) يساعدنا كثيرا لفحص البرنامج قبل تحميله ولكن يتطلب اولا تنصيب برنامج (S7PLCSIM) المرفق مع اسطوانة برنامج (Step7) ثم تفعيله بنفس طريقة تفعيل برنامج (Step7)

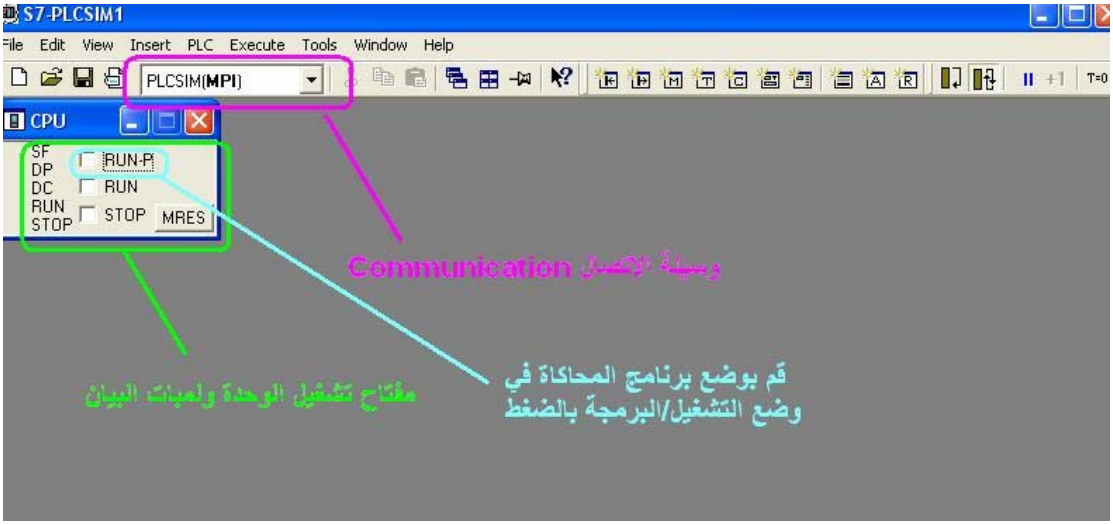
خطوات استخدام (Simulator)

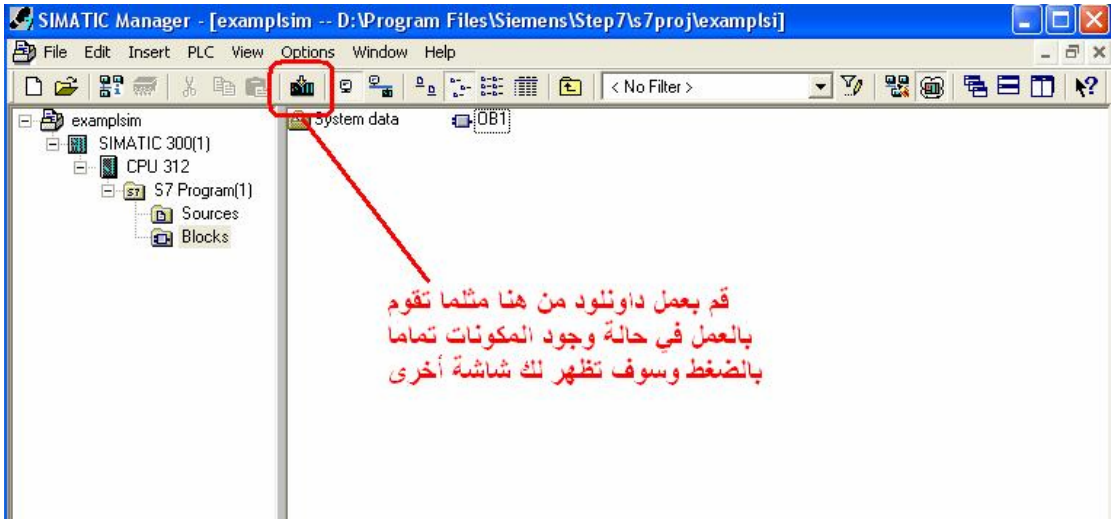


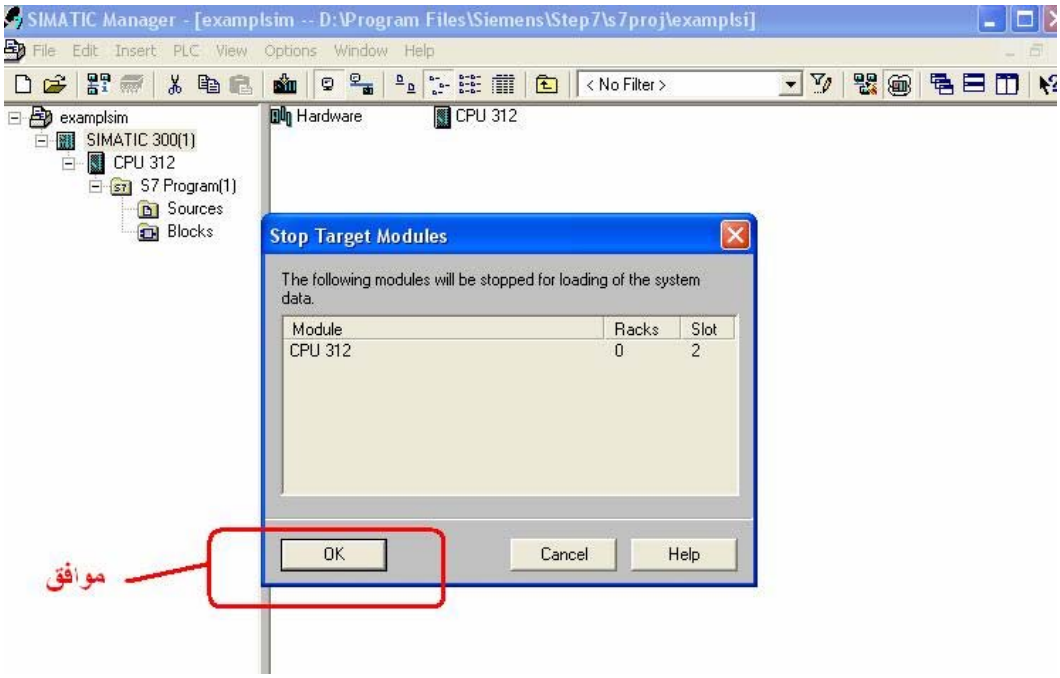


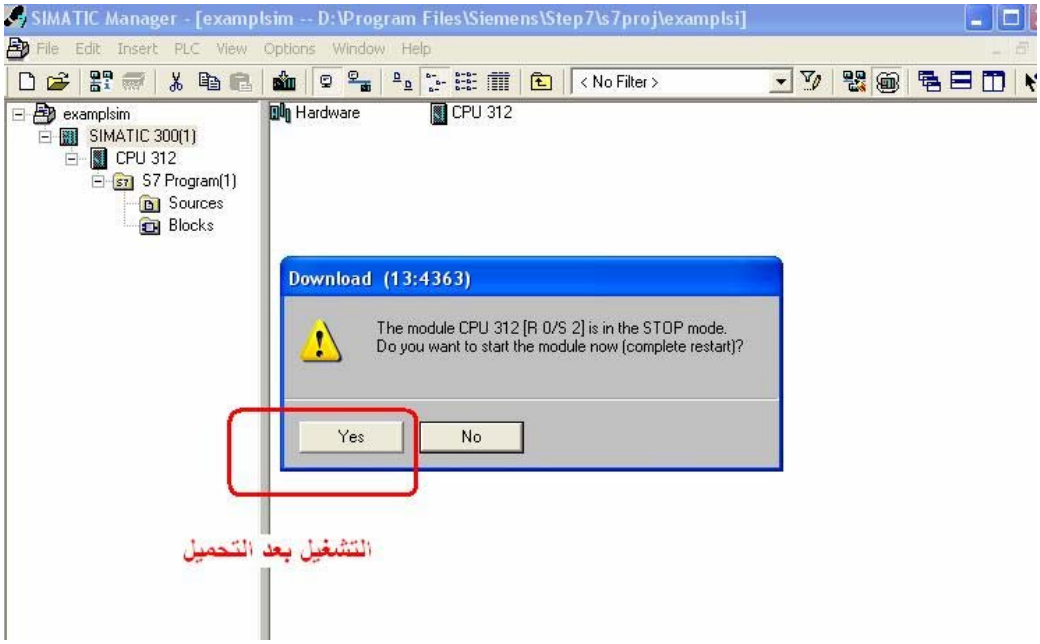

SIMATIC Manager - Node_10
File Edit Insert PLC View Options Window Help
Node_10 -- C:\Prograr
Node_10
Node 10
CPU 313C-2 DF
CP 343-1 Lean
Customize... Ctrl+Alt+E
Access Protection
Change Log
Text Libraries
Language for Display Devices...
Manage Multilingual Texts
Rewire...
Run-Time Properties...
Compare Blocks...
Reference Data
Define Global Data
Configure Network
Simulate Modules
Configure Process Diagnostics
CAx Data
Set PG/PC Interface...

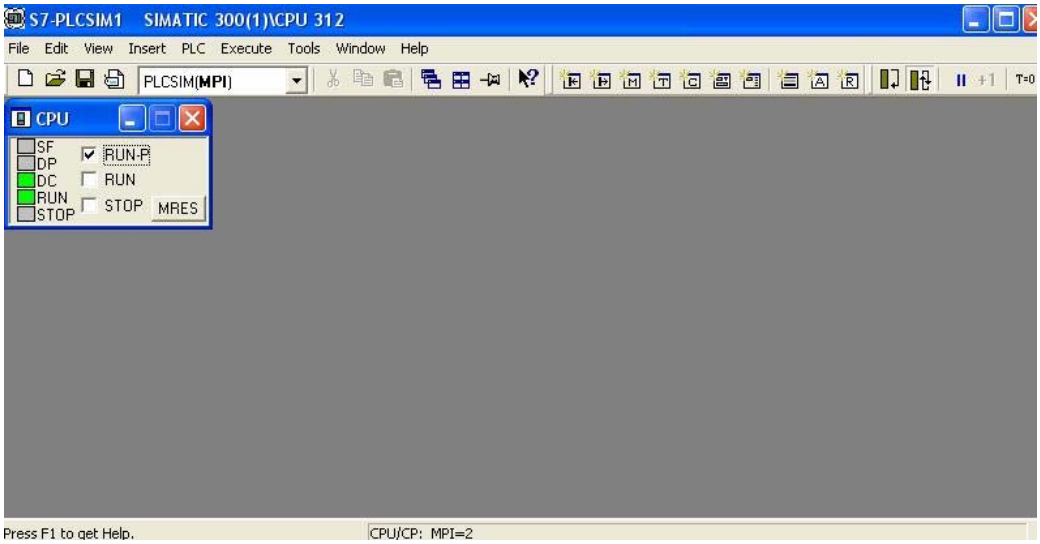

Set PG/PC Interface
Access Path
Access Point of the Application:
S7ONLINE (STEP 7) --> PC Adapter(MPI)
(Standard for STEP 7)
Interface Parameter Assignment Used:
PC Adapter(MPI)
Properties...
ISO Ind. Ethernet -> Marvell Yukon
mpi
PC Adapter(Auto)
PC Adapter(MPI)
Copy...
Delete
(Parameter assignment of your PC adapter for an MPI network)
Interfaces
Add/Remove:
Select...
OK
Cancel
Help

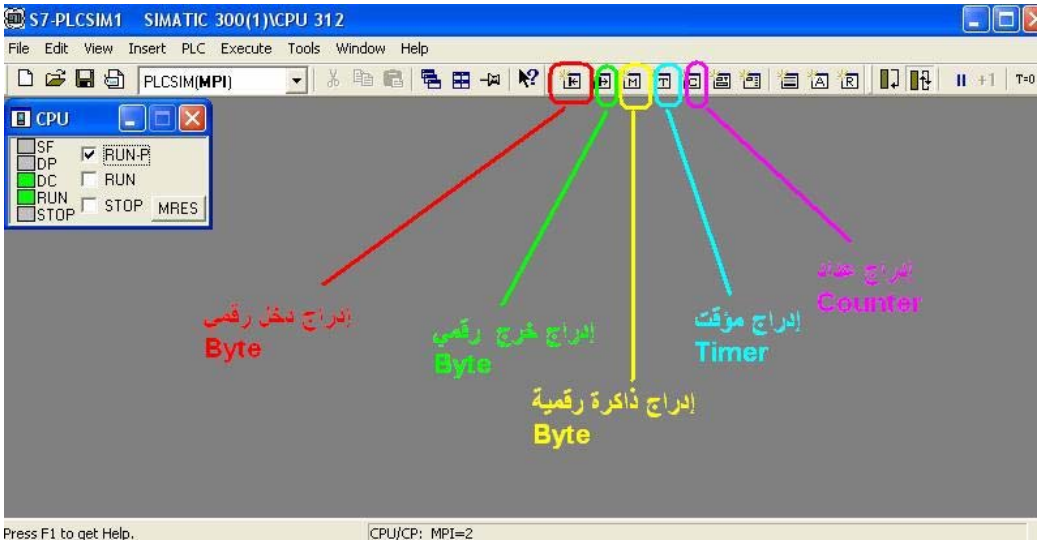

S7-PLCSIM1
File Edit View Insert PLC Execute Tools Window Help
PLCSIM(MPI)
CPU
SF
DP
DC
RUN
STOP
RUN-P
RUN
STOP
MRES
Communication وسيلة الاتصال
مفتاح تشغيل الوحدة ولعبات البيان
قم بوضع برنامج المحاكاة في
وضع التشغيل/البرمجة بالضغط

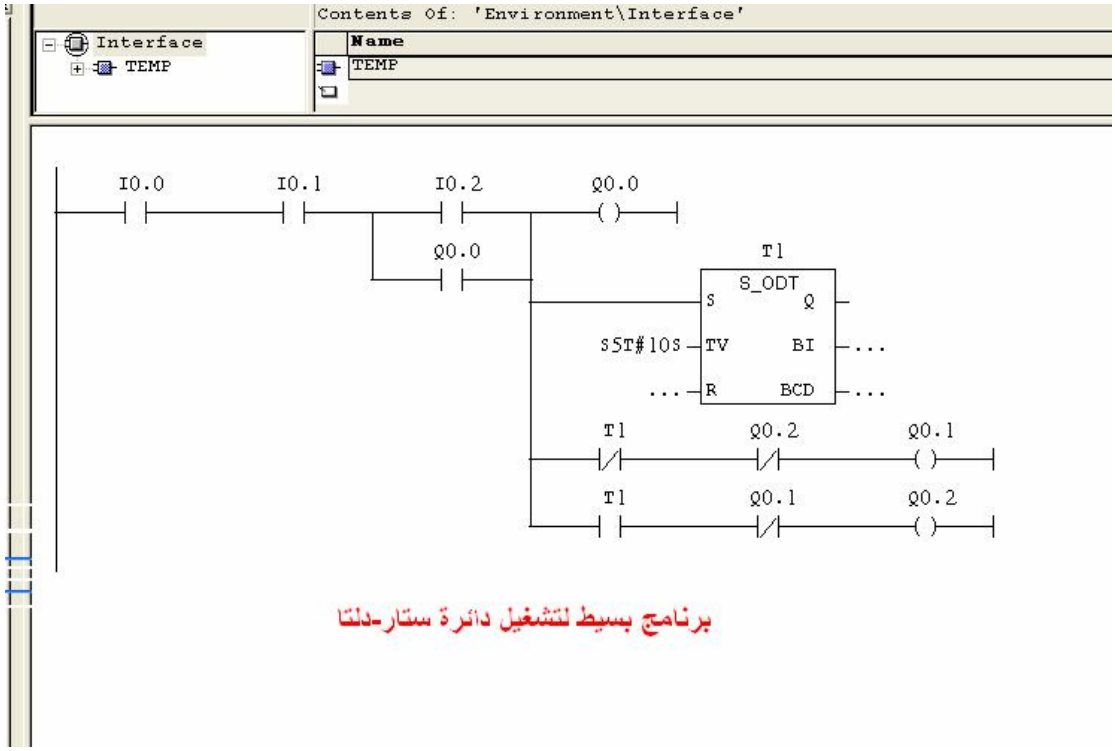

SIMATIC Manager - [examplsim -- D:\Program Files\Siemens\Step7\s7proj\examplsi]
File Edit Insert PLC View Options Window Help
< No Filter >
examplsim
SIMATIC 300(1)
CPU 312
S7 Program(1)
Sources
Blocks
System data
OB1
قم بعمل داونلود من هنا مثلما تقوم
بالعمل في حالة وجود المكونات تماما
بالضغط وسوف تظهر لك شاشة أخرى


SIMATIC Manager - [examplsim -- D:\Program Files\Siemens\Step7\s7proj\examplsi]
File Edit Insert PLC View Options Window Help
< No Filter >
examplsim
SIMATIC 300(1)
CPU 312
S7 Program(1)
Sources
Blocks
Hardware
CPU 312
Stop Target Modules
The following modules will be stopped for loading of the system data.
Module
Racks
Slot
CPU 312
0
2
OK
Cancel
Help
موافق

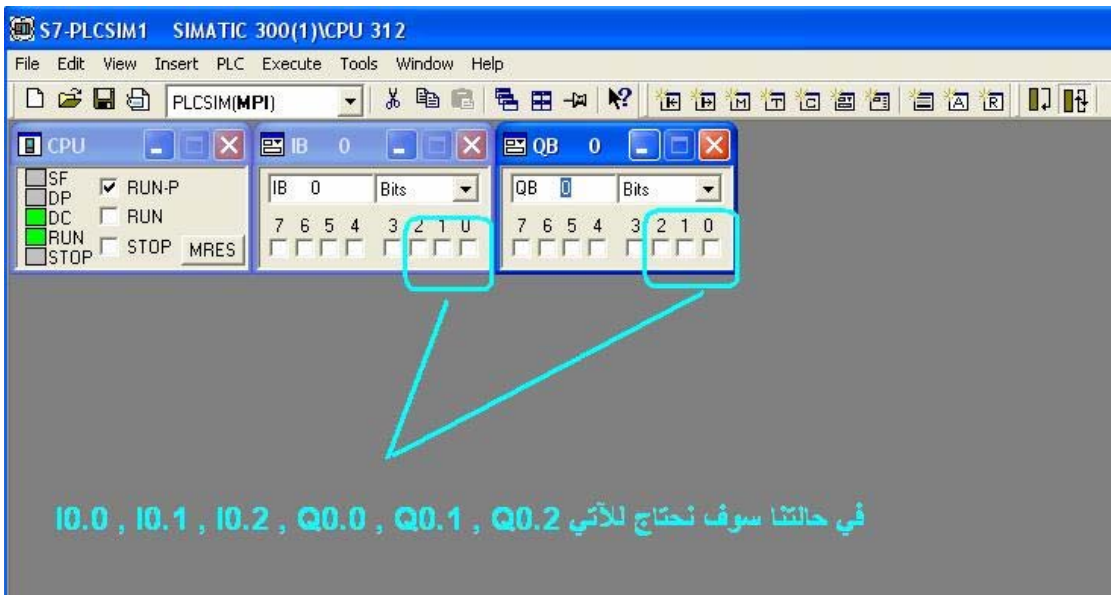

SIMATIC Manager - [examplsim -- D:\Program Files\Siemens\Step7\s7proj\examplsi]
File Edit Insert PLC View Options Window Help
< No Filter >
examplsim
SIMATIC 300(1)
CPU 312
S7 Program(1)
Sources
Blocks
Hardware
CPU 312
Download (13:4363)
The module CPU 312 [R 0/S 2] is in the STOP mode.
Do you want to start the module now (complete restart)?
Yes
No
التشغيل بعد التحميل

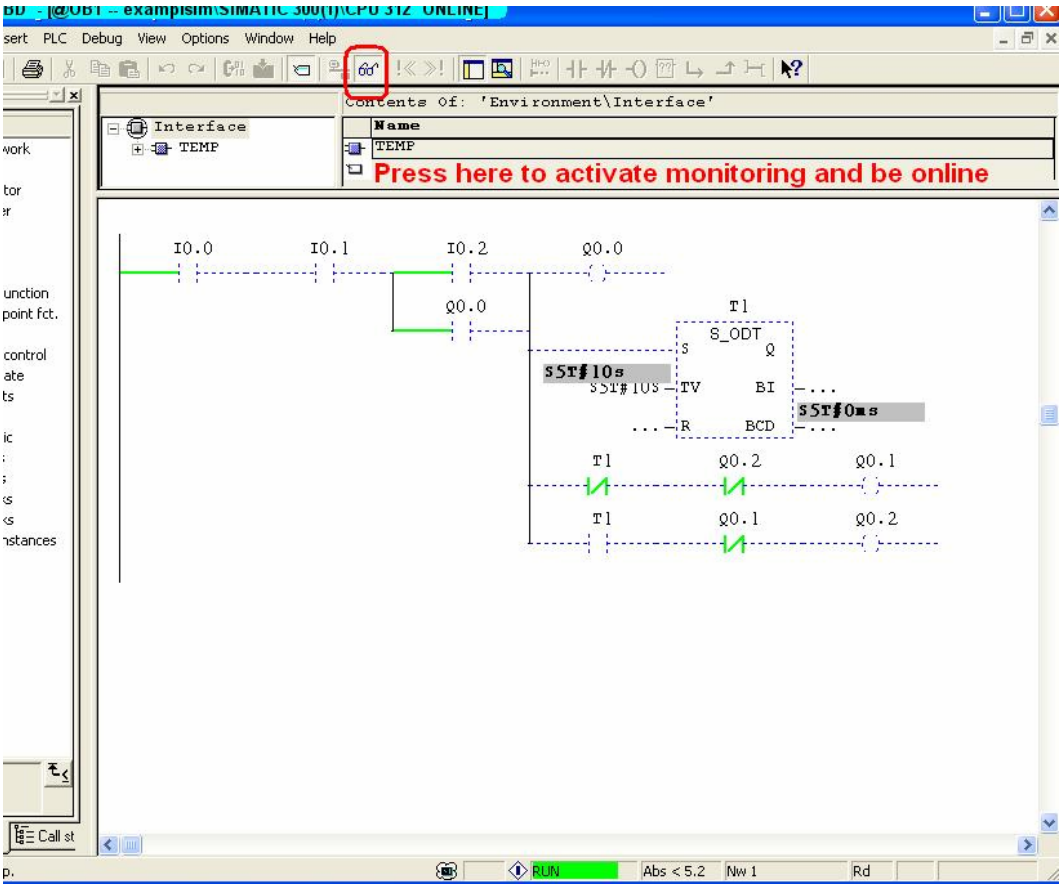

S7-PLCSIM1 SIMATIC 300(1)\CPU 312
File Edit View Insert PLC Execute Tools Window Help
PLCSIM(MPI)
CPU
SF
DP
DC
RUN
STOP
RUN-P
RUN
STOP
MRES
Press F1 to get Help.
CPU/CP: MPI=2

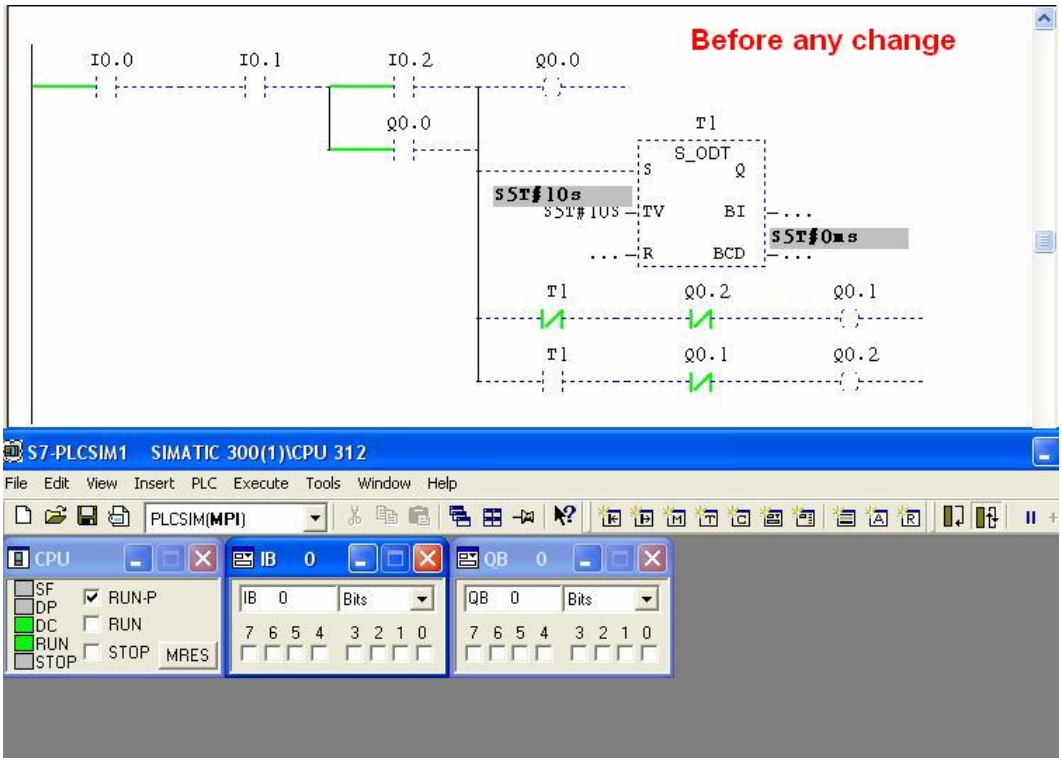

S7-PLCSIM1 SIMATIC 300(1)\CPU 312
File Edit View Insert PLC Execute Tools Window Help
PLCSIM(MPI)
CPU
SF
DP
DC
RUN
STOP
RUN-P
RUN
STOP
MRES
إدراج دخل رقمي
Byte
إدراج خرج رقمي
Byte
إدراج ذاكرة رقمية
Byte
إدراج مؤقت
Timer
إدراج عداد
Counter
Press F1 to get Help.
CPU/CP: MPI=2

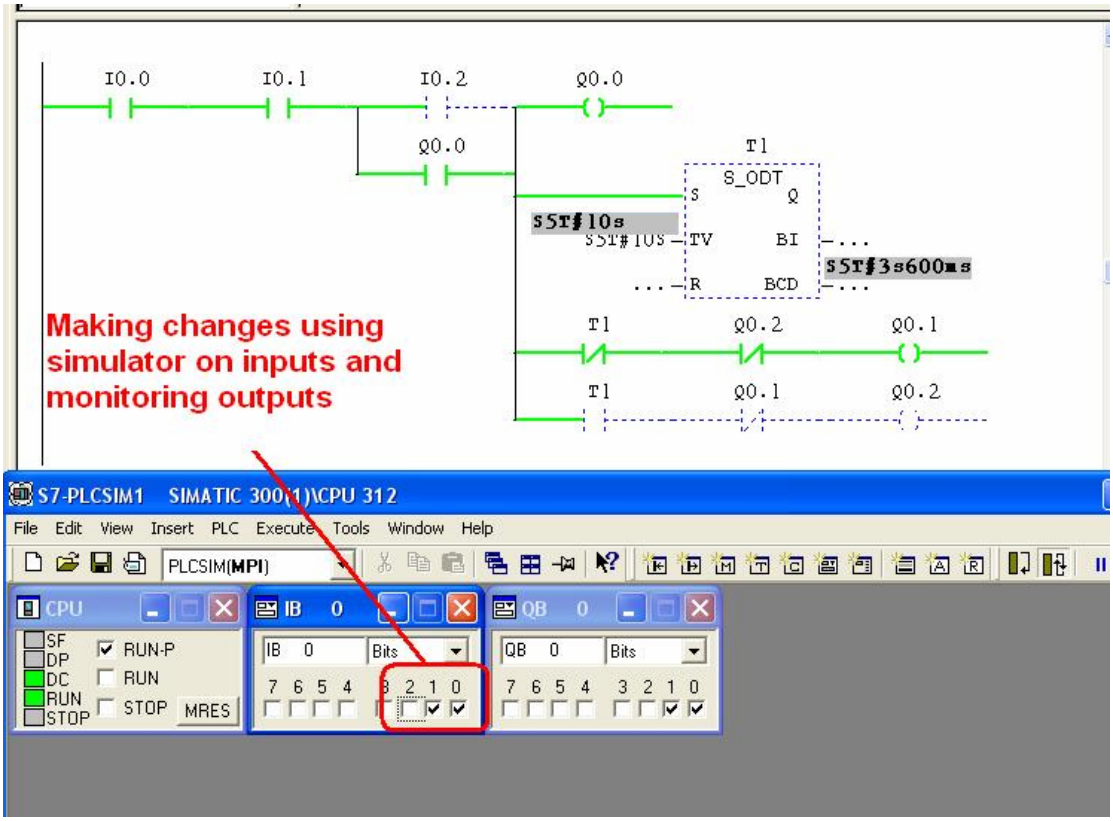


برنامج بسيط لتشغيل دائرة ستار-دلتا



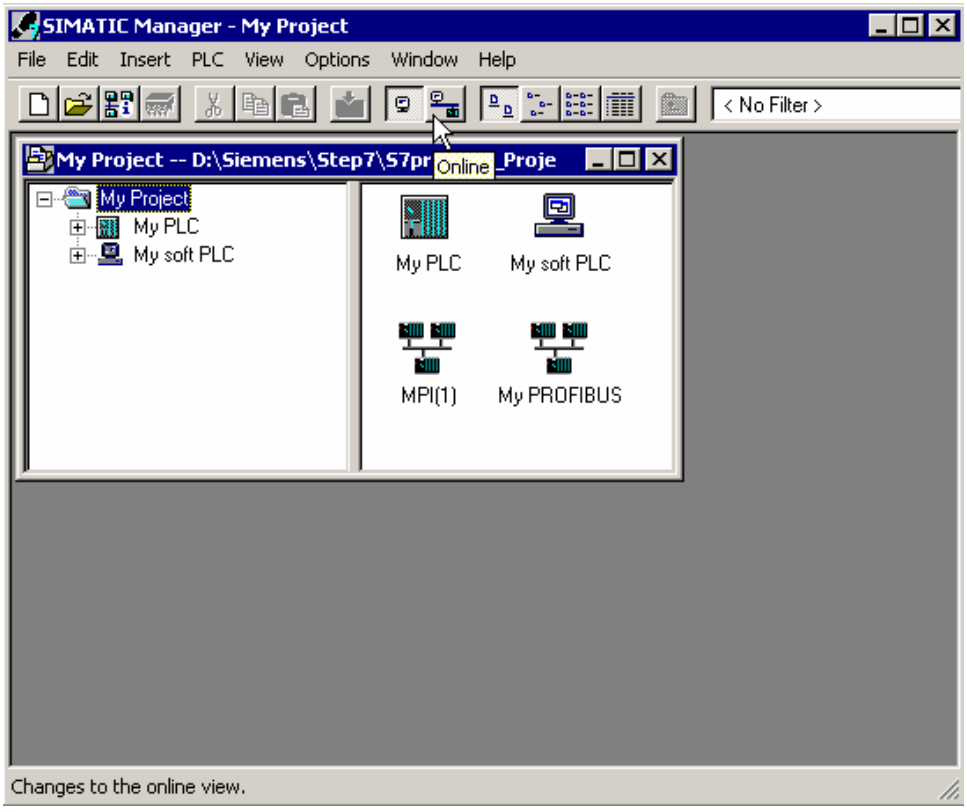

Press here to activate monitoring and be online
Interface
TEMP
Name
Contents of: 'Environment\Interface'
I0.0
I0.1
I0.2
Q0.0
T1
S_ODT
S5T#10s
S5T#0ms
Q0.1
Q0.2
RUN


Before any change
I0.0
I0.1
I0.2
Q0.0
T1
S_ODT
S5T#10s
S5T#0ms
Q0.1
Q0.2
S7-PLCSIM1 SIMATIC 300(1)\CPU 312
File Edit View Insert PLC Execute Tools Window Help
PLCSIM(MPI)
CPU
IB 0
QB 0
RUN-P
RUN
STOP
MRES

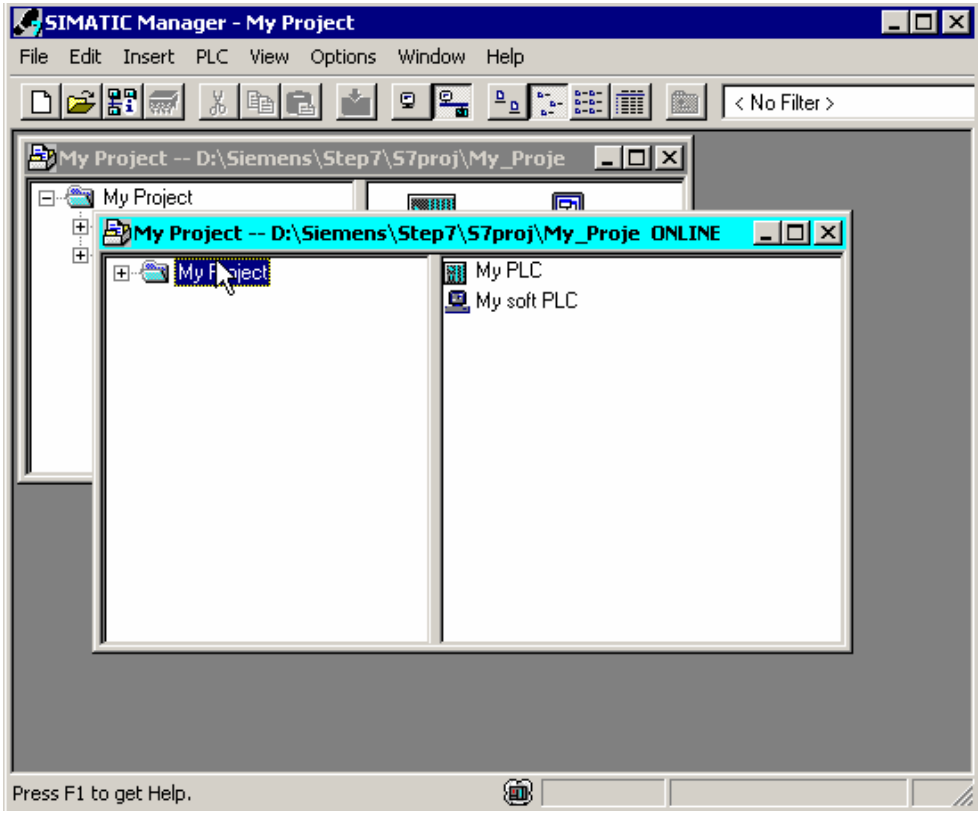

I0.0
I0.1
I0.2
Q0.0
Q0.0
T1
S_ODT
S
Q
S5T#10s
S5T#10S
TV
BI
R
BCD
S5T#3s600ms
Making changes using simulator on inputs and monitoring outputs
T1
Q0.2
Q0.1
T1
Q0.1
Q0.2
S7-PLCSIM1 SIMATIC 300(1)\CPU 312
File Edit View Insert PLC Execute Tools Window Help
PLCSIM(MPI)
CPU
SF
DP
DC
RUN
STOP
RUN-P
RUN
STOP
MRES
IB 0
IB 0
Bits
7 6 5 4 3 2 1 0
QB 0
QB 0
Bits
7 6 5 4 3 2 1 0

اليوم السابع:
١ـ فحص البرنامج اثناء العمل
ويشمل الفقرات التالية:
أـمعاينة البرنامج (Online)
بـتشخيص الاخطاء داخل البرنامج المنطقي
جـمعاينة العناوين والمتغيرات
دـتغير حالة العناوين والمتغيرات
هـمقارنة البرنامج

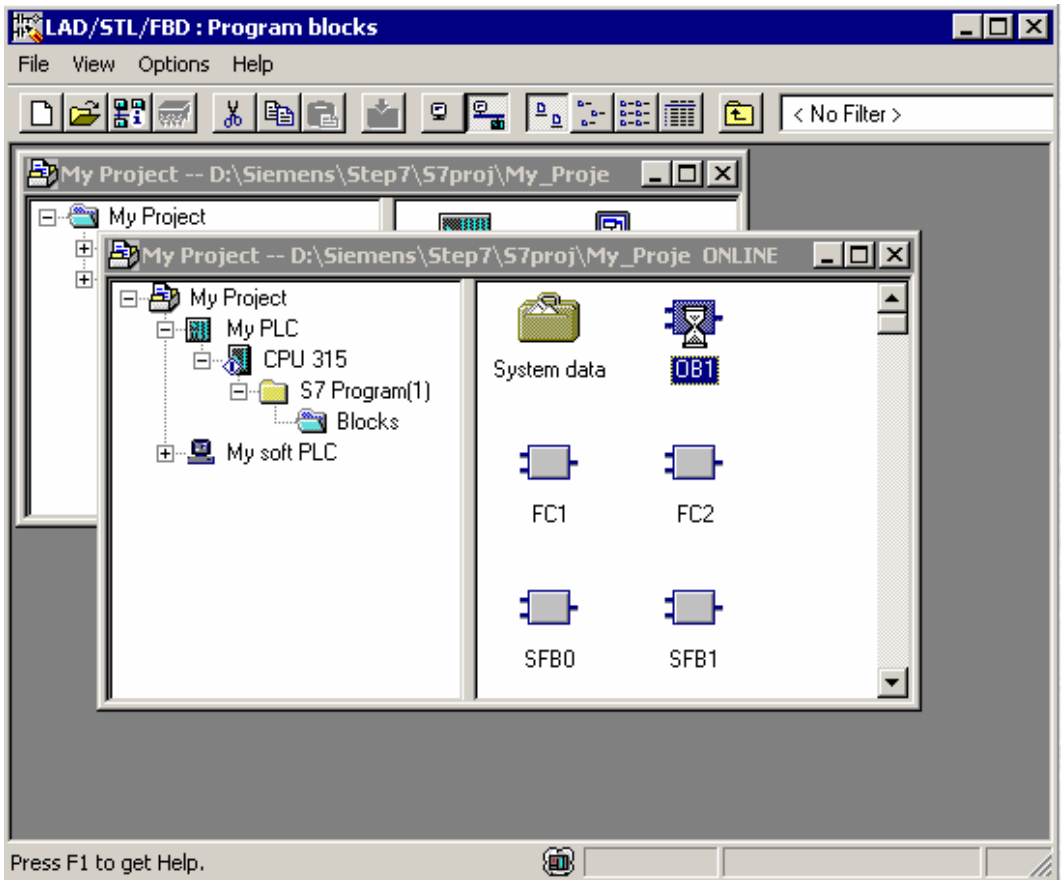
أـمعاينة البرنامج (Online)
١ـ نضغط على ايقونة (Online)



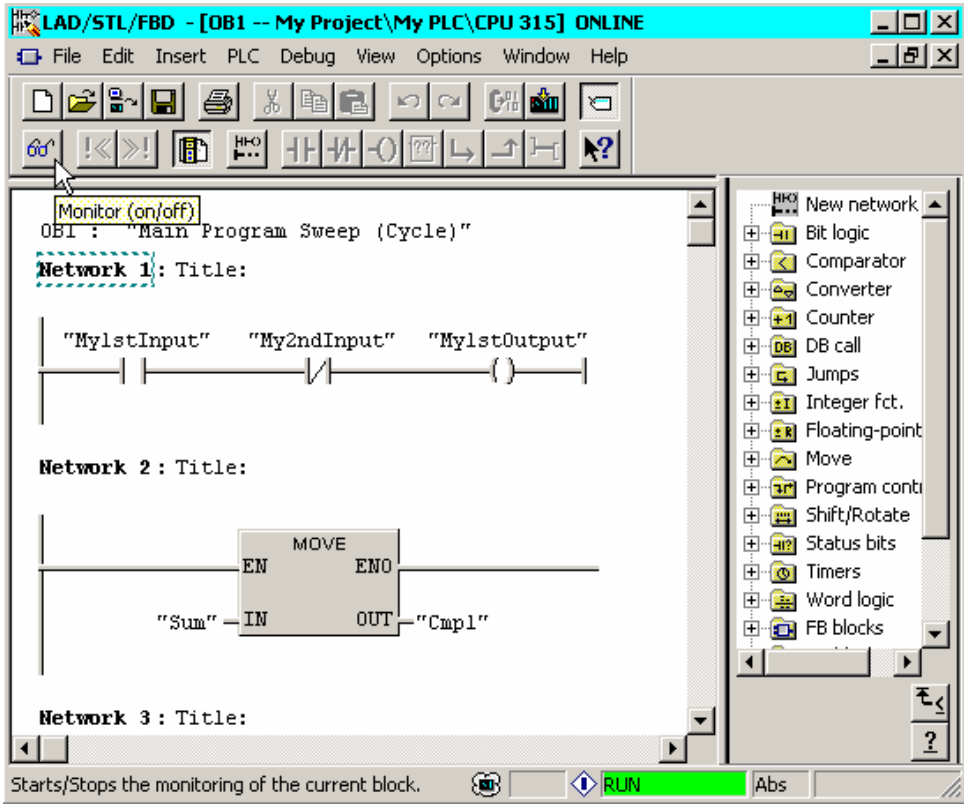
٢ـستظهر النافذة التالية



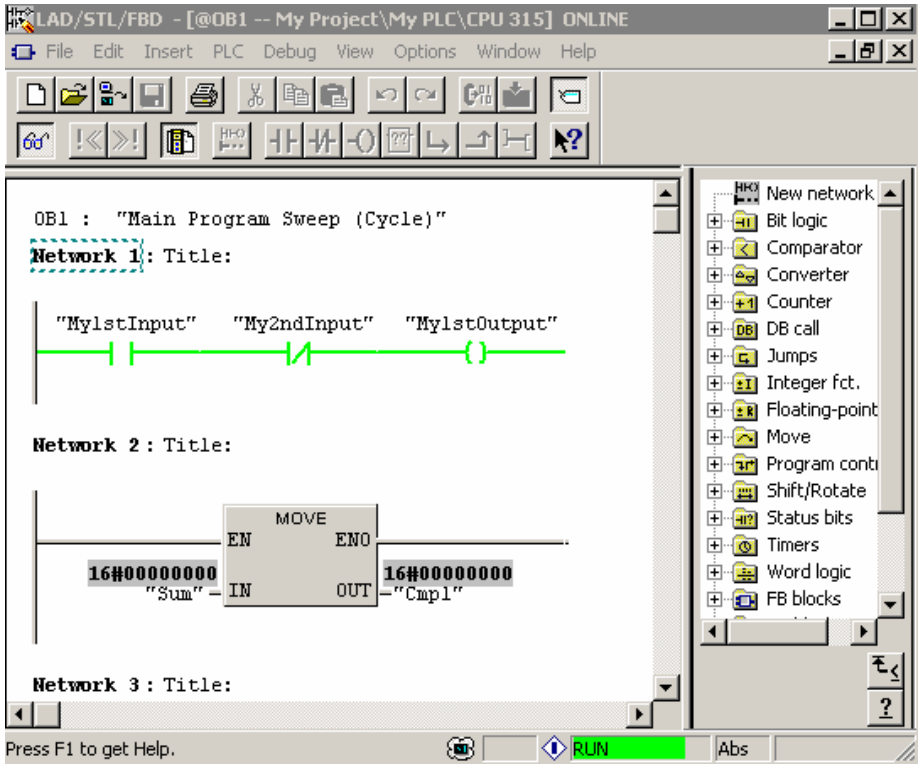
٣ـنقوم بفتح ملفات المشروع وبالنقر المزدوج على (OB1)



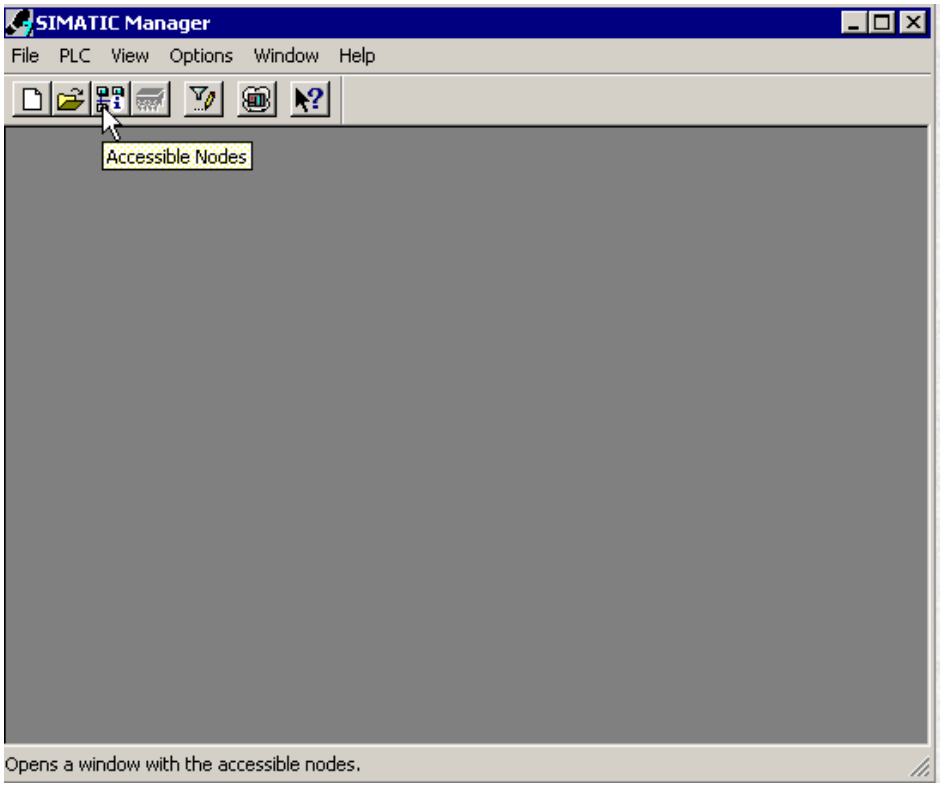
٤ـستظهر النافذة التالية:

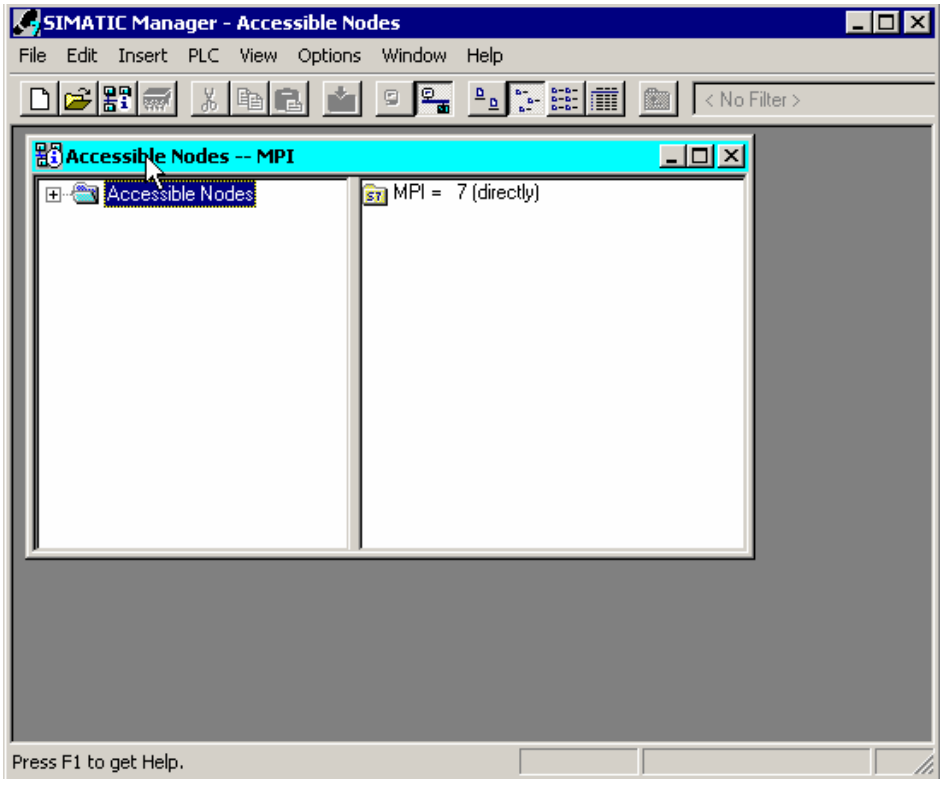


٥ـنضغط على ايقونة (Monitor) ستظهر النافذة التالية:

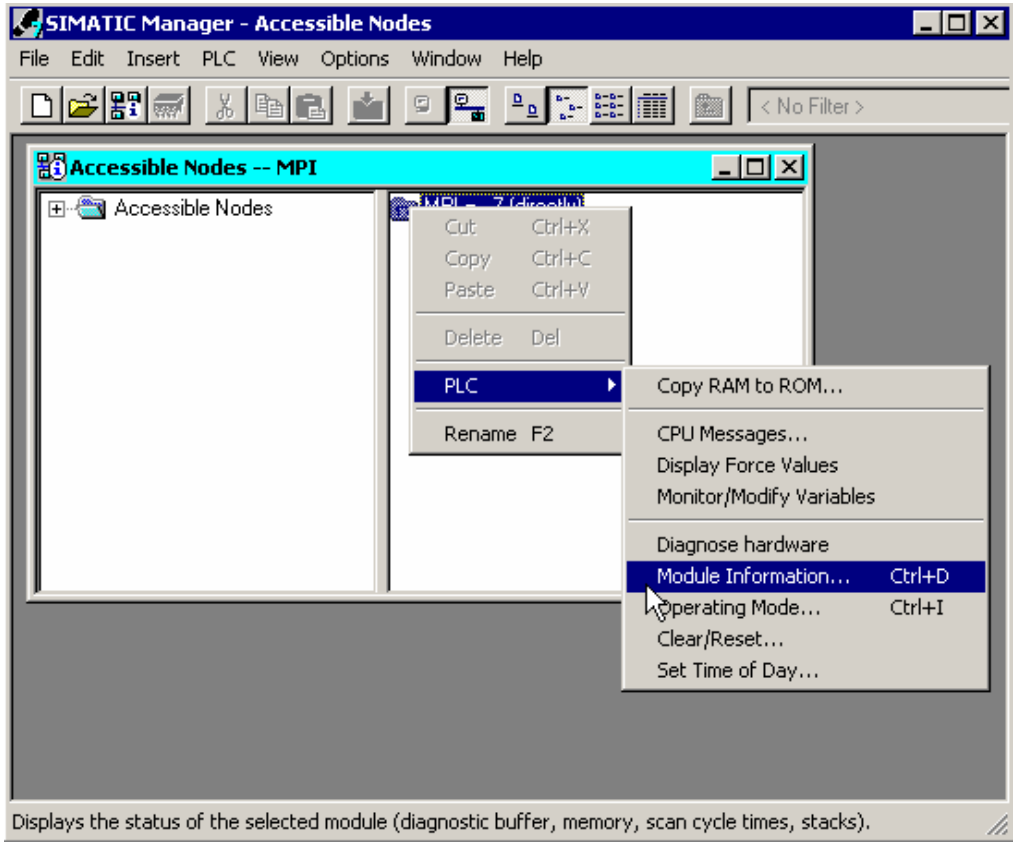


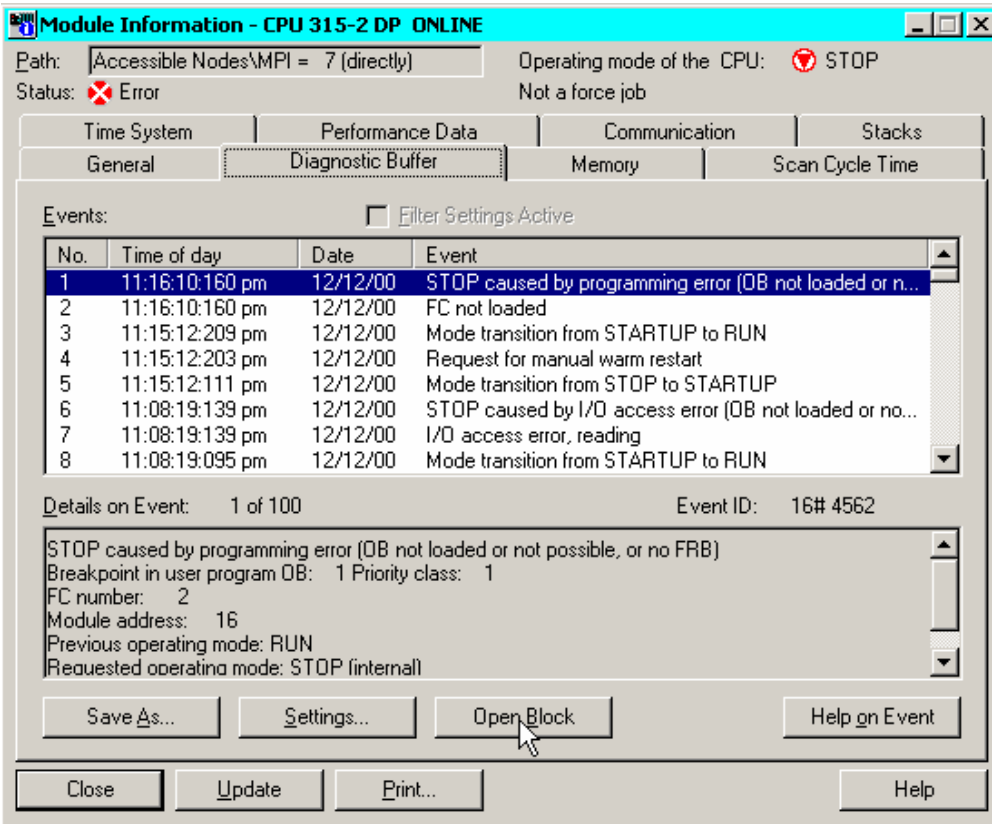
ب-تشخيص الاخطاء داخل البرنامج المنطقي

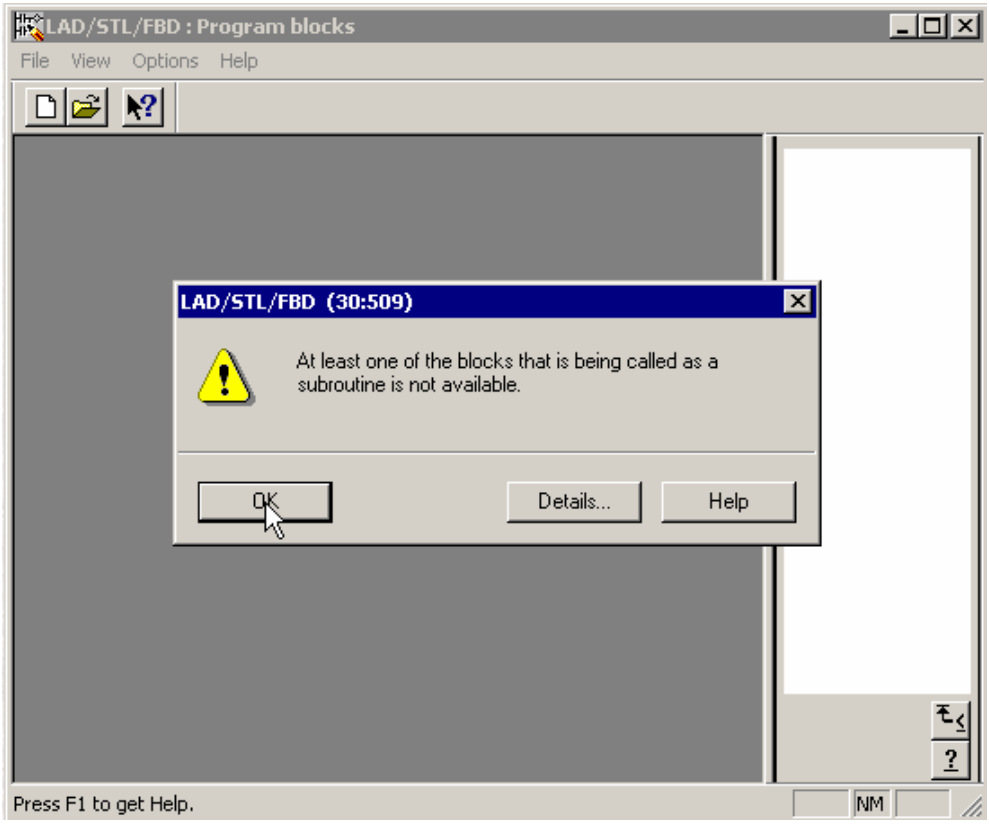


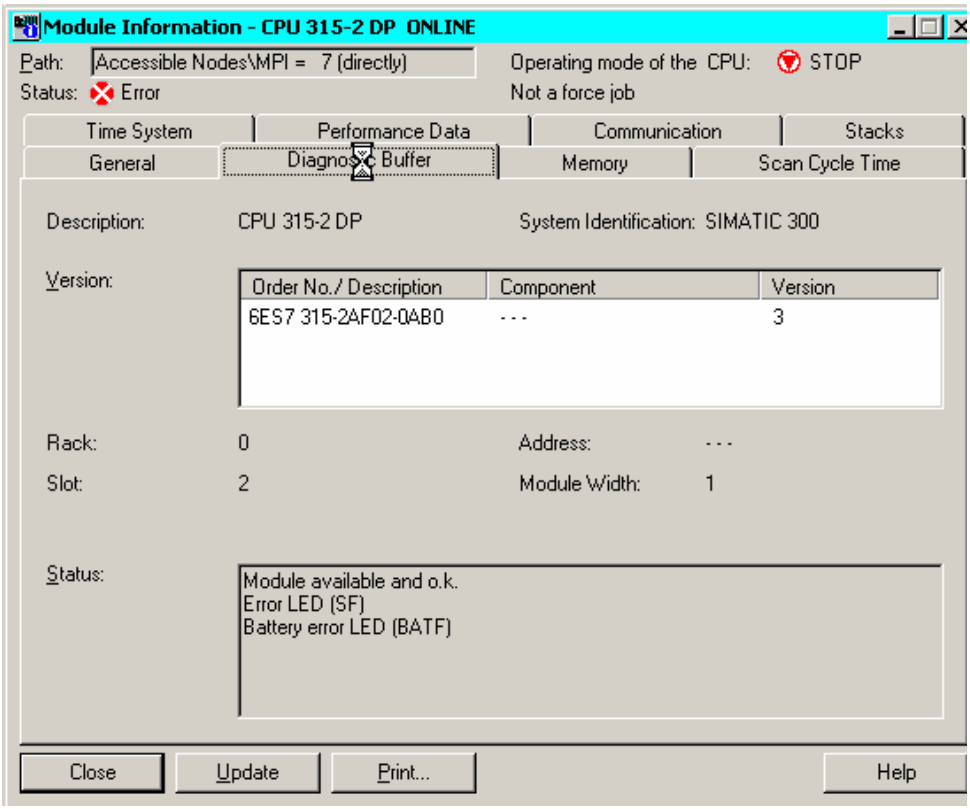


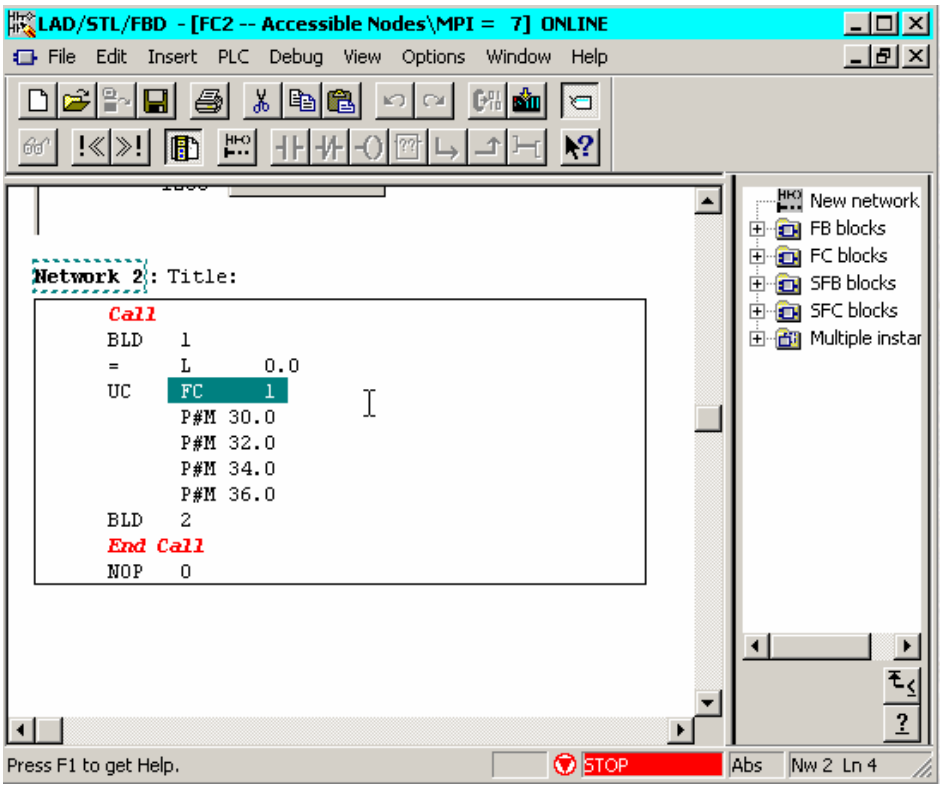
نضغط بالزر الايمن على (MPI)



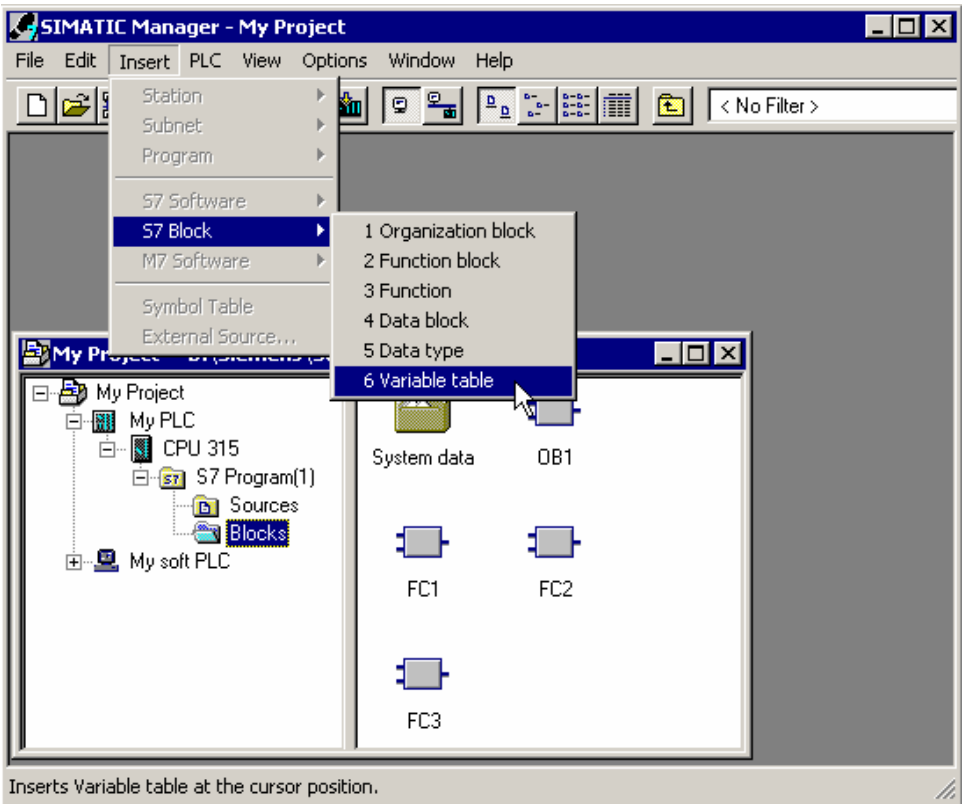


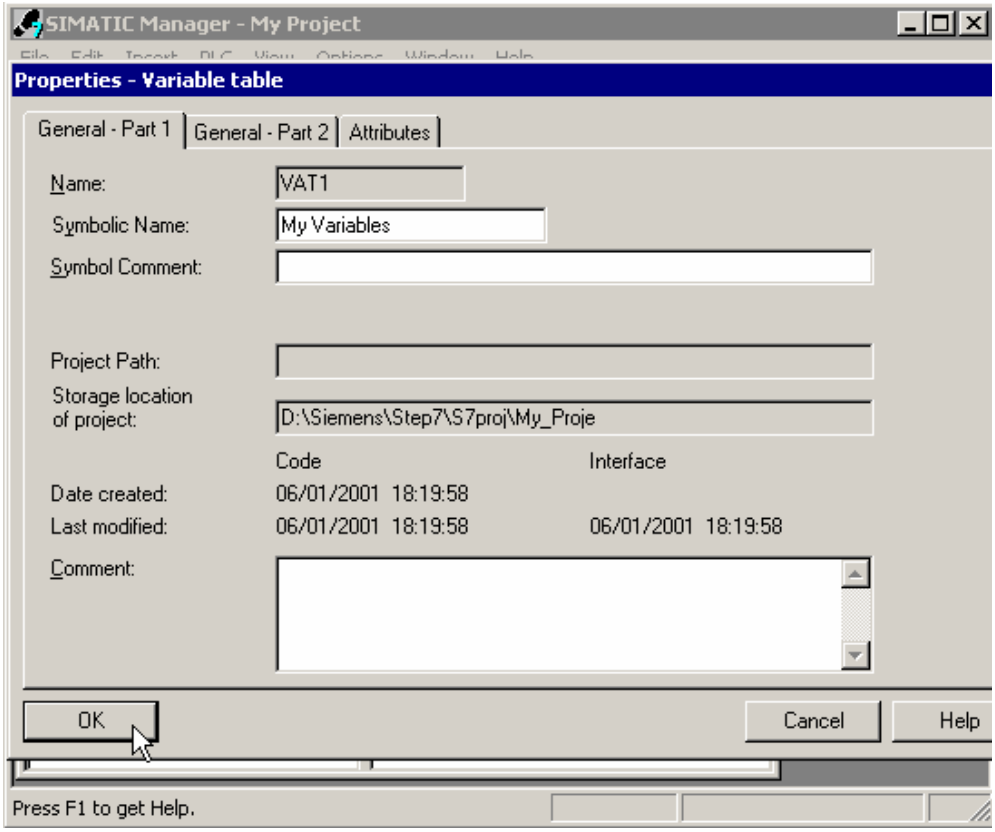

LAD/STL/FBD : Program blocks
File View Options Help
LAD/STL/FBD (30:509)
At least one of the blocks that is being called as a subroutine is not available.
OK
Details...
Help
Press F1 to get Help.
NM

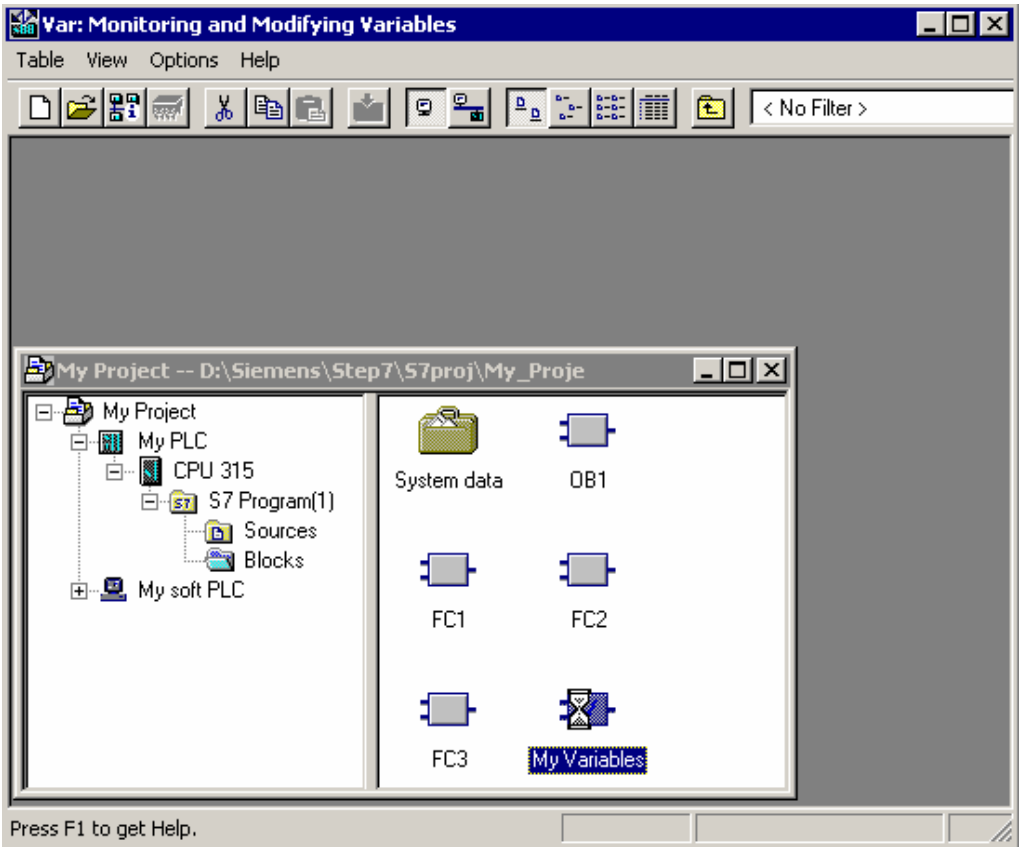

Module Information - CPU 315-2 DP ONLINE
Path: Accessible Nodes\MPI = 7 (directly)
Operating mode of the CPU: STOP
Status: Error
Not a force job
Time System
Performance Data
Communication
Stacks
General
Diagnostic Buffer
Memory
Scan Cycle Time
Description: CPU 315-2 DP
System Identification: SIMATIC 300
Version:
Order No./ Description
Component
Version
6ES7 315-2AF02-0AB0
- - -
3
Rack: 0
Address: - - -
Slot: 2
Module Width: 1
Status:
Module available and o.k.
Error LED (SF)
Battery error LED (BATF)
Close
Update
Print...
Help

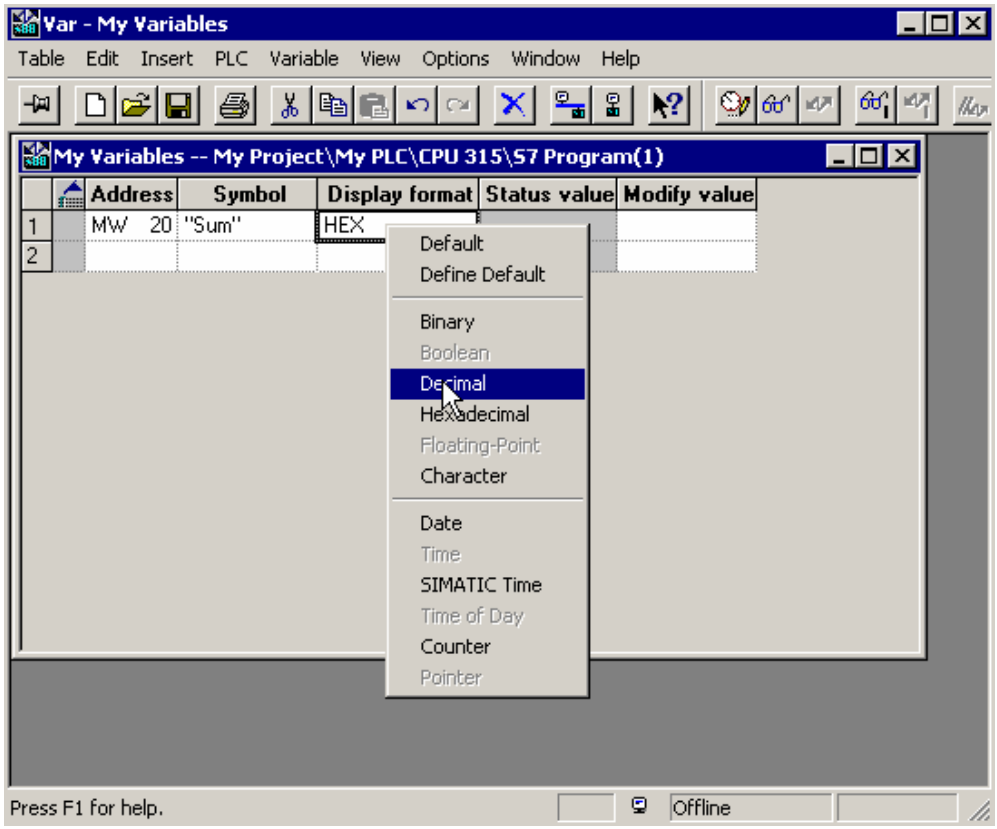


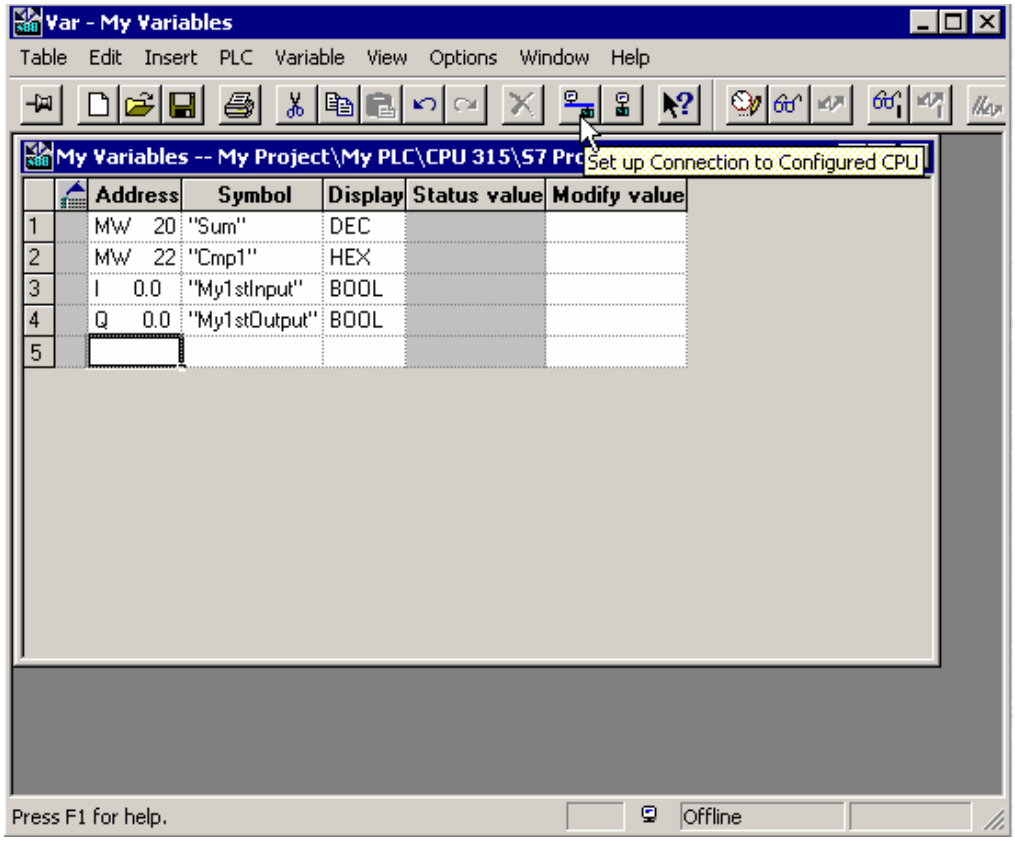
ج-معاينة العناوين والمتغيرات

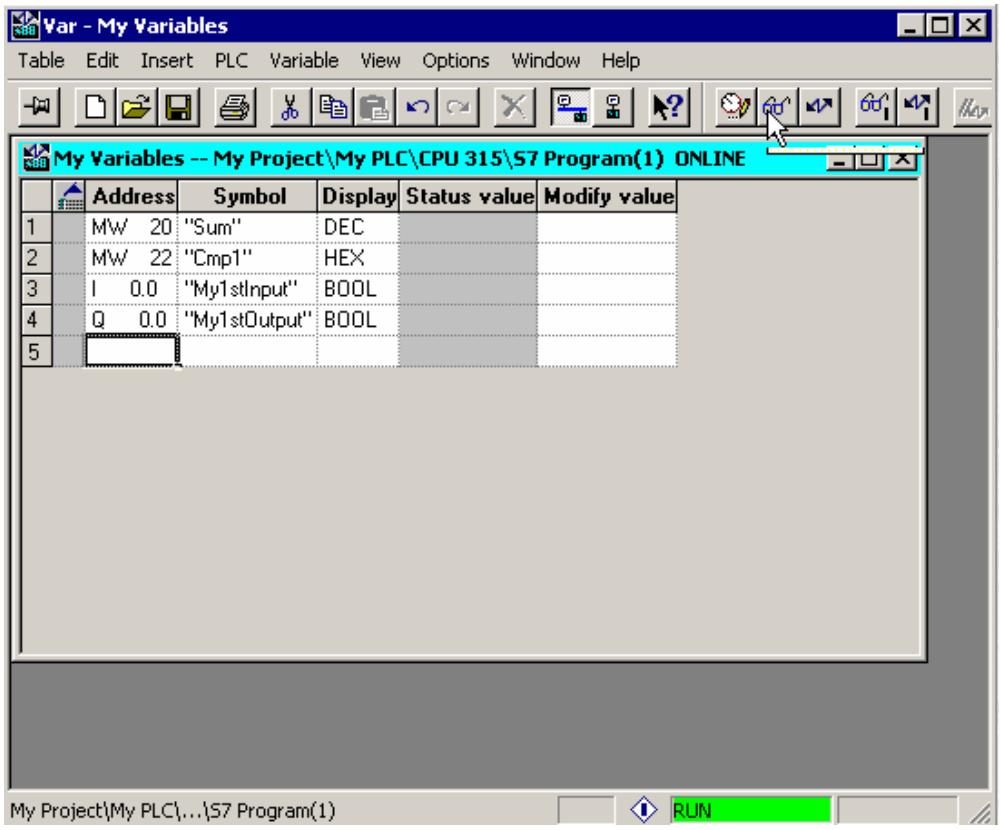


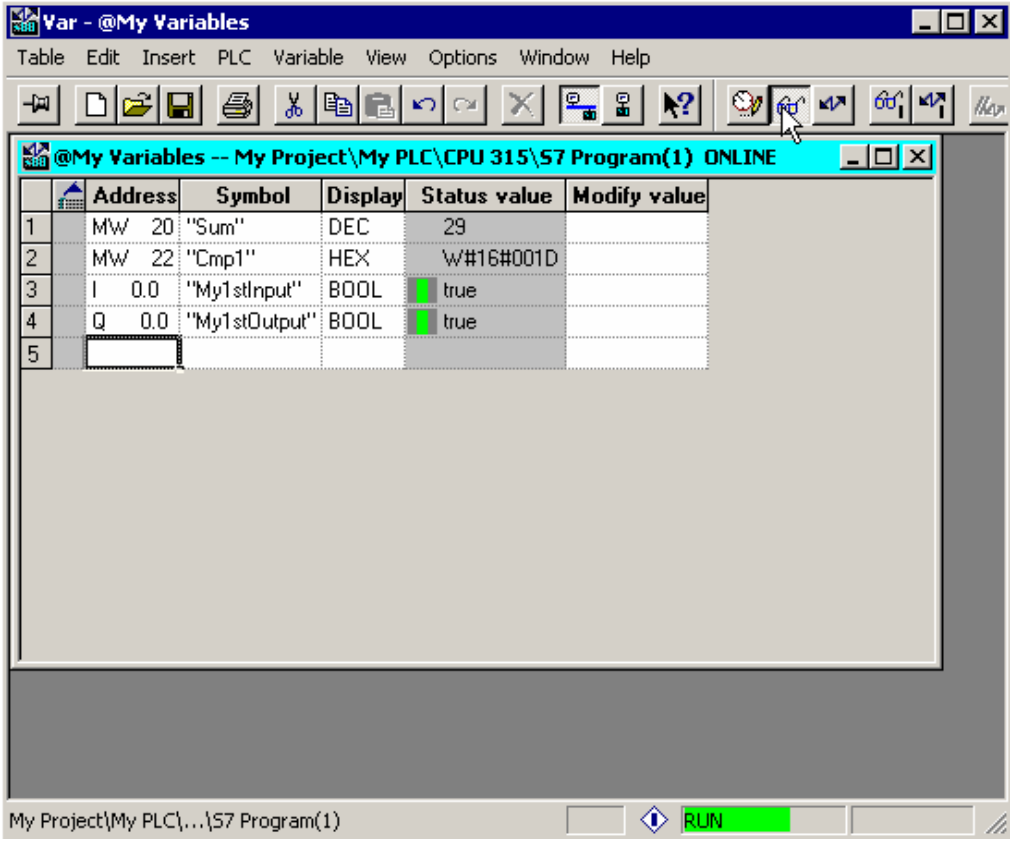

SIMATIC Manager - My Project
Properties - Variable table
General - Part 1
General - Part 2
Attributes
Name:
VAT1
Symbolic Name:
My Variables
Symbol Comment:
Project Path:
Storage location of project:
D:\Siemens\Step7\S7proj\My_Proje
Code
Interface
Date created:
06/01/2001 18:19:58
Last modified:
06/01/2001 18:19:58
06/01/2001 18:19:58
Comment:
OK
Cancel
Help
Press F1 to get Help.


Var: Monitoring and Modifying Variables
Table View Options Help
< No Filter >
My Project -- D:\Siemens\Step7\S7proj\My_Proje
My Project
My PLC
CPU 315
S7 Program(1)
Sources
Blocks
My soft PLC
System data
OB1
FC1
FC2
FC3
My Variables
Press F1 to get Help.

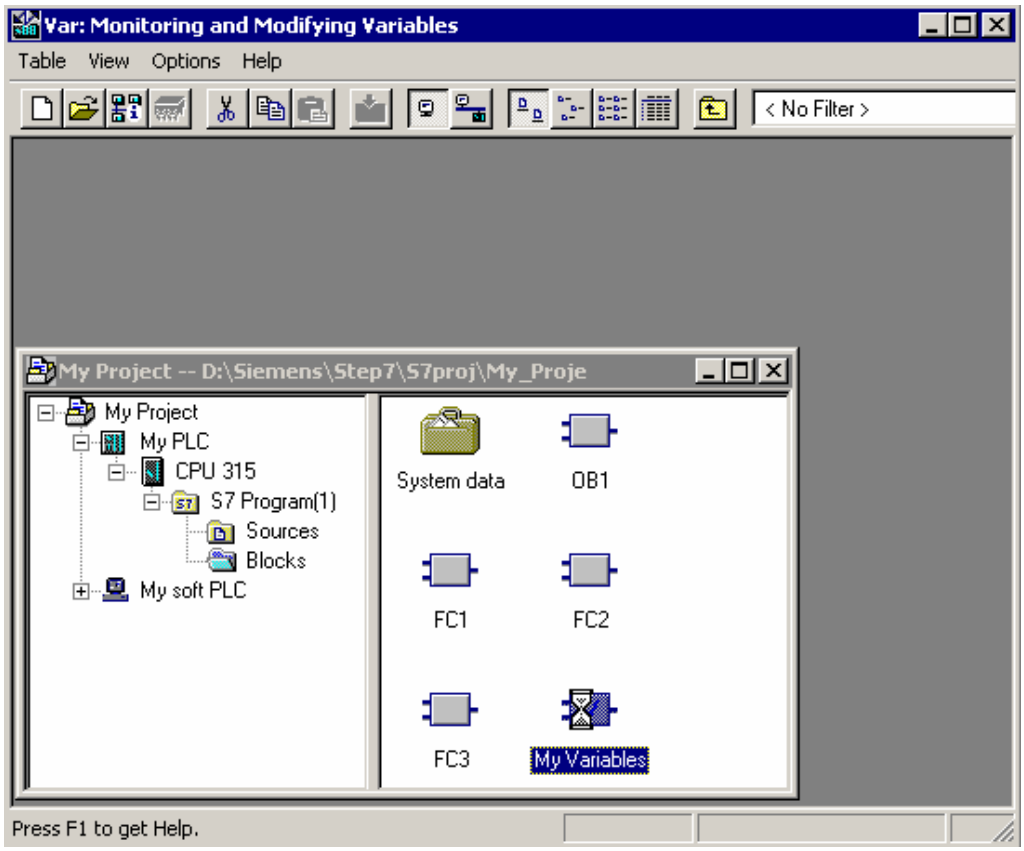

Var - My Variables
Table Edit Insert PLC Variable View Options Window Help
My Variables -- My Project\My PLC\CPU 315\S7 Program(1)
Address Symbol Display format Status value Modify value
1 MW 20 "Sum" HEX
2
Default
Define Default
Binary
Boolean
Decimal
Hexadecimal
Floating-Point
Character
Date
Time
SIMATIC Time
Time of Day
Counter
Pointer
Press F1 for help.
Offline

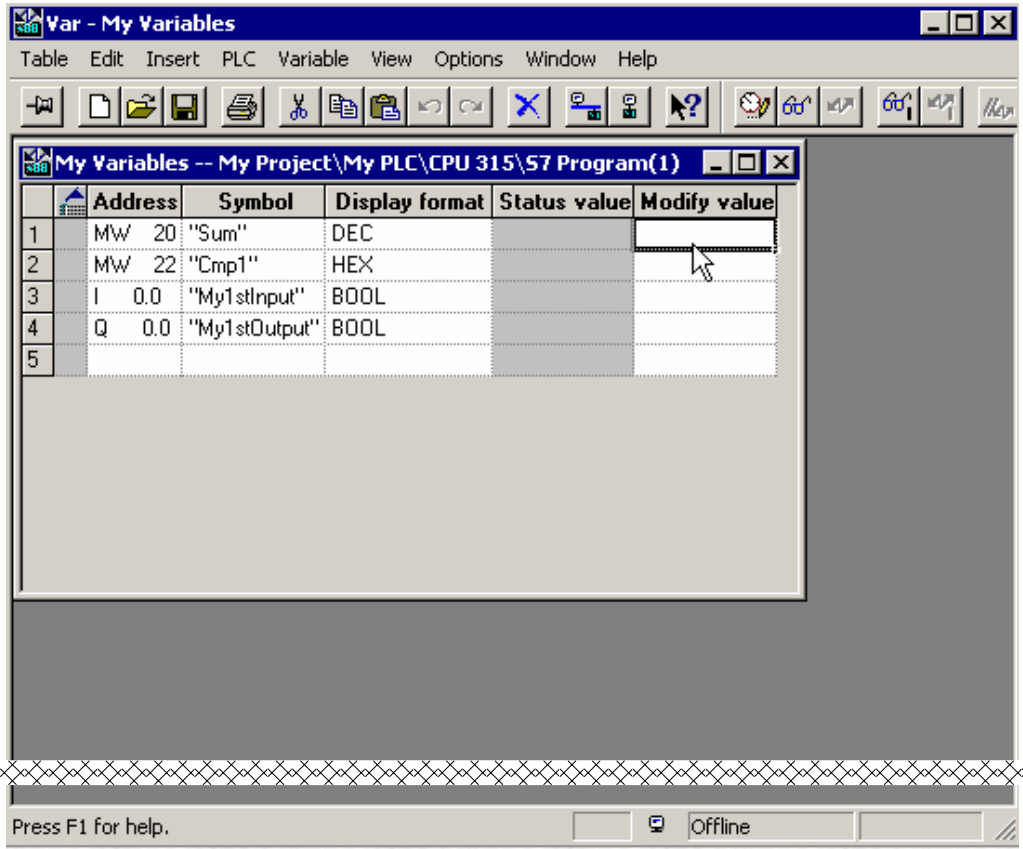

Var - My Variables
Table Edit Insert PLC Variable View Options Window Help
My Variables -- My Project\My PLC\CPU 315\S7 Pro
Set up Connection to Configured CPU
Address Symbol Display Status value Modify value
1 MW 20 "Sum" DEC
2 MW 22 "Cmp1" HEX
3 I 0.0 "My1stInput" BOOL
4 Q 0.0 "My1stOutput" BOOL
5
Press F1 for help.
Offline

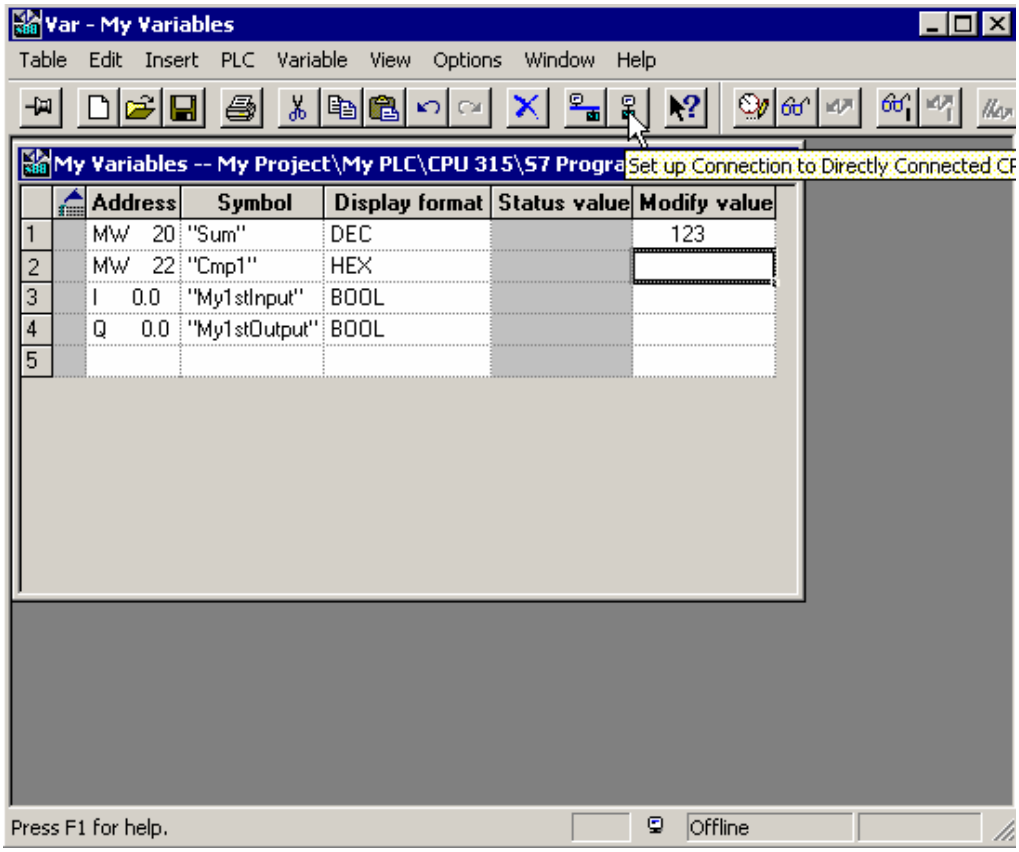

Var - My Variables
Table Edit Insert PLC Variable View Options Window Help
My Variables -- My Project\My PLC\CPU 315\S7 Program(1) ONLINE
Address Symbol Display Status value Modify value
1 MW 20 "Sum" DEC
2 MW 22 "Cmp1" HEX
3 I 0.0 "My1stInput" BOOL
4 Q 0.0 "My1stOutput" BOOL
5
My Project\My PLC\...\S7 Program(1)
RUN

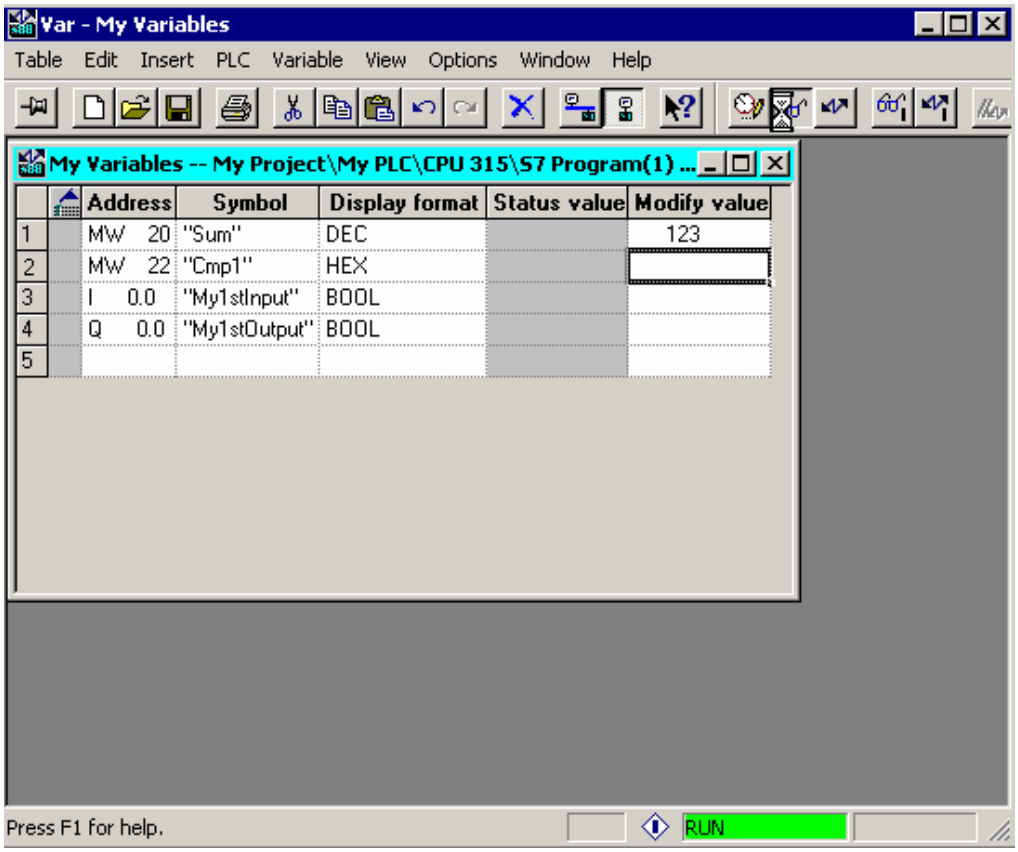

Var - @My Variables
Table Edit Insert PLC Variable View Options Window Help
@My Variables -- My Project\My PLC\CPU 315\S7 Program(1) ONLINE
Address Symbol Display Status value Modify value
1 MW 20 "Sum" DEC 29
2 MW 22 "Cmp1" HEX W#16#001D
3 I 0.0 "My1stInput" BOOL true
4 Q 0.0 "My1stOutput" BOOL true
5
My Project\My PLC\...\S7 Program(1)
RUN

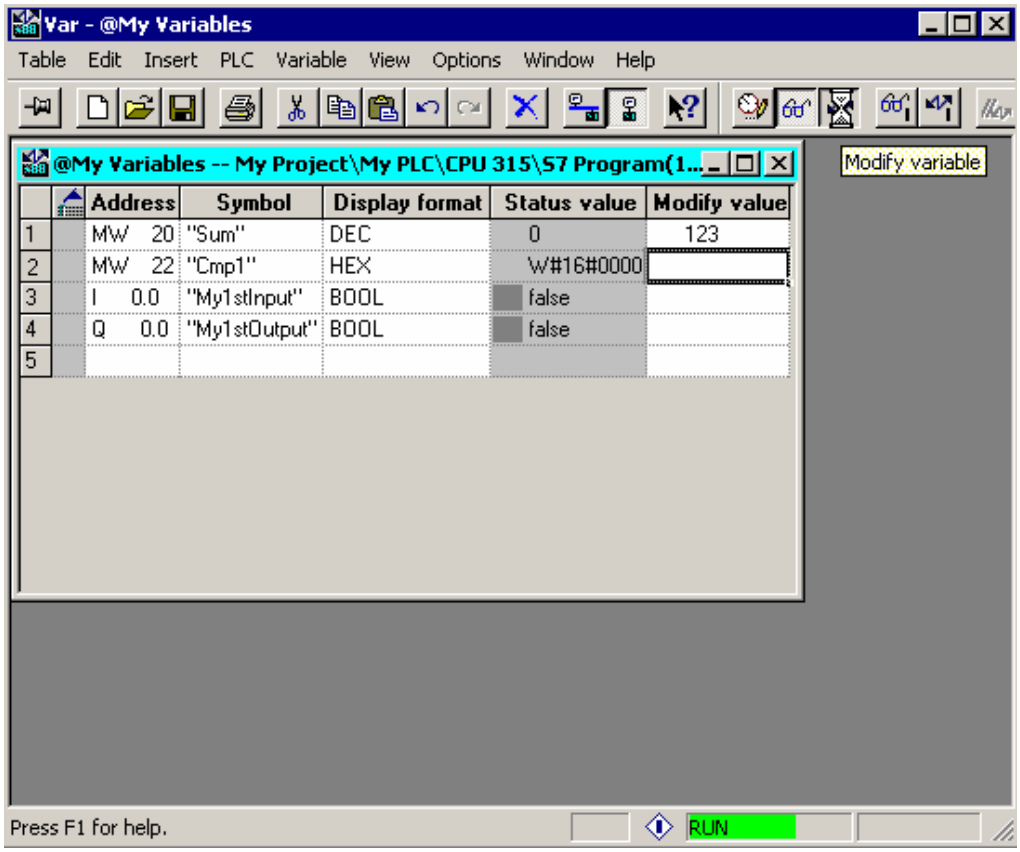
د-تغير حالة العناوين والمتغيرات

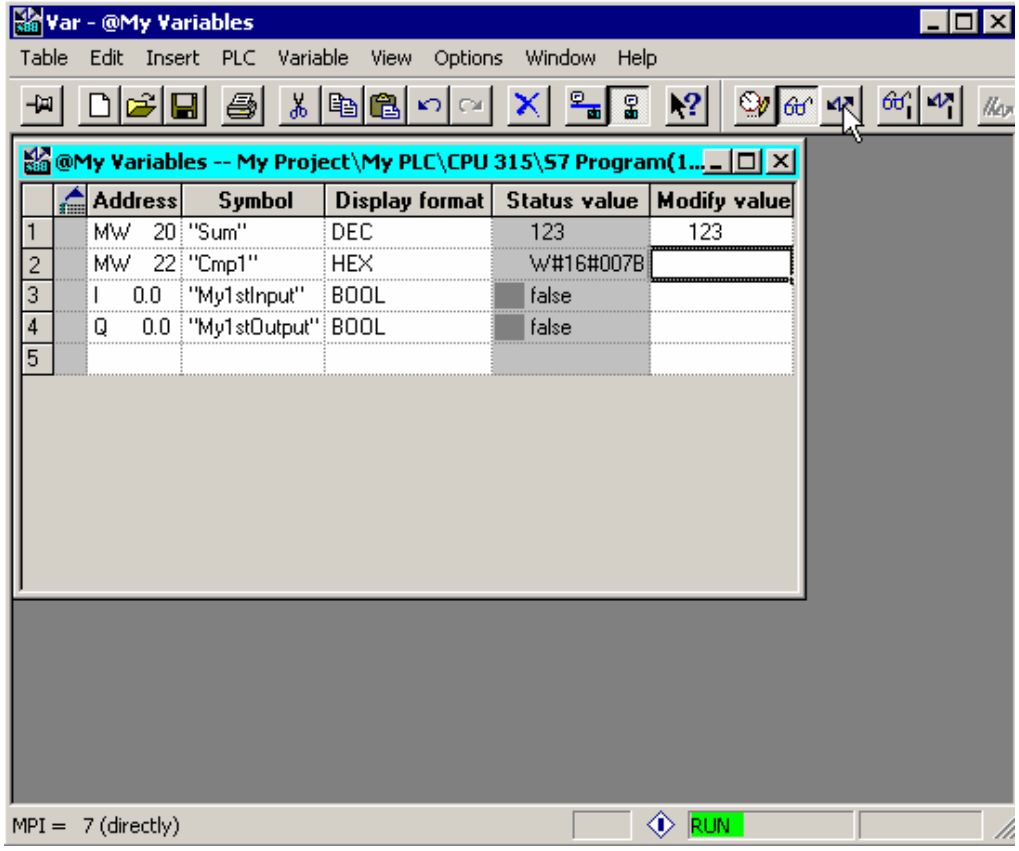





Var - My Variables
Table Edit Insert PLC Variable View Options Window Help
My Variables -- My Project\My PLC\CPU 315\S7 Progra
Set up Connection to Directly Connected CP
Address Symbol Display format Status value Modify value
1 MW 20 "Sum" DEC 123
2 MW 22 "Cmp1" HEX
3 I 0.0 "My1stInput" BOOL
4 Q 0.0 "My1stOutput" BOOL
5
Press F1 for help.
Offline

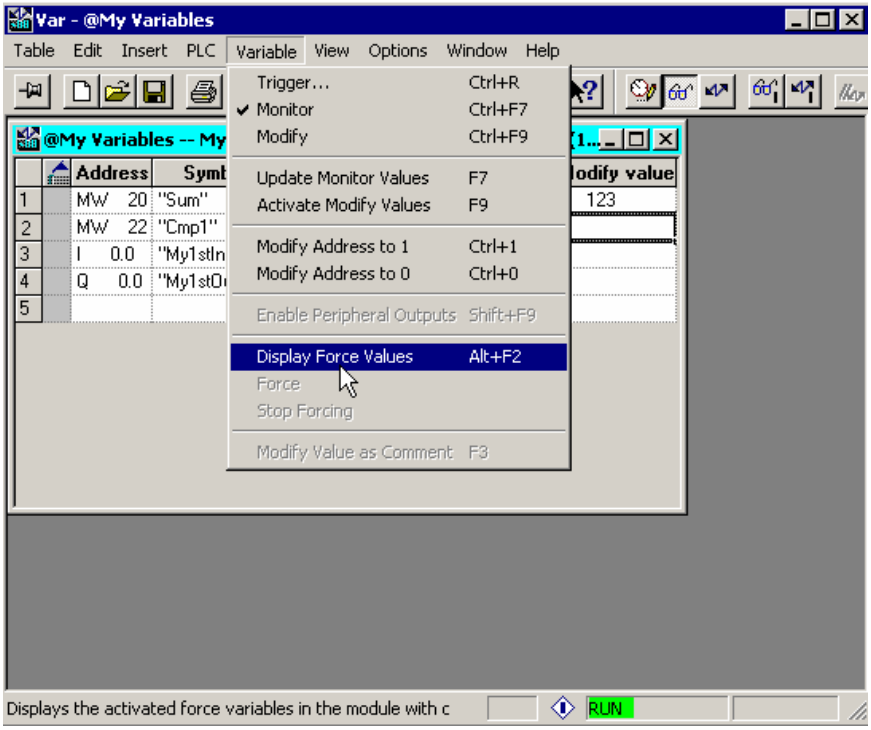

Var - My Variables
Table Edit Insert PLC Variable View Options Window Help
My Variables -- My Project\My PLC\CPU 315\S7 Program(1) ...
Address Symbol Display format Status value Modify value
1 MW 20 "Sum" DEC 123
2 MW 22 "Cmp1" HEX
3 I 0.0 "My1stInput" BOOL
4 Q 0.0 "My1stOutput" BOOL
5
Press F1 for help.
RUN

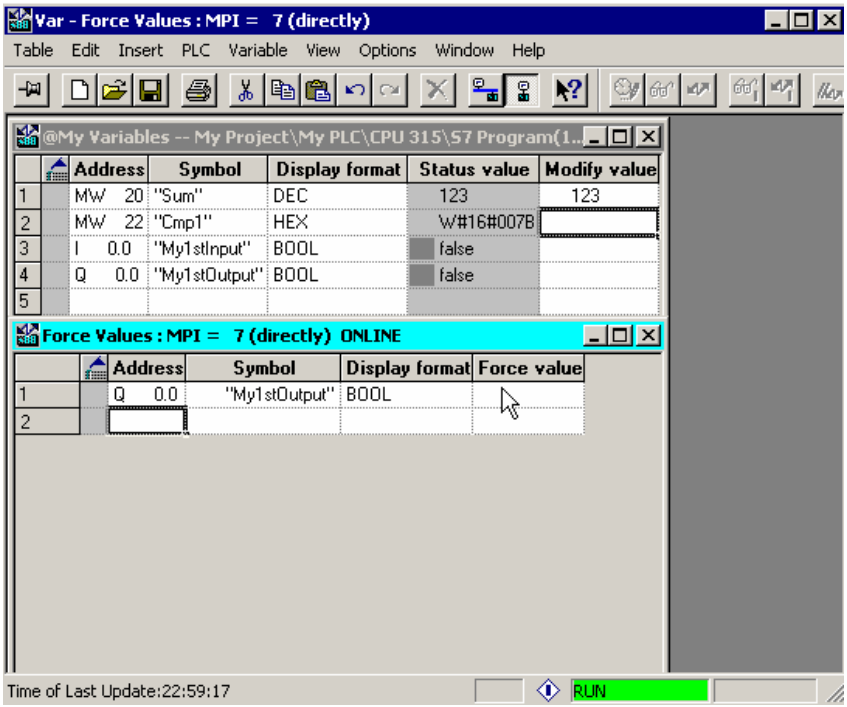

Var - @My Variables
Table Edit Insert PLC Variable View Options Window Help
@My Variables -- My Project\My PLC\CPU 315\S7 Program(1...
Modify variable
Address Symbol Display format Status value Modify value
1 MW 20 "Sum" DEC 0 123
2 MW 22 "Cmp1" HEX W#16#0000
3 I 0.0 "My1stInput" BOOL false
4 Q 0.0 "My1stOutput" BOOL false
5
Press F1 for help.
RUN


Var - @My Variables
Table Edit Insert PLC Variable View Options Window Help
@My Variables -- My Project\My PLC\CPU 315\S7 Program(1...
Address Symbol Display format Status value Modify value
1 MW 20 "Sum" DEC 123 123
2 MW 22 "Cmp1" HEX W#16#007B
3 I 0.0 "My1stInput" BOOL false
4 Q 0.0 "My1stOutput" BOOL false
5
MPI = 7 (directly)
RUN

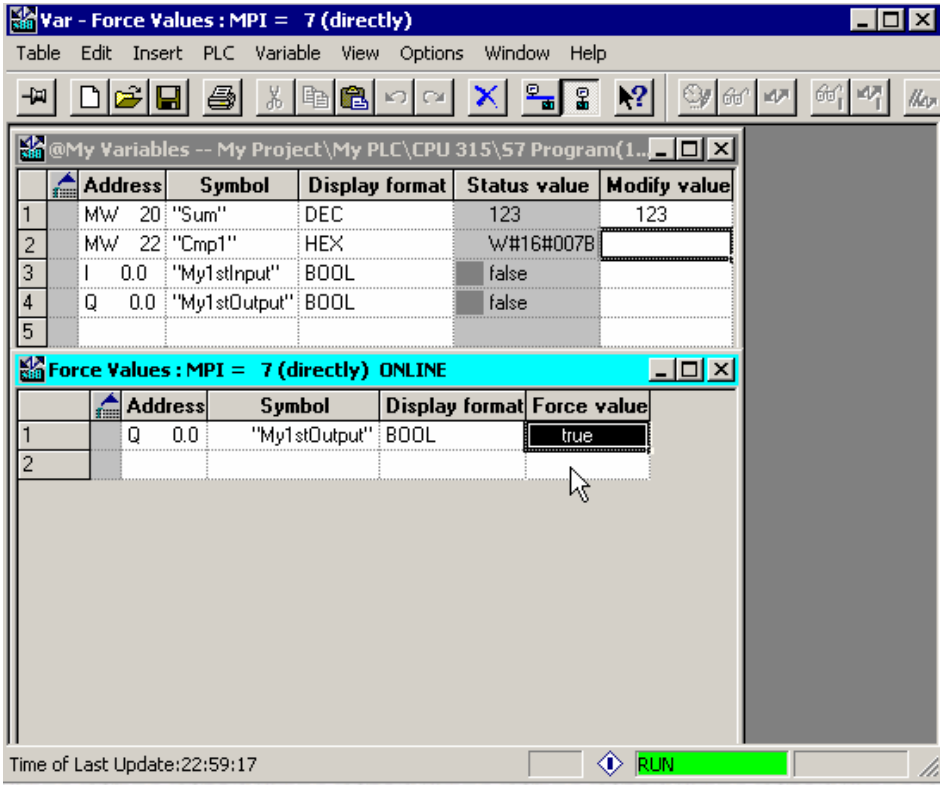
لاحظ انه تم تغير قيمة العدد والان سنتعلم كيفية عمل (Force) للعناوين والفرق بين (Force) و(Modify) هو انه ال (Force) يغير العناوين من نوع (Bool) وغيرها ولا تثبت قيمة التغيير في العناوين المتغيرة لحظيا فقط بالعناوين الثابتة القيمة

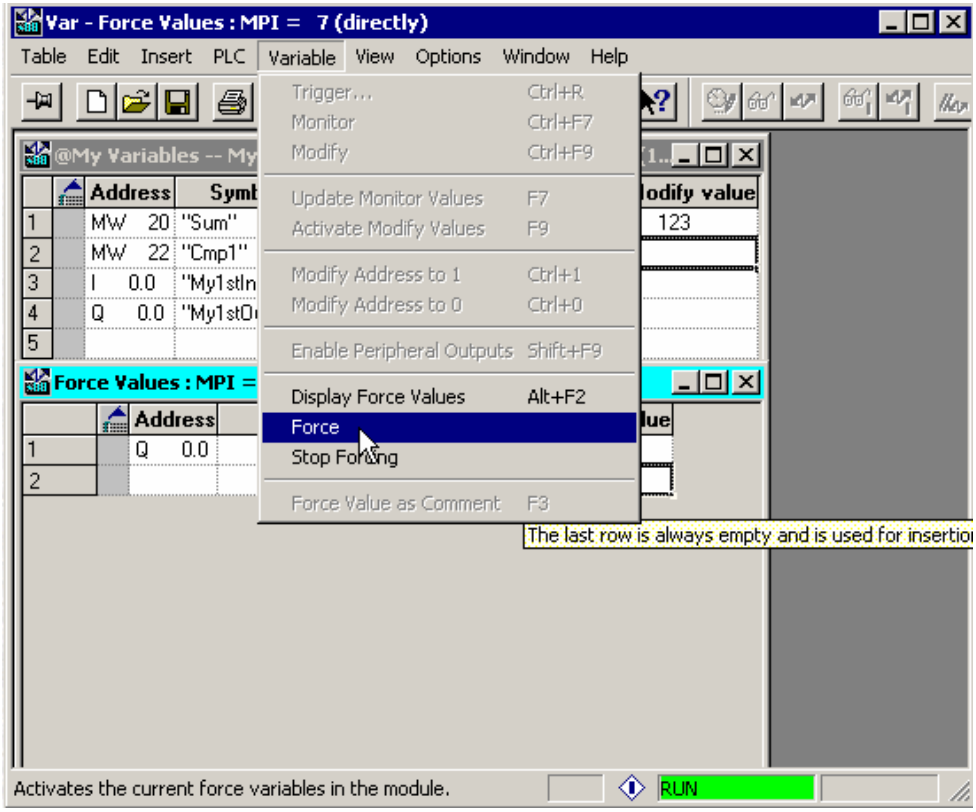
اما ال (Force) فيتعامل مع عناويين من نوع (Bool) فقط ويثبت قيمة العنوان بصورة دائمية لحين رفع عملية ال (Force)

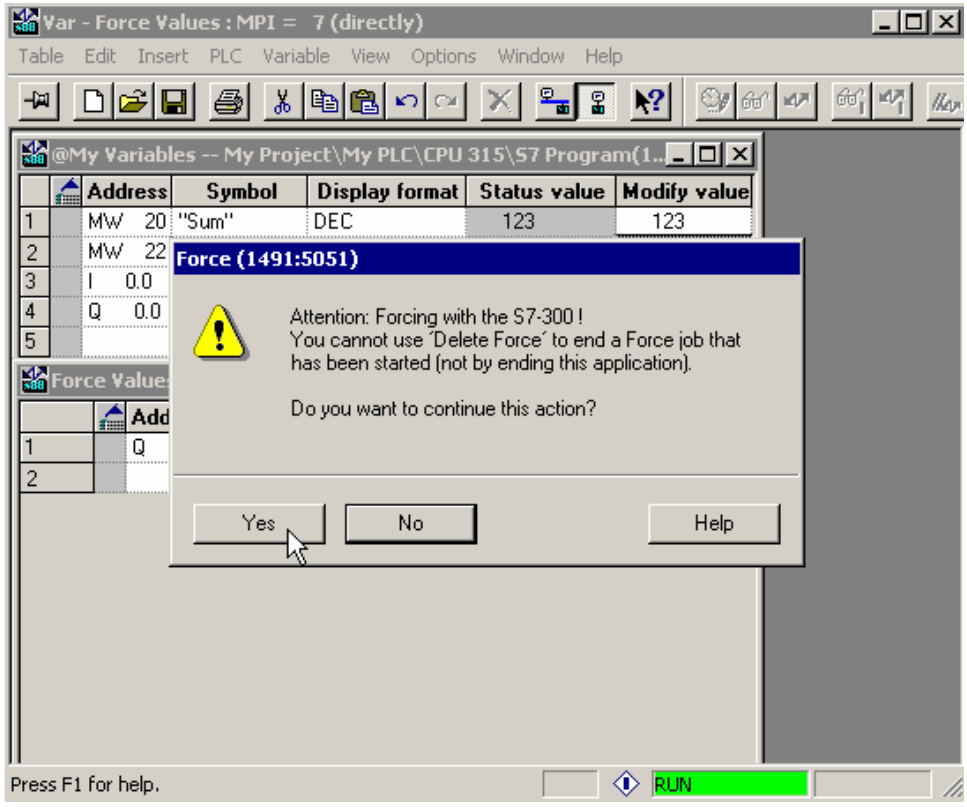


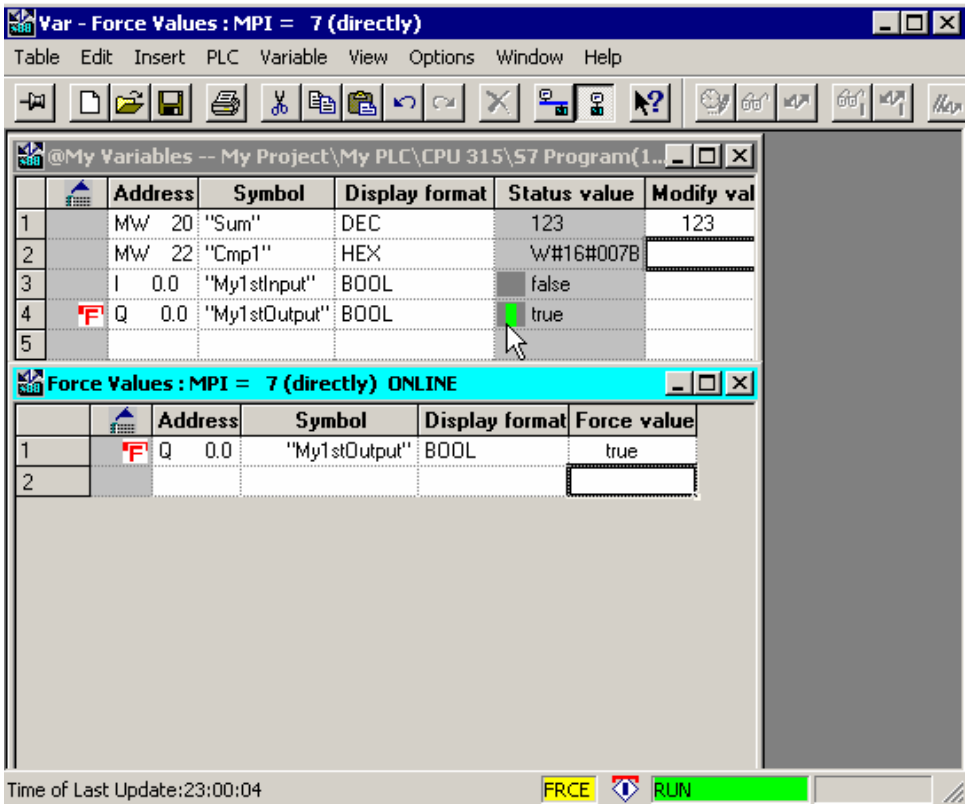


نكتب القيمة المراد التغيير اليها

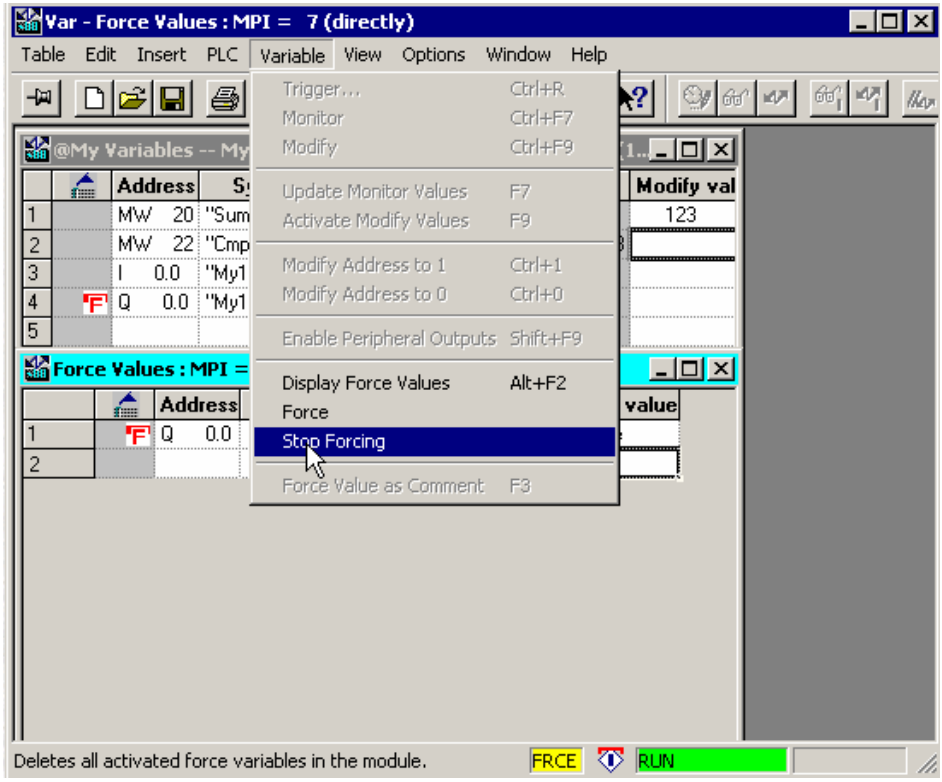


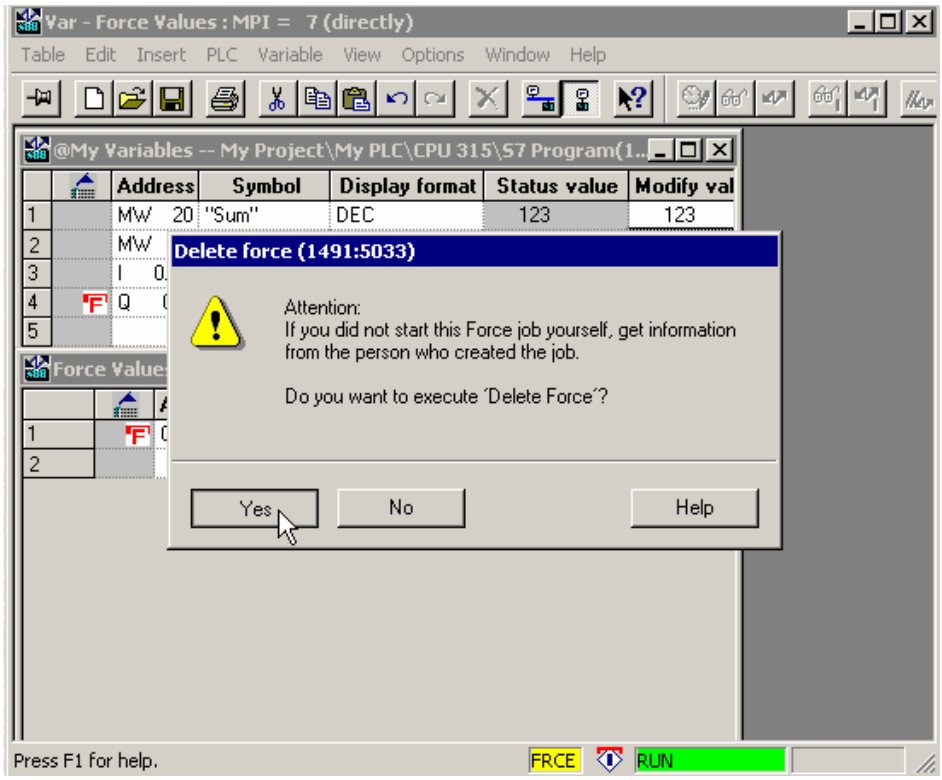


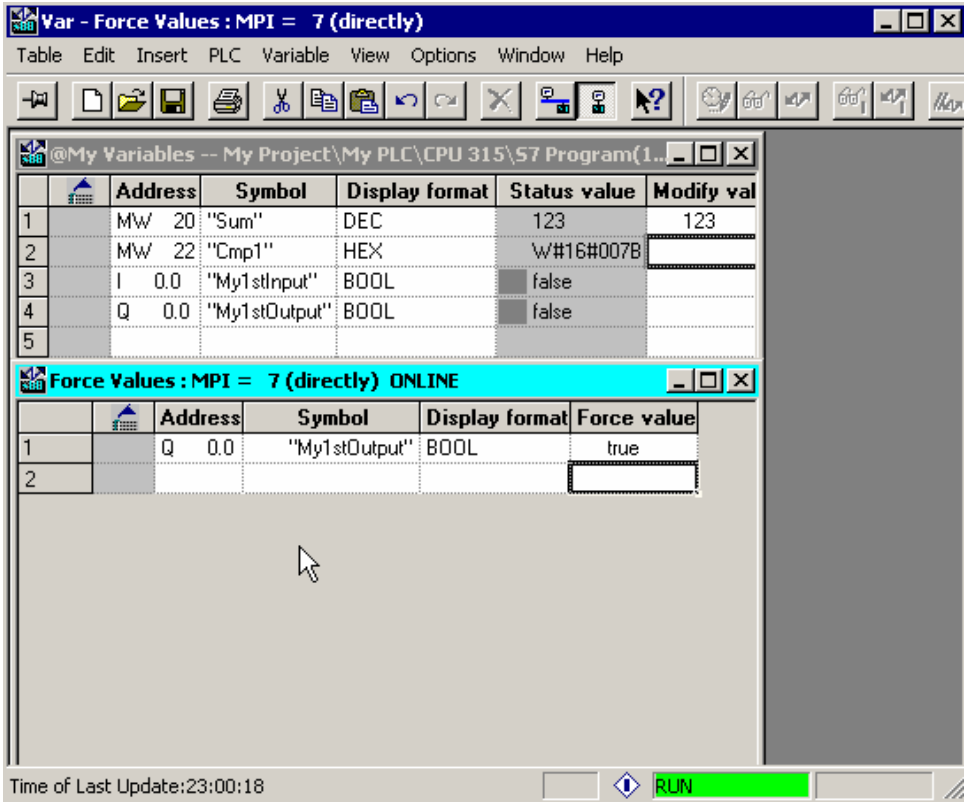

Var - Force Values : MPI = 7 (directly)
Table Edit Insert PLC Variable View Options Window Help
@My Variables -- My Project\My PLC\CPU 315\S7 Program(1...
Address Symbol Display format Status value Modify value
1 MW 20 "Sum" DEC 123 123
2 MW 22
3 I 0.0
4 Q 0.0
5
Force Values
Add
1 Q
2
Force (1491:5051)
Attention: Forcing with the S7-300 !
You cannot use 'Delete Force' to end a Force job that has been started (not by ending this application).
Do you want to continue this action?
Yes
No
Help
Press F1 for help.
RUN


Var - Force Values : MPI = 7 (directly)
Table Edit Insert PLC Variable View Options Window Help
@My Variables -- My Project\My PLC\CPU 315\S7 Program(1...
Address Symbol Display format Status value Modify val
1 MW 20 "Sum" DEC 123 123
2 MW 22 "Cmp1" HEX W#16#007B
3 I 0.0 "My1stInput" BOOL false
4 Q 0.0 "My1stOutput" BOOL true
5
Force Values : MPI = 7 (directly) ONLINE
Address Symbol Display format Force value
1 Q 0.0 "My1stOutput" BOOL true
2
Time of Last Update:23:00:04
FRCE
RUN

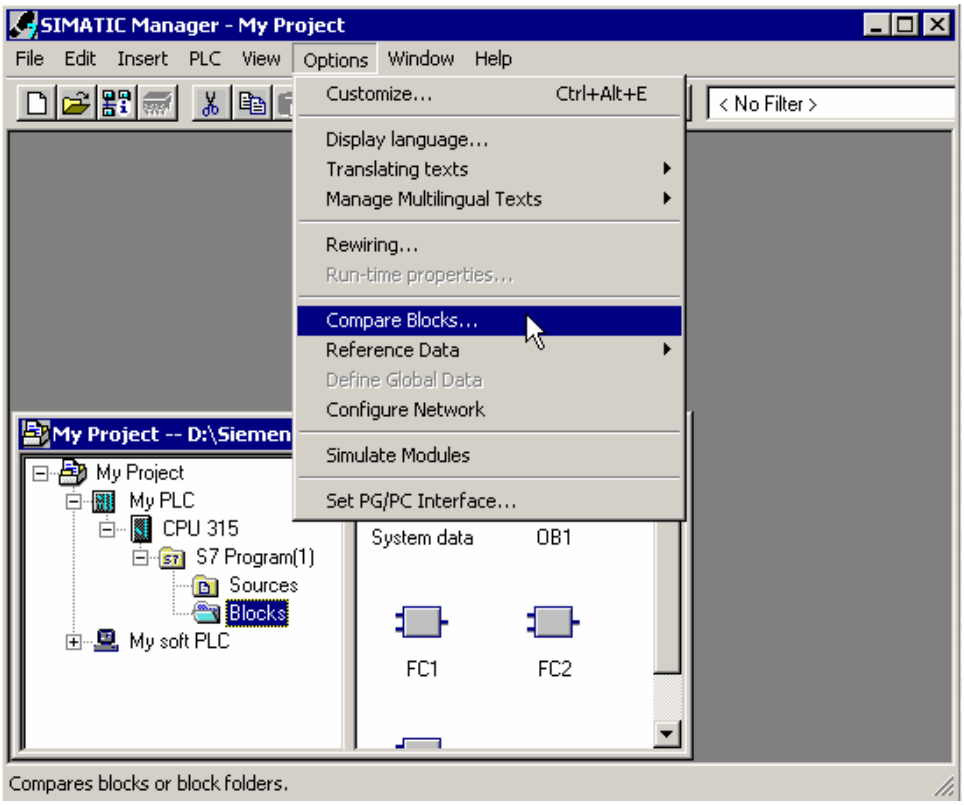
لالغاء عملية (Force) كالتالي:

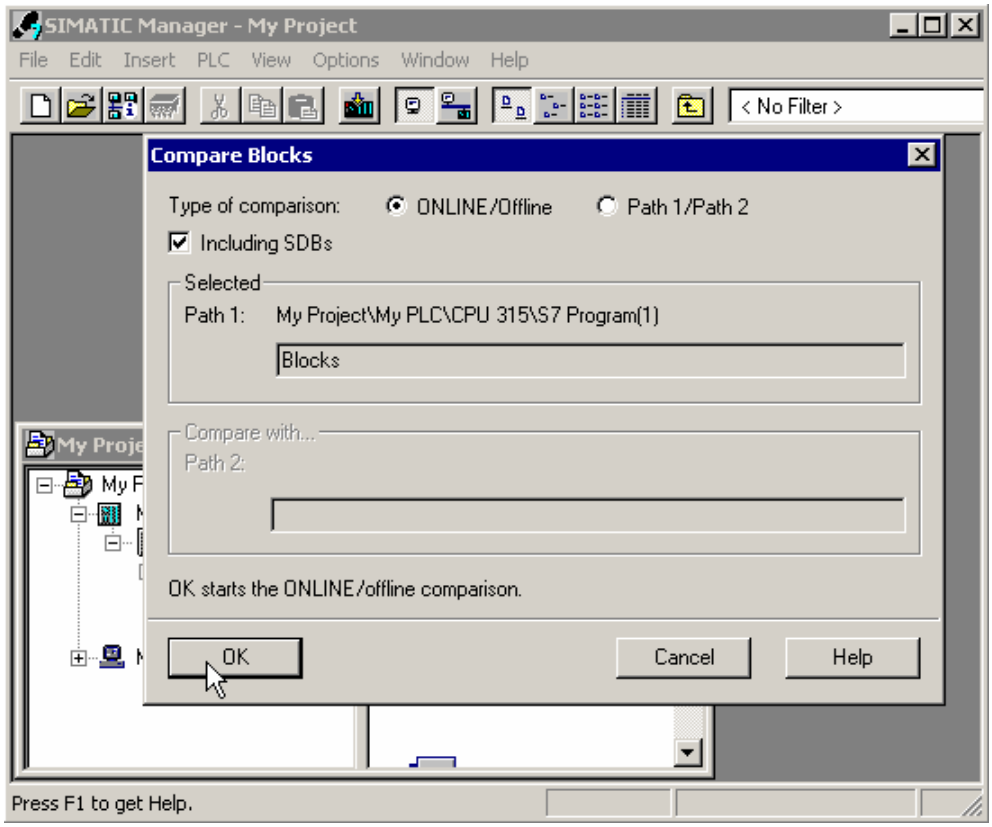


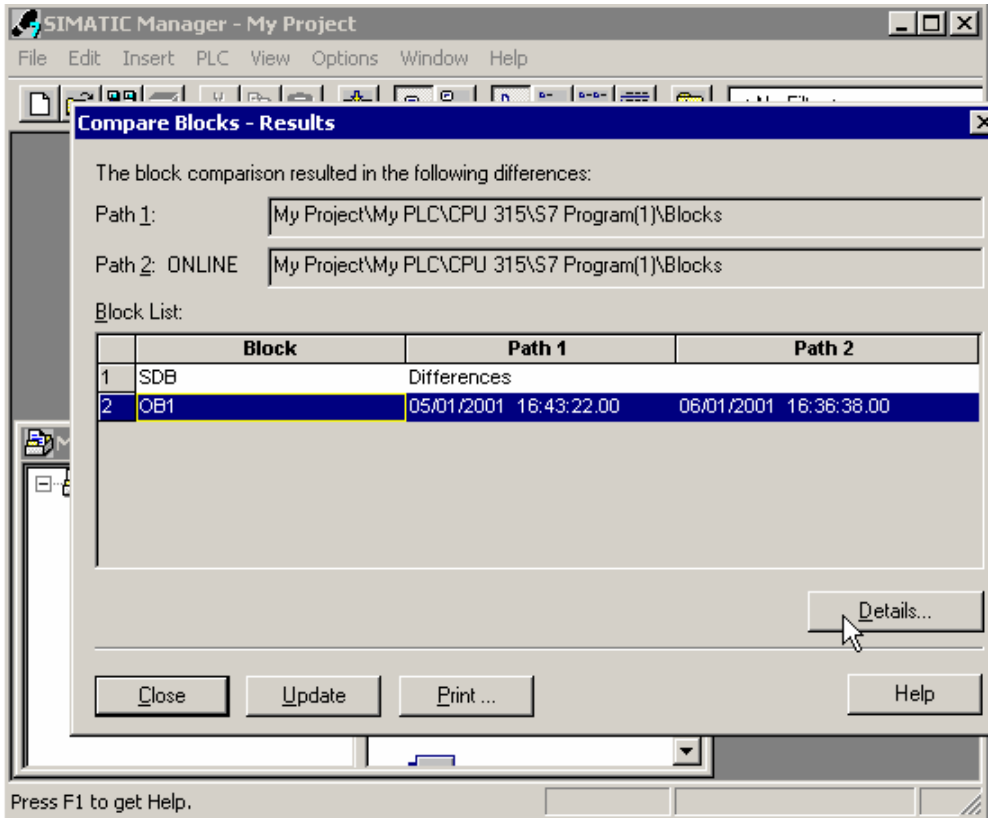


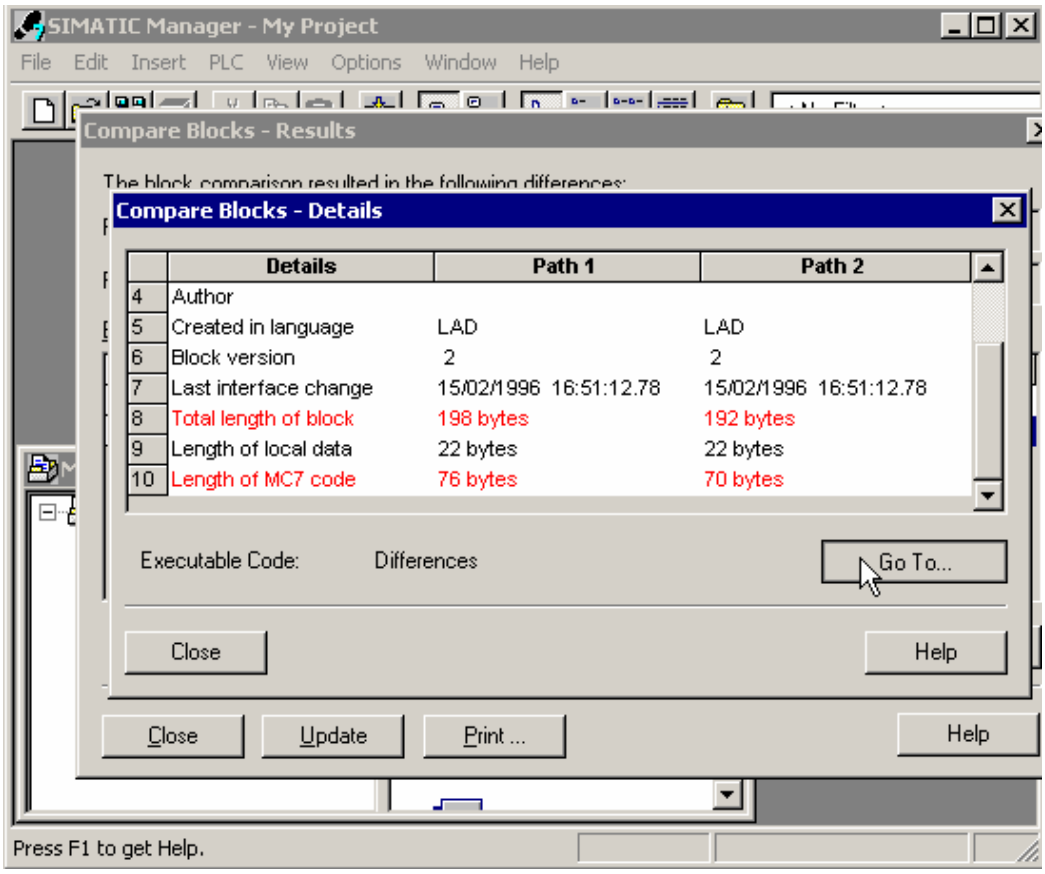


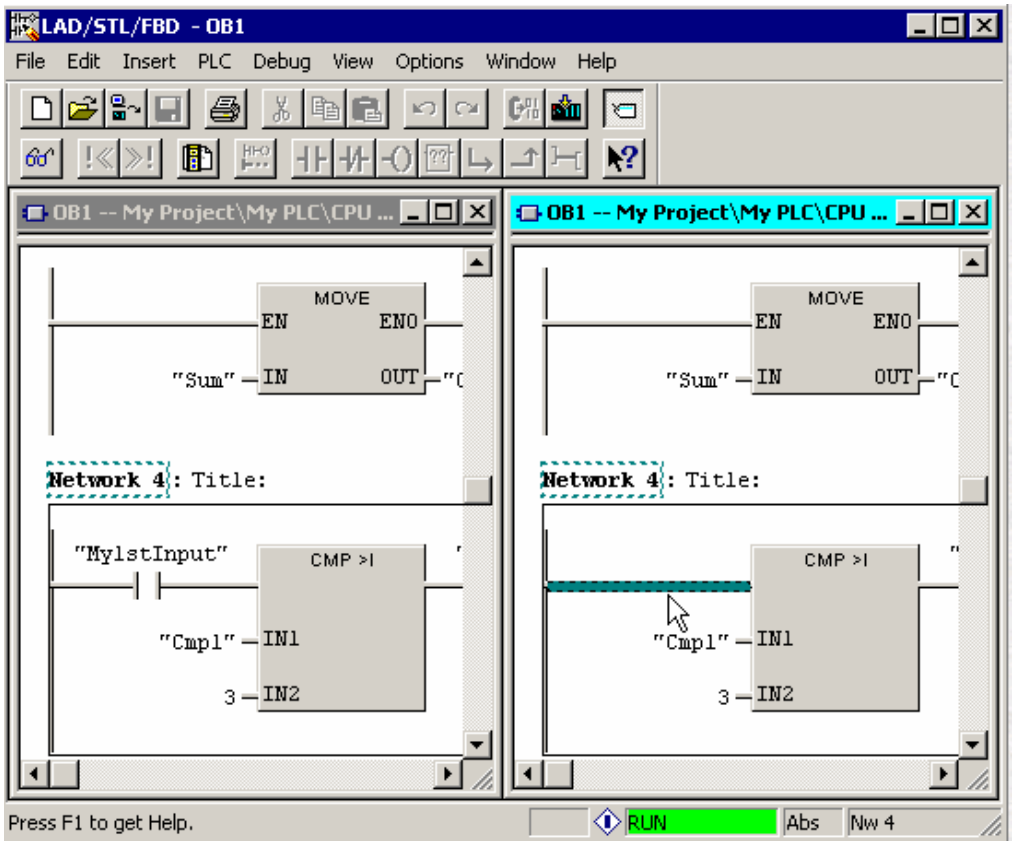
هـ-مقارنة البرنامج



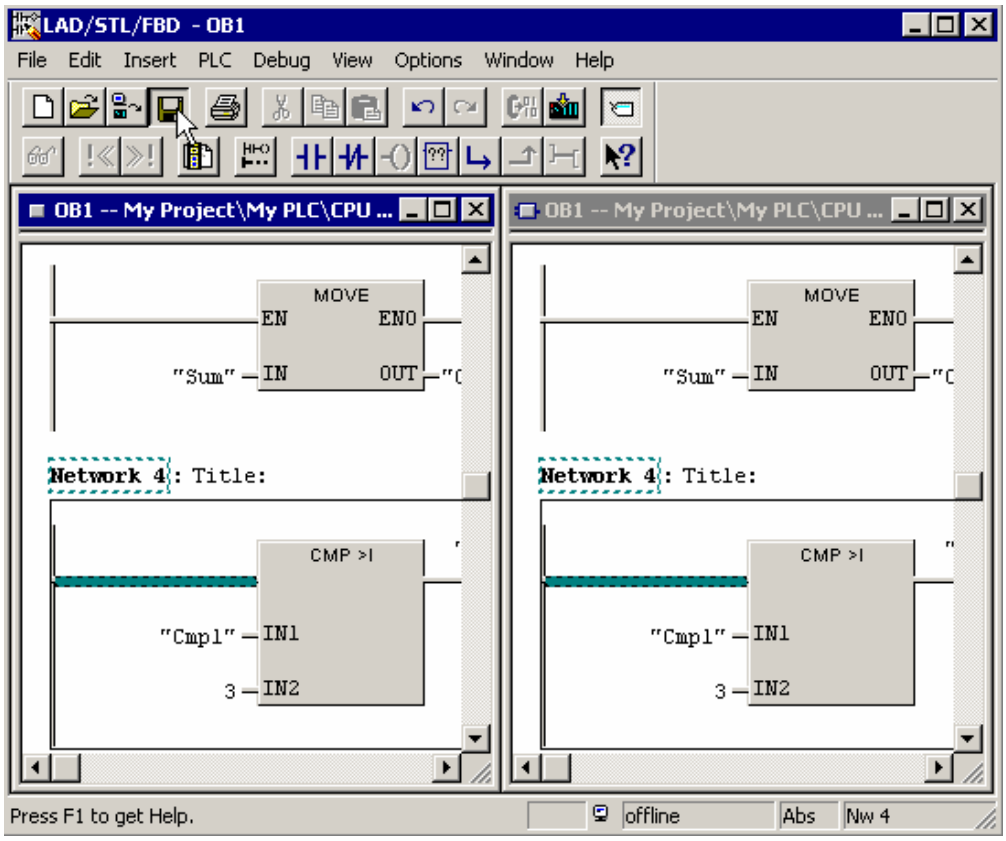

SIMATIC Manager - My Project
File Edit Insert PLC View Options Window Help
< No Filter >
Compare Blocks
Type of comparison:
ONLINE/Offline
Path 1/Path 2
Including SDBs
Selected
Path 1:
My Project\My PLC\CPU 315\S7 Program(1)
Blocks
Compare with...
Path 2:
OK starts the ONLINE/offline comparison.
OK
Cancel
Help
Press F1 to get Help.

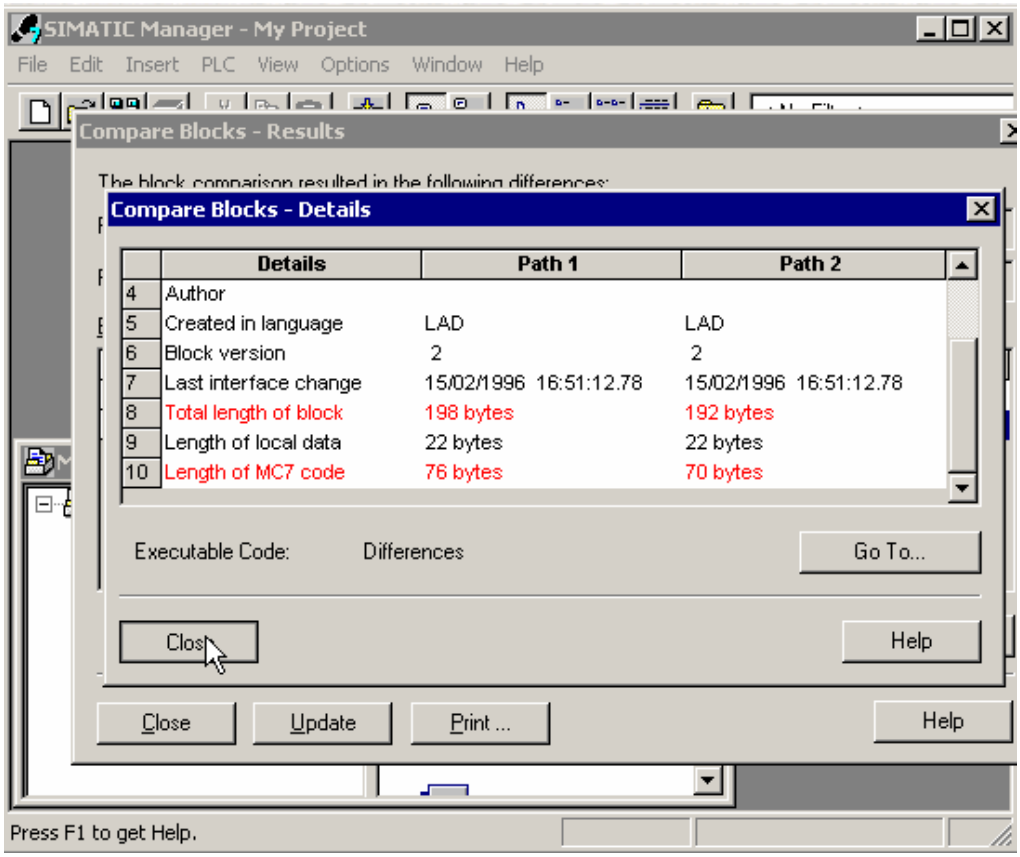

SIMATIC Manager - My Project
File Edit Insert PLC View Options Window Help
Compare Blocks - Results
The block comparison resulted in the following differences:
Path 1:
My Project\My PLC\CPU 315\S7 Program(1)\Blocks
Path 2: ONLINE
My Project\My PLC\CPU 315\S7 Program(1)\Blocks
Block List:
Block
Path 1
Path 2
1
SDB
Differences
2
OB1
05/01/2001 16:43:22.00
06/01/2001 16:36:38.00
Details...
Close
Update
Print ...
Help
Press F1 to get Help.

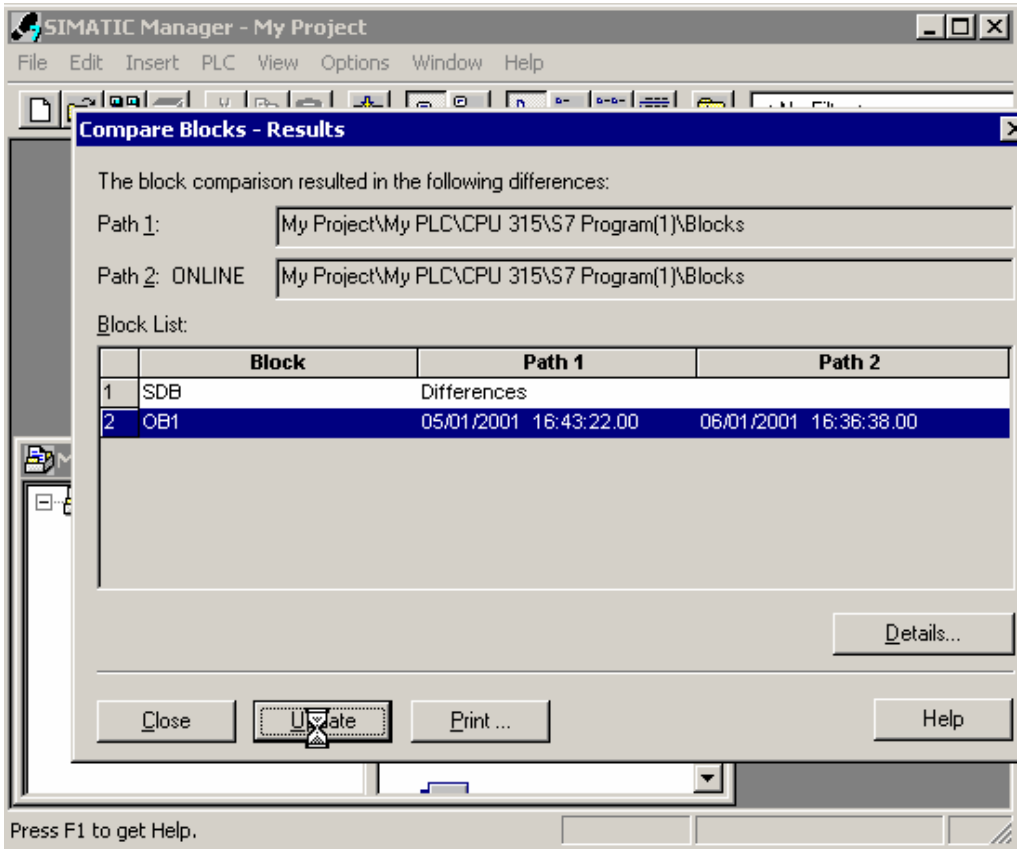

SIMATIC Manager - My Project
File Edit Insert PLC View Options Window Help
Compare Blocks - Results
The block comparison resulted in the following differences:
Compare Blocks - Details
Details
Path 1
Path 2
4 Author
5 Created in language LAD LAD
6 Block version 2 2
7 Last interface change 15/02/1996 16:51:12.78 15/02/1996 16:51:12.78
8 Total length of block 198 bytes 192 bytes
9 Length of local data 22 bytes 22 bytes
10 Length of MC7 code 76 bytes 70 bytes
Executable Code: Differences
Go To...
Close
Help
Close
Update
Print ...
Help
Press F1 to get Help.

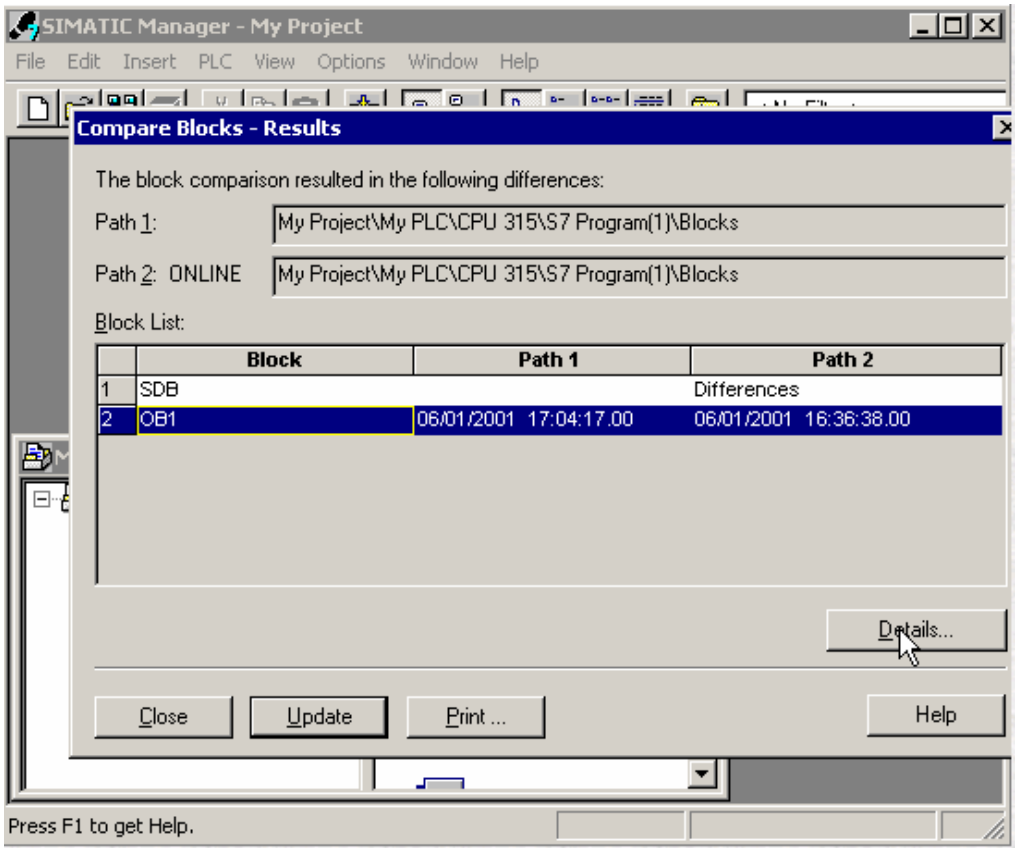

LAD/STL/FBD - OB1
File Edit Insert PLC Debug View Options Window Help
OB1 -- My Project\My PLC\CPU ...
MOVE
EN ENO
"Sum" IN OUT
Network 4: Title:
"MylstInput"
CMP >I
"Cmpl" IN1
3 IN2
OB1 -- My Project\My PLC\CPU ...
MOVE
EN ENO
"Sum" IN OUT
Network 4: Title:
CMP >I
"Cmpl" IN1
3 IN2
Press F1 to get Help.
RUN
Abs
Nw 4

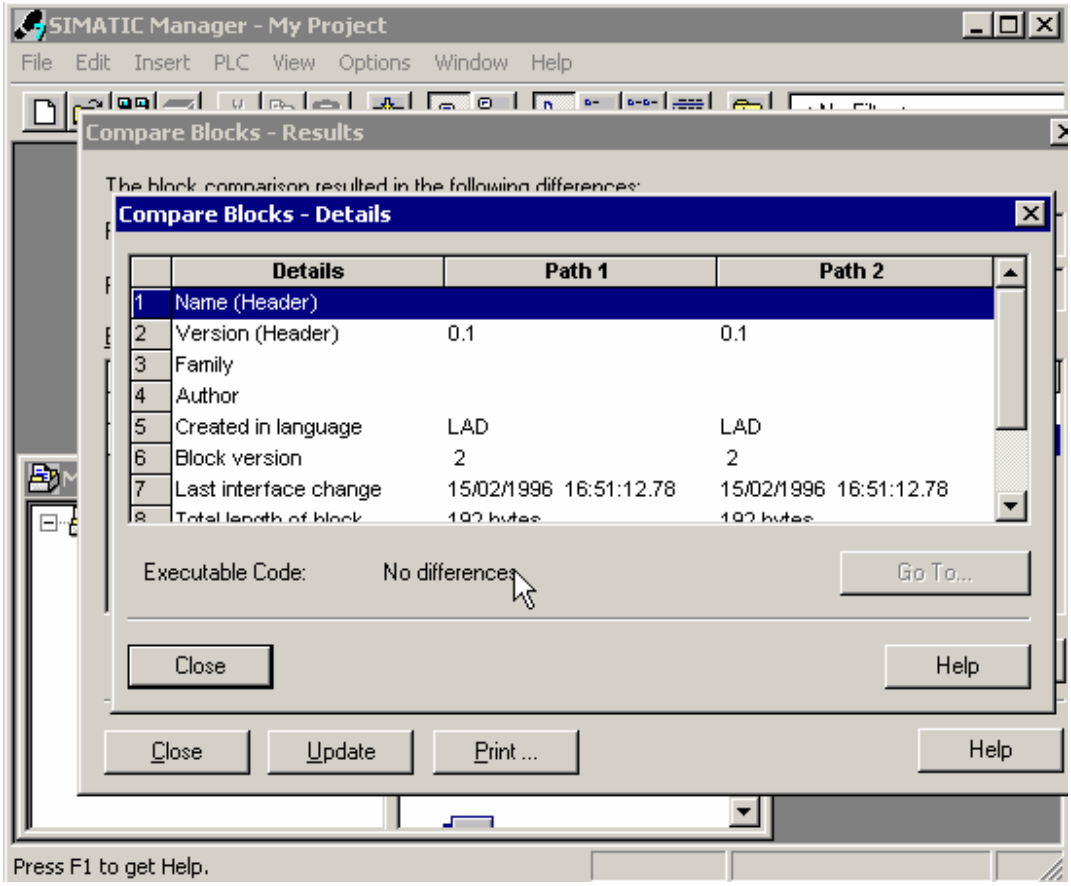
بعد ازالة الاختلاف نقوم بحفظ البرنامج

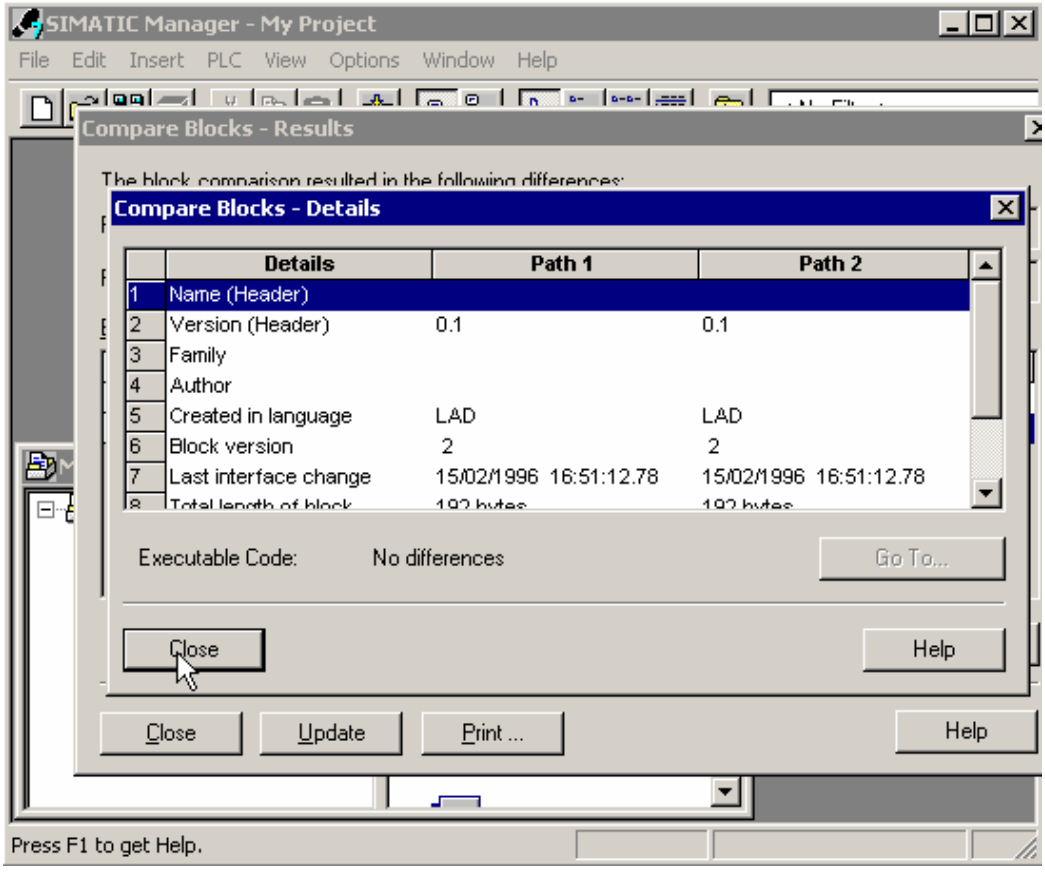





SIMATIC Manager - My Project
File Edit Insert PLC View Options Window Help
Compare Blocks - Results
The block comparison resulted in the following differences:
Path 1:
My Project\My PLC\CPU 315\S7 Program(1)\Blocks
Path 2: ONLINE
My Project\My PLC\CPU 315\S7 Program(1)\Blocks
Block List:
Block
Path 1
Path 2
1 SDB Differences
2 OB1 05/01/2001 16:43:22.00 06/01/2001 16:36:38.00
Details...
Close
Update
Print ...
Help
Press F1 to get Help.


SIMATIC Manager - My Project
File Edit Insert PLC View Options Window Help
Compare Blocks - Results
The block comparison resulted in the following differences:
Path 1:
My Project\My PLC\CPU 315\S7 Program(1)\Blocks
Path 2: ONLINE
My Project\My PLC\CPU 315\S7 Program(1)\Blocks
Block List:
Block
Path 1
Path 2
1 SDB Differences
2 OB1 06/01/2001 17:04:17.00 06/01/2001 16:36:38.00
Details...
Close
Update
Print ...
Help
Press F1 to get Help.


SIMATIC Manager - My Project
File Edit Insert PLC View Options Window Help
Compare Blocks - Results
The block comparison resulted in the following differences:
Compare Blocks - Details
Details
Path 1
Path 2
1 Name (Header)
2 Version (Header) 0.1 0.1
3 Family
4 Author
5 Created in language LAD LAD
6 Block version 2 2
7 Last interface change 15/02/1996 16:51:12.78 15/02/1996 16:51:12.78
Executable Code: No differences
Go To...
Close
Help
Close
Update
Print ...
Help
Press F1 to get Help.


SIMATIC Manager - My Project
File Edit Insert PLC View Options Window Help
Compare Blocks - Results
The block comparison resulted in the following differences:
Compare Blocks - Details
Details
Path 1
Path 2
1 Name (Header)
2 Version (Header) 0.1 0.1
3 Family
4 Author
5 Created in language LAD LAD
6 Block version 2 2
7 Last interface change 15/02/1996 16:51:12.78 15/02/1996 16:51:12.78
Executable Code: No differences
Go To...
Close
Help
Close
Update
Print ...
Help
Press F1 to get Help.

اليوم الثامن

١-الايعازات المنطقية الجزءالاول

يحتوي (Step7) على الكثير من الايعازات والتي يمكن كتابتها باكثر من لغة وبما ان مستوى الدورة مبتدئ سنتعرض فقط الى ايعازات من النوع (LAD) في هذا الفصل والفصول القادمة:

الجدول التالي يبيبن جميع ايعازات (Step7) من نوع (LAD)

English Mnemonics	German Mnemonics	Program Elements Catalog	Description
---|/|---	---|/|---	Bit logic Instruction	Normally Closed Contact (Address)
---| |---	---| |---	Bit logic Instruction	Normally Open Contact (Address)
---( )	---( )	Bit logic Instruction	Output Coil
---(#)---	---(#)---	Bit logic Instruction	Midline Output
==0 ---| |---	==0 ---| |---	Status bits	Result Bit Equal 0
>0 ---| |---	>0 ---| |---	Status bits	Result Bit Greater Than 0
>=0 ---| |---	>=0 ---| |---	Status bits	Result Bit Greater Equal 0
<=0 ---| |---	<=0 ---| |---	Status bits	Result Bit Less Equal 0
<0 ---| |---	<0 ---| |---	Status bits	Result Bit Less Than 0
<>0 ---| |---	<>0 ---| |---	Status bits	Result Bit Not Equal 0
ABS	ABS	Floating point Instruction	Establish the Absolute Value of a Floating-Point Number
ACOS	ACOS	Floating point Instruction	Establish the Arc Cosine Value
ADD_DI	ADD_DI	Integer Math Instruction	Add Double Integer
ADD_I	ADD_I	Integer Math Instruction	Add Integer
ADD_R	ADD_R	Floating point Instruction	Add Real
ASIN	ASIN	Floating point Instruction	Establish the Arc Sine Value
ATAN	ATAN	Floating point Instruction	Establish the Arc Tangent Value
BCD_DI	BCD_DI	Convert	BCD to Double Integer
BCD_I	BCD_I	Convert	BCD to Integer
BR ---| |---	BIE ---| |---	Status bits	Exception Bit Binary Result
----(CALL)	----(CALL)	Program control	Call FC SFC from Coil (without Parameters)
CALL_FB	CALL_FB	Program control	Call FB from Box
CALL_FC	CALL_FC	Program control	Call FC from Box
CALL_SFB	CALL_SFB	Program control	Call System FB from Box
CALL_SFC	CALL_SFC	Program control	Call System FC from Box
----(CD)	----(ZR)	Counters	Down Counter Coil
CEIL	CEIL	Convert	Ceiling
CMP ? D	CMP ? D	Compare	Compare Double Integer (==, <>, >, <, >=, <=)
CMP ? I	CMP ? I	Compare	Compare Integer (==, <>, >, <, >=, <=)

CMP ? R	CMP ? R	Compare	Compare Real (==, <>, >, <, >=, <=)
COS	COS	Floating point Instruction	Establish the Cosine Value
----(CU)	---(ZV)	Counters	Up Counter Coil
DI_BCD	DI_BCD	Convert	Double Integer to BCD
DI_R	DI_R	Convert	Double Integer to Floating-Point
DIV_DI	DIV_DI	Integer Math Instruction	Divide Double Integer
DIV_I	DIV_I	Integer Math Instruction	Divide Integer
DIV_R	DIV_R	Floating point Instruction	Divide Real
EXP	EXP	Floating point Instruction	Establish the Exponential Value
FLOOR	FLOOR	Convert	Floor
I_BCD	I_BCD	Convert	Integer to BCD
I_DI	I_DI	Convert	Integer to Double Integer
INV_I	INV_I	Convert	Ones Complement Integer
INV_DI	INV_DI	Convert	Ones Complement Double Integer
---(JMP)	---(JMP)	Jumps	Unconditional Jump
---(JMP)	---(JMP)	Jumps	Conditional Jump
---(JMPN)	---(JMPN)	Jumps	Jump-If-Not
LABEL	LABEL	Jumps	Label
LN	LN	Floating point Instruction	Establish the Natural Logarithm
---(MCR>)	---(MCR>)	Program control	Master Control Relay Off
---(MCR<)	---(MCR<)	Program control	Master Control Relay On
---(MCRA)	---(MCRA)	Program control	Master Control Relay Activate
---(MCRD)	---(MCRD)	Program control	Master Control Relay Deactivate
MOD_DI	MOD_DI	Integer Math Instruction	Return Fraction Double Integer
MOVE	MOVE	Move	Assign a Value
MUL_DI	MUL_DI	Integer Math Instruction	Multiply Double Integer
MUL_I	MUL_I	Integer Math Instruction	Multiply Integer
MUL_R	MUL_R	Floating point Instruction	Multiply Real
---(N)---	---(N)---	Bit logic Instruction	Negative RLO Edge Detection
NEG	NEG	Bit logic Instruction	Address Negative Edge Detection
NEG_DI	NEG_DI	Convert	Twos Complement Double Integer
NEG_I	NEG_I	Convert	Twos Complement Integer
NEG_R	NEG_R	Convert	Negate Floating-Point Number
---| NOT |---	---| NOT |---	Bit logic Instruction	Invert Power Flow
---(OPN)	---(OPN)	DB call	Open Data Block: DB or DI
OS ---| |---	OS ---| |---	Status bits	Exception Bit Overflow Stored
OV ---| |---	OV ---| |---	Status bits	Exception Bit Overflow
---(P)---	---(P)---	Bit logic Instruction	Positive RLO Edge Detection
POS	POS	Bit logic Instruction	Address Positive Edge Detection
---(R)	---(R)	Bit logic Instruction	Reset Coil
---(RET)	---(RET)	Program control	Return
ROL_DW	ROL_DW	Shift/Rotate	Rotate Left Double Word
ROR_DW	ROR_DW	Shift/Rotate	Rotate Right Double Word

ROUND	ROUND	Convert	Round to Double Integer
RS	RS	Bit logic Instruction	Reset-Set Flip Flop
---(S)	---(S)	Bit logic Instruction	Set Coil
---(SAVE)	---(SAVE)	Bit logic Instruction	Save RLO into BR Memory
---(SC)	---(SZ)	Counters	Set Counter Value
S_CD	Z_RUECK	Counters	Down Counter
S_CU	Z_VORW	Counters	Up Counter
S_CUD	ZAEHLER	Counters	Up-Down Counter
---(SD)	---(SE)	Timers	On-Delay Timer Coil
---(SE)	---(SV)	Timers	Extended Pulse Timer Coil
---(SF)	---(SA)	Timers	Off-Delay Timer Coil
SHL_DW	SHL_DW	Shift/Rotate	Shift Left Double Word
SHL_W	SHL_W	Shift/Rotate	Shift Left Word
SHR_DI	SHR_DI	Shift/Rotate	Shift Right Double Integer
SHR_DW	SHR_DW	Shift/Rotate	Shift Right Double Word
SHR_I	SHR_I	Shift/Rotate	Shift Right Integer
SHR_W	SHR_W	Shift/Rotate	Shift Right Word
SIN	SIN	Floating point Instruction	Establish the Sine Value
S_ODT	S_EVERZ	Timers	On-Delay S5 Timer
S_ODTS	S_SEVERZ	Timers	Retentive On-Delay S5 Timer
S_OFFDT	S_AVERZ	Timers	Off-Delay S5 Timer
---(SP)	---(SI)	Timers	Pulse Timer Coil
S_PEXT	S_VIMP	Timers	Extended Pulse S5 Timer
S_PULSE	S_IMPULS	Timers	Pulse S5 Timer
SQR	SQR	Floating point Instruction	Establish the Square
SQRT	SQRT	Floating point Instruction	Establish the Square Root
SR	SR	Bit logic Instruction	Set-Reset Flip Flop
---(SS)	---(SS)	Timers	Retentive On-Delay Timer Coil
SUB_DI	SUB_DI	Integer Math Instruction	Subtract Double Integer
SUB_I	SUB_I	Integer Math Instruction	Subtract Integer
SUB_R	SUB_R	Floating point Instruction	Subtract Real
TAN	TAN	Floating point Instruction	Establish the Tangent Value
TRUNC	TRUNC	Convert	Truncate Double Integer Part
UO ---| |---	UO ---| |---	Status bits	Exception Bit Unordered
WAND_DW	WAND_DW	Word logic Instruction	AND Double Word
WAND_W	WAND_W	Word logic Instruction	AND Word
WOR_DW	WOR_DW	Word logic Instruction	OR Double Word
WOR_W	WOR_W	Word logic Instruction	OR Word
WXOR_DW	WXOR_DW	Word logic Instruction	Exclusive OR Double Word
WXOR_W	WXOR_W	Word logic Instruction	Exclusive OR Word

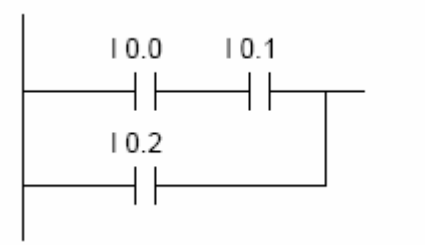
أ- ايعازات (Bit Logic): وتتعامل مع البيانات من نوع (Bool)

١-

---| |--- Normally Open Contact

يقوم بتمرير الاشارة من خلاله عندما تكون قيمة العنوان الذي يحمله (1) وبالعكس وعند مرور الاشارة يتغير الى اللون الاخضر

مثال:



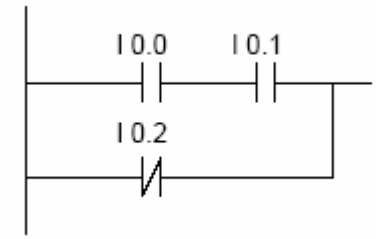
لغرض تمرير الاشارة يجب ان تكون قيمة كلا العنوانين (I0.0,I0.1) تساوي (1) او قيمة العنوان (I0.2) فقط تساوي (1)

٢-

---| / |--- Normally Closed Contact

يقوم بتمرير الاشارة من خلاله عندما تكون قيمة العنوان الذي يحمله (0) وبالعكس وعند مرور الاشارة يتغير الى اللون الابيض

مثال:



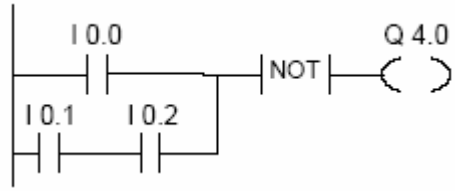
لغرض تمرير الاشارة يجب ان تكون قيمة كلا العنوانين (I0.0,I0.1) تساوي (1) او قيمة العنوان (I0.2) فقط تساوي (0)

٣-

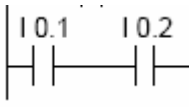
--|NOT|-- Invert Power Flow

يعكس قيمة الاشارة المارة خلاله من (1) الى (0) او بالعكس

مثال:

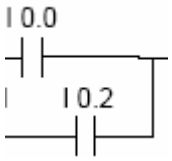


عندما تكون قيمة كلا العنوانين (I0.2,I0.1) تساوي (1) او قيمة العنوان (I0.0) فقط تساوي (1) تكون قيمة العنوان (Q4.0) تساوي (0)

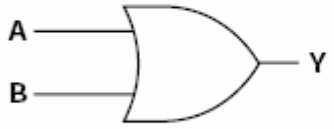


حيث ان الشكل التالي يمثل (And Gate) بين قيمة العنوانين

AND Truth Table		
Inputs		Output
A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

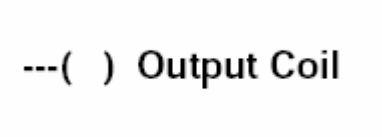


و الشكل التالي يمثل (OR Gate) بين قيمة العنوانين

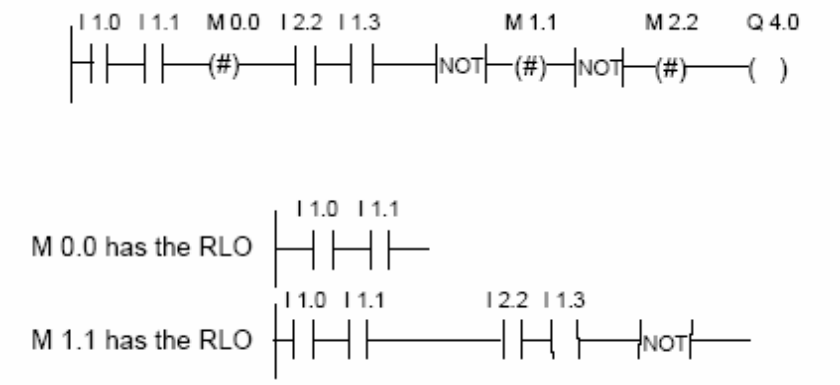


OR Truth Table		
Inputs		Output
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

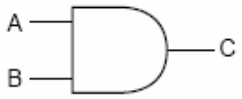
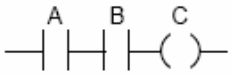
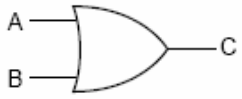
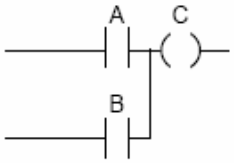
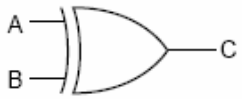
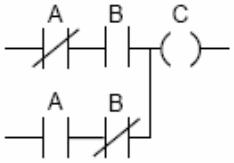
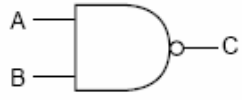
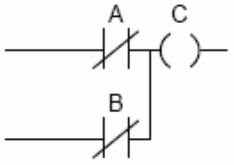
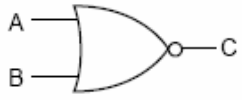
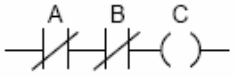
٤-



يقوم بتمرير الاشارة من خلاله عندما تكون قيمة العنوان الذي يحمله (1) ويمكن ان يكتب بمواق مختلفة كالتالي:



والجدول التالي يبين العلاقة بين ايعازات (PLC) والبوابات المنطقية

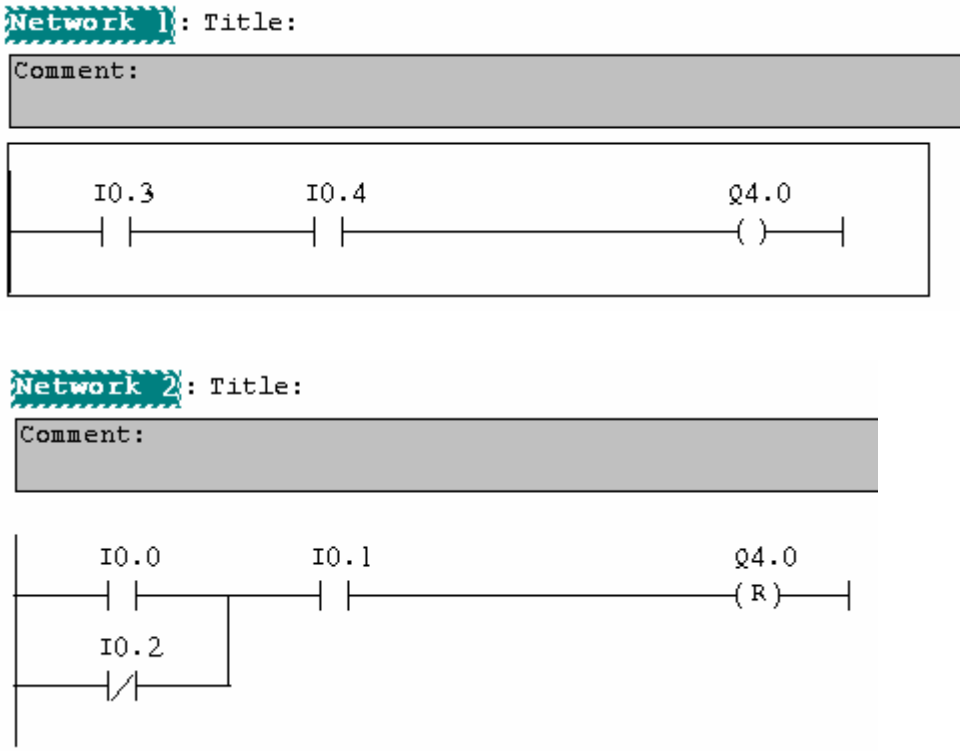
Logic Diagram	Truth Table	Ladder Diagram
 AND Gate	A B C 0 0 0 0 1 0 1 0 0 1 1 1	 AND Equivalent Circuit
 OR Gate	A B C 0 0 0 0 1 1 1 0 1 1 1 1	 OR Equivalent Circuit
 Exclusive-OR Gate	A B C 0 0 0 0 1 1 1 0 1 1 1 0	 Exclusive-OR Equivalent Circuit
 NAND Gate	A B C 0 0 1 0 1 1 1 0 1 1 1 0	 NAND Equivalent Circuit
 NOR Gate	A B C 0 0 1 0 1 0 1 0 0 1 1 0	 NOR Equivalent Circuit

٥-

---(R) Reset Coil

عندما تمر الاشارة من خلاله يقوم بتغيير قيمة العنوان الذي يحمله الى (0) ولا توجد علاقة بين مداخل هذا الايعاز مع العنوان الذي يحمله

مثال:



في الدائرة الاولى:
تصبح قيمة العنوان (Q4.0) تساوي (1) عندما تكون قيمة كلا العنوانين (I0.4,I0.3) تساوي (1)
في الدائرة الثانية:
تصبح قيمة المخرج (R) تساوي (1) عندما تكون قيمة كلا العنوانين (I0.1,I0.0) تساوي (1) او قيمة العنوان (I0.2) تساوي (0) فيقوم المخرج (R) بجعل قيمة العنوان الذي يحمله (Q4.0) تساوي (0)

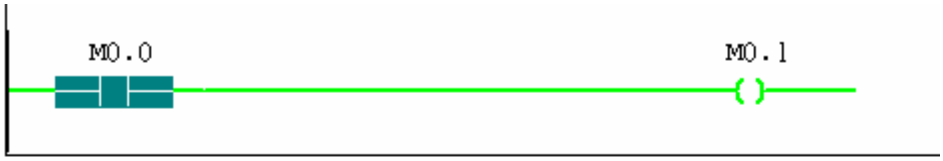
٦-

---(S) Set Coil

عندما تمر الاشارة من خلاله يقوم بتغيير قيمة العنوان الذي يحمله الى (1) ولا توجد علاقة بين مداخل هذا الايعاز مع العنوان الذي يحمله عكس الايعاز السابق

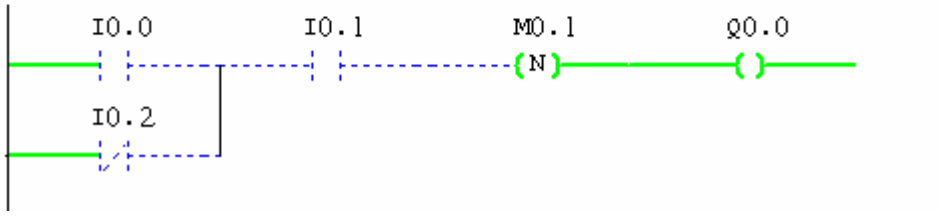
-٧

---(N)--- Negative RLO Edge Detection



Network 2 : Title:

Comment:

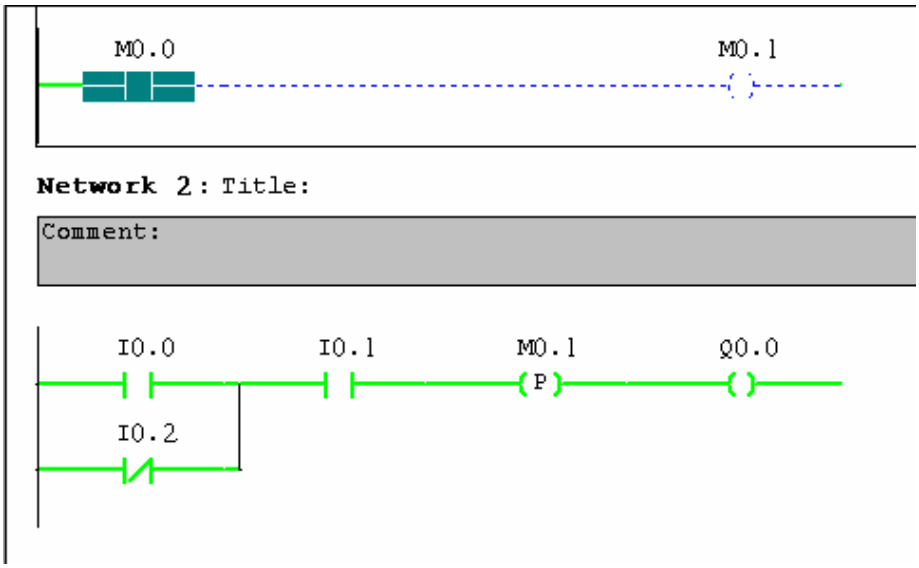


لفهم عمل هذا الايعاز يجب فهم العلاقة بين الايعز والعنوان الذي يحمله وحسب المثال اعلاه سنكتب العلاقة بالجدول التالي:

M0.1	N	Q0.0
0	0	0
0	1	0
1	0	1
1	1	0

-٨

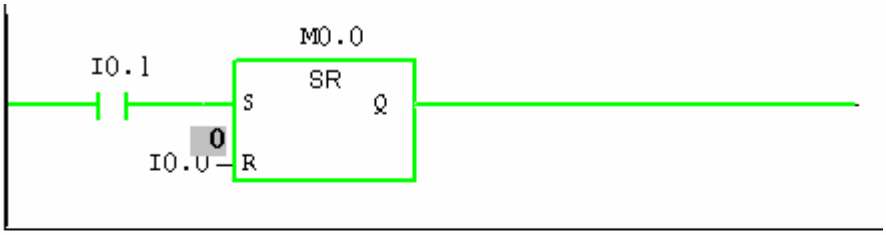
---(P)--- Positive RLO Edge Detection



M0.1	P	Q0.0
0	0	0
0	1	1
1	0	0
1	1	0

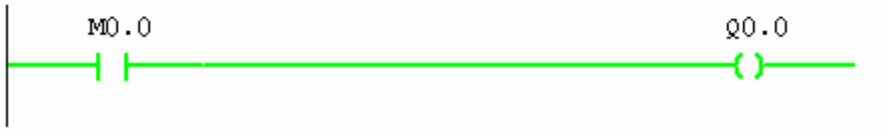
(SR FLIP FLOP) -٩

S	R	Q
0	0	No change
0	1	0
1	0	1
1	1	0



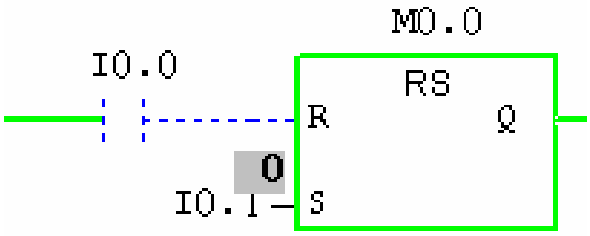
Network 2 : Title:

Comment:



I0.1	I0.0	M0.0
0	0	No change
0	1	0
1	0	1
1	1	0

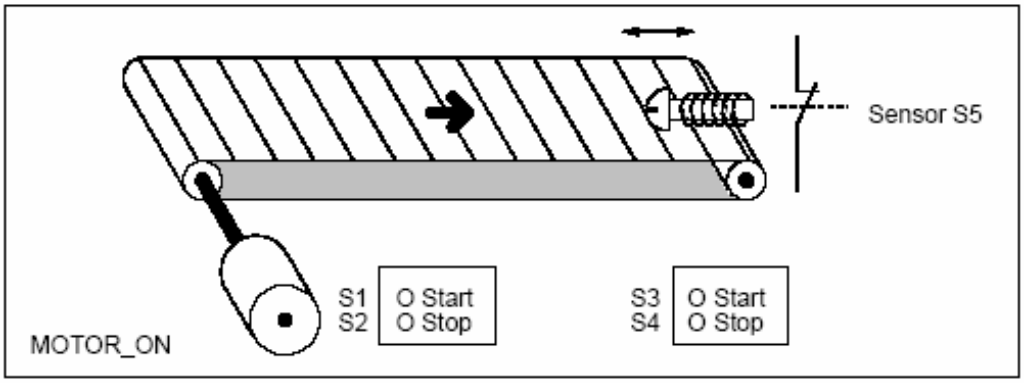
(RS FLIP FLOP)-١٠



S	R	Q
0	0	No change
0	1	0
1	0	1
1	1	1

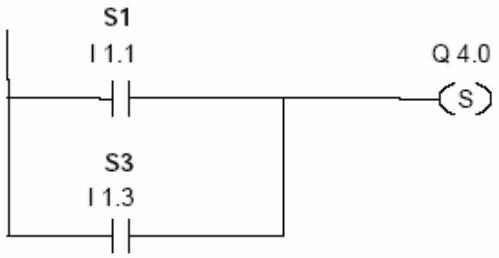
Example 1: Controlling a Conveyor Belt

The following figure shows a conveyor belt that can be activated electrically. There are two push button switches at the beginning of the belt: S1 for START and S2 for STOP. There are also two push button switches at the end of the belt: S3 for START and S4 for STOP. It it possible to start or stop the belt from either end. Also, sensor S5 stops the belt when an item on the belt reaches the end.

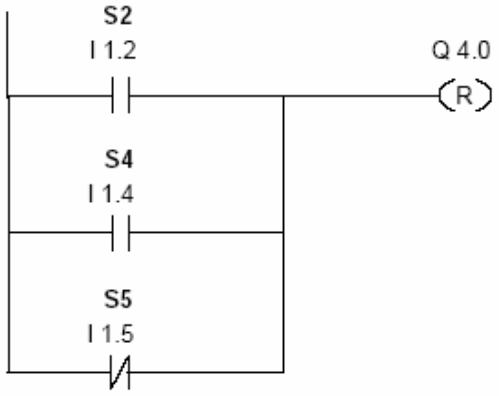


System Component	Absolute Address	Symbol	Symbol Table
Push Button Start Switch	I 1.1	S1	I 1.1 S1
Push Button Stop Switch	I 1.2	S2	I 1.2 S2
Push Button Start Switch	I 1.3	S3	I 1.3 S3
Push Button Stop Switch	I 1.4	S4	I 1.4 S4
Sensor	I 1.5	S5	I 1.5 S5
Motor	Q 4.0	MOTOR_ON	Q 4.0 MOTOR_ON

Network 1: Pressing either start switch turns the motor on.

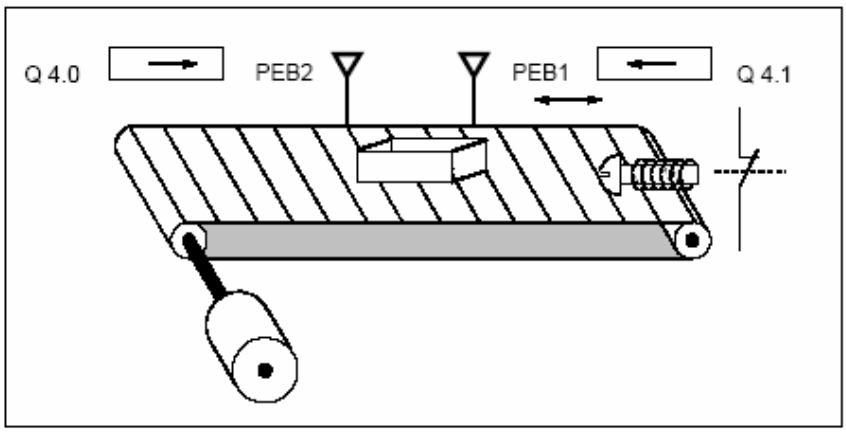


Network 2: Pressing either stop switch or opening the normally closed contact at the end of the belt turns the motor off.



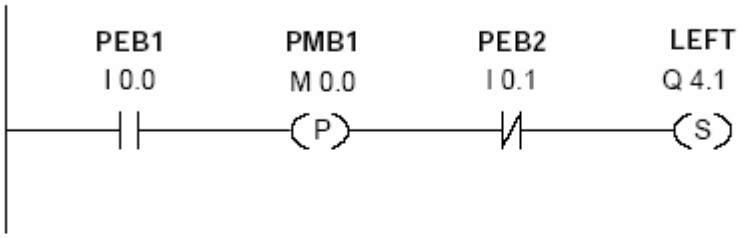
Example 2: Detecting the Direction of a Conveyor Belt

The following figure shows a conveyor belt that is equipped with two photoelectric barriers (PEB1 and PEB2) that are designed to detect the direction in which a package is moving on the belt. Each photoelectric light barrier functions like a normally open contact.

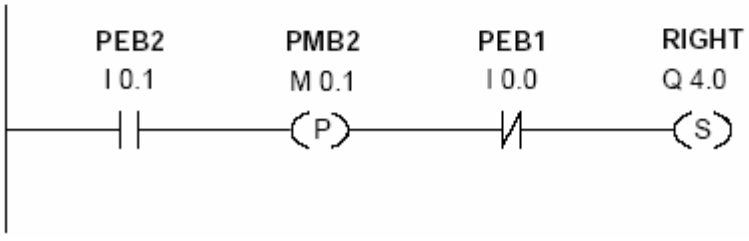


System Component	Absolute Address	Symbol	Symbol Table
Photoelectric barrier 1	I 0.0	PEB1	I 0.0 PEB1
Photoelectric barrier 2	I 0.1	PEB2	I 0.1 PEB2
Display for movement to right	Q 4.0	RIGHT	Q 4.0 RIGHT
Display for movement to left	Q 4.1	LEFT	Q 4.1 LEFT
Pulse memory bit 1	M 0.0	PMB1	M 0.0 PMB1
Pulse memory bit 2	M 0.1	PMB2	M 0.1 PMB2

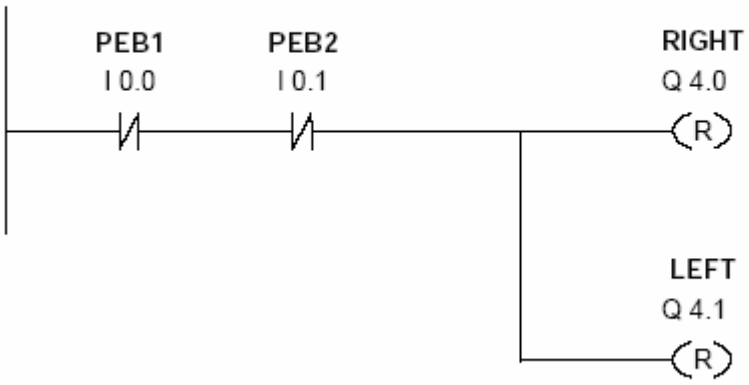
Network 1: If there is a transition in signal state from 0 to 1 (positive edge) at input I 0.0 and, at the same time, the signal state at input I 0.1 is 0, then the package on the belt is moving to the left.



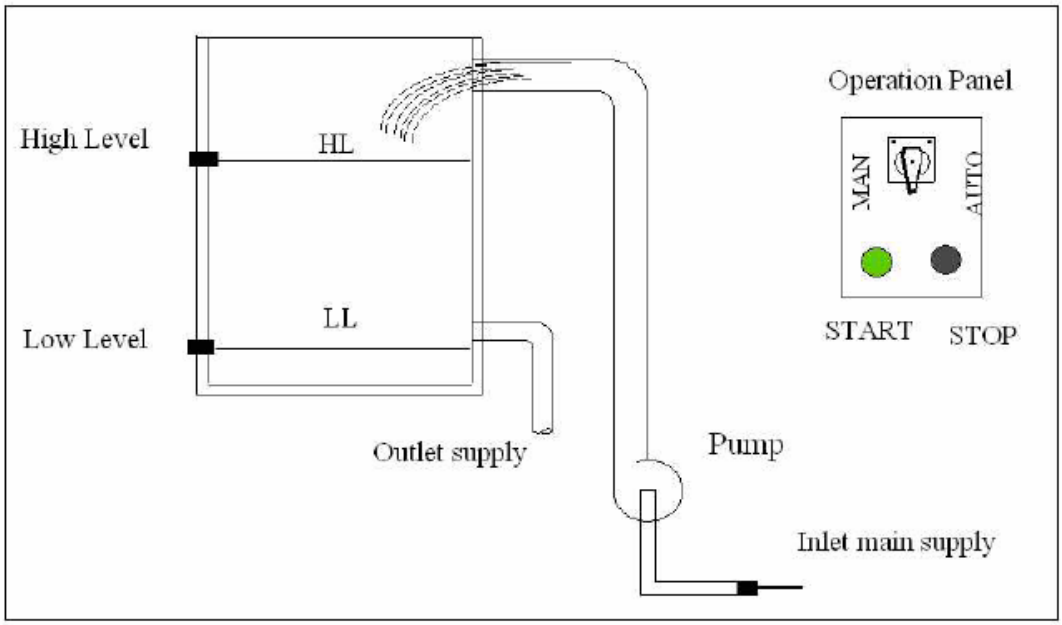
Network 2: If there is a transition in signal state from 0 to 1 (positive edge) at input I 0.1 and, at the same time, the signal state at input I 0.0 is 0, then the package on the belt is moving to the right. If one of the photoelectric light barriers is broken, this means that there is a package between the barriers.



Network 3: If neither photoelectric barrier is broken, then there is no package between the barriers. The direction pointer shuts off.



تمرين: عند تحويل وضع المفتاح الى (Auto) يكون تشغيل مضخة الاملاء عن طريق (Low Level) واطفائها عن طريق (High Level) اما عند تحويل وضع المفتاح الى (Manual) يتم تشغيل المضخة من مفتاح (Start) بشرط عدم تحقق (High Level) واطفائها من مفتاح (Stop) بشرط عدم تحقق (Low Level) اي التشغيل والاطفاء بشكل يدوي ضمن المسافة بين (HL) و(LL)



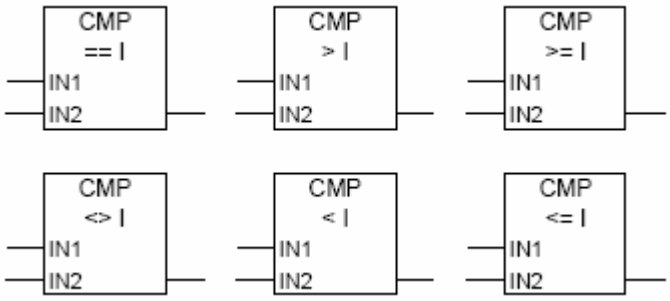
ب- ايعازات المقارنة:

وتقسم الى ثلاثة مجاميع لها نفس العمل ولكن الاختلاف بنوع البيانات الندخلة فهناك مجاميع تتعامل مع بيانات من نوع (Integer) او مع بيانات من نوع (Double Integer) او مع بيانات من نوع (Real) وحدود البيانات بالجدول التالي:

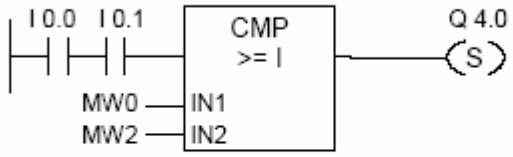
INT (Integer)	16	Decimal number signed	-32768 to 32767
DINT (Integer, 32 bits)	32	Decimal number signed	L#-2147483648 to L#2147483647
REAL (Floating-point	32	IEEE Floating-point number	Upper limit: ±3.402823e+38

١- CMP ? I Compare Integer

وتشمل الايعازات التالية ونوع المداخل (Integer)



Example



Output Q4.0 is set if the following conditions exist:

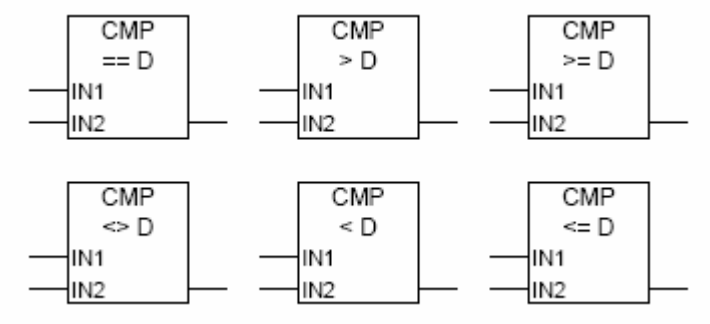
- There is a signal state of "1" at inputs I0.0 and at I0.1
- AND MW0 >= MW2

بنفس الطريقة باقي الايعازات مع اختلاف وظيفة الايعاز وهذه الايعازات سهلة الاستعمال ولا تحتاج الى شرح

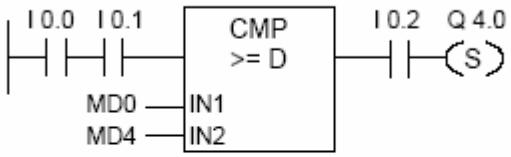
-٢

CMP ? D Compare Double Integer

وتشمل الايعازات التالية ونوع المداخل (Double Integer)



Example

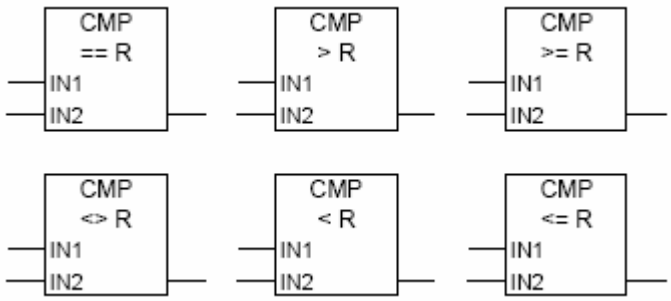


Output Q4.0 is set if the following conditions exist:

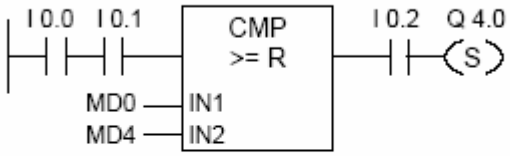
- There is a signal state of "1" at inputs I0.0 and at I0.1
- And MD0 >= MD4
- And there is a signal state of"1" at input I0.2

-٣

CMP ? R Compare Real



Example



Output Q4.0 is set if the following conditions exist:

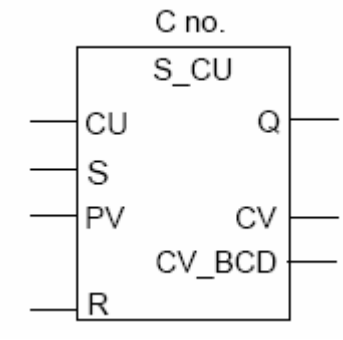
- There is a signal state of "1" at inputs I0.0 and at I0.1
- And MD0 >= MD4
- And there is a signal state of"1" at input I0.

اليوم التاسع
الايعازات المنطقية الجزءالثاني
أ-ايعازات العدادات

١-

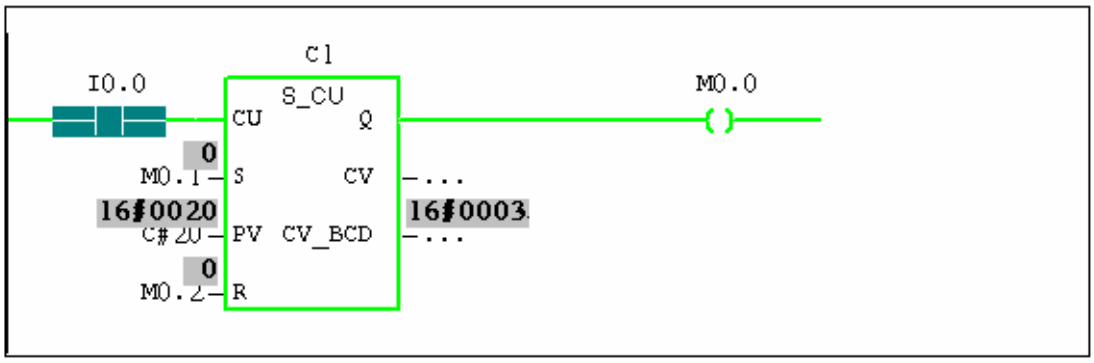
S_CU Up Counter

وهو عبارة عن عداد تصاعدي يعمل على العد من (0) الى الرقم المطلوب ويحمل المعطيات التالية:



١- (C no): اسم العداد يجب ان يعرف ب (Symbol Table) مثل (C1,C2…)
٢- (S): يعطي امر للاستعداد للعد عندما يكون واحد وبعدها لاتؤثر قيمته
٣- (CU): عندما تتغير قيمته من (0) الى (1) يبدأ العداد بالعد بزيادة رقم (1)
٤- (PV): نضع الرقم المراد الوصول اليه بالعد
٥- (R): عندما تصبح قيمته (1) يؤدي الى تصفير العداد
٦- (Q): تصبح قيمته (1) اثناء العد و (0) عندما تكون قيمة العداد (0)
٧- (CV): يعرض قيمة العداد بالنظام السداسي عشر
٨- (CV_BCD): يعرض قيمة العداد بنظام (BCD)

Parameter English	Parameter German	Data Type	Memory Area	Description
C no.	Z no.	COUNTER	C	Counter identification number; range depends of CPU
CU	ZV	BOOL	I, Q, M, L, D	Count up input
S	S	BOOL	I, Q, M, L, D	Set input for presetting counter
PV	ZW	WORD	I, Q, M, L, D or constant	Enter counter value as C#<value> in the range from 0 to 999
PV	ZW	WORD	I, Q, M, L, D	Value for presetting counter
R	R	BOOL	I, Q, M, L, D	Reset input
CV	DUAL	WORD	I, Q, M, L, D	Current counter value, hexadecimal number
CV_BCD	DEZ	WORD	I, Q, M, L, D	Current counter value, BCD coded
Q	Q	BOOL	I, Q, M, L, D	Status of the counter



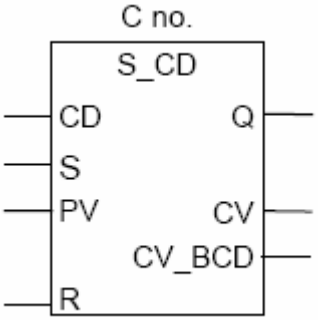
تزداد قيمة العداد بمقدار (1) كلما تغير العنوان (I0.0) من (0) الى (1)

٢-

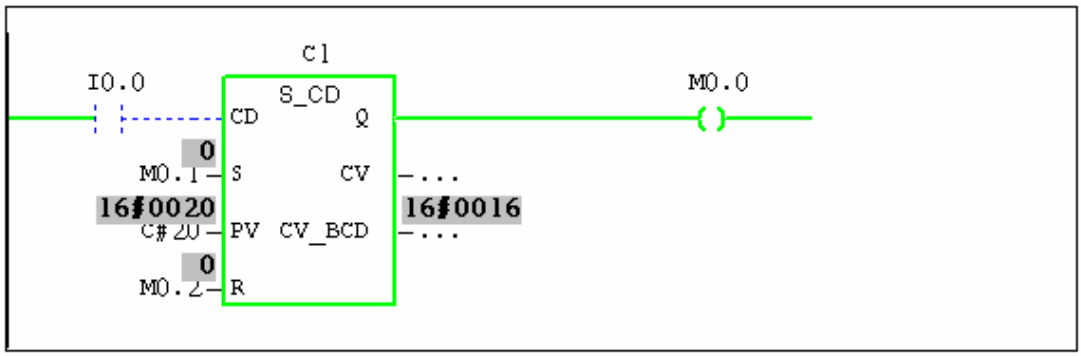
S_CD Down Counter

وهو عبارة عن عداد تنازلي يعمل على العد من الرقم المطلوب الى الصفر ويحمل المعطيات التالية:

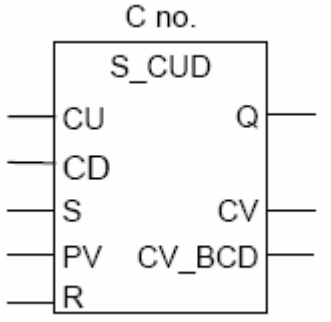
١- (Cno): اسم العداد يجب ان يعرف ب (Symbol Table) مثل (C1,C2…)
٢- (S): يعطي امر للاستعداد للعد عندما يكون واحد وبعدها لاتؤثر قيمته
٣- (CD): عندما تتغير قيمته من (0) الى (1) يبدأ العداد بالعد بنقصان رقم (1)
٤- (PV): نضع الرقم المراد الوصول اليه بالعد
٥- (R): عندما تصبح قيمته (1) يؤدي الى تصفير العداد
٦- (Q): تصبح قيمته (1) اثناء العد و (0) عندما تكون قيمة العداد (0)
٧- (CV): يعرض قيمة العداد بالنظام السداسي عشر
٨- (CV_BCD): يعرض قيمة العداد بنظام (BCD)



Parameter English	Parameter German	Data Type	Memory Area	Description
C no.	Z no.	COUNTER	C	Counter identification number; range depends of CPU
CD	ZR	BOOL	I, Q, M, L, D	Count down input
S	S	BOOL	I, Q, M, L, D	Set input for presetting counter
PV	ZW	WORD	I, Q, M, L, D or constant	Enter counter value as C#<value> in the range from 0 to 999
PV	ZW	WORD	I, Q, M, L, D	Value for presetting counter
R	R	BOOL	I, Q, M, L, D	Reset input
CV	DUAL	WORD	I, Q, M, L, D	Current counter value, hexadecimal number
CV_BCD	DEZ	WORD	I, Q, M, L, D	Current counter value, BCD coded
Q	Q	BOOL	I, Q, M, L, D	Status counter

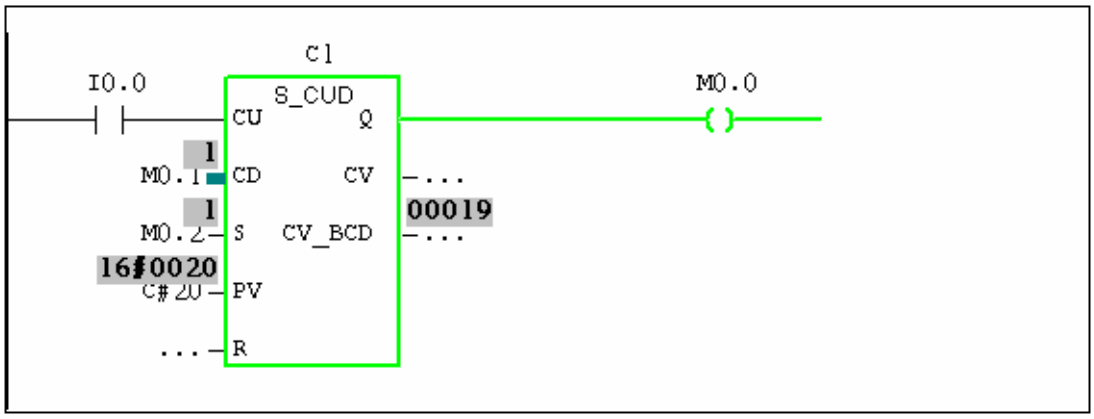


٣- S_CUD Up-Down Counter



وهو عداد يجمع بين النوعين السابقين عندما تتغير قيمة (CU) من (0) الى (1) تزداد قيمة العداد بمقدار (1) و عندما تتغير قيمة (CD) من (0) الى (1) تقل قيمة العداد بمقدار (1)

Parameter English	Parameter German	Data Type	Memory Area	Description
C no.	Z no.	COUNTER	C	Counter identification number; range depends on CPU
CU	ZV	BOOL	I, Q, M, L, D	Count up input
CD	ZR	BOOL	I, Q, M, L, D	Count down input
S	S	BOOL	I, Q, M, L, D	Set input for presetting counter
PV	ZW	WORD	I, Q, M, L, D or constant	Enter counter value as C#<value> in the range from 0 to 999
PV	ZW	WORD	I, Q, M, L, D	Value for presetting counter
R	R	BOOL	I, Q, M, L, D	Reset input
CV	DUAL	WORD	I, Q, M, L, D	Current counter value, hexadecimal number
CV_BCD	DEZ	WORD	I, Q, M, L, D	Current counter value, BCD coded
Q	Q	BOOL	I, Q, M, L, D	Status of the counter



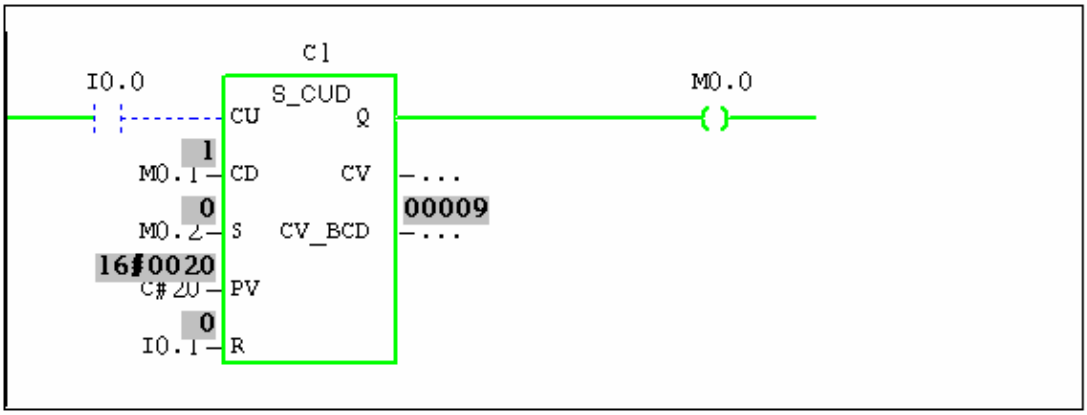
٤-

---(SC) Set Counter Value

يعمل هذا الايعاز على اعطاء قيمة جديدة للعداد بدل القيمة القديمة (PV) ويعمل كلما تغير مدخل هذا الايعاز من (0) الى (1) ويأخذ العداد قيمته من هذا الايعاز ويهمل القيمة الموضوعة في مدخل (PV) للعداد

<C no.>
---(SC)
<preset value>

Parameter English	Parameter German	Data Type	Memory Area	Description
<C no.>	<Z no.>	COUNTER	C	Counter number to be preset
<preset value>	<preset value>	WORD	I, Q, M, L, D	Value for presetting BCD (0 to 999)



Network 2 : Title:

Comment:

٥-

---(CU) Up Counter Coil

<C no.>
---(CU) **يعمل على زيادة قيمة العداد بمقدار واحد عن تغير مدخل الايعاز من صفر الى واحد**

٦-

---(CD) Down Counter Coil

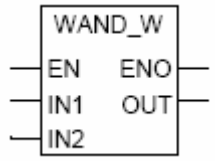
يعمل على تقليل قيمة العداد بمقدار واحد عن تغير مدخل الايعاز من صفر الى واحد

	Decimal		Binary					BCD								Hexadecimal	
	$*10^1$ =10	$*10^0$ =1	$*2^4$ =16	$*2^3$ =8	$*2^2$ =4	$*2^1$ =2	$*2^0$ =1	Tens Tetrad				Ones Tetrad				$*16^1$ = 16	$*16^0$ = 1
								8	4	2	1	8	4	2	1		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	0	1
2	0	2	0	0	0	1	0	0	0	0	0	0	0	1	0	0	2
3	0	3	0	0	0	1	1	0	0	0	0	0	0	1	1	0	3
4	0	4	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
5	0	5	0	0	1	0	1	0	0	0	0	0	1	0	1	0	5
6	0	6	0	0	1	1	0	0	0	0	0	0	1	1	0	0	6
7	0	7	0	0	1	1	1	0	0	0	0	0	1	1	1	0	7
8	0	8	0	1	0	0	0	0	0	0	0	1	0	0	0	0	8
9	0	9	0	1	0	0	1	0	0	0	0	1	0	0	1	0	9
10	1	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0	A
11	1	1	0	1	0	1	1	0	0	0	1	0	0	0	1	0	B
12	1	2	0	1	1	0	0	0	0	0	1	0	0	1	0	0	C
13	1	3	0	1	1	0	1	0	0	0	1	0	0	1	1	0	D
14	1	4	0	1	1	1	0	0	0	0	1	0	1	0	0	0	E
15	1	5	0	1	1	1	1	0	0	0	1	0	1	0	1	0	F
16	1	6	1	0	0	0	0	0	0	0	1	0	1	1	0	1	0
17	1	7	1	0	0	0	1	0	0	0	1	0	1	1	1	1	1
18	1	8	1	0	0	1	0	0	0	0	1	1	0	0	0	1	2
19	1	9	1	0	0	1	1	0	0	0	1	1	0	0	1	1	3

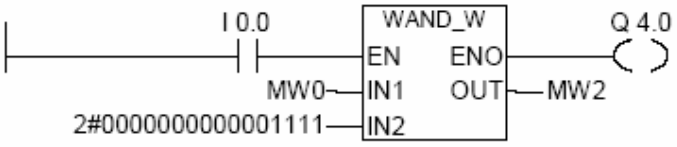
ب- الايعازات المنطقية لبيانات من نوع (Word)

١-

WAND_W (Word) AND Word



Example

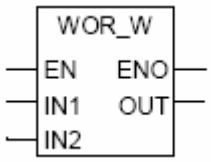


The instruction is executed if I0.0 is "1". Only bits 0 to 3 of MW0 are relevant, the rest of MW0 is masked by the IN2 word bit pattern:

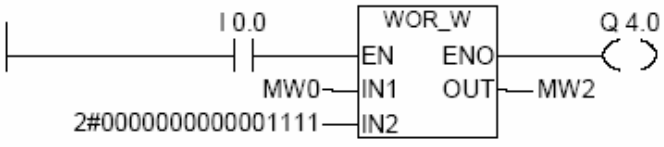
MW0	=	01010101 01010101
IN2	=	00000000 00001111
MW0 AND IN2 = MW2	=	00000000 00000101

Q4.0 is "1" if the instruction is executed.

٢- WOR_W (Word) OR Word



Example



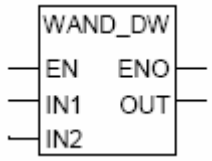
The instruction is executed if I0.0 is "1". Bits 0 to 3 are set to "1", all other MW0 bits are not changed.

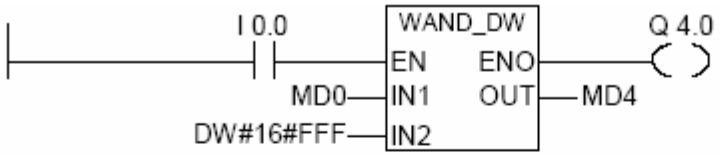
MW0	=	01010101 01010101
IN2	=	00000000 00001111
MW0 OR IN2=MW2	=	01010101 01011111

Q4.0 is "1" if the instruction is executed.

-٣

WAND_DW (Word) AND Double Word





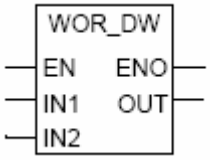
The instruction is executed if I0.0 is "1". Only bits 0 and 11 of MD0 are relevant, the rest of MD0 is masked by the IN2 bit pattern:

MD0	=	01010101 01010101 01010101 01010101
IN2	=	00000000 00000000 00001111 11111111
MD0 AND IN2 = MD4	=	00000000 00000000 00000101 01010101

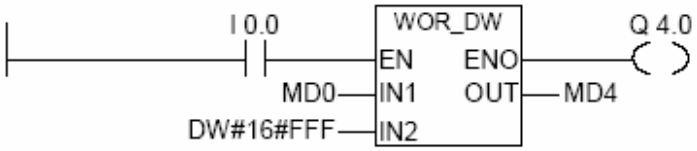
Q4.0 is "1" if the instruction is executed.

-٤

WOR_DW (Word) OR Double Word



Example



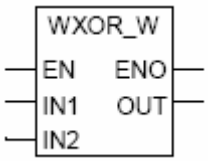
The instruction is executed if I0.0 is "1". Bits 0 to 11 are set to "1", the remaining MD0 bits are not changed:

MD0	=	01010101 01010101 01010101 01010101
IN2	=	00000000 00000000 00001111 11111111
MD0 OR IN2 = MD4	=	01010101 01010101 01011111 11111111

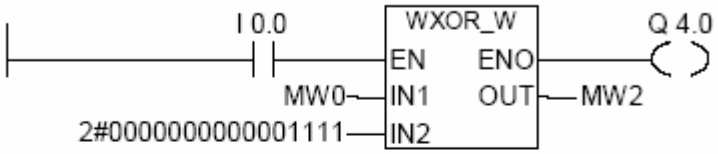
Q4.0 is "1" if the instruction is executed.

-٥

WXOR_W (Word) Exclusive OR Word



Example

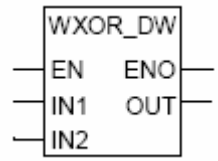


The instruction is executed if I0.0 is "1":

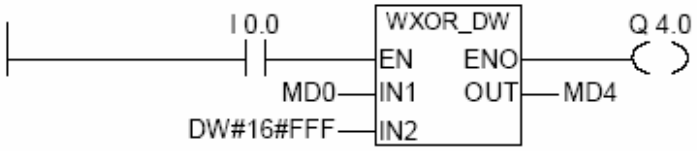
MW0	=	01010101 01010101
IN2	=	00000000 00001111
MW0 XOR IN2 = MW2	=	01010101 01011010

Q4.0 is "1" if the instruction is executed.

WXOR_DW (Word) Exclusive OR Double Word



Example



The instruction is executed if I0.0 is "1":

MD0	=	01010101 01010101 01010101 01010101
IN2	=	00000000 00000000 00001111 11111111
MW2 = MD0 XOR IN2	=	01010101 01010101 01011010 10101010

Q4.0 is "1" if the instruction is executed.

ج- ايعازات المؤقتات (Timers)
تكون صيغة البيانات ليتم تعريفها كوقت كالتالي:

- S5T#a**H**_b**M**_c**S**_d**MS**
  - Where H = hours, M = minutes, S = seconds, and MS = milliseconds

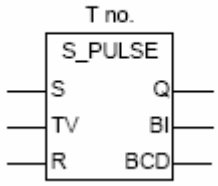
The maximum time value that you can enter is 9,990 seconds, or 2H_46M_30S.

S5TIME#4S = 4 seconds
s5t#2h_15m = 2 hours and 15 minutes
S5T#1H_12M_18S = 1 hour, 12 minutes, and 18 seconds

١-

S_PULSE Pulse S5 Timer

English

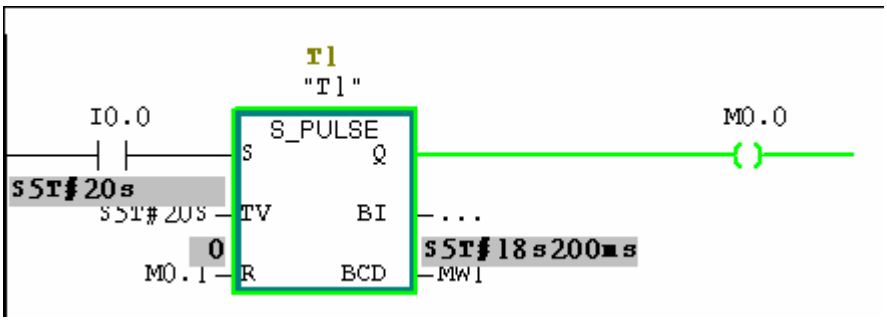


Parameter English	Parameter German	Data Type	Memory Area	Description
T no.	T-Nr.	TIMER	T	Timer identification number; range depends on CPU
S	S	BOOL	I, Q, M, L, D	Start input
TV	TW	S5TIME	I, Q, M, L, D	Preset time value
R	R	BOOL	I, Q, M, L, D	Reset input
BI	DUAL	WORD	I, Q, M, L, D	Remaining time value, integer format
BCD	DEZ	WORD	I, Q, M, L, D	Remaining time value, BCD format
Q	Q	BOOL	I, Q, M, L, D	Status of the timer

المعطيات:

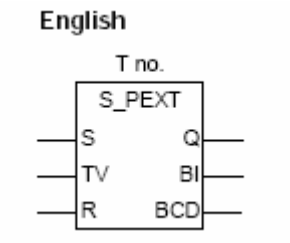
١- (Tno): اسم المؤقت مثلا (T1,T6....)
٢- (S): عند تغير قيمته من (0) الى (1) يبدأ التايمر بالعمل
٣- (TV): زمن التايمر
٤- (R): عندما تصبح قيمته واحد يتوقف التايمر عن العمل
٥-(BI): يعرض قيمة الزمن المتبقي بالنظام السداسي عشر

٦-(BCD): يعرض قيمة الزمن المتبقي بنظام (BCD)
٧-(Q): تكون قيمته (1) عند بدأ التايمر بالعمل وتصبح (0) عند توقف التايمر بعد انتهاء الزمن



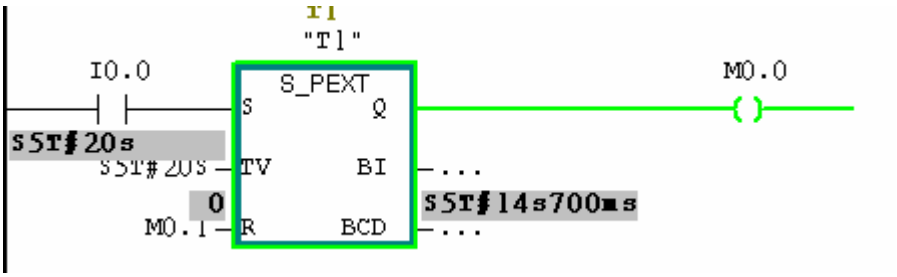
-٢

S_PEXT Extended Pulse S5 Timer



نفس مبدأ عمل النوع الاول ولكن الفرق ان (Q) تبقى قيمته (1) حتى لواصبحت قيمة (S) تساوي (0) الى ان ينتهي الزمن لترجع قيمته الى الصفر اما النوع الاول فأن (Q) تصبح قيمته (0) اذا تغيرت قيمة (S) الى (0)

Parameter English	Parameter German	Data Type	Memory Area	Description
T no.	T-Nr.	TIMER	T	Timer identification number; range depends on CPU
S	S	BOOL	I, Q, M, L, D	Start input
TV	TW	S5TIME	I, Q, M, L, D	Preset time value
R	R	BOOL	I, Q, M, L, D	Reset input
BI	DUAL	WORD	I, Q, M, L, D	Remaining time value, integer format
BCD	DEZ	WORD	I, Q, M, L, D	Remaining time value, BCD format
Q	Q	BOOL	I, Q, M, L, D	Status of the timer

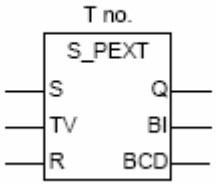


-٣

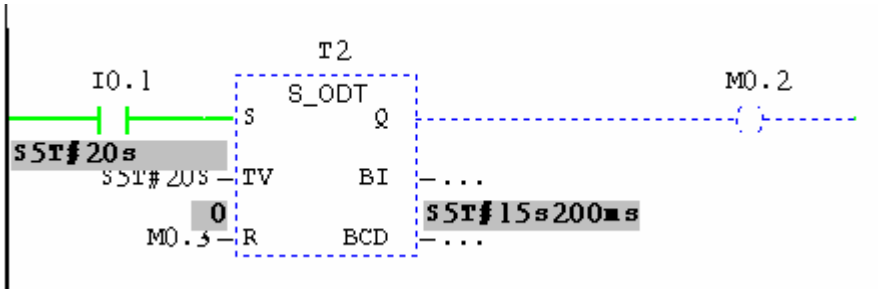
S_ODT On-Delay S5 Timer

نفس النوع الاول ولكن تكون قيمته (Q) تساوي (0) عند بدأ التايمر بالعمل وتصبح (1) بعد انتهاء الزمن

English



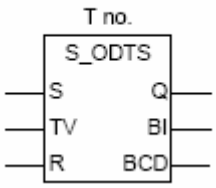
Parameter English	Parameter German	Data Type	Memory Area	Description
T no.	T-Nr.	TIMER	T	Timer identification number; range depends on CPU
S	S	BOOL	I, Q, M, L, D	Start input
TV	TW	S5TIME	I, Q, M, L, D	Preset time value
R	R	BOOL	I, Q, M, L, D	Reset input
BI	DUAL	WORD	I, Q, M, L, D	Remaining time value, integer format
BCD	DEZ	WORD	I, Q, M, L, D	Remaining time value, BCD format
Q	Q	BOOL	I, Q, M, L, D	Status of the timer



٤-

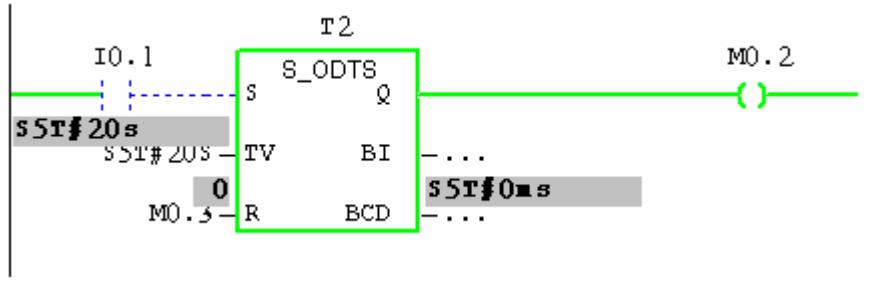
S_ODTS Retentive On-Delay S5 Timer

English



هذا الايعاز يجمع بين الايعاز الثاني والثالث حيث ان قيمة (Q) تساوي (0) عند بدأ التايمر بالعمل وتصبح (1) بعد انتهاء الزمن وايضا(Q) تبقى قيمته (1) حتى لواصبحت قيمة (S) تساوي (0)

Parameter English	Parameter German	Data Type	Memory Area	Description
T no.	T-Nr.	TIMER	T	Timer identification number; range depends on CPU
S	S	BOOL	I, Q, M, L, D	Start input
TV	TW	S5TIME	I, Q, M, L, D	Preset time value
R	R	BOOL	I, Q, M, L, D	Reset input
BI	DUAL	WORD	I, Q, M, L, D	Remaining time value, integer format
BCD	DEZ	WORD	I, Q, M, L, D	Remaining time value, BCD format
Q	Q	BOOL	I, Q, M, L, D	Status of the timer

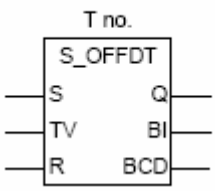


٥-

S_OFFDT Off-Delay S5 Timer

نفس النوع الاول ولكن يعمل التايمر عن تغير قيمة (S) من (1) الى (0)

English



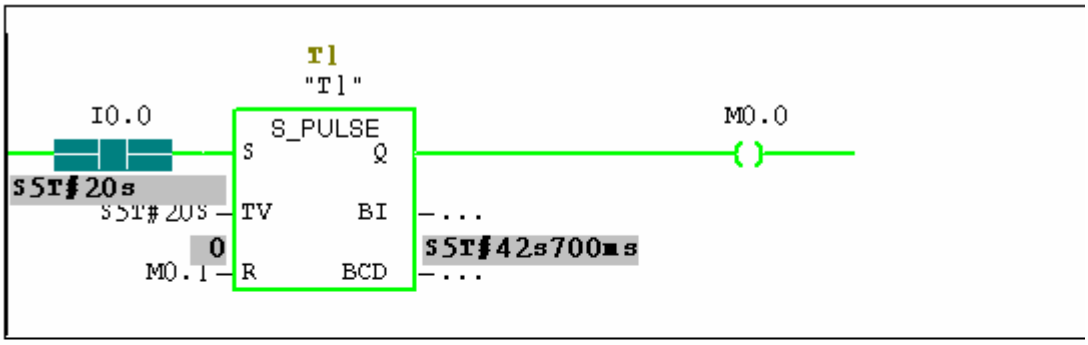
Parameter English	Parameter German	Data Type	Memory Area	Description
T no.	T-Nr.	TIMER	T	Timer identification number; range depends on CPU
S	S	BOOL	I, Q, M, L, D	Start input
TV	TW	S5TIME	I, Q, M, L, D	Preset time value
R	R	BOOL	I, Q, M, L, D	Reset input
BI	DUAL	WORD	I, Q, M, L, D	Remaining time value, integer format
BCD	DEZ	WORD	I, Q, M, L, D	Remaining time value, BCD format
Q	Q	BOOL	I, Q, M, L, D	Status of the timer

-٦

---(SP) Pulse Timer Coil

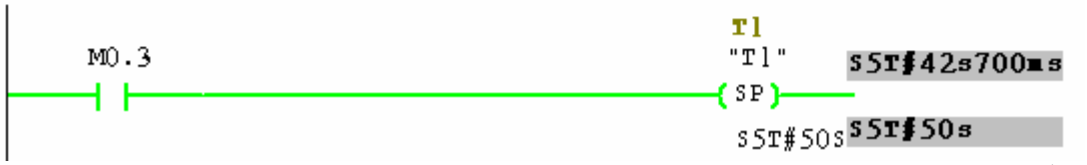
يعمل هذا الايعاز على اعطاء قيمة زمنية جديدة للتايمر مع اهمال القيمة القديمة ولكن لا يغير القيمة والتايمر اثناء العمل ولكن ينتظر زمن التايمر القديم ينتهي ثم يقوم باعطاء التايمر قيمة زمنية جديدة

English
<T no.>
---(SP)
<time value>



Network 2 : Title:

Comment:



باقي الايعازات تقريبا نفس العمل

اليوم العاشر
١-الايعازات المنطقية الجزءالثالث
والآن سنذكر مختصر لايعازات (Step7)

Table 4-10. LAD/FBD Bit-Logic Instructions — Basic Operations

LAD	FBD	Brief Description
Output ??.? —( )—	Output ??.? ??.?— =	If the result of the logic operation (RLO) preceding the referenced output address = logic 1 then assign logic 1 to the output address, otherwise assign logic 0 to the output.
Normally-Open Contact ??.? —| |—	Input ??.? —|	Examine the referenced address for a logic 1 status. A status of logic 1 allows logic continuity; logic 0 interrupts continuity.
Normally-Closed Contact ??.? —|/|—	Negate Input ??.? —○|	Examine the referenced address for a logic 0 status. A status of logic 0 allows logic continuity; logic 1 interrupts continuity.
Set Output ??.? —(S)—	Set Output ??.? ??.?— S	If the result of the logic operation (RLO) preceding the referenced output transitions from logic 1 -to- logic 0, then assign logic 1 to the address. Remain at logic 1 until reset.
Reset Output ??.? —(R)—	Reset Output ??.? ??.?— R	When the result of logic operations (RLO) preceding the referenced output transitions from logic 1 -to- logic 0, then assign logic 0 to the address, Remain at logic 0 until set again.
Invert RLO —| NOT |—	Negate RLO —○	Invert the RLO (result of logic operation) status at the point at which the instruction is inserted (i.e., logic 0 -to- logic 1; logic 1 -to- logic 0).

Table 4-11. LAD/FBD Bit-Logic Instructions — Special Operations

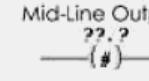
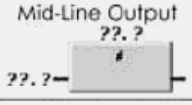
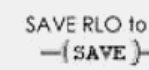
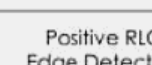
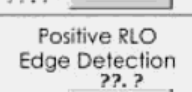
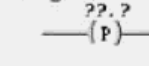
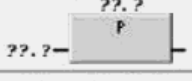
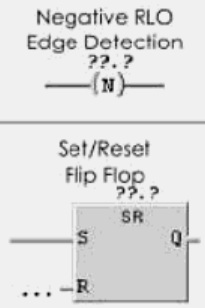
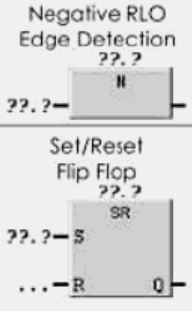
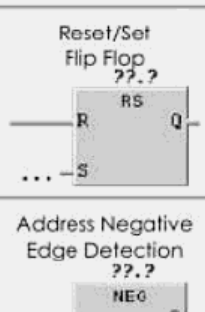
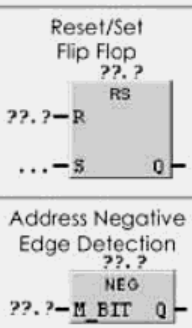
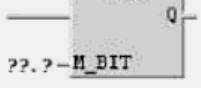
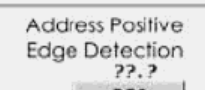
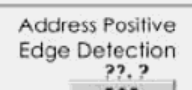
LAD	FBD	Brief Description
Mid-Line Output 	Mid-Line Output 	Stores intermediate result of logic operation (RLO) for the logic circuit up to and preceding the point of insertion.
SAVE RLO to BR 	SAVE RLO to BR 	Saves result of the preceding logic operation (RLO), to the binary result bit of the status word.
Positive RLO Edge Detection 	Positive RLO Edge Detection 	Detects 0-to-1 transition resulting from the preceding logic operation (RLO). An edge detection is signaled by the output going high for one CPU scan (single pulse).
Negative RLO Edge Detection 	Negative RLO Edge Detection 	Detects 1-to-0 transition resulting from the preceding logic operation (RLO). An edge detection is signaled by the output going high for one CPU scan (single pulse).
Set/Reset Flip Flop 	Set/Reset Flip Flop 	Implements Set/Reset flip-flop, giving priority to the Reset function if both **S** and **R** input lines go TRUE simultaneously.
Reset/Set Flip Flop 	Reset/Set Flip Flop 	Implements Reset/Set flip-flop, giving priority to the Set function if both **R** and **S** input lines go TRUE simultaneously.
Address Negative Edge Detection 	Address Negative Edge Detection 	Detect a negative-edge transition of a specific bit address. An edge detection is signaled by the output going high for one CPU scan (single pulse).
Address Positive Edge Detection 	Address Positive Edge Detection 	Detects a positive-edge transition of a specific bit address. An edge detection is signaled by the output going high for one CPU scan (single pulse).

Table 4-12. LAD/FBD Counter Instructions Summary

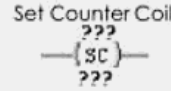
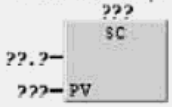
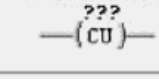
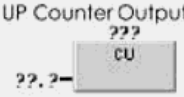
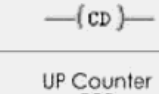
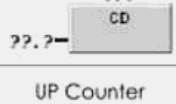
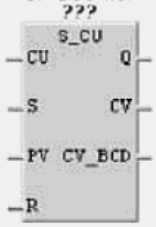
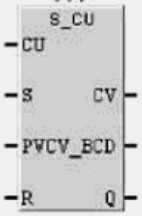
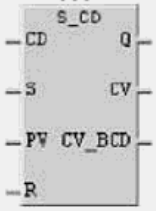
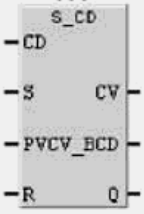
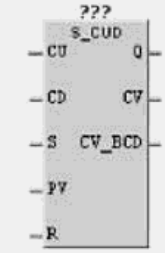
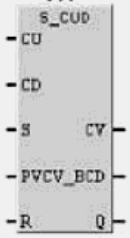
LAD	FBD	Brief Description
Set Counter Coil 	Set Counter Output 	Whenever the driving logic transitions from logic 0 to logic 1, load the specified preset value **PV** to the addressed counter.
UP Counter Coil 	UP Counter Output 	Whenever the driving logic transitions from logic 0 -to- logic 1, increment the specified counter by one count.
DOWN Counter Coil 	DOWN Counter 	Whenever the driving logic transitions from logic 0 -to- logic 1, decrement the specified counter by one count.
UP Counter 	UP Counter 	Whenever the logic input line **CU** transitions from logic 0 -to- logic 1, increment the specified counter by one count. Logic 1 on the reset input line **R** resets the counter to zero.
Down Counter 	Down Counter 	Whenever the logic input line **CD** transitions from logic 0 -to- logic 1, decrement the specified counter by one count. Logic 1 on the reset input line **R** resets the counter to zero.
UP-Down Counter 	UP-Down Counter 	For each logic 0 -to- logic 1 transition on the **CU** input line, increment the specified counter by +1; for each logic 0 -to- logic 1 transition on the **CD** input line, decrement the count by +1. Logic 1 on the reset input line **R** resets the counter to zero.

Table 4-13. LAD/FBD Box Timer Instructions Summary

LAD	FBD	Brief Description
ON-Delay Timer ??? S_ODT S Q TV BI R BCD	ON-Delay Timer ??? S_ODT S BI TV BCD R Q	When the enable input **S** transitions from 0 -to- 1, the timer starts timing and continues to time unless the enable goes false or the reset input **R** transitions from 0 -to- 1. The output **Q** is activated after the preset time has elapsed, and stays energized until the enable signal transitions from 1 -to- 0 or the timer is reset.
OFF-Delay Timer ??? S_OFFDT S Q TV BI R BCD	OFF-Delay Timer ??? S_OFFDT S BI TV BCD R Q	When the enable input **S** transitions from 0 -to- 1, the timer output **Q** is activated. The timer starts timing when the enable signal **S** transitions from 1-to- 0. The timer will run until the programmed delay expires, unless the reset signal **R** transitions from 0 -to- 1. After the timed delay expires, the timer output **Q** is de-activated. The output Q is also de-activated whenever the timer is reset.
Retentive ON-Delay Timer ??? S_ODTS S Q TV BI R BCD	Retentive ON-Delay Timer ??? S_ODTS S BI TV BCD R Q	When the enable input **S** transitions from 0 -to- 1, the timer starts timing and continues to time and continues to time even if the enable signal **S** transitions from 1-to- 0. If the enable signal changes back to '1' before the timer expires, the timer will restart. The output **Q** is activated after the preset time has elapsed, and remains activated until the timer is reset.
Pulse Timer ??? S_PULSE S Q TV BI R BCD	Pulse Timer ??? S_PULSE S BI TV BCD R Q	When the enable input **S** transitions from 0 -to- 1, the timer starts timing and continues to time for as long as the **S** input is '1', or until the programmed delay expires. While the timer is running, the output **Q** is activated for as long as the enable signal **S** is activated. If the enable signal **S** returns to '0' before the timer expires, the timer is stopped and the output Q is de-activated. While the timer is running, a transition from 0 -to- 1 on the reset line **R** resets the timer.
Extended Pulse Timer ??? S_PEXT S Q TV BI R BCD	Extended Pulse Timer ??? S_PEXT S BI TV BCD R Q	When the enable input **S** transitions from 0 -to- 1, the timer starts timing and continues to time until the programmed delay expires, even if the input **S** returns to '0'. The output **Q** is activated whenever the timer is running. While the timer is running, a transition from 0 -to- 1 on the reset line **R** resets the timer.

Table 4-14. LAD/FBD Conversion Instructions Summary

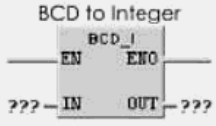
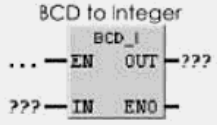
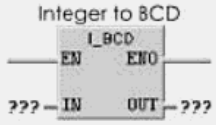
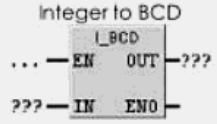
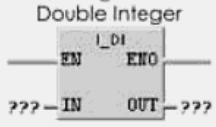
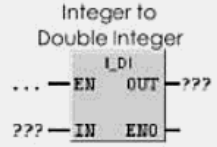
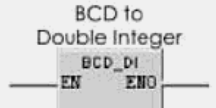
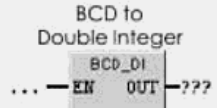
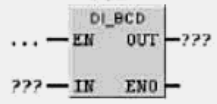
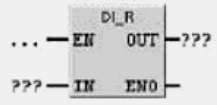
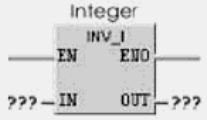
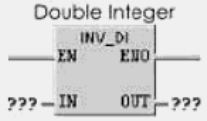
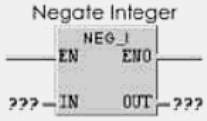
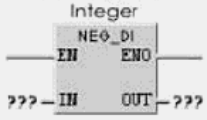
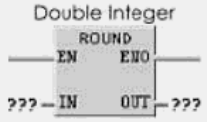
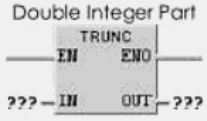
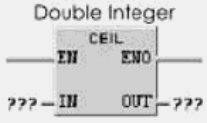
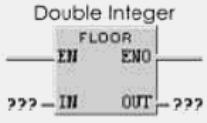
LAD	FBD	Brief Description
BCD to Integer 	BCD to Integer 	When **EN** is at logic 1, convert the 3-digit BCD value at supplied at **IN** to a 16-bit INT value. Put the result in the location specified at **OUT**.
Integer to BCD 	Integer to BCD 	When **EN** is at logic 1, convert the 16-bit INT value supplied at **IN** to a 3-digit BCD value from 000-999. Put the result in the location specified at **OUT**.
Integer to Double Integer 	Integer to Double Integer 	When **EN** is at logic 1, convert the 16-bit INT value supplied at **IN** to a 32-bit DINT value. Put the result in the location specified at **OUT**.
BCD to Double Integer 	BCD to Double Integer 	When **EN** is at logic 1, convert the 7-digit BCD value supplied at **IN** to a 32-bit DINT value. Put the result in the location specified at **OUT**.
Double Integer to BCD 	Double Integer to BCD 	When **EN** is at logic 1, convert the 32-bit DINT value supplied at **IN** to a 7-digit BCD value. Put the result in the location specified at **OUT**.
Double Integer to REAL 	Double Integer to REAL 	When **EN** is at logic 1, convert the 32-bit DINT value supplied at **IN** to a REAL value (floating-point). Put the result in the location specified at **OUT**.

Table 4-15. LAD/FBD Conversion Instructions Summary (Continued).

LAD	FBD	Brief Description
Ones Complement Integer 		When **EN** is at logic 1, complement or invert the integer value supplied at **IN** and **put** the result in the location specified at **OUT**.
Ones Complement Double Integer 		When **EN** is at logic 1, complement or invert the double integer value supplied at **IN** and **put** the result in the location specified at **OUT**.
Negate Integer 		When **EN** is at logic 1, negate the integer value supplied at **IN** (change sign of value; positive-to-negative, negative-to-positive). Put the result in the location specified at **OUT**.
Negate Double Integer 		When **EN** is at logic 1, negate the double integer value supplied at **IN** (positive-to-negative, negative-to-positive). Put the result in the location specified at **OUT**.
Negate REAL 		When **EN** is at logic 1, negate the REAL value supplied at **IN** (positive-to-negative, negative-to-positive). Put the result in the location specified at **OUT**.
Round to Double Integer 		When **EN** is at logic 1, convert the REAL value supplied at **IN** to Integer by Rounding to nearest Double Integer value. Put the result in the location specified at **OUT**.
Truncate Double Integer Part 		When **EN** is at logic 1, convert the REAL value supplied at **IN** to Integer by Truncating the fractional part of the REAL value. Put the result in the location specified at **OUT**.
Ceiling to Double Integer 		When **EN** is at logic 1, convert the REAL value supplied at **IN** to DINT value by Rounding to lowest DINT value greater-than-or-equal to **IN**. Put the result in the location specified at **OUT**.
Floor to Double Integer 		When **EN** is at logic 1, convert the REAL value supplied at **IN** to DINT value by Rounding to highest DINT value less-than-or-equal to **IN**. Put the result in the location specified at **OUT**.

Note: When EN is '1' ENO will also be '1' except on error; then while EN is '1'; ENO is '0'.

Table 4-17. Example LAD/FBD Integer, Double Integer, and REAL Addition Instructions.

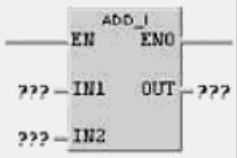
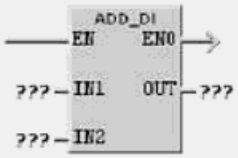
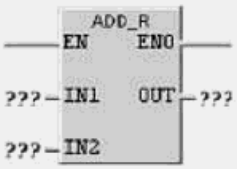
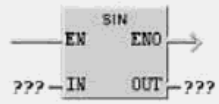
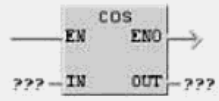
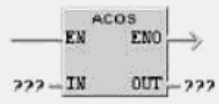
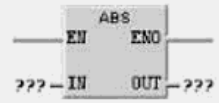
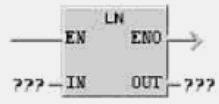
IN1/IN2/OUT	INT	DINT	REAL
			
Constants (in permissible range)	-32,768 to +32,767	-214,783,648 to +214,783,647	± 1.75495e-38 to ± 3.402823+e38
S7 Memory Locations (Absolute or Symbolic)	IW 28 QW 42 MW 54 DB6.DBW12	ID 28 QD 42 MD 54 DB6.DBD12	ID 28 QD 42 MD 54 DB6.DBD12
Declared Variables (of correct data type)	#Value_1 #Value_2 #Total	#Value_1 #Value_2 #Total	#Value_1 #Value_2 #Total

Table 4-16. STEP 7 Integer and REAL Arithmetic Instructions

	LAD/FBD Arithmetic Instructions Using Data Type		
	INT	DINT	REAL
	Permissible Range	Permissible Range	Permissible Range
Operation	-32,768 to +32,767	-214,783,648 to +214,783,647	± 1.75495e-38 to ± 3.402823+e38
Addition	ADD_I	ADD_DI	ADD_R
Subtraction	SUB_I	SUB_DI	SUB_R
Multiplication	MUL_I	MUL_DI	MUL_R
Division with Quotient Result	DIV_I	DIV_DI	DIV_R
Division with Remainder	-	MOD_DI	-

Table 4-18. LAD/FBD: Standard Math Functions (REAL number operations).

LAD/FBD	Name	Brief Description
	Sine	When the **EN** signal is at logic 1, find the Sine of the REAL value (radian angle) supplied at **IN**. Put the result in location specified at **OUT**.
	Cosine	When the **EN** signal is at logic 1, find the Cosine of the REAL value (radian angle) supplied at **IN**. Put result in location specified at **OUT**.
	Tangent	When the **EN** signal is at logic 1, find the Tangent of the REAL value (radian angle) supplied at **IN**. Put the result in location specified at **OUT**.
	Arc Sine	When the **EN** signal is at logic 1, find the Arc Sine of the value supplied at **IN** (-1 to +1). Put angle result (radians) in location specified at **OUT**.
	Arc Cosine	When the **EN** signal is at logic 1, find the Arc Cosine of the value supplied at **IN** (-1 to +1). Put angle result (radians) in location specified at **OUT**.
	Arc Tangent	When the **EN** signal is at logic 1, find the Arc Tangent of the value supplied at **IN**. Put angle result (radians)in location specified at **OUT**.
	Absolute Value	When the **EN** signal is at logic 1, find the Absolute Value of the value supplied at **IN**. Put the result in location specified at **OUT**.
	Square Root	When the **EN** signal is at logic 1, find the Square Root of the value supplied at **IN**. Put the result in location specified at **OUT**.
	Square	When the **EN** signal is at logic 1, find the Square of the value supplied at **IN**. Put the result in location specified at **OUT**.
	Natural LOG	When the **EN** signal is at logic 1, find the Natural Log of the value supplied at **IN**. Put the result in location specified at **OUT**.
	Exponent	When the **EN** signal is at logic 1, find the Exponent of the value supplied at **IN**. Put the result in location specified at **OUT**.

Note: When EN is [illegible] ENO will also be [illegible] except an error then while EN is [illegible] ENO is [illegible]

Table 4-19. LAD/FBD Compare Instructions Summary.

	LAD/FBD Compare Instructions Using Data Type		
	INT	**DINT**	**REAL**
	Permissible Range	Permissible Range	Permissible Range
Compare Test	-32,768 to +32,767	-214,783,648 to +214,783,647	± 1.75495e-38 to ± 3.402823+e38
Equal	CMP==I	CMP==D	CMP==R
Not Equal	CMP<>I	CMP<>D	CMP<>R
Less Than	CMP <I	CMP <D	CMP <R
Greater Than	CMP >I	CMP >D	CMP >R
Less Than or Equal	CMP<=I	CMP<=D	CMP<=R
Greater Than or Equal	CMP>=I	CMP>=D	CMP>=R

Table 4-20. Example LAD/FBD Integer, Double Integer, and REAL Compare Equal Instructions

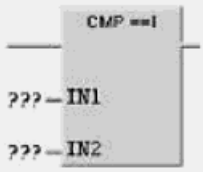
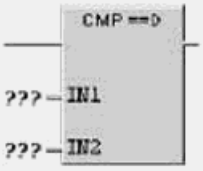
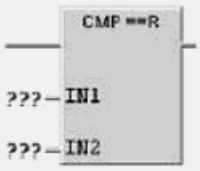
IN1/IN2	INT	DINT	REAL
			
Constants (in permissible range)	-32,768 to +32,767	-214,783,648 to +214,783,647	± 1.75495e-38 to ± 3.402823+e38
S7 Memory Locations (Absolute or Symbolic)	IW 28 QW 42 MW 54 DB6.DBW12	ID 28 QD 42 MD 54 DB6.DBD12	ID 28 QD 42 MD 54 DB6.DBD12
Declared Variables (of correct data type)	#Value_1 #Value_2	#Value_1 #Value_2	#Value_1 #Value_2

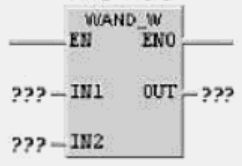
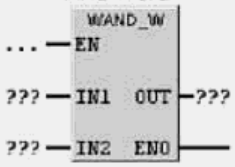
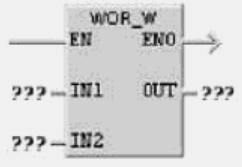
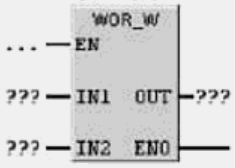
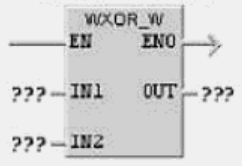
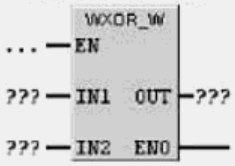
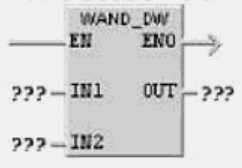
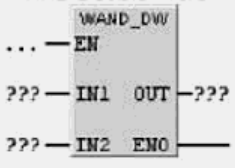
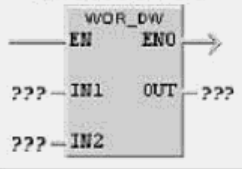
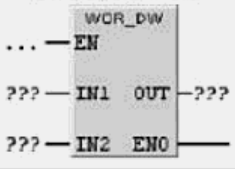
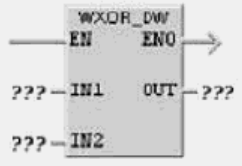
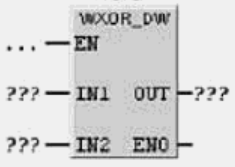
Table 4-21. LAD/FBD Program Flow Control Instructions Summary

LAD	FBD	Brief Description
Label ???	Label ???	Four-character label defining a network to which a **JMP** or **JMPN** directs program execution. Combined with jumps, allows skipping of portions of logic.
Jump ??? —(JMP)—	Jump ??? JMP	Causes internal block jump to labeled network based on conditional or unconditional logic. The jump destination is specified by label address specified above the **JMP** output.
Jump-if-NOT ??? —(JMPN)—	Jump-if-NOT ??? JMPN	When the conditional logic is at logic 0 (RLO = 0), an internal block jump is directed to the network identified by the **label** address specified above the **JMPN** output.
MCR Activate —(MCRA)—	MCR Activate MCRA	This unconditional output, when encountered, enables the use of MCR zones up to the point of the next MCR De-activate instruction. Paired with MCRD instruction.
MCR De-Activate —(MCRD)—	MCR De-Activate MCRD	This unconditional output, when encountered, disables the use of MCR zones up to the point of the next MCR Activate instruction. Paired with MCRA instruction.
MCR ON —(MCR<)—	MCR ON MCR<	This conditional output, when activated by the driving logic, begins an MCR zone. Paired with MCR-OFF (End) instruction.
MCR OFF —(MCR>)—	MCR OFF MCR>	This unconditional output, when encountered, ends an MCR zone. Paired with MCR-ON (Begin) instruction.
Call Block Coil ??? —(CALL)—	Call Block ??? CALL	When the driving logic (conditional or unconditional) is at logic 1, the referenced Function (FC) or System Function (SFC), having no formal parameters, is called for processing.
Open Data Block ??? —(OPN)—	Open Data Block ??? OPN	When this unconditional instruction is encountered, the shared data block (DB) referenced above the instruction is opened, allowing data access to the data locations.
Return Coil —(RET)—	Return Output RET	When the conditional logic driving this output is true, execution of the current block is terminated and control is returned to the calling block; otherwise continue with the following network.

Table 4-22. LAD: Status Bit (Result Bits) Instructions Summary.

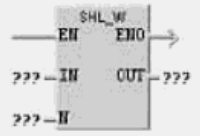
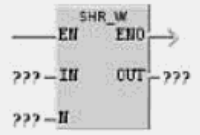
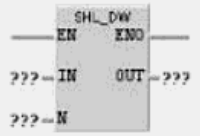
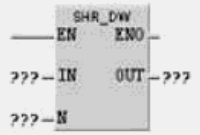
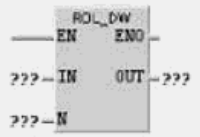
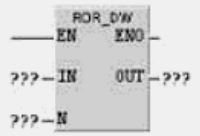
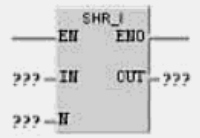
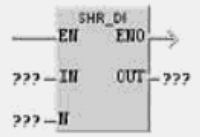
LAD Instructions		Name	Brief Description
NO	NC		
BR	BR	Binary Result Bit	Use the NO contact to check the BR bit for logic 1 to allow power flow; use NC BR bit to check for logic 0 to allow power flow.
UO	UO	Unordered Bit	Check if previous floating-point operation involved at least one invalid (Unordered) floating-point number.
OV	OV	Overflow Bit	Check if previous math operation resulted in a value outside of permissible negative or positive range.
OS	OS	Overflow Bit Stored	Check if previous series of math operations resulted in a value outside of the permissible range.
>0	>0	Greater Than Zero Result Bit	Check result of previous math operation for greater than zero.
<0	<0	Less Than Zero Result Bit	Check result of previous math operation for less than zero. Combine in series or parallel with other contacts.
>=0	>=0	Greater Than or Equal Zero Result Bit	Check result of previous math operation for greater-than-or-equal zero.
<=0	<=0	Less Than or Equal Zero Result Bit	Check result of previous math operation for less than or equal zero.
==0	==0	Equal Zero Result Bit	Check result of previous math operation for equal zero.
<>0	<>0	Not Equal Zero Result Bit	Check result of previous math operation for not equal zero.

Table 4-23. LAD/FBD Word Logic Instructions Summary

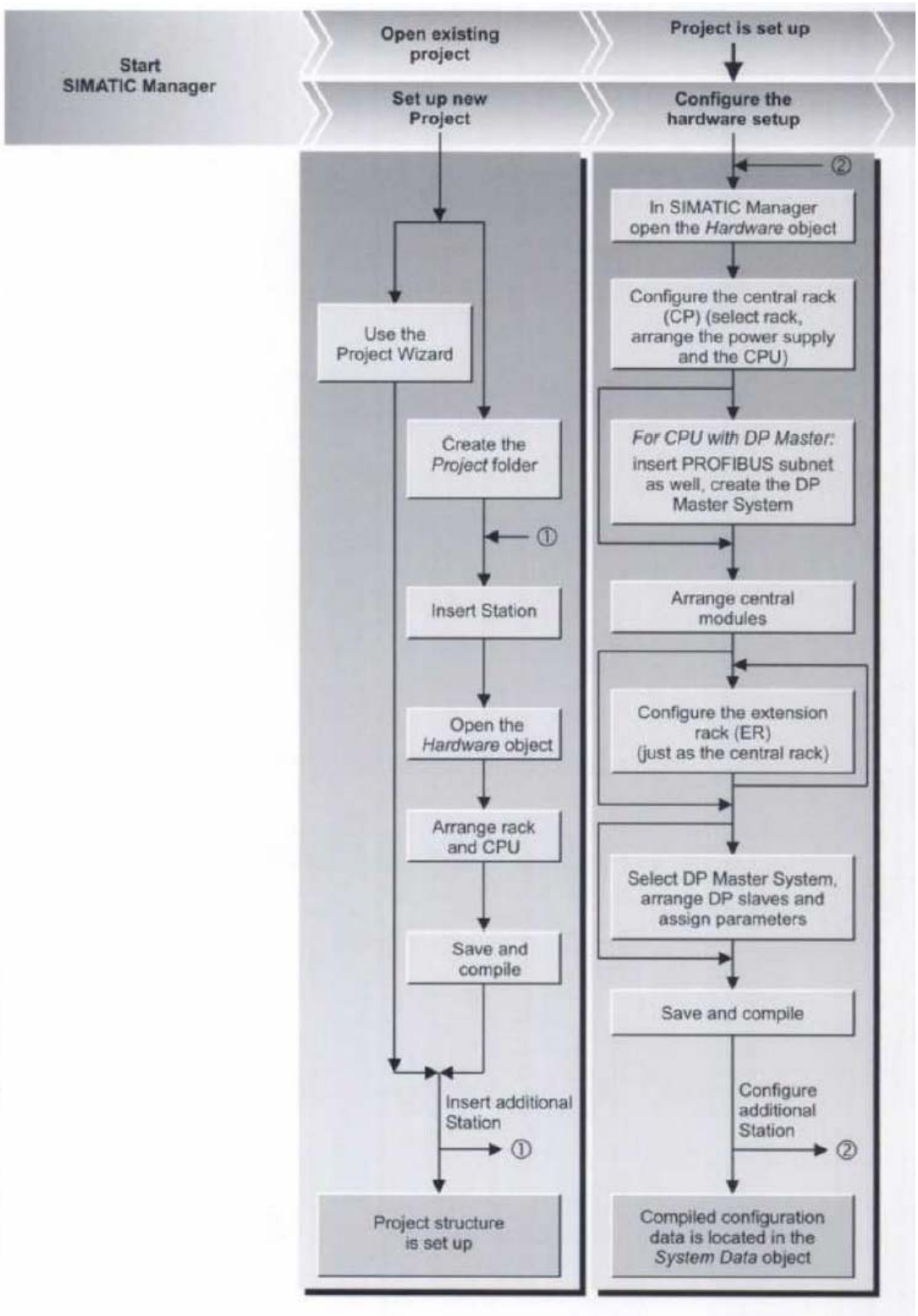
LAD	FBD	Brief Description
AND Word 	AND Word 	When the **EN** signal is at logic 1, perform a bit-by-bit logical AND on the two 16-bit values supplied at (**IN1**/**IN2**). Put result in the location specified at **OUT**.
OR Word 	OR Word 	When the **EN** signal is at logic 1, perform a bit-by-bit logical OR on the two 16-bit values supplied at (**IN1**/**IN2**). Put result in the location specified at **OUT**.
Exclusive OR Word 	Exclusive OR Word 	When the **EN** signal is at logic 1, perform a bit-by-bit logical XOR on the two 16-bit values supplied at (**IN1**/**IN2**). Put result in the location specified at **OUT**.
AND Double-Word 	AND Double-Word 	When the **EN** signal is at logic 1, perform a bit-by-bit logical AND on the two 32-bit values supplied at (**IN1**/**IN2**). Put result in the location specified at **OUT**.
OR Double-Word 	OR Double-Word 	When the **EN** signal is at logic 1, perform a bit-by-bit logical OR on the two 32-bit values supplied at (**IN1**/**IN2**). Put result in the location specified at **OUT**.
Exclusive OR Double-Word 	Exclusive OR Double-Word 	When the **EN** signal is at logic 1, perform a bit-by-bit logical XOR on the two 32-bit values supplied at (**IN1**/**IN2**). Put result in the location specified at **OUT**.

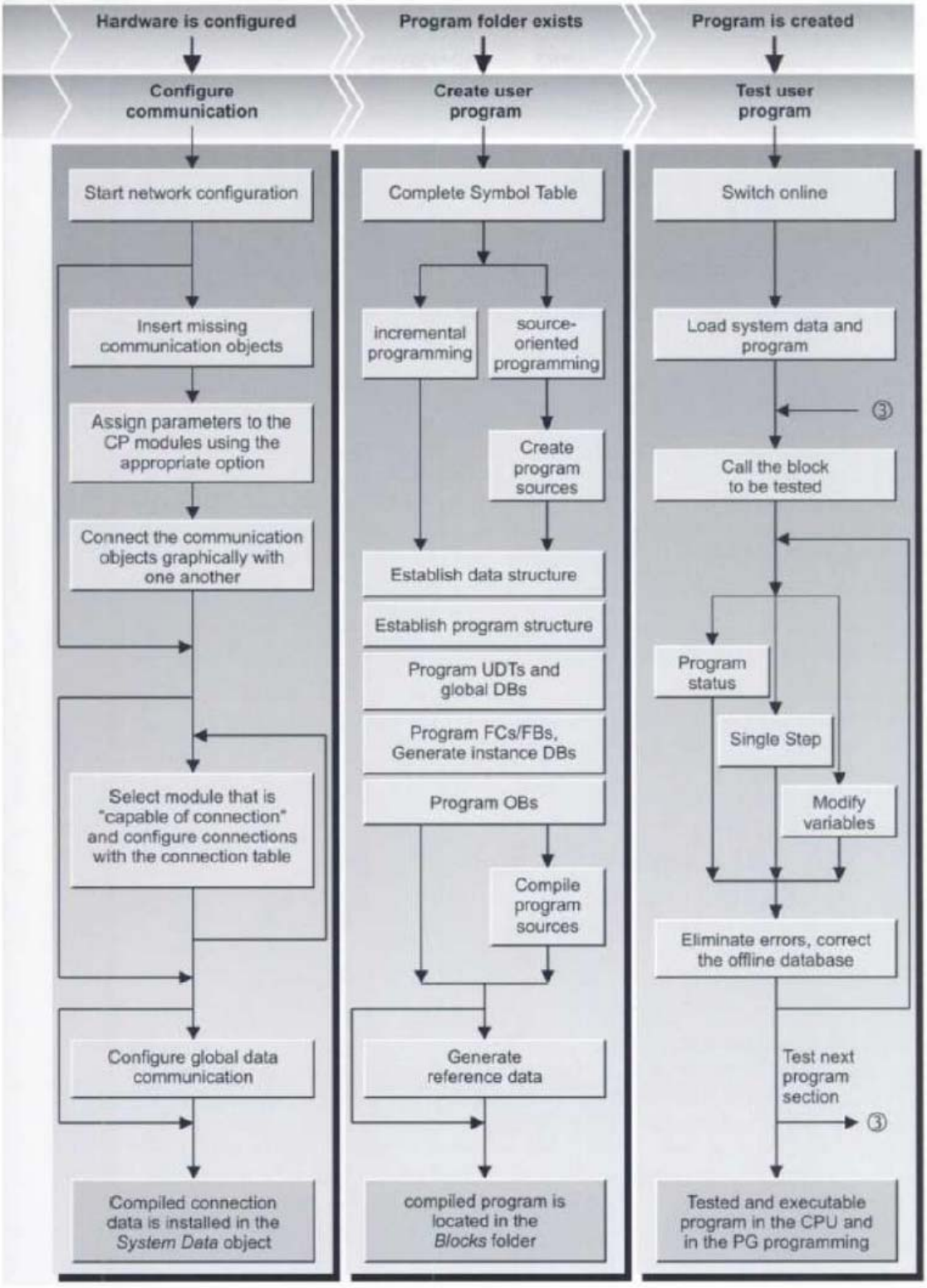
Note: When EN is '1' ENO will also be '1' except an error; then while EN is '1' ENO is '0'.

Table 4-24. LAD Shift-Rotate and Move Instructions

LAD/ FBD	Name	Brief Description
	Shift Left Word	Shift the 16-bits of the Word specified at **IN**, by n-bits to the left.
	Shift Right Word	Shift the 16-bits of the Word specified at **IN**, by n-bits to the right.
	Shift Left Double Word	Shift the 32-bits of the specified Double-Word, by n-bits to the left.
	Shift Right Double Word	Shift 32-bits of the Double-Word specified at **IN**, by n-bits to the right.
	Rotate Left Double Word	Rotate the 32-bits of the Double-Word specified at **IN**, by n-bits to the left.
	Rotate Right Double Word	Rotate the 32-bits of the Double-Word specified at **IN**, by n-bits bits to the right.
	Shift Right Integer	Shift the 16-bits bits of the Integer value specified at **IN**, by n-bits to the right, while maintaining the sign bit.
	Shift Right Double Integer	Shift the 32-bits of the Double Integer value specified at **IN**, by n-bits to the right, while maintaining sign bit.
	Move Word	Copy the variable specified at **IN** to the location specified at OUT.

ملحق١: في هذا الشكل سنبيت هيكلية المشروع المتكامل لبرنـامج (Step7) حتى نستطيع تقييم ما تعلمناه وما تبقى لنا لنتعلمه لاحقا انشاء الله تعالى




Hardware is configured
Program folder exists
Program is created
Configure communication
Create user program
Test user program
Start network configuration
Insert missing communication objects
Assign parameters to the CP modules using the appropriate option
Connect the communication objects graphically with one another
Select module that is "capable of connection" and configure connections with the connection table
Configure global data communication
Compiled connection data is installed in the System Data object
Complete Symbol Table
incremental programming
source-oriented programming
Create program sources
Establish data structure
Establish program structure
Program UDTs and global DBs
Program FCs/FBs, Generate instance DBs
Program OBs
Compile program sources
Generate reference data
compiled program is located in the Blocks folder
Switch online
Load system data and program
③
Call the block to be tested
Program status
Single Step
Modify variables
Eliminate errors, correct the offline database
Test next program section
③
Tested and executable program in the CPU and in the PG programming