DCS Experts
توسعه دانش سیستم های کنترل صنعتی

SAFETY FIRST

SIEMENS

آموزش طراحی و پیاده‌سازی سیستم S7400FH

SIMATIC Fail safe & fault Tolerrant

PROFINET® Microsoft Windows TÜV MANAGEMENT SERVICE ISO 9001

industrial ethernet PROFIBUS®

WWW.AMANDEGARI.IR

کارشناس ارشد برق @yahoo.com
کارشناس و مدرس دوره‌های زیمنس -2734

فصل – ۱ مقدمه ۱
۱–۱– دلایل بکارگیری سیستم‌های H-PLC (Motivation) ۱
1-1-1- حوزه‌هاي كاربردی اتوماسیون S7-400H ۱
1-2- ایستگاه اتوماسیون (AS) ۲
1-2-1- سامانه اتوماسیون استاندارد (سری AS400) ۴
1-2-2- سامانه اتوماسیون برای دسترسی‌پذیری بالا (AS400H) ۴
1-2-3- سامانه اتوماسیون ایمن در خطا (Fail-Safe AS400FH) ۶
فصل – ۲ – سامانه AS-400 استاندارد ۷
2-1- اجزای AS-400 استاندارد ۷
۲–۲– انواع رک S7-400 ..... ۸
2-3- انواع باس در رک S7-400 ..... ۹
فصل – ۳ – ساختار و معماری سامانه S7-400H ۱۱
۳–۱– مقدمه ۱۲
۳–۲– معماری سیستم (System architecture) ۱۲
۳–۳– اجزای سخت‌افزاری یک سیستم S7-400H (Minimum Configuration) ۱۳
3-3-1- ماژول کنترل‌کننده (CPU) ۱۴
3-3-2- ماژول‌هاي سنكرون سازی (SYNC Modules) ۱۷
3-3-3- كابل فيبر نوري ۱۸
۳–٤– مشخصه‌های افزونگی (Redundancy features) ۱۸
3-4-1- تعویض بی‌درنگ بین دو کنترل‌کننده (Bumpless Master-Stand-by switchover) ۱۸
3-4-2- سنکرون سازی بین دو CPU (Synchronization Procedure) ۱۸
3-4-3- توابع خودآزمایی (self-test) ۲۱
3-4-4- برنامه‌نویسی آنلاین (Online programming) ۲۲
3-4-5- اتصال به PG ۲۲
3-4-6- قابلیت تعمیر آنلاین (Online Repair) ۲۳
3-4-7- قابلیت CiR (Configuration Changes in Run Mode) ۲۴
3-4-8- پیکربندی کنترل‌کننده (Configuration) ۲۵

3-5- پیاده‌سازی سیستم H 400 (System integration) ۲۷
3-5-1- برنامه‌نویسی ۲۷
3-5-2- پیکربندی سخت افزاری (Hardware configuration) ۲۸
۳-۶- افزونگی (Redundancy principle) ۲۸
3-6-1- افزونگي سخت‌افزاري (Hardware Redundancy) ۲۹
3-6-2- افزونگی با اجزاءیکسان (homogeneous redundancy) ۳۰
3-6-3- افزونگی پسیو (Passive redundancy) ۳۰
3-6-4- افزونگی فعال (Active redundancy) ۳۰
۳-۷- پیکربندی و اتصال I/O در سیستم H (I/O configuration) ۳۰
3-7-1- اصول I/O افزونه ۳۲
3-7-2- افزونگي با I/O هاي مبتنی بر شبکه پروفی‌باس (FMR-Flexible Modular Redundancy) ۳۲
3-7-3- پیکربندی استفاده از یک کانال I/O واحد بدون ساختار افزونه (Single Mode) ۳۳
3-7-4- استفاده از ماژول‌هاي I/O افزونه با یک سیستم H بدون افزونه (H-CPU in single mode) ۳۳
3-7-5- استفاده از ماژول‌هاي I/O مشترك بین دو CPU (Distributed Switched I/O) ۳۳
3-7-6- استفاده از ماژول‌های I/O افزونه در رک مرکزی (Central I/O Modules) دو سیستم H افزونه ۳۴
3-7-7- ساختار I/O افزونه توزیع شده (Distributed I/O modules) ۳۵
3-7-8- ساختار افزونه کامل برای I/O برای دسترس‌پذیری خیلی بالا ۳۵
۳-۸- سیسم‌بندی کانال‌های افزونه (Wiring) ۳۸
3-8-1- سیسم‌بندي کانال‌های ورودي دیجیتال افزونه (Wiring) ۳۸
3-8-2- سیسم‌بندي کانال‌های خروجی دیجیتال افزونه ۳۹
3-8-3- سیسم‌بندي کانال‌های ورودي آنالوگ افزونه ۴۰
3-8-4- سیسم‌بندي کانال‌های خروجی‌های آنالوگ افزونه ۴۰
3-8-5- سطوح کیفیت افزونگي I/O (Redundant quality stages) ۴۱
۳-۹- یکپارچگی با برنامه کاربر ۴۱
3-9-1- پیکربندی سخت‌افزاری برای I/O افزونه ۴۱
3-9-2- نحوه Redundant کردن یک ET200M (IM153-3) بین دو station در software Redundancy ... ۴۲
3-10- شماره سفارش ماژول، رک و باس یونیت‌ها در HWConfig ***Error! Bookmark not defined.***

3-10-1- کتابخانه بلوك‌هاي I/O افزونه (Block library Functional I/O Redundancy) ۴۳
3-10-2- اتصال PROFIBUS PA از طريق يك لينك PA به سيستم H ۴۳
3-10-3- ماژول Y-Link ۴۴
3-10-4- واسط يا گذرگاه ارتباطی I/O ۴۵
۳-۱۱- ارتباطات افزونه (Redundant communication Principle) ٤٦
3-11-1- ارتباطات از طريق اتصالات تحمل‌پذير خطا (Fault-tolerant S7 Connections) ۴۸
۳-۱۲- قابليت اطمينان ماژول‌های زيمنس (Reliability of modules) ٤٩
۳-۱۳- نصب و راه‌اندازی يک سيستم S7400H ٤٩
3-13-1- مانيتورينگ سگنال‌های سيستم H ۵۱
۳-۱٤- توصيف فانکش‌بلاک‌ها ٥٤
3-14-1- فانکشن بلاک RedAn02 ..... ۵۴
فصل ٤ – ساختار و معماری سيستم‌های FH ٥٥
4-1- مقدمه ٥٥
4-1-1- تاريخچه سيستم‌های ايمنی زيمنس (History of Siemens Safety Systems) ۵۵
4-1-2- تفاوت بين سيستم SIS و سيستم کنترل پايه BPCS ۵۶
4-1-3- هدف از سيستم‌های SIS ۵۶
4-1-4- ريسک چيست و چه کسی تصميم می‌گيرد که چه ريسکی قابل قبول است؟ ۵۷
4-1-5- استانداردهای مهم مرتبط با ايمنی ۵۸
٤-۲- سخت‌افزار سازگار با سطوح SIL 1 , SIL2, SIL3 ..... ٥٨
4-2-1- انتخاب يک CPU برای يک پلنت فرآيندی ۵۹
4-3- سيستم‌های اتوماسيون خرابی امن FH (Fail Safe Automation System) ٥٩
4-3-1- اصول پشت پرده عملکرد ايمنی(Principle behind the Safety Function) ۶۰
4-3-2- پيکربندی‌های پايه برای تامين ايمنی و دردسترس‌پذيری (Basic Configuration Variants) .. ۶۰
٤-٤- معماری‌های سيستم Fail Safe ٦٢
4-4-1- افزونگی موردنياز ۶۳
4-4-2- معماري سيستم تک سنسور يا ارزيابی 1oo1 (1 Out of 1) ۶۳
4-4-3- ارزيابی 1oo2 (1 Out of 2) ۶۴
4-4-4- معماري سيستم 2 Out of 2 (2oo2) ۶۶

4-4-1- معماري سيستم (2oo3-2 Out of 3) ۶۶
4-4-2- لاجیک ارزیابی 2oo3 بدون درنظر گرفتن خطای کانال ۶۸
4-4-3- پیکربندی 2oo3 با در نظرگرفتن خطای کانال (2oo3D) ۶۹
4-4-4- معماری استفاده از سه سنسور با دو ماژول افزونه (Three sensors with redundant F-AI) ۶۹
4-4-5- معماری 1oo1D (with diagnostics – Non Redundant Safety System) ۷۰
4-4-6- معماری 1oo2D (with diagnostics) ۷۱
4-4-7- معماری طراحی سیستم S7-400F (Non-Redundant Setup) ۷۱
4-4-8- معماری طراحی سیستم S7-400HF (Fully-Redundant Setup) ۷۱
٤-٥- مثال‌هایی از پاسخ سیستم تحمل‌خطا (*Fault-Tolerant*) به یک فالت ۷۲
4-5-1- مثال 1- توقف CPU یا ماژول تغذیه ۷۲
4-5-2- مثال 2- خطا یا نقص در کابل فیبر نوري (Failure of a fiber-optic) ۷۲
٤-٦- انواع سیستم *F* ۷۳
٤-٧- نصب بسته *F System* ۷۳
4-7-1- نصب بسته F system مربوط به HMI ۷۳
٤-٨- تنظیمات پیشرفته در *PCS7* (*Advanced PCS 7 ES settings*) ۷٤
4-8-1- حفاظت از دسترسی (Access Protection) ۷۴
4-8-2- تنظیمات CFC برای کامپایل و دانلود ۷۴
٤-٩- پیکربندی سخت افزار *S7F/FH* ۷۵
4-9-1- تطبیق پارامترهای CPU ۷۵
4-9-2- سربرگ Cyclic interrupts ۷۵
٤-١٠- تنظیمات پارامترهای ماژول‌های سیگنال *F* ۷٦
4-10-1- تنظیمات سخت‌افزاری (DIP switch settings) ۷۷
4-10-2- زمان مانیتورینگ F ۷۸
4-10-3- مفهوم Passivation در حالت کلی (غیرفعال شدن کانال/ماژول‌های ورودی/خروجی) ۷۹
4-10-4- پسیو شدن گروهی (Group passivation) ۸۱
4-10-5- فعال کردن کانال‌ها (Activating channels) ۸۲
٤-١١- ارزیابی سنسور (Sensor evaluation) ۸٤
4-11-1- معماری‌های سیم‌بندی و انتخاب برای ET 200M (Wiring and Voting) ۸۴

4-11-2- سیستم انتخاب 2oo3 با ماژول آنالوگ ورودی F (2oo3 Voting with F-AI)..... ۱۶۲
فصل – ۵ – برنامه‌نویسی بلاک‌های F (Programming)..... ۸۶
۵–۱– مقدمه..... ۸۶
5-1-1- ایجاد برنامه/لاجیک F (Configuring the safety program)..... ۸۶
5-1-2- ساختاربرنامه ایمنی (safety program)..... ۸۷
5-1-1- قوانین ساختار برنامه F (Rules governing program structure)..... ۸۹
5-1-2- اجرای بلاک‌های F (Run sequence of F-blocks)..... ۸۹
5-1-3- تبادل داده بین لاجیک F و برنامه استاندارد(Data exchange)..... ۹۰
5-1-4- روند پایه برای ایجاد برنامه ایمنی..... ۹۰
5-1-5- زمان مانیتورینگ سیکل F (F-cycle monitoring)..... ۹۲
5-1-6- قرار دادن گروه‌های F-Runtime در چارت CFC..... ۹۳
5-1-7- گروه‌های F-Shutdown برنامه ایمنی..... ۹۳
۵–۲– درج بلاک‌های F و اتصال آنها به همدیگر..... ۹۳
۵–۳– مکانیزم‌های ایمنی..... ۷٤
5-1-3- تعریف رمز و حفاظت از دسترسی در CPU..... ۱۲۷
5-4- پروسه F startup & (re)start Protection..... ۹٤
5-4-1- راه اندازی F (F-Startup)..... ۹۴
5-4-2- (Re)Start Protection..... ۹۴
فصل – ٦ – کتابخانه سیستم F..... ۹٦
٦–۱– مقدمه..... ۹٦
6-1-1- نوع داده F (F-Data types)..... ۹۶
6-1-2- دسترسی به ورودی/خروجی‌های نوع F (F-I/O access)..... ۹۷
٦–۲– مجموعه بلاک‌های دایور کانال F (F-Channel drivers)..... ۹۸
6-2-1- بلاک درایور کانال دیجیتال ورودی: F_CH_DI..... ۹۹
6-2-2- بلاک درایور کانال دیجیتال خروجی (F_CH_DO)..... ۱۰۲
6-2-3- بلاک درایور F_CH_BI..... ۱۰۳
6-2-4- بلاک درایور قرائت مقدار DINT از یک آدرس DP Slave (F_CH_DII)..... ۱۰۴
6-2-5- بلاک درایور نوشتن مقدار DINT در یک آدرس DP Slave (F_CH_DIO)..... ۱۰۴

6-2-6- بلاک درایور کانال ورودی آنالوگ F_CH_AI ۱۰۵
6-2-7- بلاک درایور کانال آنالوگ خروجی ۱۰۸
۶-۳- مجموعه *BIT_LGC* – بلاک‌های لاجیک *F* با نوع داده *BOOL* ۱۰۹
6-3-1- بلاک F_2OUT3 ..... ۱۰۹
6-3-2- بلاک F_XOUTY ۱۱۰
۶-۴- بلاک‌های مقایسه کننده (*comparing two input values*) ۱۱۰
6-4-1- بلاک مقایسه کننده F_CMP_R ۱۱۱
۶-۵- بلاک‌های *Voter* برای نوع داده *BOOL* و *REAL* (Voter blocks) ۱۱۲
6-5-1- بلاک ارزیابی 2oo3 برای ورودی‌های BOOL (F_2oo3DI) ۱۱۲
6-5-2- بلاک F_2oo3AI ۱۱۳
6-5-3- بلاک RED_IN ۱۱۳
۶-۶- مجموعه *MATH_FP* – بلاک‌های محاسباتی ۱۱۳
6-6-1- بلاک F_ABS_R (F_ABS_R: Absolute value of a REAL value) ۱۱۴
6-6-2- بلاک (F_1oo2_R) ۱۱۵
6-6-3- بلاک انتخاب حداکثر مقدار بین سه ورودی (F_MAX3_R) ۱۱۶
6-6-4- بلاک انتگرال‌گیر F_INT_P ۱۱۷
۶-۷- بلاک‌های تبدیل داده (*data conversion*) ..... ***Error! Bookmark not defined.***
6-7-1- بلاک F_SWC_BO ۱۱۸
6-7-2- بلاک F_FR_FDI ۱۱۹
6-7-3- بلاک F_TON - تایمر تاخیر در روشن (F_TON: Timer switch-on delay) ۱۱۹
6-8- بلاک‌های مبدل (*Convert*) ۱۱۸
6-8-1- بلاک F_QUITES ۱۱۸
۶-۹- بلاک‌های فلیپ‌فلاپ ۱۲۶
6-9-1- تصدیق خطا توسط کاربر (Acknowledgement by ACK_REI) ۱۱۹
فصل ۷ – پیکربندی ماژول‌های *IO* ریداداانت در *PCS7* ۱۳۶
7-1-1- دیاگرام اتصالات ماژول خروجی آنالوگ ۱۳۷
7-2- پیکربندی *IO* ریداندانت در *Step7* ۱۴۷
7-2-1- تنظیم پارامترهای CPU ۱۴۷

فصل – ۸ ..... ۱۴۸
8-1-1- فانکشن‌بلاک‌های سیستمی پشتیبانی شده در ماژول CPU (Supported SFB & SFC) ۱۶۲
۸–۲– اتصال S7 REDCONNECT ۱۶۳
8-1-2- پیکربندی اتصالات RedConnect ۱۶۴
8-3- Using SFC 14 and SFC15 (for Consistent data Transfer) ۱۶۸
8-4- فضای حافظه CPU (S7 System Memory Areas) ۱۶۸
8-1-4- عمل غیر فعال سازی ماژول‌های ورودی/خروجی(Passivation) ۱۶۹
۸–۵– جدول اختصارات و تعاریف ۱۷۰
8-1-5- قابلیت‌های کارت‌های Fail Safe ۱۷۰
فصل – ۹ – نکات مربوط به پیکربندی در Step 7 ..... ۱۷۳
9-1- آدرس پایه تبادل داده برای CP343 ..... ۱۷۳
۹–۲– ۱۷۴
1- Fail-safe modules ۱۷۴
1-0-1- 1oo1 and 1oo2 evaluation with the fail safe signal module SM 326F ۱۷۴
2-0-1-F Systems: Wiring and Voting Architectures for ET200M F-DIs and F-Dos ۱۷۶
1-1- کنترل کننده‌های Fail-Safe شرکت Siemens ۱۷۷
1-1-1- چگونگی پردازش در یک کنترلر Fail-Safe شرکت Siemens ۱۷۷
2-1-1-PROFIBUS با پروتکل PROFIsafe برای ارتباطات Safety ۱۷۸
3-1-1- مزایای استفاده از کنترلرهای Fail-Safe شرکت زیمنس ۱۷۹
4-1-1- کاربردها ۱۸۰
5-1-1- استانداردها و گواهینامه ها ۱۸۰
فصل – ۱۰ نصب لایسنس در PCS7 ..... ۱۸۱
۱۰–۱– سیم‌بندی ماژول‌های S7300 ..... ۱۸۲
1-1-10- سیم‌بندی ماژول‌های دیجیتال ورودی (DI) ۱۸۲
2-1-10- سیم‌بندی ماژول‌های دیجیتال خروجی (DO) ۱۸۳
3-1-10- سیم‌بندی ماژول‌های آنالوگ ورودی (AI) ۱۸۴
4-1-10- سیم‌بندی ماژول‌های آنالوگ خروجی (AO) ۱۸۶
فصل – ۱۱ ماژول‌های SIPLUS زیمنس ۱۸۷
1-1-11- فعال‌سازی کانال‌های IO (Reintegration) ۱۹۳

فصل – ۱۲ – مدیریت خطاها ۱۹۳
12-1-1- خطا در عبور از مقدار Local data stack ۱۹۳
12-1-2- خطای ۱۹۵

فصل – ۱ مقدمه

۱–۱– دلایل بکارگیری سیستم‌های H-PLC (Motivation)

- پرهیز از وقوع خرابی‌های پرهزینه در پردازش مواد ارزشمند
- هزینه‌های بالا در راه‌اندازی مجدد پس از وقوع نقص و توقف سیستم کنترل (Control Failure)
- عملیات بدون نظارت کارکنان تعمیر و نگهداری

۱–۱–۱– حوزه‌های کاربردی اتوماسیون S7-400H

- انرژی : خطوط لوله، سازه‌های داخل دریا
- استخراج معدن
- مواد شیمیایی، پتروشیمی
- تولید و توزیع برق (Power generation and distribution)
  - برای پلنت‌های نفت، گاز
- حمل و نقل (Transport)
  - مدیریت تونل
  - کشتی‌سازی
- صنعت شیشه
- صنایع غذایی

شکل ۱-۱– صنایع کاربردی سیستم‌های FH

۱-۲- ایستگاه اتوماسیون (AS)

یک ایستگاه اتوماسیون به یک مجموعه ماژول‌های سخت‌افزاری کنترل سیستم اتوماسیون پلنت مبتنی بر سامانه DCS فرآیندی زیمنس (PCS7) اطلاق می‌شود، که شامل ماژول‌های تغذیه(PS)، ماژول‌های CPU، ماژول شبکه ارتباطی(CP) و ماژول‌های ورودی/خروجی مرکزی و توزیع شده (RIO) است. هسته مرکزی سامانه اتوماسیون PCS7 را CPUهای سری S7-400 تشکیل می‌دهد و لذا سامانه اتوماسیون بسته به نوع پردازنده سری S7400 به‌کار رفته در آن شناخته می‌شود. سامانه اتوماسیون در PCS7 براساس یکی از سه نوع کنترل‌کننده S7-400 است.که عبارت‌انداز:

- سامانه اتوماسیون استاندارد AS400 (Standard Systems)؛
- سامانه اتوماسیون افزونه AS400H(Redundant System)؛
- سامانه اتوماسیون ایمن در نقص AS400FH (Failsafe)؛

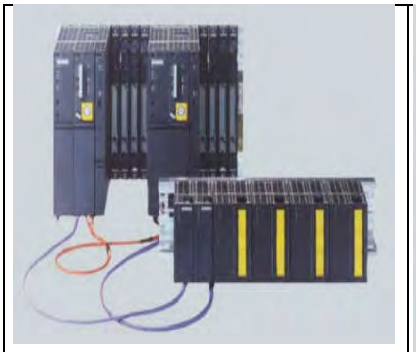
شکل ۱-۲ نمایی از سامانه‌های اتوماسیون مبتنی بر CPUهای S7-400 را نشان می‌دهد.

الف-AS400

- دارای کارایی و سرعت بالا
- قابلیت تعویض آنلاین در ماژول‌ها
- دارای زمان‌های سیکل از 10 ms به بالا

ب-AS400H

- اجتناب از هزینه‌های بالای راه‌اندازی مجدد پلنت
- اجتناب از زمان‌های توقف (down times)
- سویچ بدون مکث بین دو CPU



ج-AS400FH

- فراهم کردن ایمنی برای افراد
- سازگار با استانداردهای:
- EN 954 (up to category 4)
- IEC 61508 (up to SIL 3)
- قابلیت برنامه‌نویسی Fail-safe
- تحمل‌پذیر در نقص Faulttolerant

شکل ۱-۲- انواع سامانه اتوماسیون مبتنی بر S7-400

جدول ۱-۱- فهرست اجزای اصلی سامانه‌های اتوماسیون AS در PCS7

SIMATIC 400	Modules	Parts
CP – 400	Industrial Ethernet	CP 443-1 6GK7 443-1EX11-0XE0, V2.0 or higher
	PROFIBUS	CP 443-5 Extended 6GK7 443-5DX03-0XE0, V5.0 or higher
CPU - 400	CPU 414-4H	6ES7 414-4HJ00-0AB0, V3.1 or higher
	CPU 417-4H	6ES7 417-4HL01-0AB0, V3.1 or higher
	CPU 414-3 DP	6ES7 414-3XJ00-0AB0, V3.1 or higher
	CPU 416-2 DP	6ES7 416-2XK02-0AB0, V3.1 or higher
	CPU 416-3 DP	6ES7 416-3XL00-0AB0, V3.1 or higher
	CPU 417-4	6ES7 417-4XL00-0AB0, V3.1 or higher
PS-400	Redundant PS-400	PS 405 10A, 6ES7 405-KR00-0AA0
		PS 407 10A, 6ES7 407-KR00-0AA0
	Standard PS-400	PS 405 10A, 6ES7 405-KA01-0AA0
		PS 405 20A, 6ES7 405-RA01-0AA0
		PS 407 10A, 6ES7 407-KA01-0AA0
		PS 407 20A, 6ES7 407-RA01-0AA0
RACK - 400	9 slots, UR2	6ES7 400-1JA00-0AA0 6ES7 400-1JA01-0AA0
	18 slots, UR1	6ES7 400-1TA00-0AA0 6ES7 400-1TA01-0AA0
	H system, UR2-H	6ES7 400-2JA00-0AA0

The S7-400 is available in three versions:
S7-400

- *The power PLC for the mid to high-end performance ranges*
- *The solution for even the most demanding tasks*
- *With a comprehensive range of modules and performance-graded CPUs for optimal adaptation to the automation task*
- *Flexible in use through simple implementation of distributed structures and extensive communications capabilities*
- *User-friendly handling and uncomplicated design without a fan*
- *Can be expanded without problems when the tasks increase*

S7-400H

- *Fault-tolerant automation system with redundant design*
- *For applications with high fail-safety requirements*
  Processes with high restart costs, expensive downtimes, little supervision, and few maintenance options.
- *Redundant central functions*
- *Increases availability of I/O: Switched I/O configuration*
- *Also possible to use I/Os with standard availability: Single-sided configuration*
- *Hot stand-by: Automatic reaction-free switching to the standby unit in the event of a fault*
- *Configuration with two separate or one divided central rack*
- *Connection of switched I/O via redundant PROFIBUS DP*

S7-400F/FH

- *Failsafe automation system for systems with increased safety requirements*

- *If required, also fault tolerant through redundant design*
- *Without additional wiring of the safety-related I/O:*
  Safety-relevant communication via PROFIBUS DP with PROFIsafe profile
- *Based on S7-400H and ET 200M with safety modules*
- *Standard modules for non-safety-related applications can also be used in the automation system*

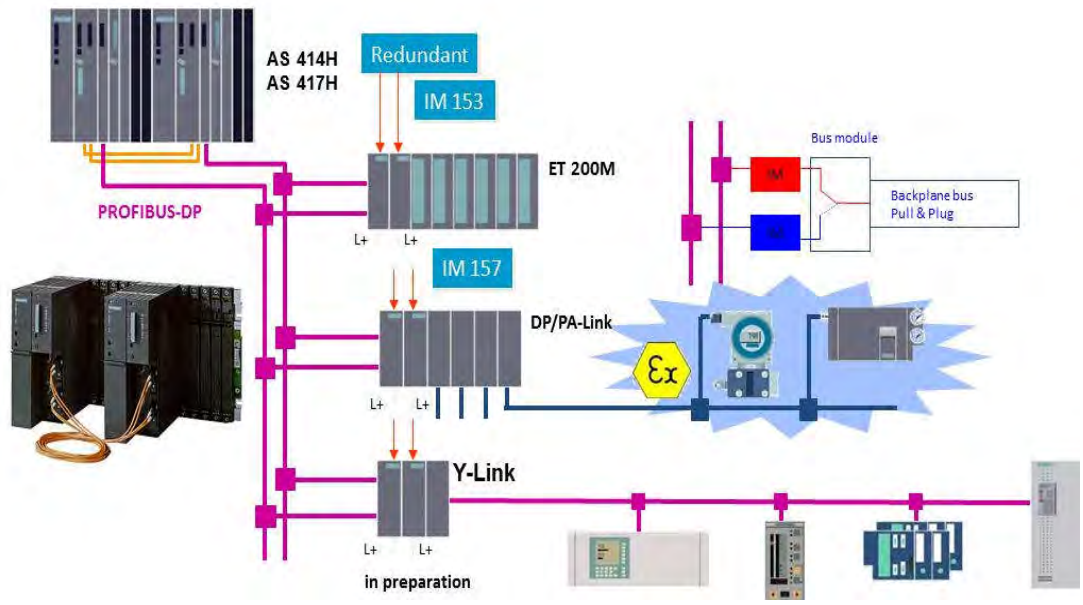
۱-۲-۱- سامانه اتوماسیون استاندارد (سری *AS400*)

در این پیکربندی سامانه اتوماسیون، تنها دارای یک پردازنده *S7-400* بوده و در کاربردهای معمولی که فاقد محیط خطرزا است و همچنین به مؤلفه «دسترس‌پذیری بالا» (افزونگی) نیازی وجود نداشته باشد. از این سری استفاده می‌شود. ماژول‌های کنترل‌کننده این نوع سامانه شامل مدل‌های زیر است:

- *Standard CPU Modules*
  - *CPU 414-3DP;*
  - *CPU 418-3DP;*
  - *CPU 418-2DP;*
  - *CPU 414-3DP;*
  - *CPU 417-4;*

۱-۲-۲- سامانه اتوماسیون برای دسترسی‌پذیری بالا (*AS400H*)

در این نوع سامانه از ماژول‌های *CPU* نوع *H* استفاده می‌شود. حرف *H* در این نوع سامانه معرف قابلیت دسترس‌پذیری (*High Availability*) بالا یا «افزونگی داغ» (*Hot Redandunt or Hot Standby*) می‌باشد. این سامانه در فرایندهایی کاربرد دارد که در آن‌ها توقف عملیات فرایند غیرمجاز است. شکل ۱-۳ یک پیکربندی نمونه از یک سیستم *H* با مدل پردازنده *CPU 414/417* را نشان می‌دهد.



شکل ۱-۳- پیکربندی نمونه از یک سیستم *H* با *CPU 414/417*

شکل ۱-٤- نمونه یک سیستم *S7-400H*

سیستم‌های *H*، سیستم‌هایی هستند که با قابلیت افزونگی در سطوح مختلف سیستم کنترل (کنترل‌کننده، ماژول‌های *I/O*، شبکه، منابع تغذیه) به صورت پشتیبان یکدیگر می‌توانند یک فرایند را کنترل کنند. به طور معمول در ساختار افزونگی سیستم‌های *H* زیمنس، از یک جفت *CPU* استفاده می‌شود. که در آن فرایند موردنظر همواره تحت کنترل یکی از آن‌ها که *Master* نامیده می‌شود، است. سیستم دیگر نیز که *Standby* نامیده می‌شود، در حالت آماده‌باش یا آماده‌به‌کار بوده و در آن لحظه نقشی در کنترل فرایند ندارد. ولی در صورت بروز نقص در *Master*، سیستم پشتیبان یا *Standby* در زمان بسیار کوتاهی وارد مدار شده و کنترل فرایند را ادامه می‌دهد. وقتی در یک سیستم کنترل، یکی از جفت کنترل‌کننده در حالت کار و دیگری آماده به کار است، مد کاری را افزونه (*Redundant*) می‌گویند. وقتی یکی از کنترل‌کننده‌ها به دلیل خطا متوقف می‌شود و دیگری در مدار می‌آید، مد کاری را *Single* می‌نامند. ساختاری که به این صورت ایجاد می‌شود افزونگی سخت‌افزاری خوانده می‌شود و نباید آن را با افزونگی نرم‌افزاری اشتباه گرفت. در واقع مطابق جدول ۱-۱ بین این دو سیستم ازنظر ساختار و ازنظر عملکرد تفاوت‌های اساسی زیر وجود دارد:

جدول ۱-۱- تفاوت‌های بین افزونگی سخت‌افزاری و نرم‌افزاری

افزونگی سخت‌افزاری	افزونگی نرم‌افزاری
فقط با *CPU*های خاصی از خانواده *S7400* قابل پیاده‌سازی است.	با *CPU*های خانواده *S7300* و *S7400* قابل پیاده‌سازی است.
عمل سنکرون‌سازی حافظه بین دو *CPU* به طریق سخت‌افزاری و از طریق ماژول‌های *SYNC* که با فیبر نوری به هم متصل‌اند انجام می‌شود.	سنکرون سازی حافظه دو *CPU* با برنامه‌نویسی بلوک توابع خاص و از طریق کابل شبکه (*MPI-Profibus-Ethernet*) انجام می‌شود.
قابلیت اطمینان بالاتر دارد و زمان جابجایی از سیستم اصلی به سیستم پشتیبان خیلی کوتاه هست.	قابلیت اطمینان آن به‌اندازه سیستم *H* نبوده و زمان جابجایی آن نیز به کوتاهی سیستم‌های *H* نیست

هزینه آن زیاد است.	هزینه بالایی ندارد.

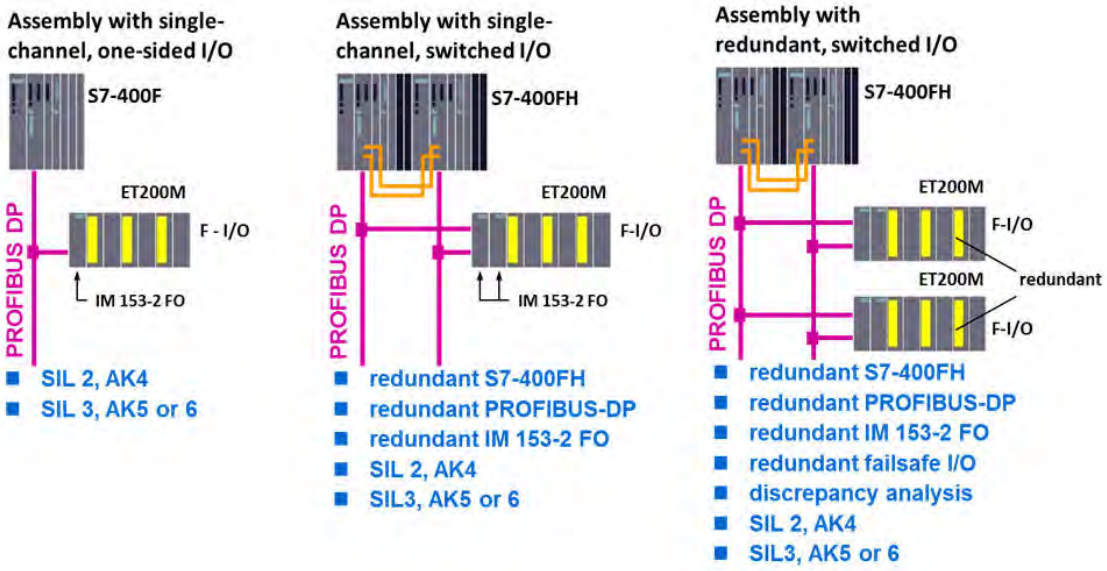
انتخاب بین دو روش فوق بر اساس نیاز فرآیند صورت می‌گیرد. ممکن است در فرایندی خسارات ناشی از توقف آن‌قدر زیاد باشد که سرمایه‌گذاری اولیه برای داشتن یک سیستم *H* در مقابل آن ناچیز محسوب شود. سیستم‌های *H* در *S7* فقط توسط *S7-400H* قابل پیاده‌سازی هستند.

۱-۲-۳- سامانه اتوماسیون ایمن در خطا (*Fail-Safe AS400FH*)

سامانه اتوماسیون *AS400FH* یک سیستم ترکیبی با مشخصه افزونه و تحمل‌پذیر خطا (*Fail-Safe*) است. به طوری که سامانه *PCS7* برای داشتن یک سیستم با میزان دسترسی بالا و در عین حال ایمن، از سری *CPU*های با ساختارهای ترکیبی *FH* استفاده می‌کند. معماری *S7-400FH*، در دسترس‌پذیری بالا و فن‌آوری ایمنی را در یک سیستم اتوماسیون واحد ترکیب می‌کند و با حفظ ایمنی (*PROFIsafe*) در برابر نقص‌ها تحمل‌پذیر است.

این سیستم دارای گواهی‌نامه *TÜV* بوده و مطابق با تمام استانداردهای مربوط به تولید و اتوماسیون فرایند است. این سامانه در محیط‌های پرخطر که نیاز به ایمنی بالا و افزونگی دارد، به کار گرفته می‌شود. شکل ۱-۵ پیکربندی‌های نمونه برای سیستم‌های *FH* را نشان می‌دهد. ماژول‌های *CPU* این نوع سامانه شامل مدل‌های زیر است:

- *AS 414-2H;*
- *AS 417–4H;*
- *AS 417–5H;*

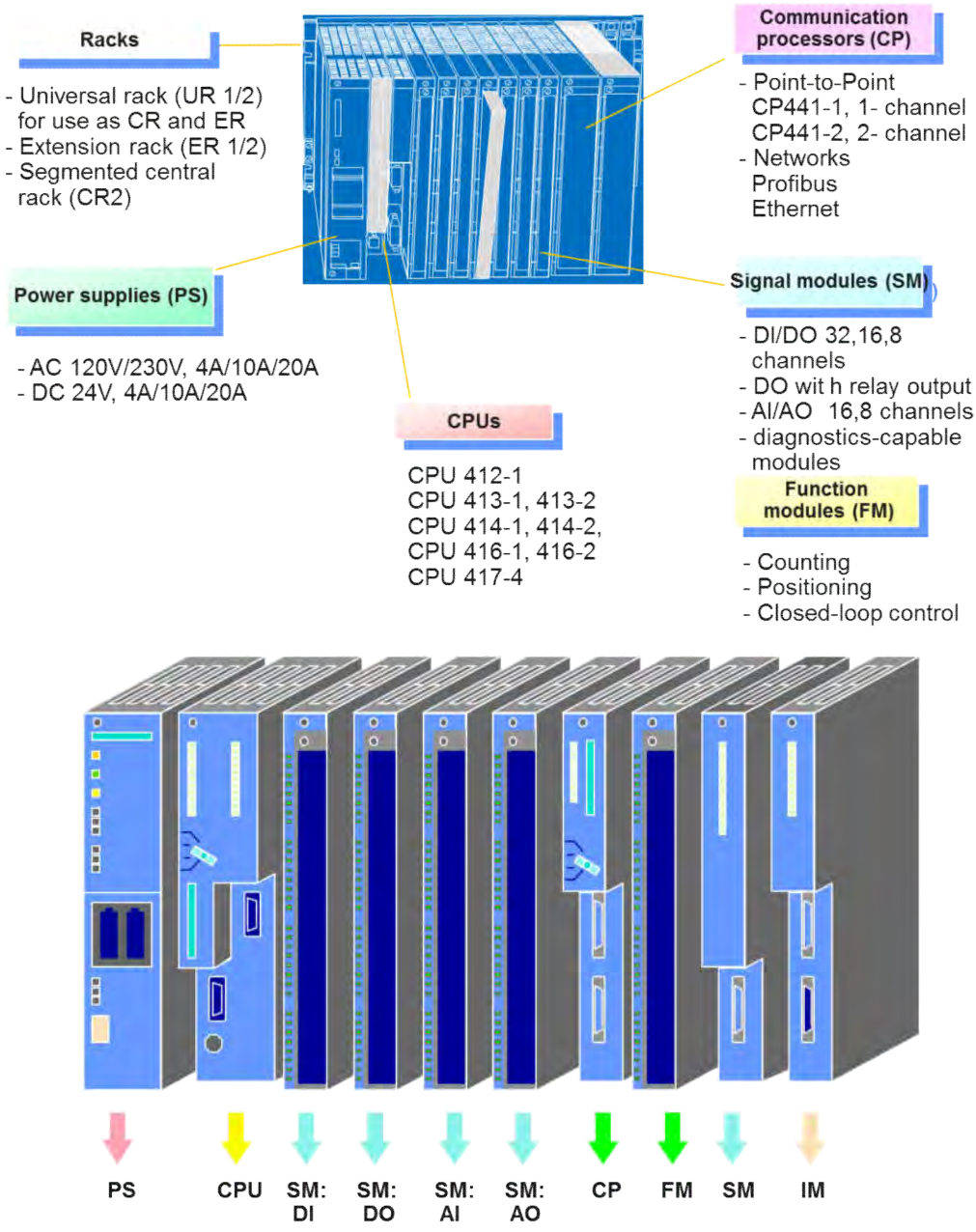


شکل ۱-۵- نمونه پیکربندی‌های تحول‌پذیر نقص سیستم‌های *S7-400FH*

فصل – ۲ – سامانه AS-400 استاندارد

۲–۱– اجزای AS-400 استاندارد

شکل زیر اجزای یک سیستم AS-400 استاندارد را به تصویر کشیده است.



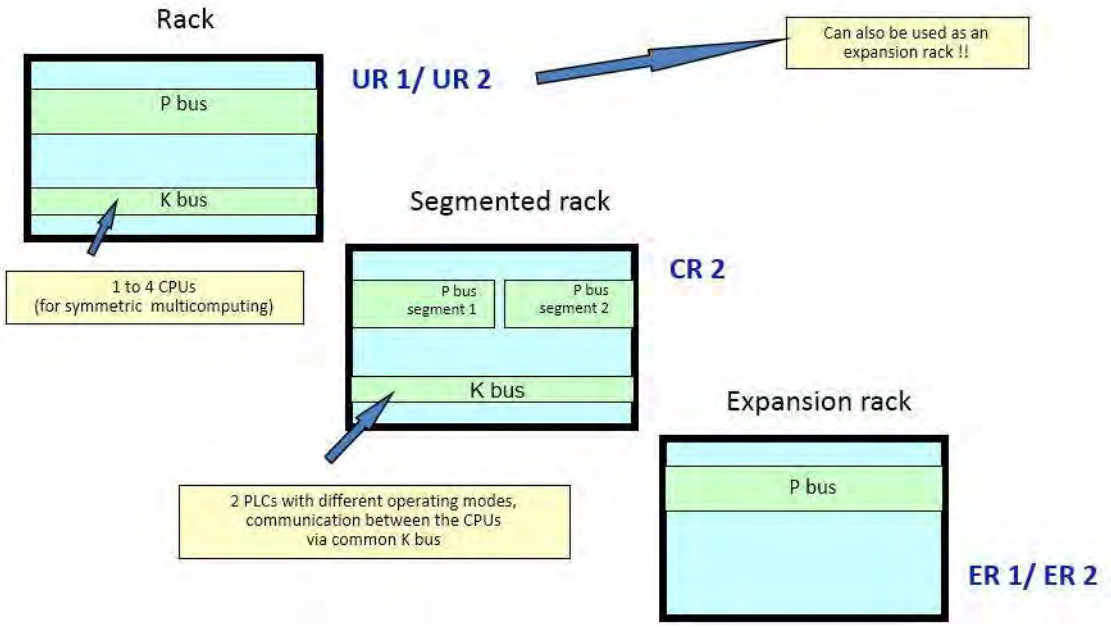
شکل ۲-۱– اجزای یک سیستم S47-400 استاندارد

۲-۲- انواع رک S7-400

برخلاف سیستم S7-300 که تنها یک نوع رک (ریل آلومنیومی) دارد. سیستم S7-400 دارای سه نوع رک می‌باشد. که هر یک دارای مشخصات و مزایای خاص خود می‌باشد.

- *CR : Centeral Rack*
- *ER : Expansion Rack*
- *UR : Universal Rack*

عنوان تمام رک‌های به‌کاررفته برای هر سه نوع سامانه AS400، UR (Universal rack) نام دارد که در چند مدل به شرح زیر استفاده می‌شود. شکل زیر انواع رک در S7-400 را به تصویر کشیده است.



شکل ۲-۲- انواع رک در S7-400

- *UR2: Universal rack, 9 slots, not suitable for redundant Power Supply Modules*
- *UR2: Universal rack, 9 slots*
- *UR2ALU: Universal aluminium rack, 9 slots*
- *UR1: Universal rack, 18 slots, not suitable for redundant power supply modules*
- *UR1: Universal rack, 18 slots*
- *UR1ALU: Aluminium universal rack, 18 slots*
- *UR2-H: Central rack, 2*9 slots, split backplane bus, suitable for compact configuration of standard and redundant PLCs*
- *UR1ALU: Central aluminium rack, 2*9 slots, split backplane bus, suitable for compact configuration of standard and redundant PLCs*

رک *UR2-H* نصب و راه‌اندازی دو زیرسیستم جداگانه را به صورت افزونه با ۹ اسلات پشتیبانی می‌کند که برای نصب در کابینت ۱۹ اینچی مناسب است. همچنین می‌توان *S7-400H* را با دو رک جداگانه *UR1* و *UR2* پیکربندی کرد.

جدول ۲-۱- انواع رک در *S7-400*

عنوان	نوع	کاربرد	تعداد اسلات	توضیح
UR1	*Universal*	بعنوان رک اصلی بعنوان رک بسط *I/O*	۱۸	دارای دو نوع است که در یک نوع آن می‌توان دو منبع تغذیه از نوع ریداندانت قرار داد.
UR2	”	”	۹	”
ER1	*Expansion*	به عنوان رک بسط *I/O*	۱۸	”
ER2	”	”	۹	”
CR2	*Segmented*	به عنوان رک اصلی	۱۰+۸	دارای دو بخش است که هر بخش آن می‌تواند یک رک اصلی مستقل باشد.
UR2-H	*Universal*	برای *PLC*های ریداندانت به کار می‌رود	۲*۹	شبیه دو رک اصلی *UR2* است.

۲-۳- انواع باس در رک *S7-400*

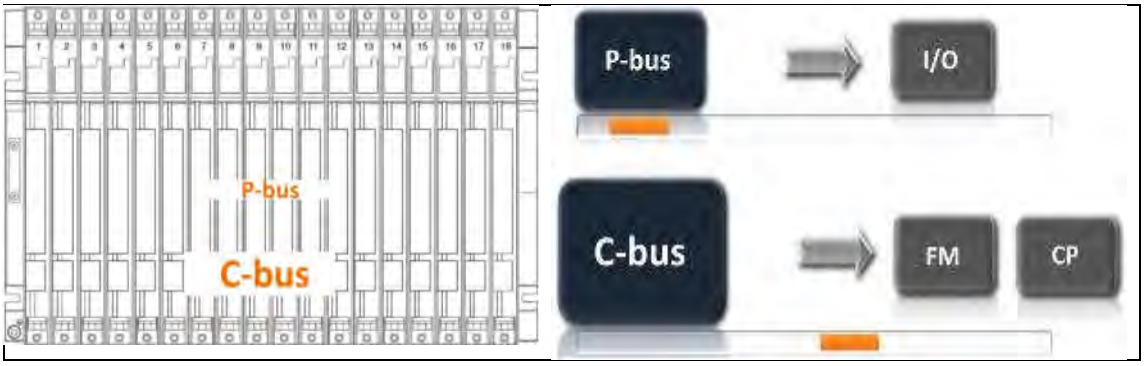
بر روی رک ٤٠٠ دو نوع باس وجود دارد

- گذرگاه *P-bus* یا *IO Bus*

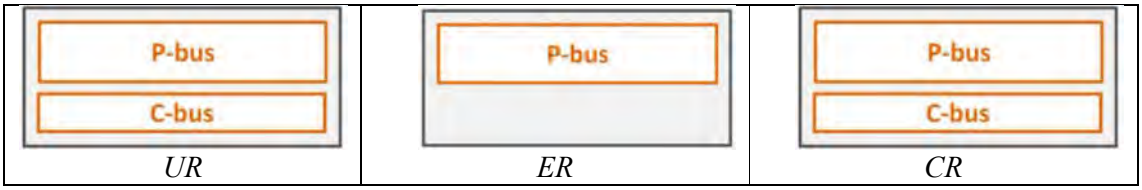
حرف *P* معرف *Perioheral* می‌باشد. این باس تبادل داده بین *CPU* و ماژول‌های *I/O* را برعهده دارد.

- گذرگاه *C-bus* یا *K-bus* (اول حرف عبارت آلمانی ارتباط *Kpmmunikation*)

حرف *C* معرف *Communication* می‌باشد. این باس تبادل داده بین *CPU* و ماژول‌های *CP* و *FM* را بر عهده دارد. سرعت تبادل داده در *C-bus* بیشتر از از *P-bus* می‌باشد.



شکل ۲-۳- دو نوع گذرگاه بر روی رک *S7-400*



شکل ۲-٤- باس‌های موجود بر روی رک‌های *S7-400*

- رک مرکزی *CR*، رکی است که تنها به عنوان *Main Rack* استفاده می‌شود و بایستی به عنوان اولین رک نصب شود. این نوع رک در پیکربندی چند لایه (*Multi tire*) به عنوان رک *Expansion* نمی‌تواند مورد استفاده قرار گیرد. رک *CR* دارای هر دو گذرگاه *P* و *C* می‌باشد.
- رک *ER* فقط به عنوان باس *Expansion* استفاده می‌شود و تنها دارای باس *P* می‌باشد. لذا تنها ماژول‌های *I/O* بر روی این باس قابل نصب هسند.
- رک *UR* به طور وسیعی به هر دو صورت رک *Main* و *Expansion* استفاده می‌شود.
- رک *UR* دارای هر دو باس *P* و *C* می‌باشد. لذا کلیه ماژول‌های *I/O* ، *CP* و *FM* قابل نصب بر روی این رک می‌باشد.

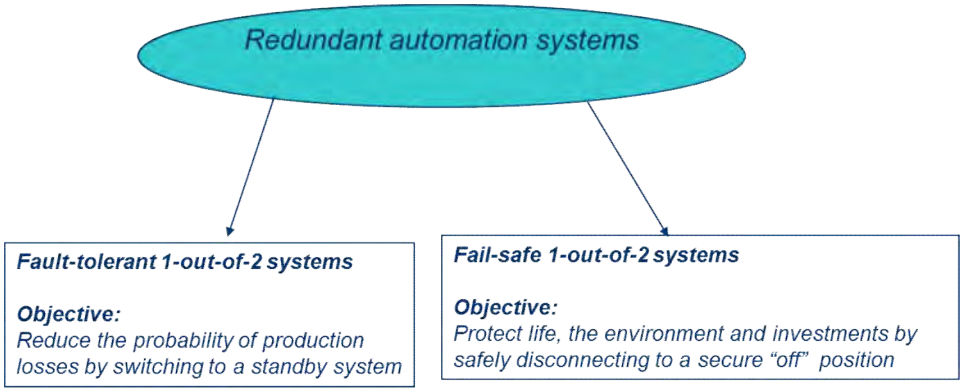
فصل – ۳ – ساختار و معماری سامانه S7-400H

۳-۱- مقدمه

در سیستم‌های اتوماسیون افزونه دو مفهوم مهم و اساسی مدنظر می‌باشد که عبارت‌انداز:

1. *Fault-tolerant*
2. *Fail-safe*

هدف از سیستم‌های نوع اول کاهش احتمال از دست دادن تولید محصول از طریق سویچ به سیستم یدکی می‌باشد. و هدف ار سیستم‌های نوع دوم حفاظت از جان افراد و تجهیزات محیط کار از طریق هدایت سیستم به موقعیت ایمن در مواقع بروز خط و اشکال می‌باشد.



چرا بایستی از یک سیستم کنترل *PLC* با مشخصه تحمل‌پذیر خطا استفاده کنیم؟

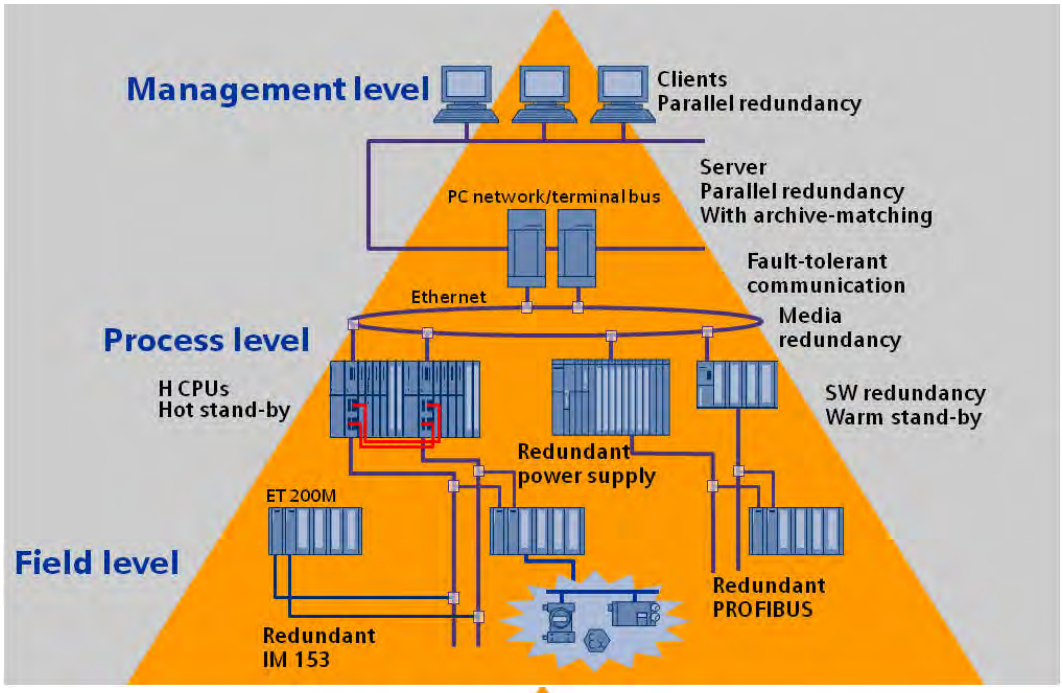
- هدف استفاده از کنترل کننده‌های *PLC* با قابلیت دردسترس‌ذیری بالا، کاهش تلفات تولید است. مهم نیست که آیا از دست دادن محصول ناشی از یک خطا است و یا در نتیجه کار تعمیر و نگهداری ایجاد می‌شود.هر چقدر هزینه‌های توقف بالاتر باشد، استفاده از یک سیستم تحمل پذیر خطا ارزشمند تر است. به طور کلی هزینه‌های سرمایه گذاری بالا در سیستم‌های تحمل پذیر خطا با اجتناب از کاهش تولید جبران می‌شود.

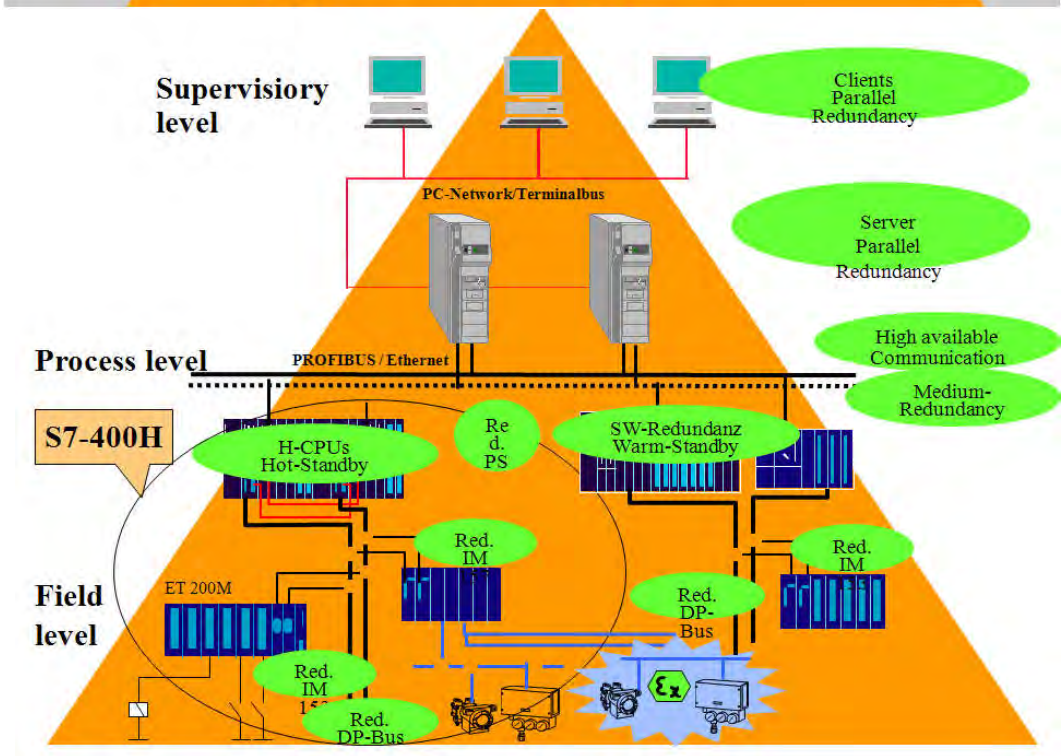
به طور کلی برای سیستم‌های اتوماسیون به دو صورت افزونگی قابل پیاده‌سازی می‌باشد:

- افزونگی نرم‌افزاری (*Software Redundancy*) – برای فرایندهای کند استفاده می‌شود.
- افزونگی سخت‌افزاری (*Hardware Redundancy*) –برای فرایندهای سریع استفاده می‌شود.

۳-۲- معماری سیستم (*System architecture*)

معماری سیستم *S7-400H*، به گونه‌ای است که ساختار افزونگی در تمام سطوح هرم اتوماسیون و اجزای سخت‌افزاری و نرم‌افزاری را پشتیبانی می‌کند. شکل زیر معماری نوعی را برای سیستم *S7-400H* به تصویر کشیده است.





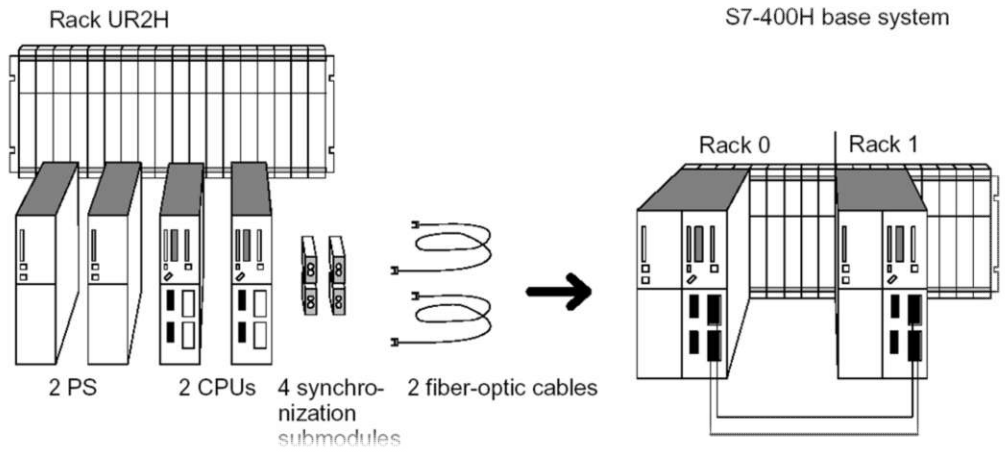
شکل ۳-۱- افزونگی سامانه *S7-400H* در سطوح هرم اتوماسیون برای دسترس‌پذیری بالا (*High Availability*)

۳-۳- اجزای سخت‌افزاری یک سیستم *S7-400H* (*Minimum Configuration*)

یک کابینت سیستم *AS400H* حداقل دارای اجزای زیر است.

۱. دو عدد ماژول *CPU*؛

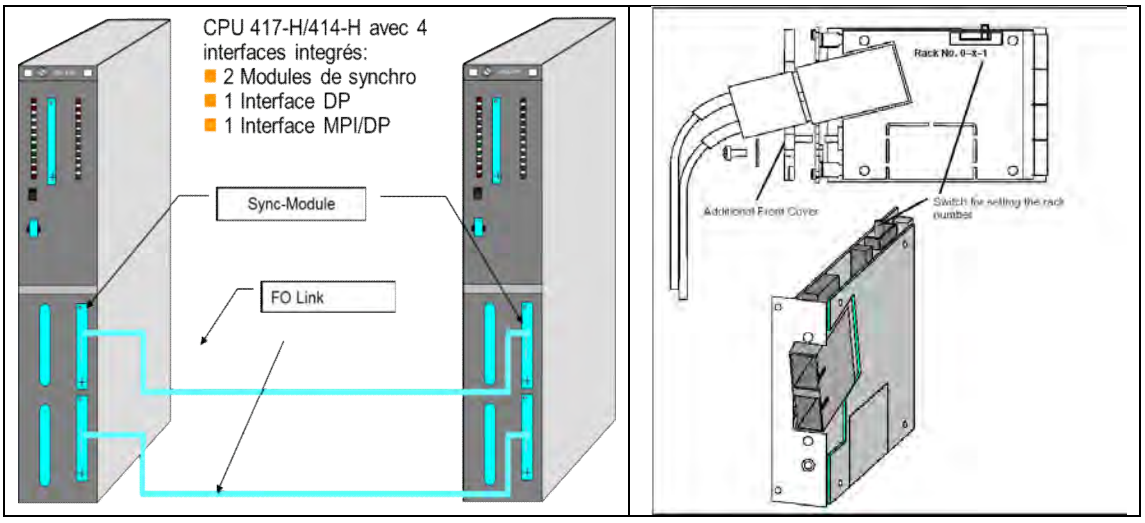
۲. دو عدد کابل فیبر نوری؛
۳. دو عدد منبع تغذیه؛
٤. جفت رک؛
٥. چهار عدد ماژول سنکرون‌سازی (دو عدد برای هر *CPU* برای دو کابل افزونه فیبر نوری)



حداقل اجزای سخت‌افزاری یک سیستم *H*

۳-۳-۱- ماژول کنترل‌کننده (*CPU*)

در مرکز یک سیستم *S7-400H* ، دو واحد *CPU* می‌باشد. تنظیم ماژول‌های سنکرون‌سازی، که به *CPU* متصل می‌شوند، شماره رک برای *CPU*ها را تعریف می‌کند. در سیستم *H* از *CPU* در رک صفر، *CPU 0* و از *CPU* نصب شده در رک ۱ با عنوان *CPU1* یاد می‌شود.



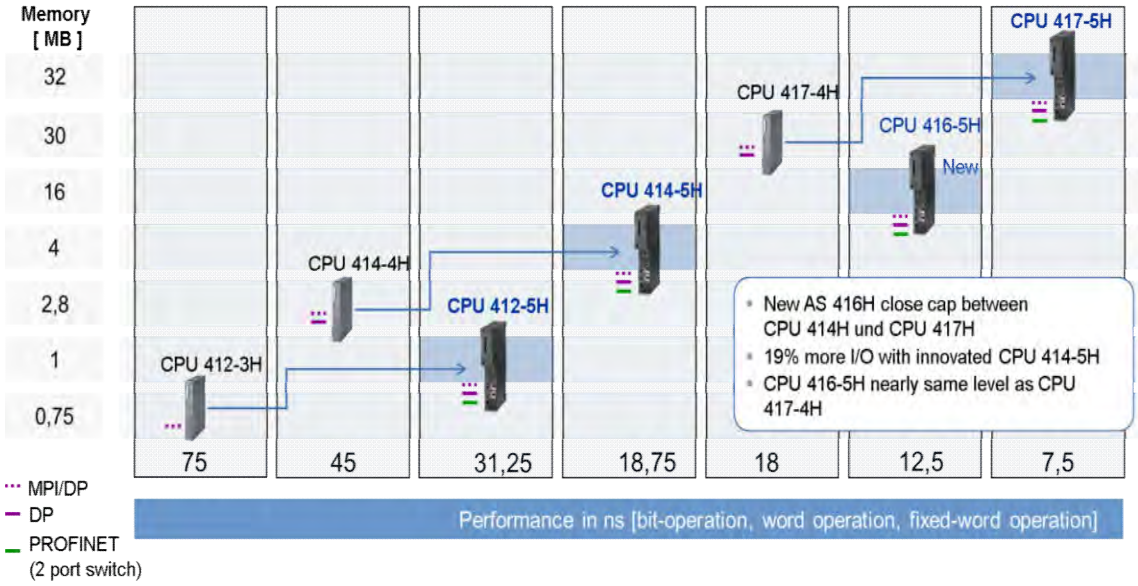
شکل ۳-۲- ساختار کنترل‌کننده *S7-400H*

Switch Position	Meaning
	The CPU has rack number 1
	The CPU has not been assigned a rack number
	The CPU has rack number 0

شکل ۳-۳- تنظیم شماره رک بر روی *H-CPU*

شکل زیر مشخصات مدل‌های مختلف ماژول *CPU* سامانه اتوماسیون *S7-400H* را نشان می‌دهد.

- *Fault-tolerant CPUs*
  - *CPU 412-3H; CPU 412-5H; CPU 414-4H; CPU 414-5H; CPU 416-5H; CPU 417-4H;*
  - *CPU 417-5H;*



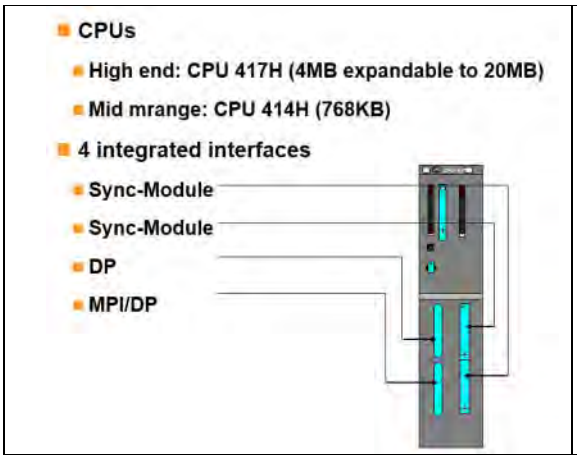
شکل ۳-٤- مدل‌های مختلف ماژول *S7400 H-CPU*

شکل ۳-۵ مشخصات فنی ماژول‌های *H CPU* را به تصویر کشیده است.

Technical data: H-CPUs				
CPU	**CPU 412-5H**	**CPU 414-5H**	**CPU 416-5H**	**CPU 417-5H**
Dimensions (mm)	50x290x219			
Slots	2			
Order No. group: 6ES7	412-5HK.	414-5HM.	416-5HS.	417-5HT.
Firmware	V6.0			
Work memory				
Integrated	1 MB	4 MB	16 MB	32 MB
For program	0.5 MB	2 MB	6 MB	16 MB
For data	0.5 MB	2 MB	10 MB	16 MB
Processing times				
Bit operation	31.25 ns	18.75 ns	12.5 ns	7.5 ns
Word operation	31.25 ns	18.75 ns	12.5 ns	7.5 ns
Fixed-point operation	31.25 ns	18.75 ns	12.5 ns	7.5 ns
Floating-point operation	62.5 ns	37.5 ns	25 ns	15 ns
Bit memories, timers, counters				
Bit memory	8192			
S7 timers/counters	2048/2048			
IEC timers/counters	Unlimited (limited only by the work memory)			
Address ranges				
I/O	8192 bytes			
I/O process image	256 bytes		1 024 bytes	
DP interfaces				
Number of MPI/DP interfaces	1			
Number of DP interfaces	1			
Plug-in interfaces	-			
PN interfaces				
Number of PN interfaces	1			
PROFINET IO	Yes			
PROFINET with IRT	No			
PROFINET CBA	No			
TCP/IP	Yes			
UDP	Yes			
ISO-on-TCP (RFC 1006)	Yes			
Web server	No			
Data set gateway	Yes			

شکل ۳-۵- مشخصات فنی ماژول‌های *H CPU*

نکته: عناوینی چون *3H* یا *4H* و غیره به تعداد پورت‌های موجود بر روی *CPU* اشاره می‌کند. به عنوان مثال *CPU 414-4H* دارای دو پورت *SYNC*، یک پورت *DP* و یک پورت *MPI/DP* می‌باشد.



- *1 PROFIBUS DP interface*
- *1 MPI/PROFIBUS DP interface*
- *2 interfaces for accommodating the Sync modules*
- *1 PROFINET interface with 2-port switch*

شکل ۳-۶- واسط‌های ارتباطی برای *CPU 414-4H*

جدول ۳-۱- مقدار حافظه و تعداد پورت‌های روی *CPU*های مختلف نوع *H*

	CPU 412-5H	*CPU 414-5H*	*CPU 416-5H*	*CPU 417-5H*

(Code/Data)	1MB 512kB/512KB	4MB 2MB/2MB	16MB 6MB/10MB	32MB 16MB/ 16MB
DP/MPI	1	1	1	1
DP	1	1	1	1
PN (nkl. 2port-Switch	1	1	1	1
Safety (optional)	✓	✓	✓	✓
Max. length	10 km	10 km	10 km	10 km

مشخصات نسخه FW V6.0 یک CPU از مدل CPU 412-5H عبارت انداز:

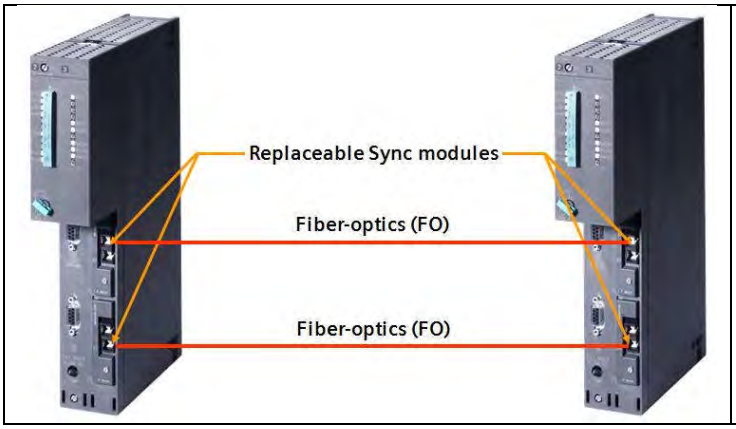
- New middle-class controller CPU 416H
- Increased performance
- Increased resources for communication
- Ethernet integrated (without additional CP)
- Routing for integrated PROFIBUS interface
- PROFINET System Redundancy
- CPU 412-5H: Synchronization up to 10 Km

ماژول‌های I/O نیز به صورت ایستگاه‌های ET200M به سیستم H متصل می‌شوند.

- IO – ET 200M
  - ET 200M : IM 153-4 PN (6ES7153-4BA00-0XB0)
  - ET 200MP

۳-۳-۲- ماژول‌های سنکرون‌سازی (SYNC Modules)

این ماژول‌ها برای اتصال CPUهای نوع H به یکدیگر از طریق دو کابل فیبر نوری به صورت افزونه به کار می‌روند. روی هر CPU دو ماژول سنکرون‌سازی (SYNC) قرار می‌گیرد. روی ماژول‌های SYNC یک DIP Switch وجود دارد که دارای وضعیت ۰ و ۱ است. برای رک صفر آن را روی صفر و برای رک دیگر آن را روی یک تنظیم می‌کنیم. با روشن شدن سیستم ابتدا آنکه روی صفر تنظیم‌شده به‌عنوان Master کار خواهد کرد. بسته به کابل Sync یا نوع پیکربندی رک برای فواصل ۱۰متر یا ۱۰کلیومتر دو نوع ماژول SYNC وجود دارد.



- For Sync cables up to 10 m in length
- For Sync cables up to 10 km in length with the CPUs 414-4H or 417-4H for applications in which the subunits have to be set up at some distance

شکل ۳-۷- نمایی از درگاه *CPU* برای ماژول‌های سنکرون‌سازی

۳-۳-۳- کابل فیبر نوری

بین هر دو ماژول *SYNC* یک زوج فیبر برای برقراری ارتباط افزونه متصل می‌شود.

۳-۴- مشخصه‌های افزونگی (*Redundancy features*)

افزونگی سیستم‌های *H* دارای ویژگی‌های زیر است:

۳-۴-۱- تعویض بی‌درنگ بین دو کنترل‌کننده (*Bumpless Master-Stand-by switchover*)

سری *S7-400H* تعویض آنی و بدون مکث (*Bumpless*) بین *Master* و *Stand-by* را با مشخصات زیر فراهم می‌کند.

- زمان تعویض (*Switchover time*):
  - زمان تعویض زیر ۱۰۰ میلی‌ثانیه؛
  - حفظ وضعیت خروجی‌ها در طول زمان تعویض (*Outputs are Retained*)؛
  - از دست نرفتن هیچ‌گونه اطلاعات آلارم یا وقفه (*Alarm/Interrupt*)؛
- معیار تعویض (*Switchover Criteria*)؛

معیار تعویض از *Master* به *Stand-by* وجود خرابی در *Master* می‌باشد. در صورت خرابی در *Master* که ناشی از نقص در یکی از اجزای زیر است، عمل تعویض به سیستم *Stand By* صورت می‌گیرد.

- *Power supply;*
- *Rack;*
- *Sync module;*
- *Sync cable;*
- *CPU;*

نقص یا خرابی (*Failure*) در ماژول واسط *DP Slave* موجب تعویض نمی‌گردد.

۳-۴-۲- سنکرون‌سازی بین دو *CPU* (*Synchronization Procedure*)

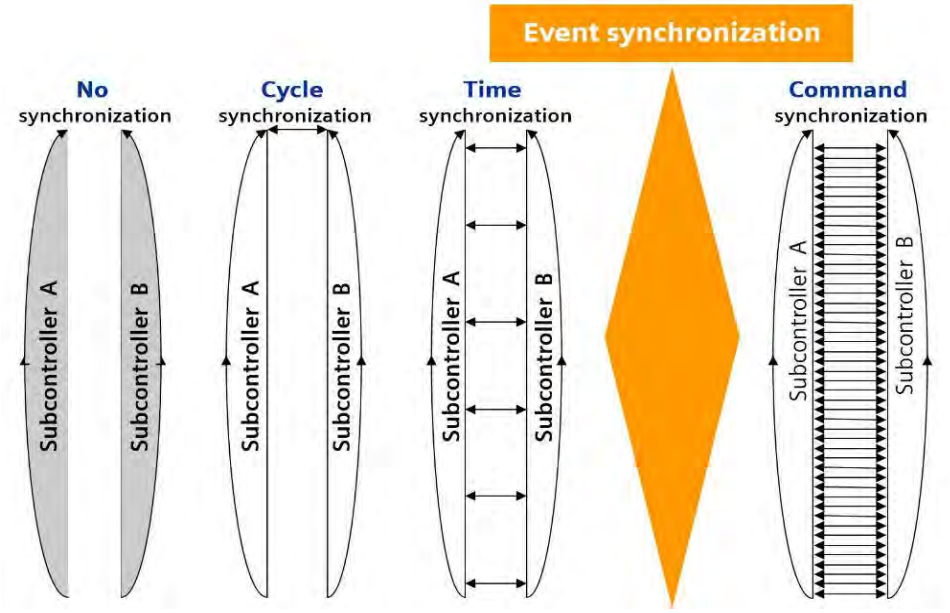
برای انجام سوییچ بدون مکث بین دو *CPU* در زمان بروز خرابی در یکی از آن‌ها، سنکرون‌سازی بین دو سیستم *Master* و *Stand-By* ضروری است. مطابق شکل زیر سه روش سنکرون‌سازی برای عملیات دو *CPU* قابل‌تعریف است:

- سنکرون‌سازی در انتهای هر سیکل اسکن (*Cycle Synchronization*)؛
- سنکرون‌سازی زمانی (*Time synchronization*) – عمل سنکرون کردن در بازه‌های زمانی به خصوصی صورت می‌گیرد؛
- سنکرون‌سازی با اجرای هر دستور لاجیک (*Command synchronization*)؛

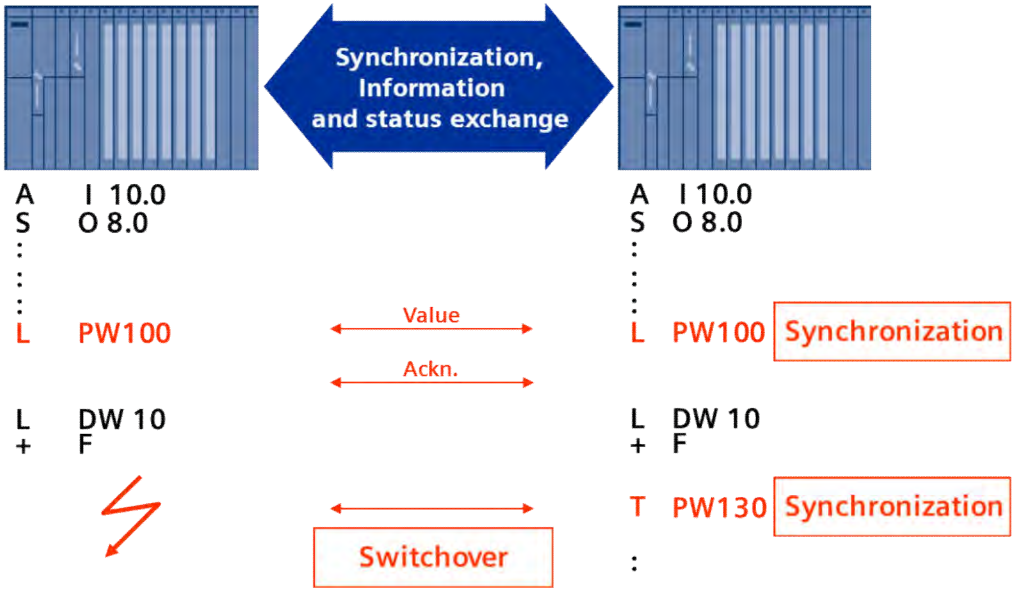
این روش همان تکنیک سنکرون‌سازی مبتنی بر رخداد (*Event Driven Synchronization*) است که *CPU*های *H* زیمنس از آن استفاده می‌کنند و حق امتیاز (*Patent*) آن برای زیمنس ثبت شده است. در این روش عمل سنکرون کردن داده بین دو *CPU* در زمان اجرای برنامه لاجیک (*User Program*) صورت می‌گیرد. به عبارت دیگر هر زمان که رخدادها در *CPU* مستر، منجر به وضعیت داخلی مختلف شد، یک عملیات هماهنگ‌سازی صورت می‌گیرد. عمل سنکرون بین دو *CPU* در مواقع زیر صورت می‌گیرد:

- در زمان دسترسی مستقیم به *I/O*؛
- در موقع وقوع وقفه‌ها و آلارم‌ها؛
- به‌روزرسانی زمان‌ها در برنامه‌ی کاربر؛
- اصلاح یا تغییر داده‌ها با استفاده از توابع شبکه یا ارتباطات؛

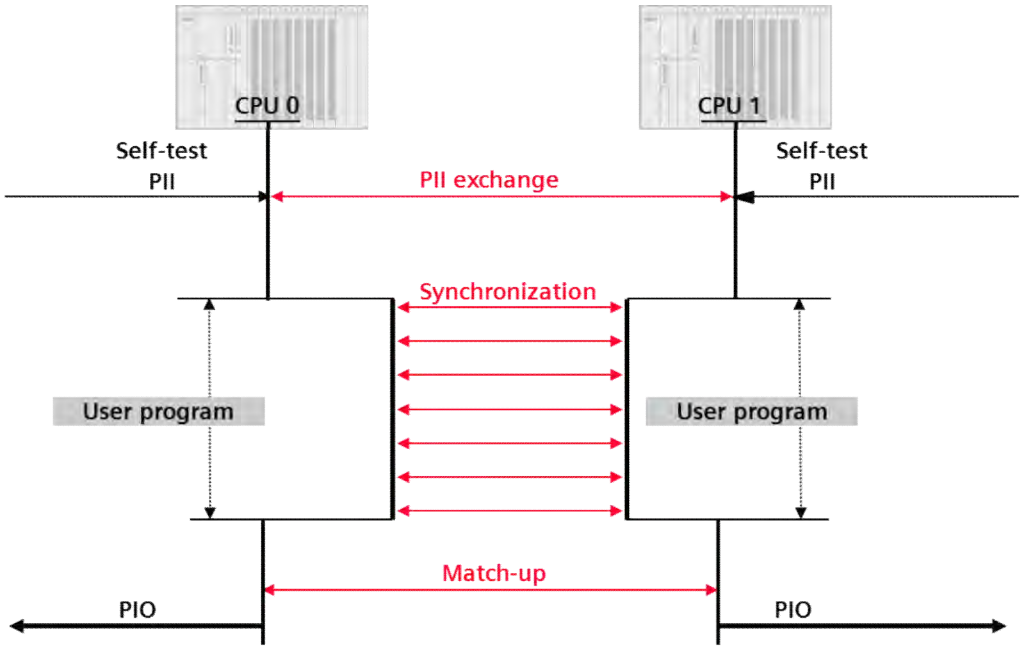
عمل هماهنگ‌سازی به‌طور خودکار و توسط سیستم‌عامل صورت می‌گیرد و نیاز به برنامه‌نویسی توسط کاربر ندارد.



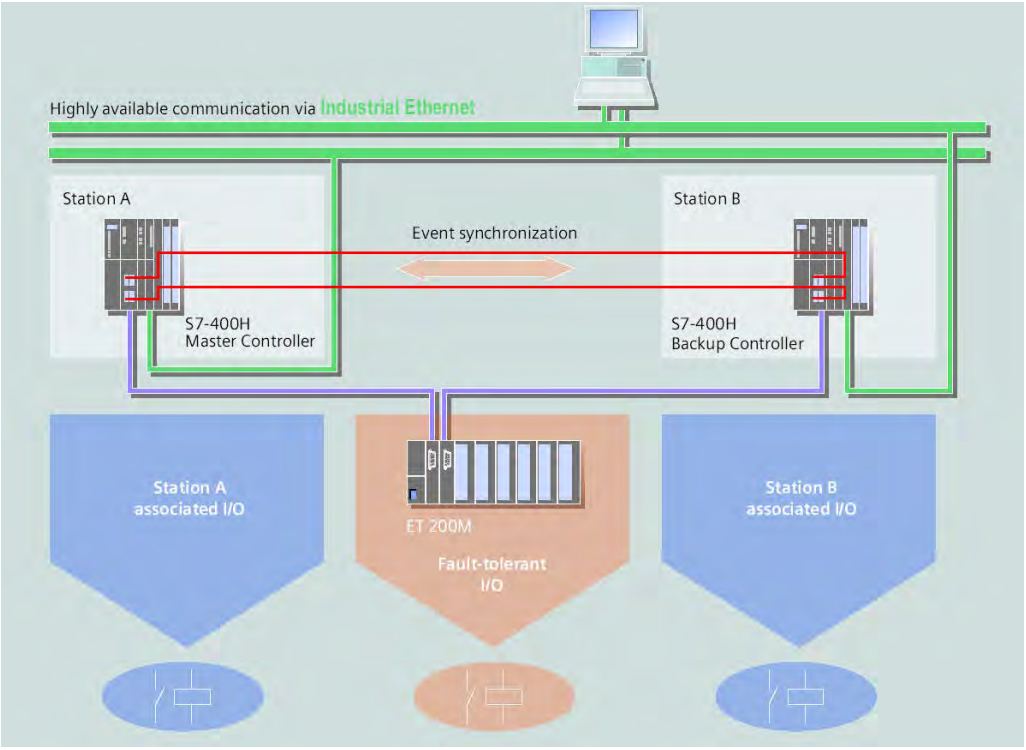
شکل ۳-۸- پروسه سنکرون‌سازی بین دو *CPU*



شکل ۳-۹- مفهوم سنکرون سازی بین دو *CPU* در سیستم *H* (*Principle*)



شکل ۳-۱۰- سیکل سنکرون سازی بین دو *CPU* در سیستم *H*



سنکرون‌سازی بین دو *CPU* به صورت *Event Synchronization*

۳-۴-۳- توابع خودآزمایی (*self-test*)

*CPU*های سری *S7-400H* دارای توابع خودآزمایی برای اجزای سخت‌افزاری (*Scope*) زیر هستند.

- *CPU;*
- *Memory;*
- *Synchronization link or connection;*

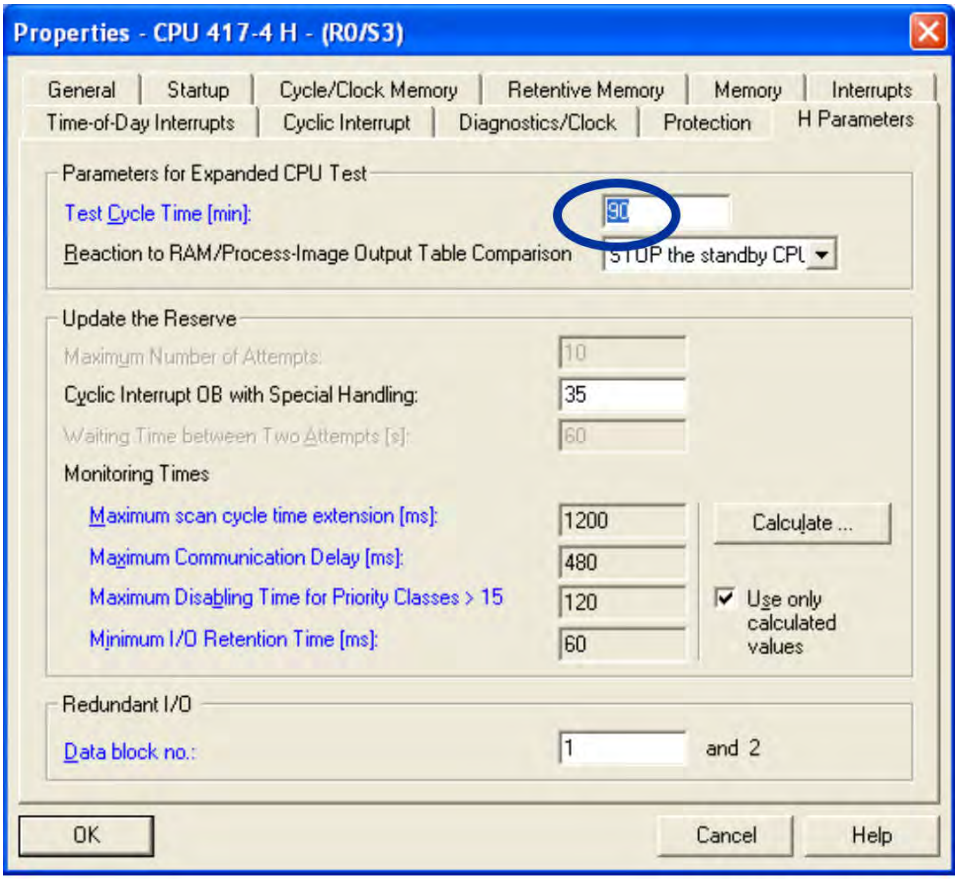
توابع آزمون به دو صورت اجرا می‌شوند (ساختار آزمودن)

۱- خودآزمایی در زمان راه‌اندازی (*Start-up self test*)

- آزمودن کاملی بوده و در زمان راه‌اندازی (*Startup self-test*) اجرا می‌شوند. زمانی که خطایی رخ می‌دهد، *CPU* به حالت *defect* می‌رود.

۲- خودآزمایی در زمان اجرای برنامه سیکلیک (*Self test in cycle operation*)

- این آزمون به صورت دائمی و به عنوان یک وظیفه (*Task*) در پس زمینه اجرا می‌شود.
- اجرای کل آزمون می‌تواند بین سیکل‌های مختلف توزیع شود و یا می‌تواند به طور کامل در یک دوره زمانی قابل‌تعریف در پارامترهای *CPU* (مثال ۹۰ دقیقه) اجرا شود.



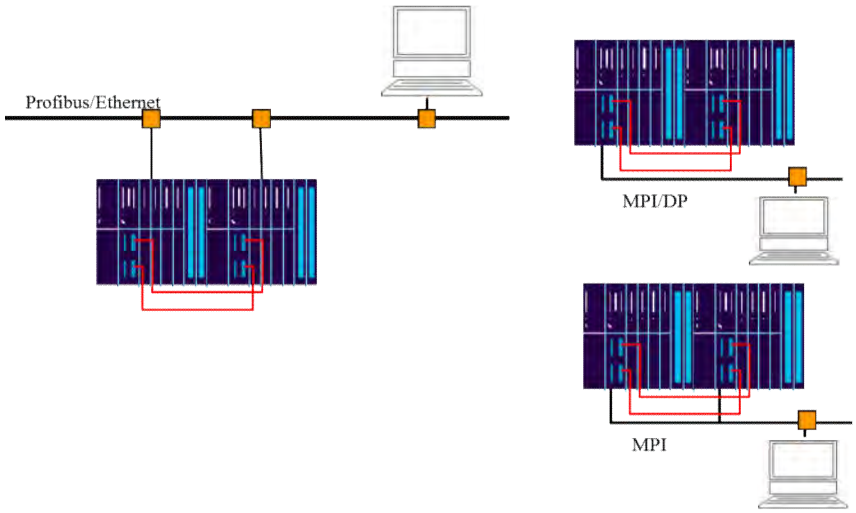
شکل ۳-۱۱- تنظیم دوره زمانی خودآزمایی در پارامترهای *CPU*

۳-۴-۴- برنامه‌نویسی آنلاین (*Online programming*)

همانند سیستم‌های استاندارد *S7-400*، قابلیت برنامه‌نویسی کامل آنلاین برای این سیستم‌ها نیز وجود دارد؛ که شامل تغییر برنامه، تخصیص پارامترها و تغیر پیکربندی سخت‌افزاری است. کلیه تغییرات و اصلاحات به صورت خودکار در هر دو *CPU* کپی می‌شود.

۳-۴-۵- اتصال به *PG*

برای اتصال سیستم *H* به کامپیوتر مهندسی(*PG*) و دانلود پیکربندی به آن می‌توان، از آداپتور *MPI* یا یکی از گذرگاه‌های شبکه اترنت و پروفیباس استفاده کرد.



شکل ۳-۱۲- اتصال سیستم *H* به کامپیوتر مهندسی (*PG*)

۳-۴-۶- قابلیت تعمیر آنلاین (*Online Repair*)

اجزایی از *S7-400* که به صورت آنلاین قابل تعمیر (حذف و اضافه کردن ماژول) هستند، عبارت‌اند از:

۱- ماژول‌های *I/O* و *CP* در مد *RUN* قابل تعویض هستند.

۲- ماژول‌های *IM* (واقع در رک مرکزی) و *CPU* از طریق قطع تغذیه، قابل تعویض هستند(*via Power off*).

۳- پس از روشن کردن *CPU* تعویض شده، به صورت خودکار بروزآوری می‌شود

٤- ماژول‌های *Sync*

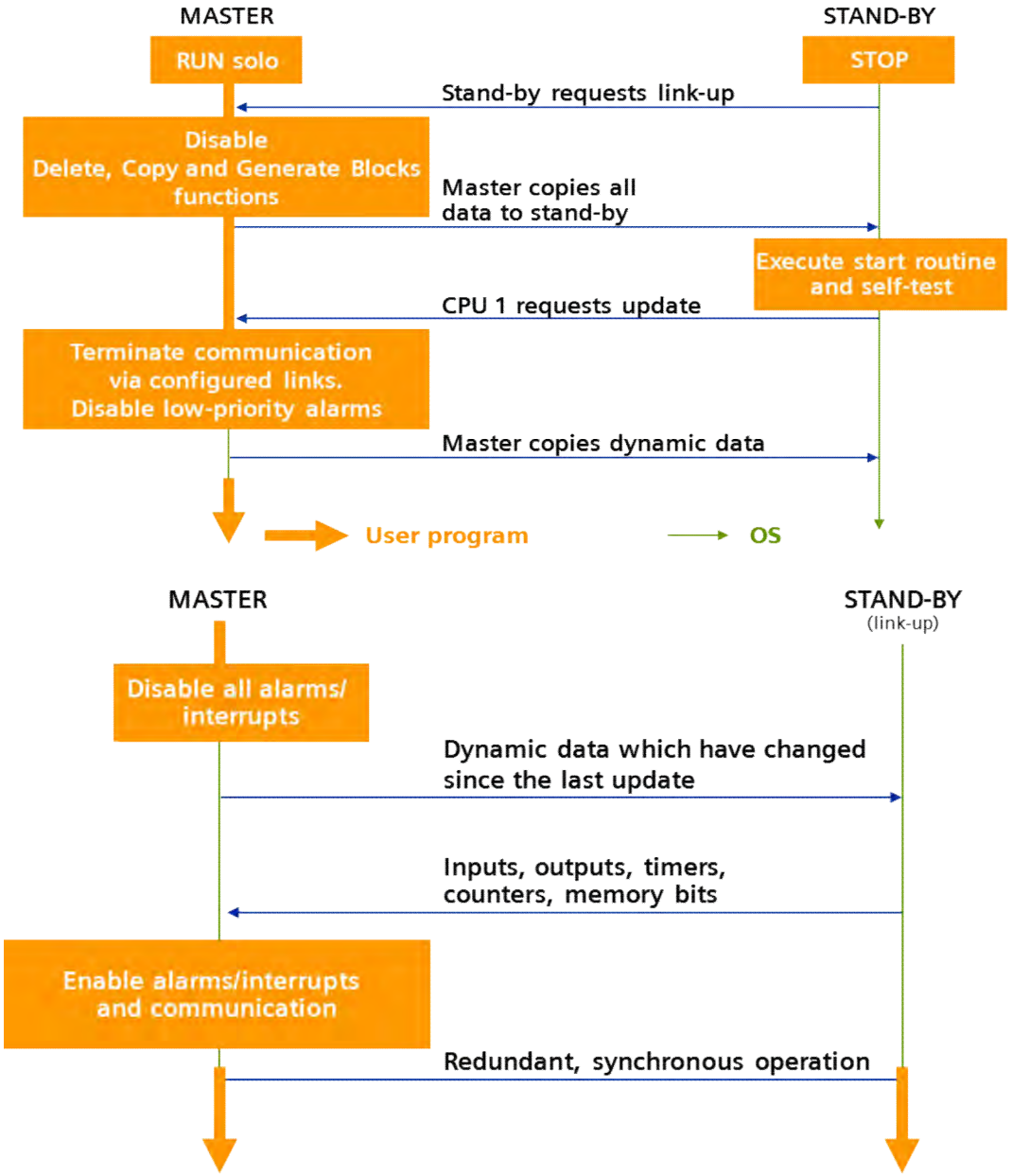
٥- ماژول‌های تغذیه (*PS*)

- ماژول‌های *PS* استاندارد از طریق قطع تغذیه (*Power off*)
- ماژول‌های *PS* افزونه در مد *RUN*؛

٦- اجزاء *DP* (*DP Components*)

- تعویض ماژول *IM 153-2* در مد *RUN*؛
- تعویض ماژول‌های *I/O* در مد *RUN*؛

مطابق شکل ۳-۱۳، *CPU*های سری *S7-400H* پس از تعمیر سیستم *Standbye* و قرار دادن ماژول در رک افزونه برای برگردان آن به حالت افزونه و بروز آوری آن، پروسه بازگشت مجدد خودکاری را فراهم می‌کنند.



شکل ۳-۱۳- بازگشت خودکار سیستم *Standby* به پیکربندی افزونه پس از تعمیر

۳-۴-۷- قابلیت *CiR* (*Configuration Changes in Run Mode*)

برای پیکربندی سخت‌افزار یک سیستم *H* شامل کنترل‌کننده‌ها، ماژول‌های *I/O* و غیره از ابزار *HWConfig* استفاده می‌شود. در هنگام دانلود تنظیمات پیکربندی در محیط *HW-Config*، به‌طورمعمول *CPU*ها به منظور اعمال تغییرات متوقف می‌شوند. لذا ازآنجایی‌که توقف سیستم‌های کنترل دل‌چسب نبوده یا در برخی موارد به خاطر پیچیدگی سیستم، غیرممکن است، تغییرات جزئی یا اضافه کردن چند ماژول *I/O* باید بدون توقف سیستم‌های کنترل امکان‌پذیر باشد. برای این منظور سری *S7400-H* برای نصب یا برداشتن ماژول‌های

سخت‌افزاری در سیستم کنترل در زمانی که کنترل‌کننده در حال اجرا است از روش *CiR* یا تغییرات پیکربندی سیستم در حالت اجرا استفاده می‌کند. درصورتی‌که یک ایستگاه *I/O* در حالت *Runtime* به پیکربندی اضافه شود یا از قبل در یک ایستگاه *ET200M*، اسلاتی برای ماژول‌های *CiR* رزرو شده باشد. این ایستگاه‌ها در محیط *HWConfig* به رنگ نارنجی دیده می‌شوند. *CPU*های سری *S7400-H* قابلیت‌های پیکربندی آنلاین زیر را فراهم می‌کنند.

- پیکربندی حافظه *CPU* (*CPU memory configuration*)؛
- تغییر پارامترهای *CPU*؛
- حذف یا اضافه کردن ماژول‌های:
  - ماژول‌های *I/O or CP*؛
  - *DP slaves*؛
  - *PA interface and PA slaves*؛
  - *Y-link and slaves*؛
  - ماژول‌های *Sync*؛
  - *Modules in modular DP slaves*
  - *Redundant IM 153-2*
  - *Redundant Power Supplies*

اجزای افزونه که می‌توانند به‌صورت تغذیه خاموش، جایگذاری شوند عبارت‌اند از:

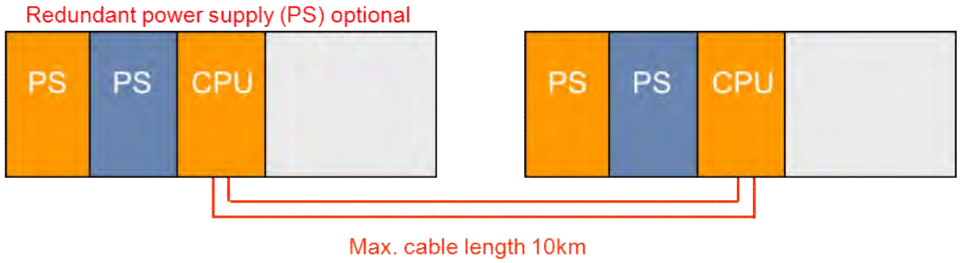
- *Standard power supplies;*
- *Central IM;*
- *CPU;*

ماژول *CPU* به محض جایگذاری در رک بروز آوری (*Program and Data*) می‌شود.

۳-۴-۸- پیکربندی کنترل‌کننده (*Configuration*)

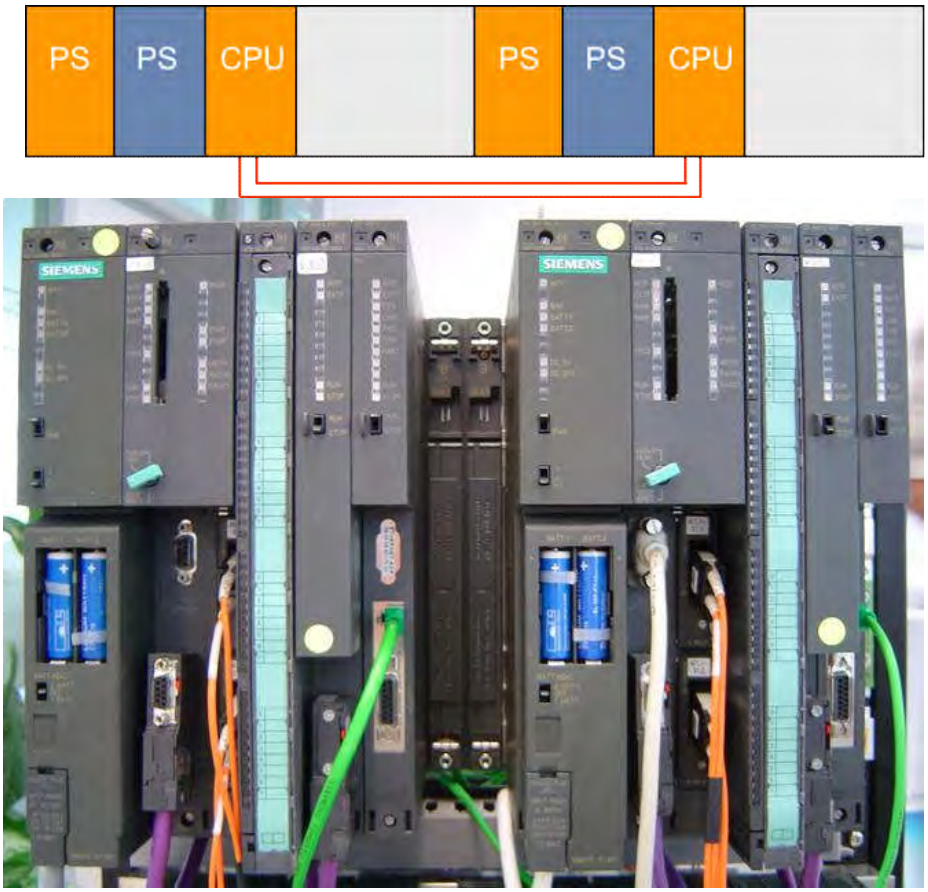
کنترل‌کننده‌های مرکزی را می‌توان به دو روش مختلف پیکربندی کرد:

۱- اگر قرار است که دو زیرسیستم افزونه به دلایل در دسترس بودن به طور کامل از هم جدا شوند، مناسب است که از دو رک استاندارد (*UR1* و *UR2*) استفاده شود. دراین روش در هر یک از رک‌ها یک *CPU* و یک منبع تغذیه (*PS*) قرار می‌گیرد. همچنین اگر میزان در دسترس‌پذیری بالایی لازم باشد، دو ماژول منبع تغذیه می‌تواند به صورت افزونه پیکربندی شود. در این آرایش فاصله بین دو رک می‌تواند تا ۱۰ کیلومتر باشد.



شکل ۳-۱٤- پیکربندی *CPU*های *H* با دو رک استاندارد به فاصله دور از همدیگر

۲- در پیکربندی دیگر، دو *CPU* هر یک با یک منبع تغذیه تکی یا افزونه، به رک نوع *UR2-H* با یک گذرگاه *Backplane* جدا متصل می‌شود.



شکل ۳-۱۵- نمونه پیکربندی سیستم *H* بر روی یک *backplane* با رک *UR2H*

مشخصات برخی از ماژول‌های *CPU* به شرح زیر می‌باشد:

- میزان حافظه:

- *417-4H from 4 MB to 20MB;*
- *414-4H from 768KB to 1,4MB;*

- فراهم کردن قابلیت اطمینان بالا (*Higher Reliability*)

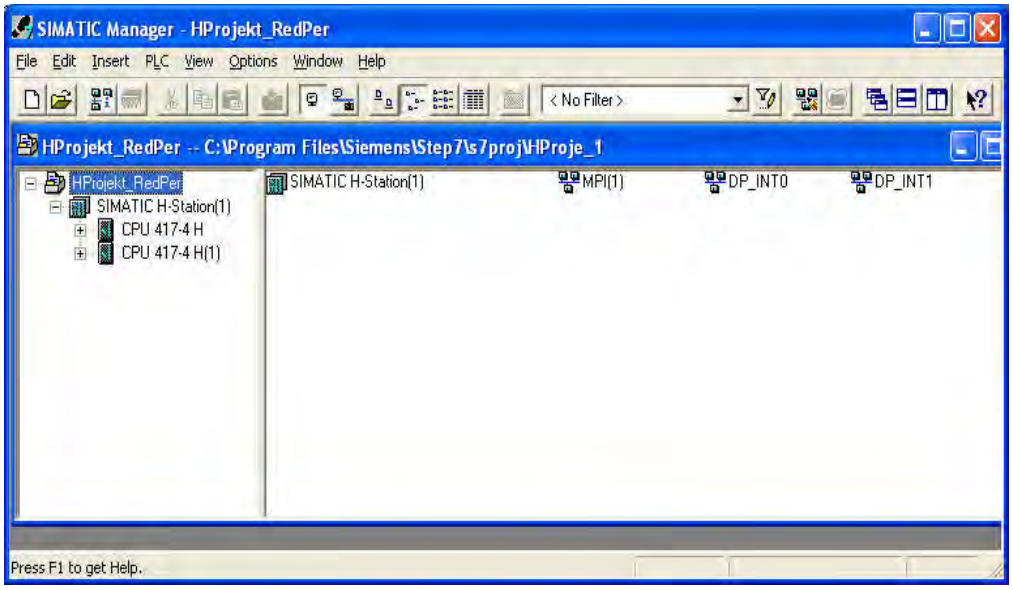
حافظه *CPU* دارای فناوری *EDC* (*Error Detection and Correction*) برای تشخیص و تصحیح خودکار خطااست.

- پشتیبانی از فاصله بین کنترل‌کننده‌ها تا ۱۰ کیلومتر (در *CPU*های قبلی تا ۵۰۰ متر پشتیبانی می‌شد) فاصله بین کنترل‌کننده‌های *H* بسته به نوع رک و کابل فیبر نوری دارای پیکربندی‌های زیر است:
  - *FO-Cable 1m;*
  - *FO-Cable 2m;*
  - *FO-Cable 10m;*
  - *Monomode FO-Cable LC/LC Duplex crossed 9/125µ: 10km;*
  - با دو رک استاندارد جداگانه (*10km*)
  - با یک رک نوع *H* و گذرگاه پشتی جداگانه (*with split backplane bus*) –(*1m,2m,10m*)
- دارای ٤ واسط (*Integrated interfaces*):
  - دو واسط ماژول‌های *Sync*؛
  - یک واسط *DP*؛
  - یک واسط *MPI/DP*؛

۳–۵– پیاده‌سازی سیستم *400H* (*System integration*)

۳–۵–۱– برنامه‌نویسی

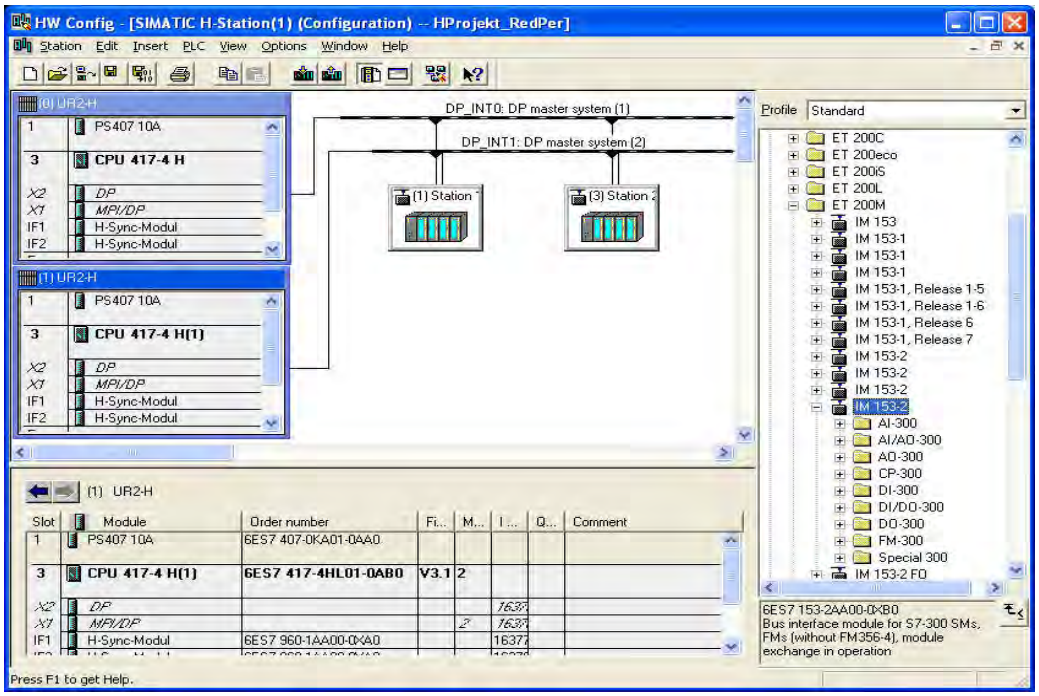
برای برنامه‌نویسی الگوریتم کنترل سیستم *H* از همان ابزار سیستم استاندارد یعنی *Simatic Manager* به همراه کتابخانه‌های مختلف استفاده می‌شود. به طوری که زیمنس از آن تحت عنوان *Transparent Programming* یاد می‌کند. لذا از کلیه زبان‌های برنامه‌نویسی بکار رفته در سیستم استاندارد بدون هیچ محدودیتی می‌توان استفاده کرد.



شکل ۳-۱۶- نمایی از پروژه *S7-400H* در محیط *Simatic Manager*

۳-۵-۲- پیکربندی سخت افزاری (*Hardware configuration*)

برای پیکربندی سیستم *H* از همان ابزار *HWConfig* بکار رفته برای سیستم استاندارد استفاده می‌کند. سخت‌افزار یک سیستم *H* پلات‌فرم پایه برای سیستم‌های *F* و *FH* می‌باشد.



شکل ۳-۱۷- نمایی از پیکربندی سیستم *H* در محیط *HwConfig*

۳-۶- افزونگی (*Redundancy principle*)

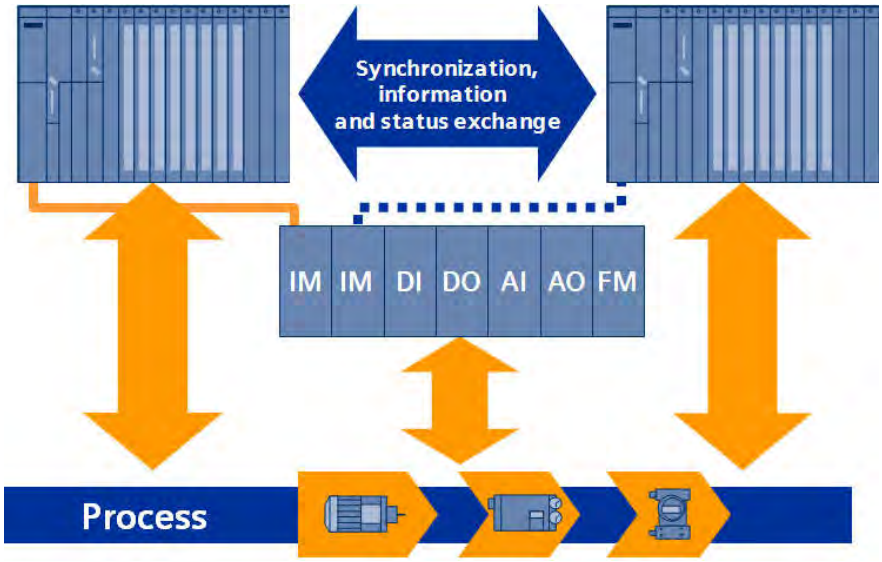
افزونگی سیستم‌های کنترل اتوماسیون برای دو هدف کلی زیر صورت می‌گیرد.

- داشتن یک سیستم کنترل با مشخصه تحمل خطا (*Fault-tolerant*) برای دسترس‌پذیری بالا (*High Availibility*) است. به طوری که هدف نهایی، کاهش احتمال تلفات تولید از طریق سوئیچ به یک سیستم آماده به کار می باشد.
- داشتن یک سیستم کنترل با مشخصه ایمن در نقص در سیستم (*Fail-safe*) با هدف حفاظت از زندگی افراد، محیط زیست و سرمایه‌ها از طریق قطع امن تجهیزات

در سیستم‌های کنترل دو نوع افزونگی در سطح کنترل‌کننده وجود دارد که عبارت انداز:

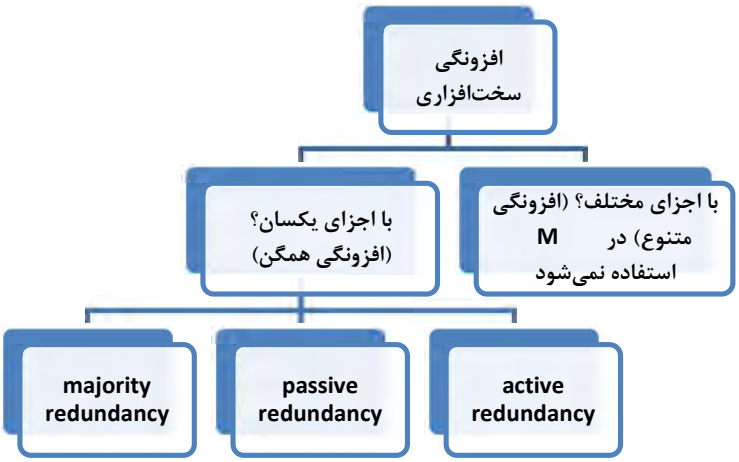
- افزونگی سخت‌افزاری – برای فرآیندهای سریع– زمان سویچ از سیستم اصلی به رزرو در حد میلی‌ثانیه می‌باشد.

- افزونگی نرم‌افزاری – برای فرآیندهای کند– زمان سویچ از سیستم اصلی به رزرو در حد ثانیه می‌باشد.

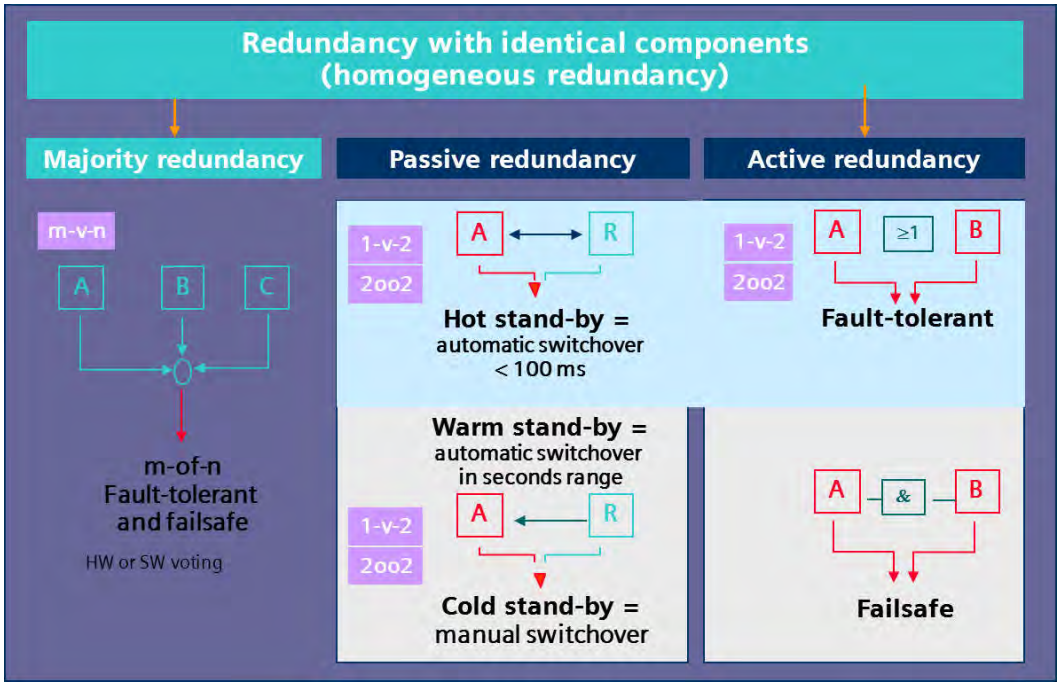


شکل ۳-۱۸– افزونگی در سیستم S7-400H

۳–۶–۱– افزونگی سخت‌افزاری (*Hardware Redundancy*)



۳-۶-۲- افزونگی با اجزاءیکسان (*homogeneous redundancy*)



انواع افزونگی سخت‌افزاری با اجزاءیکسان

۳-۶-۳- افزونگی پسیو (*Passive redundancy*)

به سه صورت پیاده‌سازی می‌شود. که عبارت انداز:

Passive Redundancy (1-v-2 , 2oo2: 2 out of 2)

- *Hot Standby : automatic switchover < 100 ms*
- *Warm Standby : automatic switchover in seconds range*
- *Cold Standby : manual switchover*

۳-۶-۴- افزونگی فعال (*Active redundancy*)

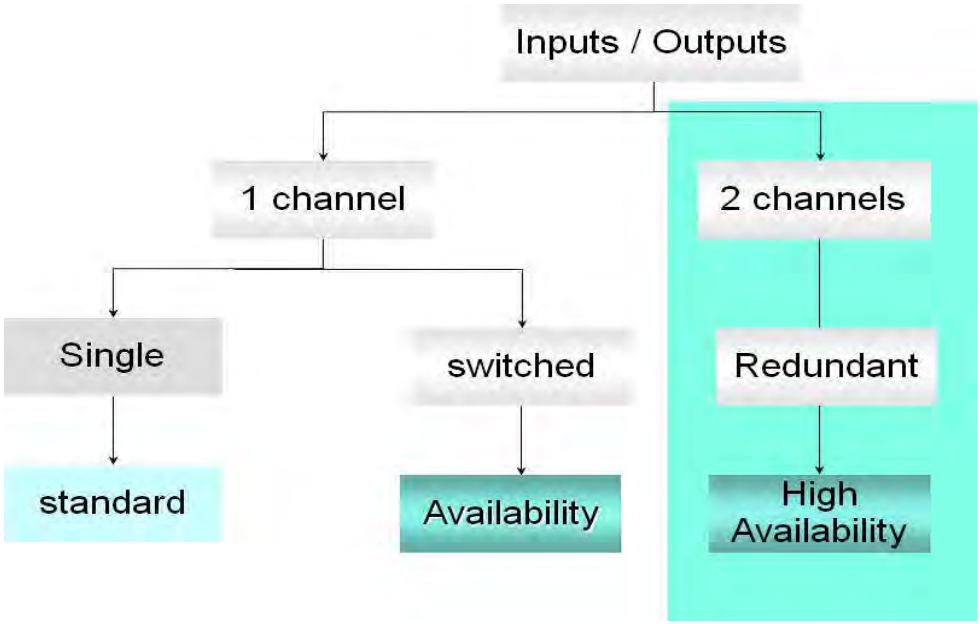
برای سیستم‌های تحمل پذیر خطا تنها یکی از *CPU*ها نتایج پردازش را در خروجی می‌نویسند، در واقع عمل *OR* بین *CPU*ها صورت می‌گیرد. ولی در مقابل برای سیستم‌های ایمن در خطا (*Failsafe*)، نتیجه پردازش توسط هر دو *CPU* در خروجی نوشته می‌شود.

۳-۷- پیکربندی و اتصال *I/O* در سیستم *H* (*I/O configuration*)

مطابق شکل ۳-۱۹ اتصال ماژول‌های ورودی/خروجی در سیستم *H* بسته به میزان «دسترس‌پذیری» مورد نیاز به چند روش می‌تواند، انجام شود. در این روش‌ها ماژول‌های *I/O* می‌توانند به صورت مرکزی در کنار خود *CPU* نصب شوند و یا بر روی رک *RIO* در شبکه *Profibus* قرار بگیرند.

- استفاده از یک کانال *I/O* واحد – برای دسترس‌پذیری استاندارد،

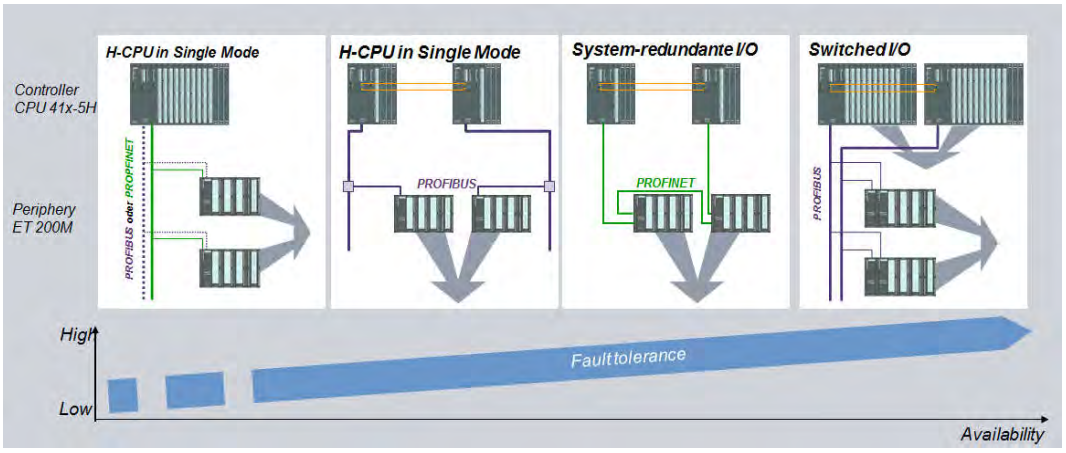
- استفاده از یک کانال *I/O* واحد با دو ماژول واسط *IM* افزونه که بین دو *CPU* سوئیچ می‌شود (پیکربندی *Switched I/O*) برای دسترس‌پذیری بالا؛
- استفاده از دو کانال *I/O* افزونه:



شکل ۳-۱۹- میزان حصول دسترس‌پذیری *I/O* در سیستم‌های *S7-400H*

ساختارهای افزونه ممکن برای ماژول‌های ورودی/خروجی *I/O* عبارت‌انداز:

- *H-CPU in single mode with Redundant I/O Modules*
- *Central I/O Modules*
- *Distributed I/O Modules*
- *Distributed Switched I/O Modules*
- *Full Redundant I/O*



شکل ۳-۲۰- میزان دسترس‌پذیری با پیکربندی‌های مختلف سیستم *H* با *ET200M*

۳-۷-۱- اصول I/O افزونه

I/Oهای افزونه ماژول‌های ورودی و خروجی هستند که به صورت افزونه پیکربندی و کار می‌کنند. در این حالت حداکثر میزان در دسترس‌پذیری حاصل می‌شود بدلیل این که در این روش نقص در CPU، خرابی در خط PROFIBUS و یا حلقه PROFINET و ماژول سیگنال جبران می‌شود.

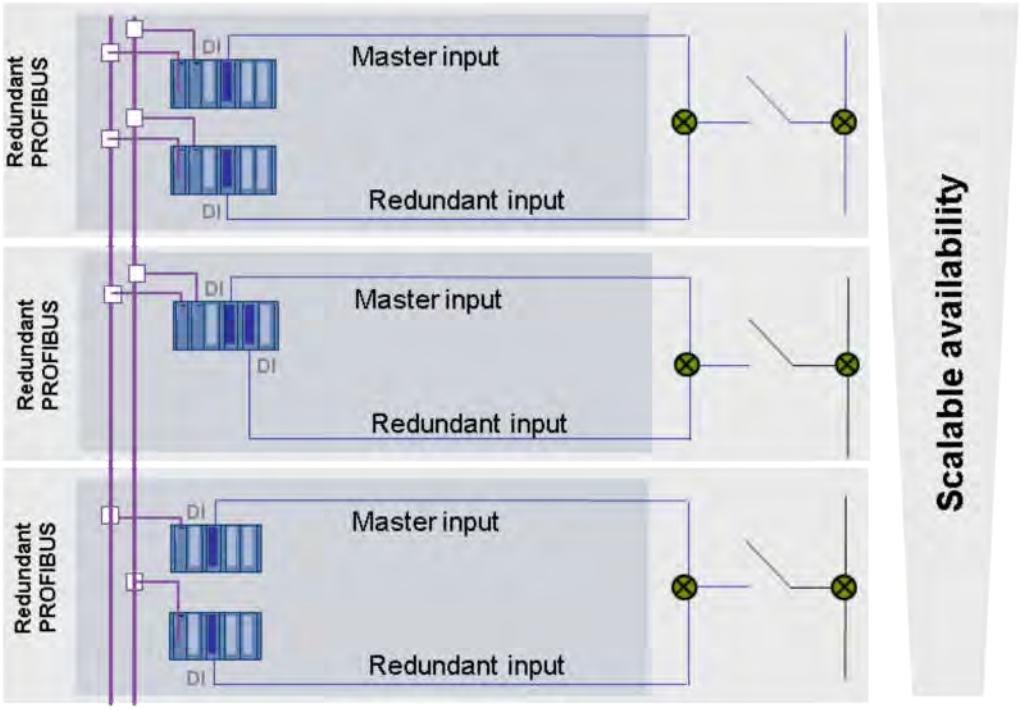
در طول عملکرد عادی، هر دو ماژول فعال هستند. در ماژول‌های ورودی افزونه، مقدار سنسور به طور مشترک توسط دو ماژول قرائت شده، نتیجه قرائت دو ورودی مقایسه شده و به عنوان یک مقدار واحد برای پردازش بیشتر در در اختیار برنامه لاجیک قرار می‌گیرد.

در مورد خروجی‌های افزونه نیز مقدار محاسبه شده توسط برنامه کاربر به خروجی‌های هر دو ماژول ارسال می‌شود. ولی یکی از ماژول‌ها در وسیله خروجی می‌نویسد.

در صورت وقوع خطا، به عنوان مثال خراب شدن یکی از دو ماژول ورودی، ماژول معیوب، دیگر آدرس‌دهی نمی‌شود. وقوع خطا اطلاع داده شده و پردازش تنها با ماژول بی عیب همچنان ادامه می‌یابد. بعد از تعمیر که ممکن است به صورت آنلاین انجام گیرد، هر دو ماژول می‌توان دوباره آدرس‌دهی شود.

۳-۷-۲- افزونگی با I/O های مبتنی بر شبکه پروفی‌باس (FMR—Flexible Modular Redundancy)

- *Scalable fault tolerance*
  - *Single or redundant connection*
  - *Can be mixed in one and same system*
- *Advantage*
  - *Redundancy has to only be made available where it is actually needed .*
  - *More cost-efficient solutions than conventional architectures with single design.*



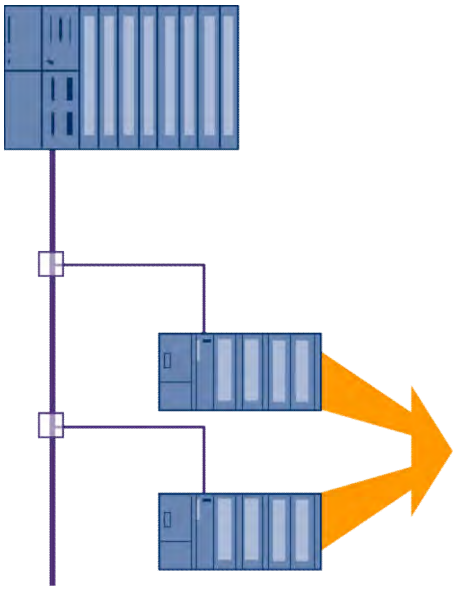
شکل ۳-۲۱-

۳-۷-۳- پیکربندی استفاده از یک کانال *I/O* واحد بدون ساختار افزونه (*Single Mode*)

در این روش کنترل‌کننده *S7-400H* در یک ساختار افزونه پیکربندی نشده است. به‌طوری که هر سنسور به یک کانال تنها از یک کارت *I/O* متصل می‌گردد.

۳-۷-۴- استفاده از ماژول‌های *I/O* افزونه با یک سیستم *H* بدون افزونه (*H-CPU in single mode*)

در این پیکربندی، هر سنسور به دو کانال *I/O* که در دو رک *RIO* جداگانه قرار گرفته است، متصل می‌شود. به طوری که کارت‌های *I/O* توسط یک *H-CPU* در یک ساختار غیر افزونه مورد دسترسی قرار می‌گیرد.

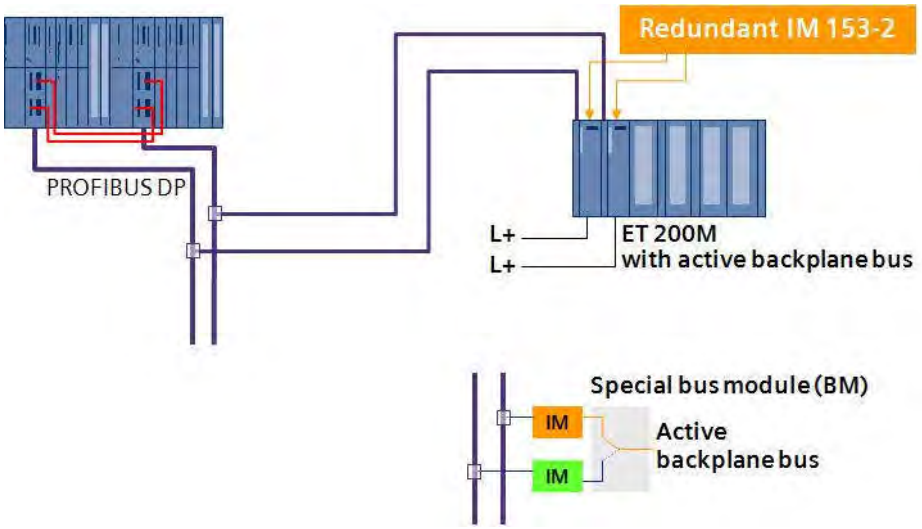


شکل ۳-۲۲- استفاده از دو کانال *I/O* افزونه در دو رک *RIO* جداگانه با یک سیستم *H* بدون افزونه

۳-۷-۵- استفاده از ماژول‌های *I/O* مشترک بین دو *CPU* (*Distributed Switched I/O*)

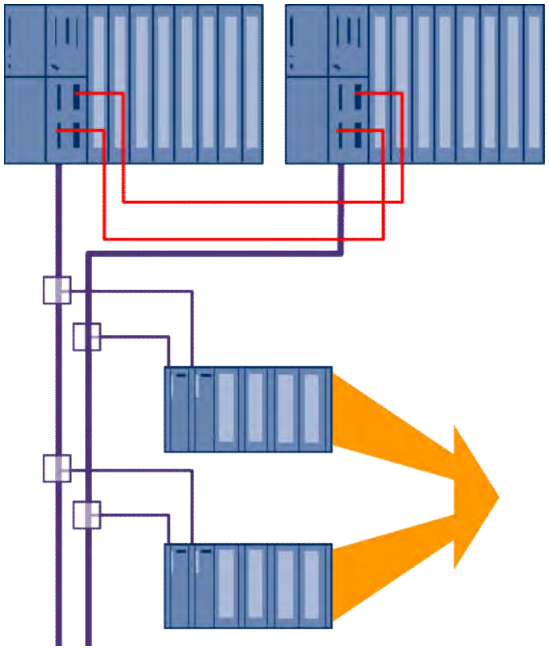
در این روش مطابق شکل ۳-۲۳ یک کارت *I/O* بین دو *CPU* مشترک بوده و توسط دو ماژول *IM* و از طریق یک *Active backplane bus* سوییچ می‌شود.

در «ساختار افزونه ماژول‌های *I/O* قابل سویچ بین دو سیستم *H* افزونه» یک کانال از ماژول‌های *I/O*، در رک‌های *RIO* جداگانه به صورت افزونه به یک سنسور متصل شده و بین دو سیستم *H* افزونه سویچ می‌شود.



شکل ۳-۲۳- ساختار *Switched I/O*

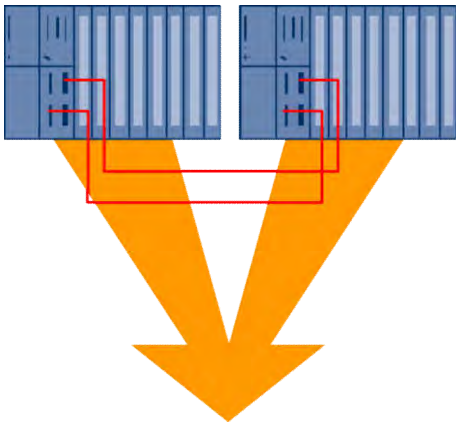
باس ماژول استفاده برای جاگذاری ماژول‌های *IM* دارای شماره سفارش *7HD10* می‌باشد.



شکل ۳-۲٤- اتصال دو کارت *I/O* در دو رک *RIO* جداگانه در یک ساختار افزونه سیستم *H*

۳-۷-۶- استفاده از ماژول‌های *I/O* افزونه در رک مرکزی (*Central I/O Modules*) دو سیستم *H* افزونه

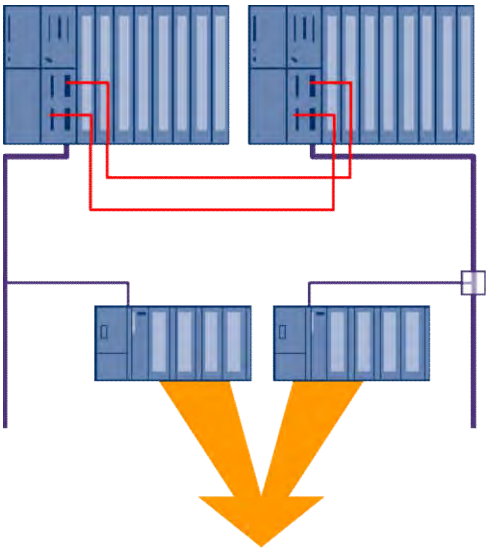
در این پیکربندی دو ماژول *I/O* قرار گرفته در رک مرکزی به صورت ساختار افزونه نصب می‌شوند.



شکل ۳-۲۵- ساختار افزونه ماژول‌های *I/O* در رک مرکزی دو سیستم *H* افزونه

۳-۷-۷- ساختار *I/O* افزونه توزیع شده (*Distributed I/O modules*)

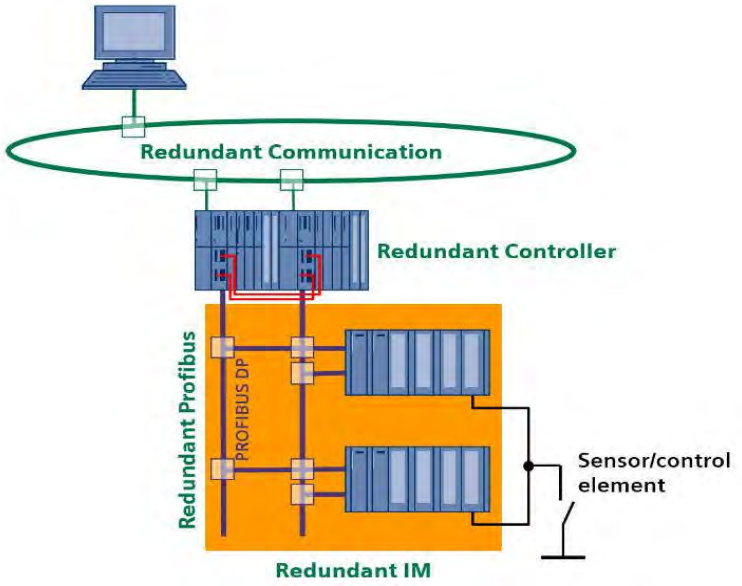
دو ماژول *I/O* در رک‌های جداگانه *RIO* (توزیع شده) به صورت افزونه باهم، نصب می‌شوند.



شکل ۳-۲۶- ساختار افزونه *I/O* در رک‌های جداگانه *RIO* در یک سیستم *H* افزونه

۳-۷-۸- ساختار افزونه کامل برای *I/O* برای دسترس‌پذیری خیلی بالا

در این روش که افزونگی کاملی را فراهم می‌کند. اجزای *I/O* نظیر کارت *I/O*، کارت *IM* و کابل ارتباطی دارای ساختار افزونه دوتایی است. شکل ۳-۲۷ ساختار افزونگی کارت‌های *I/O* و گذرگاه *I/O* را به همراه افزونگی شبکه و کنترل‌کننده نشان می‌دهد.



شکل ۳-۲۷- ساختار افزونگی کارت‌ها و گذرگاه *I/O* در *PCS7* برای دسترسی بالا

توصیف اجزای ایستگاه *RIO* (*ET200M*) به شرح زیر می‌باشد.

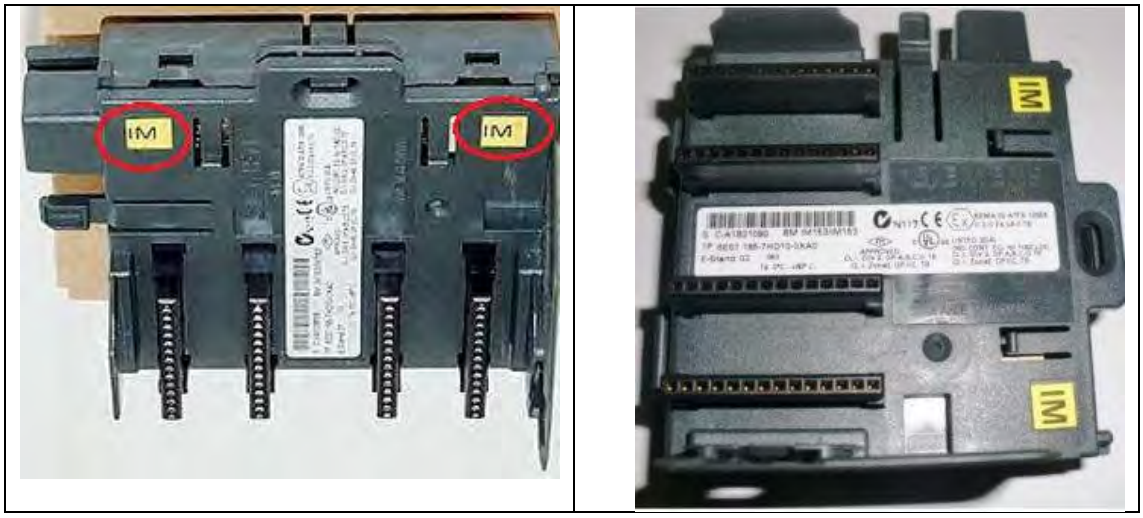
۱- استفاده از یک رک آلومنیومی با گودی بیشتر از یک رک معمولی

6ES7 195-1GF30-0XA0	*SIMATIC DP, RAIL FOR ET 200M 530 MM LONG F. MAX. 5 BUS MOD. FUNCTION: INSERT/REMOVE*

شکل ۳-۲۸- رک برای نصب و مونتاژ ایستگاه *Switched I/O*

۲- استفاده از قطعات *Bus Unit* مخصوص نصب ماژول‌های *IM153-2* با مشخصات زیر:

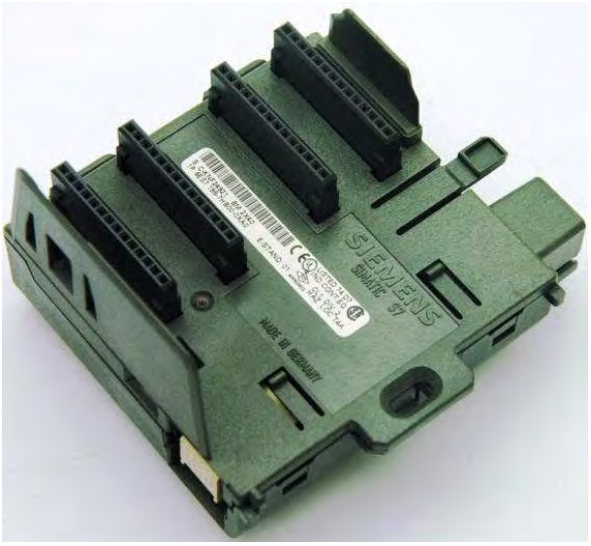
6ES7 195-7HD10-0XA0	*SIMATIC DP, BUS Unit for ET200M for Integrating 2 IM153-2 RED. function: Insert/Remove While Operating Mode Run*



شکل ۳-۲۹- *Bus Unit* برای نصب ماژول‌های *IM153-2*

۳- استفاده از قطعات *Bus Unit* مخصوص نصب ماژول‌های *I/O* با مشخصات زیر:

6ES7195-7HB00-0XA0	*SIMATIC DP, BUS UNIT FOR ET200M F. THE INTEGR.OF TWO 40 MM WIDE I/O SUBMODULES FOR INSERT/REMOVE*



شکل ۳-۳۰- *Bus Unit* برای نصب ماژول‌های *I/O*

٤- ماژول‌های واسط *IM 153-2*

6ES7 153-2BA00-0XB0	*SIMATIC DP, ET 200M INTERFACE IM 153-2 HIGH FEATURE FOR MAX. 8 S7-300 MODULES, WITH REDUNDANCY, TIME STAMPING FIT FOR ISOCHRONE MODE*

در این مد هر دو *DP Master* فعال بوده و به درستی عمل می‌کنند؛ و عملیات *I/O* به شرح زیر صورت می‌گیرد:

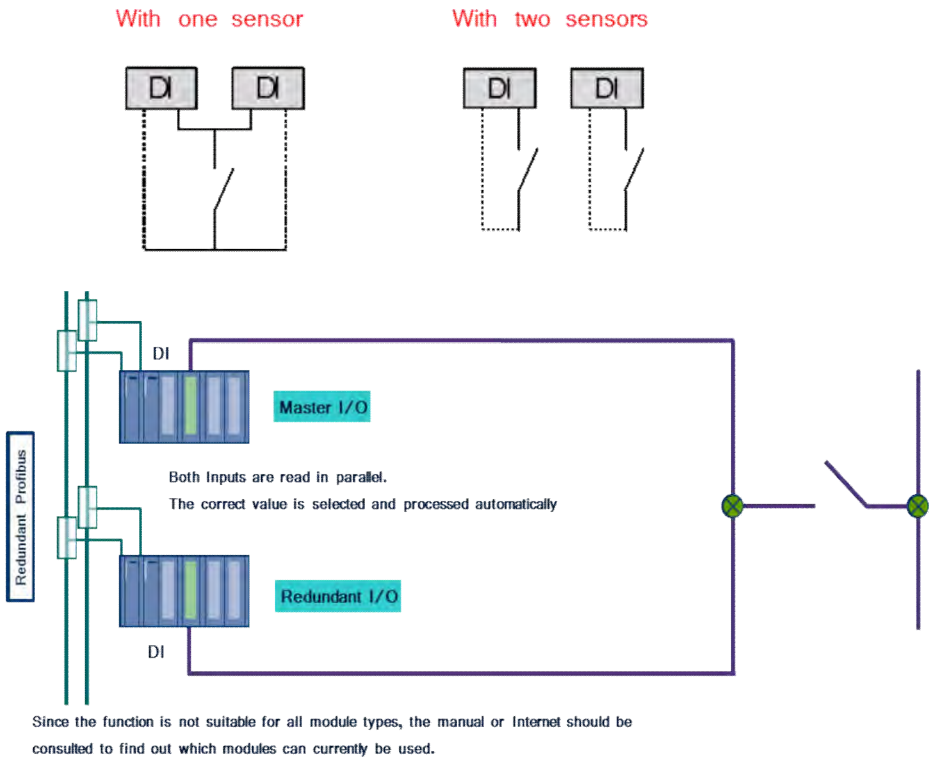
- **قرائت ورودی (*Reading inputs*)**: ورودی‌ها تنها از کانال *IM* فعال خوانده می‌شوند
- **نوشتن خروجی‌ها(*Writing outputs*)**: داده توسط هر دو کانال دریافت می‌شود ولی تنها از کانال *IM* فعال به خروجی‌ها ارسال می‌شود.

۳-۸- سیسم‌بندی کانال‌های افزونه (*Wiring*)

۳-۸-۱- سیسم‌بندی کانال‌های ورودی دیجیتال افزونه (*Wiring*)

مطابق نحوه سیم‌بندی کانال‌های ورودی افزونه دیجیتال به دو صورت می‌باشد

- اتصال کانال‌های افزونه به یک سنسور
- اتصال کانال‌های افزونه به دو عدد سنسور

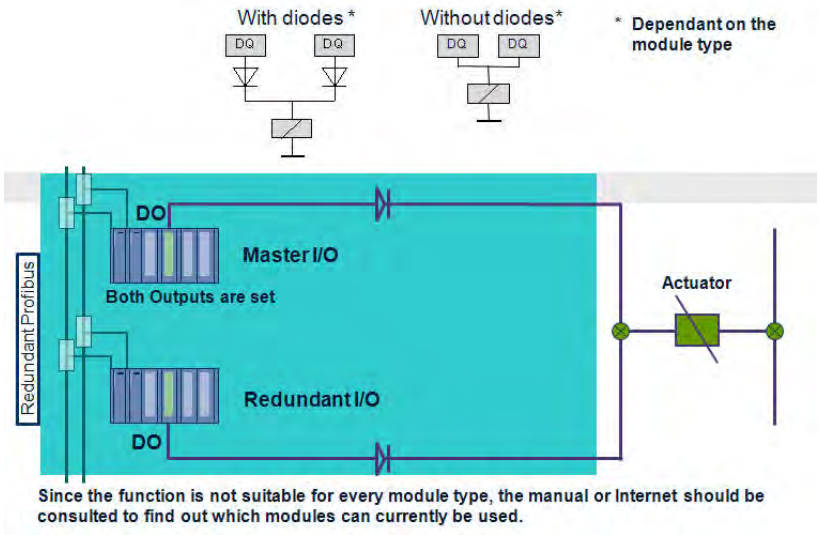


شکل ۳-۳۱- سیسم‌بندی کانال‌های ورودی دیجیتال افزونه

مطابق شکل بالا هر دو ورودی قرائت می‌شود. مقدار درست انتخاب و پردازش می‌شود.

نکته: از آنجایی که ارایش نشان داده شده در شکل بالا برای همه ماژول‌های ورودی مناسب نمی‌باشد. لذا برای هر نوع ماژول *I/O* بایستی به فایل راهنمای مربوطه مراجعه شود.

۳-۸-۲- سیسم‌بندی کانال‌های خروجی دیجیتال افزونه

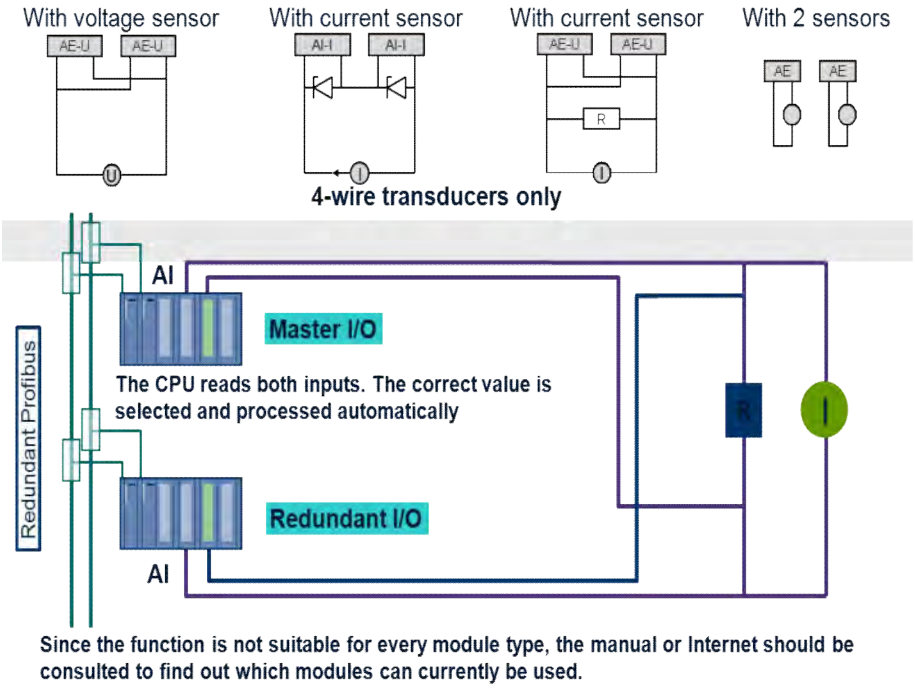


شکل ۳-۳۲- سیم‌بندی خروجی‌های دیجیتال افزونه

نکته: از آنجایی که ارایش نشان داده شده در شکل بالا برای همه ماژول‌های ورودی مناسب نمی‌باشد. لذا برای هر نوع ماژول *I/O* بایستی به فایل راهنمای مربوطه مراجعه شود.

۳-۸-۳- سیسم‌بندی کانال‌های ورودی آنالوگ افزونه

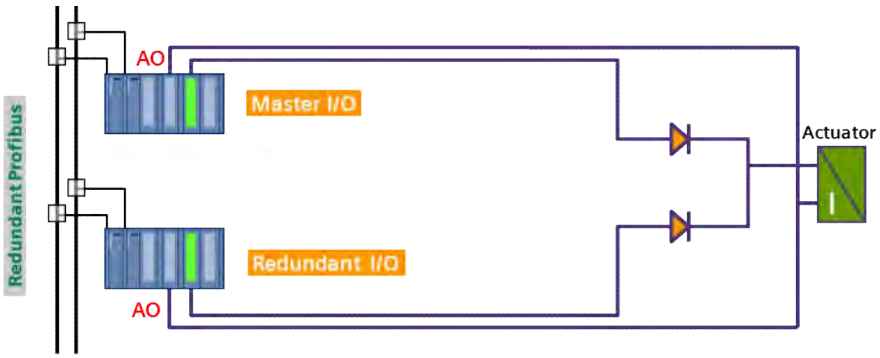
در کانال‌های آنالوگ بسته به نوع سنسور که ولتاژی باشد یا جریانی و یا از یک سنسور یا دو عدد سنسور استفاده شده باشد، سیم‌بندی متفاوت می‌باشد.



شکل ۳-۳۳- سیم‌بندی ورودی‌های آنالوگ افزونه

۳-۸-۴- سیسم‌بندی کانال‌های خروجی‌های آنالوگ افزونه

هر کانال خروجی، نیمی‌از مقدار خروجی را فراهم می‌کند. هنگامی‌که یکی از ماژول‌ها نقص پیدا می‌کند، کانال خروجی که سالم است مقدار کامل خروجی را فراهم می‌کند.



شکل ۳-۳٤- سیم‌بندی خروجی‌های آنالوگ افزونه

۳-۸-۵- سطوح کیفیت در افزونگی I/O (Redundant quality stages)

بسته به این که زیر سیستم I/O به چه صورت پیکربندی شود، سه سطح از کیفیت برای افزونگی I/O فراهم می‌شود که عبارت‌انداز:

- سطح کیفیت بالا (Highest quality level)

در صورتی که در پیکربندی I/O یک سیستم H، از ماژول‌های F-I/O، (Failsafe I/O) با توابع تشخیصی (Diagnostic Functions) استفاده شود، سطح کیفیت بالا برای افزونگی سیستم تعریف می‌شود. این سطح کیفیت، برای عملیات ایمن(failsafe) در یک پلنت مورد نیاز می‌باشد.

- سطح کیفیت متوسط (Medium quality level)

در این سطح کیفیت، از ماژول‌های استاندارد (نه از نوع Fail-safe) با عملکرد تشخیصی (Diagnostic Functions) استفاده می‌شود.

- سطح کیفیت کم هزینه(Low-cost quality level)

از ماژول‌های استاندارد بدون عملکرد تشخیصی استفاده کنید.

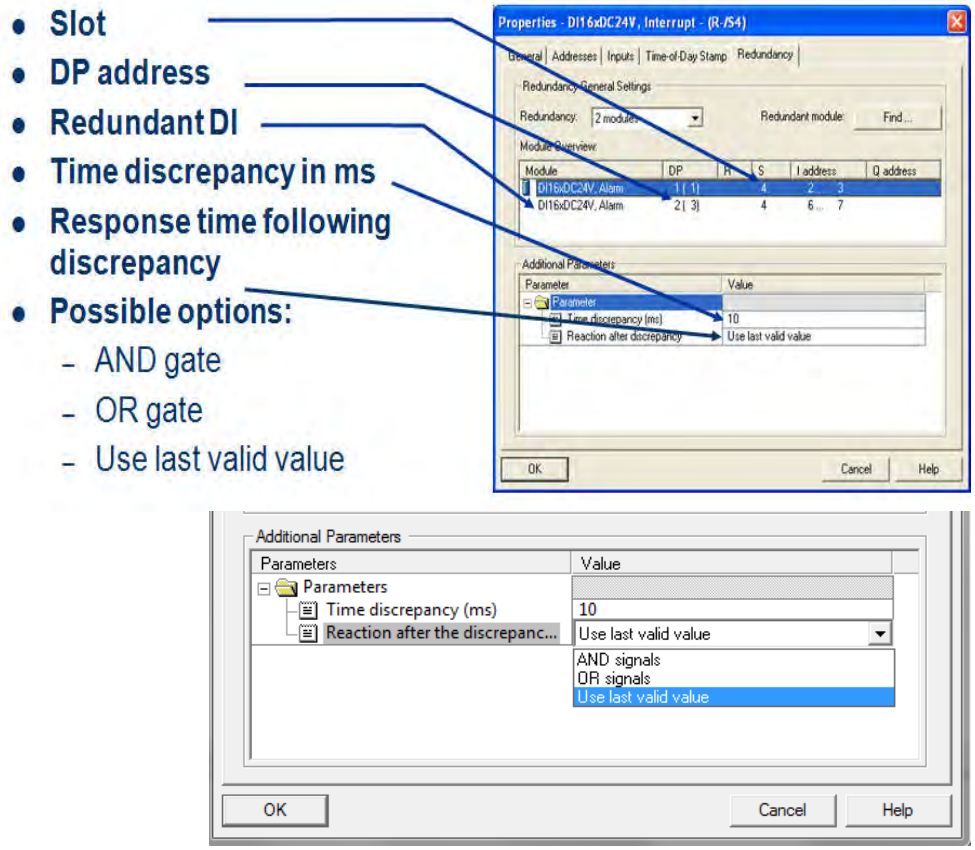
۳-۹- یکپارچگی با برنامه کاربر

برای پیاده‌سازی برنامه کاربر یا لاجیک I/Oهای افزونه، در برنامه Step 7 از کتابخانه‌ی افزونگی که شامل فانکشن‌بلاک‌های مختلف می‌باشد، استفاده می‌شود. قوانین موجود بیان می‌کند که در سیستم H با I/O افزونه، همیشه پایین‌ترین آدرس باید برای برنامه نویسی استفاده شود. روش کار (Method of operation) به شرح زیر می‌باشد:

- ورودی توسط بلاک FB RED_IN خوانده شده و پس از آنالیز وجود اختلاف بین دو ورودی (discrepancy) مقدار خوانده شده به POI کپی می‌شود.
- در برنامه لاجیک کاربر نیز، یک خروجی تولید شده در برنامه، توسط بلاک FB RED_OUT به پایین‌ترین آدرس یک کانال خروجی به روش معمول نوشته می‌شود. سپس بلاک FB RED_OUT به طور خودکار مقدار مربوط را به آدرس دوم کپی می‌کند.

۳-۹-۱- پیکربندی I/O افزونه

شکل زیر پنجره پراپرتی یک ماژول I/O پیکربندی شده به صورت افزونه را در محیط HW Config نشان می‌دهد.



شکل ۳-۳۵- پیکربندی سخت‌افزاری *I/O* افزونه

۳-۹-۲- نحوه *Redundant* کردن یک *ET200M (IM153-3)* بین دو *Remote I/O station* در *software Redundancy*

۱. دو تا *Station* یکسان ایجاد می‌کنیم.

۲. در هر دو *Station* پورت *DP* مربوط به *CPU* را به *DP Master* تنظیم می‌کنیم.

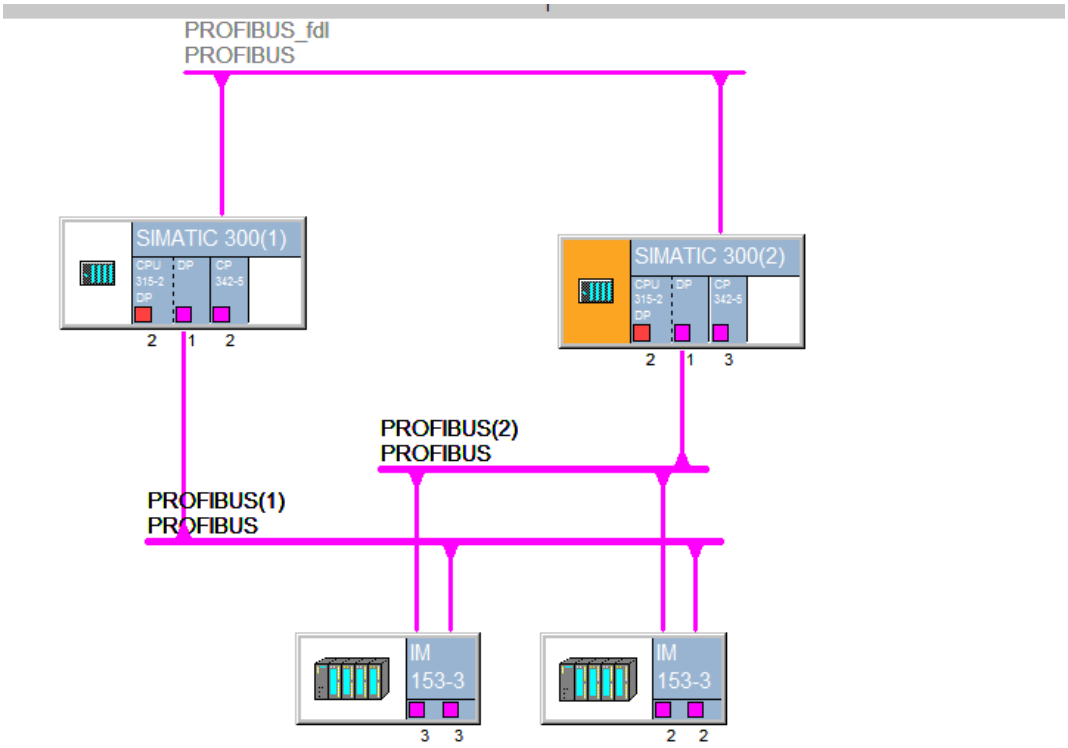
۳. در *Station A* یک *ET200 M (IM 153)* قرار می‌دهیم. سپس ماژول‌های *IO* را در این *ET* جایگذاری می‌کنیم.

٤. در *Station A* از *ET200 M (IM 153)* کپی می‌گیریم و پنجره *HWConfig* مربوطه را می‌بندیم.

۵. پنجره *HWConfig* مربوط به *Station B* را باز می‌کنیم.

٦. باس *DP Master* مربوط به *CPU Station B* را انتخاب و از منوی *EDIT* گزینه *Insert Redundant* را اجرا می‌کنیم.

۷. پس از کامپایل پروژه ، پیکربندی کلی به شکل زیر خواهد بود.



در این شکل دو تا باس *PROFIBUS* برای ارتباط *Redundant* با *ET200 IO* و یک گذرگاه *PROFIBUS* برای ارتباط *FDL* بین دو *CPU* جهت سنکرون کردن در *Software Redundancy* در نظر گرفته شده است.

۳-۹-۳- کتابخانه بلوک‌های *I/O* افزونه (*Block library Functional I/O Redundancy*)

کتابخانه مربوط به بلوک‌های توابع *I/O* افزونه در مسیر زیر قرار دارد

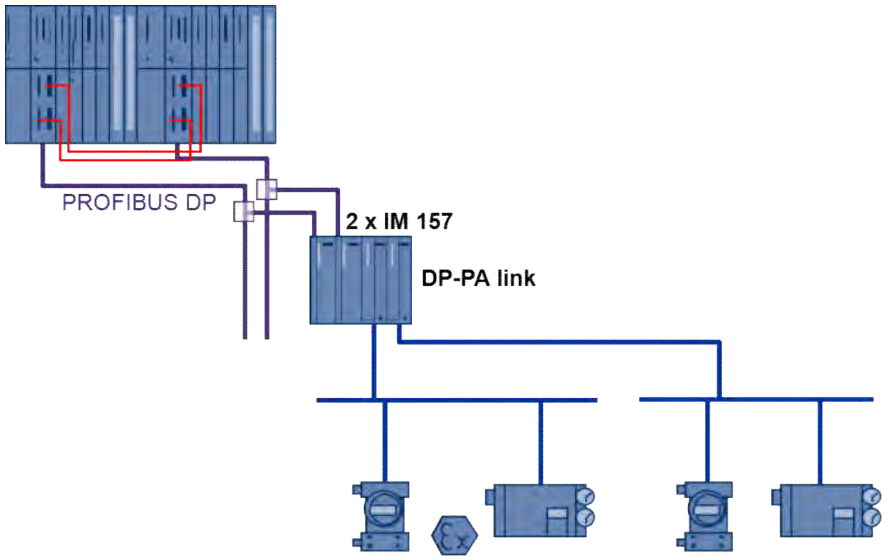
STEP 7\S7_LIBS\RED_IO

- *FC 450 "RED_INIT": Initialization function*
- *FC 451 "RED_DEPA": Trigger depassivation*
- *FB 450 "RED_IN": Function block for reading redundant inputs*
- *FB 451 "RED_OUT": Function block for Writing redundant outputs*
- *FB 452 "RED_DIAG": Function block for diagnostics of redundant I/O*
- *FB 453 "RED_STATUS": Function block for redundancy status information*

۳-۹-۴- اتصال *PROFIBUS PA* از طریق یک لینک *PA* به سیستم *H*

از آن جایی که باس *PROFIBUS PA* به طور مستقیم به *CPU* متصل نمی‌شود، اتصال این باس از طریق باس *DP* صورت می‌گیرد. لذا برای این کار از ماژول واسط *IM 157* (*DP-PA Link*) استفاده می‌شود.

- *IM 157: 6ES7 157-0AA82-0XA0*
- *Bus module BM IM 157 : 6ES7 195-7HD80-0XA0*

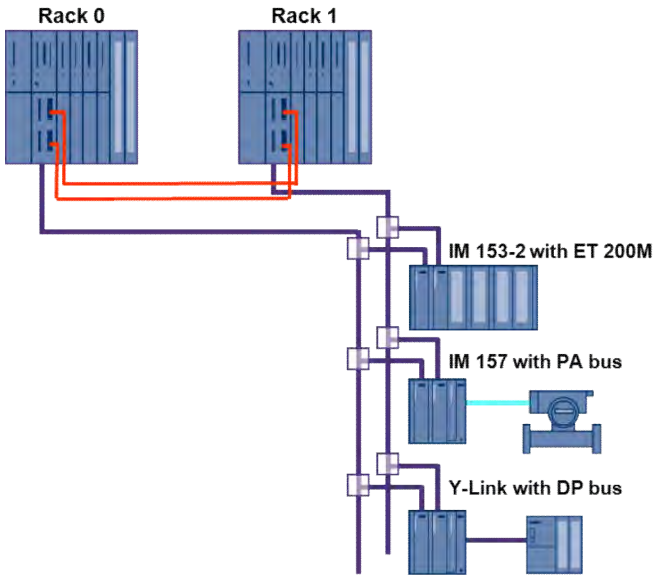


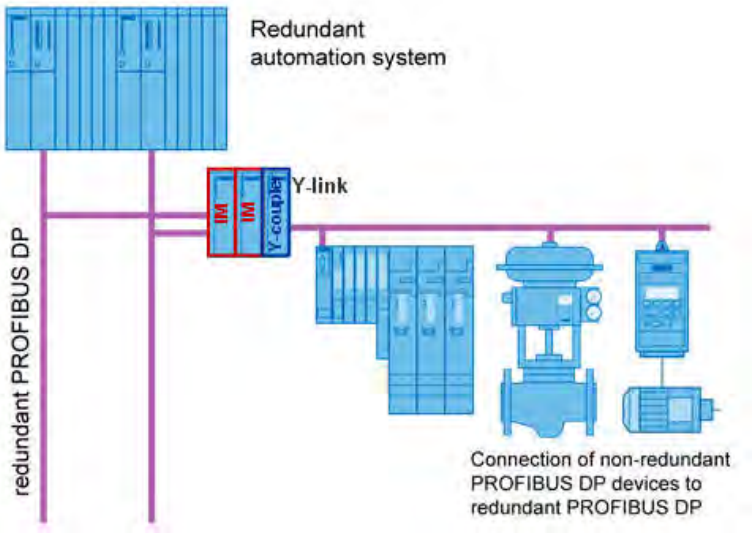
شکل ۳-۳۶- اتصال *PROFIBUS PA* از طریق یک لینک *PA* به سیستم *H*

۳-۹-۵- ماژول *Y-Link*

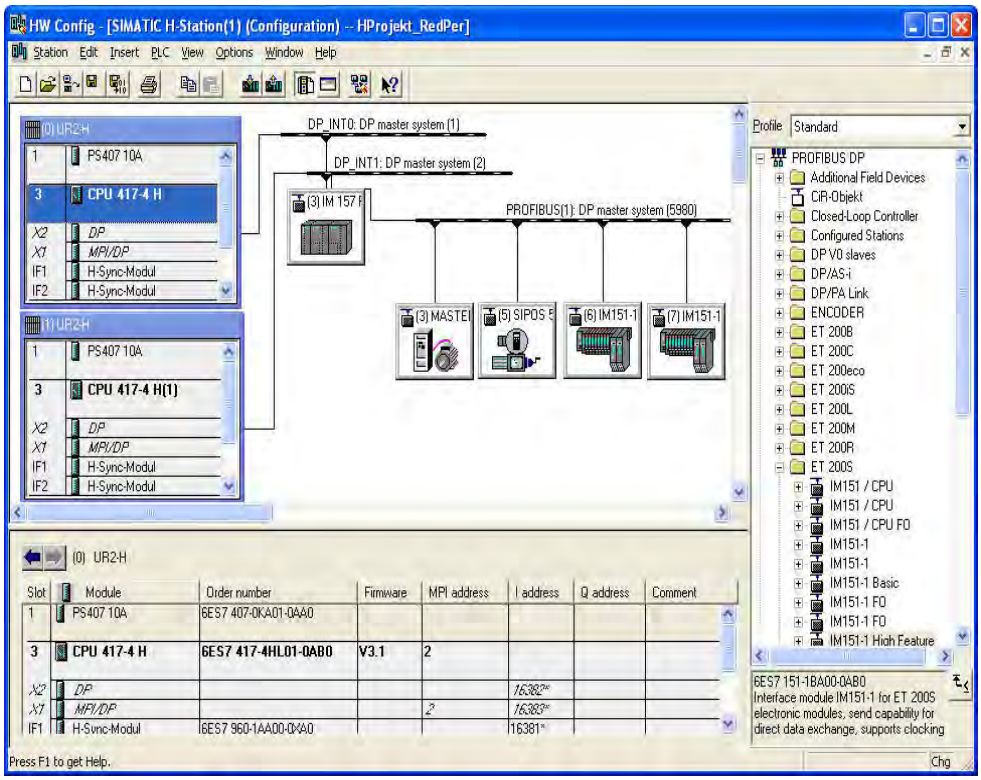
کار این ماژول همانند *DP-PA Link* می‌باشد. ماژول *Y-Link* زمانی استفاده می‌شود که بخواهیم یک شبکه *DP* غیر افزونه (تک کاناله) را به یک شبکه افزونه (دو کاناله) متصل کنیم.

- *Y-Link:* ***6ES7 197-1LB00-0XA0***
- *Bus module BM Y-Link : 6ES7 654-7HY00-0XA0*





شکل ۳-۳۷- اتصال شبکه *DP* تکی به یک شبکه *DP* افزونه از طریق واسط *Y-Link*



شکل ۳-۳۸- نمونه پیکربندی *I/O* از طریق واسط *Y-Link* در سیستم *H*

۳-۹-۶- واسط یا گذرگاه ارتباطی *I/O*

سیگنال‌های *I/O* می‌توانند به دو روش از طریق گذرگاه *PROFIBUS* و *PROFINET* متصل شوند. پیکربندی *PROFIBUS* و *PROFINET* را می‌توان با یکدیگر ترکیب کرد. واسط *PROFIBUS* پیکربندی‌های مختلف زیر را می‌تواند فراهم نماید.

- واسط یک‌طرفه – در دسترس‌پذیری عادی *(Normal availability)*
- یک اتصال تک کانال (*a single-channel Connection*) – در دسترس‌پذیری عادی
- واسط قابل سوییچ (*Switched interface*)–در دسترس‌پذیری افزونه (*Increased Availability*)

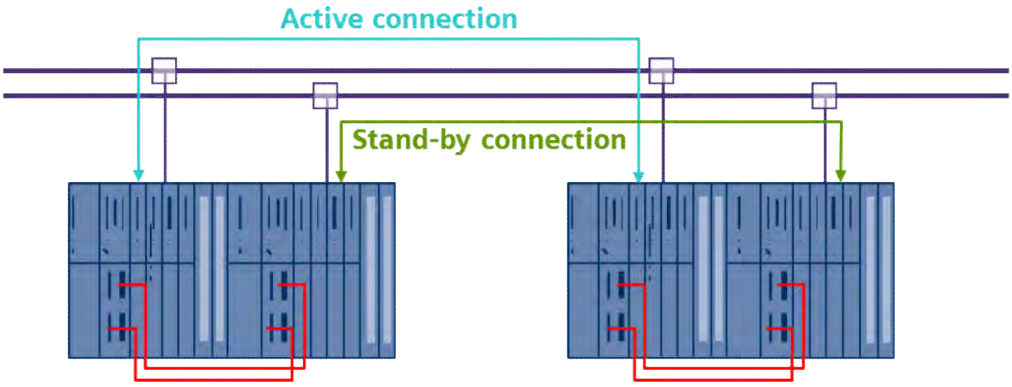
واسط *PROFINET* یک رابط یک طرفه (دسترس‌پذیری عادی) یا واسط سیستم افزونه از طریق حلقه باز (*open ring*) فراهم می‌کند. دسترس‌پذیری در نوع حلقه باز زمانی که دستگاه‌ها از افزونگی سیستم *I/O PROFINET*، مانند *I/O* توزیع‌شده*ET200M* پشتیبانی می‌کنند، افزایش می‌یابد.

۳–۱۰– ارتباطات افزونه (*Redundant communication Principle*)

ارتباطات افزونه بین دو یا چند سیستم از طریق اتصالات افزونه حاصل می‌شود. اتصالات افزونه (*Redundant connections*) را می‌توان از ایستگاه‌های *H* به ایستگاه‌های زیر ایجاد کرد.

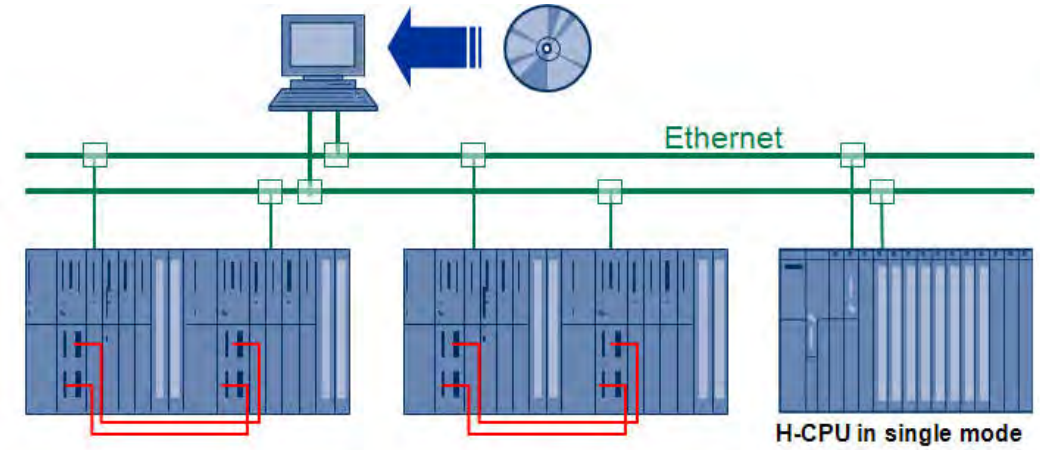
- *Other H stations (one- or two-channel)*
- *HMI PCs (software Redconnect required)*

نوع اول جهت اتصال به دیگر ایستگاه‌های اتوماسیون نوع *H* و اتصال نوع دوم جهت اتصال به کامپیوترهای *HMI* استفاده می‌شود. در اتصال نوع دوم نصب نرم‌افزار *Redconnect* مورد نیاز است.

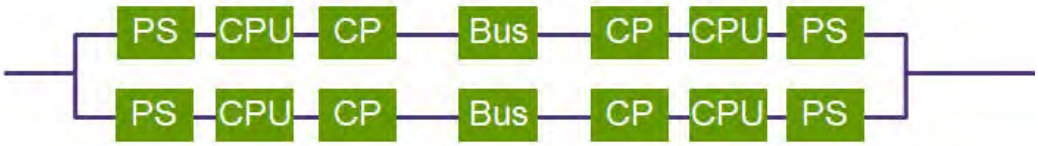


اتصالات افزونه بین دو سیستم افزونه *H*

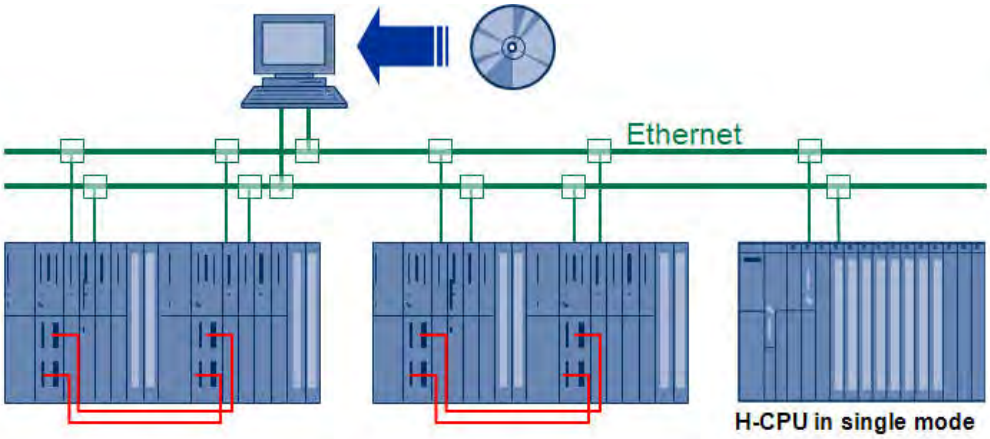
سامانه اتوماسیون *S7-400H* از طریق «پردازنده‌های ارتباطی یا ماژول‌های *CP*» به گذرگاه پلنت متصل می‌شود. برای رسیدن به سطوح بالاتر افزونگی، می‌توان از دو عدد *CP* استفاده کرد.



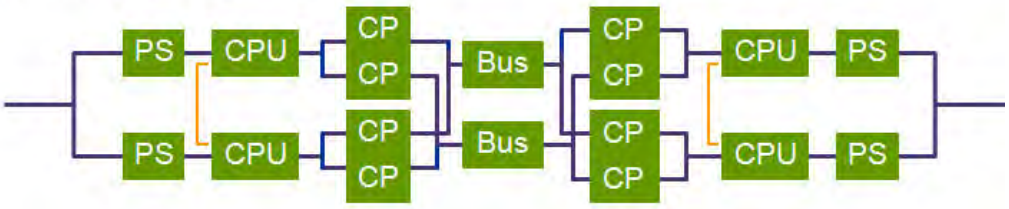
Equivalent circuit diagram:



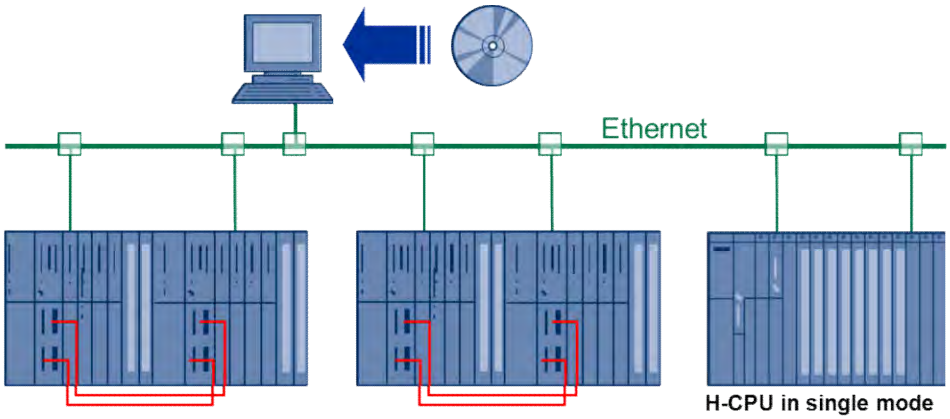
شکل ۳-۳۹- پیکربندی ارتباطات افزونه با باس افزونه



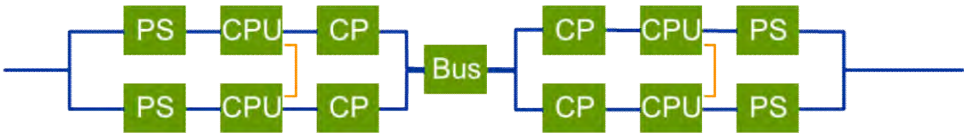
Equivalent circuit diagram:



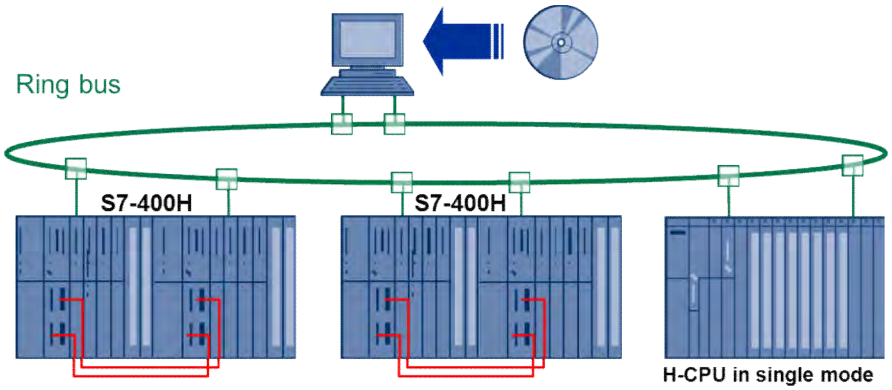
شکل ۳-٤۰- پیکربندی ارتباطات افزونه با باس افزونه و کارت *CP* افزونه



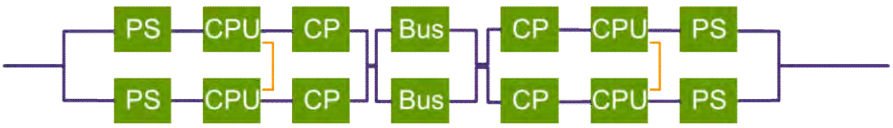
Equivalent circuit diagram:



شکل ۳-۴۱- پیکربندی سیستم *H* با باس تکی



Equivalent circuit diagram:



شکل ۳-۴۲- پیکربندی سیستم *H* با باس *Ring*

۳-۱۰-۱- ارتباطات از طریق اتصالات تحمل‌پذیر خطا (*Fault-tolerant S7 Connections*)

پیکربندی سخت افزار هر دو زیر سیستم که به صورت یکپارچه یک سیستم مقاوم در برابر خطا را ایجاد می‌کنند، بایستی یکسان باشد. این امر به ویژه برای اسلات‌ها (*slots*) درست است. بسته به شبکه‌ای که استفاده می‌شود، ماژول‌های *CP* زیر را می‌توان برای ارتباطات تحمل پذیر خطا استفاده کرد:

- *Industrial Ethernet >> S7: CP 443-1*

- *PROFIBUS >> S7: CP 443-5 Extended (not configured as DP master system)*

نکته : برای اینکه بتوانه از اتصالات *fault-tolerant S7 connections* بین یک سیستم مقاوم در برابر خطا (*fault-tolerant system*) و یک کامپیوتر (*PC*) استفاده کرد، بایستی بسته نرم افزار *S7-REDCONNECT* را بر روی کامپیوتر نصب کرد.

۳-۱۱- قابلیت اطمینان ماژول‌های زیمنس (*Reliability of modules*)

قابلیت اطمینان اجزای سیماتیک در نتیجه اقدامات تضمین کیفیت گسترده در توسعه و تولید محصولات سیماتیک، بسیار بالا است. فاکتورهای ارزیابی قابلیت اطمینان و دسترس‌پذیری بالا (*High Availability & Reliability*) عبارت‌انداز:

- *Mean Time Between Failure (MTBF)*
- *Mean Time To Repair (MTTR) that depends on:*
  - *Being Hot Plug*
  - *Fault Diagnostic Ability*
  - *Maintenance engineer reaction*

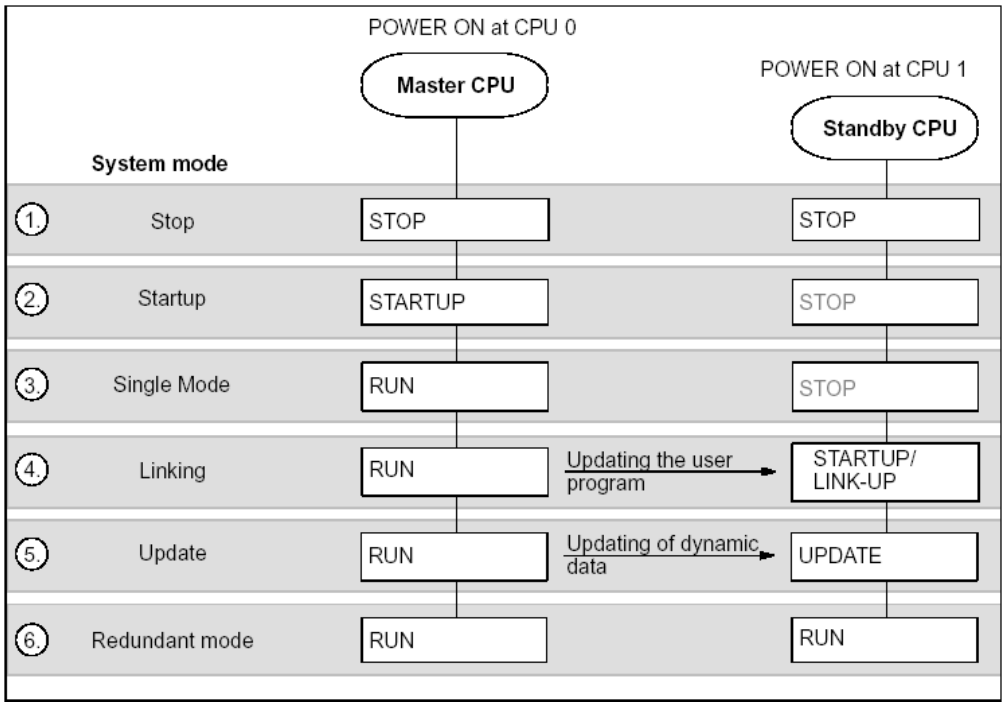
مقادیر متوسط زیر به قابلیت اطمینان ماژول سیماتیک اعمال می‌شود:

- *MTBF of a central processing unit: 15 years*
- *MTBF of an I/O module: 50 years*

۳-۱۲- نصب و راه‌اندازی یک سیستم *S7400H*

۱- پس از نصب سخت‌افزارهای سیستم *S7400H* برای راه‌اندازی آن ابتدا منبع تغذیه واقع در رک ۰ و رک ۱ را روشن می‌کنیم. چراغ *LED* (*DC 5V* و *DC 24V*) مربوط به *Power* به رنگ سبز روشن می‌گردند. در این حالت چراغ *Stop* روی *CPU*ها به رنگ زرد و چشمک زن می‌باشد. لذا تا زمانی که چراغ زرد *Stop* به صورت چشمک زن بوده و قطع نشده است. نمی‌توان با *CPU*ها ارتباط داشت و آنها را برنامه‌ریزی کرد. معمولاً مدت زمان چشمک زن چراغ زرد در حدود ۱۰ الی ۱۵ دقیقه می‌باشد. در این حالت *CPU* در مد اجرای «خود آزمون راه‌اندازی» (*Self Test*) می‌باشد و اصطلاحاً *CPU* تمام قسمت‌های مختلف را چک می‌کند. این حالت همواره برای اولین بار که سیستم *H* مونتاژ و روشن می‌شود، اتفاق می‌افتد.

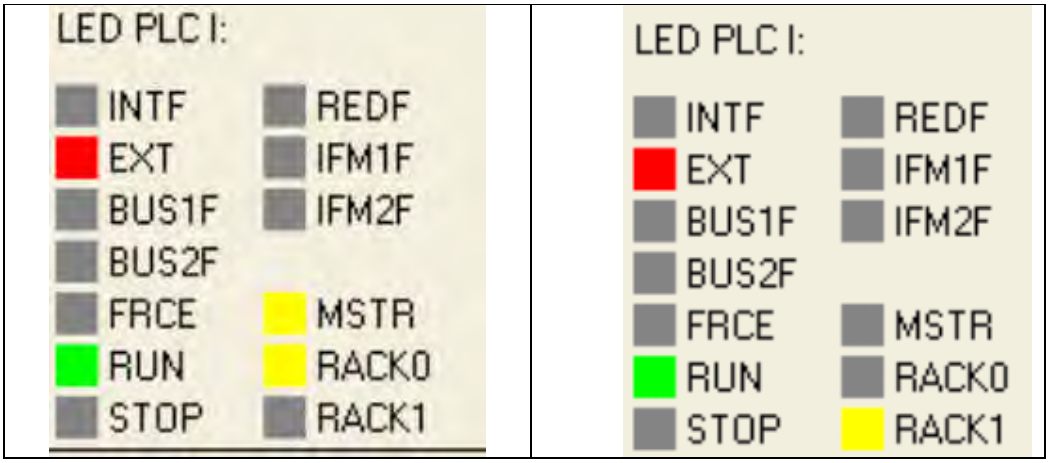
۲- بعد از این که سیستم یک بار تست شد و به عبارت دیگر عمل «خود آزمون» را انجام داد. در دفعات بعد که برق قطع و وصل شود، اگر باتری روی ماژول تغذیه *PS* موجود بوده و سالم باشد، عمل «خود آزمون» کوتاه خواهد بود. ولی اگر باتری نباشد و یا سویچ مربوطه در وضعیت خاموش باشد، با قطع و وصل شدن برق عمل «خود آزمون دوباره از نو شروع می‌شود. به این خاطر باتری نقش مهمی را در سیستم‌های *S7400H* بازی می‌کند.



شکل ۳-۴۳- راه‌اندازی سیستم *S7 400H*

همچنین اگر باتری وجود داشته باشد و عمل «خود آزمون» یک بار انجام شده باشد، ولی اگر ماژول *CPU* از رک جدا شده و دوباره نصب شود، عمل «خود آزمون» از نو تکرار خواهد شد.

۱- بعد از اتمام عمل «خود آزمون» چراغ *Stop* دیگر چشمک نمی‌زند و به رنگ زرد ثابت روشن می‌شود. برخی از چراغ‌ها به صورت قرمز روشن هستند. توضیحات مربوط به چراغ‌های سیستم *S7 400H* به شرح جدول زیر می‌باشد.



شکل ۳-۴۴- چراغ‌های ماژول‌های *CPU* در سیستم *S7 400H*

جدول ۳-۲- توضیحات چراغ‌های روی ماژول منبع تغذیه (*PS*) در *S7 400H*

توضیحات چراغ‌های روی ماژول منبع تغذیه (*PS*)	***LED***

INTF	*Internal faul* در شرایط نرمال نباید روشن باشد. به عنوان مثال اگر منبع تغذیه سالم باشد و یا مشکلی در داخل *CPU* نباشد، خاموش می‌باشد.
BAF *BATTF*	چراغ‌های مربوط به باتری هستند. اگر باتری‌هادر سوکت مربوطه نباشند روشن می‌شوند.
DC 5V *DC 24V*	چراغ روشن بودن تغذیه بوده و در صورت برقراری و سالم بودن با رنگ سبز روشن می‌شوند.

جدول ۳-۳- توضیحات چراغ‌های روی ماژول *CPU* در *S7 400H*

LED	**توضیحات چراغ‌های روی *CPU***
INTF	*Internal faul* در صورتی که در برنامه *CPU* یا پردازش آن مشکلی باشد، روشن می‌شود.
EXTF *External Fault*	در صورت اشکال در ماژول‌های *I/O* روی رک مرکزی یا *RIO* این چراغ روشن می‌شود. به عنوان مثال اگر پیکربندی غلطی برای ماژول *I/O* روی ایستگاه *ET200M* برنامه‌ریزی شود.
BUS1F *Bus Fault*	در صورت اشکال در شبکه پروفیباس این چراغ روشن می‌شود.
IFM1F *IFM2F*	*Interface Module 1 &2* → *SYNC Modules* اگر ارتباط فیبر نوری ماژول سینک بالایی مشکل داشته باشد (قطع باشد)، چراغ *IFM1F* روی هر دو *CPU* روشن می‌شود. اگر ارتباط فیبرنوری ماژول سینک پایینی مشکل داشته باشد (قطع باشد)، چراغ *IFM2F* روی هر دو *CPU* روشن می‌شود.
FRCE	در صورتی که یکی از کانال‌های *I/O* به یک مقداری *Force* شده باشد این چراغ روشن می‌شود.
REDF	در صورت بروز اشکال در ارتباط افزونه بین دو *CPU* از طریق فیبر نوری، (کابل فیبر نوری قطع شود) و یا وقوع اشکال در پروفیباس مربوط به ایستگاه‌های *ET200M*، این چراغ روشن می‌شود

۳-۱۲-۱- مانیتورینگ سگنال‌های سیستم *H*

در سیستم‌های *H* در کنار هر *CPU* یک کارت اترنت وجود دارد و به روش معمولی نمی‌توان عمل مانیتورینگ را از طریق اترنت انجام داد و همواره با مشکلاتی همراه است. لذا باید به یک روش خاصی این کار را انجام داد.

۱- اگر از طریق *MAC address* به کارت اترنت *CPU Master* وصل شویم، در صورت خاموش شدن کارت اترنت *Standby*، اختلالی در ارتباط بین مانیتورینگ و *PLC* رخ نمی‌دهد. ولی اگر کارت اترنت واحد *Master* خاموش شود، وقتی سیستم به مد *Solo* (*Single*) می‌رود، اون سیستمی که کارت اترنت آن را در *Wincc* معرفی کردیم، خاموش بوده و مانیتورینگ از دست می‌رود.

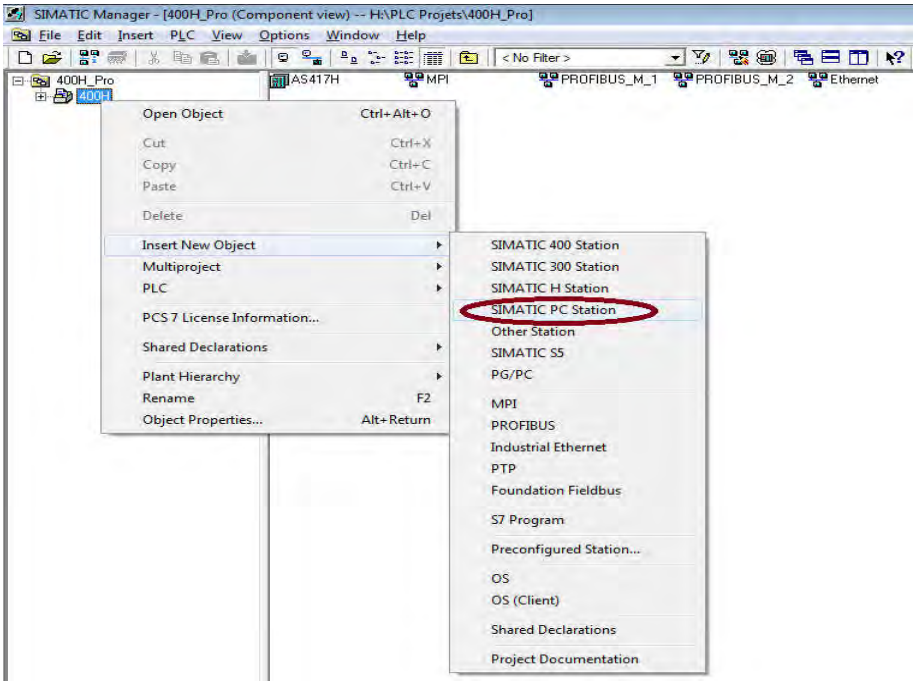
کارت اترنت زیمنس *CP1613* ، قابلیت مانیتورینگ را با یک پیکربندی خاص در سیستم *H* فراهم می‌کند.

ایجاد *Connection* موسوم به *Named Connection*

در این روش همانند سیستم *H* ، تنظیمات خود *PC* را در محیط *HwConfig* را انجام می‌دهیم از کاتالوگ کارت *CP1613* را انتخاب و در اسلات قرار می‌دهیم.

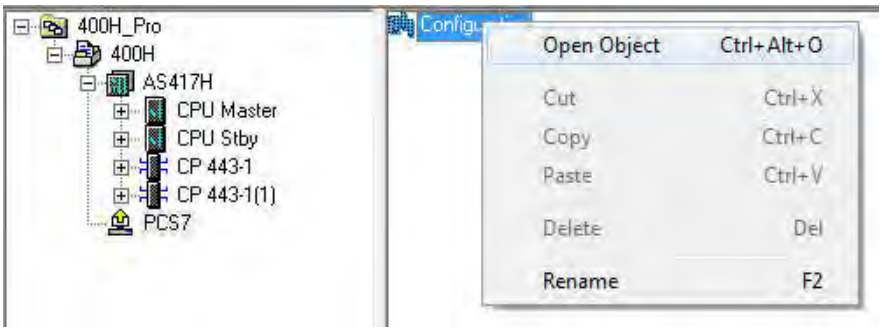
از طریق نرم‌افزار *SIMATIC Net* می‌توان مشخصات کارت *CP1613* را استخراج کرد.

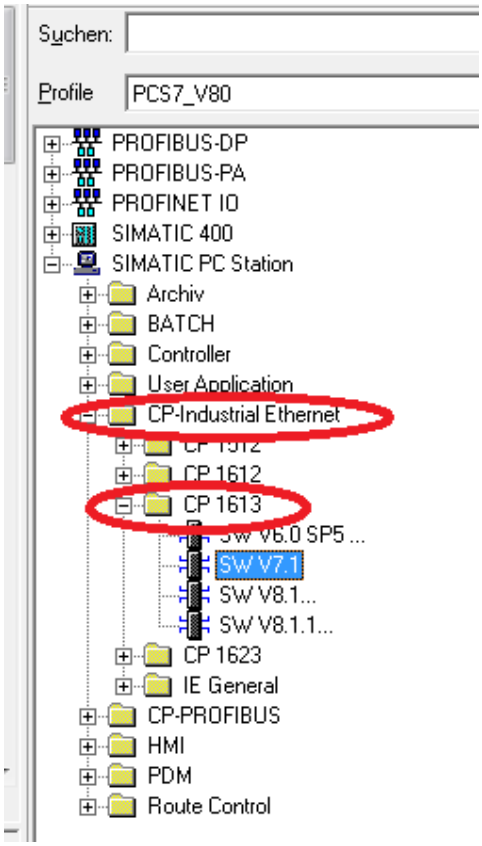
از بخش *Version* باید نسخه مربوطه را استخراج کرد.



شکل ۳-۴۵- ایجاد یک *PC Station* برای سیستم مانیتورینگ

اسم *PC Station* را به اسم کامپیوتر (*PCS7*) تغییر می‌دهیم.





شکل ۳-٤٦-

بایستی یک *Wincc Application* هم معرفی کنیم. برای این کار

پس از اتمام کار بایستی عمل *Save & Compile* انجام داد. در *Wincc* یک پروژ ایجاد می‌گردد.

برنامه *NetPro* را باز می‌کنیم.

فایل *XDB* را باید از مسیر پروژه *Import* کنیم.

در سیستم‌های *H* کارت‌های *I/O* به ندرت در کنار سیستم *PLC* نصب می‌شوند. بیشتر به بر روی یک رک *RIO* نوع *Active Bus* (با استفاده از *Bus Module* یا *Bus unit*) نصب می‌شوند.

اتصال بین این ماژول‌های از طریق دو ماژول *IM 153* ، کابل و کانکتور ۹ پین پروفیباس صورت می‌گیرد.

بین دو قسمت (۹ اسلات در هر قسمت) رک *UR2H* هیچ ارتباط الکتریکی وجود ندارد. باس‌های موجود در هر دو طرف جدا از هم هستند.

منابع تغذیه را در اسلات‌های ۱ هر قسمت نصب می‌کنیم

بعد از نصب منبع تغذیه، ماژول *CPU* می‌تواند در هر اسلاتی نصب شود. در پشت ماژول *CPU* یک *Dip Switch* است. که در کنار آن بسته به حالت قرار گیری سویچ در بالا یا پایین عبارت‌های *Rack 0* و *Rack 1* قرار گرفته است. در یک سمت یکی از سویچ‌ها بر روی *Rack 0* قرار گیرد و در سمت

دیگر بایستی بر روی *Rack 1* قرار گیرد. این تنظیم فرقی نمی‌کند که در کدام طرف روی یک یا صفر قرار گیرد. لازم به یادآوری است که هر دو *CPU* بایستی در هر دو طرف بر روی یک اسلات با شماره یکسان نصب شوند.

بر روی *Bus Module* مربوط به کارت‌های واسط *IM 153* عبارت *IM* در دو طرف آن نوشته شده است. بر روی هر *Bus Module* دو عدد ماژول *IM 153* قابل نصب می‌باشد.

برای نصب کارت‌های *I/O* معمولی از یک *Bus Module* دیگر استفاده می‌شود. بر روی این *Bus Module* نیز دو عدد کارت *I/O*نصب می‌شود

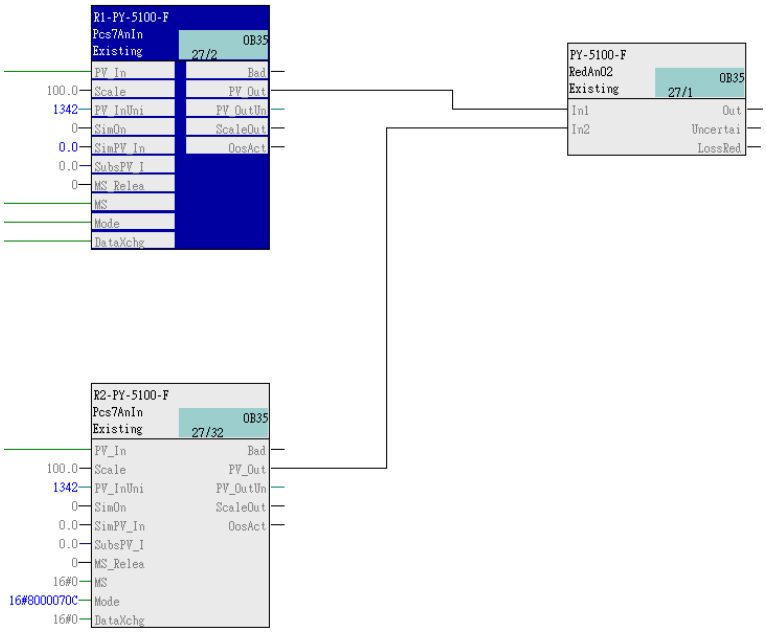
برای نصب کارت‌های *I/O* نوع *Fail safe* از یک *Bus Module* دیگر که عرض آن بیشتر از کارت‌های معمولی است،استفاده می‌شود. بر روی این *Bus Module* یک عدد کارت *I/O* نوع *Fail safe*نصب می‌شود.

۳-۱۳- توصیف فانکشن‌بلاک‌ها

۳-۱۳-۱- فانکشن بلاک *RedAn02*

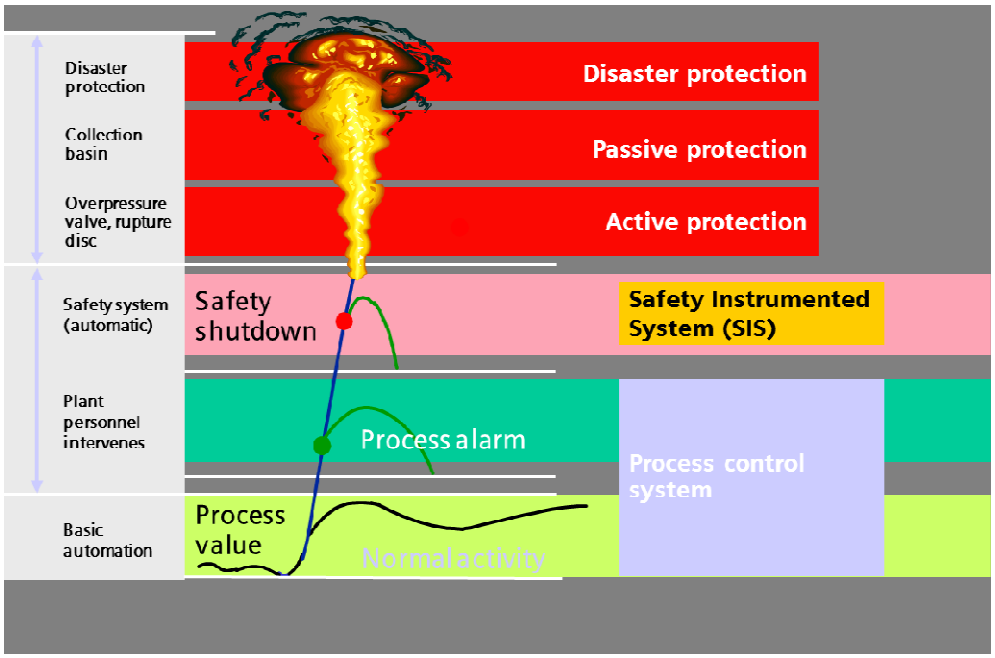
این فانکشن بلاک برای عملیات انتخاب مقدار سیگنال آنالوگ بین دو ورودی افزونه استفاده می‌شود.

- *1 out of 2 selection for redundant analog values*



شکل ۳-۴۷- پیاده‌سازی درایور کانال ریداندانت در محیط *CFC*

فصل – ۴ – ساختار و معماری سیستم‌های FH

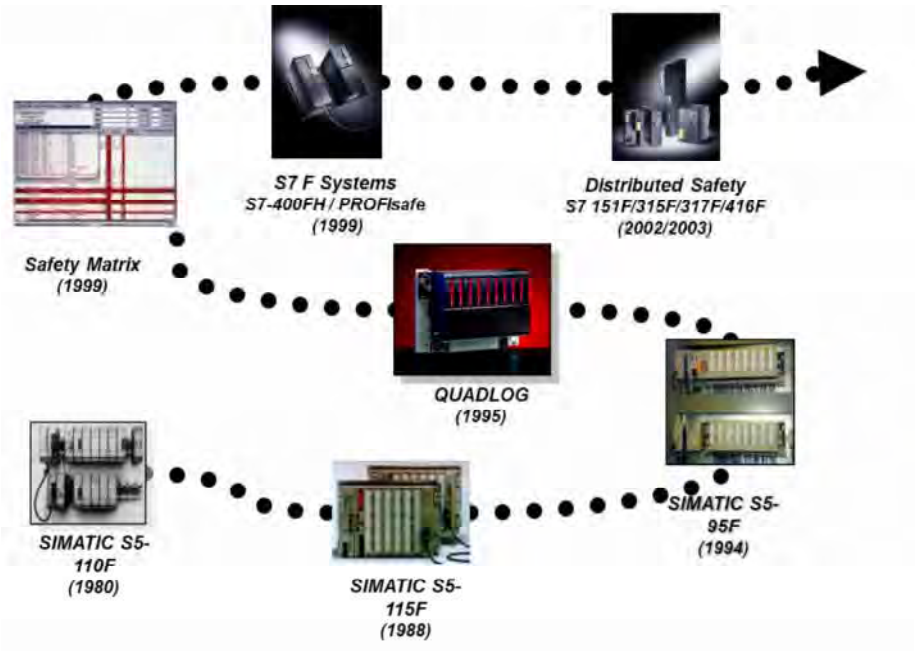


جایگاه سیستم *SIS* در لایه‌های حفاظت

۴–۱– مقدمه

۴–۱–۱– تاریخچه سیستم‌های ایمنی زیمنس (*History of Siemens Safety Systems*)

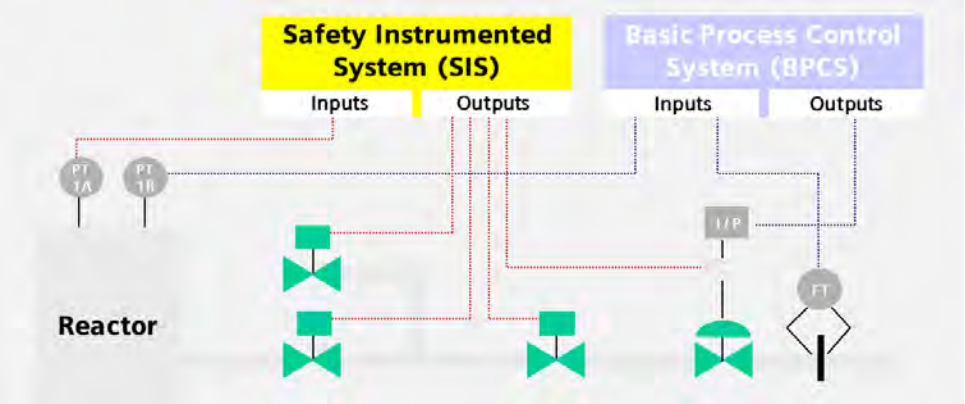
شکل زیر تاریخچه سیستم‌های کنترل ایمن در نقص زیمنس را به تصویر کشیده است.



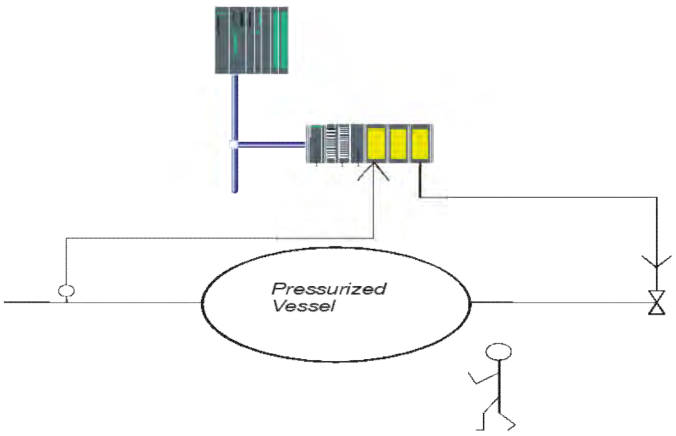
شکل ٤-١- تاریخچه سیستم‌های کنترل ایمن در نقص زیمنس

۴-۱-۲- تفاوت بین سیستم *SIS* و سیستم کنترل پایه *BPCS*

به طور معمول در پلنت‌هایی همچون پتروشیمی‌که نیاز به ایمنی همزمان با کنترل حلقه‌های آنالوگ نیاز است از یک سیستم *SIS* برای کنترل حلقه‌های کنترل‌کننده ایمنی سیستم و از یک سیستم کنترل با عنوان *DCS* یا *BPCS* برای کنترل حلقه‌های آنالوگ استفاده می‌شود.



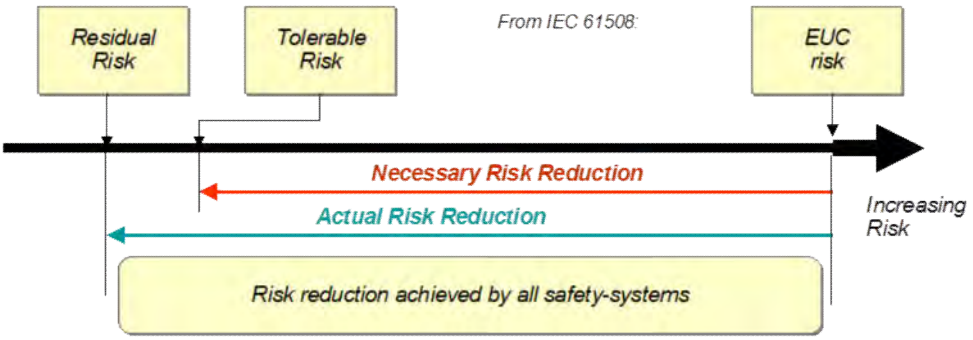
شکل ٤-٢- حلقه‌های کنترل *SIS* و *BPCS* برای پلنت راکتور



شکل ٤-٣- نمونه کاربرد سیستم *F*

۴-۱-۳- هدف از سیستم‌های *SIS*

هدف از بکارگیری سیستم‌های *SIS* کاهش ریسک از طریق سیستم *SIS* می‌باشد.



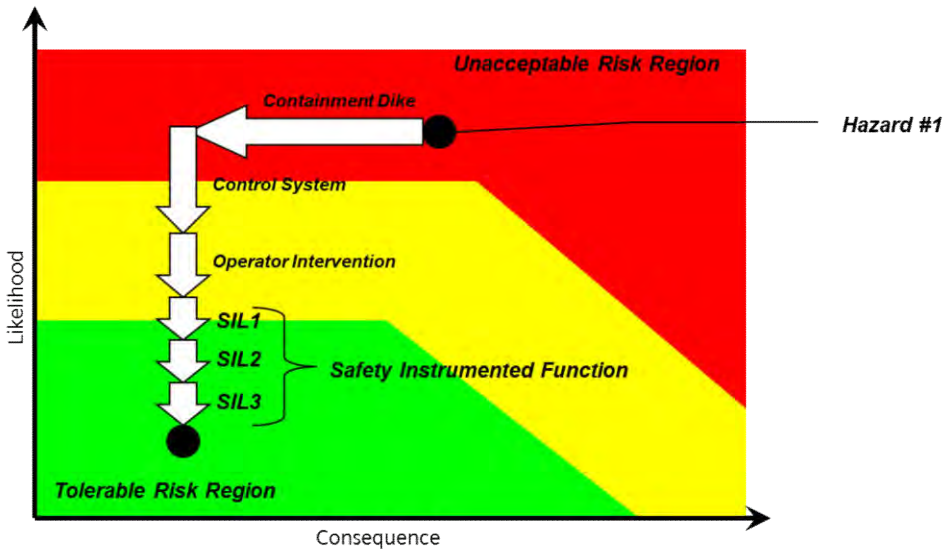
۴-۱-۴- ریسک چیست و چه کسی تصمیم می‌گیرد که چه ریسکی قابل قبول است؟

نمونه‌هایی از شکل‌های مختلف احتمال مرگ عبارت‌انداز:

- *Road accident → 100cpm 1.0x10-4/yr 1 av 100 (at life 100 years)*
- *Car accident 150cpm 1.5x10-4/yr 1,5 av 100*
- *Accident at work 10cpm 1.0x10-5/yr 1 av 1000*
- *Falling Aircraft 0.02 cpm 2.0x10-8/yr 2 av 1000 000*
- *Lightning strike 0.1cpm 1.0x10-7/yr 1 av 100 000*
- *Insect/Snake bite 0.1cpm 1.0x10-7/yr 1 av 100 000*
- *Smoking 20 per day 5000 cpm 5.0x10-3/yr 1 av 2*

که در آن *cpm* عبارت است از احتمال وقوع در بین یک میلیون نفر در سال:

cpm = chances per million of the population (per year)



برخی از کاربردهای سیستم‌های *SIS* عبارت‌انداز:

- *ESD, Emergency Shutdown*
- *F&G, Fire & Gas Detection, Fire-fighting*
- *Process Shutdown*
- *Fire-pump Logic*
- *Ballast Control*
- *Blow-down*
- *Riser release / Anchor Release*
- *Fire Dampers, Active Smoke Control*

- *HIPPS, High Integrity Pressure Protection System*

۴-۱-۵- استانداردهای مهم مرتبط با ایمنی

- *IEC 61508 (Application-independent standard*)*
  Function safety of electrical / electronic / programmable electronic safety-related systems"
- *DIN V VDE 0801 Fundamentals for computers in systems with safety tasks.*
- *VDI/VDE 2180 Ensuring the safety of process plants using instruentation and control*
- *prEN50156 Standard for burner controls Electrical equipment for burner systems…" (en)*
- *EN 298 Standard for burner controls "Automatic gas burner control system…"*
- *EN 954 Standard for safety of machinery*
  Safety related parts of control systems"
- *DIN V 19250 "Fundamental safety considerations for protection facilities in measuring and control technology"*

۴-۲- سخت‌افزار سازگار با سطوح *SIL 1 , SIL2, SIL3*

در شکل زیر اطلاعات لازم برای درک سطوح *SIL 1 , SIL2, SIL3* نمایش داده شده است. هر یک از سطوح *SIL* وابسته به سه مقدار*HFT* (*hardware fault tolerence*)،مقدار *DC* (*diagnostic coverage*) و *SFF* (*safe failure fraction*) می‌باشد. اجزای الکترومکانیکی به عنوان مثال سوئیچ‌های موقعیت یا کنتاکتورها دارای *HFT= 0* هستند. اگر از دو جزء استفاده می‌کنید، لذا *HFT=1* خواهد بود. با توجه *SIL2* یا *SIL3* در ترکیب با قطعات الکترو مکانیکی باید از دو جزء استفاده کنید.

قطعات الکترونیکی به طور معمول یک مقدار *SIL CL* ثابت دارند. این مقدار از تولید کننده دریافت می‌شود.

Table 5 – Architectural constraints on subsystems: maximum SIL that can be claimed for a SRCF using this subsystem

Safe failure fraction	Hardware fault tolerance (see Note 1)		
	0	1	2
< 60 %	Not allowed (for exceptions see Note 3)	SIL1	SIL2
60 % – < 90 %	SIL1	SIL2	SIL3
90 % – < 99 %	SIL2	SIL3	SIL3 (see Note 2)
≥ 99 %	SIL3	SIL3 (see Note 2)	SIL3 (see Note 2)

NOTE 1 A hardware fault tolerance of *N* means that *N*+1 faults could cause a loss of the safety-related control function.
NOTE 2 A SIL 4 claim limit is not considered in this standard. For SIL 4 see IEC 61508-1.
NOTE 3 See 6.7.6.4 or for subsystems where fault exclusions have been applied to faults that could lead to a dangerous failure, see 6.7.7.

۴-۲-۱- انتخاب یک *CPU* برای یک پلنت فرآیندی

در انتخاب یک *CPU* برای یک پلنت، معیارهای بسیاری وجود دارد که باید در نظر گرفته شود. در ابتدا باید مشخص کنید که *CPU* را برای چه نوع پلنت فرآیندی انتخاب می‌کنید. جزئیات زیر را می‌توان برای انتخاب یک *CPU* برای یک پلنت فرآیندی در نظر گرفت:

۱- تعداد ورودی/خروجی - به طور معمول پلنت‌های فرایندی دارای تعداد *I/O* از ۵۰۰ تا ۱۰۰۰۰۰ هستند.

۲- تعداد حلقه‌های بسته (*Closed loops*). اگر تعداد حلقه‌های کنترل آنالوگ بیشتر باشد. در آن صورت به سیکل اسکن سریع‌تر نیاز هست.

۳- تعداد حلقه‌های *PID* - بیشتر قدرت پردازش در حلقه‌های بسته صرف می‌شود.

٤- نوع پلنت (نوع فرآیند). اگر یک فرایند از نوع حیاتی است. پس نیاز است که یک *CPU* با قابلیت دسترس‌پذیری بالا (*high availability*) را انتخاب کنید. به عنوان مثال *CPU 417H*

۵- نوع سیستم ابزار دقیق – روش اتصال سنسورها به سیستم کنترل چیست.

- *4-20mA*
- *HART*
- *Profibus*
- *MODBUS*
- *Foundation Fieldbus*

٦- نوع معماری سیستم کنترل (توزیع شده یا کنترل متمرکز)

۷- همچنین در برآورد واقع بینانه از عملکرد مورد نیاز برای واسط *HM* دقت خاصی باید شود.

۸- آیا *CPU* برای کنترل پایه فرایند (*DCS*) است یا برای کاربردهای مربوط به *SIL* مورد نیاز است

اینها برخی از مهم‌ترین ملاحظاتی که در انتخاب یک *CPU* باید مدنظر باشد.

۴-۳- سیستم‌های اتوماسیون خرابی امن *FH* (*Fail Safe Automation System*)

برای طراحی سیستم کنترل پلنت‌هایی که نیاز به قابلیت اطمینان و ایمنی بالایی دارند. به عبارت دیگر در سیستم‌هایی که هر دو قابلیت ایمنی (*Safety*) حداکثر و در دسترس‌پذیری بالا (*Availablity*) مورد نیاز است، سیستم *S7-400FH* یکی از گزینه‌ها می‌باشد. در چنین فرایندهایی به هنگام وقوع خطا سیستم کنترل باید فرآیند را به شرایط ایمن از پیش تعریف شده ببرد تا خسارتی متوجه افراد، محیط زیست، دستگاهها و فرآیند نگردد.این *PLC*ها هر دو مشخصات سری *H* و *F* را دارا می‌باشند و علاوه بر نرم‌افزار استاندارد *Step7* به دو بسته *F-System* و *H-System* نیز نیاز است.

سیستم‌های *F* فرآیندهای صنعتی را کنترل و یا از آنها محافظت می‌کنند. به طوری که یک توقف فوری، جان پرسنل یا محیط زیست را به خطر نمی‌اندازد.

۴-۳-۱- اصول پشت پرده عملکرد ایمنی(*Principle behind the Safety Function*)

- رفتار خرابی امن با استفاده از توابع ایمنی در درجه اول در نرم افزار حاصل می‌شود.
- هنگامی‌که یک رویداد خطرناک رخ می‌دهد، توابع ایمنی(*Safety functions*) سیستم را به حالت امن هدایت می‌کنند و یا اینکه سیستم را در یک حالت امن نگه می‌دارند.
- در صورت بروز یک فالت، سیستم*F* دیگر نمی‌تواند تابع ایمنی تعریف شده را اجرا کند. لذا تابع مربوط به عکس العمل در برابر فالت را اجرا می‌کند. به عنوان مثال، خروجی‌های مرتبط خاموش (*switched off*) شده و برنامه ایمنی یا بخش‌هایی از برنامه ایمنی، در صورت لزوم غیر فعال می‌شود.

مثالی از توابع ایمنی کاربر و توابع واکنش در برابر فالت عبارت‌انداز:

- در صورت وقوع یک رخداد فشار بیش از حد، سیستم‌های *F*، یک ولو را باز می‌کنند.
- در صورت وقوع یک فالت خطرناک در *F-CPU*، تمام خروجی‌ها، خاموش می‌شود. به این قابلیت تابع عکس العمل فالت یا *F shutdown* گفته می‌شود.

به طور معمول قابلیت‌های توابع ایمنی در اجزای زیر گنجانیده می‌شود:

- در برنامه لاجیک ریخته شده در *CPU* از طریق بلاک‌های کتابخانه *F* (*F Librrary*)
- در ماژول‌های ورودی/ خروجی *Fail-Safe* از طریق تنظیم پارامترهای ماژول *F*

شکل ٤-٤- پیاده‌سازی توابع ایمنی

۴-۳-۲- پیکربندی‌های پایه برای تامین ایمنی و دردسترس‌پذیری (*Basic Configuration Variants*)

شرکت زیمنس سیستم‌های مختلفی برای پیکربندی‌های مستقل و ترکیبی از *Fail safe* و *fault-tolerant* به بازار عرضه کرده است.

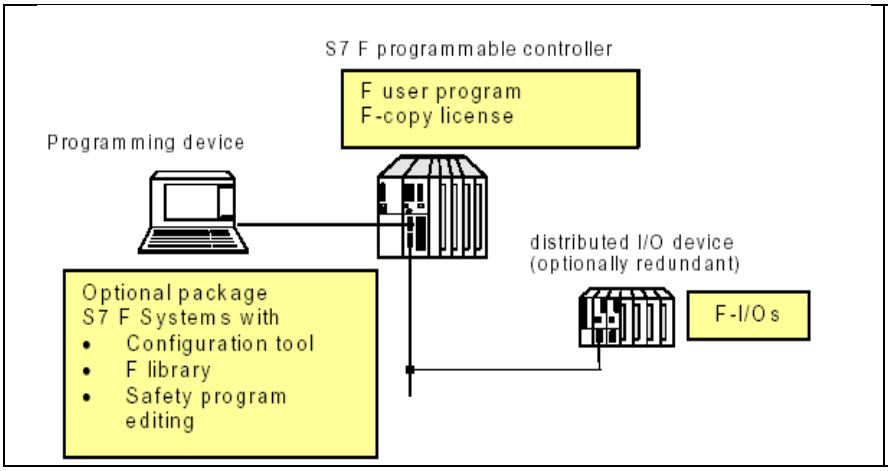
- سیستم‌های مستقل خرابی امن (*Fail-safe S7 F System*)
- سیستم‌های ترکیبی از نوع *Fail-safe* و *fault-tolerant* که به سیستم‌های *FH* معروفند.

- *Safety : Fail-safe S7 F System*
  - *CPU 315-F*
  - *CPU 317-F*
  - *CPU 416-F*
- *Safety & Availability: Fail-safe, fault-tolerant S7 FH System*
  - *CPU 412-F/H*
  - *CPU 414-F/H*

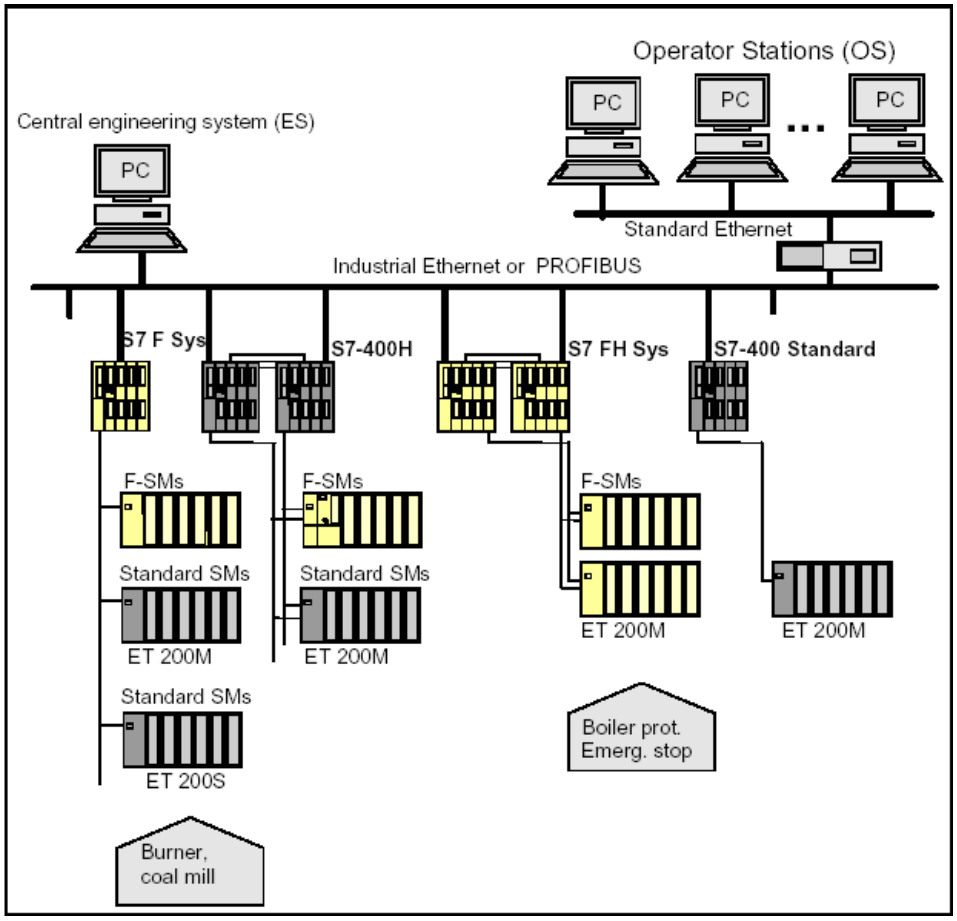
- *CPU 417-F/H*

سخت‌افزار یک سیستم اتوماسیون امن در خرابی *S7 F* متشکل از حداقل اجزای زیر است:

- یک ماژول *CPU* با قابلیت اجرای برنامه *F*، مانند *CPU 417-4 H*، که می‌تواند یک برنامه لاجیک از نوع خرابی امن (*F*) را اجرا کند.
- یک یا چند ورودی/خروجی از نوع خرابی امن (*F-I/Os*) در یک رک *I/O* توزیع شده (با قابلیت افزونگی اختیاری)



شکل ٤-٥- مولفه‌های یک سیستم *FH*

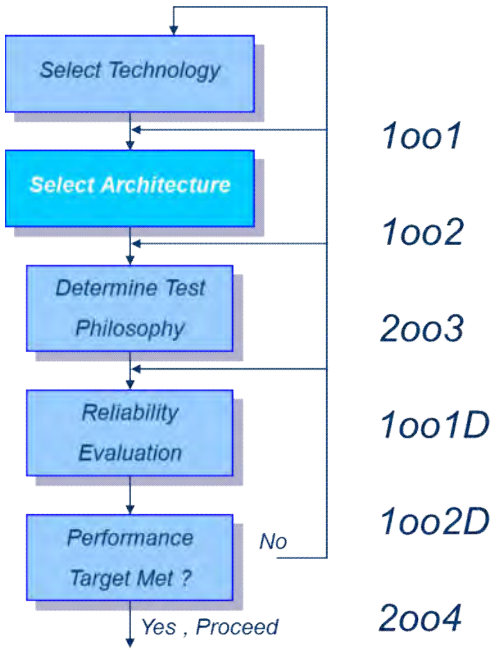


شکل ٤-٦- مثالی از پیکربندی سیستم *FH* زیمنس

۴-۴- معماری‌های سیستم *Fail Safe*

Safety Integrity Level	High demand or continuous mode of operation (probability of a dangerous failure per hour)	Low demand mode of operation (average probability of failure to perform its design function on demand)
4	≥ 10^{-9} to < 10^{-8}	≥ 10^{-5} to < 10^{-4}
3	≥ 10^{-8} to < 10^{-7}	≥ 10^{-4} to < 10^{-3}
2	≥ 10^{-7} to < 10^{-6}	≥ 10^{-3} to < 10^{-2}
1	≥ 10^{-6} to < 10^{-5}	≥ 10^{-2} to < 10^{-1}

۴-۴-۱- افزونگی موردنیاز



۴-۴-۲- معماری سیستم تک سنسور یا ارزیابی 1oo1 (1 Out of 1)

سیستم تک کانال

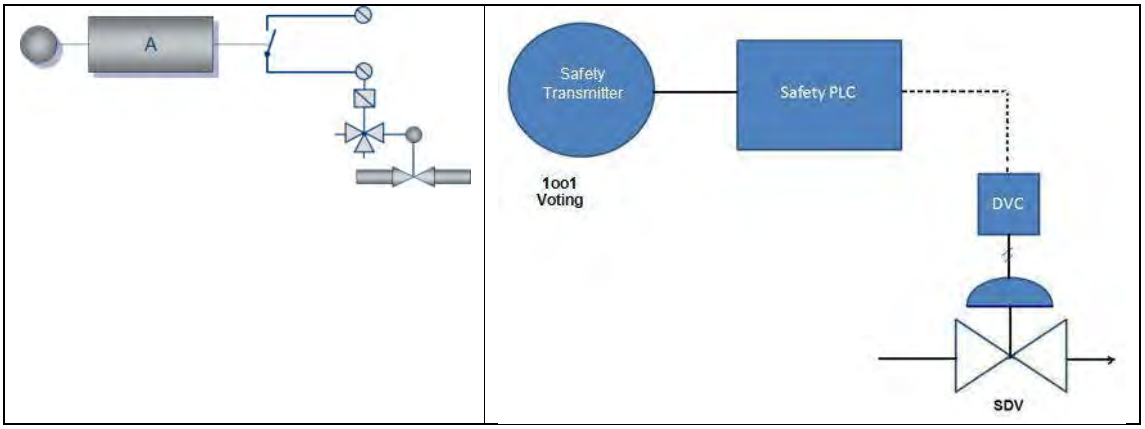
در این نوع ارزیابی تنها یک سنسور از طریق کانال سیگنال به ماژول سیگنال F متصل می‌شود.

این معماری متشکل از یک سنسور واحد می‌باشد، که در آن هر نقص یا فالت خطرناک منجر به شکست تابع ایمنی(*safety function*) در مواقع نیاز می‌شود. به عنوان مثال عملکرد ایمنی (*safety function*) در شکل زیر باز کردن یا بستن یک شیر می‌باشد.

SF یا *SIF* (*Safety Function or Safety instrumented Function*) همان حلقه ایمن ابزاردقیق می‌باشد

شکست امن (*Safe failure*): در حالتی که سیستم با شکست مواجه و ایمنی سیستم و افراد تامین می‌شود، یعنی سیستم پس از شکست به یک وضعیت امن می‌رود، به این معنی است که با عملکرد برنامه ایمنی (*SF*) کنتاکت‌های رله باز شده و برق سیستم (تابع ایمنی که در اینجا همان ولو می‌باشد) را قطع می‌کند. یعنی با وقوع شکست در سیستم، سیستم به یک حالت امن سوق داده می‌شود.

شکست خطرناک (*Dangerous failure*): در این نوع شکست با فرمان از طرف کنترل‌کننده، کنتاکت‌های رله جوش خورده و به هنگام نیاز عمل نمی‌کنند.



شکل ٤-٧- معماری سیستم *1 Out of 1 (1oo1)* (*SIF – SIL 2*)

۴-۴-۳- ارزیابی *(1 Out of 2) 1oo2*

در این نوع ارزیابی برای یک سیگنال فرآیند، یک یا دو سنسور به دو ورودی مقابل هم (*opposite inputs*) در یک ماژول سیگنال *F* متصل می‌شود. وضعیت سیگنال ورودی‌ها (هم ارز یا غیر هم‌ارز) به صورت داخلی مقایسه می‌شوند. با این ارزیابی کلاس‌های ایمنی زیر قابل دستیابی می‌باشد.

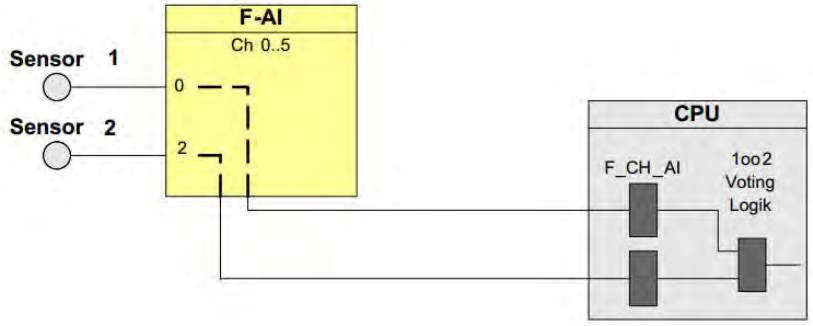
- **سیستم تک کانال- *SIL2*** : در مورد چند کانال، کلاس *SIL3* را می‌توان از طریق سیستم *Voting* در *CPU* **بدست آورد.**
- **دو کانال- *SIL3* : عمل *Voting* در ماژول انجام می‌شود.**

<u>سیستم دو کانال(*Dual Channel system*)</u>

- در این معماری از دو عدد سنسور برای ارزیابی سطوح ایمنی موردنیاز استفاده می‌شود. استفاده از دو سنسور یا طرح ارزیابی *1oo2* به کاربردهایی اشاره می‌کند که برای رسیدن به سطح *SIL* (*Safety Integration Level*) مورد نیاز به دو سنسور نیاز دارد. ارزیابی *1oo2* بدان معنی است که برای صدور فرمان تریپ (*trip*) یا تریگر شدن لاجیک ایمنی (*safety logic*) تنها یکی از دو سنسور نیاز است که *Fail* کند. به عبارت دیگر اگر یکی از دو سنسور خراب شود، مدار تریپ فعال می‌شود. لذا در زمانی که یکی از دو زیر سیستم با شکست مواجه شده است، سیستم دیگر عملیاتی نمی‌باشد.
- همچنین می‌توان گفت که در معماری *1oo2* تنها نیاز است که یک کانال عمل خاموش کردن تجهیز را انجام دهد.
- در این معماری خروجی‌های دو کانال به صورت سری باهم بسته می‌شوند.*(1oo2-1 Out of 2)*
- *2 von 2 (2v2)* : یعنی تنها زمانی که هر دو کانال عملیاتی باشند، سیستم فعال باقی می‌ماند.

برای ارزیابی *1oo2* دو معماری پایه قابل انتخاب، توصیه می‌شود:

- انتخاب ۱: مطابق شکل ۱- هر دو سنسور به یک ماژول *F-AI* سیم‌بندی می‌شود. در این شکل یک سنسور به کانال ۰ و سنسور دیگر به کانال ۲ ماژول *F-AI* سیم‌بندی می‌شود.
- انتخاب ۲: اولین سنسور به یک کارت *F-AI*، و دومی‌سنسور به یک کارت *F-AI* دوم سیم بندی می‌شود. همانطور که در شکل ۲- نشان داده شده. در این شکل یک سنسور به کانال ۰ اولین کارت *F-AI* و سنسور دوم به کانال ۰ از دومین کارت *F-AI* سیم‌بندی می‌شود.

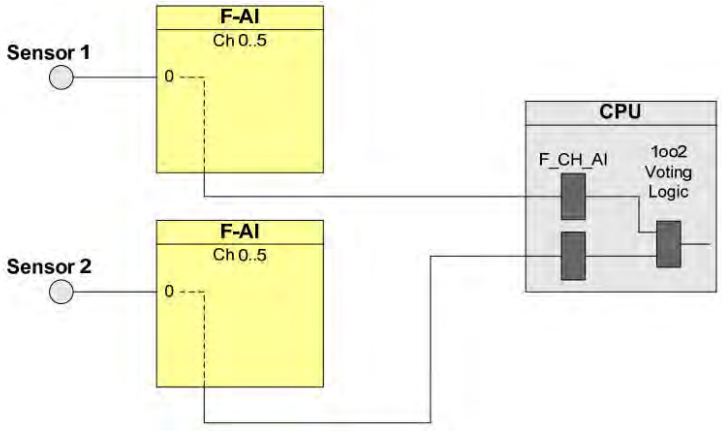


شکل ٤-٨- انتخاب ۱- ارزیابی *1oo2* در برنامه لاجیک با یک ماژول *F-AI*

پیکرندی مطابق شکل انتخاب اول برای رسیدن به *SIL3* مناسب می‌باشد.

اگر یک سنسور یا ماژول *F-AI* فالت دهد، تابع واکنش خطا عملکرد ایمنی (*safety function*) را فراهم می‌کند (از طریق سیستم *failsafe*). جدول زیر زمان فعال شدن تابع واکنش خطا (*error reaction function*) را نشان می‌دهد.

Component has failed ?			Error reaction function has been triggered?
Sensor 1	Sensor 2	F-AI	
no	no	no	no
X	yes	X	yes
yes	X	X	yes
X	X	yes	yes



انتخاب ۲- ارزیابی *1oo2* در برنامه لاجیک با استفاده از دو کارت *F-AI*

پیکربندی انتخاب دوم برای رسیدن به *SIL3* مناسب می‌باشد.

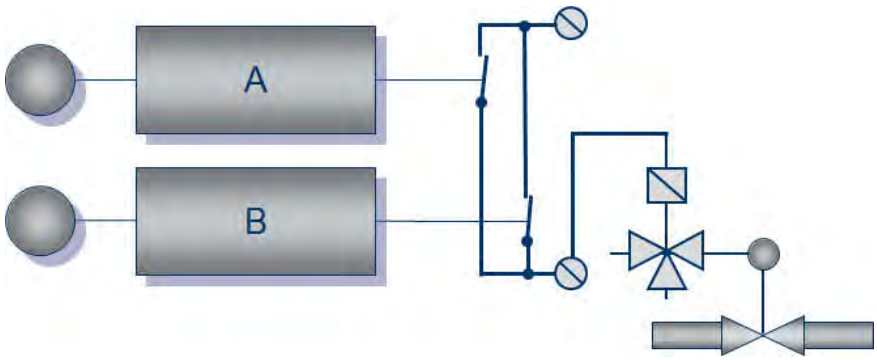
اگر یک سنسور یا یکی از ماژول‌های *F-AI* فالت دهد، تابع واکنش خطا، عملکرد ایمنی (*safety function*) را فراهم می‌کند (از طریق سیستم *failsafe*). جدول زیر زمان فعال شدن تابع واکنش خطا (*error reaction function*) را نشان می‌دهد.

Component has failed ?				Error reaction function has been triggered?
Sensor 1	Sensor 2	F-AI 1	F-AI 2	
no	no	no	no	no
X	X	X	yes	yes
X	X	yes	X	yes
X	yes	X	X	yes
yes	X	X	X	yes

۴-۴-۴- معماری سیستم *(2 Out of 2 (2oo2)*

سیستم دو کانال (*Dual Channel system*)

- خروجی کانال‌ها به صورت موازی سیمی‌بندی می‌شوند. (*2 Out of 2 : 2oo2*)
- برای انجام عمل قطع یا خاموش نمودن تجهیز، انرژی هر دو کانال باید قطع شود.
- هنگامی که هر دو کانال یا زیرسیستم از کار بیافتد، سیستم دیگر کار نمی‌کند *(1v2-1 von 2)*
- در زمانی که تنها یکی از دو زیر سیستم عملیاتی است سیستم فعال باقی می‌ماند

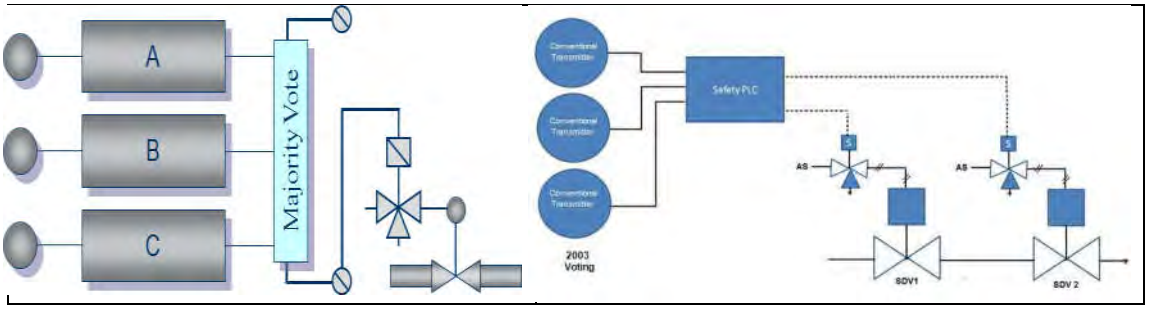


شکل ٤-۹- معماری سیستم *(2 Out of 2 2oo2)*

۴-۴-۱- معماری سیستم *(2oo3-2 Out of 3)*

سیستم سه نسخه (*TMR*)

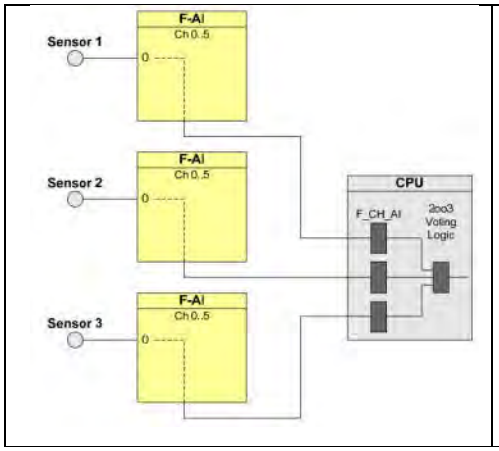
هنگامی‌که دو تا از سه زیرسیستم نقص داشته باشندهٔ سیستم دیگر عملیاتی نمی‌باشد. *(1 von 3)* یعنی سیستم تنها از دست دادن یکی از سه کانال را تحمل می‌کند و اگر دو تا از سه زیرسیستم عمل کند، فعال باقی می‌ماند.



شکل ٤-١٠- معماری سیستم *2 Out of 3*

معماری سه سنسور (یا طرح ارزیابی *2oo3*) به کاربردهایی اشاره می‌کند که برای رسیدن به سطح *SIL* موردنیاز به سه سنسور برای افزایش در دسترس بودن نیاز دارد. ارزیابی *2oo3* بدان معنی است که دو سنسور از سه سنسور تریپ دهد، مدار تریپ فعال می‌شود. اگر حداقل دو تا از سنسورها تریپ دهد. برنامه لاجیک ایمنی تریپ خواهد شد.

در سیستم *F* زیمنس، معماری پایه *2oo3* با ارزیابی در برنامه لاجیک، از سه سنسور و سه *F-AI* استفاده می‌کند. در شکل ٥-١ یک بلوک دیاگرام از این پیکرندی نشان داده شده اشت. در این شکل هر سنسور به کانال ٠ یک ماژول *F-AI* نصب شده در *ET200M*، سیم‌بندی شده است.



Component has failed?						Error reaction function has been triggered?
Sensor 1	Sensor 2	Sensor 3	F-AI 1	F-AI 2	F-AI 3	
X	no	no	X	no	no	no
			no	X	no	
			no	no	X	
no	X	no	X	no	no	
			no	X	no	
			no	no	X	
no	no	X	X	no	no	
			no	X	no	
			no	no	X	
X	yes	yes	X	X	X	yes
yes	X	yes				
yes	yes	X				
X	X	X	X	yes	yes	
			yes	X	yes	
			yes	yes	X	

شکل ٤-١١- پیاده‌سازی معماری ارزیابی *2oo3* در سیستم *F* زیمنس

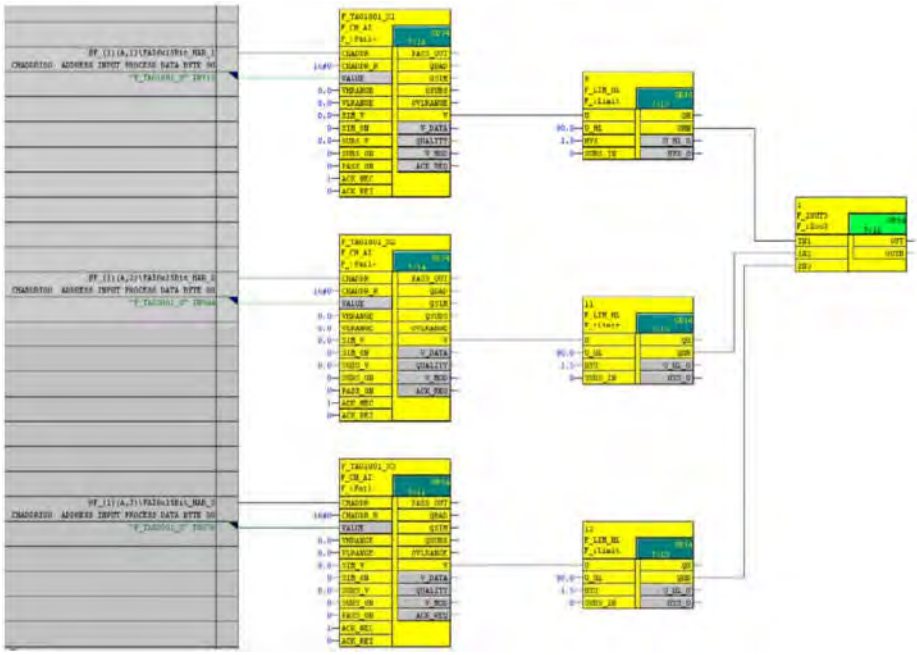
اگر دو سنسور یا دو ماژول *F-AI* شکست بخورد، تابع واکنش خطا از طریق سیستم *failsafe* عملکرد ایمنی را فراهم می‌کند.

توجه شود که با توجه به انعطاف پذیری سیستم، ساختارهای دیگری نیز امکان پذیر است. که با توجه به در دسترس بودن ماژول‌ها در رک *ET 200M*، متفاوت از نمونه توصیف شده می‌باشد. برای مثال:

- برای کاربردهای در دسترس پذیری پایین (*Lower availability*)، تمامی‌هر سه سنسور به یک ماژول متصل می‌شود.
- برای کاربردهای در دسترس‌پذیری یکسان ماژول‌ها (*Similar availability of the modules*)، هر سه سنسور به دو ماژول که با هم به صورت افزونه هستند، متصل می‌شود. هر دو ماژول در یک رک *ET200M* نصب شده‌اند.
- برای کاربردهای در دسترس‌پذیری بالا در ماژول‌ها و رک‌های *ET200M* (*High availability*)هر سه سنسور به دو ماژول که باهم به صورت افزونه هستند، و دو ماژول در دو رک *ET200M* متفاوت نصب شده‌اند، متصل می‌شود.
- برای کاربردهای در دسترس‌پذیری بالا در ماژول‌ها و رک‌های *ET200M* (*High availability*)، هر سنسور به یک ماژول متصل می‌شود. ماژول‌ها در رک‌های *ET200M* متفاوت نصب شده‌اند.

۴-۴-۲- لاجیک ارزیابی *2oo3* بدون درنظر گرفتن خطای کانال

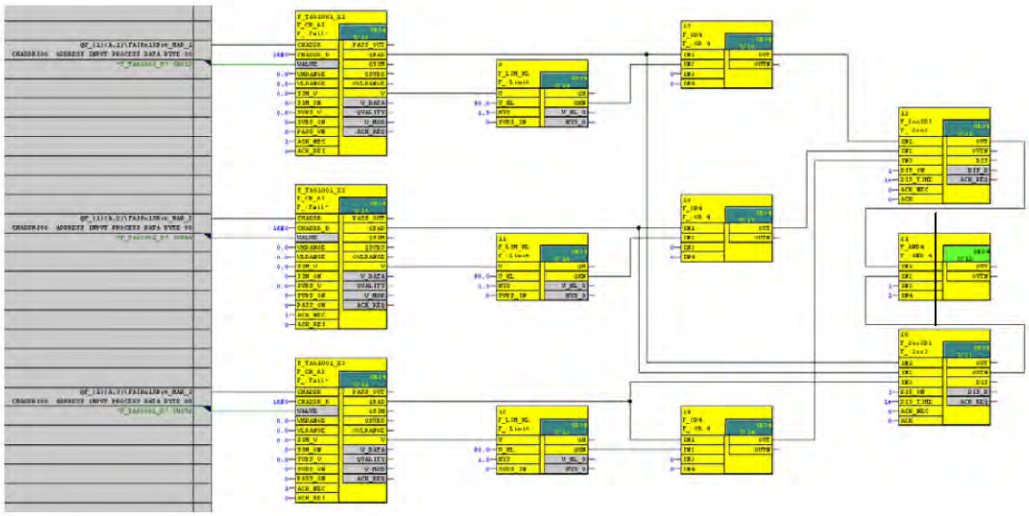
شکل زیر یک مثال از پیکربندی ارزیابی *2oo3* را در محیط ویرایشگر *CFC* نشان می‌دهد. که در آن خطای کانال (پایه خروجی*BAD* از بلاک درایور کانال آنالوگ) درنظر گرفته نشده است. توجه شود که در این مثال یک مقدار حد ماکزیمم فرض کرده و خروجی لاجیک ارزیابی برای دستیابی به وضعیت *failsafe*به صفر سویچ می‌کند. (*normal state =1, failsafe state = 0*)



شکل ٤-١٢- نمونه لاجیک ارزیابی *2oo3*

۴-۴-۳- پیکربندی *2oo3* با در نظرگرفتن خطای کانال (*2oo3D*)

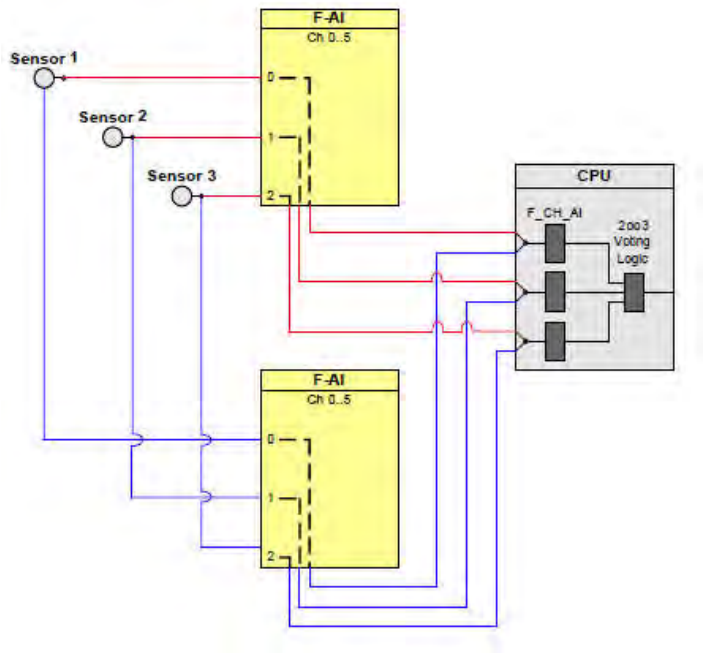
شکل زیر یک پیکربندی نمونه از ارزیابی *2oo3D* (*D→ With Diagnostic*) در ویرایشگر *CFC* را نشان می‌دهد. که در آن خطای کانال در لاجیک ارزیابی درنظر گرفته شده است. توجه شود که در این مثال یک در ارزیابی مقدار سنسور آنالوگ، یک مقدار حد *MAX* فرض شده و خروجی لاجیک ارزیابی برای دستیابی به وضعیت *failsafe* به صفر سویچ می‌کند. (*normal state =1, failsafe state = 0*)



شکل ٤-١٣- نمونه لاجیک ارزیابی *2oo3D*

۴-۴-۴- معماری استفاده از سه سنسور با دو ماژول افزونه (*Three sensors with redundant F-AI*)

در این معماری که ارزیابی *2(2oo3)* نامیده می‌شود. سه سنسور به دو ماژول *F-AI* افزونه متصل می‌شود. همانند معماری قبلی این طرح ارزیابی *2oo3* نیز به کاربردهایی اشاره می‌کند که در آنها برای رسیدن به سطح ایمنی موردنیاز به دو سنسور نیاز دارد.سنسور سوم دردسترس‌بودن (*availability*) را افزایش می‌دهد. به طوری که دو تا از سه سنسور بایستی کار بکنند. اگر حداقل دو تا از سه سنسور به شرایط تریپ برود، لاجیک تریپ را فعال خواهند کرد.



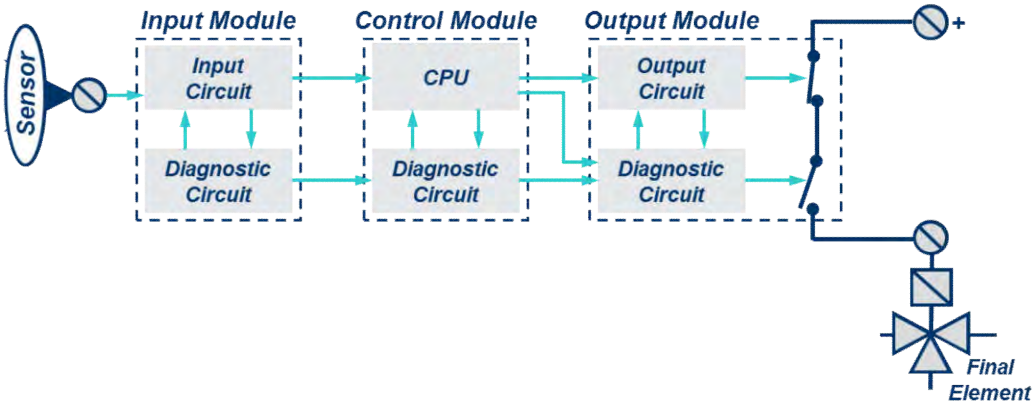
شکل ٤-١٤- اتصال دو سنسور بخ دو ماژول ریداندانت در معماری *2oo3*

۴-۵- معماری *1oo1D* (*with diagnostics – Non Redundant Safety System*)

این معماری مطابق شکل زیر یک معماری تک کانال با قابلیت خودآزمون(*Self-Tests*) جامع می‌باشد. که دارای یک مسیر ثانویه *Shutdown* کنترل شده با مدار *diagnostics* می‌باشد.

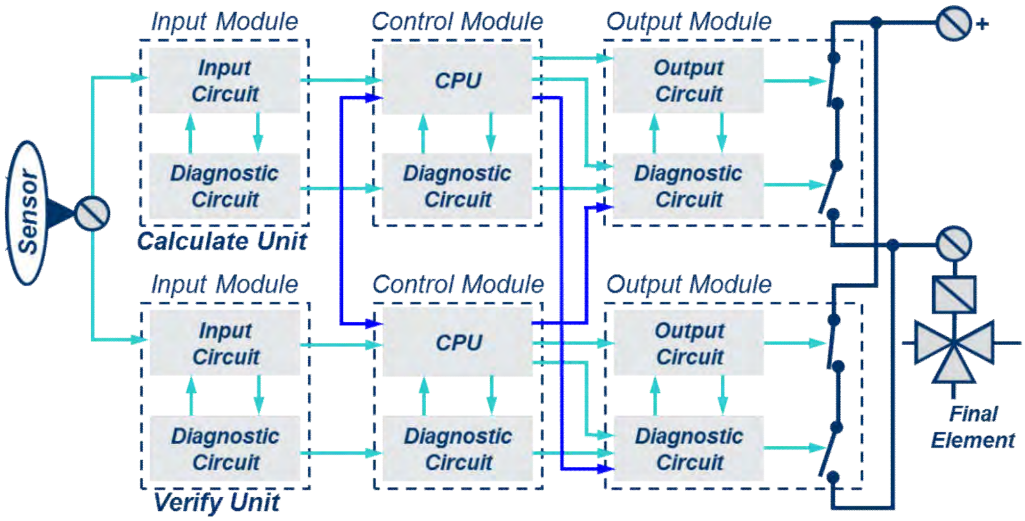
معماری *1oo1D* برای تبدیل شکست‌های خطرناک به شکست‌های امن (*safe failures*) از طریق قطع انرژی خروجی (*de-energizing*) از مدار تشخیص خاص بهره گیری می‌کند.

مدار *I/O* طراحی شده برای عملکرد به صورت خرابی امن دارای قابلیت تشخیص یکپارچه (*built-in diagnostics*) می‌باشد. پردازنده نیز دارای مدار تشخیص برای حافظه، عملیات *CPU*، تایمر *watchdog* و کلیه سیستم‌های ارتباطی می‌باشد.



۴-۴-۶- معماری 1oo2D (with diagnostics)

این معماری مطابق شکل زیر یک سیستم دو کانال (1oo2D) با قابلیت خودآزمون(Self-Tests) جامع می‌باشد. که دارای یک مسیر ثانویه Shutdown کنترل شده با مدار diagnostics می‌باشد.

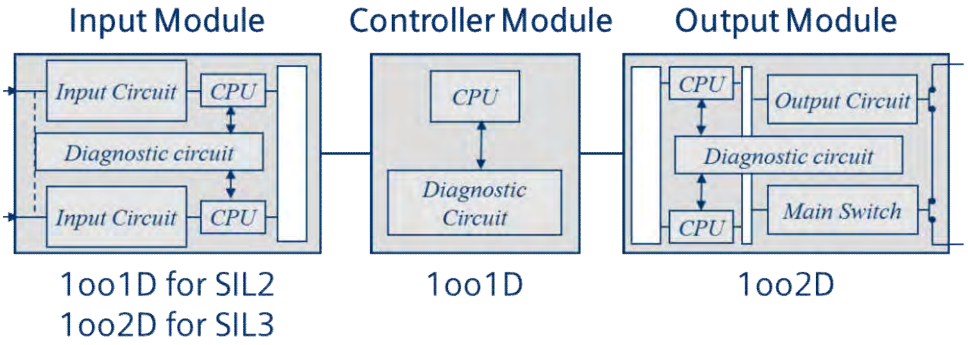


در این معماری همان شکست‌های موجود در معماری 1OO1D نیز وجود دارد. با این تفاوت که ماژول‌های IO به صورت پیکربندی افزونه بسته می‌شوند.

اگر مدار تشخیص، یک شکست در یک کانال از سیستم (ورودی/ پردازش‌گرها/ خروجی) را تشخیص دهد. طرف دیگر کنترل را حفظ و سیستم در دسترس باقی می‌ماند.

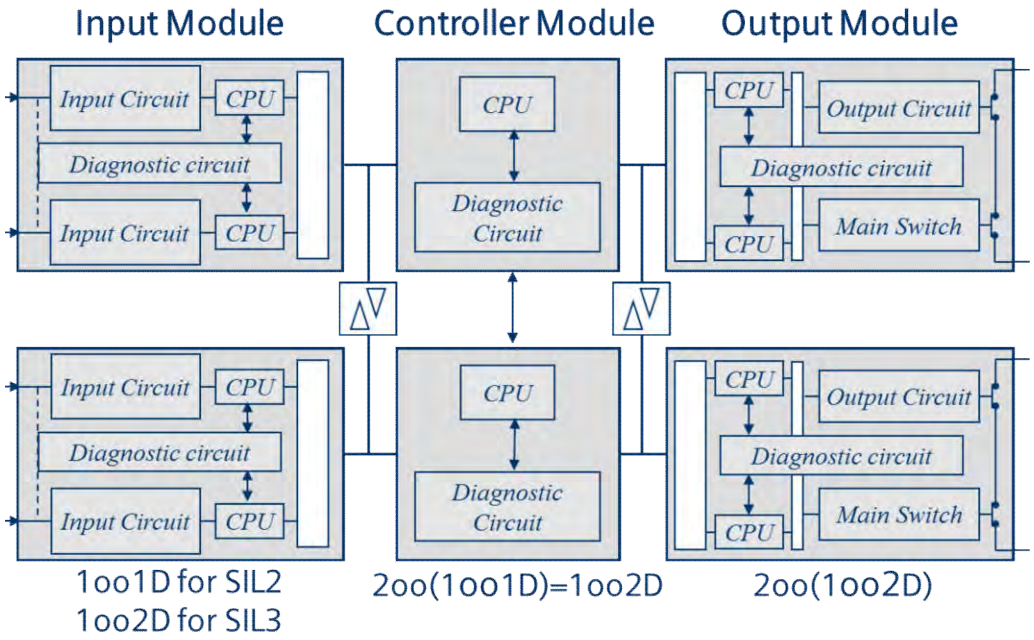
۴-۴-۷- معماری طراحی سیستم S7-400F (Non-Redundant Setup)

- *Single or Dual Sensors for inputs*
- *Redundant circuitry within I/O modules*
- *Redundant output circuits*



۴-۴-۸- معماری طراحی سیستم S7-400HF (Fully-Redundant Setup)

- *Single, Dual, triple or quad Sensors for inputs*
- *Quad output circuits*



۴–۵– مثال‌هایی از پاسخ سیستم تحمل‌خطا (*Fault-Tolerant*) به یک فالت

۴–۵–۱– مثال ۱– توقف *CPU* یا ماژول تغذیه

وضعیت اولیه: سیستم *S7-400H* در حالت افزونه می‌باشد.

۱– با خاموش کردن ماژول تغذیه *CPU0* را به حالت توقف می‌بریم.

نتیجه: *LED*های *REDF*، *IFM1F* و *IFM2F* در *CPU1* روشن می‌شوند. ماژول *CPU1* به حالت عملکرد تکی (*solo mode*) می‌رود و اجرای برنامه کاربر ادامه می‌یابد.

۲– منبع تغذیه رک صفر مربوط به *CPU0* را دوباره روشن کنید.

نتیجه:

- *CPU0* عمل *LINK-UP* و *UPDATE* را به صورت خودکار انجام می‌دهد.
- *CPU0* به حالت *Run* می‌رود و به عنوان *CPU* آماده به کار (*standby*) عمل می‌کند.
- حال سیستم *S7-400H* در مد سیستم افزونه می‌باشد.

۴–۵–۲– مثال ۲– خطا یا نقص در کابل فیبر نوری (*Failure of a fiber-optic*)

وضعیت اولیه: سیستم *S7-400H* در مد افزونه می‌باشد. سویچ انتخاب مد عملیاتی در هر دو *CPU* در مد *RUN* یا *RUN-P* می‌باشد.

۱– کانکتور یکی از کابل‌های فیبر نوری را قطع کنید.

نتیجه: *LED*های *REDF* و *IFM1F* یا *IFM2F* (بسته به این که کدام کابل فیبر نوری قطع می‌شود) در هر دو *CPU* روشن می‌گردد.*CPU* اصلی (*CPU0*) به حالت تکی سویچ می‌کند و برنامه کاربر همچنان اجرا می‌شود.

۲- کابل فیبر نوری که قطع شده بودهٔ دوباره وصل کنید.

۳- *CPU* آماده به کار (*CPU1*) را که در حال حاضر در مد *STOP* می‌باشد، با استفاده از *STEP7* راه‌اندازی مجدد کنید.

نتیجه:

- *CPU1* عمل *LINK-UP* و *UPDATE* را به صورت خودکار انجام می‌دهد.
- سیستم *S7-400H* به حالت افزونه می‌رود.

۴-۶- انواع سیستم *F*

برای سیستم‌های *S7* دو نوع سیستم ایمنی (*Safety Systems*) یا بسته نرم‌افزار توسعه برنامه ایمنی وجود دارد. که عبارت‌انداز:

- *Distributed Safety* برای محیط‌های اتوماسیون کارخانه
- *S7 F Systems* برای محیط‌های اتوماسیون فرآیند

این دو دارای مشترکات و تفاوت‌هایی هستند. برای محاسبات زمان مانیتورینگ *f*برای هر دو این سیستم‌ها از سوی زیمنس دو فایل اکسل معرفی شده است.

فایل *S7ftimeb.xls* برای *S7 F Systems*

فایل *S7fcotib.xls* برای *Distributed Safety*

۴-۷- نصب بسته *F System*

برای پیاده‌سازی برنامه یا لاجیک *F* در یک پروژه *PCS7* بایستی بسته‌های نرم‌افزاری زیر روی سیستم ایستگاه مهندسی(*ES*) نصب گردد:

- *S7 F Configuration Pack V5.5 SP9 Upd. 1*
- *S7 F Systems V6.1 SP1*
- *S7 F Systems Lib V1_3 SP1 Upd.1*

نکته : بسته *S7 F Systems V6.1 SP1* را می‌توان با کتابخانه *F* نسخه *Lib V1_2* نیز به کار برد.

۴-۷-۱- نصب بسته *F system* مربوط به *HMI*

در صورتی که از *SDW* یا *MOS* استفاده می‌شود، بایستی بسته *S7 F Systems HMI V6.1 SP1* روی کامپیوتر *OS* نصب شود.

- *SDW : Safety Data Write*
- *MOS : Maintenance Override Switch*

هم چنین بسته *Safety Matrix Viewer V6.2 SP1* نیز روی *PC* مربوط به *OS* نصب شود.

۴-۸- تنظیمات پیشرفته برای سیستم‌های *FH* در *PCS7* (*Advanced PCS 7 ES settings*)

۴-۸-۱- مکانیزم‌های ایمنی

برای پیاده‌سازی توابع ایمنی به صورت *Fail safe* در سیستم *S7* در *CPU* های *S7* دو نوع مکانیزم ایمنی پیش‌بینی شده است. که عبارت است از:

- حفاظت از برنامه کاربر در مقابل دسترسی‌های غیرمجاز به سیستم *F*، که این کار از وقوع خطاهای عمدی و سهی پیشگیری می‌کند.
- اجرای توابع خودآزمون (*SelfTest*) در *CPU* که به تشخیص خطا در ماژول‌ها کمک می‌کند

۴-۸-۲- حفاظت از دسترسی (*Access Protection*)

یک ایستگاه *S7-400F/FH* که به عنوان یک سیستم *safety* (*SIS or ESD*) عمل می‌کند توسط دو تا پسوورد محافظت می‌شود.

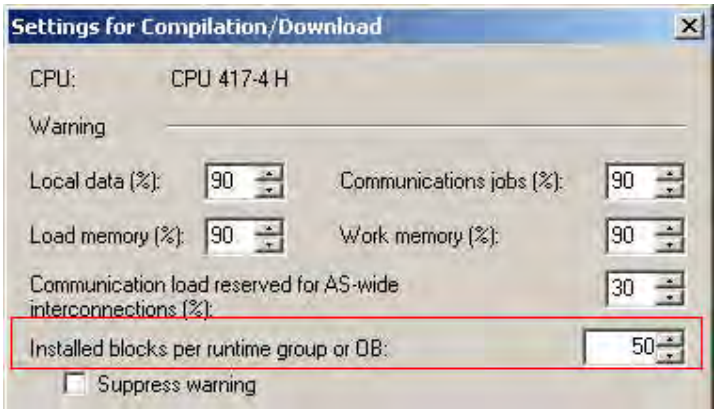
۱. یکی، پسوورد دسترسی به *CPU* می‌باشد. که با هدف محافظت از پارامترهای *CPU* در مقابل دانلود تصادفی یا دانلود اشتباهی برنامه می‌باشد. این پسوورد در *HWConfig* پیکربندی می‌شود.
۲. پسوورد دیگر، برای ویرایش برنامه *F* می‌باشد که در پنجره *Edit Safety Program* پیکربندی می‌شود. به طوری که از برنامه یا لاجیک *fail-safe*، پارامترهای پیکربندی شده برای ماژول‌های سیگنال و پارامترهای *fail-safe* ماژول *CPU* (گزینه ای در سربرگ *Protection* مربوط به *CPU*) محافظت می‌کند.

نکته: در خصوص *PCS 7 V7.0* و بالاتر با *SIMATIC logon*، امکان تعریف رمز عبور برای حفاظت از پروژه *PCS 7* نیز وجود دارد. دسترسی به *ES* نیز توسط یک رمز عبور در سطح سیستم عامل (ورود به ویندوز، محافظ صفحه نمایش) محافظت می‌شود.

۴-۸-۳- تنظیمات *CFC* برای کامپایل و دانلود

مطابق شکل زیر برای کامپایل صحیح چارت‌های *CFC* دارای بلوک‌های *F*، مقدار آستانه هشدار مبتنی بر تعداد بلوک‌ها به ازای هر *Runtime group* را تنظیم کنید. مقدار پیش فرض ۵۰ است.

CFC Editor > Options > Cusomize > Compile/download



۴–۹– پیکربندی سخت افزار *S7F/FH*

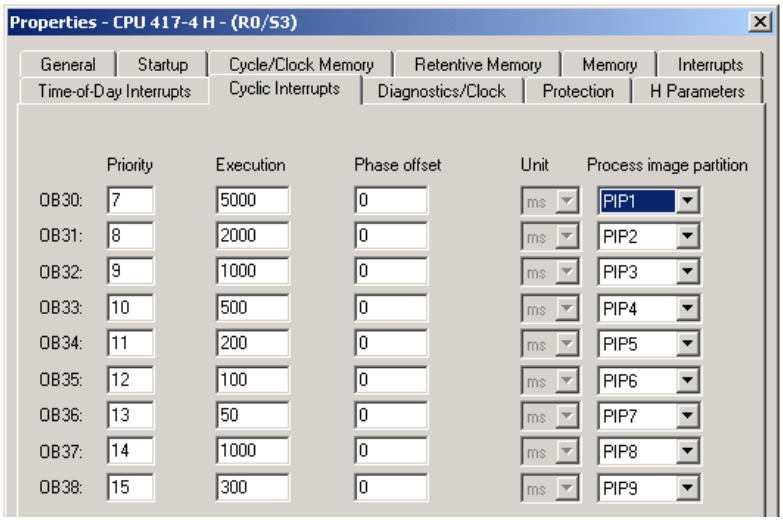
انواع ماژول‌های *CPU* نوع *H* که برای بکارگیری به عنوان سیستم *S7 F* در *PCS7* ارائه شده است، عبارت اندار:

- *CPU 412H ,CPU 414H, CPU 416H, CPU 417H*

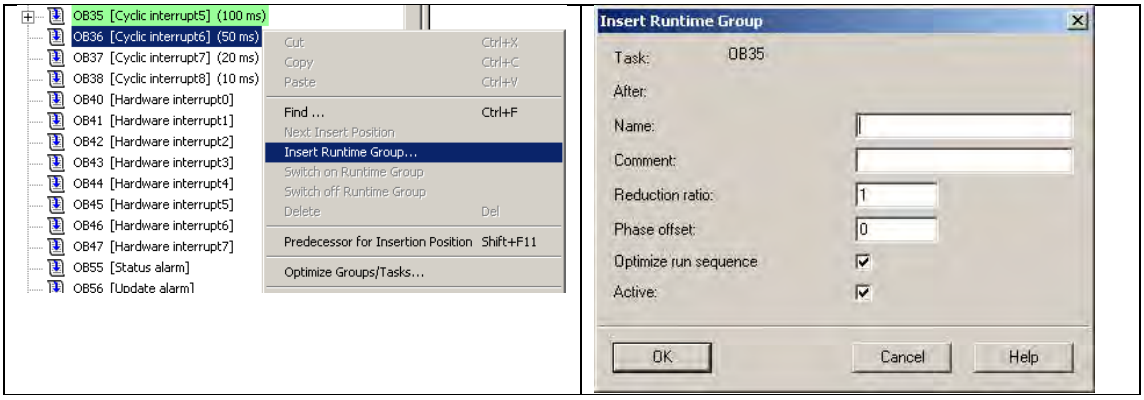
۴–۱۰– تطبیق پارامترهای *CPU*

۴–۱۰–۱– سربرگ *Cyclic interrupts*

برای بخش‌هایی از برنامه لاجیک *CPU* که از نوع *F* هستند، نیازی به پیکربندی پارتیشن‌های *Process image* در پنجره پراپرتی *CPU* نمی‌باشد. به این دلیل که داده استخراج شده توسط ماژول‌های سیگنال *F*، در سیستم‌های نوع *S7 F/FH* از طریق حافظه *process image* مورد دسترسی واقع نمی‌شوند. بلکه دسترسی به آنها به صورت مستقیم از طریق درایور ماژول *F* صورت می‌گیرد. با این حال، اگر قطعات برنامه استاندارد در *OB*های سیکلیک مورد استفاده قرار می‌گیرد. تعریف بخش‌های *Process image* در سربرگ *Cyclic Interrupts* نیاز می‌باشد.



نکته: پس از درج یک *Runtime group* در پنجره *Run Sequence* گزینه‌های *Reduction ratio* و *Phase offset* مرتبط با *OB*های وقفه سیکلیک در برنامه *F* پیکربندی نمی‌شود.

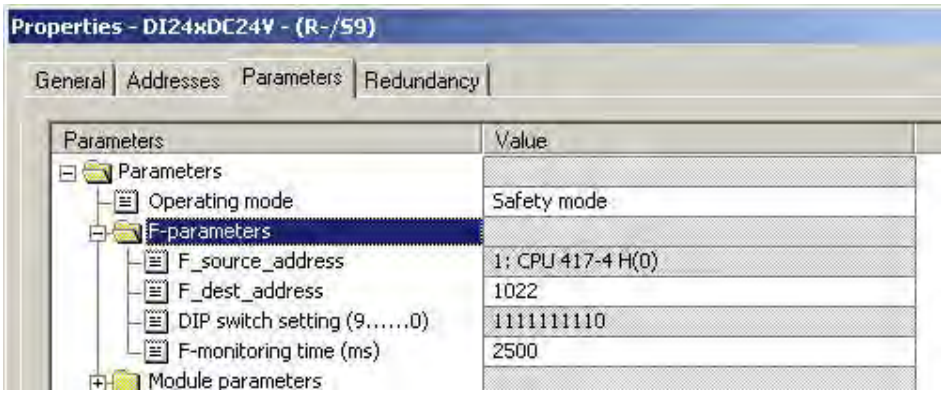


۴-۱۱- تنظیمات پارامترهای ماژول‌های سیگنال *F*

شبیه ماژول‌های سیگنال استاندارد، ماژول‌های سیگنال *F* نیز در محیط *HW Config* پیکربندی می‌شوند. برای این کار بایستی بسته *F Configuration Pack* نصب شود.

در سیستم *F* کانال‌های استفاده نشده را در زمانی که سیستم کار می‌کند، را می‌توان به برنامه اضافه کرد. مشروط بر این که در طول اولین راه‌اندازی، در *HW Config* فعال شده و به منظور جلوگیری از صدور فالت، کانال مربوطه به یک مقاومت مجهز شده باشد.

همین که ماژول‌های سیگنال *F* را به ایستگاه *ET200M* در *HwConfig* اضافه کردید، با دوبار کلیک روی ماژول سیگنال و یا از طریق منوی *Edit > Object Properties* می‌توان به پنجره پیکربندی ماژول *F* دسترسی داشت.



شکل ٤-١٥- پنجره پراپرتی یک ماژول سیگنال *FDI*

- **مد کاری (*Operating mode*):**

مطابق شکل بالا جهت عملکرد به عنوان ماژول *F* نه به عنوان ماژول استاندارد، بایستی مد کاری *Safety* برای ماژول‌های سیگنال *F* دیجیتال فعال شود. برای ماژول‌های‌هارت آنالوگ *F* این گزینه وجود ندارد.

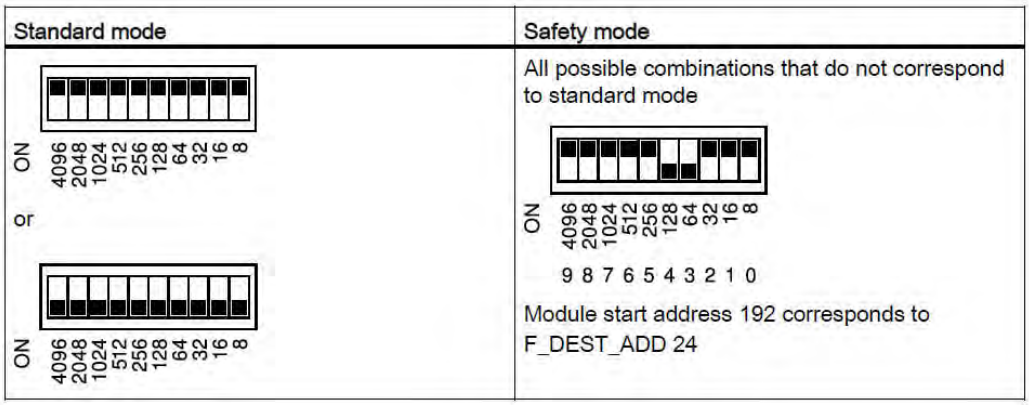
- **آدرس‌های *PROFIsafe*:**

آدرس‌های مبدا و مقصد پروتکل *PROFIsafe* جهت ارتباط *PROFIsafe* بین *CPU* و ماژول می‌باشد. که عبارت است از *F_SOURCE_ADD* و *F_DEST_ADD* . این آدرس‌ها به صورت منحصر به فرد برای شناسایی مبدا و مقصد در هنگام برقراری ارتباط *PROFIsafe* استفاده می‌شود. آدرس *F_DEST_ADD* مقصد *PROFIsafe* (ماژول) را شناسایی می‌کند. این آدرس باید، در هر دو، یعنی شبکه و ایستگاه منحصر به فرد باشد. آدرس *F_SOURCE_ADD* به طور دائمی به *CPU* اختصاص داده می‌شود.

برای جلوگیری از وقوع اشتباهات در انتساب پارامتر، آدرس‌های مبدا و مقصد پروفی‌سیف در پارامترهای ماژول *F* به طور خودکار اختصاص داده می‌شوند.

۴-۱۱-۱- تنظیمات سخت‌افزاری (*DIP switch settings*)

آدرس‌های تولید شده در بخش *Parameters* ماژول بایستی به صورت سخت‌افزاری در کارت تنظیم شوند. برای تنظیم آدرس *F_DEST_ADD*، در هر یک از ماژول‌های *F* یک دیپ سویچ تعبیه شده است، که یک نمایش باینری از همان آدرس می‌باشد. از آن جایی که این دیپ سویچ در پشت ماژول است، این تنظیم بایستی قبل از نصب ماژول سیگنال *F* بر روی رک انجام شود.

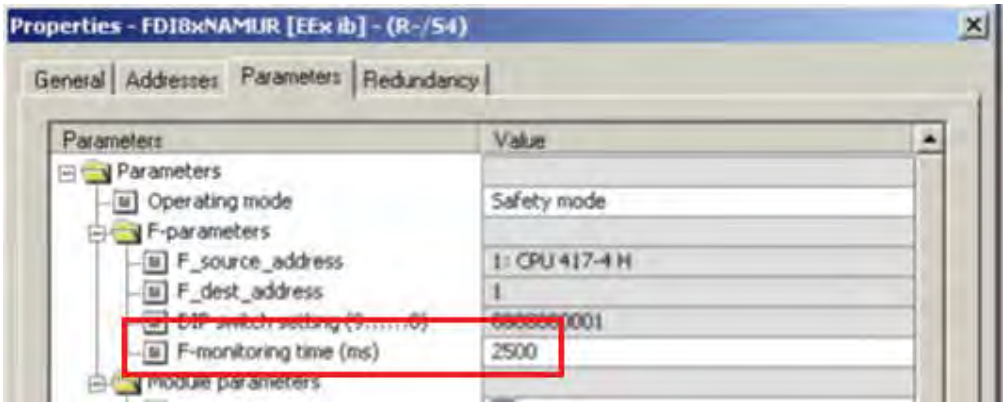


زمانی که این ماژول‌ها در مد استاندارد کار می‌کنند. آدرس *F_DEST_ADD* همواره به مقدار صفر (*delivery condition*) تنظیم می‌شود.

در مورد نسخه‌های قدیمی ماژول‌های نوع *F* آدرس *F_DEST_ADD* وابسته به آدرس شروع ماژول می‌باشد. که عبارت است از: *F_DEST_ADD = Module start address/8*

۴-۱۱-۲- زمان مانیتورینگ *F*

اگر ماژول‌های سیگنال *F* در مد کاری *Safety* کار می‌کنند، بایستی در پنجره پراپرتی ماژول *F*، زمان مانیتورینگ *F*، برای برقراری ارتباطات *Safety* بین *F CPU* و ماژول سیگنال *F* تنظیم شود. (زمان مانیتورینگ *PROFIsafe*). مقدار پیش فرض این زمان ۲۵۰۰ میلی ثانیه است. که معمولاً از همین مقدار استفاده می‌شود.



شکل ٤-١٦- زمان مانیتورینگ ارتباطات *F* در *Profisafe*

1. *Diagnostic interrupt*

برای فعال کردن قابلیت گزارش وقفه در بلوک‌های داریور *PCS7*، وقفه تشخیصی برای ماژول سیگنال *F* باید همیشه در حالت *safety* فعال شود.

رویدادهای مختلف خطا، که ماژول سیگنال خرابی امن می‌تواند با استفاده از تابع تشخیص تعریف کند، یک وقفه تشخیص را تریگر می‌کند. رویدادهای تشخیصی که رخ می‌دهد توسط ماژول *F CPU* در دسترس قرار می‌گیرد.

2. *Group diagnostics*

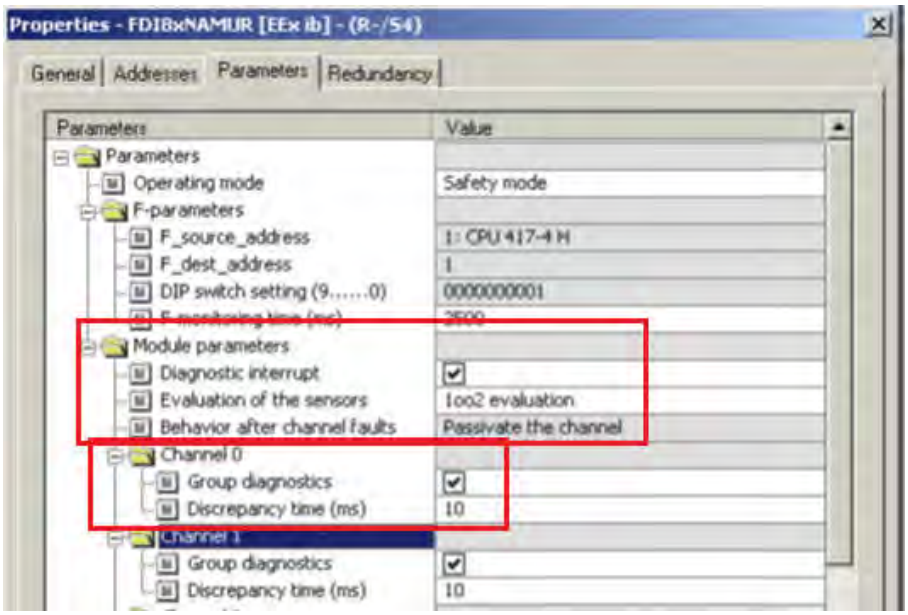
اگر مطابق شکل زیر باکس مربوط به یک کانال خاص را تیک بزنید، وقوع یک رویداد خاص یک کانال (مانند قطعی سیم) یک واکنش خطا در برنامه امنیتی را تریگر می‌کند (مقدار جایگزین در بلاک درایور کانال فعال شده و *QBAD* ست می‌شود). اگر گزینه «*Enable diagnostics interrupt*» انتخاب شود، یک اینتراپت تشخیص در *CPU* تریگر شده و مطابق آن یک از نوع *process control message* به سیستم *OS* ارسال می‌شود.

پارامتر *Group diagnostics* برای فعال و غیرفعال کردن پیام‌های *channel-specific diagnostic* (به عنوان مثال قطع سیم یا اتصال کوتاه) روی ماژول‌های سیگنال *F* به *CPU* استفاده می‌شود. در ماژول‌های سیگنال *F* هر موقع که کانال فعال شود (یعنی *Spare* نباشد)، گزینه *group diagnostics* نیاز است که فعال شود.

برخی از ماژول‌های ورودی/خروجی نوع *F* زیمنس عبارت انداز:

- *SM 326; DI 8 x NAMUR (order no. 6ES7326-1RF00-0AB0)*
- *SM 326; DI 24 x DC 24 V (order no. 6ES7326-1BK00-0AB0)*

- *SM 336; DO 10 x DC 24V/2A (order no. 6ES7326-2BF00-0AB0)*
- *SM 326; AI 6 x 13 Bit (order no. 6ES7336-1HE00-0AB0)*

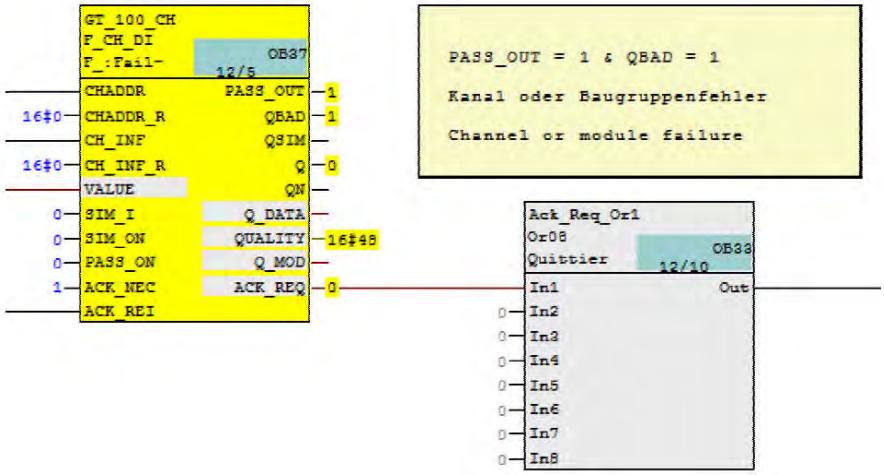


شکل ٤-١٧- فعال‌سازی قابلیت *Diagnostic Interrupt* در ماژول *FDI*

۴-۱۱-۳- مفهوم *Passivation* در حالت کلی (غیرفعال شدن کانال/ماژول‌های ورودی/خروجی)

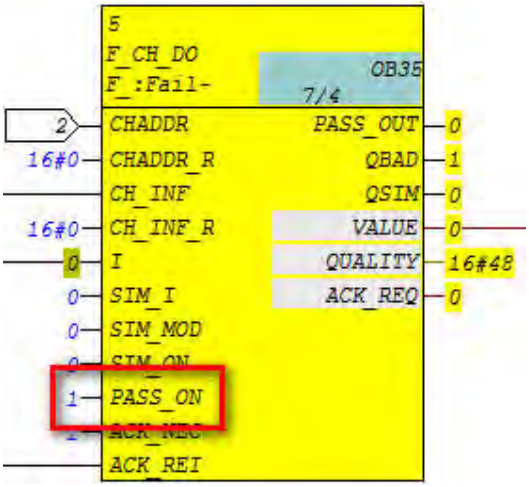
Passivation بدان معنی است که در صورت وقوع یک خطا، یک یا تعدادی از کانال‌های یک ماژول سیگنال *F*، به وضعیت امن (منطق صفر) (*safe state*) سویچ می‌کند. در صورت وقوع خطا در یک کانال (برای مثال معیوب شدن یک سنسور)، تنها کانال‌های آسیب دیده پسیو/غیرفعال می‌شوند. در صورت وقوع خطا در ماژول (به عنوان مثال یک خطای ارتباطی)، تمام کانال‌ها بر روی ماژول *F-I/O* پسیو می‌شوند.

اگر یک ماژول سیگنال *F* خطایی تشخیص دهد، کانال‌های آسیب دیده و یا همه کانال‌های آن را به حالت امن سوئیچ می‌کند. به عبارت دیگر، کانال‌های این ماژول غیرفعال می‌شوند. ماژول سیگنال *F* برای نشان دادن این که خطایی شناسایی شده است، یک پیام به درایور کانال *F* و *PCS 7 OS* می‌فرستد.

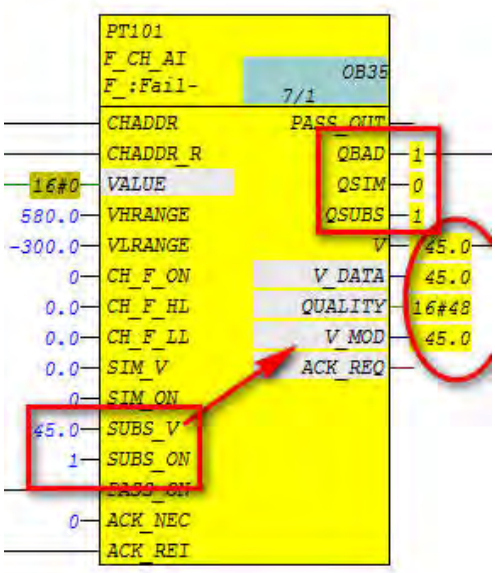


همچنین از ورودی *PASS_ON* در یک بلاک درایور کانال نیز می‌توان برای برای فعال و غیرفعال کردن *passivate* شدن کانال در برنامه *F* استفاده کرد. به عنوان مثال، با استفاده از یک شرایط خاص در ترتیب زمانی برنامه (*program sequence*) و یا راه اندازی مجدد حفاظت (*restart protection*).

هنگامی‌که کانال‌های خروجی پسیو می‌شوند، تغذیه خروجی‌ها قطع می‌شود (*de-energized*) (به جریان صفر و یا ولتاژ صفر تنظیم می‌شوند). بلاک درایور یک کانال خروجی دیجیتال *F* پسیو شده، یک مقدار جایگزین (*substitute*) با کد کیفیت *16 # 48* را صادر و خروجی *QBAD* را به ۱ تنظیم می‌کند.



بلاک درایور یک کانال ورودی دیجیتال *F* پسیو شده، مقدار جایگزین *0* را با کد با کیفیت *16 # 48* در خروجی قرار داده و خروجی *QBAD* را یک می‌کند. با توجه به انتساب مقدار در پارامتر ورودی *SUBS_ON*، درایور یک کانال ورودی آنالوگ *F*، هم یک مقدار جایگزین با کد کیفیت *16 # 48* و یا آخرین مقدار معتبر با کد با کیفیت *16 # 44* را در خروجی قرارداده و خروجی *QBAD* را نیز به یک تنظیم می‌کند. در صورتی که یک مقدار جایگزین وارد شده در ورودی *SUBS_V* در خروجی بلاک ظاهر شود، در آن صورت خروجی *QSUBS* نیز یک می‌شود.



در صورتی که یک چارت *F-CFC* برای اولین بار باز می‌شود، پیغام زیر نمایش داده می‌شود.که محتوای پیغام این است که عمل دانلود چارت‌ها به *CPU* تنها در مد *Safety* غیرفعال امکان‌پذر است.

۴-۱۱-۴- پسیو شدن گروهی (*Group passivation*)

اگر در حین پسیو شدن یک ماژول *F-I/O* یا یک کانال *F-I/O* بخواهید ماژول‌های *F-I/O* را پسیو کنید، برای این کار می‌توان از خروجی *PASS_OUT* و ورودی *PASS_ON* جهت پسیو کردن گروهی استفاده کرد.

عمل پسیو کردن گروهی از طریق ورودی/خروجی‌های *PASS_OUT/PASS_ON* به عنوان مثال برای *Force* کردن عمل *reintegration* همزمان تمام *F-I/O*ها بعد از این که سیستم *F* راه‌اندازی گردید، می‌تواند مورد استفاده قرار گیرد.

برای فعال کردن *Group passivation* بایستی تمام پایه‌های خروجی *PASS_OUT* بلاک‌های درایور کانال *F* در یک گروه را توسط یک بلاک *F_OR4* به هم متصل نموده و خروجی *OUT* از بلاک *F_OR4* را به ورودی‌های *PASS_ON* بلاک‌های درایور کانال در این گروه متصل کنید.

۴-۱۱-۵- فعال کردن کانال‌ها (*Activating channels*)

به واسطه ساختار ماژول‌های سیگنال *F*، بدون *Passivate* کردن ماژول، امکان ایجاد تغییرات در پیکربندی سخت‌افزار یا دانلود به آنها وجود ندارد. *Passivate* کردن ماژول‌های خروجی شامل ایجاد یک وضعیت ایمن (*safe state*) روی تمام خروجی‌ها (صفر کردن مقادیر خروجی) می‌باشد. ولی *passivate* کردن ماژول‌های ورودی شامل صفر کردن خروجی بلاک‌های درایور می‌باشد.

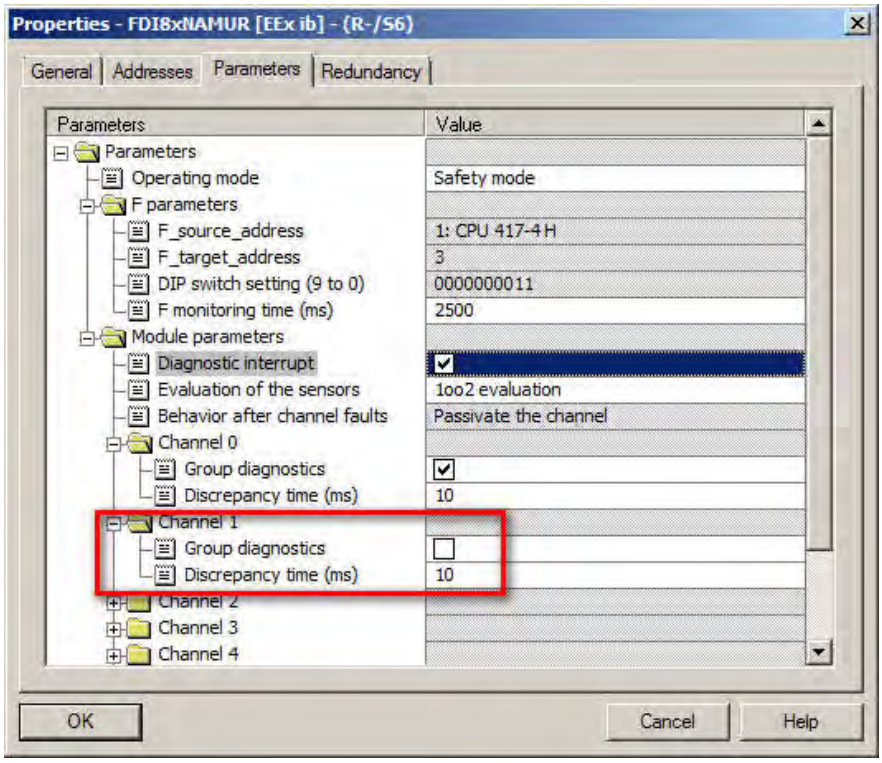
برای داشتن امکان فعال کردن کانال‌های استفاده نشده (*free channels*) در ماژول‌های سیگنال *F* در حین کار، بایستی کانال‌ها را در *HW Config* فعال کرد. ولی با توجه به قابلیت تشخیص (*diagnostics*) در ماژول‌های سیگنال *F*، کانال‌های فعال شده که بدون استفاده هستند، منجر به خطا می‌شوند، که می‌توانید با تجهیز کانال به یک مقاومت معادل از بروز خطا جلوگیری کرد.

- **پروسه فعال کردن ماژول سیگنال ورودی دیجیتال در حین کار**

SM326; DI 8 x NAMUR [EEx ib] (6ES7 326-1RF00-0AB0 / 6ES7 326-1RF01-0AB0)

۱. مد کاری (*Safety mode*) را تعریف کنید.
۲. پارامترهای مد کاری را تعریف کنید
۳. پارامترهای ماژول را تعریف کنید.
٤. گزینه *Group Diagnostics* را برای کانال فعال کنید.
۵. کانال را به یک مقاومت *1 kohm* متصل کنید.

مطابق شکل زیر می‌توان با برداشتن تیک مربوط به *group diagnostics* در یک کانال، آن کانال را غیرفعال کرد.



- **پروسه فعال کردن ماژول سیگنال خروجی دیجیتال در حین کار**

SM326; DO 10 x DC 24V/2A (6ES7 326-2BF01-0AB0)

۱. تعیین مد کاری (*Safety mode*)،
۲. تعریف پارامترهای ماژول،
۳. فعال کردن گزینه *Group Diagnostics* برای کانال،
٤. برای شبیه‌سازی یک عملگر (*actuator*) خروجی کانال را به یک مقاومت(مثال *2.7 kilohms*) و سر دیگر مقاومت را به زمین متصل کنید.

- **پروسه فعال کردن ماژول سیگنال خروجی دیجیتال *F* در حین کار**

SM326; F-DO 10 x DC 24V/2A PP (6ES7 326-2BF10-0AB0)

۱. تعریف پارامترهای ماژول،
۲. فعال کردن گزینه *Group Diagnostics* برای کانال،
۳. برای شبیه‌سازی یک عملگر (*actuator*) خروجی کانال را به یک مقاومت(مثال *2.7 kilohms*) و سر دیگر مقاومت را به زمین متصل کنید.

- **پروسه فعال کردن ماژول سیگنال ورودی آنالوگ *F* در حین کار**

SM 336; F-AI 6 x 13 Bit (6ES7 336-1HE00-0AB0)

۱. تعریف پارامترهای ماژول،
۲. فعال کردن گزینه *Group Diagnostics* برای کانال،

۳. انجام تنظیمات خاص کانال
٤. ورودی مثبت (*plus*) کانال را از طریق یک مقاومت (به عنوان مثال *3.9 kilohms*) به منبع ولتاژ (*L+*) و پایه منفی کانال ورودی را به زمین متصل کنید.

- **پروسه فعال کردن ماژول سیگنال ورودی آنالوگ *F* هارت در حین کار**

SM 336; F-AI 6 x 0/4...20mA HART (6ES7 336-4GE00 0AB0)

۱. تعریف پارامترهای ماژول
۲. تعیین نوع اتصال سنسور (دوسیمه یا چهار سیمه)
۳. انجام تنظیمات خاص کانال
٤. انجام تنظیمات مربوط به ارتباطات هارت
۵. ورودی مثبت (*plus*) کانال را از طریق یک مقاومت (به عنوان مثال *3.9 kilohms*) به منبع ولتاژ (*L+*) و پایه منفی کانال ورودی را به زمین متصل کنید.

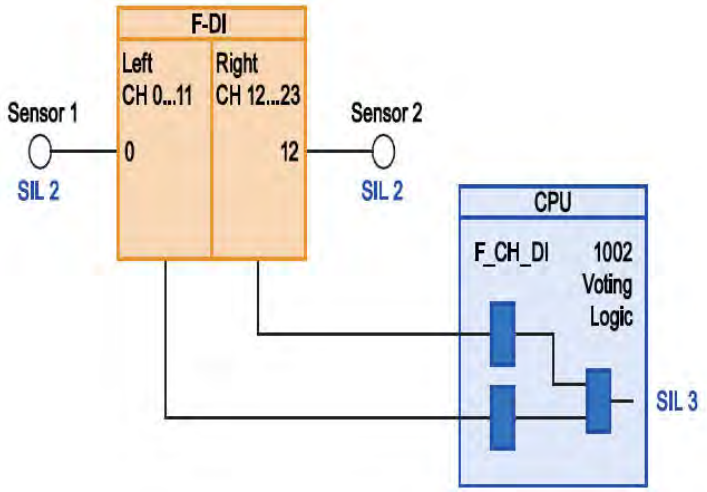
۴-۱۲- ارزیابی سنسور (*Sensor evaluation*)

۴-۱۲-۱- معماری‌های سیم‌بندی و انتخاب برای *ET 200M* (*Wiring and Voting*)

در ساختار معماری *1oo2* از دو سنسور استفاده می‌شود. در سیستم کنترل زیمنس برای حصول به سطوح *SIL3* عمل *Voting* سیگنال‌های ورودی *fail-safe* را می‌توان به دو صورت انجام داد که عبارت‌انداز:

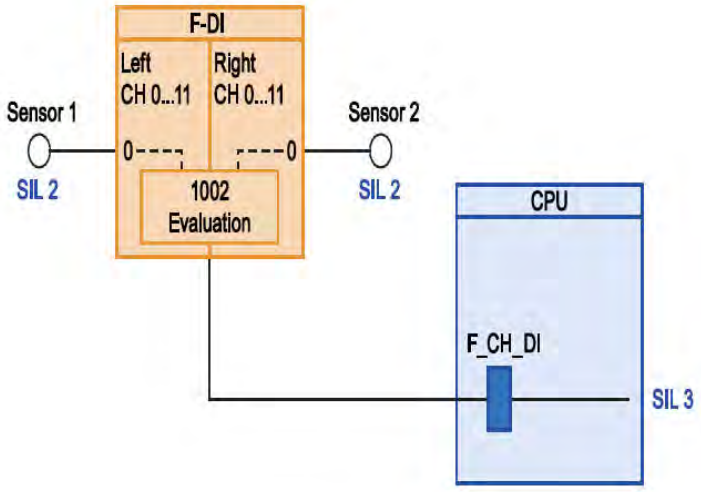
- انجام *Voting* در داخل ماژول سیگنال *F*؛
- پیاده‌سازی *Voting* در برنامه لاجیک (چارت‌های *CFC*)؛

شکل زیر حالتی را نشان می‌دهد که در آن از دو سنسور استفاده شده و این دو سنسور به دو کانال یک ماژول *F-IO* متصل شده است. همچنین عمل *voting* در برنامه *F* برای پیاده‌سازی کلاس *SIL 3* صورت گرفته است.



شکل ٤-١٨- انجام عمل *voting* به صورت برنامه *cfc* در *CPU*

شکل زیر نیز حالتی را نشان می‌دهد. که درآن برای پیاده‌سازی کلاس *SIL 3*، عمل *voting* بین دو سیگنال در داخل ماژول *F* انجام می‌شود.



شکل ٤-١٩- انجام عمل *voting* به صورت سخت‌افزاری در داخل کارت

نکته: برای دستیابی به این نوع سیم‌بندی *SIL3* سنسورهای مناسبی نیاز است.

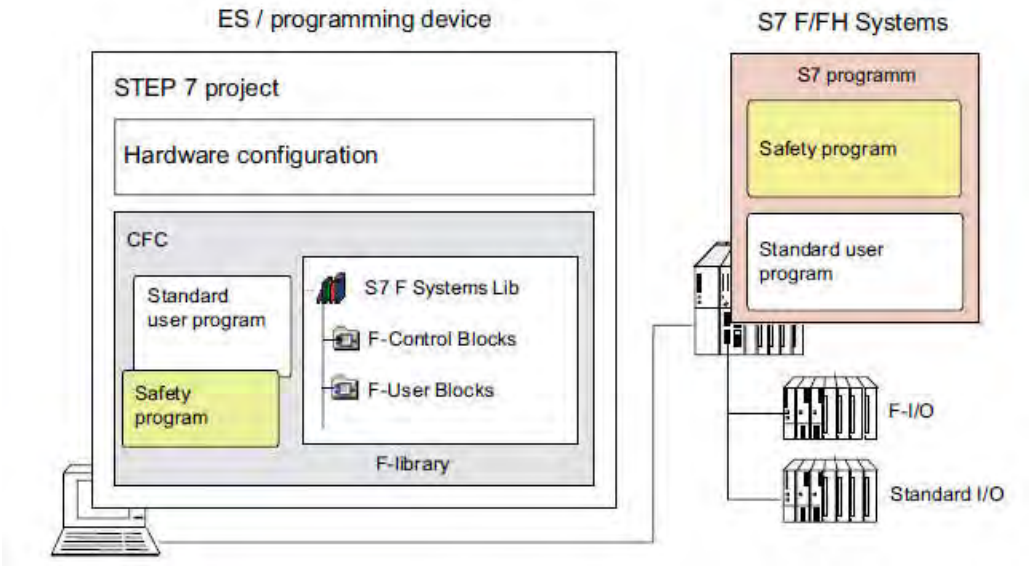
فصل – ۵ – برنامه‌نویسی بلاک‌های F (Programming)

۵–۱– مقدمه

یک برنامه F متشکل از بلوک‌های F می‌باشد. که از یک کتابخانه F انتخاب شده‌اند. بلاک‌های F با استفاده از چارت‌های CFC به هم متصل شده و به طور خودکار در زمان کامپایل برنامه ایمنی به پوشه Blocks اضافه می‌شوند. در طول کامپایل، اقدامات کنترل خطا (فالت) نیز به طور خودکار به برنامه لاجیک افزوده می‌شود و آزمون‌های مرتبط با ایمنی انجام می‌شود.

شکل ۵-۱ ساختار شماتیکی یک پروژه مبتنی بر برنامه S7 که حاوی هر دو لاجیک استاندارد و لاجیک ایمنی می‌باشد را به تصویر کشیده است. به طور کلی یک برنامه S7 ماژول‌های FH متشکل از دو نوع لاجیک می‌باشد:

- برنامه نوشته شده با بلاک‌های استاندارد که برای توابع یا عملکرد ایمنی ضروری نمی‌باشد.
- برنامه F که توابع ایمنی را پیاده‌سازی می‌کند.



شکل ۵-۱– ساختار شماتیکی یک پروژه مبتنی بر برنامه S7

۵–۱–۱– ایجاد برنامه/لاجیک F (*Configuring the safety program*)

برنامه لاجیک در CPU را می‌توان با استفاده از بلوک‌های F و بلوک‌های غیرF (بلاک‌های استاندارد) ایجاد کرد. بلاک‌های F در گروه‌های زمان اجراء جداگانه (*runtime groups*) پیکربندی می‌شود.

با توجه به این که یک سیستم FH هر دو برنامه استاندارد و F را در قالب یک برنامه S7 می‌تواند اجرا کند، بایستی یک روش امن در تبادل داده بین بخش F و بخش استاندارد برنامه S7 وجود داشته باشد. لذا انتقال داده‌ها بین برنامه استاندارد و F با استفاده از بلوک‌های تبدیل(*Convert*) واقع در کتابخانه F هندل

می‌شود. توجه داشته باشید که پیاده‌سازی عملکردهای ایمنی (مثلاً پیاده‌سازی یک حلقه کنترل ایمن) بایستی از نقطه قرائت سیگنال‌های ورودی(توسط بلاک‌های درایور ورودی) تا نقطه نوشتن فرامین در خروجی‌ها (توسط بلاک‌های درایور خروجی) به صورت پیوسته با بلوک‌های F پیاده‌سازی شود. به این معنی که بین این‌ها نباید از بلاک‌های استاندارد استفاده کرد.

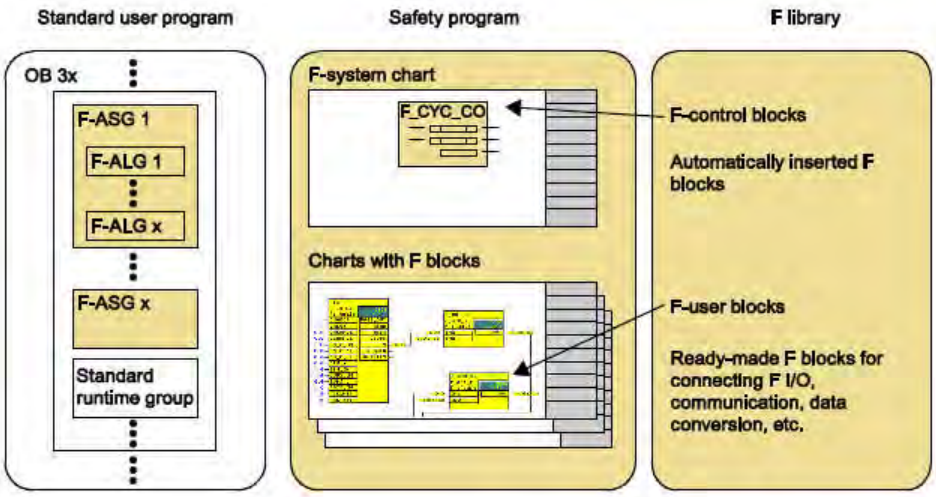
علاوه بر توابع پیاده‌سازی کننده عملکرد ایمنی، بلوک‌های F شامل توابعی برای تشخیص و واکنش به خطاها است. این فانکشن بلاک‌های تشخیص فالت، تضمین می‌کنند که شکست‌ها و خطاها شناسایی شده و برای حفظ سیستم F در حالت امن و یا هدایت آن به یک وضعیت امن، واکنش مناسب را تریگر می‌کنند.

در طول کامپایل، توابع خاص تشخیص خطا و واکنش به خطاها به طور خودکار به برنامه F اضافه می‌شوند.

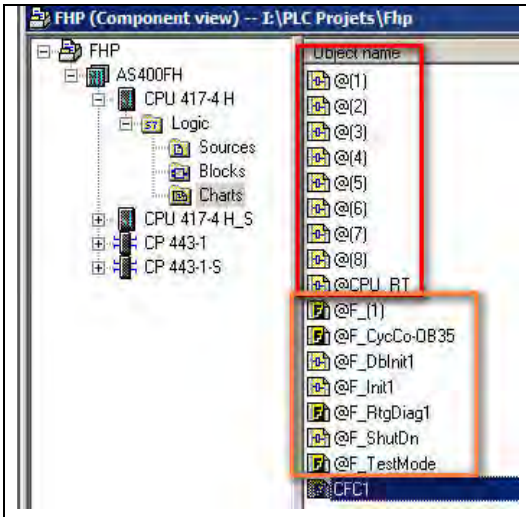
نکته: به طور کلی وضعیت *fail-safe* در توابع F، منطق صفر می‌باشد. به این معنی که حالت نرمال سیگنال یک بوده و وضعیت صفر، در یک سیگنال ورودی یا خروجی منجر به تریپ شده و سیستم متوقف می‌شود.

۵-۱-۲- ساختار برنامه ایمنی (*safety program*)

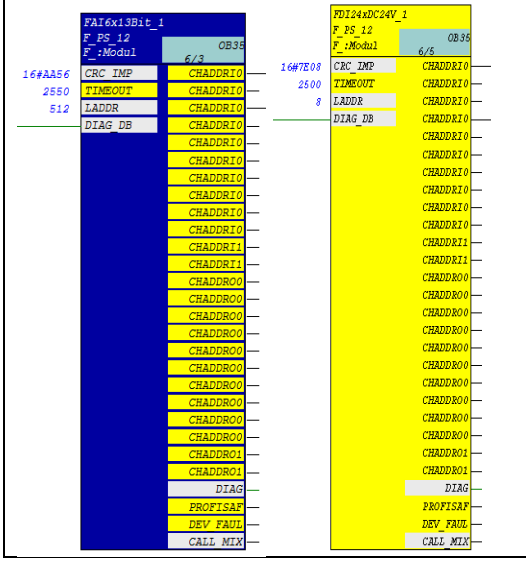
شکل زیر یک دیاگرام از ساختار یک برنامه F متشکل از چارت‌های CFC با بلاک‌های F را که جهت اجرا توسط CPU به گروه‌های $F\text{-}runtime$ تخصیص داده شده است را به تصویر کشیده است.



شکل ۵-۲- دیاگرام/ساختار یک برنامه ایمنی



همانطور که از شکل دیده می‌شود یک برنامه *F* متشکل از چارت‌های *F_CFC* می‌باشد که شامل بلاک‌های *F* می‌باشد. خود این بلاک‌های *F* نیز دو دسته می‌باشند. دسته اول که به بلاک‌های کنترل معروف هستند، به صورت اتوماتیک در زمان کامپایل برنامه *F* در چارت‌های *CFC* با پیشوند@ اضافه می‌شوند. دسته دیگر که به بلاک‌های کاربر معروف هستند، برحسب نیاز و کد لاجیک برنامه *F* توسط کاربر از طریق اتصال آنها در چارت‌های *CFC* به همدیگر، برنامه‌نویسی می‌شوند.



به عنوان مثال یکی از این بلاک‌های کنترل بلاک درایور ماژول می‌باشد. که مسئول ارتباطات *PROFIsafe* بین برنامه *F* و ماژول *F-I/O* می‌باشد. درایور ماژول *F* با فعال کردن تیک گزینه *Generator Module Drivers* در پنجره تنظیم کامپایل *CFC* به طور خودکار در چارت *F_(1)@* قرار داده شده و به کانال‌های مربوطه در برنامه *F* متصل می‌شود.

برخی از مشخصات اجزای مختلف برنامه *F* عبارتند از:

۱. برنامه *F* شامل *runtime group*های نوع *F* و چارت‌های تعریف شده به آنها می‌باشد.
۲. چارت‌ها شامل بلوک‌های*F* با تنظیمات پارامتری و اتصالات بین هم می‌باشد.
۳. برای اجرای *runtime group*های نوع *F* توسط *CPU*، آنها به یک یا چند *OB* وقفه سیکلیک اضافه می‌شوند.
٤. *Runtime group*های نوع *F* می‌تواند در *F-shutdown group*ها (*F-SG*) ترکیب شود.
۵. *OB* وقفه سیکلیک همچنین می‌تواند شامل گروه‌های زمان اجرای استاندارد باشد.
٦. بلوک‌های *F* در کتابخانه *S7 F* سیستم *F* به رنگ زرد می‌باشد.
۷. چارت‌های *CFC* و گروه‌های زمان اجرای *F* به رنگ زرد ظاهر می‌شود.

۸. چارت‌های *CFC* و گروه‌های زمان اجرا نوع *F* با بلوک‌های *F* به رنگ زرد ظاهر می‌شوند و به منظور تمایز آنها از چارت‌ها و گروه‌های زمان اجرا مربوط به برنامه لاجیک استاندارد با حرف *F* مشخص می‌شوند.

۵-۱-۱- قوانین ساختار برنامه *F* (*Rules governing program structure*)

موقع طراحی و آماده‌سازی پیش نویس یک برنامه *S7* برای یک سیستم *S7 F/FH* بایستی به سوالات زیر در مقایسه برنامه *F* با برنامه استاندارد پاسخ داده شود.

- چه بخش‌هایی از برنامه *S7* بایستی به صورت *F* باشد.
- به چه پاسخ زمانی‌هایی نیاز دارید. لذا بر این اساس بایستی برنامه *S7* را به *OB 3x*های مختلف تقسیم کنید.
- تنها فقط می‌توان *F-Shutdown groups*های حاوی بلاک‌های *F* را به *OB 3X* اختصاص داد(*OB 30* تا *OB 38*) به عبارت دیگر گروه‌های زمان اجرا نوع *F* دارای بلوک‌های *F* را تنها می‌توان به *OB*های وقفه *30* تا *38* اختصاص داد.
- یک برنامه *S7* می‌تواند شامل هر دو نوع بلوک *F* و بلوک استاندارد باشد؛ ولی نیاز است که این چارت‌ها در *Runtime group*های جداگانه وارد شوند. به عبارت دیگر بلاک‌های *F* و بلاک‌های استاندارد بایستی در چارت‌های جداگانه برنامه‌نویسی شوند. همچنین از این چارت‌ها نمی‌توان به عنوان بلاک تایپ *F* استفاده کرد.
- داده *F-I/O* تنها از طریق درایور *F*-کانال قابل دسترسی است. یعنی دسترسی به ماژول‌های سیگنال *F* تنها از طریق درایور کانال *F* (*F_CH_xx*) مجاز است.
- برای اتصال بلوک‌های *F* به هم از نوع داده بخصوص *F* (*F data types*) استفاده می‌شود. فرمت داده *F* قادر می‌سازد که خطاهای داده و آدرس شناسایی شود.

۵-۱-۲- اجرای بلاک‌های *F* (*Run sequence of F-blocks*)

سیستم اجرایی *S7* ، برای اجرای برنامه کاربر از روش ساختار یافته استفاده می‌کند به این طریق که رابط اجرایی برنامه کاربر و سیستم عامل *CPU* بلاک‌هایی بنام *OB* می‌باشد که دارای سیکل‌های اجرایی متفاوت و با اولویت‌های اجراییی از قبل تعریف شده می‌باشد. به این معنی که هر یک از این بلاک‌های *OB* به صورت سیکلیک با اینتروال‌های زمانی تعریف شده در پنجره پراپرتی *CPU* فراخوانی شده و برنامه داخل آنها اجرا می‌شود به عنوان مثال *OB35* توسط سیستم عامل *CPU* هر ۱۰۰ میلی ثانیه فراخوانی می‌شود. زیمنس از بلاک‌های *OB* تحت عنوان سطوح اجرایی *CPU* (*runtime level*) یاد می‌کند.

همچنین از آنجایی که برنامه کاربر توسط برنامه‌نویسی چارت‌های *CFC* صورت می‌گیرد، سیستم *S7* برای رعایت ترتیب زمانی اجرای بلاک‌های واقع در چارت‌ها، آنها (چارت‌های *CFC*) را به گروه‌های اجرایی

تحت عنوان *Runtime Group* تقسیم می‌کند. لذا زمانی که یک چارت جدید *CFC* در *PCS 7* ایجاد می‌شود، سیستم به طور خودکار یک *Runtime Group* به نام *CFC*، که بلوک‌های برنامه در آن قرار داده شده است را ایجاد می‌کند.

به همان روش برنامه استاندارد، موقع درج بلاک‌های *F* در چارت‌های *CFC* یک *Runtime Group* نوع *F* ایجاد می‌شود. یعنی به محض قرار دادن یک بلاک *F* در یک چارت *CFC* ، آن چارت تبدیل به چارت *F* می‌شود. لذا یک *Runtime group* تنها زمانی *F-runtime group* می‌شود که در آن بلوک *F* فراخوانی شود.

مشخصه *Run sequence* در چارت‌های *F* نیز به همان روش برنامه استاندارد، در ویرایشگر *CFC* تعریف می‌شود. تغییر در *Run sequence* همچنین موجب تغییر در *collective signature* می‌شود.

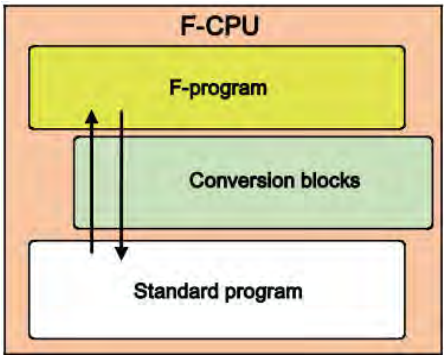
نکته: در طول برنامه‌نویسی برنامه ایمنی، بلوک‌های *F* را نمی‌توان به طور مستقیم در بلاک‌های *OB* وقفه سیکلیک درج کرد.

۵-۱-۳- تبادل داده بین لاجیک *F* و برنامه استاندارد(*Data exchange*)

برنامه استاندارد و برنامه *F* از فرمت‌های داده متفاوت استفاده می‌کند. بر این اساس، بلوک‌های تبدیل خاص برای تبادل داده بین آنها استفاده می‌شود.

اگر نیاز باشد که برنامه استاندارد، داده‌هایی از برنامه *F* را پردازش کند(مثال برای مانیتورینگ در *OS*)، از یک بلوک *F_Fdatatype_data* برای تبدیل نوع داده استفاده خواهد شد.

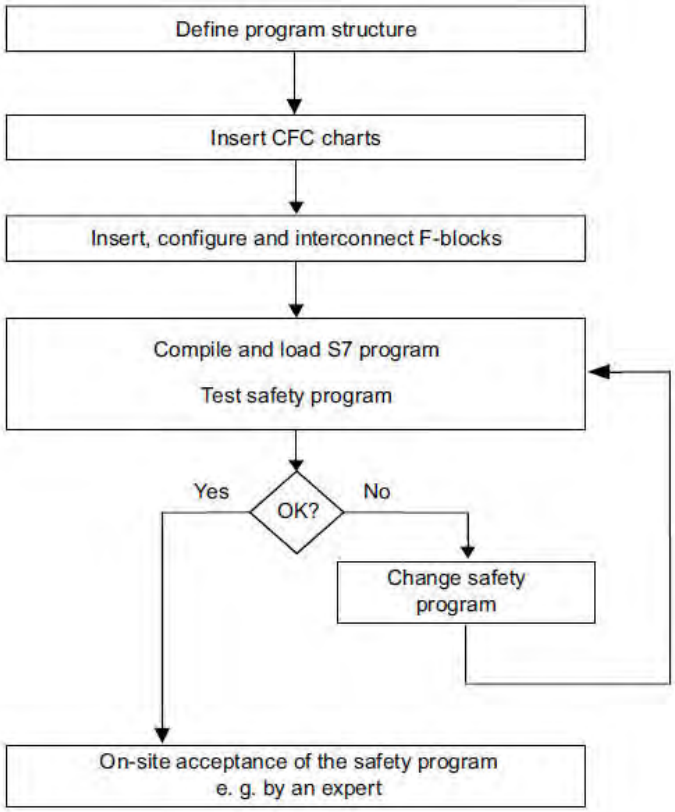
بلاک *F* که برای تبدیل نوع داده *F* به نوع داده استاندارد استفاده می‌شود، بایستی در گروه اجرایی استاندارد فراخوانی شود(*standard runtime group*)



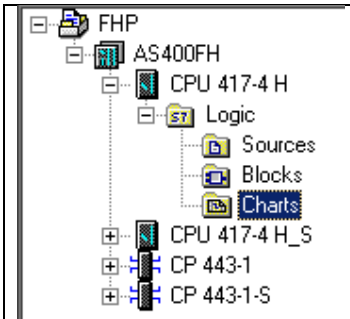
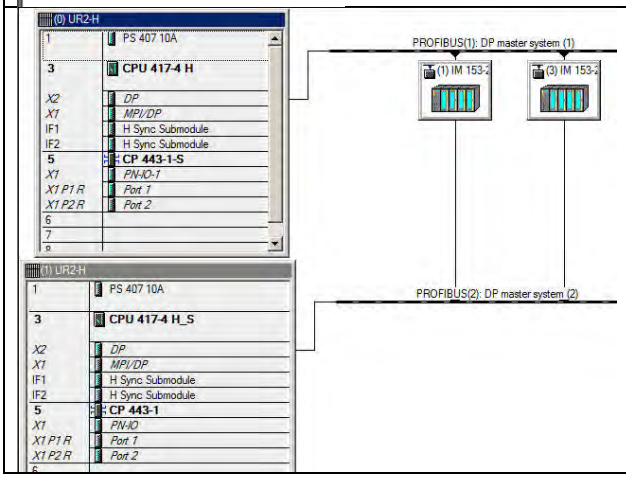
شکل ۵-۳-تبادل داده بین لاجیک *F* و برنامه استاندارد توسط بلاک‌های تبدیل صورت می‌گیرد

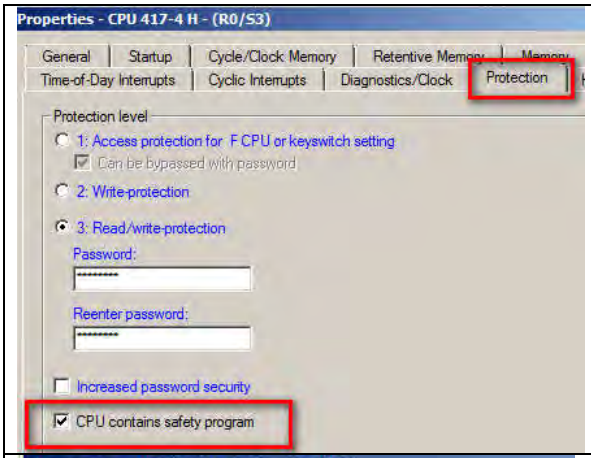
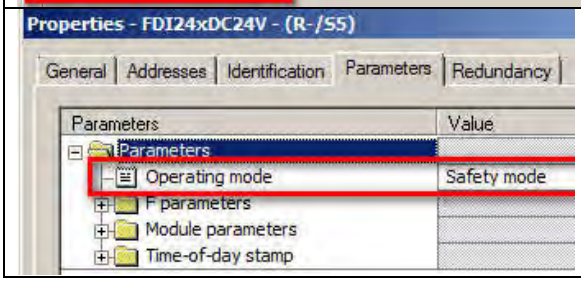
۵-۱-۴- روند پایه برای ایجاد برنامه ایمنی

شکل ٤-۵ یک روند پایه برای ایجاد یک برنامه *F* را به تصویر کشیده است. قبل از شروع به ایجاد لاجیک *F* کارهای زیر بایستی صورت گیرد.



شکل ٥-٤- روند پایه برای ایجاد برنامه *F*

	۱. یک پروژه با نام دلخواه را در *Simatic Manager* ایجاد کنید.
	۲. سخت‌افزار*F-CPU* و *F-IO* در پنجره *HWConfig* پیکربندی شده باشد. به این معنی که *CPU*ای که قادر به اجرای برنامه *F* باشد (مانند *CPU 412-3H, CPU 414-4H, or CPU 417-4H*) و اجزای مورد نیاز یعنی رک، ماژول تغذیه، ماژول *CP* و غیره را پیکربندی شود. *ET200M > IM153-2 > 6ES7 153-2BA10-0XB0* *F-AI Module : 6ES7 336-1HE00-0AB0*

۳. در سربرگ *Protection* از پنجره پراپرتی *CPU* گزینه *CPU Contains safety program* را تیک زده و برای محافظت از برنامه *CPU* یک رمز تعریف کنید.	
٤. در پنجره پراپرتی ماژول سیگنال *F*، در قسمت *Parameters* مد کاری را به *Safety mode* تنظیم کنید. در غیر این صورت موقع کامپایل برنامه *F* خطا ظاهر خواهد شد.	

۵-۱-۵- زمان مانیتورینگ سیکل *F* (*F-cycle monitoring*)

فانکشن بلاک *F_CYC_CO*

هنگام استفاده از بسته *S7 F system*، این بلوک کنترلی *F* یعنی "*F_CYC_CO*" به طور خودکار در اولین کامپایل چارت‌های *CFC* در داخل یک چارت *@F_CycCo-OB3x* (مانند *@F_CycCo-OB35*) قرار داده شده و پیکربندی می‌شود. بلوک "*F_CYC_CO*" دارای نام سطح اجرایی *CPU* (مانند *OB35*) مربوطه می‌باشد.

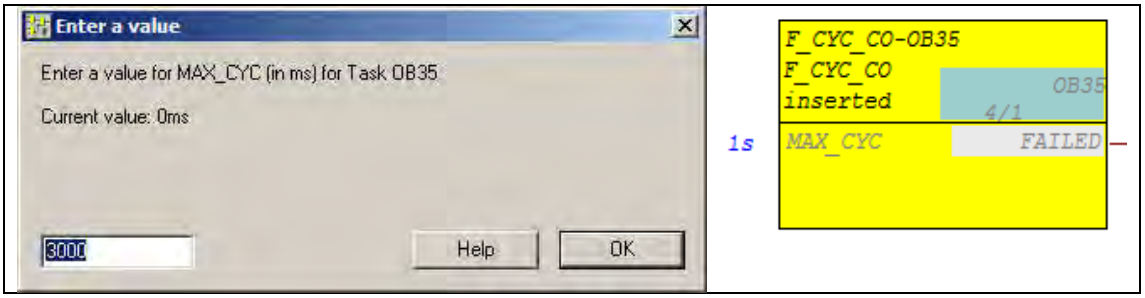
همچنین به صورت اتوماتیک در یک گروه *F-Runtime group* ایجاد شده به صورت خودکار، با شناسه «*@F_*» در طول کامپایل برنامه *S7* به منظور ایجاد برنامه ایمنی اجرایی از برنامه ایمنی کاربر درج می‌شود.

ماژول *F-CPU*، زمان زمان *F-Cycle* را برای هر بار اجرای *OB 3x* که حاوی گروه‌های *F-Runtime* می‌باشد را مانیتور می‌کند.

وقتی برای اولین بار در برنامه *S7-F*، چارت‌های *CFC* کامپایل می‌شود، پنجره‌ای برای ورود یک مقدار برای زمان سیکل حداکثر (*MAX_CYC*) که می‌تواند بین دو فراخوانی *OB* صرف شود، نمایان می‌شود.

اگر بعد از اولین کامپایل برنامه *s7* نیاز باشد که حداکثر زمان *F-Cycle* را تغییر دهید. برای تغییر مقدار *F-Cycle time* بایستی پارامتر *MAX_CYC*، را در بلوک *F_CYC_CO-OB3x* در چارت *@F_CycCo-OB3x* تغیر دهید.

در صورتی که در کامپایل‌های بعدی، اگر مقدار پارامتر "*MAX_CYC*" نامعتبر باشد، یک باکس محاوره‌ای باز می‌شود که در آن می‌توان مقدار معتبر را وارد کرد. مقدار پیش فرض داده شده برای این پارامتر برابر ۳۰۰۰ میلی‌ثانیه (حداکثر زمان مانیتورینگ) است.



دلایل نیاز به تعریف زمان *F-runtime-group-time*

یک دلیل برای تعریف زمان *F-runtime-group* این است که مجبور نیستی که برنامه *F* را در هر سیکل *OB3x* فراخوانی کنید. نیاز به این که هر چند وقت یا با چه سرعتی برنامه *F* باید فراخوانی شود، فقط به فرآیند بستگی دارد. اگر هیچ الزامات خاصی در مورد زمان اجرا ندارید، می‌توانید به عنوان مثال سیکل زمانی *OB35* را برای *F-runtime-group* دو برابر فرض کنید.

۵–۱–۶– قرار دادن گروه‌های *F-Runtime* در چارت *CFC*

۱. موقع ترکیب گروه‌های *F-Runtime* و *standard runtime* در یک *OB* سیکلیک اینتراپت، به منظور جلوگیری از طولانی‌تر شدن زمان اجرای گروه *F-Shutdown* و تاثیرگذاری در زمان پاسخ، گروه‌های *standard runtime* را قبل از گروه‌های *F-Runtime* اجرا کنید.

۲. برای یک گروه *F-Runtime* بایستی تنظیم پیش فرض پارامترهای *Reduction ratio* و *Phase shift* (*Reduction ratio = 1, Phase shift = 0*) حفظ شود. این مقادیر نبایستی تغیر داده شود.

۳. گروه‌های *F-Runtime* که به صورت خودکار ایجاد شده‌اند نباید شیفت داده شوند و تغییرات نباید در داخل گروه *F-Runtime* انجام شود.

۵–۱–۷– گروه‌های *F-Shutdown* برنامه ایمنی

به محض این که بلوک‌های *F* در ویرایشگر *CFC* برای اولین بار قرار داده می‌شوند، همه گروه‌های *F Runtime* در هر یک *OB 3x*ها یک گروه *F-Shutdown* شکل می‌دهند.

نکته : کاربر مجاز به اتصال بلوک‌های *F* متعلق به گروه‌های *F-Shutdown* به هم نمی‌باشد.

۵–۲– درج بلاک‌های *F* و اتصال آنها به همدیگر

بلوک‌های موجود در پوشه *F-Control Blocks* به طور خودکار در زمان کامپایل برنامه *S7* درج می‌شوند و کاربر مجاز به درج این بلوک‌ها به صورت دستی نمی‌باشد. همچنین کاربر مجاز به درج یک بلاک *F-Block* *instance* در چند گروه *F-Runtime* نمی‌باشد.

۵-۳- پروسه *F startup & (re)start Protection*

در صورت توقف *CPU* اجرای لاجیک *Safety* نیز متوقف می‌شود. لذا به این نوع توقف *F-Stop* گفته می‌شود.

۵-۳-۱- راه اندازی *F* (*F-Startup*)

در سیستم‌های *S7-F* تمایزی بین راه‌اندازی مجدد سرد و راه‌اندازی مجدد گرم (*Warm restart*) برای ماژول *CPU* وجود ندارد. ولی بلاک‌های *F_CHG_BO*، *F_CHG_R* (بخشی از تابع نوشتن داده *Safety*) و *F_MOV_R* در این مورد استثناء می‌باشند.

در هر دو نوع راه‌اندازی مجدد سرد و گرم *CPU*، یک راه اندازی *F* صورت می‌گیرد. لذا در صورت وقوع راه‌اندازی *F*، اجرای برنامه ایمنی به طور خودکار با مقادیر اولیه شروع می شود.راه اندازی *F* در موارد زیر انجام می‌شود.

- اگر پس از توقف *CPU*، *F-CPU* راه اندازی مجدد *Warm* شود.
- پس از *F-STOP*، اگر بلوک *F_SHUTDN* لبه مثبت در ورودی *RESTART* تشخیص دهد.

پس از *partial shutdown* برنامه ایمنی، تنها گروه‌های *F-shutdown* که در *F-STOP* سهیم بودند، یک عمل *F startup* را انجام می‌دهند. گروه‌های *F-shutdown* دارای خطا در *F-STOP* باقی می‌ماند.

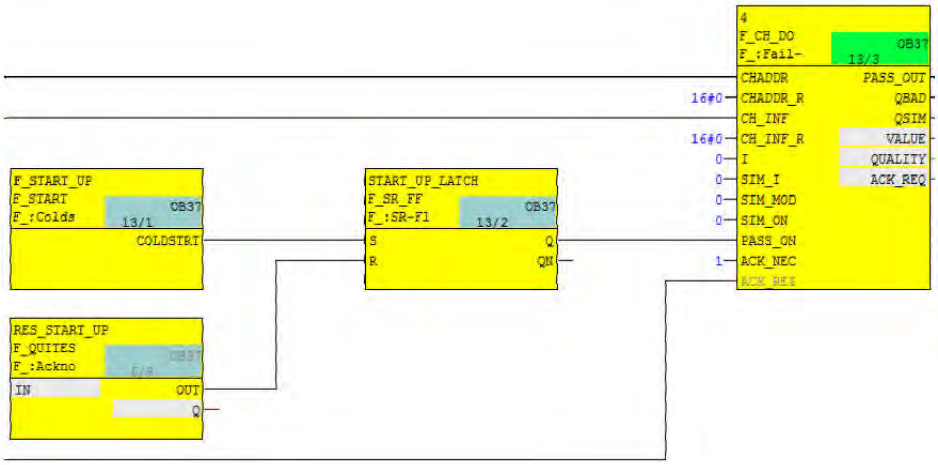
۵-۳-۲- *(Re)Start Protection*

اگر فرایند اجازه نمی‌دهد که برنامه ایمنی (*safety program*) به صورت خودکار با مقادیر اولیه راه‌اندازی (*start up*) شود، نیاز خواهید داشت که یک پاسخ به *F-startup* را برنامه‌نویسی کند.

بلاک *F_START* در برنامه ایمنی برای انجام *F-startup* با مقادیر اولیه استفاده می‌شود. پارامتر خروجی *COLDSTRT* نشان می‌دهد که یک *F-startup* تریگر شده است.

در مثال زیر، موقعی که برنامه *F* راه‌اندازی (*started up*) می‌شود (یعنی ماژول *CPU* از *Stop* به *Run* می‌رود)، خروجی فلیپ فلاپ یک می‌شود. این خروجی به ورودی *Pass_ON* بلاک‌های درایور متصل می‌شود. به طوری که تمام درایورهای خروجی را غیرفعال (*passivate*) می‌کند. همین که تمام شرایط فرآیند برآورده شد، اپراتور می‌تواند با استفاده از بلاک *F_QUITES* (*F acknowledgment*) و با ریست کردن فلیپ فلاپ سیگنال‌های خروجی را از حالت *passivate* خارج کند.

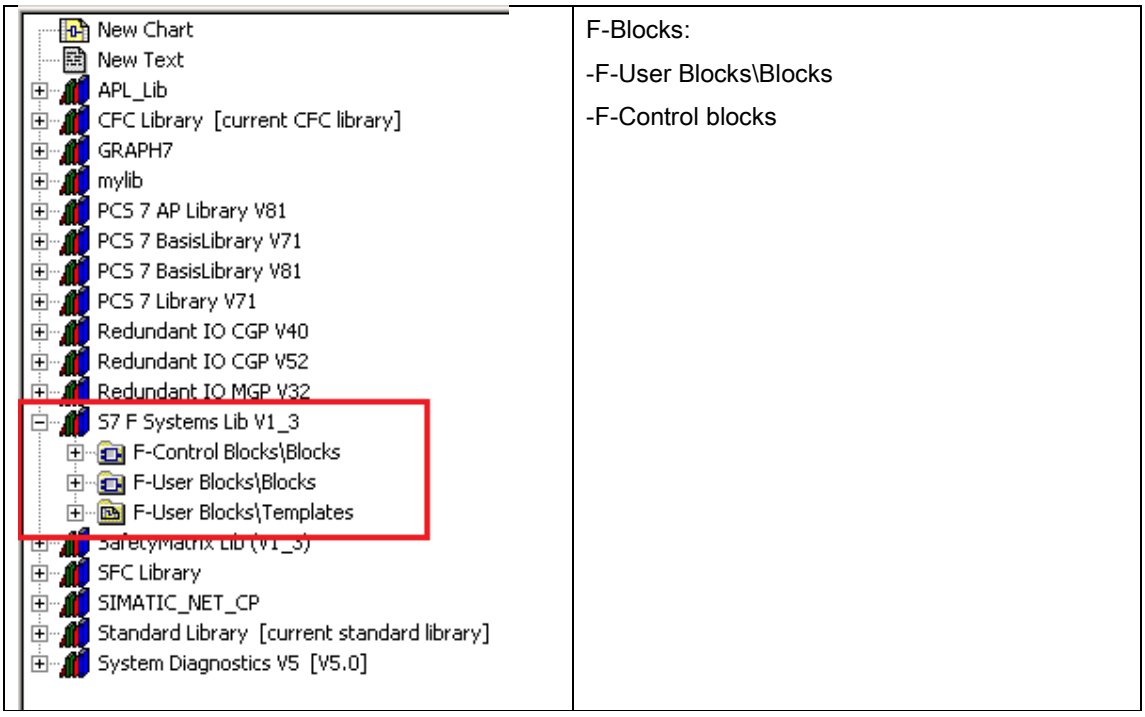
در این مثال در هنگام راه‌اندازی *CPU* خروجی *COLDSTRT* بلاک *F_START* یک شده و فلیپ فلاپ را *Set* می‌کند. و در نهایت موجب غیرفعال شدن بلاک‌های درایور متصل به خروجی فلیپ فلاپ می‌شود.


4
F_CH_DO
F_:Fail-
OB37
13/3
CHADDR
CHADDR_R
CH_INF
CH_INF_R
I
SIM_I
SIM_MOD
SIM_ON
PASS_ON
ACK_NEC
ACK_REI
PASS_OUT
QBAD
QSIM
VALUE
QUALITY
ACK_REQ
16#0
0
1
F_START_UP
F_START
F_:Colds
OB37
13/1
COLDSTRT
START_UP_LATCH
F_SR_FF
F_:SR-Fl
OB37
13/2
S
R
Q
QN
RES_START_UP
F_QUITES
F_:Ackno
OB37
IN
OUT
Q

فصل – ۶ – کتابخانه سیستم F

۶–۱– مقدمه

پس از نصب بسته *F-System* کتابخانه *F-library V1.3* در کنار کتابخانه‌های دیگر در محیط *CFC* ظاهر می‌گردد. توابع این کتابخانه بر حسب تکنولوژی در پوشه‌های مختلف دسته‌بندی‌شده است.



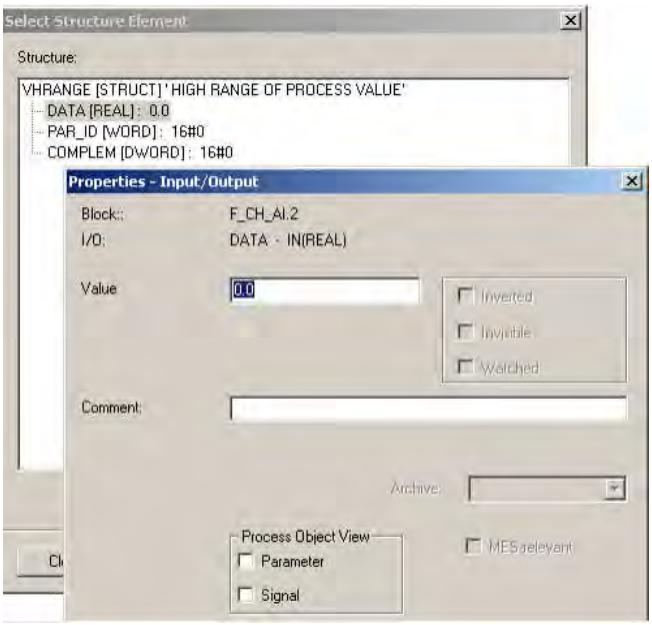
نکته: اسم کتابخانه *F* نبایستی تغییر داده شود. همچنین نبایستی شماره بلاک‌های *F* تغییر داده شود.

نکته : بلاک‌های *F* بایستی پیش از کامپایل در داخل *CFC* به یک آدرس درست و معتبر متصل شوند، در غیر این صورت خطا ظاهر می‌شود.

همچنین پیش از کامپایل بایستی در *HW Config* پارامترهای ماژول‌های *F* به مقادیر مناسبی تنظیم شوند. به عنوان مثال مد کاری ماژول به *Safety mode* تنظیم شود. در غیر این صورت خطا خواهد داد.

۶–۱–۱– نوع داده *F* (*F-Data types*)

در کتابخانه *S7 F-system* برای پارامترهای ورودی/خروجی بلاک‌های *F* از نوع‌داده‌های بخصوص در عملیات محاسباتی و منطقی استفاده می‌شود. نوع‌داده‌های بخصوص *F* در یک قالب داده ایمنی برای واسط‌های بلوک‌های *fail safe* استفاده می‌شود. قالب داده *F* (safety data format) برای نمایش داده و آدرس‌دهی خطاها استفاده می‌شود. از نظر برنامه نویسی نوع داده‌های *F* به صورت *structures* پیاده‌سازی می‌شوند به طوری که در آن تنها جزء DATA به کاربر مربوطه می‌شود.

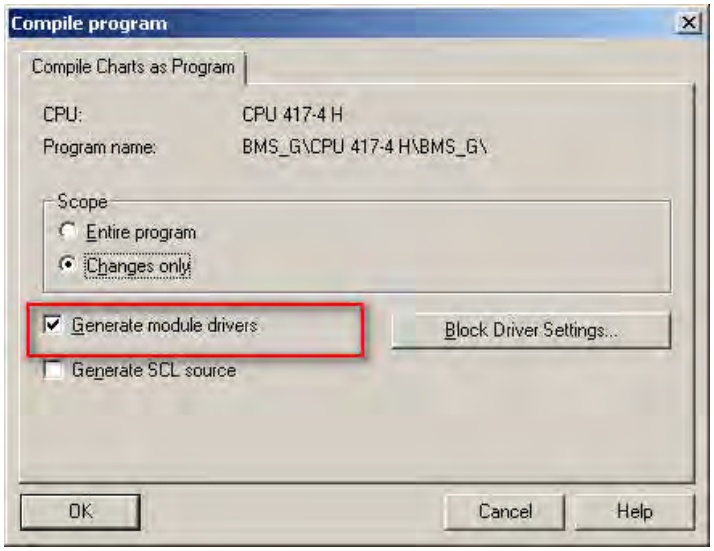


شکل ٦-١- استفاده از نوع داده *Structure* برای پارامترهای واسط بلاک

۶-۱-۲- دسترسی به ورودی/خروجی‌های نوع *F* (*F-I/O access*)

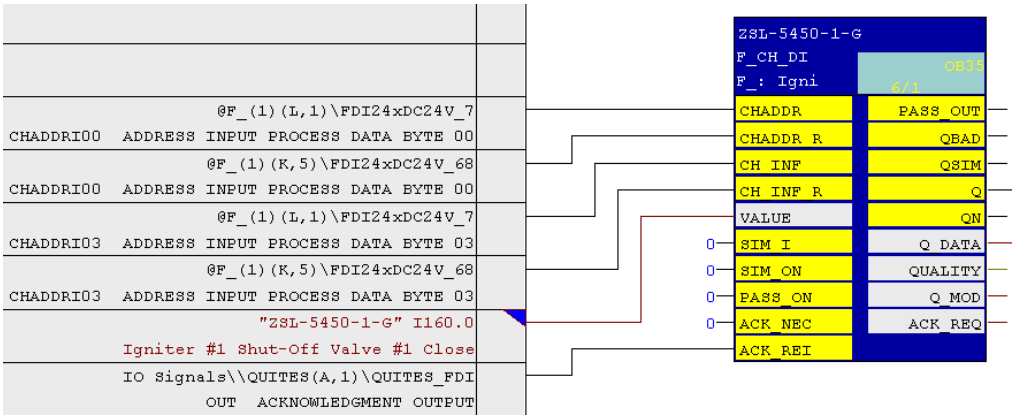
در سیستم‌های نوع *S7-FH* ، دسترسی به داده ماژول‌های سیگنال *F*، به جای دسترسی از طریق حافظه تصویر فرآیند ورودی (*Process Image*) از طریق بلاک‌های درایور*F* صورت می‌گیرد. برای این منظور دو نوع درایور یعنی درایور ماژول سیگنال و درایور کانال سیگنال برای دسترسی به مقادیر *F-I/O* استفاده می‌شود. به این معنی که به ازای هر ماژول *F I/O* یک درایور ماژول *F* و به ازای هر کانال ورودی/خروجی*F* از یک بلاک درایور کانال *F* استفاده می‌شود.

- درایور کانال *F* (*F channel driver*): به عنوان مثال *F_CH_xx* برای هر کانال ورودی/خروجی*F*. در کانال‌های سیگنال ریداندانت تنها از یک درایور کانال سیگنال استفاده می‌شود.
- درایور ماژول (*module driver*): درایور ماژول *F* مسئول ارتباطات *PROFIsafe* بین برنامه *F* و ماژول *F-I/O* می‌باشد. به ازای هر ماژول *F-I/O* یک درایور ماژول *F* (*F-Module driver*) با فعال کردن تیک گزینه *Generator Module Drivers* در پنجره تنظیم کامپایل *CFC* به طور خودکار در چارت‌های *CFC* با پیشوند @ قرار داده شده و به کانال‌های مربوطه در برنامه *F* متصل می‌شود.



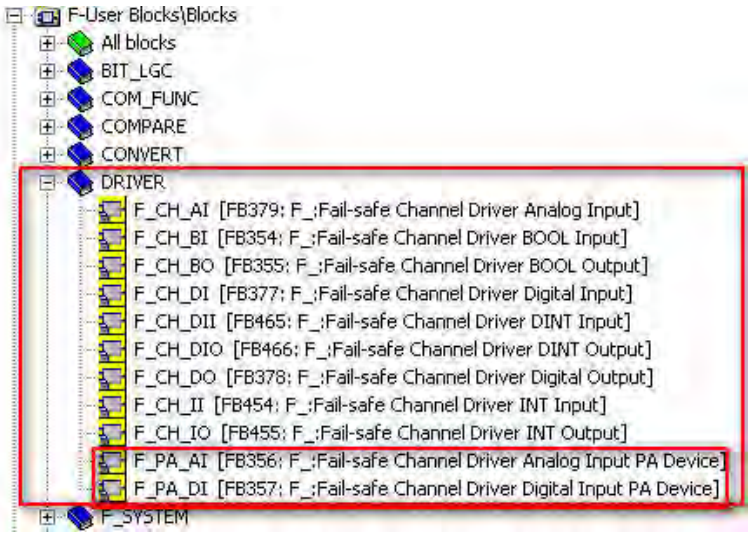
شکل ۶-۲- تنظیم *Generator Module Drivers* در پنجره کامپایل *CFC*

پس از اتمام کامپایل چارت‌های *CFC* مطابق شکل زیر دیده می‌شود که ورودی‌های *CHADDR* به طور خودکار به بلاک‌های درایور ماژول متصل شده‌اند.



۶-۲- مجموعه بلاک‌های دایور کانال *F* (*F-Channel drivers*)

بلاک‌های دایور کانال، ارتباط برنامه *F* را با یکی از کانال‌های یک ماژول *F-I/O* فراهم کرده و عمل پردازش سیگنال را انجام می‌دهند. بسته به نوع کانال (دیجیتال،آنالوگ، پروفی‌باس) درایورهای مختلف کانال *F* وجود دارد. شکل زیر بلاک‌های دایور موجود در کتابخانه *F* را برای کانال‌های دیجیتال و آنالوگ ورودی/خروجی نشان می‌دهد.

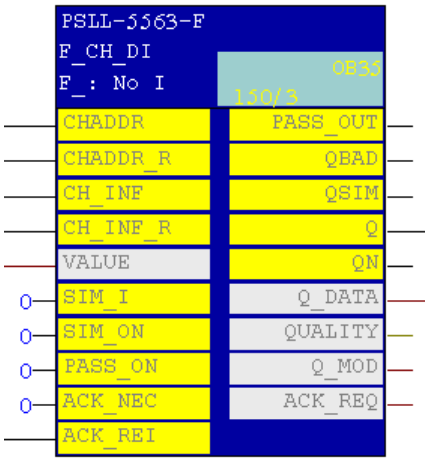


شکل ٦-٣- بلاک‌های دایور کانال‌های ورودی/خروجی *F*

٦-٢-١- بلاک درایور کانال دیجیتال ورودی (*F_CH_DI*)

Fail-safe channel driver for digital inputs of F-I/O (except fail-safe DP standard slaves)

این بلاک برای دریافت داده از یک کانال دیجیتال *F* استفاده می‌شود.



شکل ٦-٤- بلاک درایور کانال دیجیتال ورودی معمولی(نه برای پروفیباس *DP*)

ورودی/خروجی	نوع داده	توضیح
ADR_CODE	*DWORD*	
VALUE	*BOOL*	آدرس سیگنال کانال ورودی دیجیتال کارت به این ورودی متصل می‌شود. به عنوان مثال ورودی دیجیتال با سیمبل *ZSL-5450-1-G*. نوع داده این پایه به جای *F_BOOL* از نوع *BOOL* معمولی می‌باشد.

SIM_I	*F_BOOL*	در صورتی که بخواهیم سیگنال این کانال را در نبود آن شبیه سازی کنیم، مقدار۱ به این پایه داده می‌شود.
SIM_ON	*F_BOOL*	امکان شبیه‌سازی سیگنال را برای کانال فعال می‌کند.
PASS_ON	*F_BOOL*	این پایه برای غیرفعال کردن کانال از طریق برنامه استفاده می‌شود. با یک کردن این پایه، کانال مربوطه غیرفعال می‌گردد. با یک شدن این ورودی، خروجی *QBAD* یک می‌شود. *1 = Activate Passivation*
ACK_NEC	*F_BOOL*	در صورت یک شدن این پایه، *Ack* کردن خطای *Passivation* ضروری می‌باشد. *1 = Acknowledgement Necessary*
ACK_REI	*F_BOOL*	*Acknowledgment Reintegration* با یک کردن این ورودی عمل *Depassivate* کانال‌ها صورت می‌گیرد.
ACK_REQ =1		یک شدن این خروجی به این معنی است که خطا از بین رفته است و تصدیق کاربر در ورودی *ACK_REI* برای فعال‌سازی مجدد کانال (*reintegration*) لازم است.
QUALITY	*BYTE*	کد کیفیت سیگنال ورودی را نشان می‌دهد. اگر مقدار ورودی دیجیتال دریافت شده از *F-I/O* معتبر باشد، مقدار دریافت شده از کانال با کد کیفیت *16#80* در خروجی *QUALITY* در خروجی *Q* قرار می‌گیرد.
QN	*F_BOOL*	*Negated Process Value* *Not* شده خروجی *Q* می‌باشد.
Q_DATA	*BOOL*	*PROCESS VALUE DATA (for monitoring)* مقدار دریافت شده ار کانال دیجیتال ورودی درصورت معتبر بودن در سه پایه *Q*، *QN* (به صورت معکوس) و *Q_DATA* ظاهر می‌شود. از آنجایی که نوع داده این پایه خروجی از نوع *BOOL* می‌باشد، می‌توان به طور مستقیم به یک بلاک مانیتورینگ دیجیتال متصل کرد.
DISCF	*BOOL*	*Discrepancy Error Module* این خروجی زمانی معنی پیدا می‌کند. که کانال‌های ماژول به صورت ریداندانت پیکربندی شده باشند. در صورت وقوع خطای *Discrepancy* این خروجی یک می‌شود.

- **فعال‌سازی مجدد کانال پس از حذف خطا (*Reintegration after error elimination*)**

بعد از این که خطا رفع شد، قرائت مقدار ورودی دیجیتال دریافت شده از کانال *F-I/O* می‌تواند به طور خودکار و یا پس از *Ack* کردن توسط کاربر فعال شود.

اگر ورودی *ACK_NEC =1* باشد، عمل تصدیق کاربر (*user acknowledgement*) از طریق ورودی *ACK_REI*، بعد از رفع خطا ضروری است و اگر *ACK_NEC = 0* باشد، فعال‌سازی (*reintegration*) کانال

به صورت خودکار (*automatic reintegration*) انجام می‌گیرد. برای حالتی که *PASS_ON = 1* است و یا راه‌اندازی *F* (*F-Startup*) پس از یک *CPU-STOP* به تصدیق کاربر نیاز نیست.

Fail-safe value

در صورت بروز خطا مقدار صفر در خروجی *Q* بلاک قرار می‌گیرد که *Fail-safe value* نامیده می‌شود. مقدار صفر در شرایط زیر در خروجی *Q* قرار می‌گیرد.

- مقدار دیجیتال کانال به دلیل خطای ارتباطی (*PROFIsafe*) غیر معتبر باشد.
- مقدار دیجیتال کانال به دلیل فالت در ماژول یا کانال (مانند قطعی سیم) غیر معتبر باشد.
- یک پروسه *F-Startup* در حال اجرا باشد.
- یک حالت *Passivation* با *PASS_ON = 1* وجود داشته باشد.

- **آنالیز** اختلاف بین دو کانال ریداندانت (***Discrepancy analysis***)

در خصوص ماژول‌های *F-I/O* پیکربندی شده به صورت افزونه، اگر زمان اختلاف (*discrepancy time*) تنظیم شده در *HWConfig* مخالف صفر باشد، بلوک *F* تجزیه و تحلیل اختلاف مقدار بین دو کانال افزونه را انجام می‌دهد.

اگر اختلاف بین کانال‌های ورودی دیجیتال ریداندانت بیش از زمان اختلاف تعریف شده باشد، یک خطای اختلاف تشخیص داده می‌شود. در این صورت بلوک *F* خروجی *DISCF* را یک می‌کند. اگر کانال ورودی دیجیتال که در ورودی *VALUE* آدرس‌دهی شده است، سیگنال صفر را اعمال کند، بلاک *F* خروجی *DISCF* را یک می‌کند. همچنین اگر مقدار کانال ریداندانت نیز صفر باشد، بلوک *F* خروجی *DISCF_R* را یک می‌کند خروجی‌های *DISCF / DISCF_R* به محض رفع شدن اختلاف ریست می‌شوند.

به عنوان مثال، عمل تجزیه و تحلیل اختلاف، تشخیص سنسور معیوب را فراهم می‌سازد. به این دلیل که فرض می‌کند که وقتی سنسور *F* مقدار صفر را تولید می‌کند، معیوب می‌باشد. این کار می‌تواند در دسترس بودن سیستم را افزایش دهد.

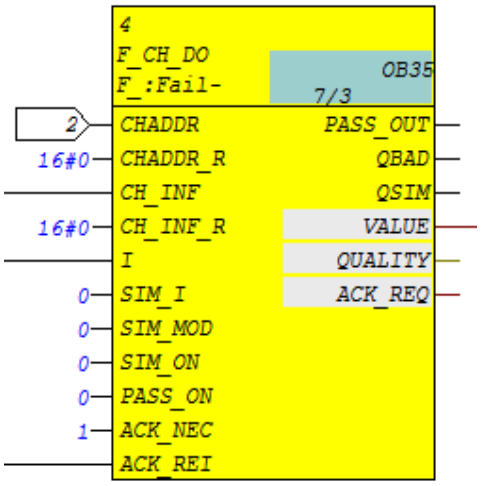
نکته– خطاهای اختلاف هیچ تاثیری بر روی خروجی‌های *Q*،*QBAD* یا *PASS_OUT* ندارد. خروجی‌های *DISCF/DISCF_R* را که از نوع *non-fail-safe* هستند را می‌توان توسط یک *OS* (مانیتورینگ) برای فعالیت هایی مانند ارزیابی در برنامه استاندارد قرائت کرد.

رفتار درایور کانال در زمان *F-STOP*

در صورت وقوع یک *F-STOP*، کد کیفیت *16#00* در خروجی *QUALITY* و *QBAD.DATA = 1* می‌شود. دیگر متغیرهای کانال فریز (*frozen*) می‌شوند.

۶-۲-۲- بلاک درایور کانال دیجیتال خروجی (*F_CH_DO*)

این بلاک برای نوشتن مقادیر دیجیتال *Fail safe* در کانال خروجی *F-I/O* استفاده می‌شود.



شکل ٦-٥- نمایی از بلاک درایور کانال دیجیتال خروجی (*F_CH_DO*)

پایه ورودی/خروجی	نوع داده	توضیح
I	*F_BOOL*	بیت خروجی (*Process Value*) که قرار است به یک خروجی منتقل شود، به این پایه داده می‌شود.
SIM_MOD	*F_BOOL*	*1=Simulation Has Priority* با یک شدن این پایه، اولویت با شبیه‌سازی خواهد بود.
SIM_I	*F_BOOL*	در صورتی که بخواهیم سیگنال این کانال را در نبود آن شبیه سازی کنیم، مقدار ۰، ۱ به این پایه داده می‌شود.
SIM_ON	*F_BOOL*	شبیه سازی سیگنال را برای کانال فعال می‌کند.
PASS_ON	*F_BOOL*	*1 = Activate Passivation* این پایه برای غیر فعال کردن کانال از طریق برنامه استفاده می‌شود. با یک کردن این پایه، کانال مربوطه غیرفعال می‌گردد.
ACK_NEC	*F_BOOL*	در صورت یک شدن این پایه، *Ack* کردن خطای *Passivation* ضروری می‌باشد. *1 = Acknowledgement Necessary*
ACK_REI	*F_BOOL*	*Acknowledgment reintegration*
ACK_REQ =1		به این معنی است که خطا از بین رفته است و تصدیق کاربر در ورودی *ACK_REI* برای فعال‌سازی مجدد کانال (*Reintegration*) لازم است.
QUALITY Quality Code Of Process Value	*BYTE*	کد کیفیت سیگنال ورودی را نشان می‌دهد. اگر مقدار ورودی دیجیتال دریافت شده معتبر باشد، کد کیفیت *16#80* در خروجی *QUALITY* قرار می‌گیرد.
QSIM	*F_BOOL*	*1 = simulation active*

		فعال بودن مد شبیه‌سازی در کانال را نشان می‌دهد.
QBAD	*F_BOOL*	*1 = process value invalid*
Normal value		در صورت عدم وجود خطا در کانال، داده دریافت شده از *I* با کد کیفیت *16#80* در ماژول *F-I/O* نوشته می‌شود.
PASS_OUT	*F_BOOL*	*1 = Passivation Because Of Error* در صورت وقوع *Passivation* در کانال به دلیل خطا این خروجی یک می‌شود.

شبیه‌سازی

۱. اگر ورودی *SIM_ON=1* و *SIM_MOD=0* باشد، مقدار *SIM_I* در کانال *F-I/O* نوشته شده و به شرطی که هیچ خطای ارتباطی (*PROFIsafe*) و هیچ فالتی در کانال یا ماژول وجود نداشته باشد، در خروجی *VALUE* ظاهر می‌شود.

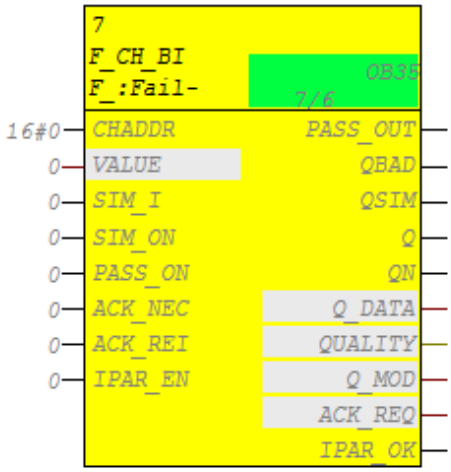
۲. اگر ورودی *SIM_ON=1* و *SIM_MOD=1* باشد، در صورت وقوع خطای ارتباطی (*PROFIsafe*) و فالت در کانال یا ماژول، مقدار *SIM_I* در خروجی *VALUE* ظاهر می‌شود.

۳. در هر دو حالت بالا خروجی *QUALITY* به مقدار *16#60* تنظیم شده و *QSIM = 1* می‌شود.

۶-۲-۳- بلاک درایور برای ورودی باینری از شبکه *DP* (*F_CH_BI*)

F_CH_BI: F-Channel driver for inputs of data type BOOL of fail-safe DP standard slaves and fail-safe standard I/O devices

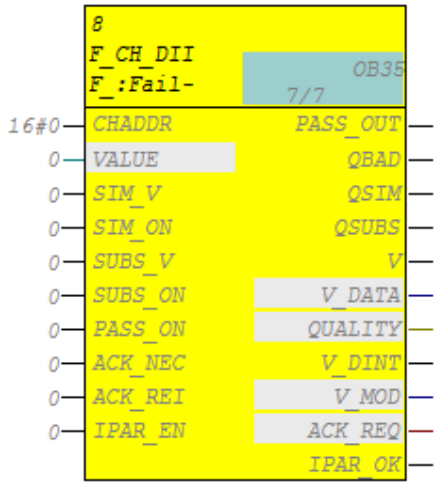
این بلاک برای پردازش یک سیگنال ورودی باینری (*BOOL*) از *Slave*های *fail-safe DP standard* و وسایل *fail-safe standard I/O* استفاده می‌شود. این بلوک *F*، به صورت سیکلیک مقدار ورودی *BOOl* را از یک *Slave* نوع *fail-safe DP* آدرس‌دهی شده در ورودی *VALUE* را می‌خواند مقدار ورودی از طریق درایور ماژول خرابی امن *F_PS_12* که با *fail-safe DP standard* تبادل داده می‌کند، بر اساس پروفایل *PROFIsafe* دریافت می‌شود. اگر مقدار ورودی دیجیتال معتبر باشد، در خروجی *Q* قرار می‌گیرد.



شکل ٦-٦- نمایی از بلاک درایور کانال ورودی *F_CH_BI*

۶-۲-۴- بلاک درایور قرائت مقدار *DINT* از یک آدرس *DP Slave* (*F_CH_DII*)

F-Channel driver for inputs of data type DINT of fail-safe DP standard slaves and fail-safe standard I/O devices

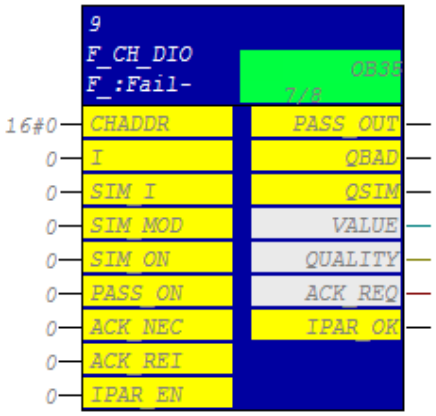


شکل ٦-٧- بلاک درایور قرائت مقدار *DINT* از یک آدرس *DP Slave* (*F_CH_DII*)

۶-۲-۵- بلاک درایور نوشتن مقدار *DINT* در یک آدرس *DP Slave* (*F_CH_DIO*)

F-Channel driver for outputs of data type DINT of fail-safe DP standard slaves and fail-safe standard I/O devices

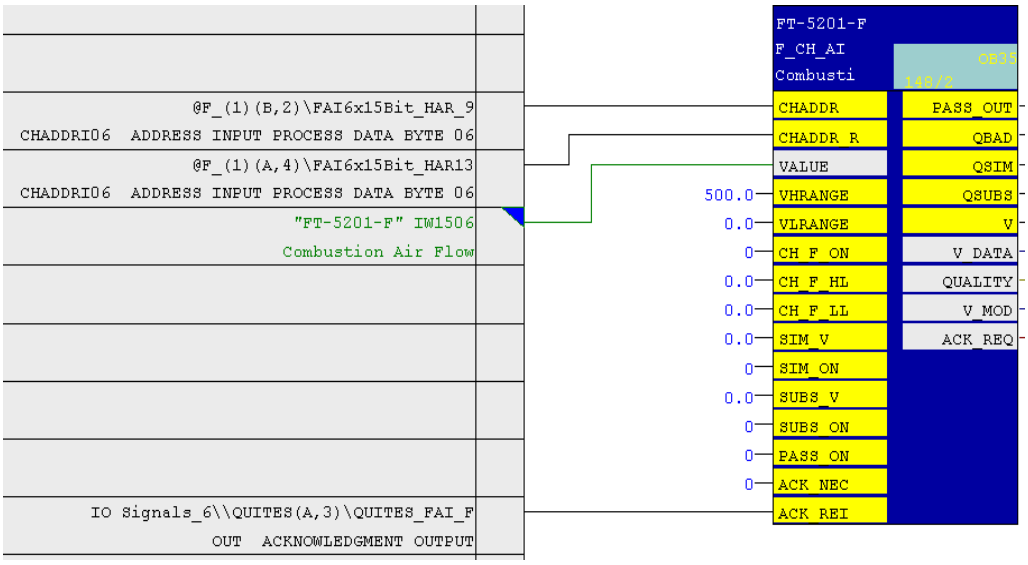
این بلوک *F* برای پردازش سیگنال یک مقدار خروجی از نوع *DINT* در *slave*های *fail-safe DP* *standard* یا دستگاههای *fail-safe standard I/O* استفاده می شود.



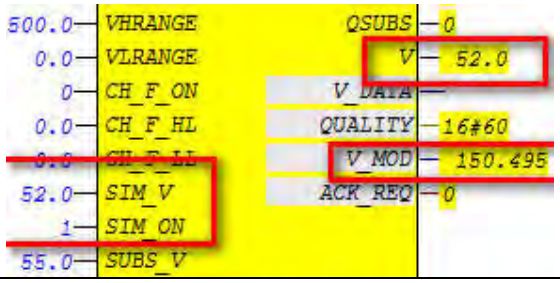
شکل ٦-٨- بلاک درایور نوشتن مقدار *DINT* در یک آدرس *DP Slave*

۶-۲-۶- بلاک درایور کانال ورودی آنالوگ F_CH_AI

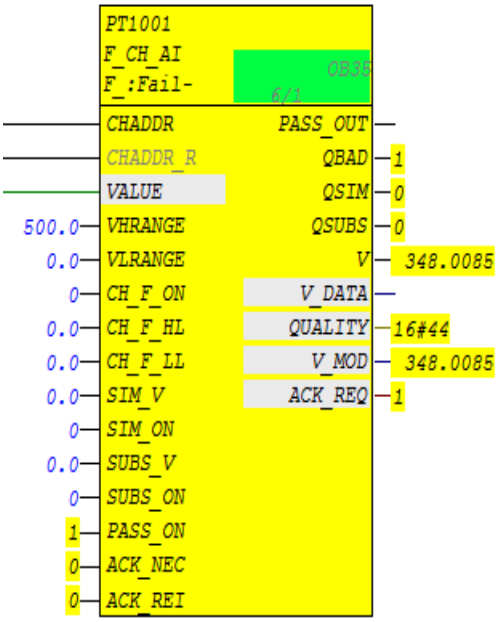
Fail-safe channel drivers for analog inputs of F-I/O (except fail-safe DP standard slaves)



پایه ورودی/خروجی	نوع داده	توضیح
CHADDR	*STRUCT*	*CHANNEL-ADDRESS IN F_MOD-BLOCK*
CHADDR_R	*STRUCT*	*REDUNDANT CHANNEL-ADDRESS IN F_MOD-BLOCK*
VALUE	*WORD*	به آدرس *IW* کانال آنالوگ متصل می‌شود.
VHRANGE	*F_REAL*	رنج بالای متغیر وردی فرآیند (*PV*) را مشخص می‌کند.
VLRANGE	*F_REAL*	رنج پایین متغیر وردی فرآیند (*PV*) را مشخص می‌کند.
CH_F_ON	*F_BOOL*	*1 = Enable Limit Value Monitoring* با یک شدن این ورودی مقدار *PV* با حدود مقادیر تعریف شده در ورودی‌های *CH_F_HL* و *CH_F_LL* مانیتور می‌شود.
CH_F_HL	*F_REAL*	*Overrange limit of input value (mA)*
CH_F_LL	*F_REAL*	*Underrange Limit Of Input Value (mA)*
SIM_V	*F_REAL*	مقدار ورودی برای شبیه سازی می‌باشد.
SIM_ON	*F_BOOL*	امکان شبیه‌سازی مقدار ورودی را فعال می‌کند.
SUBS_V	*F_REAL*	مقدار جایگزین در مواقع بروز خطا به این ورودی داده می‌شود.
SUBS_ON	*F_BOOL*	امکان جایگزینی مقدار ورودی با مقدار پایه *SUBS_V* را فعال می‌کند.
PASS_ON	*F_BOOL*	این پایه برای غیر فعال کردن کانال از طریق برنامه استفاده می‌شود. با یک کردن این پایه، کانال مربوطه غیرفعال می‌گردد. خروجی *QBAD* یک می‌شود. *1 = Activate Passivation*

ACK_NEC	*F_BOOL*	در صورت یک شدن این پایه، *Ack* کردن خطای *Passivation* ضروری می‌باشد. *1 = Acknowledgement Necessary*
ACK_REI	*F_BOOL*	*Acknowledgment Reintegration*
Normal value		اگر عدد دیجیتال دریافت شده از *F-I/O* در ورودی *VALUE* معتبر باشد، آن را به کمیت فیزیکی بر اساس مقادیر ورودی‌های *VLRANGE* و *VHRANGE* اسکیل کرده و در خروجی *V* با کد کیفیت*16#80* قرار می‌دهد. برای فعال کردن امکان انتقال مقادیر *VLRANGE* و *VHRANGE* به پارامترهای سایر بلوک‌ها، مقادیر ورودی‌های *VLRANGE* و *VHRANGE* در خروجی‌های *OVLRANGE* و *OVHRANGE* نیز نوشته می‌شود. الگوریتم تبدیل ورودی را یک سیگنال خطی فرض می‌کند هنگامی که *VLRANGE* *0.0 =* و *VHRANGE = 100.0*، است، یک مقدار درصد در خروجی تولید می‌شود. اگر *VHRANGE = VLRANGE* تنظیم شود،
V_MOD	*REAL*	*Value From Module* این خروجی مقدار آنالوگ خوانده شده از کارت را نشان می‌دهد. در صورتی که مد شبیه سازی با *SIM_ON=1* فعال شده باشد. خروجی *V* مقدار وارد شده در *SIM_V* را و پایه *V_MOD* مقدار خواند شده از *F-I/O* را به عنوان متغیر فرآیندی نشان خواهد داد. 
V_DATA	*REAL*	*Process Value Data (For Monitoring)* نوع داده این خروجی از نوع معمولی (*F_REAL* نیست) بوده و می‌توان به صورت مستقیم به بلاک‌های مانیتورینگ متصل کرد.
ACK_REQ	*BOOL*	*Acknowledgement Request* یک شدن این خروجی نشان می‌دهد. که خطا رفع شده و نیاز به *Ack* توسط کاربر می‌باشد.

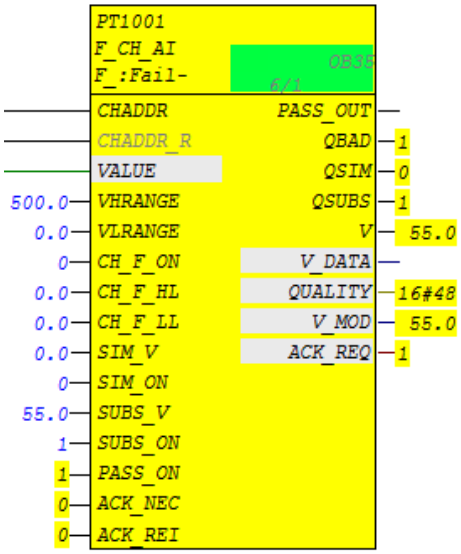
در صورتی که به صورت دستی ورودی *PASS_ON* را یک کنیم، کانال غیرفعال شده و مطابق شکل خروجی *BAD=1* یک شده و مقدار خروجی *V* در آخرین مقدار خود ثابت می‌ماند.



Fail-safe value → *SUBS_V*

اگر ورودی *SUBS_ON=1* باشد و به ورودی *SUBS_V* مقدار داده باشیم. در آن صورت با وقوع خطا و *passivate* شدن کانال، درحالت‌های زیر مقدار *SUBS_V* در خروجی *V* قرار می گیرد. همانطور که مشاهده می‌شود. خروجی *ACK_REQ=1* نیز یک می‌شود.

- مقدار آنالوگ ورودی به دلیل خطای ارتباطی (*PROFIsafe*) غیر معتبر باشد.
- مقدار آنالوگ ورودی به دلیل فالت در ماژول یا کانال (مانند قطعی سیم) غیر معتبر باشد.
- مقدار آنالوگ ورودی به دلیل *ovwrflow* یا *underflow* غیر معتبر باشد.
- مقدار آنالوگ ورودی به دلیل فالت تخطی از حدودد (*limit vilolation*) در کانال فعال (برای رنج اندازه گیری *4-20mA*)، غیر معتبر باشد.
- یک حالت *Passivation* با *PASS_ON = 1* وجود داشته باشد.
- یک پروسه *F-Startup* در حال اجرا باشد.



فعال‌سازی کانال پس از رفع خطا (*Reintegration after error elimination*)

بعد از رفع خطا، مقدار ورودی آنالوگ دریافت شده از *F-I/O* می‌تواند به طور خودکار و یا پس از تصدیق کاربر از طریق *ACK_REI* مجدد تولید (*Reintegrate*) شود.

در صورتی که ورودی *ACK_NEC=1* باشد، بعد از رفع خطا تصدیق کاربر از طریق ورودی *ACK_REI* لازم است. ولی اگر *ACK_NEC = 0* باشد، تصدیق به صورت خودکار انجام می‌گیرد.

(NUMUR) limit value check in the 4 to 20 mA measuring range

در دستورالعمل *NUMUR* برای پردازش سیگنال آنالوگ، برای سیگنال آنالوگ نوع *life zero* یعنی سگنال *4-20 mA*، در جایی که فالت در کانال وجود دارد، حدود تعریف شده است. که عبارت انداز:

3.6 mA <analog signal<21 mA

به طور پیش فرض، این حدود تعریف شده در *NAMUR*، برای چک کردن مقادیر حدی (*limit value checking*) استفاده می‌شود. برای چک کردن حدود فالت جدید بایستی ورودی *CH_F_ON=1* تنظیم شود و مقادیر حدود جدید در ورودی‌های *CH_F_HL* و *CH_F_LL* برحسب میلی آمپر (*mA*) وارد شود.

CH_F_LL < analog signal < CH_F_HL

در صورت وقوع سرریز (*overflow*)یا *underflow* از حدود فالت کانال فعال، خروجی *QBAD* نیز یک شده و بسته به مقدار ورودی *SUBS_ON*، مقدار خرابی امن *SUBS_V* یا آخرین مقدار معتبر خروجی در خروجی قرار می‌گیرد.

۶-۲-۷- مفاهیم *Live zero* و *Dead zero*

عبارت *Live zero* به کمترین مقدار رنج سیگنال (*lowest end or 0% of the signal range*) اشاره می‌کند. و عبارت *Dead zero* به یک وضعیت شکست (*signal failure or dead signal*) در سیگنال اشاره می‌کند. به عنوان مثال اگر یک سیگنال جریان *4-20 mA* را در نظر بگیرید، مقدار *4mA* نمایش‌دهنده *live zero* یا *0%* سیگنال بوده و مقدار *0mA* (*no current*) نمایش‌دهنده یک *dead-zero* می‌باشد.

سیگنال *4-20 mA* یک مثال کلاسیک از یک سیستم سیگنالینگ است که بین یک سیگنال ٪۰ و یک شکست در سیگنال، تفاوت قائل می‌شود.در نتیجه امکان تشخیص خرابی در حلقه را فراهم می‌کند.

به تعریف دیگر یک *live zero* یک سیگنال در یک حلقه (*loop signal*) می‌باشد که در آن مقدار صفر سیگنال بالاتر از صفر واقعی می‌باشد. به عنوان مثال *4-20 mADC*،*1-5 VDC*، *10-50mV* و غیره.

یک *dead zero* سیگنالی است که در آن مقدار صفر سیگنال، خودش مقدار صفر حلقه می‌باشد. مانند سیگنال *0-20 mADC*، *0-100 mv*، *0-5 vdc* و غیره.

۶-۲-۸- بلاک درایور کانال آنالوگ خروجی

به طور معمول در سیستم‌های *Fail-safe* تنها ماژول *F-AI* وجود دارد و از ماژول *F-AO* استفاده نمی‌شود.

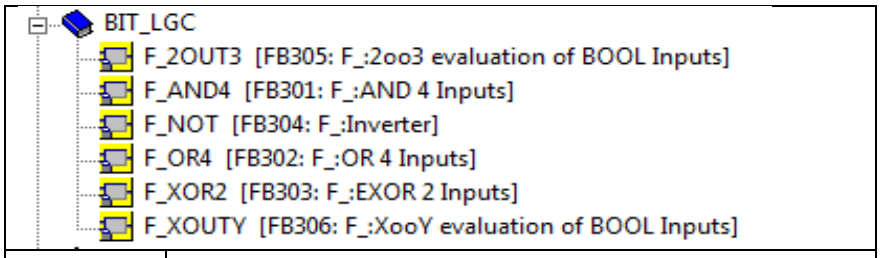
با توجه به این که کارت‌های *Fail safe* برای *EMS* و سیستم‌های *SAFETY* استفاده می‌شوند. خروجی آنالوگ تو این دسته کارتها وجود ندارد.

همان‌طوری که از عنوان «قطع اضطرای» بر می‌آید، در سیستم‌های *ESD* یا *fail-safe* هدف غایی، هدایت پلنت به یک وضعیت ایمن (*fail-safe*) از طریق توقف یا تریپ سریع یک زیر سیستم پس از وقوع یک رخداد خطرناک می‌باشد این کار با استفاده از عملگرهای باینری که به صورت *On/Off* هستند انجام می‌شود.

۶-۳- مجموعه *BIT_LGC*

BIT_LGC Family: Logic blocks with the BOOL data type

بلاک‌های این مجموعه برای پیاده‌سازی عملیات بیتی (نوع داده *BOOL*) لاجیک *F* استفاده می‌شود. گروه *BIT_LGC* شامل شش بلاک برای عملیات باینری می‌باشد:



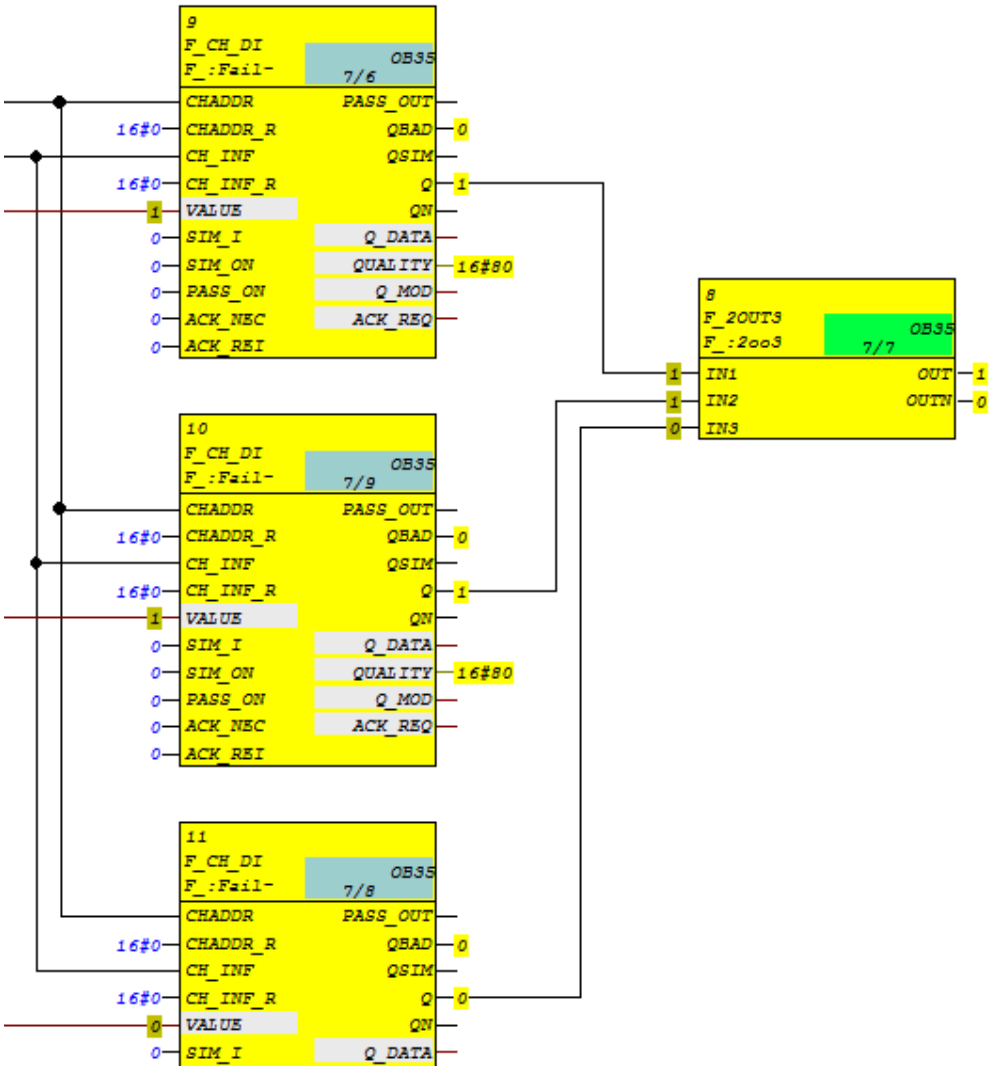
F_AND4	چهار ورودی *F_BOOL* را با هم *AND* می‌کند.
F_OR4	چهار ورودی *F_BOOL* را با هم *OR* می‌کند.
F_XOR2	دو ورودی *F_BOOL* را با هم *XOR* می‌کند.
F_NOT	ورودی نوع *F* را *Not* می‌کند.
F_2OUT3	ارزیابی *2oo3* را بر روی ورودی‌های نوع داده *F_BOOL* انجام می‌دهد.
F_XOUTY	ارزیابی *XooY* را بر روی ورودی‌های نوع داده *F_BOOL* انجام می‌دهد.

۶-۳-۱- بلاک *F_2OUT3*

F_2OUT3: 2oo3 evaluation of inputs of data type BOOL

این بلاک وضعیت سه ورودی باینری را برای مقدار یک مانیتور می‌کند. تا زمانی که دو وردی از سه ورودی یک باشد. خروجی *Q* یک بوده و در غیر این صورت خروجی *Q* صفر خواهد بود. مقدار پیش‌فرض پایه‌های بلاک صفر می‌باشد.

فرق این بلاک با بلاک *F_2OO3DI* این است که این بلاک آنالیز اختلاف زمانی (*Discripancy Analysis*) را بین ورودی‌ها انجام نمی‌دهد.



شکل ٦-٩- مثال از پیاده‌سازی لاجیک *F_2OUT3*

۶-۳-۲- بلاک *F_XOUTY*

این بلاک تا ١٦ ورودی باینری را برای وضعیت سیگنال یک در ورودی‌های *IN1* تا ورودی *INY* را مانیتور می‌کند. تعداد ورودی‌های باینری که بایستی مانیتور شود، توسط پارامتر *Y* تعیین می‌شود. خروجی *Q* زمانی یک است که حداقل به تعداد *X* ورودی از ١٦ ورودی یک باشد.

نکته: در شرایط $X>Y$ یا $X \le 0$، $X > 16$، $Y \le 0$ خروجی *Q* صفر می‌باشد.

۶-۴- بلاک‌های مقایسه کننده (*comparing two input values*)

بلاک‌های مقایسه کننده که در گروه *COMPARE* قرار دارند. دو ورودی را بایکدیگر مقایسه می‌کنند.

COMPARE F_CMP_R [FB313: F_:REAL Comparator for two REAL] F_LIM_HL [FB314: F_:Limit Monitoring high Level] F_LIM_LL [FB315: F_:Limit Monitoring low Level]	
F_CMP_R	دو عدد اعشاری را با هم مقایسه می‌کند
F_LIM_HL	حد بالای یک مقدار *Real* را مانیتور می‌کند.
F_LIM_LL	حد پایین یک مقدار *Real* را مانیتور می‌کند.

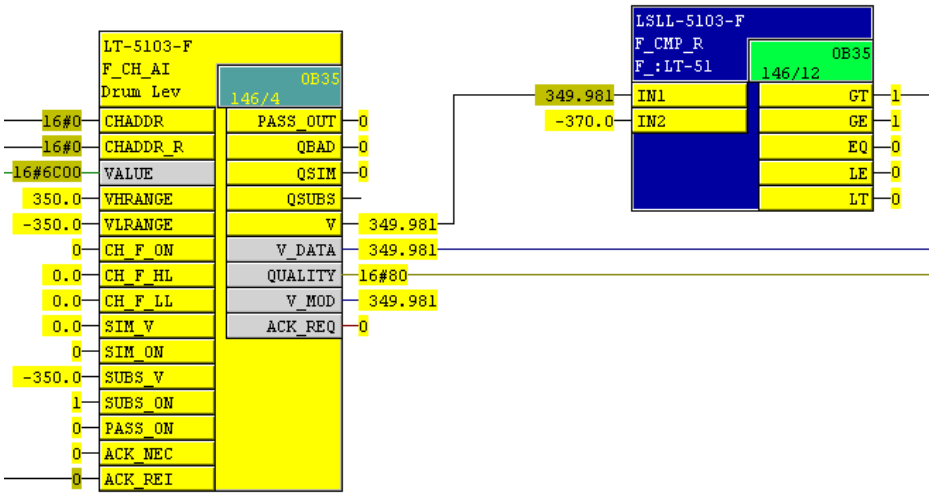
- در بلاک *F_CMP_R* در صورتی که یکی از ورودی‌های *IN1* یا *IN2* یک عدد نامعتبر *(NaN)* باشد. خروجی‌های *GT* و *LT* به یک تنظیم می‌شود.

۶-۴-۱- بلاک مقایسه کننده دو عدد اعشاری (*F_CMP_R*)

F_CMP_R Comparator for two REAL values

دو ورودی *IN1* و *IN2* با نوع داده *F_STRUCT* را باهم مقایسه کرده و چهار خروجی *F_BOOL* را تولید می‌کنند. مقدار پیش‌فرض پایه‌ها صفر می‌باشد.

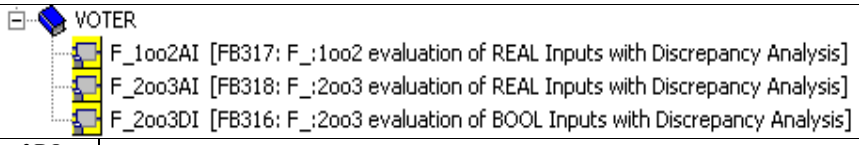
Name	Data type	Description
IN1	*F_REAL*	*Input 1*
IN2	*F_REAL*	*Input 2*
GT	*F_BOOL*	*IN1 > IN2*
GE	*F_BOOL*	*IN1 ≥ IN2*
EQ	*F_BOOL*	*IN1 = IN2*
LT	*F_BOOL*	*IN1 < IN2*
LE	*F_BOOL*	*IN1 ≤ IN2*



شکل ٦-١٠- مثالی از پیاده‌سازی بلاک *F_CMP_R*

۶-۵- بلاک‌های *Voter* (Voter blocks)

این بلاک‌ها مطابق جدول زیر در گروه *Voter* از کتابخانه *F* قرار دارند.



F_2oo3DI	ارزیابی *2oo3* را برای ورودی‌های *BOOL* به همراه آنالیز *discrepancy* انجام می‌دهد.
F_2oo3AI	ارزیابی *2oo3* را برای ورودی‌های *REAL* به همراه آنالیز *discrepancy* انجام می‌دهد.
F_1oo2AI	ارزیابی *1oo2* را برای ورودی‌های *REAL* به همراه آنالیز *discrepancy* انجام می‌دهد.

۶-۵-۱- بلاک ارزیابی *2oo3* برای ورودی‌های *BOOL* (*F_2oo3DI*)

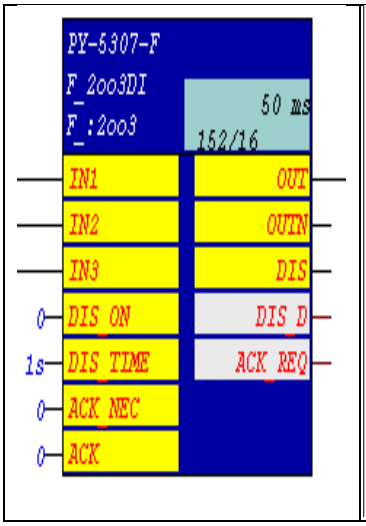
2oo3 evaluation of inputs of data type BOOL with discrepancy analysis

این بلوک سه سیگنال باینری *F_BOOL* را برای وضعیت یک مانیتور می‌کند. اگر حداقل دو ورودی از سه ورودی *INX* یک باشد، خروجی *OUT* یک می‌شود. در غیر این صورت خروجی *OUT* صفر می‌شود. خروجی *OUTN*،معکوس شده (*Not*) خروجی *OUT* است.

اگر ورودی *DIS_ON* به یک تنظیم شود، در آن صورت ورودی‌ها بایستی در یک زمان مشخص برسند. به این معنی که آنالیز اختلاف زمانی بین ورودی‌ها انجام می‌گیرد. اگر اختلاف بین ورودی *INX* و دو ورودی دیگر *INy* طولانی‌تر از زمان اختلاف اختصاص داده شده در ورودی *DIS_TIME* باشد، یک خطای اختلاف زمانی تشخیص داده شده و در خروجی *DIS* و *DIS_D* با مقدار ۱ ذخیره می‌شود.

در صورتی که ورودی *ACK_NEC=0* باشد و خطای اختلاف زمانی تشخیص داده شود، در آن صورت خطای اختلاف زمانی به صورت اتوماتیک تصدیق می‌شود.

خروجی *ACK_REQ = 1* به این معنی است که خطا رفع شده و کاربر بایستی از طریق ورودی *ACK* عمل صدیق خطا را باید انجام دهد.



Inputs/outputs

	Name	Data type	Description	Default
Inputs:	IN1	F_BOOL	Input 1	0
	IN2	F_BOOL	Input 2	0
	IN3	F_BOOL	Input 3	0
	DIS_ON	F_BOOL	1 = Discrepancy analysis	0
	DIS_TIME	F_TIME	Discrepancy time in ms	1000
	ACK_NEC	F_BOOL	1 = Acknowledgment necessary	0
	ACK	F_BOOL	Acknowledgment	0
Outputs:	OUT	F_BOOL	Output	0
	OUTN	F_BOOL	NEGATING OUTPUT	1
	DIS	F_BOOL	DISCREPANCY ERROR	0
	DIS_D	BOOL	DISCREPANCY ERROR DATA	0
	ACK_REQ	BOOL	ACKNOWLEDGMENT REQUEST	0

ACK_NEC	*F_BOOL*	*Acknowledgment necessary=1* با یک شدن این ورودی عمل تصدیق فالت در بلاک ضروری می‌باشد.

نکته : در صورت وقوع یک خطا در فرمت داده ایمنی (*safety data format*) بلاک داده مربوطه (*instance DB*) یک *F-STOP* تریگر می‌شود.سپس یک پیغام تشخیص خطا (*iagnostic event*) در بافر *F-CPU* وارد می‌شود.

"Safety program: Error in safety data format in DB" (Event ID 16#75DA)

۶–۵–۲– بلاک *F_2oo3AI*

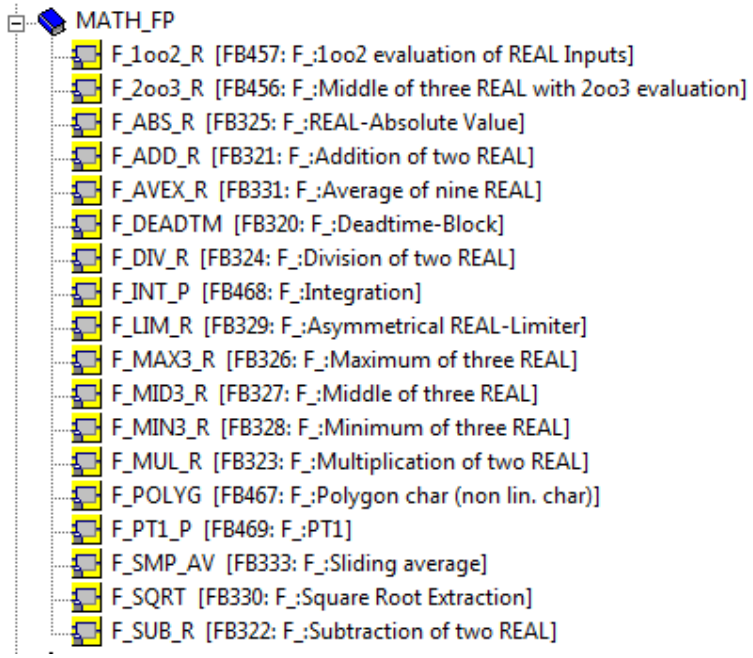
در صورتی که بخواهیم خروجی *V* یک بلاک درایور *F_CH_AI* را به یکی از ورودی‌های بلاک *F_2oo3AI* متصل کنیم. بایستی ورودی *QBADx* بلاک *F_2oo3AI* به خروجی *QBAD* بلاک *F_CH_AI* وصل کنیم.

۶–۵–۳– بلاک افزونه کردن ورودی ها (*RED_IN*)

این بلاک برای تنظیم کانال‌های ماژول‌های ورودی به صورت افزونه به کار می‌شود. به صورت اتوماتیک با کامپایل چارت و در صورت افزونه تعریف شدن کارت‌های *I/O* در چارت‌های @ درج می‌شود.

۶–۶– مجموعه بلاک‌های محاسباتی *F* اعشاری (*MATH_FP*)

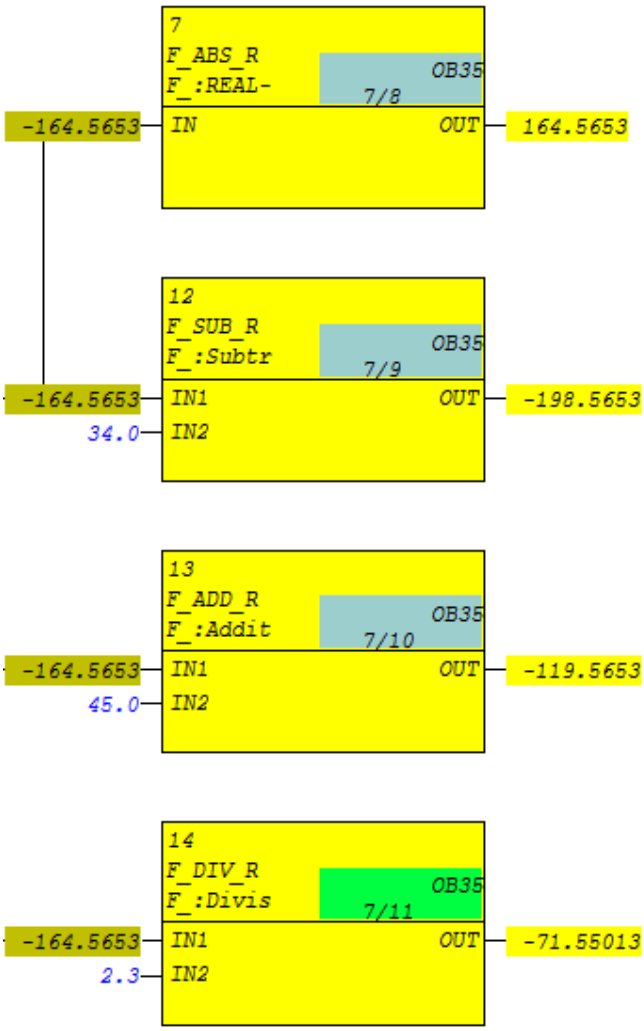
این بلاک‌های برای عملیاتی محاسباتی بر روی ورودی‌های نوع *F* به کار می‌رود.



شکل ٦-١١- مجموعه *MATH_FP* - بلاک‌های محاسباتی

۶-۶-۱- بلاک *F_ABS_R* (*F_ABS_R: Absolute value of a REAL value*)

این بلاک مقدار مطلق *F_REAL* در ورودی *IN* را در خروجی *OUT* تولید می‌کند.



شکل ۱۲۶-- مثالی از بلاک‌های محاسباتی

۶-۶-۲- بلاک (*F_1oo2_R*)

1oo2 evaluation of inputs of data type REAL

این بلاک بسته به مقدار ورودی *QBAD1*، داده دریافت شده از یکی از ورودی‌های *IN1* و *IN2* را به خروجی *OUT* منتقل می‌کند.

QBAD1 = 0 >> OUT = IN1
QBAD1 = 1 >> OUT = IN2

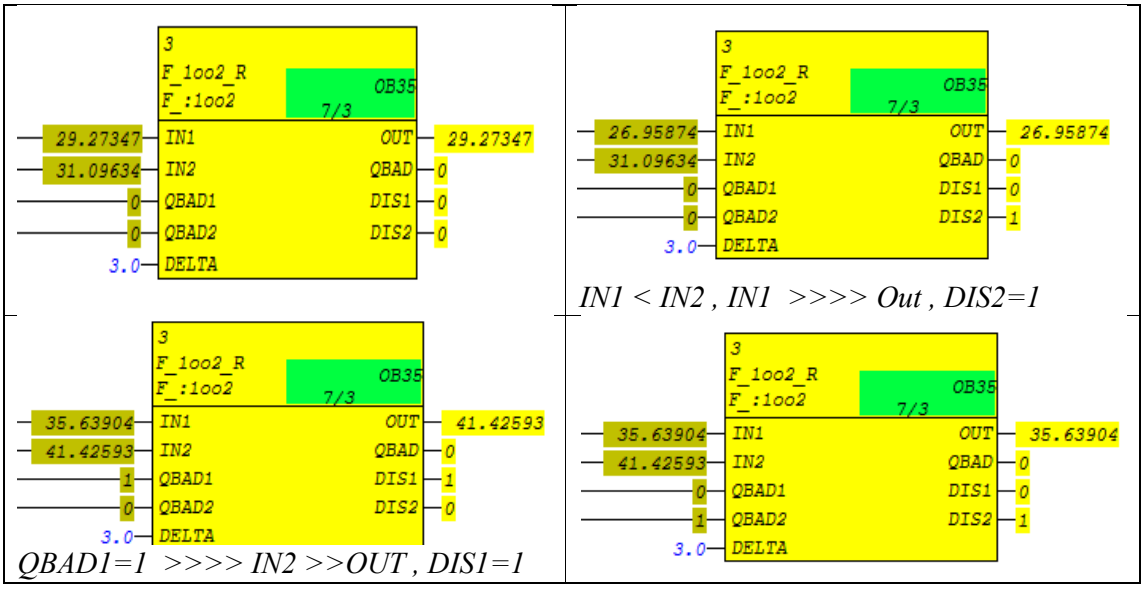
اگر هر دو ورودی *IN1* و *IN2* نامعتبر باشد (*QBAD1=1* و *QBAD2 = 1*)، خروجی *OUT* نیز نامعتبر و خروجی *QBAD=1* خواهد بود.

اگر اختلاف بین ورودی *IN1* و *IN2* بیش از مقدار تلورانس *DELTA* اختصاص داده شده باشد، یک خطای اختلاف (*discrepancy error*) را تشخیص داده و خروجی‌های *DIS1* و *DIS2* به شرح زیر فعال می‌شوند.

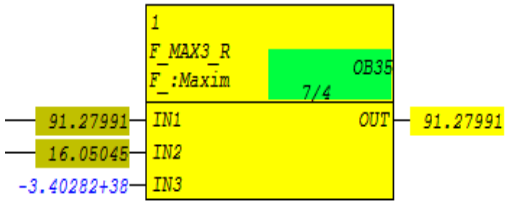
DIS1 = 1 if IN2 is output at the OUT output.

DIS2 = 1 if IN1 is output at the OUT output.

DELTA	F_REAL	مقدار تولورانس در اختلاف بین دو ورودی را مشخص می‌کند.

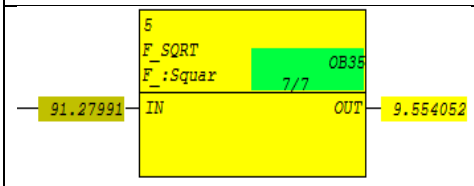


۶-۶-۳- بلاک انتخاب حداکثر مقدار بین سه ورودی (*F_MAX3_R*)



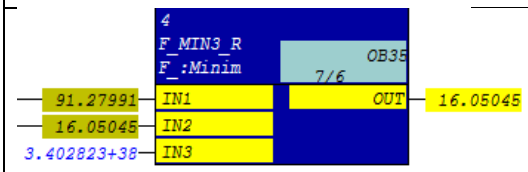
Maximum of three REAL values

این بلوک ورودی‌های *IN1* ، *IN2* و *IN3* را مقایسه و حداکثر آنها را در خروجی *OUT* انتقال می‌دهد. مقدار پیش‌فرض برای همه ورودی‌ها برابر مقدار -*3,402823e+38* (بزرگترین عدد حقیقی منفی) است. لذا می‌توان مقدارحداکثر، بین دو ورودی را نیز حساب کرد.



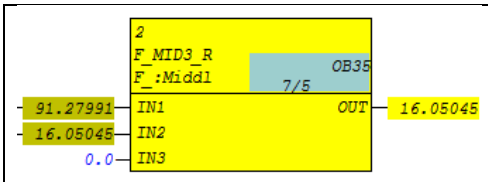
Square root of a REAL value

ریشه مربع ورودی را محاسبه و در خروجی *OUT* قرار می‌دهد.



Minimum of three REAL values

حداقل بین سه ورودی را در خروجی قرار می‌دهد.

	*Mean value of three REAL values* متوسط سه مقدار ورودی را در خروجی قرار می‌دهد.

۶-۶-۴- بلاک انتگرال‌گیر *F* (*F_INT_P*)

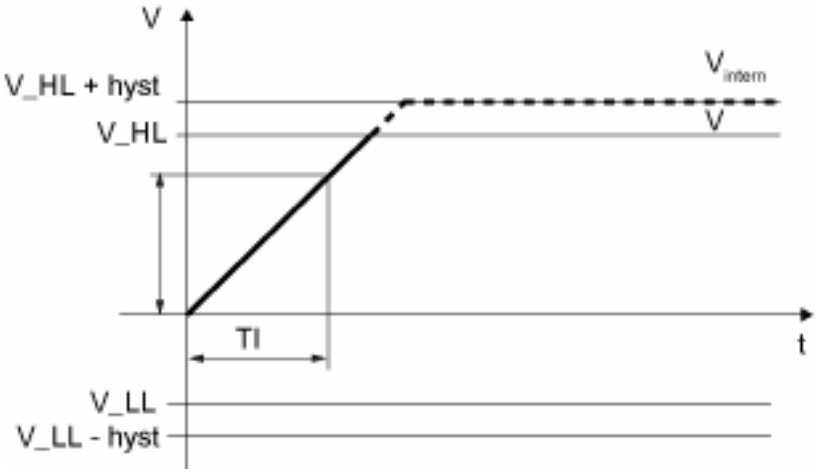
Integration function with integration and track mode

بلاک *F_INT_P* از زیرمجموعه *MATH_FP*، معادل *F* بلاک انتگرال‌گیر معمولی در کتابخانه *PCS7* می‌باشد. این بلاک می‌تواند در دو مد متفاوت زیر کار کند.

- *Integration mode*
- *TRACK mode*

۱. مد انتگرالی

در مد انتگرالی، خروجی *V* با یک سیگنال مثبت در ورودی *U* افزایش و با سیگنال منفی در ورودی *U*، کاهش می‌یابد. این بلوک با تشکیل جمع کل مقادیر ورودی با توجه به قانون ذوزنقه‌ای برای هر فاصله نمونه برداری (*TS*) در مد اتگرالی کار می‌کند. نتیجه *Vinternal* مطابق شکل زیر در محدوده *V_HL+hyst* تا *V_LL-hyst* قرار می‌گیرد. نتیجه حاصله پس از این که به مقادیر *V_LL* و *V_HL* محدود می‌شود، در خروجی *V* نوشته می‌شود.



*Hyst = HYS / 100 * (V_HL - V_LL)*

شکل ۶-۱۳- پاسخ شیب بلاک *F_INT_P*

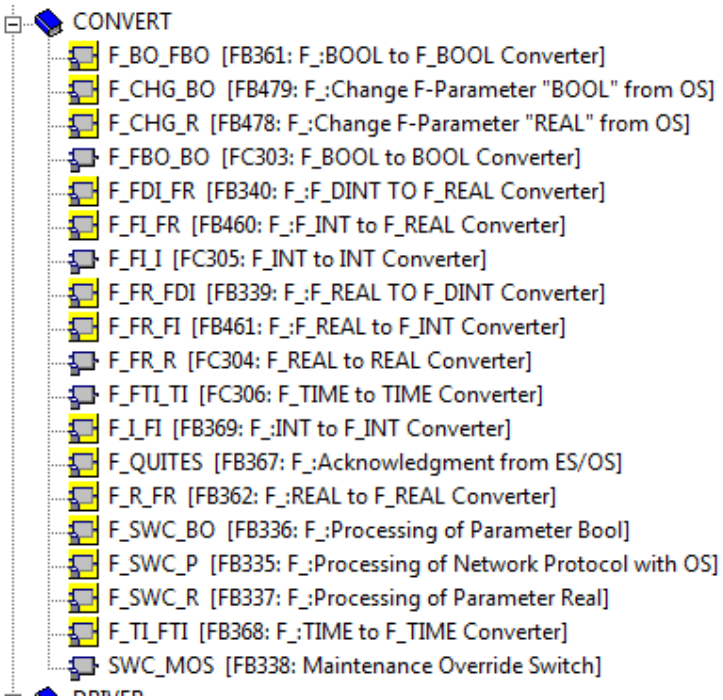
خروجی *V* مطابق فرمول زیر محاسبه می‌شود.

$$V_X = V_{X-1} + U_X * \frac{Ts}{TI}$$

VX	*Current internal output value*
VX-1	*Last internal output value (Vinternal)*
Ts	*Sampling time (time elapsed between two F-Block processing cycles) in seconds*
TI	*Integration time in seconds*
UX	*Current input value*

نکته: می‌توان با دادن یک مقدار ثابت به ورودی بلوک *F_INT_P* ، افزایش سطح یک مخزن را تا ۱۰۰ درصد شبیه‌سازی کرد.

۶-۷- بلاک‌های مبدل (*Convert*)



۶-۷-۱- بلاک *F_SWC_BO*

F_SWC_BO: Processing of a parameter of data type F_BOOL for operator input via the OS

بلاک *F_SWC_BO* امکان ایجاد تغییرات در پارامترهای *F* نوع داده *BOOL* را در برنامه ایمنی از سیستم مانیتورینگ ایستگاه اپراتوری فراهم می‌کند.به عبارت دیگر برای ورود یا دریافت یک داده *Bool* از *OS* در برنامه *F* بکار می‌رود. خروجی *OUT* به ورودی که در برنامه *F* که بایستی تغییر داده شود متصل می‌گردد.

خروجی‌های *OUT* و *AKT_VAL* می‌تواند به طور مستقل از طریق ورودی‌های *S* و *R*، *Set* یا *Reset* شوند. خروجی‌های *OUT* و *AKT_VAL* با یک لبه مثبت در *S*، *Set* می‌شوند. اولویت با ورودی *Reset* است، در نتیجه، تا زمانی که ورودی *R* برابر با ۱ باشد (*R = 1*)، این خروجی‌ها *Reset* خواهند شد.

Inputs/outputs

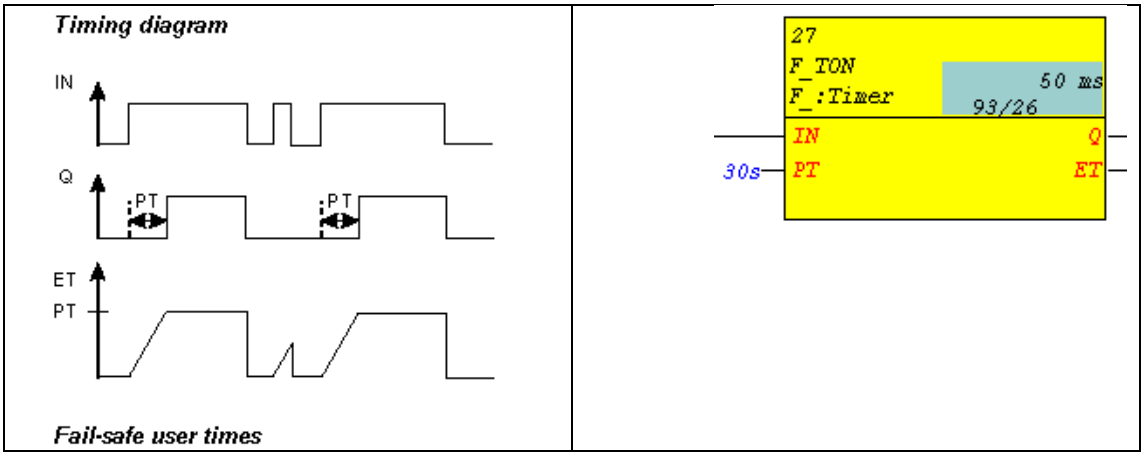
	Name	Data type	Description	Default
Inputs:	S	F_BOOL	Set input	0
	R	F_BOOL	Reset input	0
	CS_VAL	F_BOOL	Cold restart	0
Outputs:	OUT	F_BOOL	Current value of the operated parameter	0
	AKT_VAL	BOOL	Current value of the operated parameter for the OS	0

۶–۷–۲– بلاک F_FR_FDI

تبدیل داده از F_REAL به F_DINT را انجام می‌دهد.

۶–۷–۳– بلاک تایمر تاخیر در روشن (F_TON: Timer switch-on delay)

با لبه بالا رونده سیگنال در ورودی IN تایمر شروع و پس از زمان PT خروجی Q یک می‌شود. این تایمر با صفر شدن ورودی IN ریست می‌شود.



۶–۷–۴– تصدیق خطا توسط کاربر (Acknowledgement by ACK_REI)

Regeneration of F-I/O Modules or depassivation of F- I/O modules

برای فعال کردن یک کانال پس از Passivation و یا به اصطلاح فعال‌سازی Regenerate کردن کانال، بایستی پس از رفع خطا، یک تصدیق از طریق پایه ACK_REI بلاک‌های درایور کانال آنالوگ و دیجیتال انجام داد. برای این منظور از فانکشن بلاک F_QUITES کتابخانه F استفاده می‌کنیم. به این ترتیب که به پایه ورودی IN بلاک F_QUITES مقدار ٦ و پس از حداکثر یک دقیقه مقدار ۹ را اعمال می‌کنیم. با این کار در خروجی بلاک F_QUITES (متصل به ورودی ACK_REI بلاک‌های درایور) به مدت یک سیکل مقدار ۱ ایجاد شده

و خطا ریست می‌شود. با این کار بلاک به عملیات عادی یعنی تولید مجدد مقدار از روی سیگنال کانال (*Regenerate*) ادامه می‌دهد.

عمل فعال سازی *ACK_REI* را می‌توان از طریق ورود مقدار ٦ و ٩ در یکی از صفحات *HMI* پیاده سازی کرد. که در ادامه تشریح می‌شود.

۶-۷-۵- بلاک *F_QUITES*

Fail-safe acknowledgement via the ES/OS

بلاک *F_QUITES* امکان *fail-safe acknowledgment* را از یک کامپیوتر *non-fail-safe ES/OS* فراهم می‌کند. این کار امکان کنترل فعال‌سازی دوباره (*reintegration*) کانال‌های *I/O* پس از *Passivate* شدن را فراهم می‌کند. عمل *Acknowledgment* در دو مرحله انجام می‌شود.

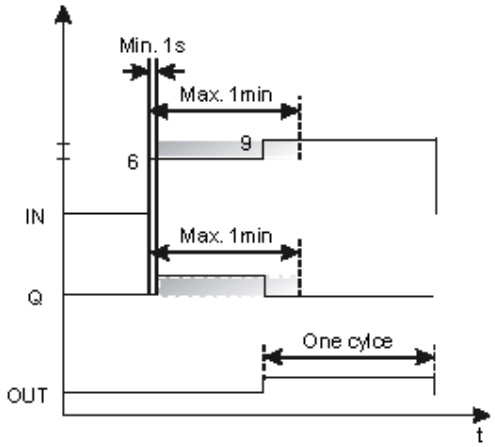
Inputs/outputs

	Name	Data type	Description	Default
Input:	*IN*	*INT*	*Input*	*0*
Outputs:	*OUT*	*F_BOOL*	*ACKNOWLEDGMENT OUTPUT*	*0*
	Q	*BOOL*	*Status of the time evaluation*	*0*

۱- ابتدا مقدار ٦ به ورودی *IN* اعمال می‌شود.

۲- سپس در مدت کمتر از یک دقیقه ورودی *IN* بلاک از مقدار "٦" به مقدار "٩" تغییر می‌یابد.

بلوک *F* پس از این که مقدار ورودی به ٦ تغییر یافت ارزیابی می‌کند. که آیا مقدار ورودی *IN* در حداقل زمان ۱ ثانیه یا در حداکثر زمان ۱ دقیقه به مقدار ۹ تغییر یافته است یا نه. سپس مقدار یک را در خروجی *OUT* بلاک (خروجی برای *acknowledgement*) برای مدت زمان یک سیکل قرار می‌دهد.



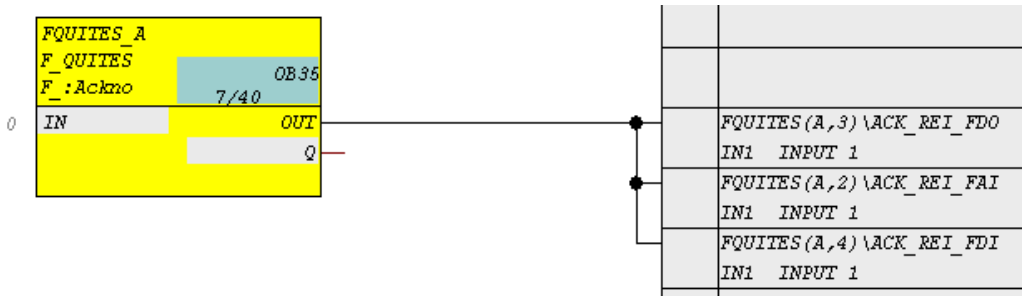
اگر یک مقدار نامعتبر وارد شود و یا اگر تغییر به مقدار "٩" قبل از ۱ ثانیه و یا پس از ۱ دقیقه رخ دهد، ورودی *IN* به صفر ست شده و دو مرحله بالا باید تکرار شود.

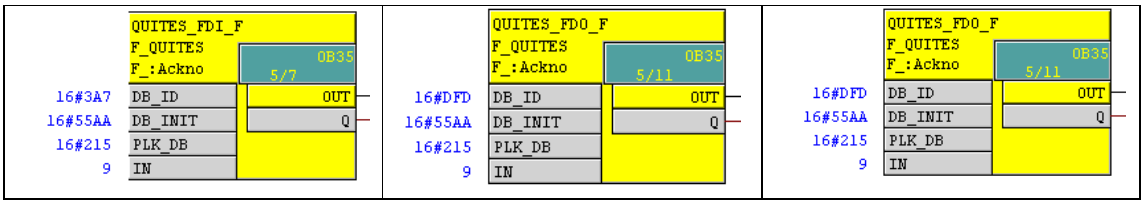
در طول زمانی که مقدار ورودی *IN* از "٦" به "٩" باید تغییر کند، خروجی *non-fail-safe Q* بلاک به یک ست می‌شود. در دیگر موارد *Q* دارای یک مقدار صفر می‌باشد.

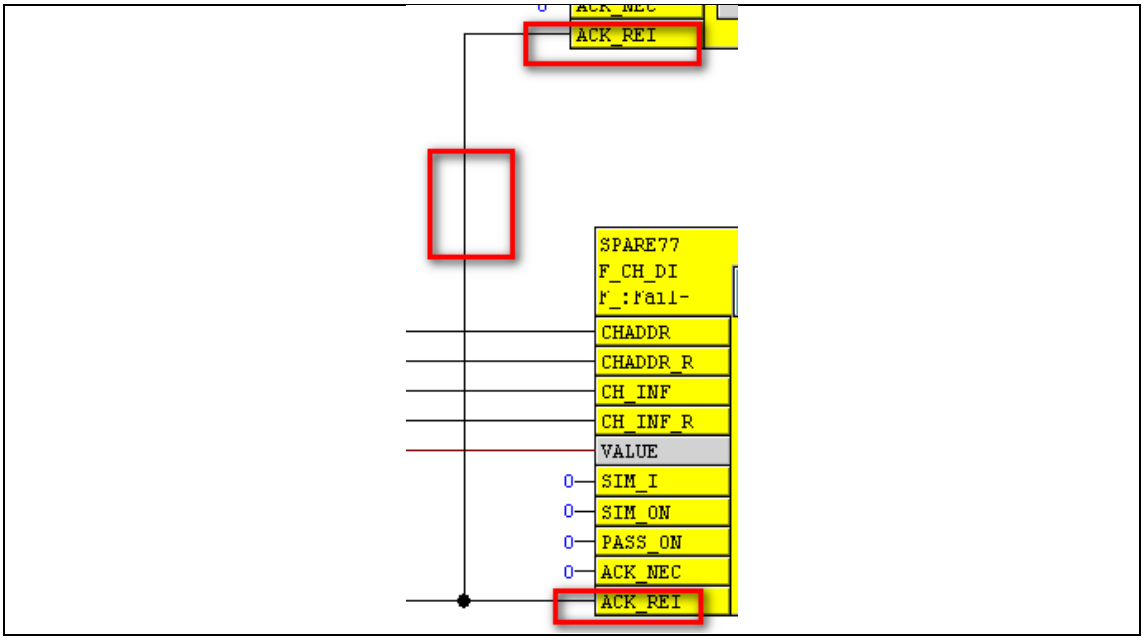
دو پارامتر ورودی *IN* و خروجی *Q* بلاک دارای *attribute S7_m_c* می‌باشد. به این معنی که این دو پارامتر در *OS* قابل دسترسی می‌باشد. برای این منظور معمولاً برای هر یک از دسته ماژول‌های ورودی/خروجی دیجیتال و آنالوگ (*F_DI,F_DO,F_AI*) یک بلاک *F_QUITES* در یک چارت *CFC* با عنوان *QUITES* قرار داده و خروجی *OUT* هر یک از بلاک‌های *F_QUITES* را به ورودی *ACK_REI* بلاک‌های درایور کانال متصل می‌کنیم. مطابق شکل زیر در یک پروژه نمونه در چارت *QUITES* سه بلاک *F_QUITES* با عناوین زیر برای عمل *Acknowledgement* استفاده شده است.

- *QUITES_FDI_F*
- *QUITES_FDO_F*
- *QUITES_FAI_F*

در شکل زیراز بلاک *F_QUITES*





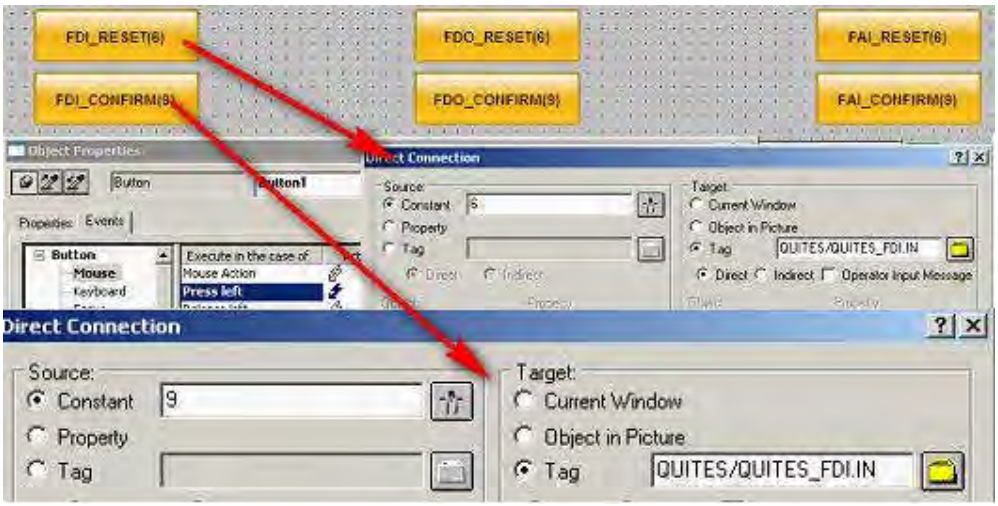


۶-۷-۶- پیاده‌سازی *user acknowledgment* از طریق سیستم *OS*

۱. سه عدد بلاک *F_QUITES* برای بلاک‌های *DI*، *DO* و *DI* به برنامه *F* در داخل یک چارت *CFC* با نام *QUITES* اضافه می‌کنیم.

۲. خروجی *OUT* بلاک‌های *F_QUITES* را به ورودی *ACK_REI* درایورهای کانال سیگنال متصل کنید.

۳. یک *Button* برای نوشتن مقدار ٦ (*Acknowledge value*) و یک *Button* دیگر برای نوشتن مقدار ۹ (*second acknowledgment step*) در یکی از *Picture*های *OS* پیکربندی کنید.

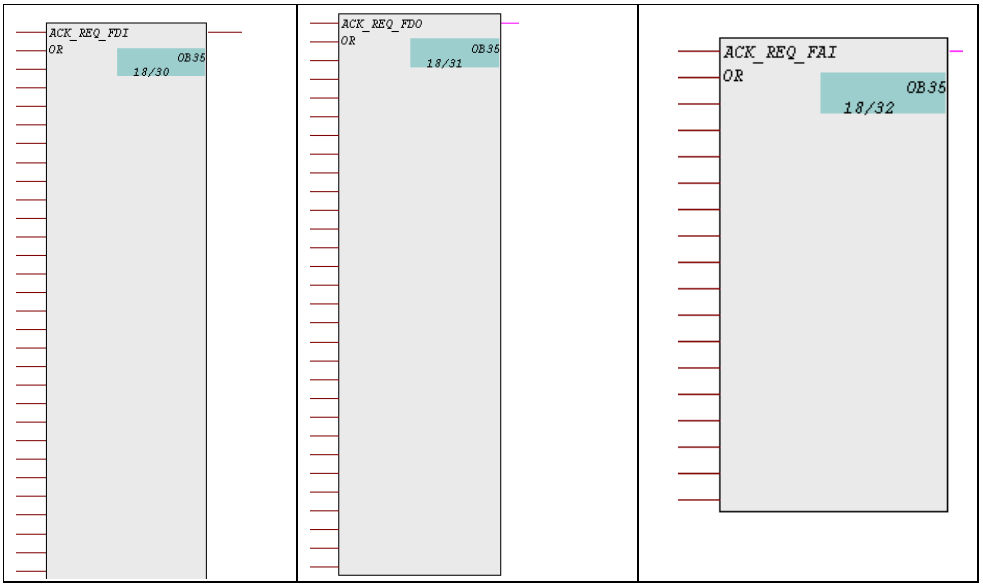


شکل ٦-١٤- اعمال مقادیر ٦ و ۹ به ورودی *IN* بلاک *F_QUITES*

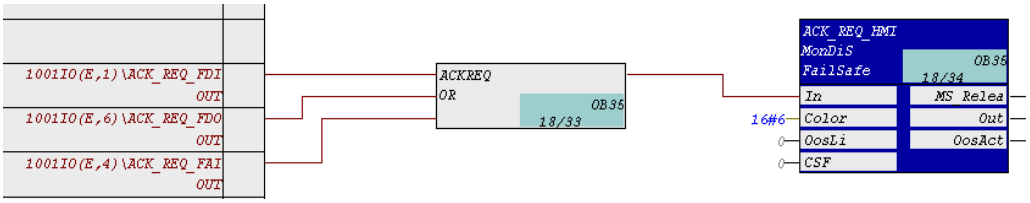
۶-۷-۷- پیاده سازی *F-user acknowledgment* در *OS*

برای این که پیغام درخواست *acknowledgment* را برای *Depassivation* ورودی‌ها را در *OS* نمایش‌دهیم. تمام پایه‌های خروجی *ACK_REQ* بلاک‌های درایور کانال‌های ورودی /خروجی را با استفاده از یک بلاک *OR* در بخش استاندارد برنامه کاربر، گروه‌بندی کرده و مقدار خروجی *OR* را از طریق یک بلاک *DIG_MON* در *OS* قابل دسترس می‌کنیم. به عنوان مثال در در یک برنامه به صورت زیر عمل می‌کنیم:

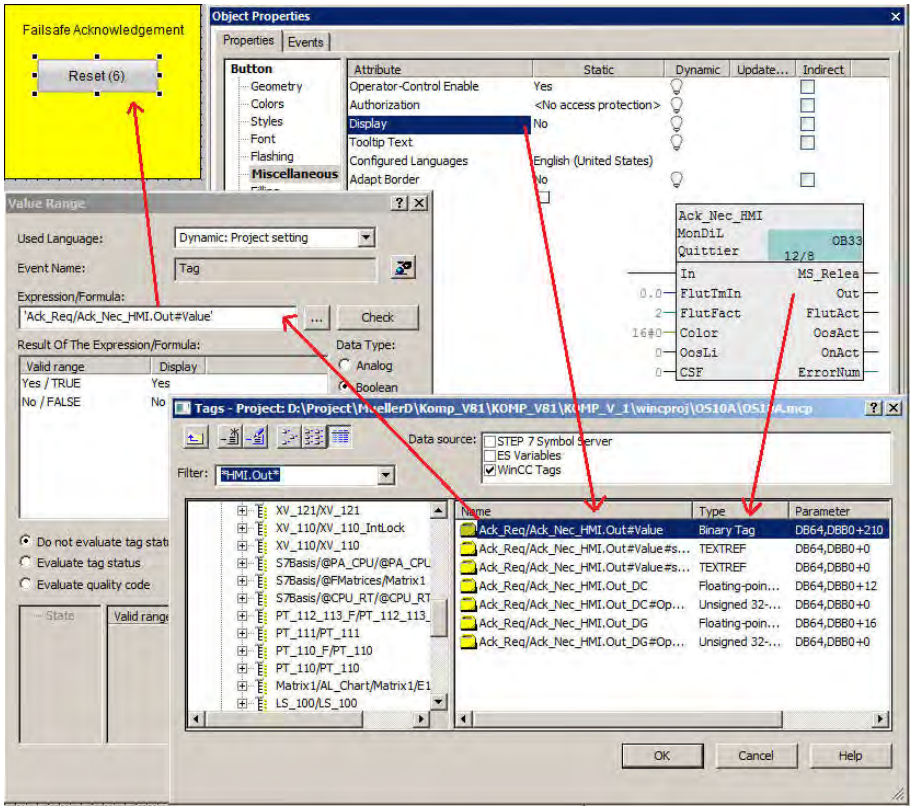
۱. تمام خروجی‌های *ACK_REQ* بلاک‌های درایور کانال دیجیتال ورودی را به یک بلاک *OR* با عنوان *ACK_REQ_FDI* متصل می‌کنیم.
۲. تمام خروجی‌های *ACK_REQ* بلاک‌های درایور کانال دیجیتال خروجی را به یک بلاک *OR* با عنوان *ACK_REQ_FDO* متصل می‌کنیم.
۳. تمام خروجی‌های *ACK_REQ* بلاک‌های درایور کانال آنالوگ ورودی را به یک بلاک *OR* با عنوان *ACK_REQ_FAI* متصل می‌کنیم.



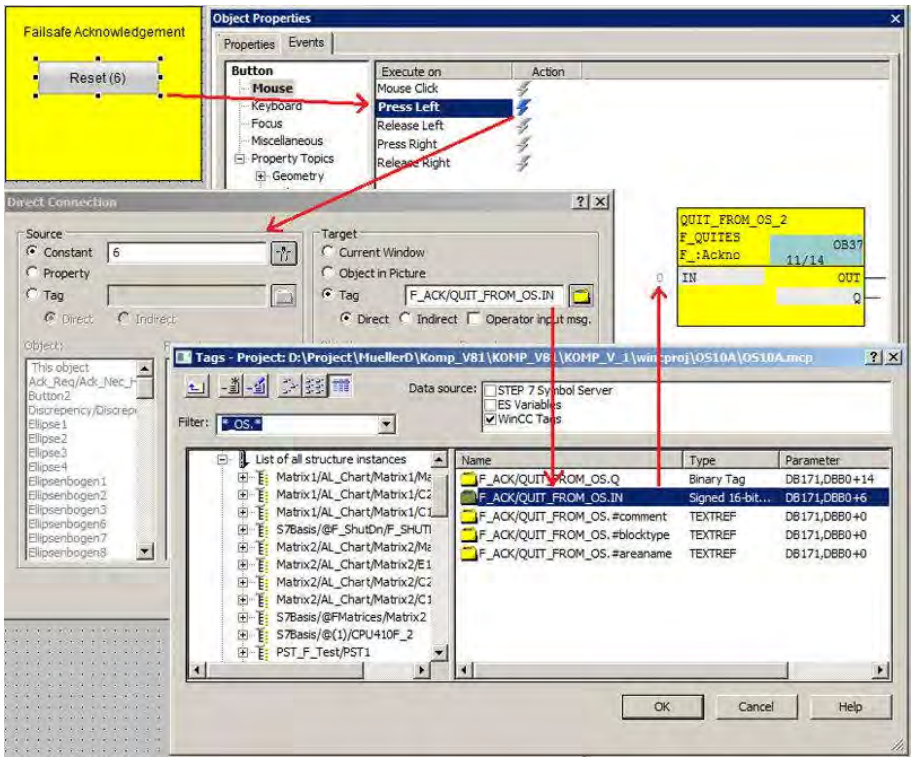
٤. سپس خروجی این سه بلاک *OR* را به یک بلاک *OR* دیگر و خروجی آن را به یک بلاک مانیتورینگ سیگنال دیجیتال *MonDis* متصل می‌کنیم.



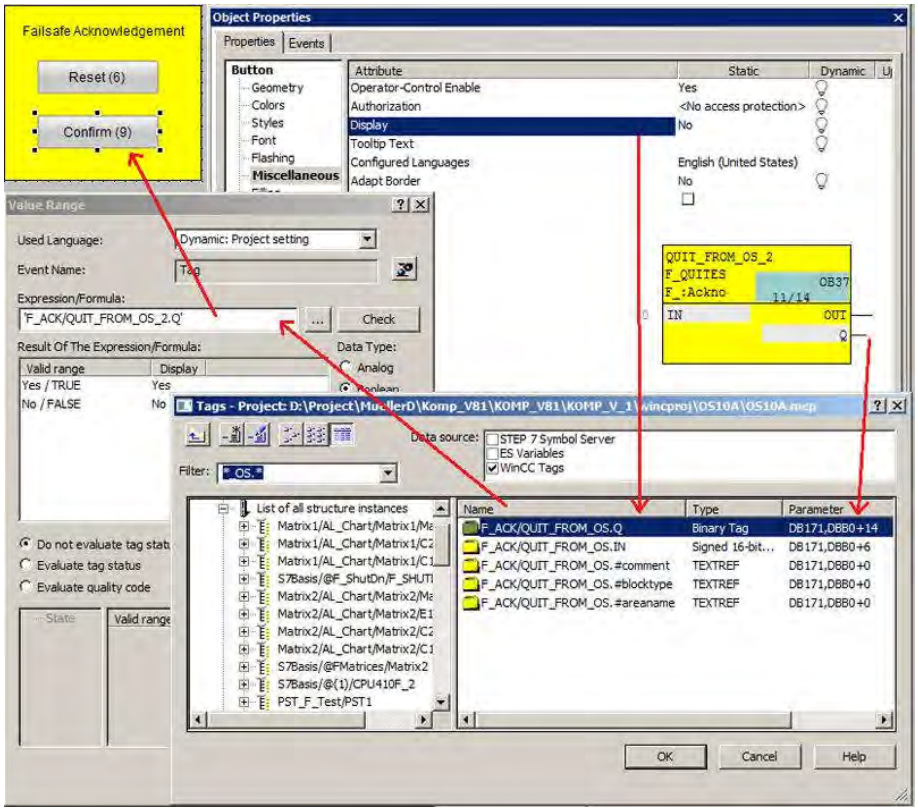
در صورت فعال شدن یک *acknowledgment* (*ACK_REQ=1*)، فیلد *acknowledge* زرد (yellow) شده و دکمه *reset (6)* در OS ظاهر می‌گردد.



۵. برای نوشتن مقدار ٦ در ورودی *IN* بلاک *F_QUITES* بر روی دکمه *reset (6)* کلیک کنید.



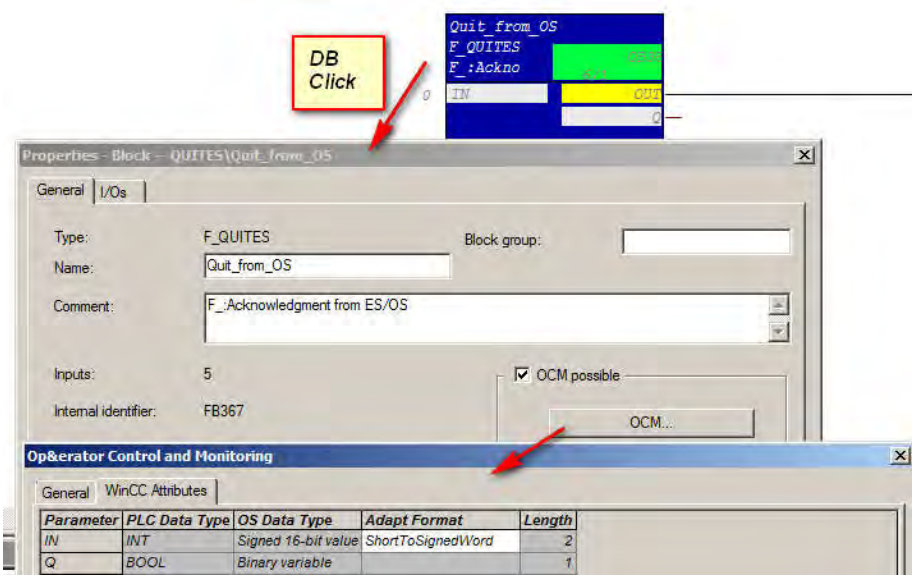
۶. اگر خروجی *Q* بلاک *F_QUITES* ، ست شود، دکمه *acknowledge* دوم یعنی *Reset (9)* ظاهر می‌شود. این خروجی برای مدت ۶۰ ثانیه یک می‌ماند.



اگر مقدار ۹ در بازه یک تا ۶۰ ثانیه به ورودی *IN* بلاک *F_QUITES* نوشته شود، خروجی *OUT* بلاک به مدت یک سیکل به یک ست می‌شود. در آن صورت بلاک‌های درایورهای کانال که ورودی *ACK_REI* آنها به این خروجی متصل شده است، تولید دوباره (*reintegrate*) مقدار در خروجی بلاک را شروع می‌کنند.

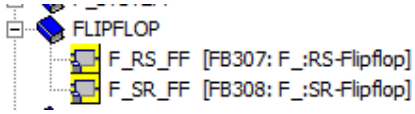
اگر هیچ درخواستی (*acknowledgment prompt*) برای تصدیق نباشد، یعنی ورودی بلاک *ACK_REQ_HMI* یک نشده باشد. دکمه‌های طراحی شده در *OS* مخفی خواهد شد.

نکته : برای این که بتوان از *OS* به مقادیر خروجی و ورودی بلاک *F_QUITES* دسترسی داشت، بایستی در پنجره پراپرتی این بلاک گزینه *OCM* را تیک زد.



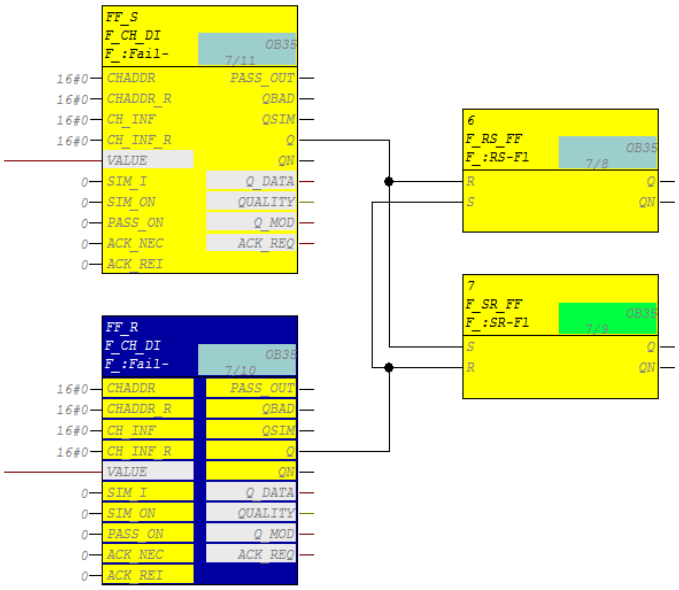
۶-۸- بلاک‌های فلیپ‌فلاپ

این دسته بلاک شامل دو نوع بلاک فلیپ‌فلاپ *RS* و *SR* می‌باشد.



در بلاک *RS* اولویت با ورودی ریست و در بلاک *SR* اولویت با ورودی *S* می‌باشد.

- *F_RS_FF: RS Flip-Flop, resetting dominant*
- *F_SR_FF: SR Flip-Flop, setting dominan*

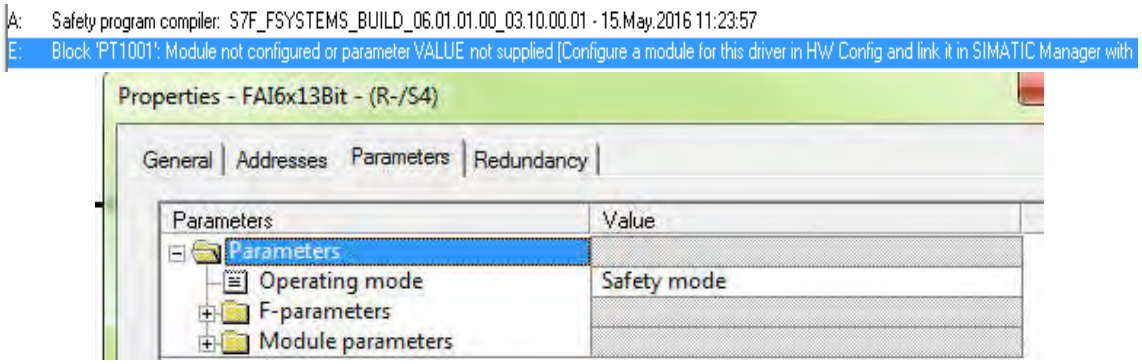


شکل ٦-١٥- بلاک‌های فلیپ فلاپ نوع *F*

۶-۹- کامپایل برنامه F

اگر یک برنامه S7 حاوی چارت‌های CFC دارای بلوک‌های F باشد، این بلاک‌های F در زمان کامپایل چارت‌های CFC، کامپایل شده و اقدامات لازم برای از بین بردن خطاها پیش‌بینی و همچنین بررسی‌های اضافی‌تر در ارتباط با برنامه ایمنی (safetyrelevant checks) انجام می‌گیرد.

نکته : پس از درج ماژول F-I/O بایستی در پنجره پراپرتی آن پارامترهای F تنظیم گردد. به عنوان مثال در صورتی که گزینه Operating Mode برای ماژول F برای مد Safety mode تنظیم نشود. در زمان کامپایل خطای زیر ظاهر خواهد شد.

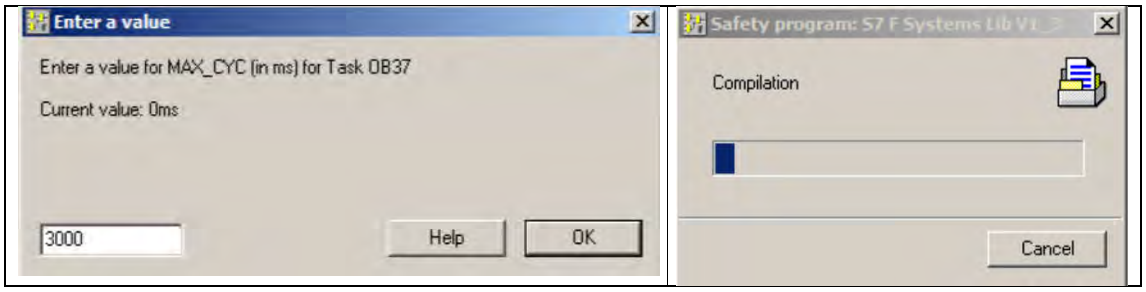


نمایش خطای کامپایل برای ماژولی که پارامترهای آن تنظیم نشده است.

۶-۹-۱- مقداردهی به حداکثر زمان مانیتورینگ سیکل F

Parameterizing the maximum F cycle monitoring time

ماژول F-CPU زمان F-cycle را در هر سیکل اینتراپت OB که حاوی F-runtime groupها می‌باشد، مانیتور می‌کند. وقتی برای اولین بار برنامه S7-F ، کامپایل می‌شود، برای هر OB اینتراپت سیکلیک، پنجره‌ای برای ورود یک مقدار برای حداکثر سیکل زمان (maximum cycle time: MAX_CYC) که می‌تواند بین دو فراخوانی OB سپری شود، نمایش داده می‌شود.

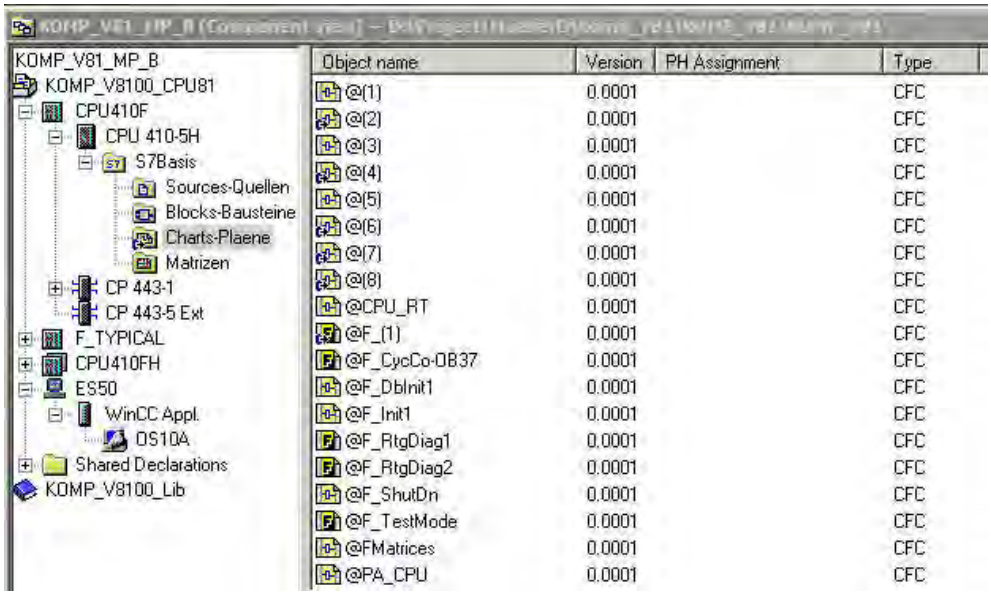


مقدار پیش فرض برای زمان F-Cycle ، ۳۰۰۰ میلی‌ثانیه (حداکثر زمان مانیتورینگ) است.

نکته : مقدار پیش فرض F-Cycle time را می‌توان توسط ورودی MAX_CYC، بلاک F_CYC_CO- OB3x در چارت @F_CycCo-OB3x تغیر داد.

۶-۹-۲- کامپایل برنامه *S7*

در زمان کامپایل، برای برنامه *S7* به صورت خودکار بلاک‌های درایور تشخیص خطا (*diagnostics drivers*) در چارت‌های سیستمی @ و بخش‌های خاص *F* درج می‌شود. بلاک‌های *F-System* در چارت‌های *@F_xxxx* ذخیره می‌شود.



درج بلاک‌های *@F..* در زمان کامپایل برنامه

نکته : درج، اتصال‌بلاک‌ها به هم و تخصیص پارامتر برای بلوک‌های سیستمی نوع *F*، به طور خودکار در طول فرایند تکمیل می‌شود و نباید تغییر کند. همچنین نباید بلاک‌های *F* درج شده به صورت خودکار را در داخل چارت‌ها تغییر داده و یا حذف کنید.

۶-۹-۳- دانلود برنامه به *CPU* و مد ایمنی (*Safety mode and downloading the safety program*)

مد ایمنی برنامه *F* در *F-CPU* را می‌توان به طور موقت غیرفعال و دوباره فعال کرد. این مشخصه امکان ایجاد تغییرات در برنامه ایمنی را در مد اجرا (*RUN*) فراهم می‌کند.

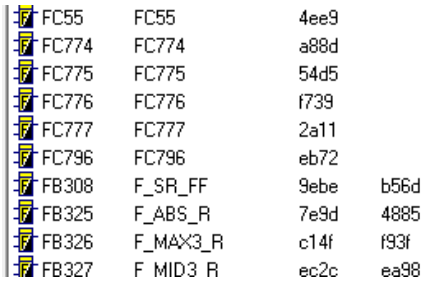
یک سیستم *S7-400 F/FH* که حاوی یک برنامه *fail-safe* می‌باشد. موقع راه‌اندازی به طور خودکار به مد ایمنی می‌رود. در مد ایمنی تمام توابع تشخیص خطا در سیستم حاضر بوده و برنامه *fail-safe* کاربر فعال می‌شود. در این وضعیت امکان تغییر یا اصلاح در برنامه ایمنی در زمانی که مد کاری *RUN* می‌باشد، وجود ندارد.

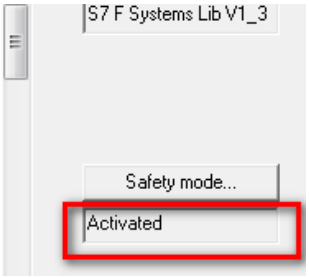
به منظور فراهم کردن امکان ایجاد تغییرات در پارامترهای *fail-safe*، از مدآنلاین *CFC* یا دانلود تغییرات ایجاد شده در برنامه *fail-safe* به *CPU*، بخشی از توابع تشخیصی (*diagnostics*) بایستی خاموش شود. لذا مد ایمنی بایستی قبل از اینکه تغییرات به صورت آنلاین اعمال شود و یا دانلود شود، غیرفعال شود.

قبل از غیرفعال کردن مد ایمنی، باید اطمینان حاصل شود که فرآیند در حالت خطرناک نبوده و توسط یک اپراتور در طول این مدت مانیتور می‌شود. برای دانلود تغییرات انجام شده در برنامه در «مد ایمنی غیرفعال شده» بخش‌های مانیتورینگ خاموش، تغییرات نرم‌افزاری شناسایی شده و *F-STOP* تریگر می‌شود. شناسایی فالت‌های تصادفی در سخت افزار ادامه می‌یابد و توابع تشخیصی برای ماژول‌ها فعال می‌ماند. پردازش برنامه ایمنی همچنان ادامه می‌یابد، تا اطمینان حاصل شود که تقاضای توقف از فیلد منجر به توقف ایمنی می‌شود. همین که تغییرات انجام شد و یا در پایان پروسه دانلود، مد ایمنی باید بلافاصله فعال شود.

۶-۹-۴- غیرفعال کردن مد ایمنی (*Deactivating safety mode*)

در زمان کامپایل یا دانلود مد ایمنی می‌تواند از طریق یک پنجره که اتوماتیک نمایش داده می‌شود، غیرفعال/ فعال شود. این کار همچنین از طریق برنامه *SIMATIC Manager* نیز قابل انجام است. برای این کار، در پنجره *SIMATIC Manager* پس از انتخاب *CPU* که حاوی برنامه ایمنی می‌باشد، گزینه *Options > Edit Safety Program* را اجرا می‌کنیم. فیلد زیر دکمه «*Safety Mode...*» را در پنجره باز شده نشان می‌دهد که می‌توان از طریق آن مد ایمنی را فعال یا غیر فعال کرد.

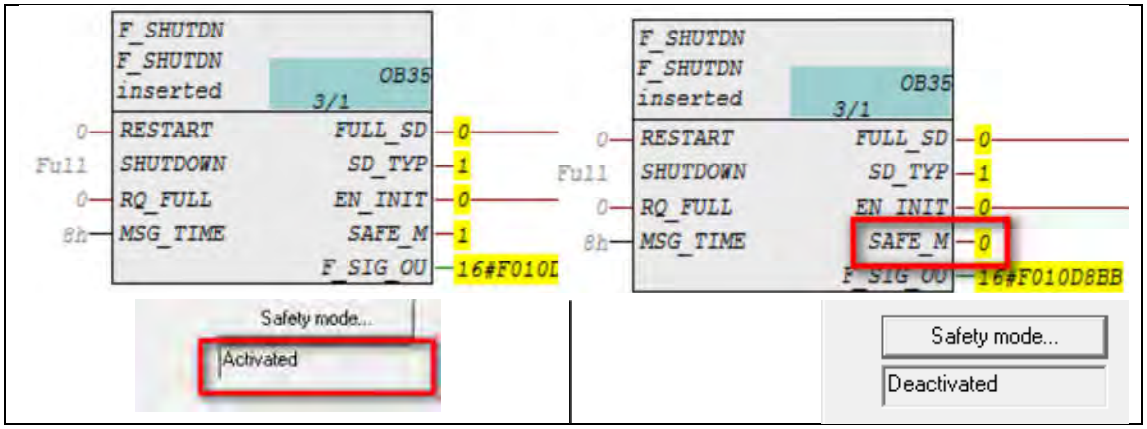




نکته: اگر برنامه ایمنی با برنامه ایمنی موجود در *F-CPU* مطابقت نداشته باشد و یا ارتباط با *F-CPU* قطع شده باشد، در فیلد ذکر شده ظاهر عبارت *unknown* خواهد شد.

در صورتی که بین ایستگاه مهندسی و *CPU* اتصال وجود داشته باشد، وضعیت فعلی برنامه ایمنی نمایش داده می‌شود و می‌توان با کلیک روی دکمه *Safety Mode...* می‌توان مد ایمنی را تغییر داد. ولی قبل از تغییر مد ایمنی پنجره دیگری باز می‌شود که تغییر وضعیت را اعلان می‌کند.

مد ایمنی در بافر تشخیص عیب *CPU* وارد شده و به *OS* اعلام می‌شود و می‌توان در چارت *F_Shutdn* در خروجی *SAFE_M* بلوک *F_SHUTDN* چک کرد.



۶–۹–۵– دانلود برنامه F (*Downloading the safety program*)

پس از کامپایل برنامه *F*، می‌توان برنامه را به *CPU* دانلود کرد. بسته به این که مد ایمنی فعال یا غیرفعال شده باشد، تغییرات برنامه را به مطابق جدول زیر می‌توان به *CPU* دانلود کرد.

Download	AS in STOP Single/H-system	AS in RUN Safety mode active	AS in RUN Safety mode inactive
... of the entire program	Possible/Possible	Not possible	Not possible
... of changes in the standard program	Possible/Not possible	Possible	Possible
... of changes in the safety program	Possible/Not possible	Not possible	Possible

قبل از این که بتوان کل برنامه یا تغیرات آن را به *CPU* دانلود کرد، بایستی شرایط زیر فراهم شود:

- بایستی پیکربندی سخت‌افزار از پنجره *HW config* به *CPU* دانلود شده باشد.
- برنامه کاربر بدون خطا کامپایل شده باشد.
- کاربر حق دسترسی لازم را داشته باشد. یعنی پسورد *CPU* و برنامه *F* را داشته باشد.
- یک اتصال آنلاین بین *CPU* و ایستگاه *EWS* وجود داشته باشد.

قوانین دانلود

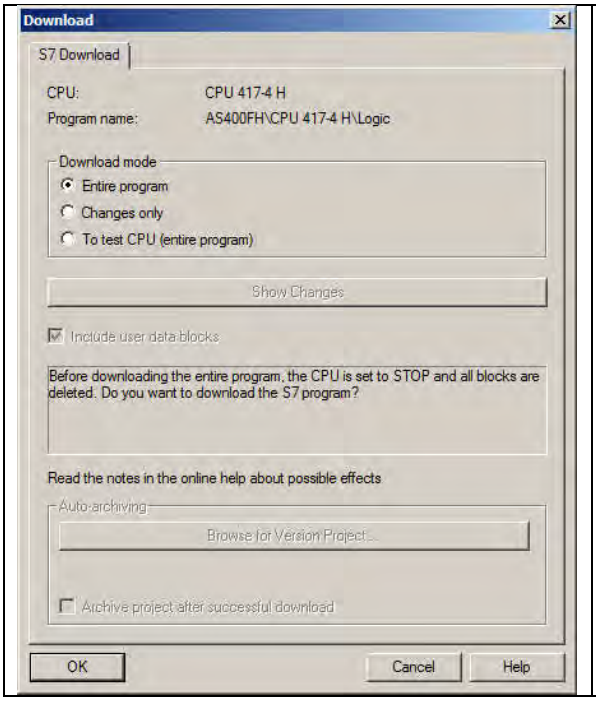
۱. قبل از دانلود برنامه ایمنی، آزمون *consistency* را انجام دهید. هر دو *Signature* در بخش اطلاعات برنامه و در فولدر ایمنی باید یکسان باشد.

۲. *F-runtime group*های خالی ممکن است پس از حذف توابع ایمنی ایجاد شوند. قبل از کامپایل آنها را حذف کنید. برای انجام این کار در *PCS 7 V7.0* و بالاتر، فرمان زیر را در ویرایشگر *CFC* اجرا کنید:

Edit > Delete Empty Runtime Groups

۳. برنامه ایمنی را تنها می‌توان از ویرایشگر *CFC* یا از *SIMATIC Manager* از پوشه *Chart* دانلود کرد.

٤. همین که برنامه *F* را دانلود کردید. نیاز است که *Collective signature* بررسی شود.

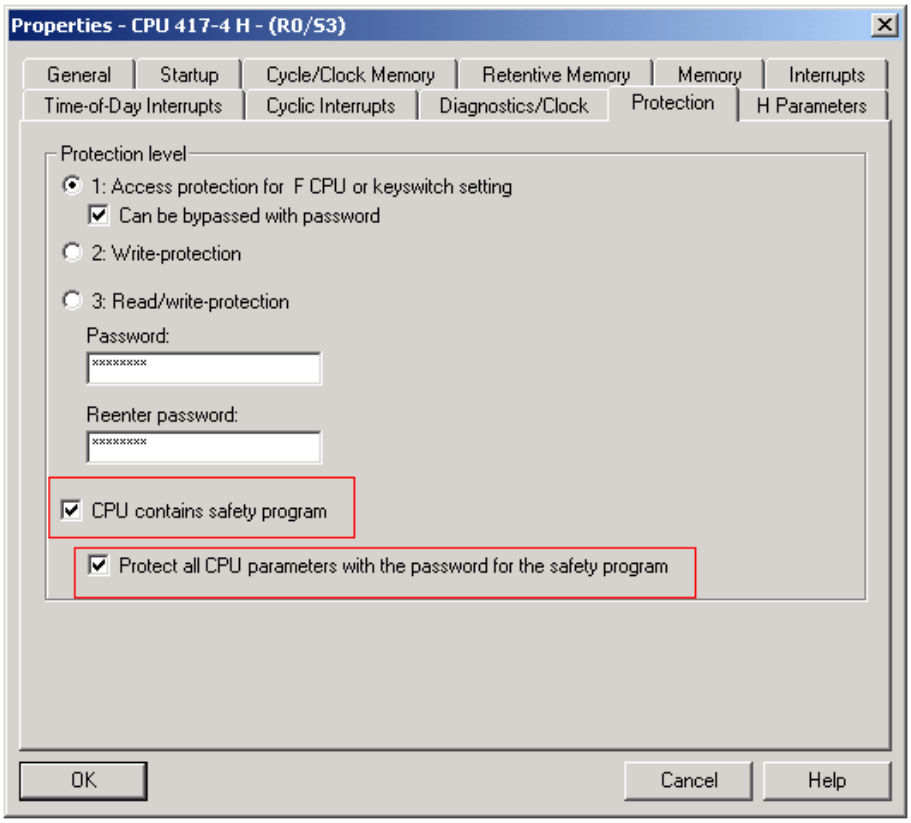


۶-۹-۶- رمز حفاظتی برای توابع مرتبط با ایمنی و تنظیمات *CPU*

اگر در یک پروژه هر دو بخش استاندارد و *F* برنامه کاربر در یک *CPU* اجرا می‌شود، تغییر در بخش استاندارد را می توان بدون نیاز به وارد کردن رمز عبور *F-password* کامپایل کرد. زیرا فرض می‌کند که هیچ تغییری در برنامه ایمنی صورت نگرفته است.

برای فعال شدن توابع ایمنی موجود در سیستم عامل *H-CPU*، در زمان دانلود برنامه به *CPU*، پنجره‌ای باز می‌شود که نیاز است رمز عبور تعریف شده در پنجره پراپرتی برای دسترسی به *CPU* وارد شود. همانطور که گفته شد این رمز برنامه *F* را در مقابل دسترسی‌های غیرمجاز محافظت می‌کند.

برای تعریف این رمز قبل از برنامه‌نویسی بلاک‌های *F* و در زمان پیکربندی سیستم *F*، مطابق شکل زیر تعریف کرده و همچنین گزینه *CPU contains safety program* را نیز برای فعال کردن قابلیت اجرای بلاک‌های *F* توسط *CPU* تیک می‌زنیم. همچنین گزینه حفاظت از پارامترهای *CPU* با رمز عبور را برای برنامه ایمنی فعال می‌کنیم.



وقتی برای اولین بار برنامه *F* کامپایل می‌شود، پنجره تعریف رمز عبور برای برنامه *F* نمایش داده می‌شود. این رمز عبور در هنگام باز کردن چارت‌ها و بلاک‌های *F* درخواست می‌شود. به عبارت دیگر در *CPU*های *S7* که قادر به اجرای برنامه *fail-safe* هستند، برنامه *F* و پارامترهای ماژول‌های *F* در برابر تغییرات غیر مجاز، با یک رمز عبور محافظت می‌شود. این رمز موقع دسترسی به بخش‌های *fail-safe* سیستم، به طور اتوماتیک پرسیده می‌شود.

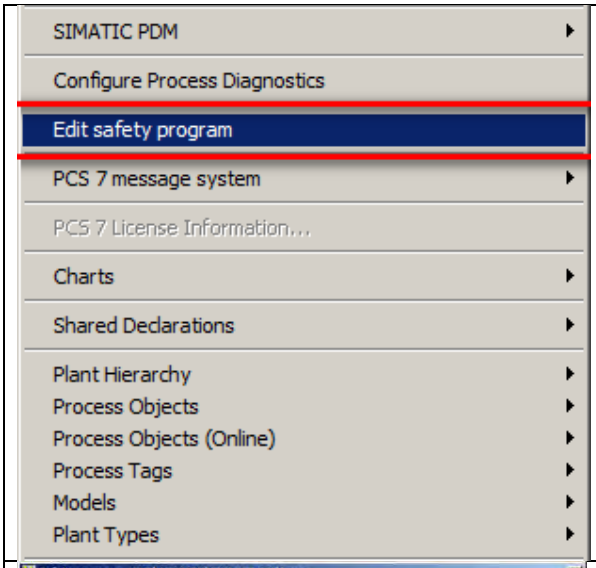
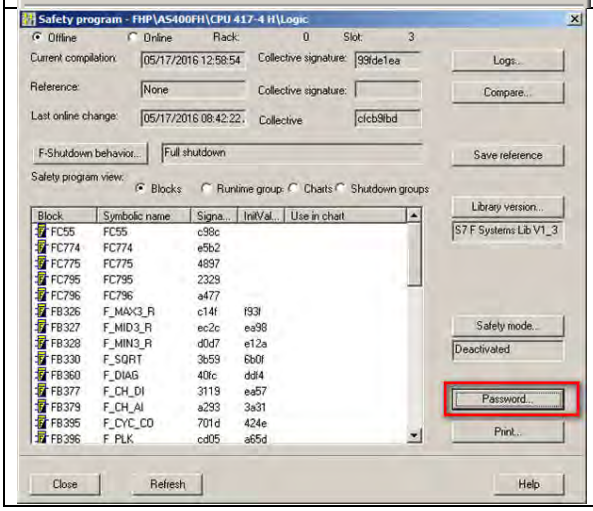
برای مثال :

- هنگام باز کردن یک چارت *fail-safe CFC* ؛
- هنگام کامپایل تغییرات در برنامه ایمنی؛
- و یا در هنگام باز کردن خواص یک ماژول *F*؛

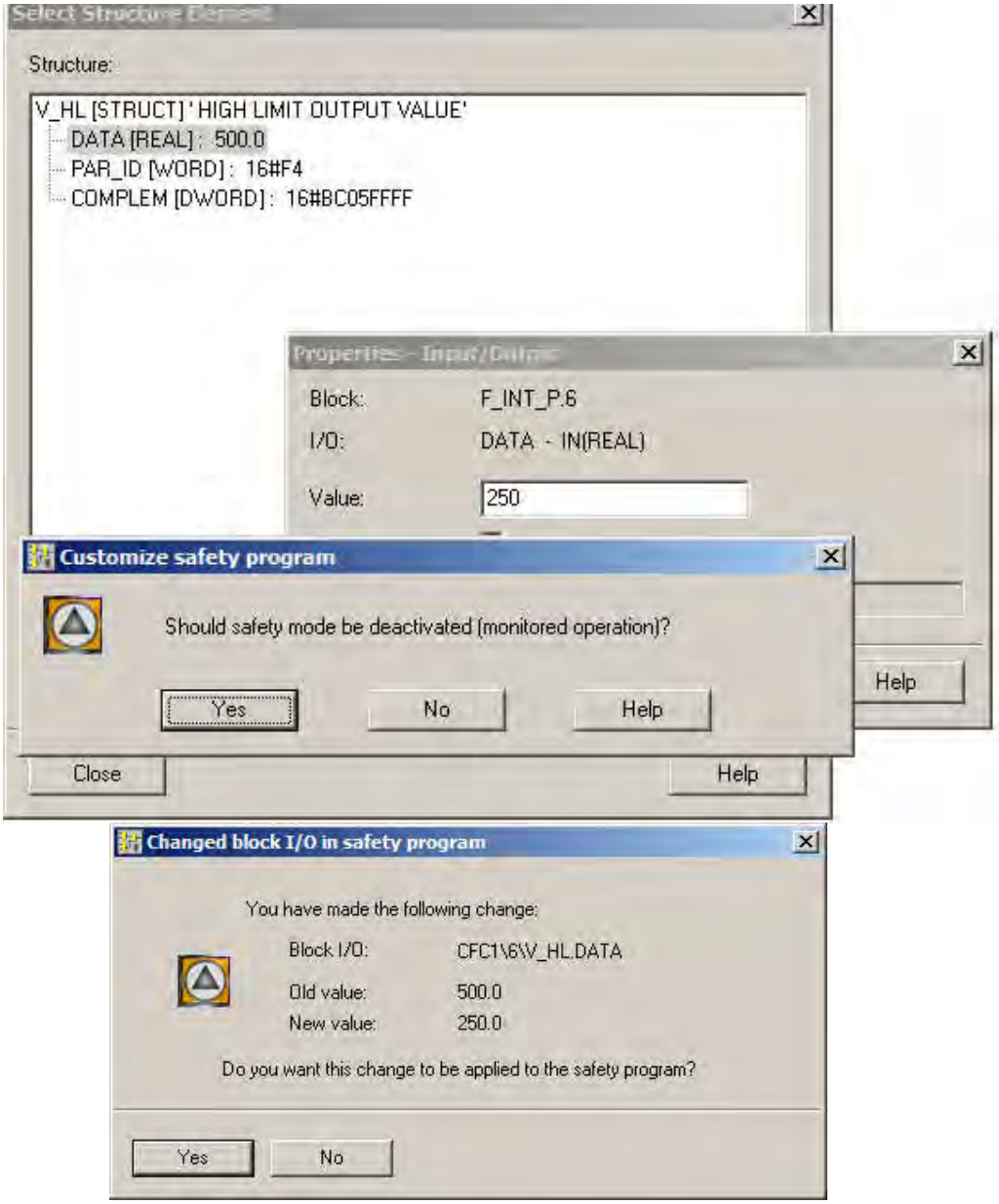
درخواست رمز موقع دسترسی به چارت‌ها یا بخش‌های مختلف برنامه *fail-safe*

درخواست ورود رمز برای اولین بار در زمان کامپایل، در صورتی که در پنجره *Edit safety program* تعریف نشده باشد.

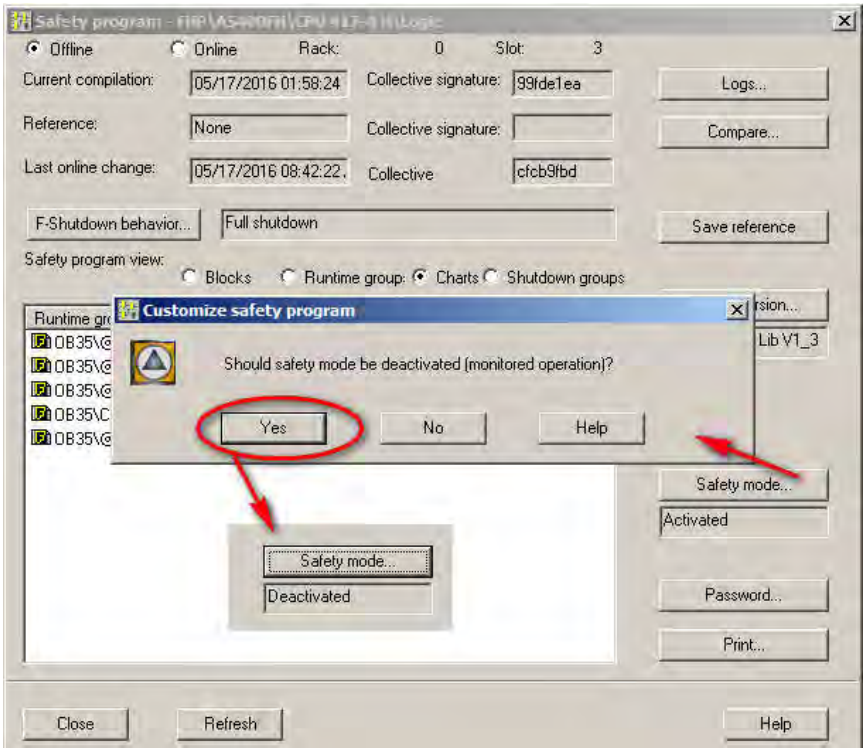
اگر پس از نمایش پنجره درخواست رمز، کلمه عبور وارد شود، به مدت یک ساعت همچنان در سیستم معتبر بوده و در دفعات بعد پرسیده نمی‌شود. در صورت اتمام کار توصیه می‌شود که پریود زمانی اعتبار رمز عبور ریست بشود. برای این کار، مطابق شکل زیر در پنجره *SIMATIC Manager* پس از انتخاب *CPU* که حاوی برنامه ایمنی می‌باشد، گزینه *Options > Edit Safety Program* را اجرا می‌کنیم . از پنجره باز شده با کلیک روی دکمه *Password* پنجره *Create Password for Safety Program* باز کنید. سپس در این پنجره با کلیک روی دکمه *Cancel* بازه اعتبار رمز به صفر تنظیم می‌شود. در این پنجره همچنین می‌توانید رمز عبور را تغییر دهید. در شکل زیر مقدار زمان باقی‌مانده برای اعتبار پسسورد، ٤٧ دقیق می‌باشد.

	در صورتی که بخواهیم این رمز را تغییر دهیم، در پنجره *Simatic Manger* از منوی *Option* گزینه *Edit safety program* را انتخاب می‌کنیم. و یا از *Tool bar* آیکون را کلیک می‌کنیم
	در پنجره باز شده از طریق دکمه *Password* پنجره تغییر رمز را باز کرده و رمز جدید را تعریف می‌کنیم.

در این حالت با هر بار ورود مقدار به یک پارامتر و اعمال آن به *CPU* در حالت *Runtime* در زمان دیباگ برنامه، پنجره‌ای باز می‌شود که اعلام می‌کند که در زمان تغییر بایستی *safety mode* بایستی غیرفعال شود.



در صورتی که بخواهیم در زمان تست برنامه، مدام پنجره پرسش غیرفعال کردن مد ایمنی نمایش داده نشود، از پنجره *Edit safety program* مد ایمنی را مطابق شکل زیر غیر فعال می‌کنیم.



موقع دانلود برنامه به *CPU* رمز تعریف شده در پنجره پراپرتی *CPU* پرسیده می‌شود و موقع باز کردن چارت *F-CFC* پنجره ورود رمز تعریف شده برای برنامه ایمنی پرسیده می‌شود.

فصل – ۷ پیکربندی ماژول‌های *IO* ریدادانت در *PCS7*

نکات

۱. کارت‌های *Fail Safe* امکان مقایسه یک سیگنال را با دو کانال دارند. مثلا اگر *DI* مد نظر باشد سیگنال آن به دو ترمینال از کارت متصل شده و مقایسه می شود.

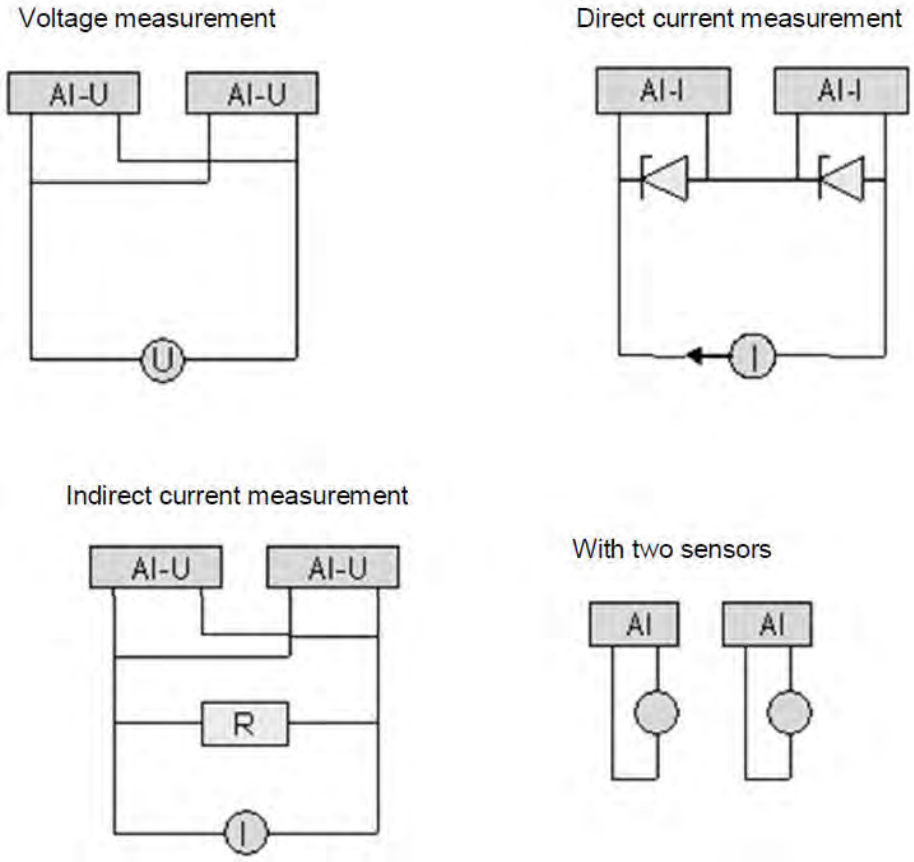
۲. نحوه *Wiring* کارت‌های *Fail Safe* نسبت به کارت‌های استاندارد متفاوت است. از نظر ظاهر نیز مقداری عریض‌تر از کارت های استاندارد هستند.

۳. ۱. کارت‌های *Fail Safe* به رنگ زرد می‌باشند.

۷-۱-۱- ورودی انالوگ

اتصال انکودر (*Encoder connectio*)

ماژول رودی آنالوگ *SM331; AI 8x16 bit* می‌تواند در یک پیکربندی ریداندانت با یک یا دو انکودر (*2oo2 structure*)استفاده شود.



به منظور رسیدن به حداکثر در دسترس‌پذیری توصیه می‌شود که از انکودرهای ریداندانت استفاده شود.

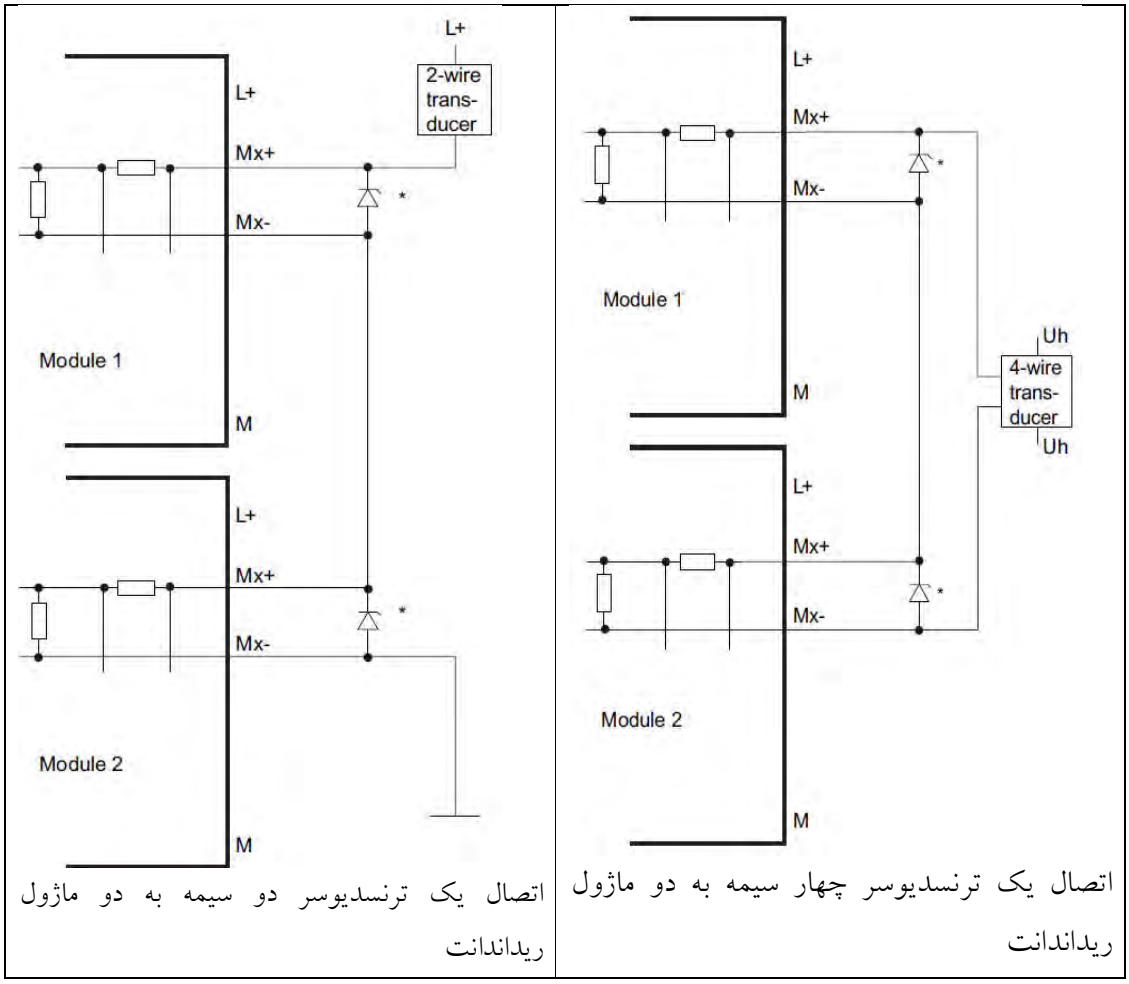
نکته: هنگام استفاده از ماژول‌های آنالوگ *HART* ریداندانت، تنها یک ترنسمیتر اندازه‌گیری (*Encoder*) می تواند متصل شود.

اتصال سنسور/عملگر در مد ریداندانت (*Connecting sensors/actuators in redundant mode*)

بکار گیری ماژول‌های آنالوگ ورودی/خروجی هارت در مد ریداندانت تنها در ساختار *Remote I/o* امکان پذیر است. ماژول‌های هارت دارای پارامترهای «*Primary Master*» و «*Secondary Master*» برای این

منظور هستند. این نوع تخصیص پارامتر برای فعال کردن ارتباطات هارت همزمان به یک وسیله فیلد با استفاده از طریق هر دو ماژول می‌باشد. ماژول با آدرس پایین همیشه *Primary Maste* می‌باشد.

۱. در مد ریداندانت، مسترهای دیگر مانند دستگاه هارت دستی، نمی‌تواند وصل شود.
۲. مد *HART-Fast-Mode* در مد ریداندانت امکان پذیر نیست.
۳. کالیبراسیون کاربر در مد ریداندانت امکان پذیر نیست.
٤. در مد ریداندانت، گزینه *substitute value behavior of the current outputs* به طور خودکار به جریان و ولتاژ صفر (*OCV*) تنظیم می‌شود. به هر حال، اگر *CPU* در مد توقف باشد، جریانی حدود ۱۱۵ میکروآمپر در خروجی هر کانال خواهد بود.
۵. زمانی که ماژول‌های آنالوگ هارت در مد ریداندانت عمل می‌کنند، یک ترنسدیوسر دو سیمه باید به صورت یک ترنسدیوسر چهار سیمه متصل شوده و همچنین به صورت یک ترنسدیوسر چهار سیمه در *HW Config* پیکربندی شود. ترمینال‌های ۱۰ و ۱۱ در *Front panel connector* نباید متصل شوند.
٦. در مد ریداندانت، افت ولتاژ در هر دو ماژول باید رعایت شود. برای اطمینان از تامین ولتاژ کافی به ترنسدیوسر، بایستی افت ولتاژ در هر دو ماژول و افت ولتاژ در سیم کشی و خود ترنسدیوسر باید رعایت شود (اتصال سری). با یک جریان سنسور ۲۲ میلی آمپر، افت ولتاژ حدود V ۳,۳ در هر ماژول رخ می‌دهد. اگر شما از مدار با دیود زنر که در شکل زیر نشان داده شده است، استفاده می‌کنید. اگر دارید ماژول را جایگزین می‌کنید، توجه داشته باشید که افت ولتاژ در ماژول برداشته شده از اسلات (*removed module*) برابر ولتاژ دیود زنر (*5.1V*) و افت ولتاژ در ماژول قرار داده شده در اسلات *3.3V* است.
۷. وقفه‌های سخت افزاری توسط بلوک‌های کتابخانه "*RedLib*" پشتیبانی نمی‌شود. اگر مایل به استفاده از وقفه‌های سخت افزاری باشید، بایستی ارزیابی در سطح کاربر را پیاده‌سازی کنید.

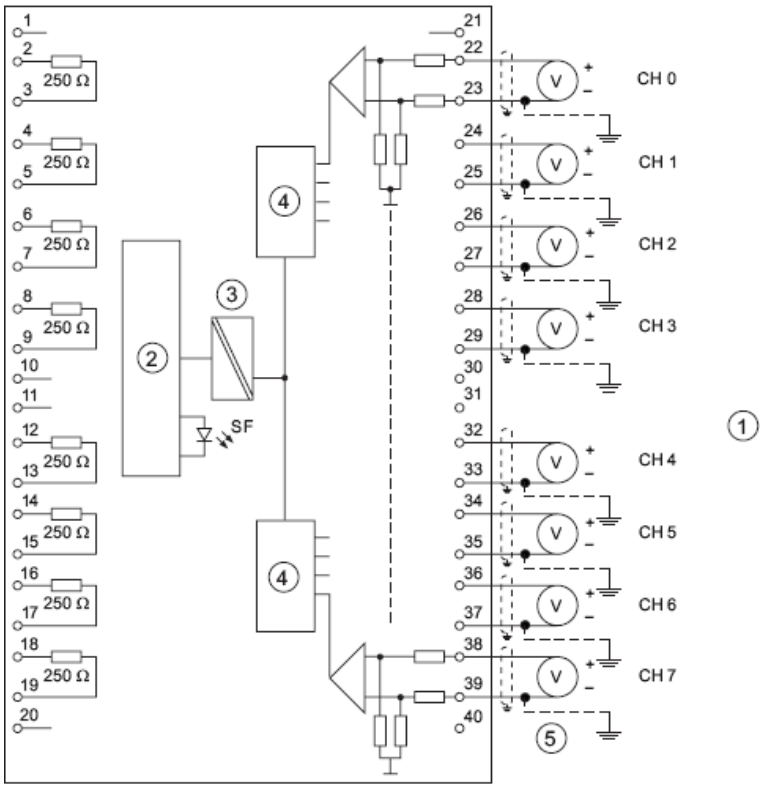


اتصال یک ترنسدیوسر دو سیمه به دو ماژول ریداندانت

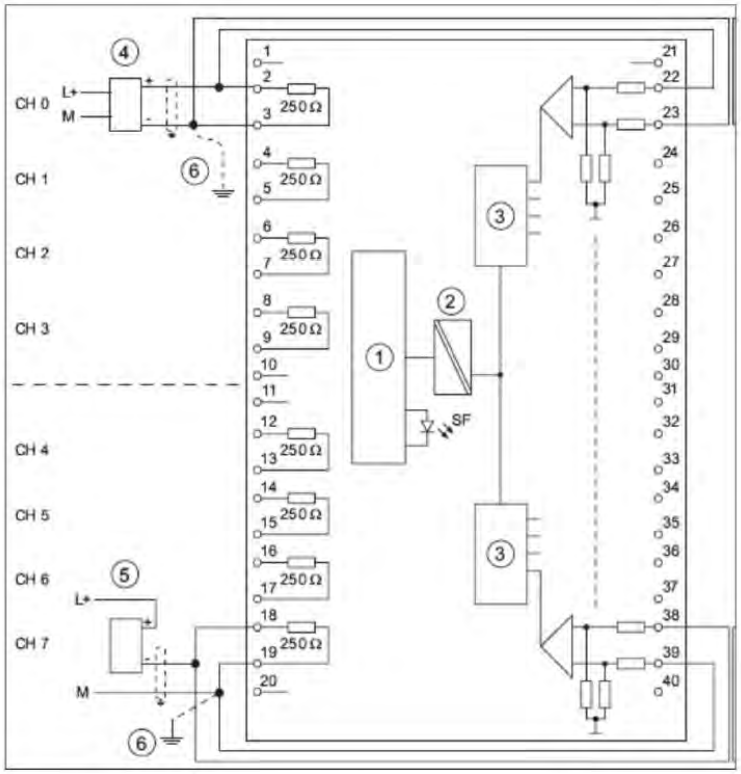
اتصال یک ترنسدیوسر چهار سیمه به دو ماژول ریداندانت

- *Zener diode 5.1 V (e.g., BZX85C5V1*

دیاگرام مداری اتصالات ماژول *SM331* (*Connection and principle circuit diagram*)

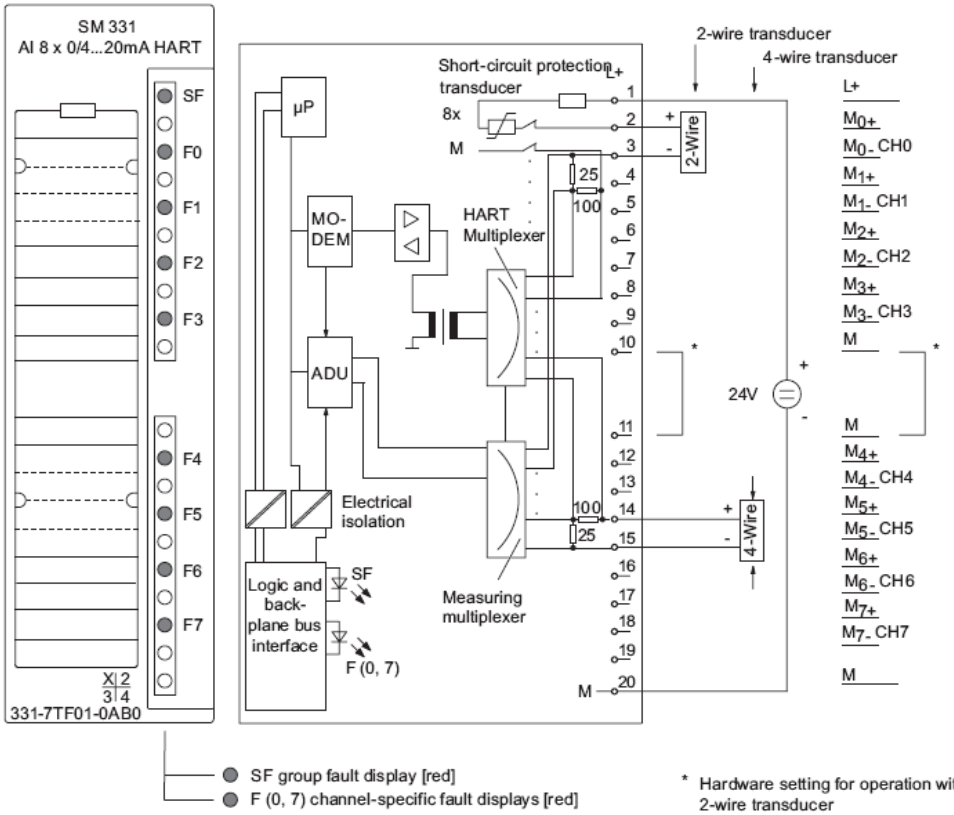


دیاگرام مداری اتصالات ماژول *SM331 AI 8x16Bit (6ES7 331-7NF00-0AB0)* برای اتصال ولتاژی



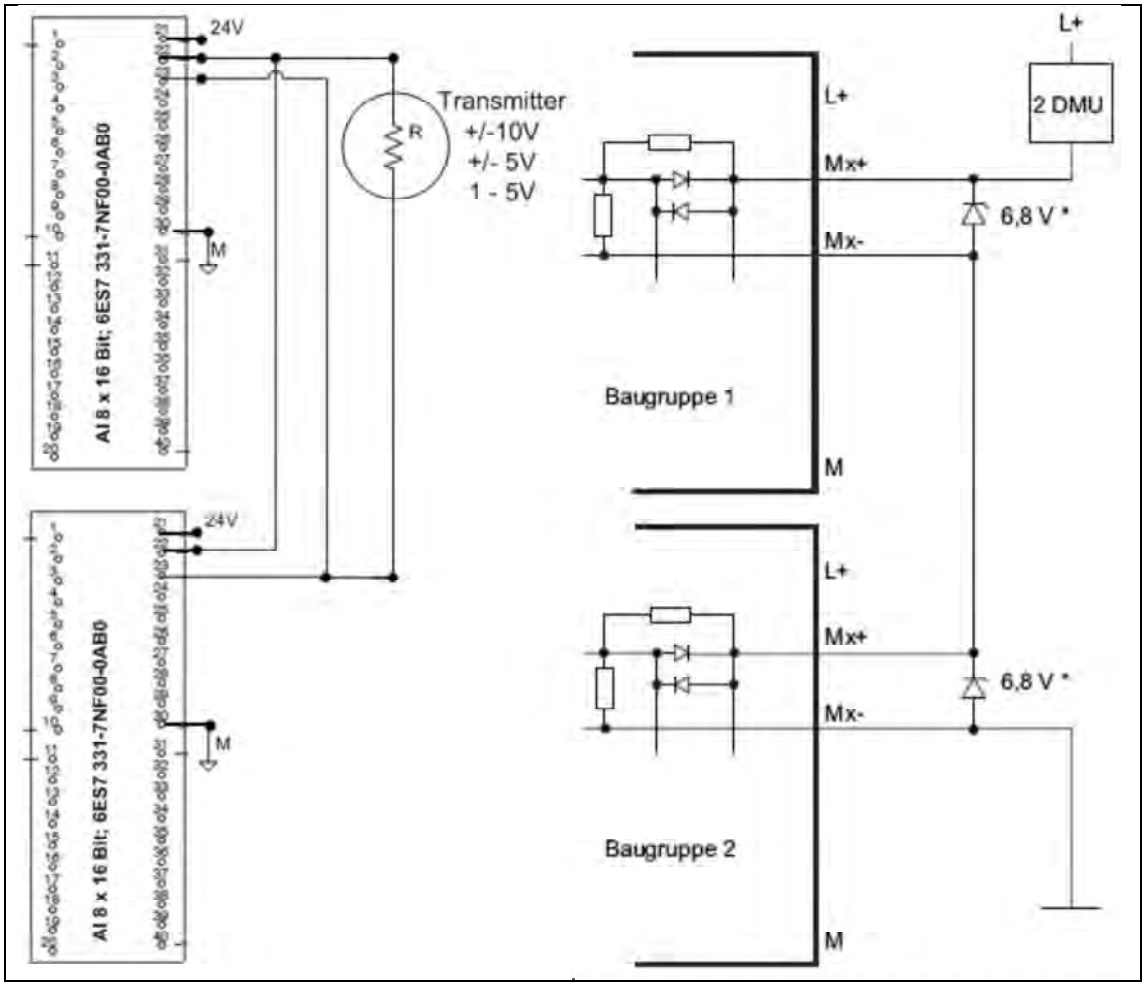
Number	Description
1	Backplane bus connection
2	Electrical isolation
3	Analog-digital converter (ADU)
4	CH 0 for 4-wire measuring transformer
5	CH 7 for 2-wire measuring transformer (with external supply)
6	Equipotential bonding

دیاگرام مداری اتصالات ماژول *SM331 AI 8x16Bit (6ES7 331-7NF00-0AB0* برای اتصال جریان



دیاگرام مداری اتصالات ماژول هارت *SM331 AI 8x0/4...20 mA (6ES7 331-7TF01-0AB0)*

نکته : در حالت ریدادانت نباید دو پایه ۱۰ و ۱۱ به هم متصل شوند.

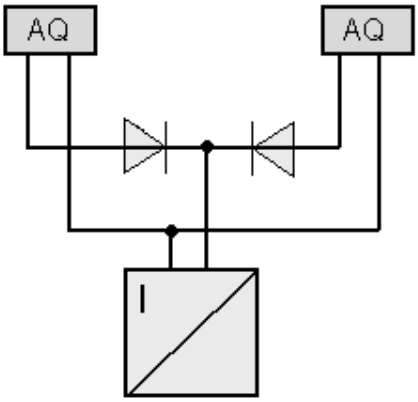


The wired Z-diodes are necessary if after pulling a module the system shall continue to run.

۷-۱-۲- خروجی آنالوگ

کنترل المان کنترل نهایی (*Controlling the final control element*)

برای کنترل تحمل پذیر خطا (*fault-tolerant*) در یک عنصر کنترل نهایی، دو خروجی به صورت موازی (ساختار 2oo2) توسط دو ماژول *SM 332; AO 8x12bit* و از طریق دو دیود روشن (*switched*) می‌شود. توجه شود که تنها ماژول‌های آنالوگ خروجی با خروجی جریان می‌تواند به صورت ریداندانت عمل کند (۰ تا ۲۰ میلی‌آمپر یا ٤ تا ۲۰ میلی‌آمپر).



دو خروجی آنالوگ، نیمی از مقدار خروجی را تولید و در پایه خروجی‌شان قرار می‌دهند. اگر یکی از ماژول‌ها با شکست مواجه (*fails*) شود، باز هم خروجی غیر فعال مقدار کامل را تحویل می‌دهد.

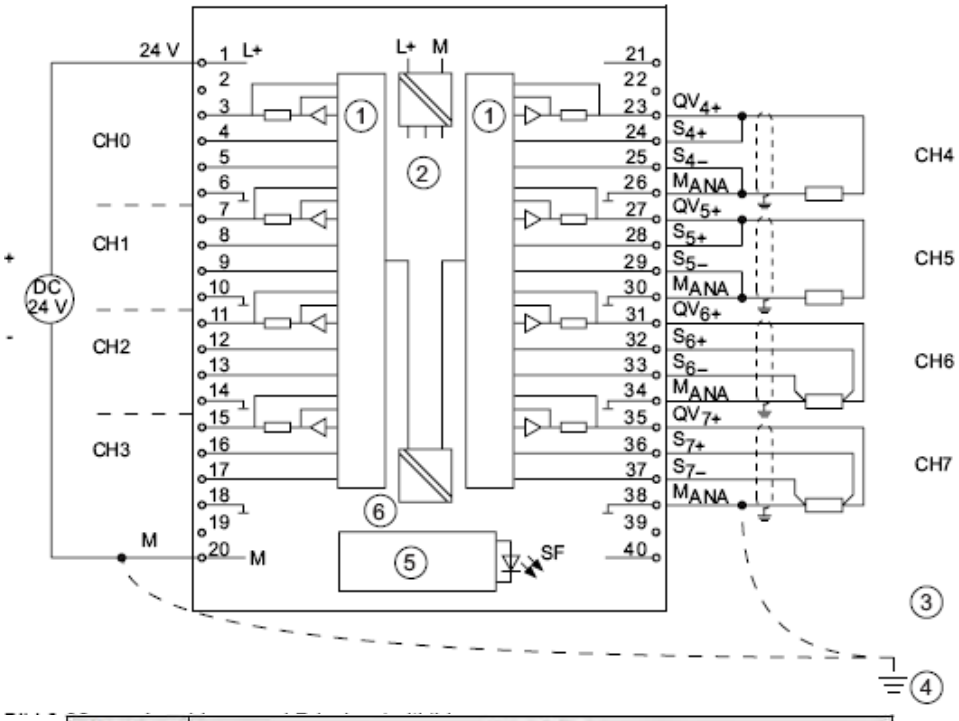
Both outputs output half of the value. If one of the modules fails, the still inactive output delivers the complete value.

دیاگرام اتصالات ماژول خروجی آنالوگ

Connection and principle circuit diagram of SM 332; AO 8 x 12 bit (voltage output)

شکل زیر دیاگرام مدار یک ماژول خروجی آنالوگ *SM332 AO 8x16Bit* (*6ES7 332-5HF00-0AB0*) را برای خروجی ولتاژ در دو حالت نشان می دهد.

- اتصال ۲ خط بدون مقاوت جبران ساز خط(*compensation of line resistors*)
- اتصال ٤ خط با مقاوت جبران ساز خط



Number	Description
1	DAU
2	Internal supply
3	Equipotential bonding
4	Functional earth
5	Backplane connection
6	Electrical isolation

شکل ۷-۱- دیاگرام اتصالات ماژول خروجی آنالوگ *6ES7 332-5HF00-0AB0*

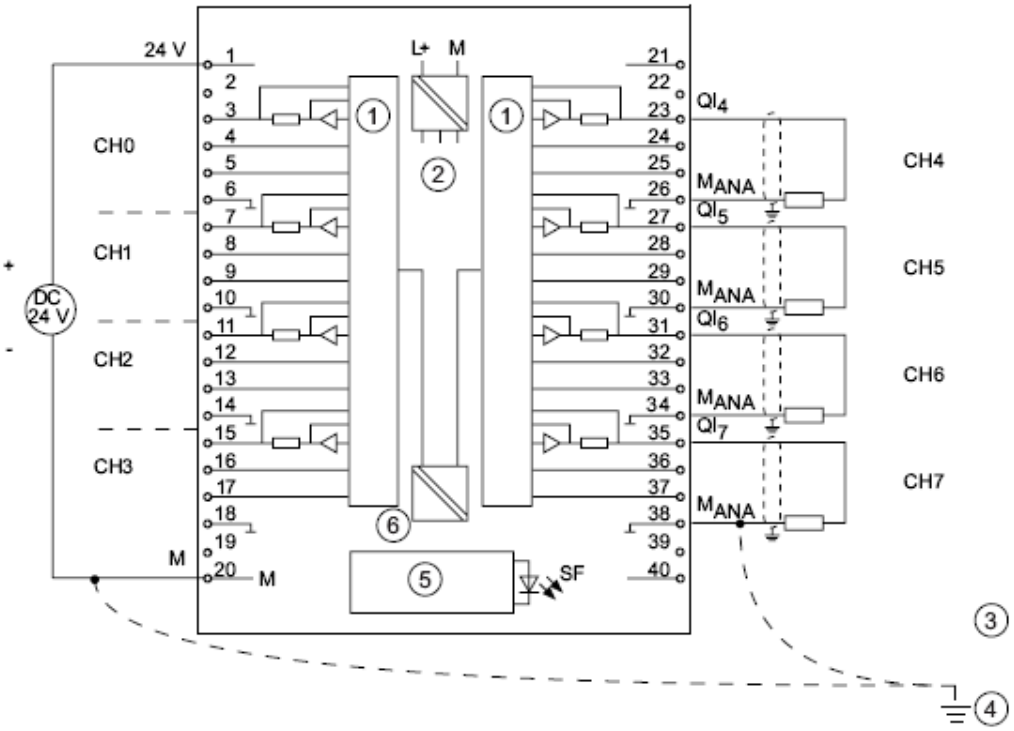
نکته : ارت تمیز و ارت کثیف (*Protective & Clean earth*)

۱. زمین حفاظتی که زمین کثیف (*dirty earth*) نیز نامیده می‌شود، برای محافظت از پرسنل و تجهیزات استفاده می‌شود. زمین حفاظتی برای حفاظت از پرسنل در برابر شوک الکتریکی در زمان وقوع یک فالت زمین در نظر گرفته شده است. همواره از لوازم الکتریکی از طریق هادی زمین یک اتصال به ترمینال زمین اصلی در اتاق تامین برق وجود دارد. به طوری که جریان ناشی از فالت از طریق این هادی به جرم زمین جاری خواهد شد.

۲. در این جا زمین *Functional* همان زمین *Clean* است. که به عنوان یک زمین مرجع برای سیگنال های دیجیتال و آنالوگ استفاده می‌شود. این زمین همچنین برای کاهش اعوجاج در سیگنال لازم است. زمین کاربردی (*Functional earth*) برای مواردی چون کاهش نویز فرکانس رادیویی، فیلتر کامپیوتر به منظور

افزایش وضوح سیگنال و یا دیگر تجهیزات که همواره جریان نشتی *10mA* دارند، در نظر گرفته شده است. زمین کاربردی باید از زمین حفاظت به جز در نقطه زمین اتصال ترمینال اصلی جدا شوند.

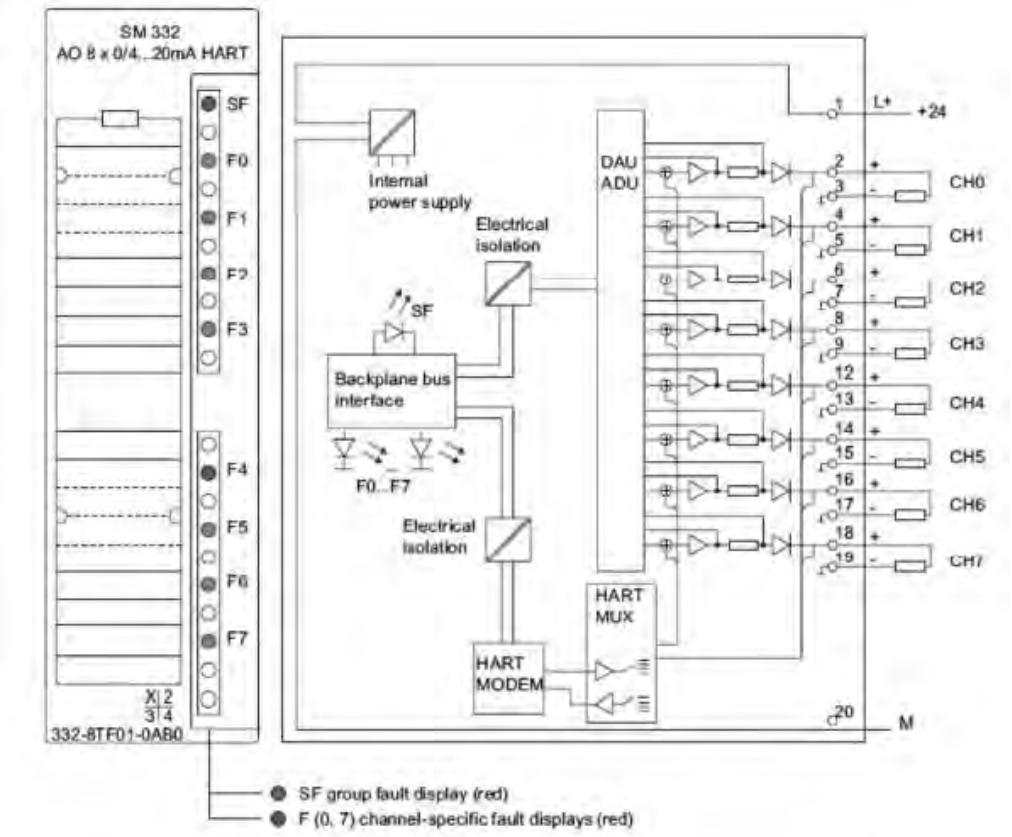
شکل زیر دیاگرام مدار یک ماژول خروجی آنالوگ *SM332 AO 8x16Bit* را برای خروجی جریان نشان می دهد.



شکل ۷-۲- دیاگرام سیم‌بندی ماژول آنالوگ خروجی جریان (*6ES7 332-5HF00-0AB0*)

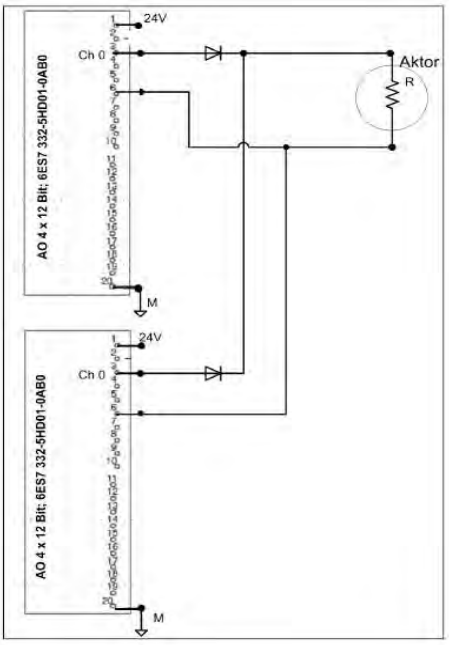
دیاگرام اتصالات ماژول آنالوگ خروجی هارت *0/4...20mA*

Connection and principle circuit diagram SM 332; AO 8 x 16Bit, 0/4...20mA HART

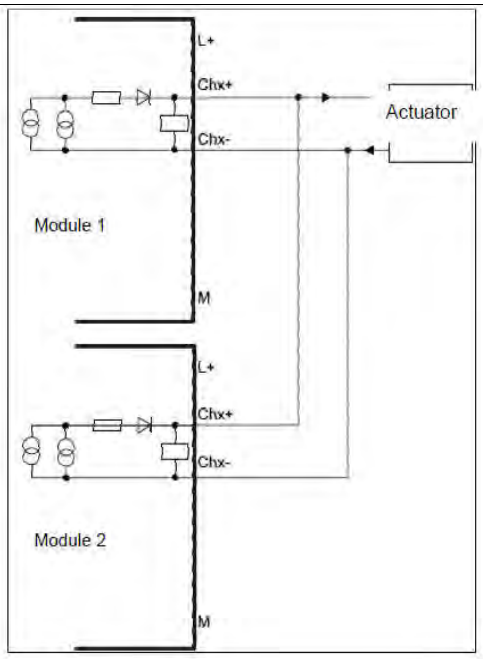


شکل ۷-۳- دیاگرام اتصالات ماژول آنالوگ خروجی هارت (*6ES7 332-8TF01-0AB0*)

شکل‌شکل ۷-۵ مثالی از سیم بندی یک عملگر به دو ماژول غیرهارت ریداندانت آنالوگ خروجی (*AO*) نوع جریان یا ولتاژ و شکل ۷-٤ اتصال یک عملگر به دو کانال از دو ماژول ریداندانت آنالوگ خروجی هارت را نشان می‌دهد.



شکل ۷-۵- مثالی از سیم بندی یک عملگر به دو ماژول غیرهارت ریداندانت آنالوگ خروجی *AO*



شکل ۷-٤- اتصال یک عملگر به دو کانال از دو ماژول ریداندانت خروجی آنالوگ هارت

دیودهای مناسب برای این منظور به عنوان مثال سری *1N4007 ... 1N4003* می‌باشد. یا هر دیود با مشخصه $U_r >= 200\ V$ و $I_F >= 1\ A$ را می‌توان استفاده کرد.

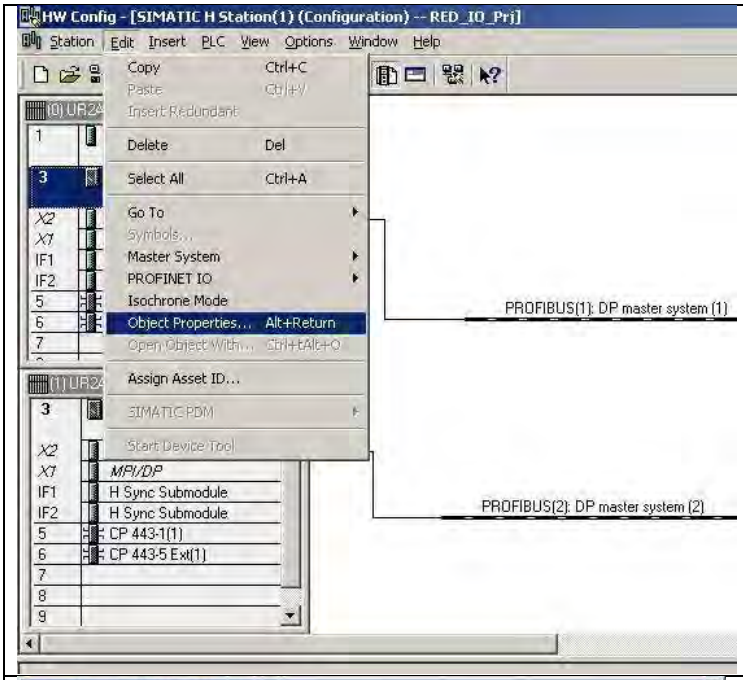
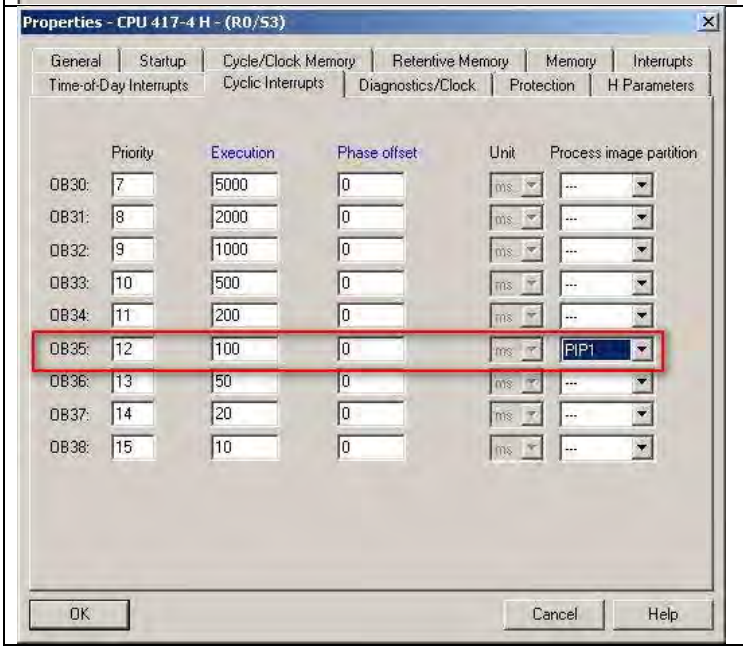
۷-۲- پیکربندی *IO* ریداندانت در *Step7*

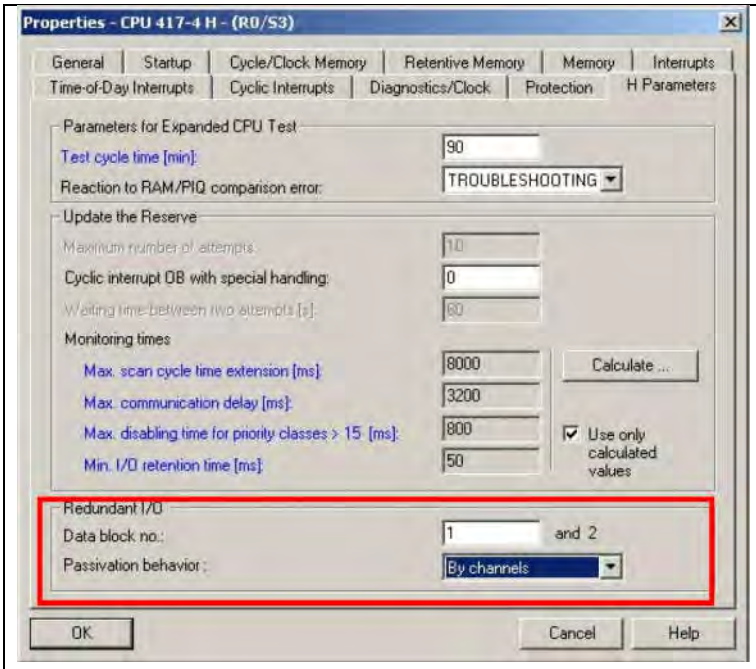
این بخش به توصیف پیکربندی اجزاء مختلف برای پیکربندی ریداندانسی *IO* در *PCS 7* پرداخته است. فرض بر این است که یک پروژه *PCS 7* از نوع *H Station* با استفاده از ویزارد پروژه، ایجاد شده است. مراحل پیکربندی زیر تشریح می‌شود:

- *Setting the CPU (H parameters)*
- *Configuring the first ET 200M (interface module IM 153-2)*
- *Configuration of the individual signal module IOs*
- *Configuring the second ET 200M by copying*
- *Redundancy settings at the signal modules*
- *Symbolic name assignment*
- *CFC configuration*

۷-۲-۱- تنظیم پارامترهای *CPU*

به منظور فعال کردن عملکرد ریداندانت بین *AS* و *I/O* توزیع شده، نیاز است که در *CPU* تنظیمات خاصی انجام شود.

	۱- *SIMATIC H Station* را در *HW Config* باز کنید. ۲- اولین *CPU* در اسلات ۳، رک (۰) را انتخاب و پنجره پراپرتی آن را باز کنید.
	۳- سربرگ *Cyclic Interrupts* را با کرده و از آن جا از طریق یک لیست *drop-down* فضای *process image partition* را برای *OB* سیکلیک (*OB35*) که برنامه در آن اجرا می‌شود را انتخاب کنیدید.



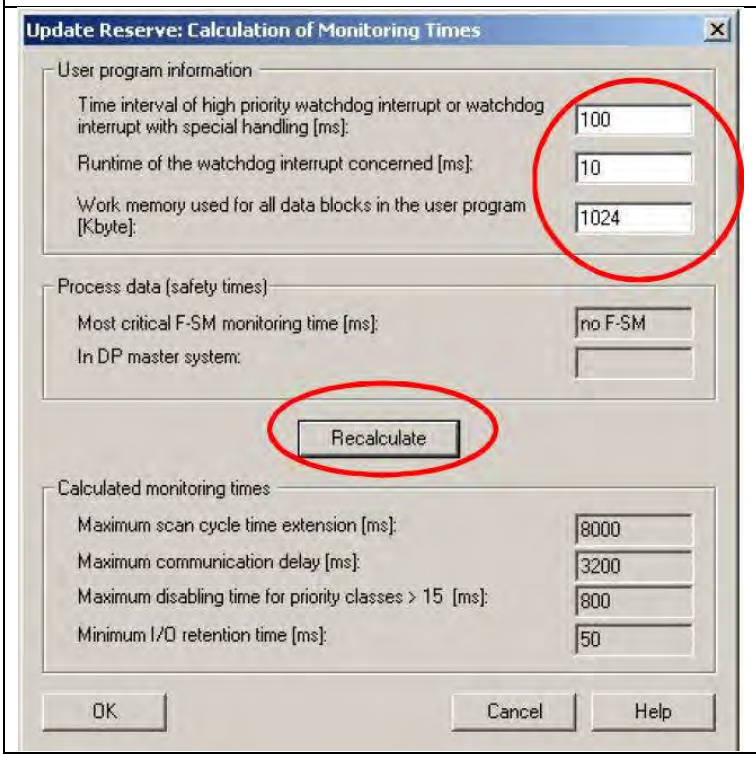
٤-سربرگ *H Parameters* را باز کنید.در این سربرگ دقت کنید که شماره *DB* که در فیلد «*Data block no*» وارد می‌شود. نبایستی در برنامه یا در پیکربندی استفاده شده باشد.

در قسمت *Passivation behavior* یکی از دو گزینه زیر را برای زمانی که خطا رخ می‌دهد، انتخاب کنید.

- *By chanel*
- *By module*

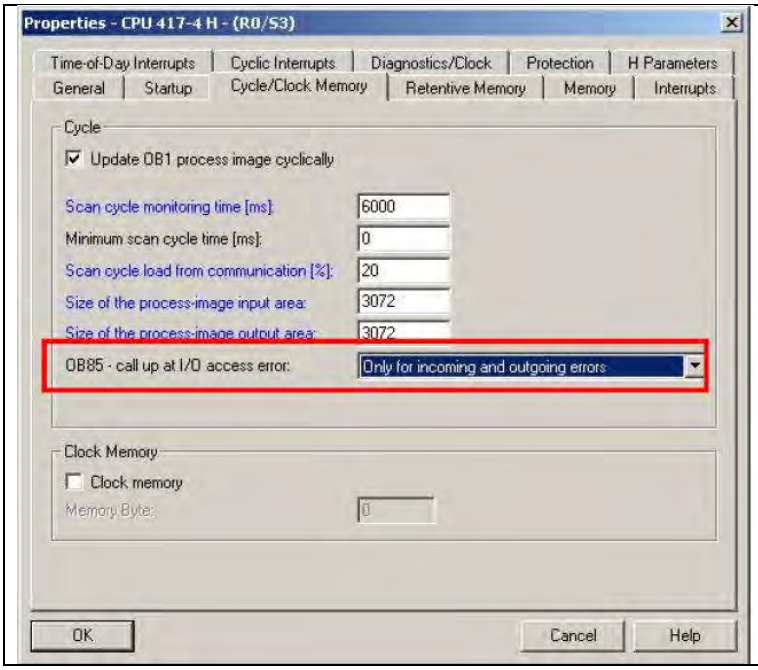
این تنظیمات به تمامی ماژول‌هایی که به صورت ریداندات به *CPU* متصل شده‌اند اعمال می‌شود.

برای محاسبه زمان مانیتورینگ دگمه *Calculate* را از قسمت *Update the Reserve* کلیک کنید.



۵- پس از کلیک روی *Calculate* در پنجره باز شده، مقادیر داده‌های مربوط به برنامه را در فیلدهای مربوطه وارد کرده و دگمه *Recalculate* را کلیک کنید.

پس از بستن این پنجره، ازطلاعات محاسبه شده جدید در فیلدهای مربوطه ظاهر می‌شود.



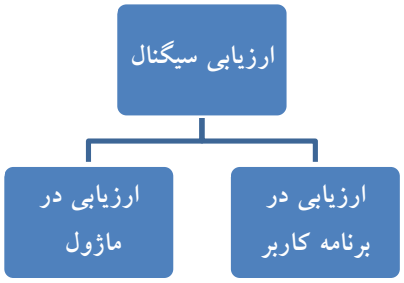
۶- سربرگ *Cycle/Clock Memory* باز کرده و در قسمت *OB 85 call for I/O error* گزینه *Only for incoming and outgoing errors* را انتخاب کنید.

۷-۲-۲- پیکربندی *ET200M* (*Configuring the first ET 200M*)

فصل – ۸ سیم‌بندی و معماری ارزیابی ایمنی برای ماژول‌های آنالوگ ورودی

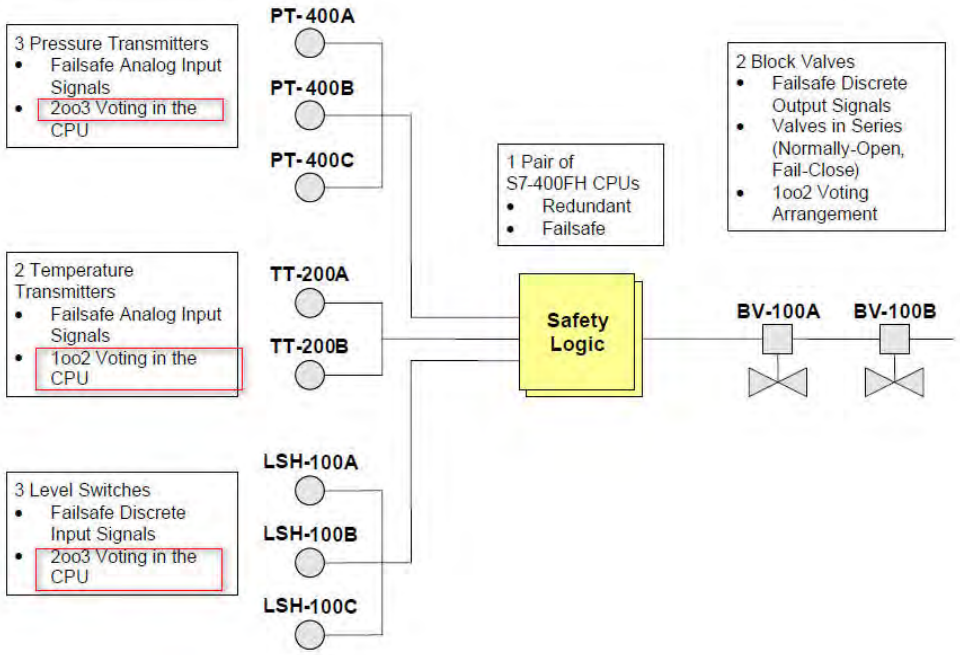
۸–۱– مثال

فرض کنید چندین سیگنال آنالوگ بایستی برای هدف *failsafe* در یک پلنت مانیتور شود. بسته به اهمیت و خطر وقوع نقص یا شکست در سیستم، چندین گزینه ، برای سیم‌بندی و ارزیابی ایمنی سیگنال ها وجود دارد. ارزیابی ممکن است، برای مثال، در ماژول ورودی آنالوگ و یا در برنامه کاربر انجام شود.



شکل ۸-۱ نشان می دهد یک مثال از سیم‌بندی ارزیابی ایمنی از سیگنال‌های ورودی آنالوگ را به تصویر کشیده است، که درآن شیرهای سولونوئید (*BV-100A* و *BV-100B*) بسته به مقادیر سگنال‌های زیر باید به صورت ایمن بسته شوند.

- *Pressure*
- *filling level*
- *temperature*



شکل ۸-۱– یک مثال از سیم‌بندی ارزیابی ایمنی از سیگنال‌های ورودی آنالوگ

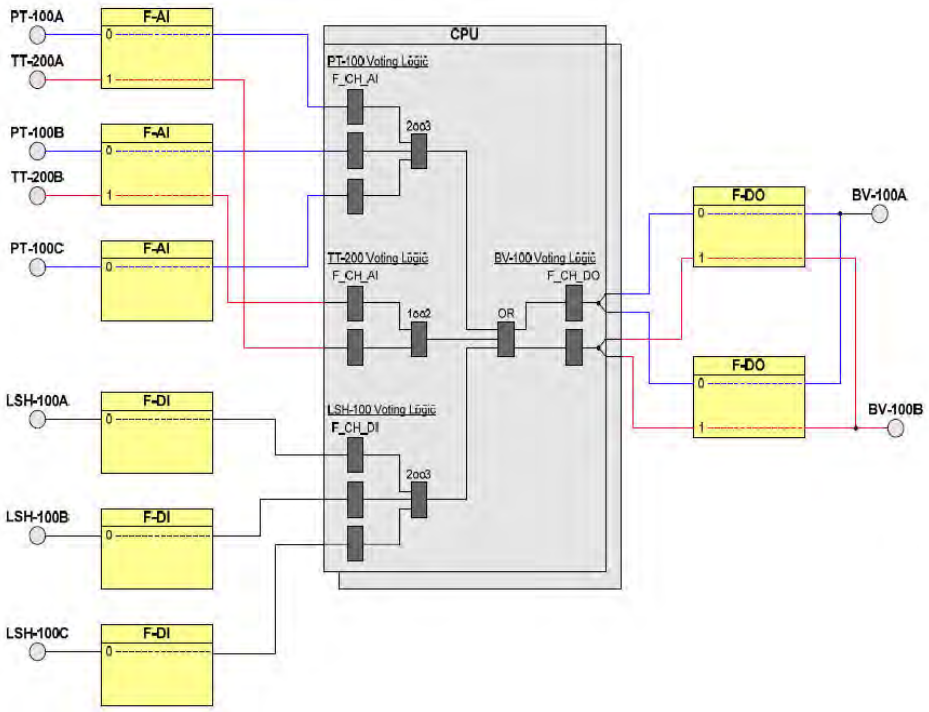
مطابق شکل بالا عمل *voting* در ماژول *CPU* صورت می‌گیرد.

نکته: در تمام مثال‌های این بخش، ماژول ورودی آنالوگ *F* از نوع هارت زیر درنظر گرفته شده است.

- *F-AI 6 x 0/4 ... 20 mA HART- 6ES7 336-4GE00-0AB0*

۸-۱-۱- راه‌حل

شکل زیر یک تحقق احتمالی از این اجرای پلنت را نشان می دهد که در آن کانکشن و معماری‌های آزمون مختلف از سیگنال‌های آنالوگ استفاده شده است.



۸-۱-۲- توصیف معماری‌های ارزیابی

معماری‌های ارزیابی که برای این مثال توصیه می شود عبارت است از:

۱. معماری تک سنسور (***1oo1***)

کاربردهای نوعی این مورد در جایی است که در آن یک سنسور تکی، سطح یکپارچه ایمنی (*SIL*) مورد نیاز را دارا بوده و در آن در دسترس بودن بالا (*increased availability*) لازم نیست.

۲. ارزیابی دو سنسور (*1oo2*) در *F-AI*

کاربردهای نوعی این مورد در جایی است که در آن یک سنسور سطح مورد نیاز ایمنی را دارا نبوده و در دسترس بودن بالا (*increased availability*) لازم نیست.

۳. ارزیابی (*1oo2*) دو سنسور در برنامه کاربر

کاربردهای نوعی این مورد در جایی است که در آن در آن یک سنسور سطح مورد نیاز ایمنی را دارا نبوده و مشاهده داده از هر دو سنسور (*visibility of the data of both sensors*) مورد نیاز است.

این معماری می‌تواند برای افزایش در دسترس بودن، به صورت *2oo2* نیز پیکربندی شود

٤. ارزیابی سه سنسور (*2oo3*) در برنامه کاربر

کاربردهای نوعی این مورد در جایی است که در آن برای رسیدن به سطح ایمنی مورد نیاز، چند سنسور لازم بوده و افزایش در دسترس بودن مدنظر است.

۸-۲- مشخصات یک ماژول ورودی آنالوگ *F*

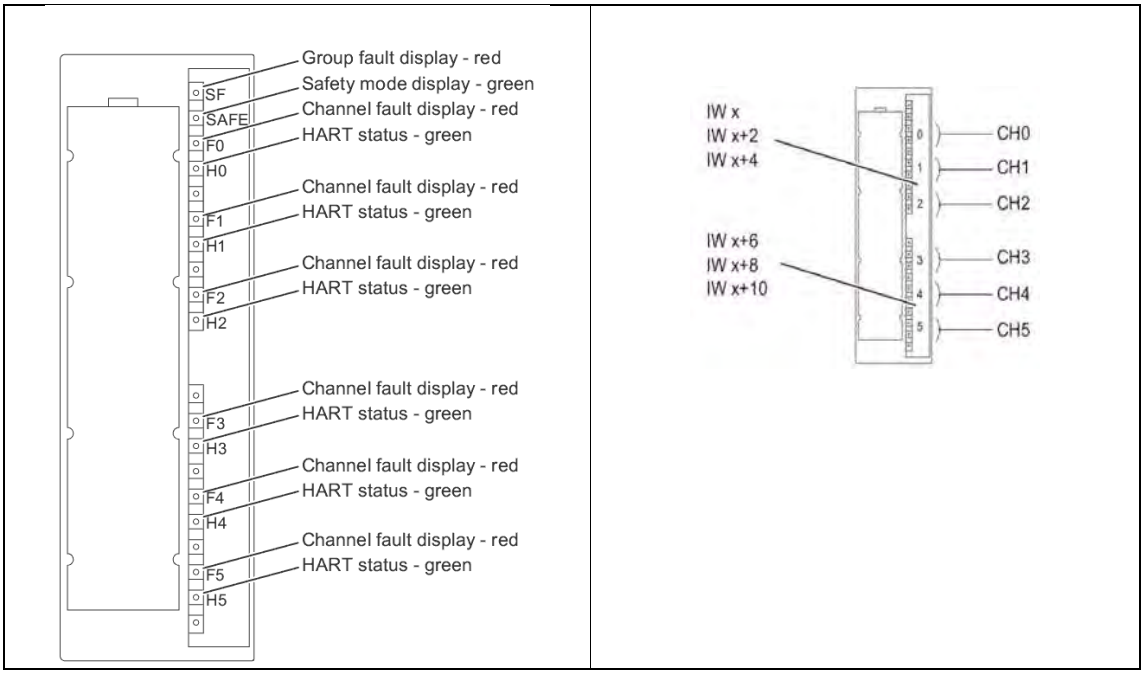
مشخصات یک ماژول ورودی آنالوگ *F* به شرح زیر می‌باشد:

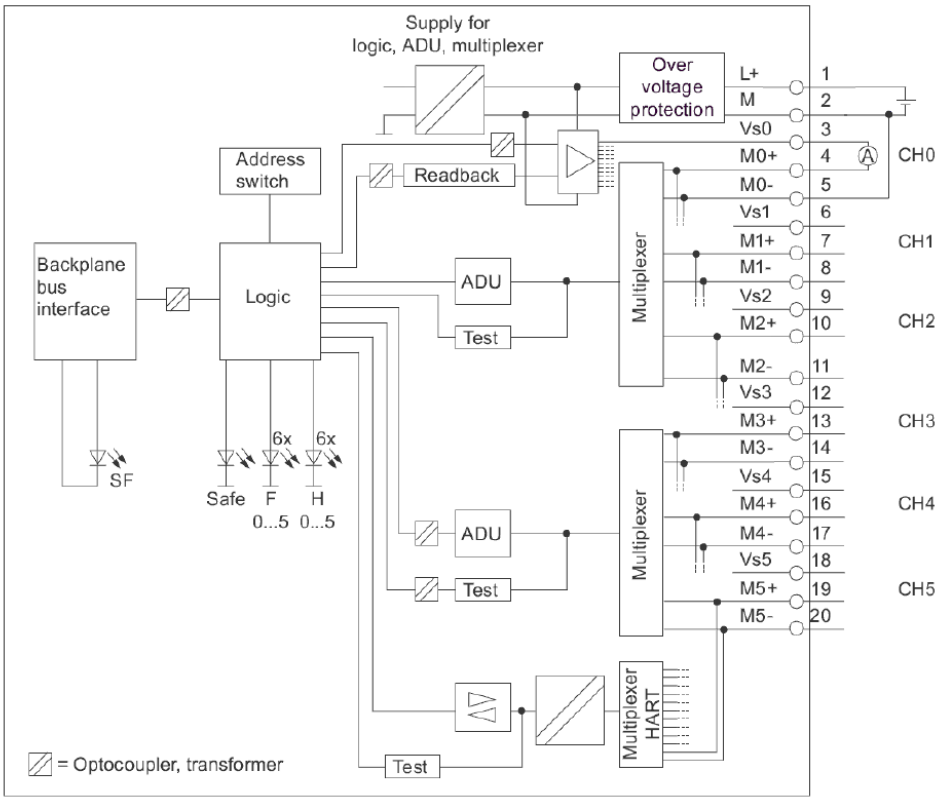
- *6 analog inputs with electrical isolation between channels and backplane bus*
- *Input ranges:*
  - *0 to 20 mA*
  - *4 to 20 mA*
- *Short-circuit resistant power supply of 2- to 4-wire measuring transmitter via*
- *the module*
- *External encoder supply possible*
- *Collective error display (SF)*
- *Display safety mode (SAFE)*
- *Display for channel-specific error (Ex)*
- *Display for HART status (Hx) (If you have switched on measuring type HART for a channel, and the HART communication is running, the green HART status display lights up.)*
- *Parameterizable diagnostics*
- *Parameterizable diagnostic alarm only in safety-mode*
- *HART communication*
- *Firmware update via HW Config*
- *Identification data*

هر یک از شش کانال آنالوگ ورودی را می‌توان برای اندازه‌گیری جریان استفاده کرد:

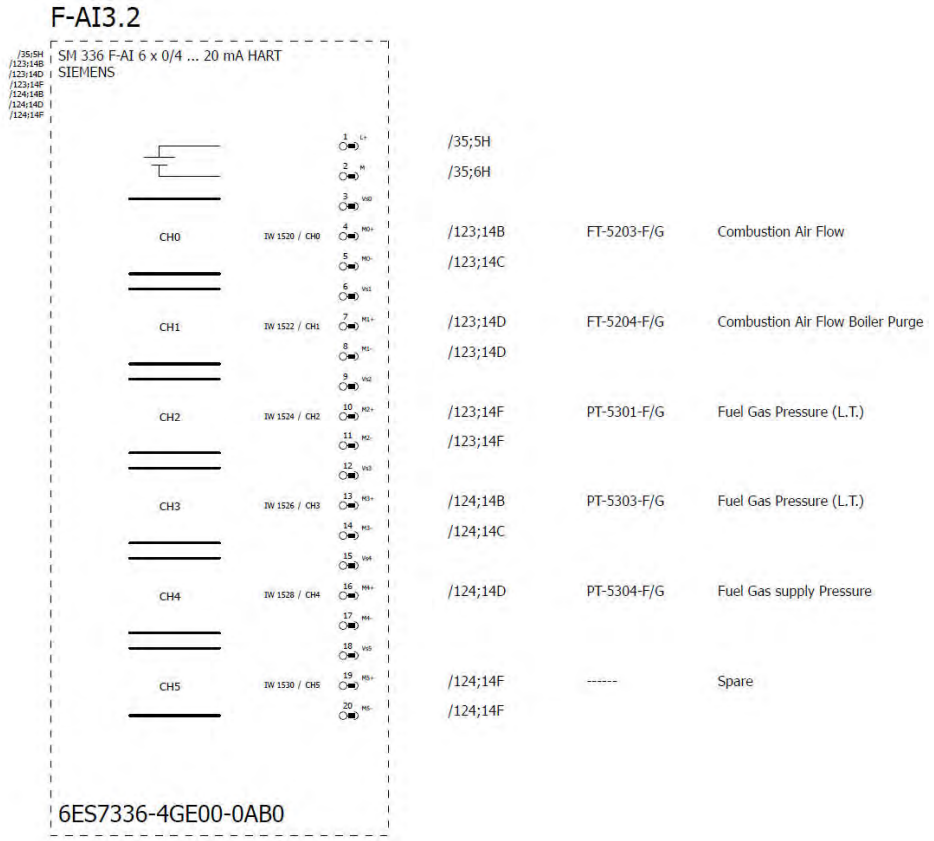
- *0 to 20 mA (without using HART)*
- *4 to 20 mA (with/without using HART)*

- این کارت بیست پین می‌باشد.
- توصیه می‌شود که همیشه از *short-circuit resistant internal encoder supply* استفاده شود.
-





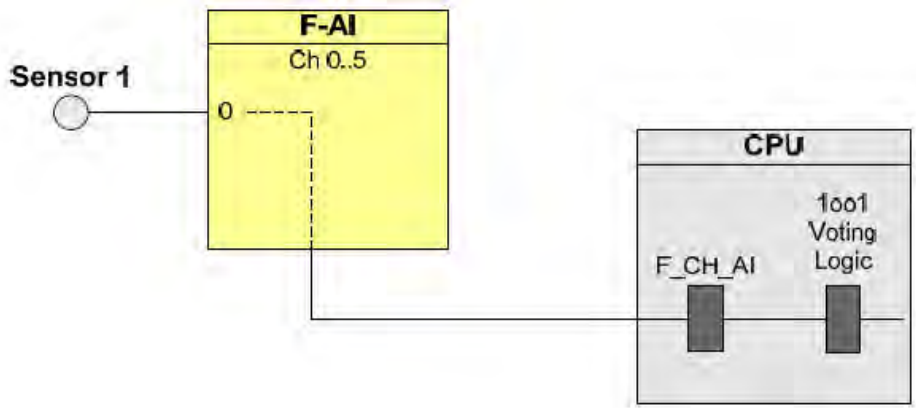
دیاگرام مدار اتصالات کارت آنالوگ *SM 336; F-AI 6 x 0/4...mA HART*



شکل ۸-۲- نمونه اتصال سیگنال‌های آنالوگ هارت به کارت

۸-۳- سیم‌بندی یک سنسور آنالوگ (*1oo1*)

طرح ارزیابی یک سنسور (یا*1oo1*)به کاربردهایی مربوط می‌شود که در آن به افزونگی سنسور نیاز نمی‌باشد. ارزیابی *1oo1* بدان معنی است که تنها یک سنسور وجود دارد. اگر در سنسور شرایط تریپ رخ دهد، لاجیک ایمنی تریپ خواهد کرد.شکل زیر معماری پایه *1oo1* را نشان می‌دهد که در آن یک سنسور به یک کانال از ماژول *F-AI* متصل شده است.



شکل ۸-۳- سیم‌بندی یک سنسور به یک کانال *F-AI* (*1oo1*)

جدول زیر شرایط تریگر شدن تابع واکنش خطا (*error reaction function*) را نشان می‌دهد.

Component has failed ?		Error reaction function has been triggered?
Sensor 1	F-AI	
No	No	No
X	Yes	Yes
Yes	X	Yes

در این معماری اگر سنسور یا *F-AI* معیوب شود، تابع واکنش خطا، عملکرد ایمنی (از طریق سیستم *failsafe*) فراهم می کند.

۸-۳-۱- محاسبه *PFD* (*Probability of Failure on Demand*)

مقدار *PFD* احتمال شکست یک تابع *failsafe* را توصیف می‌کند.

$$PFD(1oo1) = PFD_{Sensor} + PFD_{F\text{-}AI} + PFD_{CPU}$$

برای یک سنسور با معماری *1oo1* مقدار PFD_{Sensor} با استفاده از فرمول زیر محاسبه می‌شود

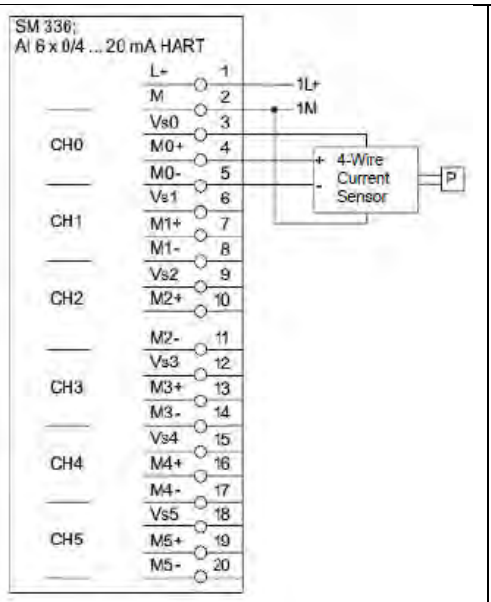
$$PFD_{1oo1} \approx \lambda_{DU} \cdot \frac{T_I}{2}$$

۸-۳-۲- سیم‌بندی (*Wiring*)

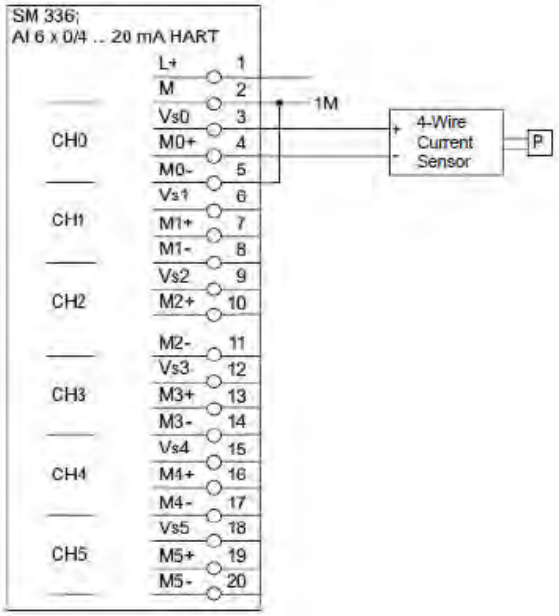
در این طرح ارزیابی *1oo1*، سنسور را می‌توان به دو صورت زیر تغذیه کرد:

- از طریق منبع داخلی ماژول *F-AI* (*internally*)
- از طریق یک منبع بیرونی (*external power supply*)

شکل زیر یک مثال از سیم‌بندی یک سنسور دو سیمه و چهار سیمه را به تصویر کشیده است. که در هر دو شکل ترنسمیتر اندازه‌گیری به کانال ۰ (ترمینال ۳,٤,۵) متصل شده و با ماژول *F-AI* تغذیه شده است. لازم به ذکر است که با توجه به هارت بودن کارت سیگنال از نوع جریان می‌باشد. همچنین در شکل های زیر، حرف *P* معرف یک سنسور می‌باشد.

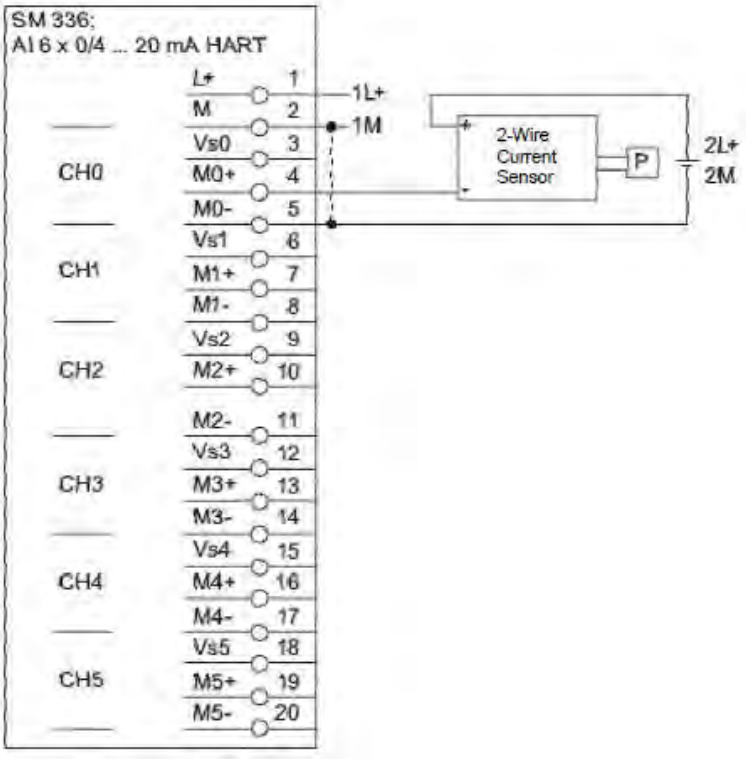


اتصال چهار سیمه

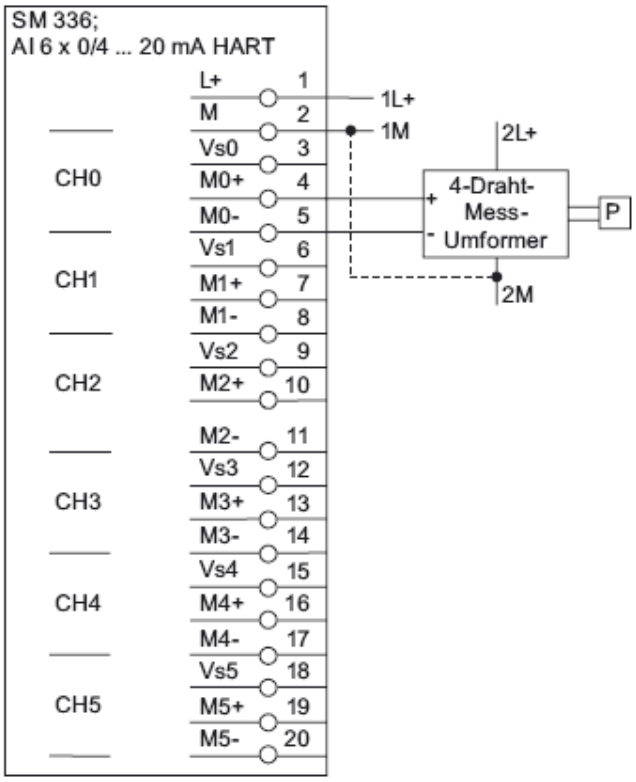


اتصال دو سیمه

شکل زیر یک مثال از سیم‌بندی یک سنسور دو سیمه را با تغذیه بیرونی به تصویر کشیده است. که در آن ترنسمیتر اندازه‌گیری به کانال ۰ (ترمینال ۲,٤,٥) متصل شده است.



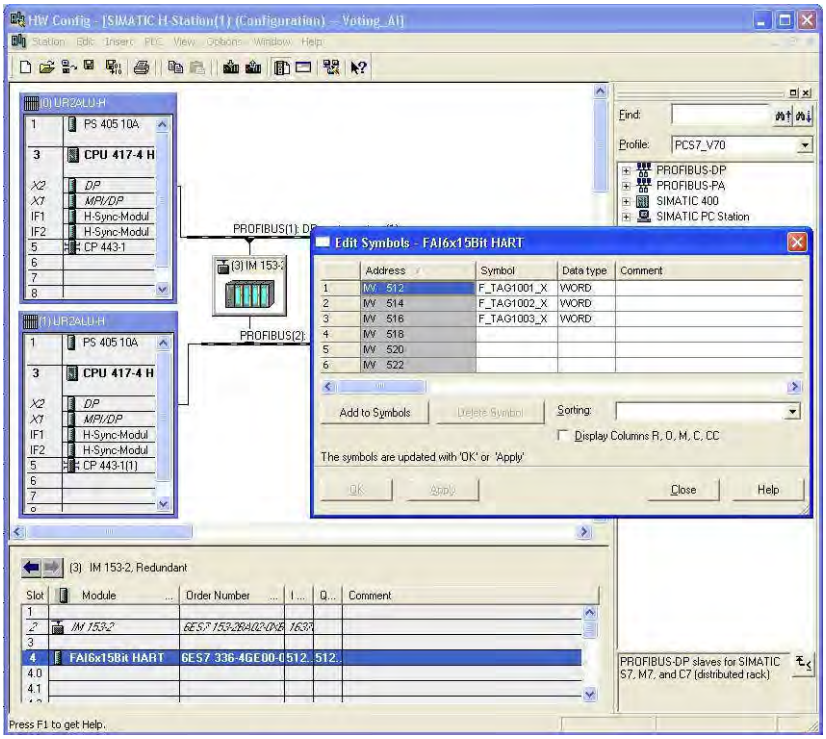
توجه شود که افت ولتاژ از حد مجاز در ترنسمیتر اندازه گیری توسط *F-AI* قابل تشخیص نمی‌باشد. بنابراین بایستی از یک ترنسمیتر اندازه‌گیری با قابلیت تشخیص افت ولتاژ استفاده شود.



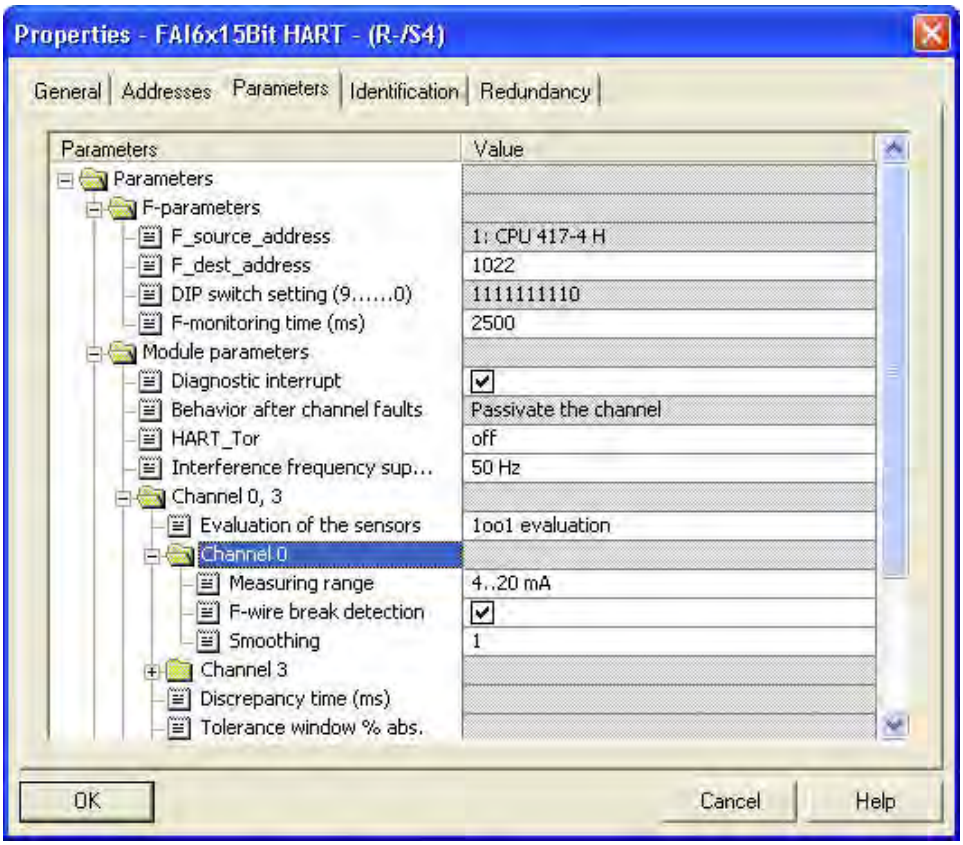
شکل ۸-٤- اتصال سنسور چهار سیمهبا تغذیه بیرونی به یک کارت *F-AI*

۸-۳-۳- پیکربندی ماژول در *Step7*

پیکربندی سخت افزاری ماژول *F-AI* در *STEP 7* (*HW Config*) همانند هر ماژول *F* نوع *ET 200M* می‌باشد.یک مثالاز پیکربندی سخت افزار *F-AI* در شکل زیر نشان داده شده است. در این مثال سیگنال سنسور به کانال ۰ از *F-AI* متصل شده است.



در پنجره *object properties* ماژول *F-AI* پارامترهای مورد نیاز برای عملکرد *F-AI* تنظیم می‌شود. پارامترها در جدول ۲-۲ خلاصه شده است.



پنجره پراپرتی یک ماژول *F-AI*

۸-۳-۴- پیکربندی معماری *1oo1* در *CFC*

دو روش ممکن است برای پیاده سازی منطق *CFC* وجود دارد:

- بدون ارزیابی خطای کانال (*1oo1*)
- با ارزیابی خطای کانال (*1oo1D*)

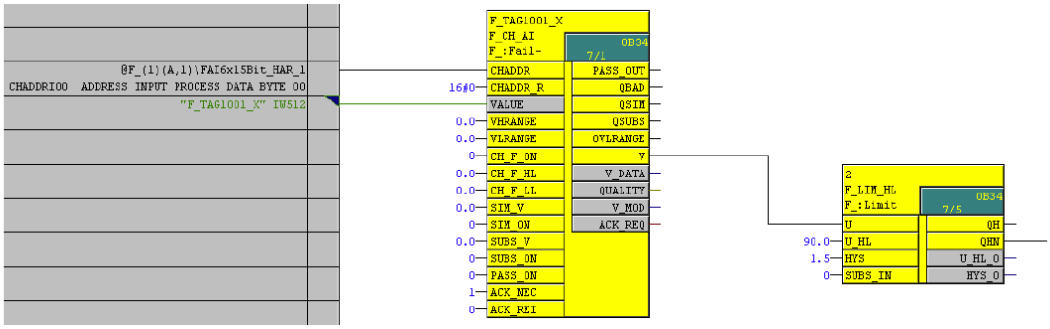
پیکربندی بدون ارزیابی خطای کانال (*1oo1*)

شکل زیر یک پیکربندی نمونه ایجاد شده در ویرایشگر *CFC* را برای قرائت یک سیگنال ورودی که در آن خطای کانال درنظر گرفته نشده است، نشان می دهد. توجه داشته باشید که در این مثال که یک مقدار حد *MAX* فرض شده است. لذا مقدار قرائت شده از کانال توسط بلاک *F_:Limit* برای تجاوز از حد بالا (۹۰ در این مثال) ارزیابی شده و خروجی لاجیک ارزیابی برای رسیدن به وضعیت ایمن، به وضعیت خاموش سویچ می‌کند. یعنی تا زمانی که مقدار کانال از حد مقدار ۹۰ تجاوز نکرده است، خروجی بلاک ۱ و به محض تجاوز از حد ۹۰ با تولورانس ۱,۵ خروجی لاجیک ۰ می‌شود. یعنی فرمان تریپ صادر می‌شود. (*normal state = 0,=1failsafe state*)

برای پیاده‌سازی لاجیک *CFC* بلاک های زیر استفاده می‌شودر:

۱- از یک بلاک *F_CH_AI* برای قرائت سیگنال کانال

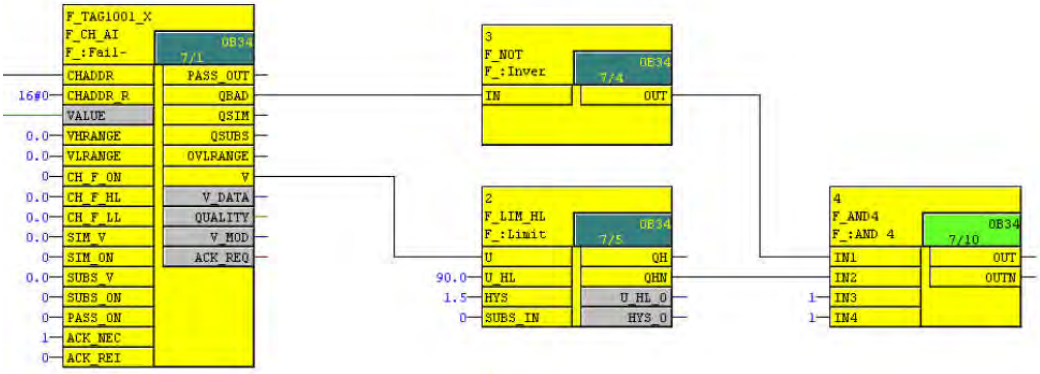
۲- از یک بلاک *limit value* مانند *F_LIM_HL* یا *F_LIM_LL* برای ارزیابی مقدار کانال



پیکربندی با ارزیابی خطای کانال (*1oo1D*)

شکل زیر یک پیکربندی نمونه ایجاد شده در ویرایشگر *CFC* را برای قرائت یک سیگنال ورودی که در آن خطای کانال در نظر گرفته شده است، نشان می دهد. توجه داشته باشید که در این مثال یک مقدار حد *MAX* فرض شده است. لذا مقدار قرائت شده از کانال توسط بلاک *F_:Limit* برای تجاوز از حد بالا (۹۰ در

این مثال) ارزیابی شده و خروجی لاجیک ارزیابی برای رسیدن به وضعیت ایمن، به وضعیت خاموش سویچ می‌کند.



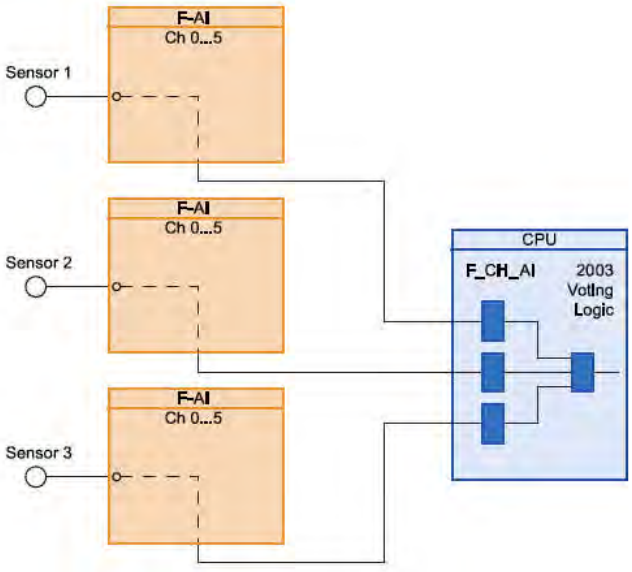
در این پیکربندی:

- برای رنج نرمال یعنی تا زمانی که مقدار کانال از حد مقدار ۹۰ تجاوز نکرده است، خروجی بلاک ارزیابی (بلاک‌های *AND,NOT,F_LIM_HL*) ۱ می‌باشد
- به محض تجاوز از حد ۹۰ با تولورانس ۱,۵ خروجی لاجیک ۰ می‌شود. یعنی فرمان تریپ صادر می‌شود. *(failsafe state = 0, normal state =1)*
- اگر در قرائت مقدار *PV*، خطا در کانال گزارش شود، خروجی لاجیک۰ می‌شود. یعنی فرمان تریپ صادر می‌شود.

۸-۴- یک

۸-۵- سیستم انتخاب 2oo3 با ماژول آنالوگ ورودی F (2oo3 Voting with F-AI)

انتخاب دو از سه (2-out-of-3) از سه سنسور و به عنوان مثال از سه ماژول F-AI استفاده می‌کند. در این مثال هر سنسور به کانال صفر یک ماژول F-AI سیم‌بندی می‌شود. سپس تک تک سیگنال‌های حاصل شده از کانال‌ها در برنامه لاجیک ارزیابی می‌شود. در این نوع معماری با یک پیکربندی دو کاناله سطح SIL3 نیز قابل دست‌یابی است.



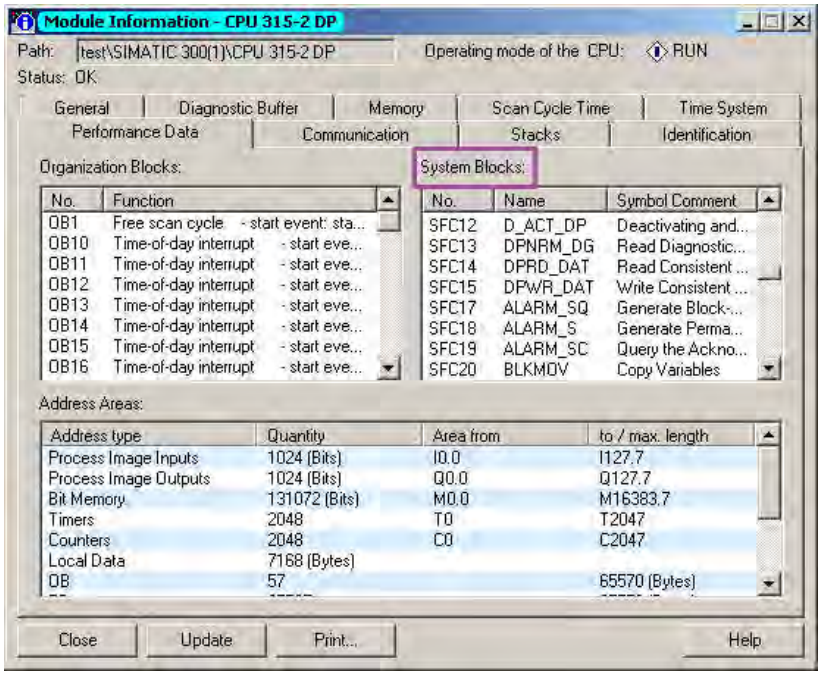
شکل ۸-۵-

۸-۵-۱- فانکشن‌بلاک‌های سیستمی پشتیبانی شده در ماژول CPU (Supported SFB & SFC)

برای تشخیص این که یک ماژول CPU چه فانکشن‌بلاک‌های سیستمی را پشتیبانی می‌کند. به دو روش زیر می‌توان عمل کرد.

۱- پروژه را در Simatic Manager باز کرده و مد آنلاین را فعال می‌کنیم. لذا کلیه SFC ها و SFBهای که CPU پشتیبانی می‌کند را می‌توان مشاهده کرد.

۲- مد آنلاین را فعال کرده و از پنجره module information مطابق شکل زیر از سربرگ System locks، بلاک‌های پشتیبانی شده قابل مشاهده است.



۸-۶- اتصال *S7 REDCONNECT*

اتصال *S7 REDCONNECT* یک لایسنس نرم افزار برای ارتباط بین سیستم *S7-400H AS* و سیستم مانیتورینگ *WinCC/PCS7 OS* (*WinCC Redundancy/PCS7 OS Redundancy*) از طریق دو عدد *CP1613 A2* یا *CP1623 A2* می‌باشد. این لایسنس در سیستم *H* در صورت وقوع نقص در یکی از کنترل‌کننده‌ها و یا با از دست دادن ارتباط بین *AS* و *OS*، به طور خودکار اتصال از یک *CP* را به *CP* دیگر سوئیچ می‌کند. برای استفاده از این قابلیت بایستی در برنامه *NETPRO* بین *AS* و *OS* یک اتصال از نوع *Fault tolarant* تعریف کنید.

اتصال ایستگاه *S7 400H* به یک *OS* را از طریق یک *CP-1613* به صورت زیر انجام دهید:

۱- در پنجره *NETPRO* بر روی *CPU-41X H* راست کلیک کرده و گزینه *Insert New Connection* را انتخاب کنید.

۲- در پنجره جدید باز شده گزینه *S7-connection fault-tolerant* را از قسمت *Connection type* انتخاب کنید.

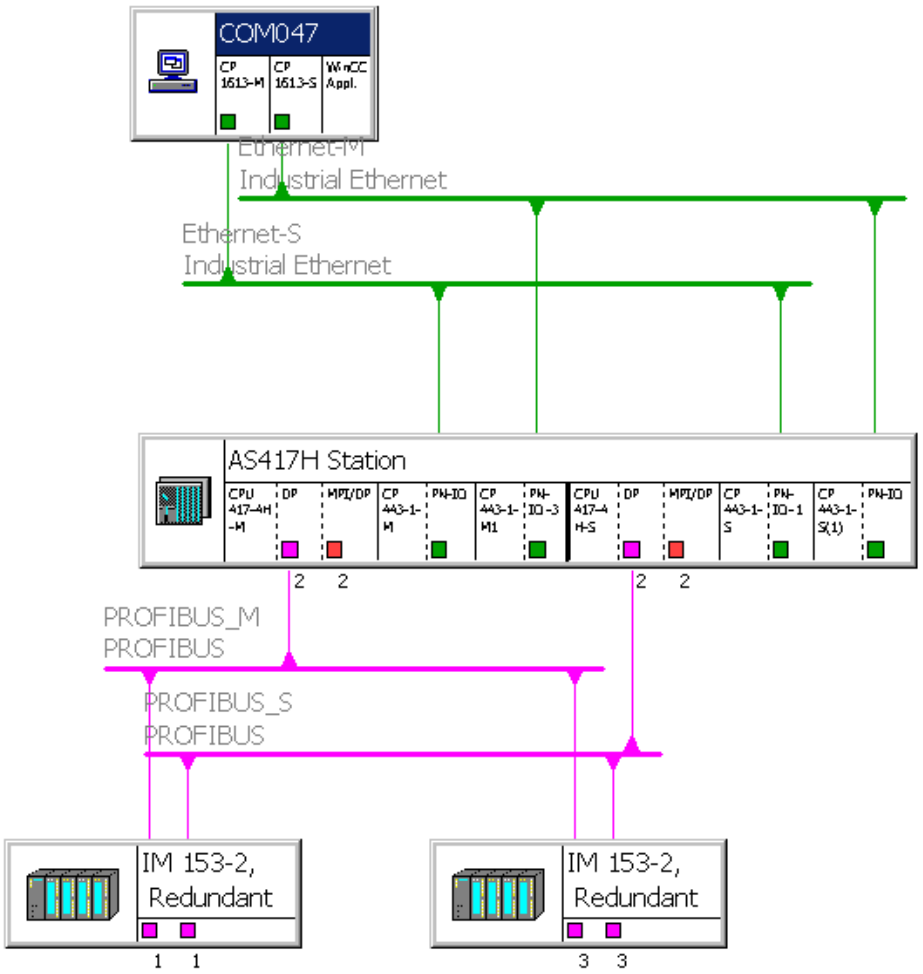
۸-۶-۱- پیکربندی اتصالات *RedConnect*

نکته: برقراری اتصالات *Fault-tolerant S7 connections* تنها با استفاده از کارت‌های *CP1613* و *CP1623* امکان‌پذیر می‌باشد. امکان پیاده‌سازی *Fault-tolerant S7 connections* با کارت‌های شبکه استاندارد وجود ندارد.

همچنین نیاز است که یکی از دو ابزار پیکربندی زیر در ایستگاه مهندسی نصب شود.

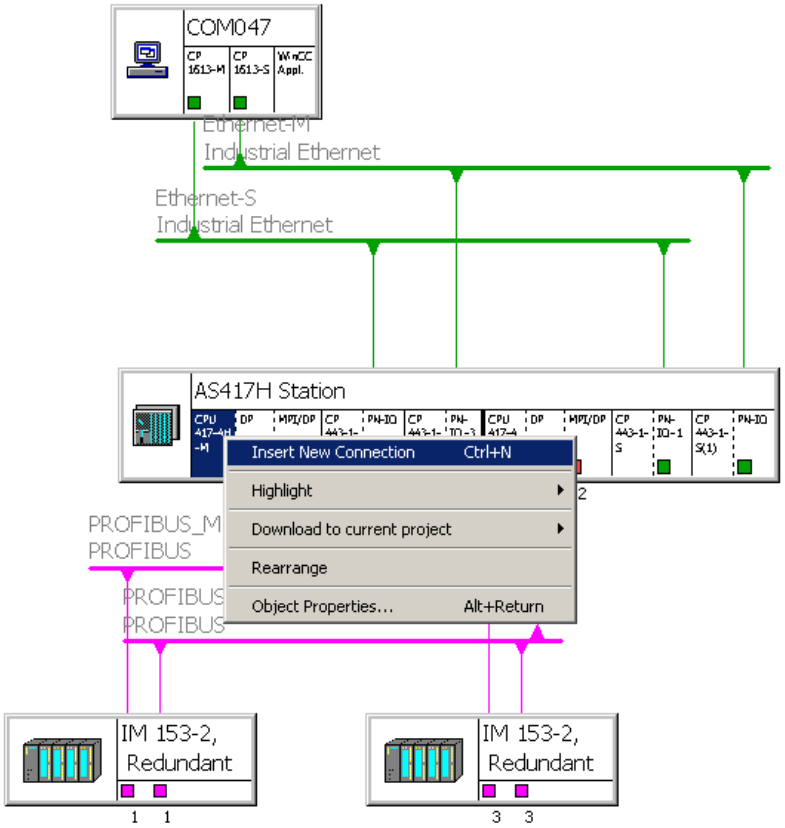
You need the license for the SIMATIC NET PC software S7-REDCONNECT on the PC station that communicates with the S7400-H system via fault-tolerant S7 connections.

در شکل زیر یک اتصال *S7 Fault-tolerant* بین سیستم *H* و کامپیوتر *HMI* برقرار شده است. که در آن خود شبکه اترنت به صورت افزونه می‌باشد. لذا در بخش *master* و *Slave* ایستگاه *AS* از دو عدد *CP* اترنت استفاده شده است.

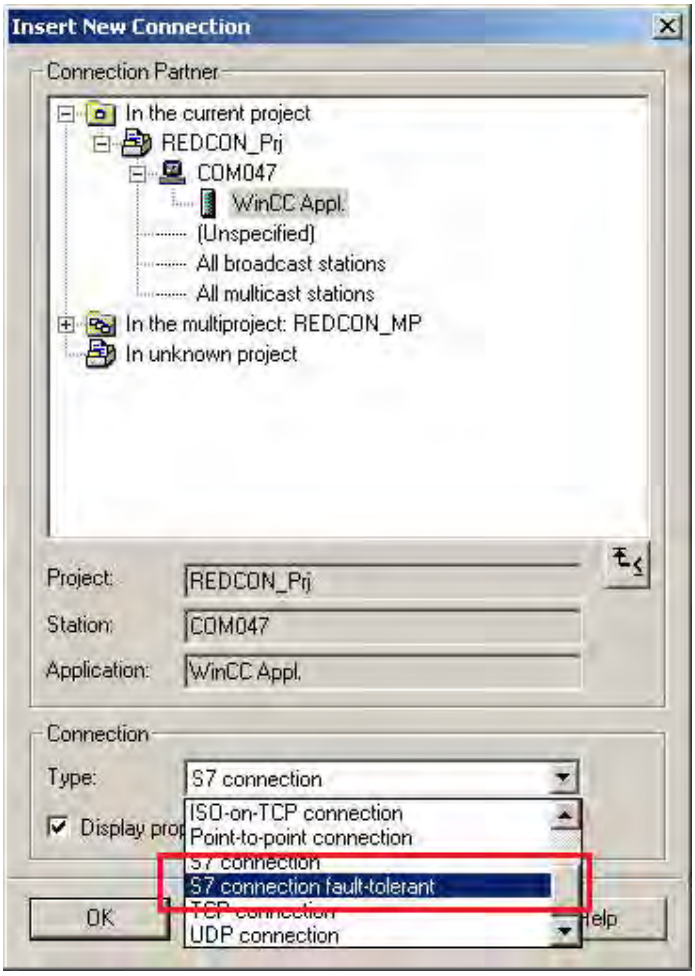


مراحل ایجاد *named Connection* به شرح زیر می‌باشد.

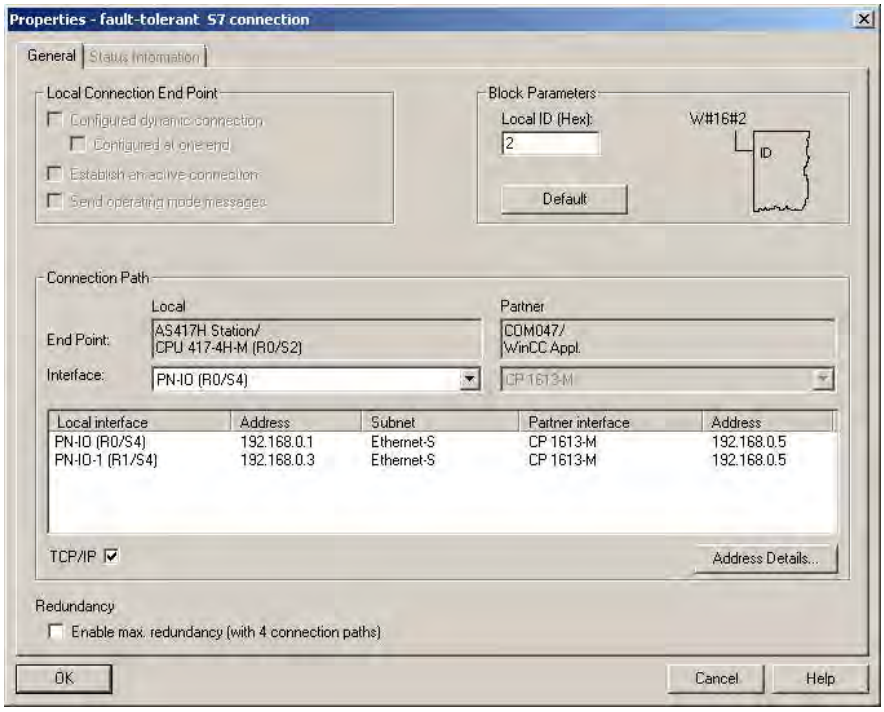
۱- در پنجره *NetPro* مطابق شکل زیر بر روی *CPU 417H* مستر راست کلیک کرده و گزینه *Insert New Connection* را انتخاب کنید.



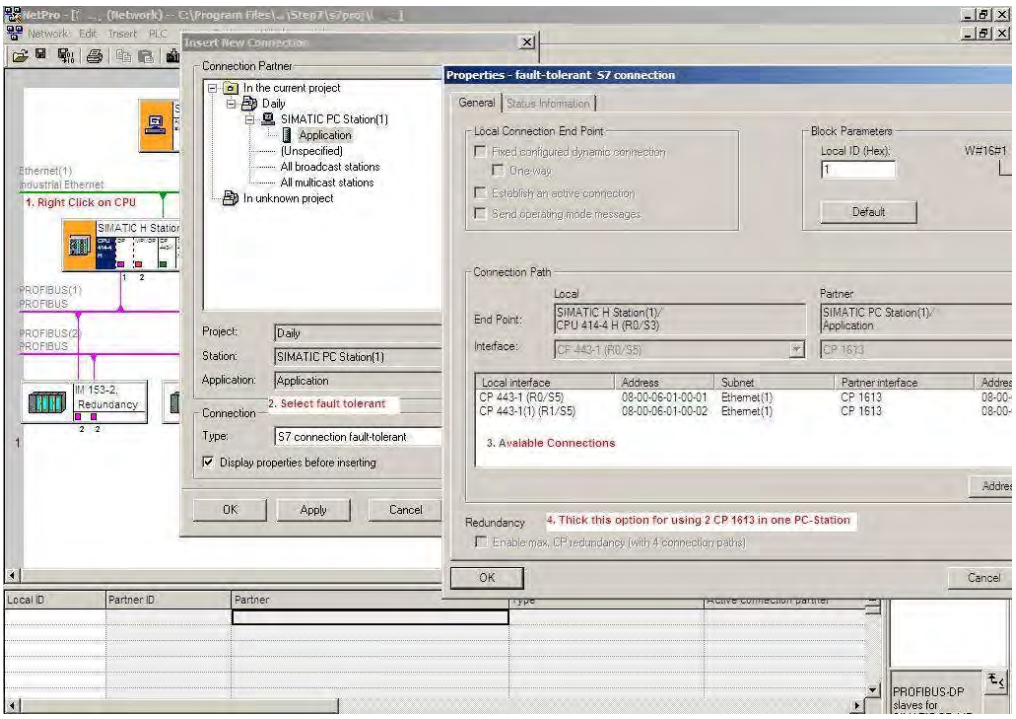
۲- در پایین پنجره ظاهر شده مطابق شکل زیر گزینه *S7 Connection fault tolerant* را انتخاب کرده و *Ok* می‌کنیم.



۳- با کلیک روی *OK* در پنجره قبلی شکل زیر ظاهر می‌شود.



٤- در صورتی که تیک *Enable max redundancy* را بزنیم. عمل ریداندنسی از طریق ٤ مسیر برقرار می‌شود.



8-7- Using SFC 14 and SFC15 (for Consistent data Transfer)

Consistent data" basically means that you are transfering large amounts of data as a single unit, in a single communication cycle.

Consistend Data with the size of 1, 2 or 4 byte should be read / write via a direct access without the SFC's.

۸-۸- فضای حافظه CPU (S7 System Memory Areas)

The system memory of the S7 CPUs is divided into address areas (see table below). Using instructions in your program, you address the data directly in the corresponding address area.

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	

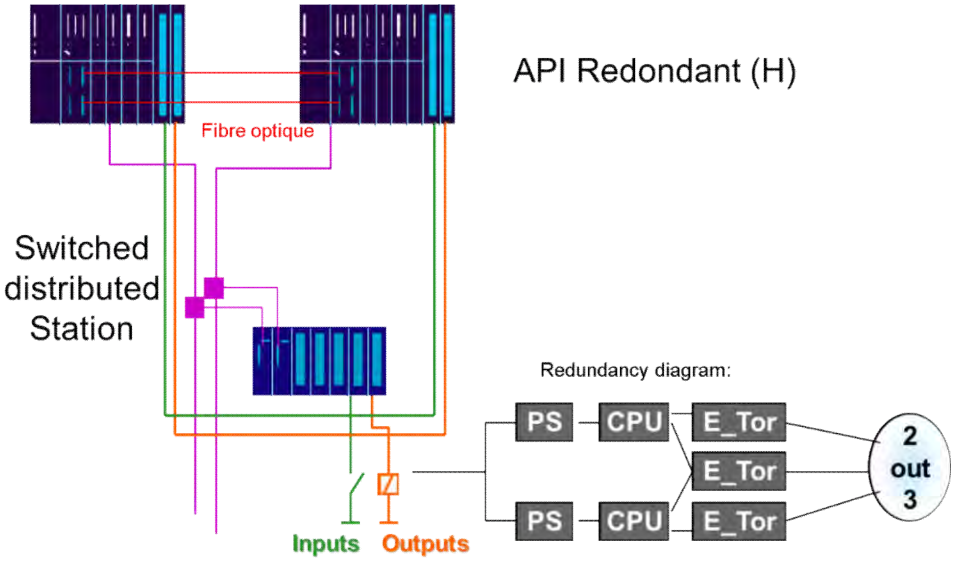
	Data double word	*DBD*	
	Data block, opened with "OPN DI":	*DI*	
	Data bit	*DIX*	
	Data byte	*DIB*	
	Data word	*DIW*	
	Data double word	*DID*	
Local data	*Local data bit*	*L*	*This area contains the temporary data of a block while the block is being executed. The L stack also provides memory for transferring 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*	

۸–۸–۱– عمل غیر فعال سازی ماژول‌های ورودی/خروجی(*Passivation*)

غیرفعال کردن بدان معنی است که در صورت وقوع یک خطا، یک یا تعدادی از کانال‌ها در یک ماژول سیگنال *F*، به یک وضعیت امن سویچ می‌کند. در صورت خطا در کانال (برای مثال یک سنسور معیوب)، تنها کانال‌های آسیب دیده پسیو می‌شوند.

در صورت وقوع خطا در ماژول (به عنوان مثال یک خطای ارتباطی)، تمام کانال‌ها بر روی ماژول *I/O* خرابی امن پسیو می‌شوند.

اگر یک ماژول سیگنال *F* خطایی تشخیص دهد، کانال‌های آسیب دیده و یا همع کانال‌های آن را به حالت امن سوئیچ می‌کند. به عبارت دیگر، کانال‌های این ماژول غیرفعال می‌شود. ماژول سیگنال *F* برای نشان دادن این که خطایی شناسایی شده است، یک پیام به درایور کانال *F* و *PCS 7 OS* می‌فرستد.



۸–۹– جدول اختصارات و نعاریف

HFT	*hardware fault tolerant*
DC	*diagnostic coverage*
SFF	*safe failure fraction*
EUC	*Equipment Under Control*

۸–۹–۱– قابلیت‌های کارت‌های *Fail Safe*

کارت‌های *Fail Safe* دارای این قابلیت هستند که با نصب بسته *S7 F system* امکان تعریف سیستم ارزیابی یا *Voting*، ۲ از ۳ برای ورودی‌های آنالوگ فراهم می‌شود. مثلا تو خط ساکشن کمپرسور گاز سه عدد *PT* نصب می‌کنند. به طوری که اگر حداقل دو عدد از سه ترانسمیتر از حد تریپ گذشتند کمپرسور تریپ می‌خورد.

قابلیت دیگر این است که برای ورودی دیجیتال هم ولتاژ، ۲۴ ولت که قرار است سر سویچ برود می‌تواند از تو خود کارت (*Vs*) ایجاد بشود و این ولتاژ مانیتور میشه که اگه تغذیه سر سویچها (یا با تنها مانند امرجنسی) مشکلی پیدا کرد آلارم ایجاد بشه.

و در آخر چون فقط برای *EMS* و سیستمهای *SAFETY* استفاده می‌شوند خروجی آنالوگ تو این دسته کارتها نیست.

برداشت من از مفهوم *Fail Safe* ارتباط اون تا اندازه ای با درجه *SIL* بود و کاملا فکر میکنم به خود کارت مربوط باشه و نوع ساخت و سخت افزار اون و رفتاری که در هنگام بروز خطا کارت از خودش نشون میده

همانطوری که گفتید در بحث *SIL* مفهومی بنام *pfd* یا *probability of failure ondemand* وجود داردکه احتمال بروز خطای منجر به حادثه را بیان میکند و همین مقدار نشان دهنده سطح *SIL* است بطوریکه هرچقدر *pfd* کمتر باشه سطح *SIL* بالاتر میره

حالا کارتها یا تجهیزاتی که احتمال بروز خطای حاد در آنها وجود دارد، *SIL* تو اونها در سطح تعریف شده باشه اون تجهیز اون سطح از *SIL* رو خواهد داشت و اصطلاحا به این تجهیزات *fail safe* گفته میشه

این تجهیزات معمولا در زمان بروز خطا عملکردی از خودشون نشان میدن تا *safety* فرایند حفظ بشه

حالا این تشخیص خطا و شرایط *fail* هم بصورت سخت افزاری و هم بصورت نرم افزاری و با استفاده از الگوریتم های خاص صورت میگیره

در مورد *redundancy* هم باید بگم که به تنهایی تضمین کننده *safety* نیست و تنها *availability* رو بالانر میبره اما در مورد تجهیزات *fail safe* که سطح *SIL* مشخصی رو دارند *redundancy* میتونه سطح *SIL* رو بالاتر ببره

برای مثال کارتهای ٢٤ کاناله *fail safr* زیمنس دارای سطح سیل ٢ است که با ریداندت کردن دو کانال روبروی هم، این سطح *SIL* به ٣ ارتقا پیدا می‌کنه و عملا کارت ١٢ کاناله میشه.

اگرمحاسبات *pfd* رو توجه کنید می بینید که *redundancy* باعث کاهش *pfd* میشه و میتونه سطح *sil* رو افزایش بده (رجوع شود به کتاب *sil* شرکت *gmi*)

If you are using Manufacturer's prior use data because a selected product does not reach the required level under FMEDA analysis, be aware that there are significant requirements on the end user. A mature product must generally be used to have the required field experience, and the design and assembly must be "frozen in time" in such a way that no upgrades, modifications or even configuration changes may be allowed that may render the "Proven In Use" data useless.

A key result of the analyses is establishing a Safe Failure Fraction (SFF) for a product. Figure F below shows the relationship of SFF values, SIL ratings and the effects of redundancy.

Figure F

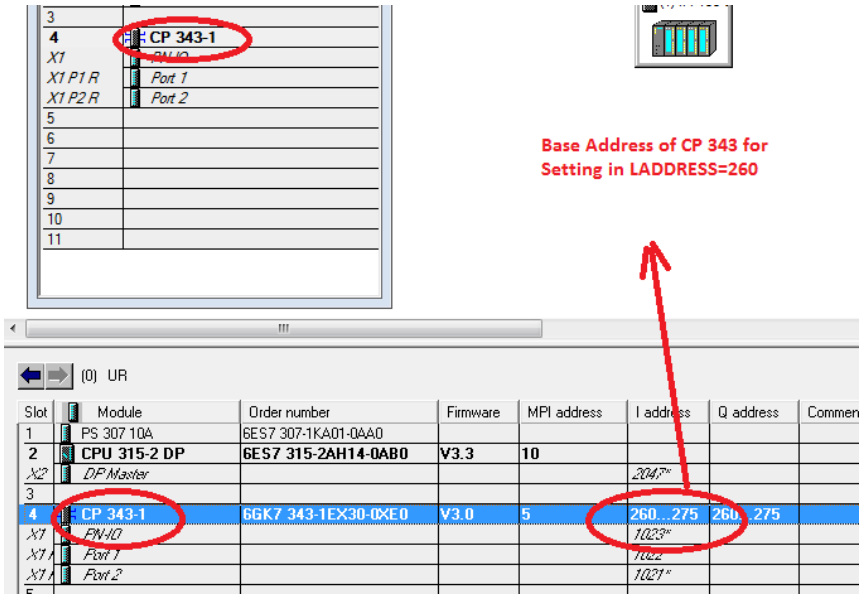
Safe Failure Fraction (SFF) *(for Type B, microprocessor-based devices)*	No Redundancy	Single Redundancy	Double Redundancy
<60%	Not Allowed	SIL 1	SIL 2
60%<90% *(typical competitor)*	SIL 1	SIL 2	SIL 3
90%<99% *(Eclipse, Jupiter, E3)*	SIL 2	SIL 3	SIL 4
>99%	SIL 3	SIL 4	SIL 4

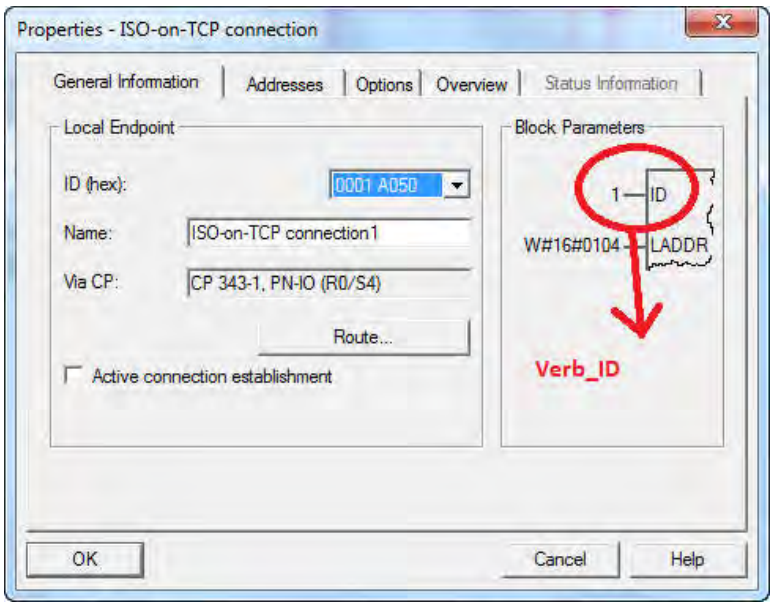
While two SIL 1 devices can be used together to achieve SIL 2 and two SIL 2 devices to achieve SIL 3 (as suggested by the chart above), it is not automatic. Using redundancy to attain a higher SIL rating has additional requirements beyond hardware. It has an additional requirement of systematic safety which includes software integrity.

It is important to note that the most conservative approach to redundancy is to use dissimilar technologies. This reduces failures due to application issues.

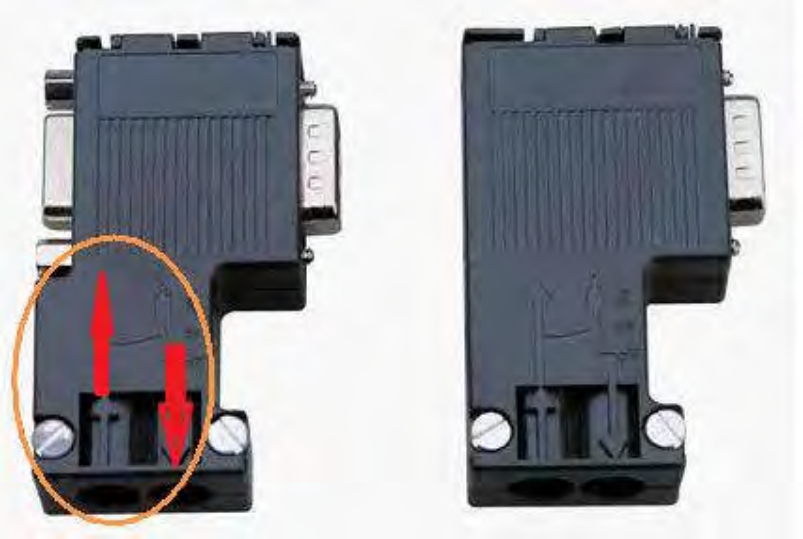
فصل – ۹ – نکات مربوط به پیکربندی در *Step 7*

۹–۱– آدرس پایه تبادل داده برای *CP343*





نکته : نحوه اتصال کانکتور پروفی‌باس در سمت *CPU* از مسیر ورود کابل و در *ET*ها از مسیر خروج کابل می‌باشد.



۹-۲-

1- Fail-safe modules

- *The major difference between fail-safe modules and standard ET 200S modules is that failsafe modules have a two-channel internal design.*
- *Both integrated processors monitor each other, automatically test the I/O circuits, and set the F-module to safe state in the event of a fault. The F-CPU communicates with the fail-safe module using the PROFIsafe safetyrelated bus profile.*

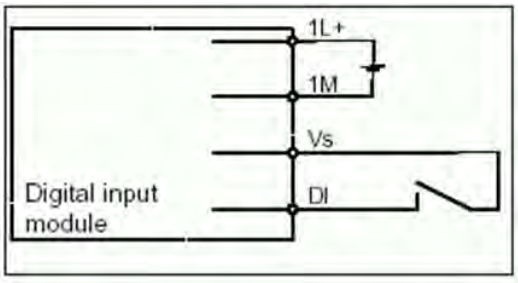
1-0-1- 1oo1 and 1oo2 evaluation with the fail safe signal module SM 326F

پرسش:

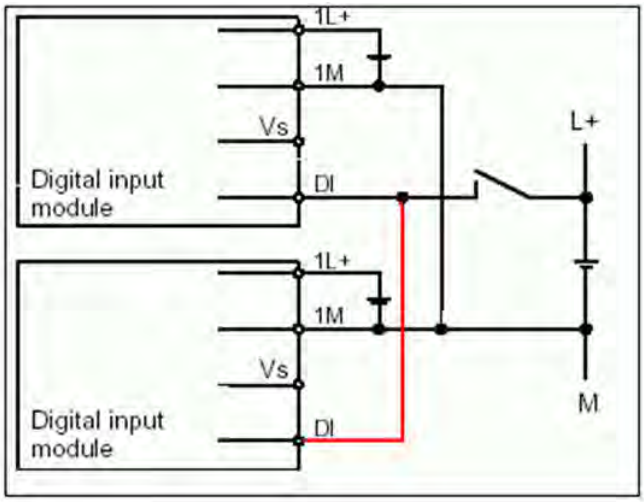
معنای آنچه که توسط ارزیابی *1oo1* و ارزیابی *1oo2* با ماژول سیگنال *fail-safe* (*SM 326F*) صورت می‌گیرد چیست دارند؟

پاسخ:

برای ارزیابی *1oo1*، به ازای هر سیگنال فرآیند یک سنسور تک کانال (*one single-channel sensor*) به ماژول‌های دیجیتال متصل می‌شود. با این حال، کاربردهای متعددی برای سنسورهای تک کانال وجود دارد. نحوه اتصال بستگی به کلاس ایمنی موردنیاز است.



اتصال یک ماژول

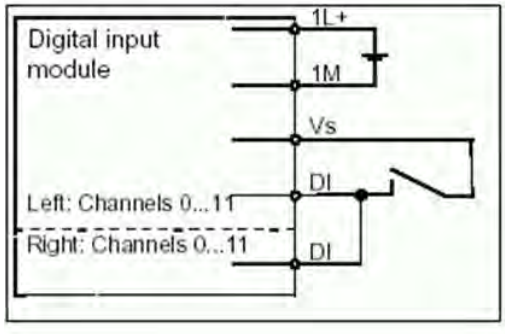


اتصال دو ماژول

نکته:

مطمئن شوید که برای اتصالی مانند شکل ۲، جامپر برای ورودی دیجیتال از ماژول دوم انجام شده باشد.

در ارزیابی *1oo2*، یک حسگر توسط یک ماژول بررسی می‌شود. یک سنسور به دو کانال مختلف که در آن کانال چپ و راست باید همیشه شماره کانال یکسانی داشته باشند، متصل می‌شود. یک نسخه دیگر از ارزیابی *1oo2* این است که در آن دو سنسور و هر یک به کانال‌های سمت راست و چپ کانال متصل می‌شود. در این جا نیز شماره کانال‌ها نیز باید یکسان باشد.



اتصال یک سنسور به دو کانال

Let us take an example of a 1oo2: an Emergency push button.
The push button is equipped with two contacts. Both contacts opens or close nearly at the same time (a few milliseconds difference is possible).
A 1oo2 input has two physical channels (let us say the left and the right channel).
One contact is connected to the left channel, the second to the right one.

But in your program you interrogate only one bit, because it is the input card that evaluates if both contacts deliver the same information.
In case of discrepancy (the contacts deliver a diffrent status) the failsafe input board will deliver a "0" (once the discrepancy timeout is elapsed).You will get a "1" only if both contacts deliver an ON signal.

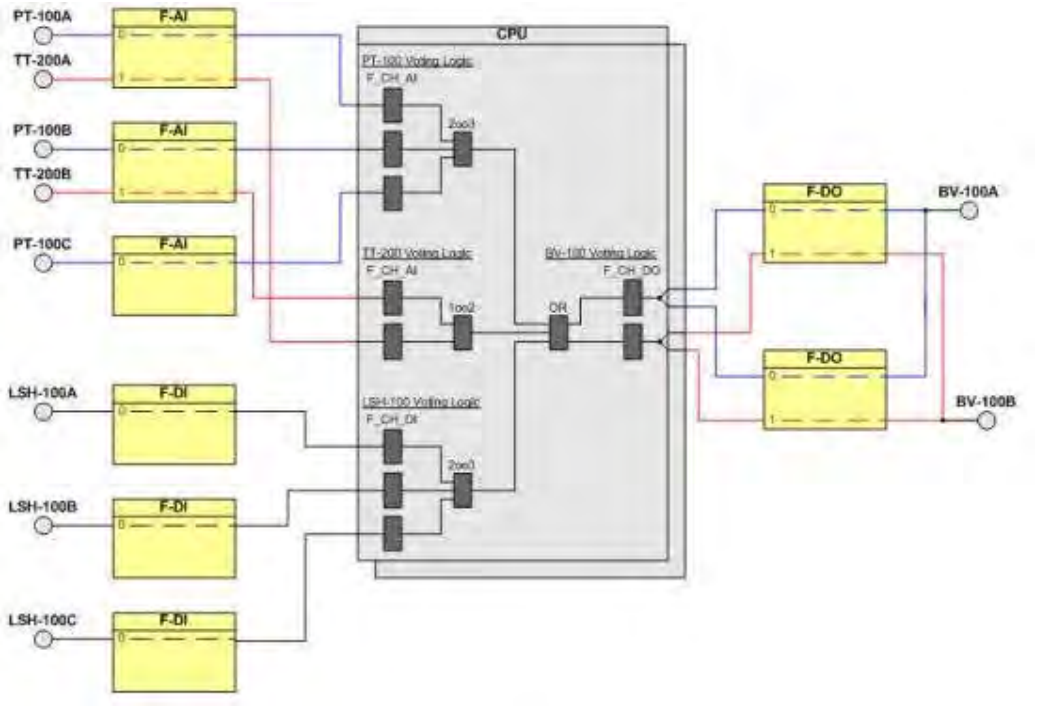
1-0-2- F Systems: Wiring and Voting Architectures for ET200M F-DIs and F-Dos

هدف:

چند سیگنال دیجیتال *failsafe* بایستی مانیتور شده و چند عملگر *failsafe* باید در یک پلنت کنترل شود. بسته به اهمیت و خطر شکست(*failure risk*)، گزینه‌های متعددی برای سیم‌بندی و داوری (*Voting*) سیگنال وجود دارد.

راه حل:

عمل داوری می‌تواند برای مثال، در ماژول ورودی دیجیتال و یا در *CPU* تحقق یابد. امکان‌های مختلف از سیم‌بندی و داوری سیگنال *failsafe* در این مثال عملکردی نشان داده شده است. همچنین سیمبندی وداوری خروجی‌های *failsafe* نیز نشان داده شده است.



۱–۱– کنترل کننده‌های *Fail-Safe* شرکت *Siemens*

شرکت زیمنس با نوآوری سیستم‌های ایمنی یکپارچه، راه حلی جدید برای طراحی کنترلرهای *Fail-Safe* ارایه کرده است که تمامی مشخصات یک کنترلر ایمن را در بر می گیرد.

شکل ۱– کنترلرهای *Fail-Safe*

این کنترلرها جهت کنترل فرایندها و همچنین هدایت و نگاه داشتن آنها در حالت ایمن در هنگام وقوع نقص و خطا در سیستم بکار می رود. در کنار کارایی بالای این کنترلرها، یکپارچه بودن نیز به خوبی رعایت شده است. با بکارگیری این کنترلرها می توان از عوارض و خسارات ناشی از نقص در عملکرد سیستم کنترل، جلوگیری نمود. بهبود در موضع یابی محل نقص به دلیل وجود اطلاعات جامع تشخیص خطا، از سرگیری تولید را پس از یک وقفه در سیستم *Safety*، قادر می سازد.

۱–۱–۱– چگونگی پردازش در یک کنترلر *Fail-Safe* شرکت *Siemens*

یکی از مشخصات اصلی یک سیستم *Fail-Safe*، تشخیص خطا می باشد. تا کنون شرکت‌های سازنده این نوع از کنترلرها، با بکارگیری دو عدد *CPU* که صحت عملیات را بازبینی می کردند، این مشخصه را محقق می ساختند. برخلاف دسترس پذیری بالای سیستم در جایی که *redundancy* ساختاری برای افزایش

دسترس پذیری سیستم اتوماسیون استفاده می شود، این دو کانال *redundancy* جهت آشکار کردن خطا بکار می رود.

بجای استفاده از روش تجاری فوق، راه کار ابتکاری *redundant* و *diverse* برای پردازش دستورات، در کنترلرهای *Fail-Safe* شرکت زیمنس مورد استفاده قرار می گیرد. در این طراحی با وجود بهره گیری از یک *CPU* (که دو پردازشگر درون آن جاسازی شده است)، می توان به همان قابلیت تشخیص خطا که در روش قدیمی بکارگیری دو *CPU* را ایجاب می کرد، دست یافت. شکل ۲ به خوبی اصول این روش را نمایش می‌دهد.

در این روش، برنامه کاربردی دو بار به صورت همزمان در دو پردازشگر مختلف در یک *CPU*، یکبار در منطق مثبت و بار دیگر در منطق متمم مورد پردازش قرار می گیرد. خطاهای احتمالی از مقایسه این دو نتیجه آشکار می شود. مزیت این روش به وضوح دیده می شود: تنها یک *CPU* برای حصول *Safety* مورد استفاده قرار می گیرد و بنابراین یک واسطه پیچیده برای تحقق سیستم *Fail-Safe* حذف می شود.

در پروسه‌هایی که از نظر *Safety* بایستی دسترس پذیری بیشتری داشته باشند، دو *CPU* با کابل نوری بهم متصل می شوند. در صورتیکه یک *CPU* دچار مشکل شود، *CPU* دیگر کنترل پروسه را ادامه می دهد.

برای دستیابی به استانداردهای ایمنی از *SIL1* تا *SIL3*، *Category4* و *AK6*، صرف نظر از پردازش به صورت *Diverse*، روتین‌های بازبینی و تشخیص خطا در این کنترلرها اعمال می گردند.

۱-۱-۲- *PROFIBUS* با پروتکل *PROFIsafe* برای ارتباطات *Safety*

PROFIsafe به عنوان اولین پروتکل مورد تایید *IEC 61508* که اجازه انتقال اطلاعات *Safety* و استاندارد را بر روی یک باس داده می دهد، شناخته شده است.

PROFisafe از همان اجزاء و سرویس‌های *PROFIBUS* برای ارتباطات *Fail-Safe* استفاده می کند. یک *PLC* از نوع *Fail-Safe* با زیر مجموعه ای از ماژول‌های *Fail-Safe*، اطلاعات لازم را بدون نیاز به سخت افزار اضافی برای این ارتباط تبادل می کند.

بنابراین *PROFIBUS* جهت کاربردهای *Safety*، بدون مجبور شدن به تغییر *Cabling* گسترش یافته است و راه حل‌های موجود بدون هیچ تغییری در سخت افزار قابل استفاده می باشند.

PROFIBUS با *PROFIsafe* دارای گواهینامه‌های استانداردهای زیر می باشد:

IEC 61508 (Up to SIL3(

EN954 (Up to Category4)
NFPA 79 - 2002

NFPA 85

شکل ۲- چگونگی استفاده از روش *redundant* و *diverse* برای پردازش دستورات در *PLC*های *Fail-Safe* شرکت زیمنس

شکل ۳- تا کنون سیستم‌های اتوماسیون *Fail-Safe* و استاندارد به صورت مجزا بودند!

شکل ٤- در حال حاضر با استفاده از تکنولوژی ایمنی یکپارچه شرکت *Siemens* سیستم‌های اتوماسیون استاندارد و *Fail-Safe* در یک سیستم تلفیق شده اند.

۱-۱-۳- مزایای استفاده از کنترلرهای *Fail-Safe* شرکت زیمنس

۱- کاهش تجهیزات مورد استفاده

- به کارگیری یک کنترلر برای اتوماسیون استاندارد و *Fail-Safe*
- امکان قرارگیری برنامه‌های استاندارد و *Fail-Safe* در یک کنترلر
- امکان استفاده از یک کابل *PROFIBUS* برای ارتباطات استاندارد و *Fail-Safe* تا *SIL3/Category4/ AK6*
- امکان استفاده از یک *I/O* توزیع شده برای ماژول‌های استاندارد و *Fail-Safe*

۲- پیکربندی و راه اندازی سریع سیستم

- یکپارچگی بهتر سیستم؛ بدلیل یکسان بودن عملیات مهندسی برای اجزای کنترلی استاندارد و *Fail-Safe*

۳- انعطاف پذیریبیشتر

- به جای سیم بندی‌های سخت افزاری، اساس کار بر نرم افزار می باشد.
- سهولت اعمال تغییرات، توسعه و مستند سازی
- پیاده سازی کارهای پیچیده مربوط به سیستم *Safety* با سهولت بیشتر

٤- قابلیت تکثیر سریعتر

- سادگی بیشتراعمال راه حل‌های نرم افزاری در قیاس با راه حل‌های سخت افزاری
- امکان بهره گیری از بلوک‌های نرم افزاری مجاز و استاندارد

۵- تعمیر و نگهداری[۱۳] ساده تر

- تعمیر و نگهداری ساده تر به علت وجود سیستم تشخیص خطای جامع

۱-۱-۴- کاربردها

تا کنون، کنترلرهای *Fail-Safe* شرکت *Siemens* در صنایع گوناگونی بکار رفته اند که تعدادی از آنها در ذیل امده است:

- *Processing industry*
- *Chemical Industry*
- *Pharmaceutical industry*
- *Automobile*
- *Mechanical Engineering*
- *Glass Industry*
- *Food and Luxury Industry*
- *Conveying Engineering*

همچنین وجود کتابخانه *F* که به صورت انحصاری برای سیستم‌های *Burner* طراحی شده است، استفاده از این نوع کنترلر ها را به صورت گسترده ای در صنایع مرتبط با *Burner* سبب شده است.

۱-۱-۵- استانداردها و گواهینامه ها

برای تایید کارکرد مناسب مربوط به ایمنی یک ماشین یا فرایند، اجزای حفاظتی مربوط به سیستم *Safety* و تجهیزات کنترل بایستی به گونه ای قابل اعتماد کار کرده و در هنگام بروز خطا فرایند مورد کنترل را به حالت ایمنی هدایت کنند.

به همین منظور هر محصول مربوط به این نوع از محصولات، بایستی برخی از شرایط و آزمون‌های خاص را با موفقیت پشت سر بگذارند. این شرایط در استانداردهای مختلفی گرد آوری شده اند. داشتن گواهینامه مربوط به یک استاندارد خاص برای یک محصول، سندی از برآورده شدن شرایط احراز استاندارد مورد نظر برای آن محصول می باشد. تمامی ماژول‌های *Fail-Safe* شرکت زیمنس شرایط استانداردهای زیر را بر آورده می سازند:

IEC 61508 (SIL3 تا)

*EN 954 (Category4*تا)

NFPA 79 - 2002
NFPA 85
TÜV certificate no: Z2 02 03 20411 009
UL 1998, UL 508, UL 991

۱- *Safety integrated*

۲- *Safe*

۳- *Availability*

٤- *Complementary*

٥- *Distributed I/O*

٦- *Configuration*

۷- *Commissioning*

۸- *Flexibility*

۹- *Modification*

۱۰- *Expansion*

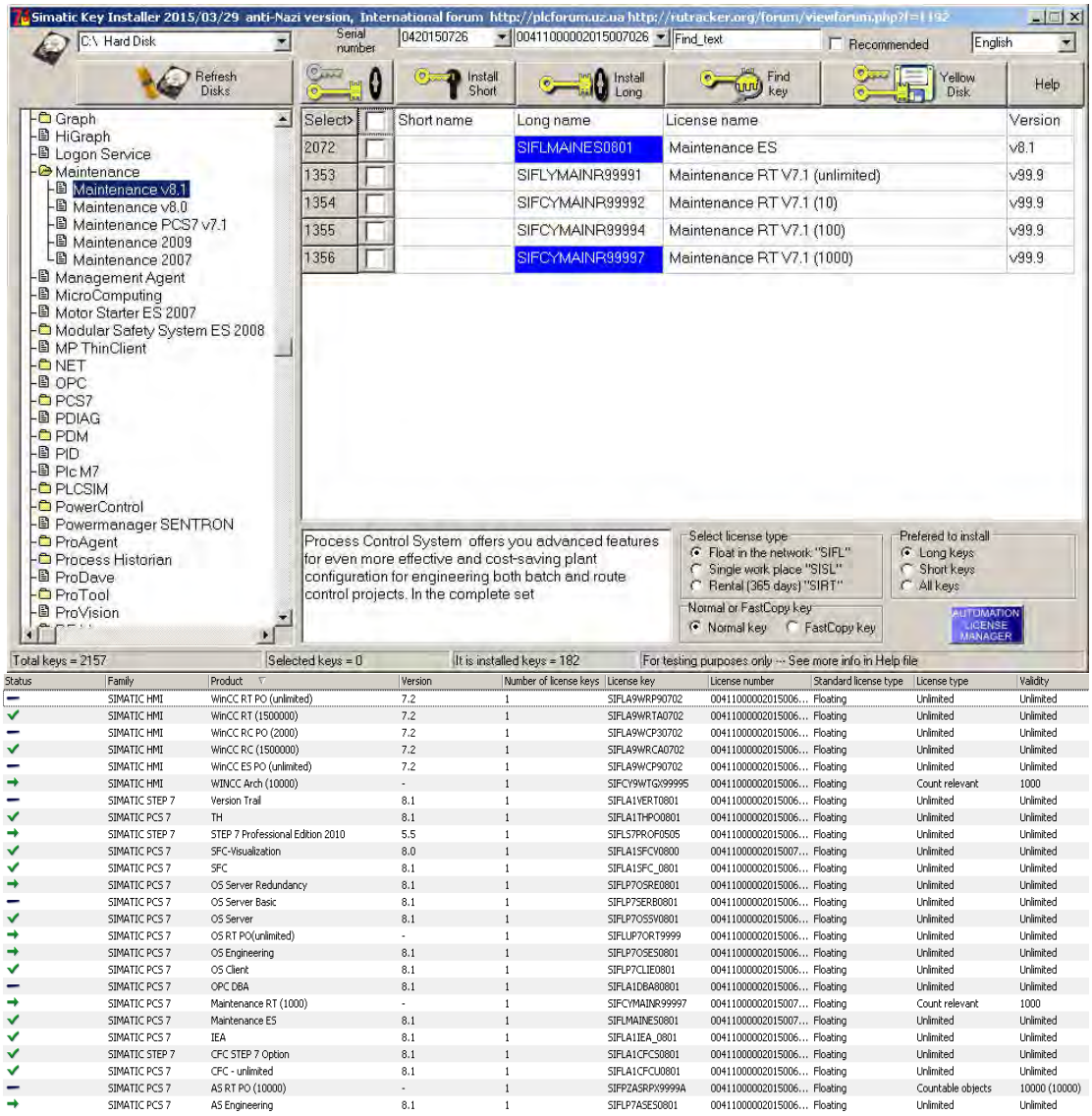
۱۱- *Documentation*

۱۲- *Reproducibility*

۱۳- *Maintenance*

فصل – ۱۰ نصب لایسنس در *PCS7*

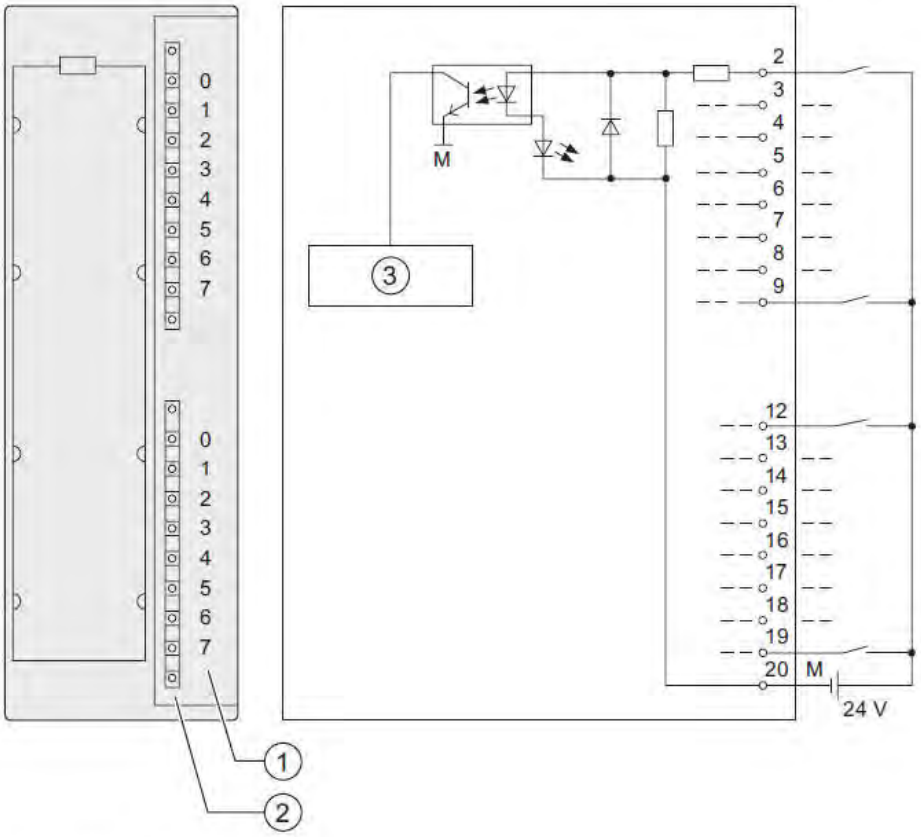
برای راه‌اندازی قابلیت *maintenance* در *PCS7* نسخه *8.1* نیاز است که لایسنس‌های زیر نصب شود.



۱۰-۱- سیم‌بندی ماژول‌های S7300

۱۰-۱-۱- سیم‌بندی ماژول‌های دیجیتال ورودی (DI)

- **6ES7 321-1BH02-0AA0**



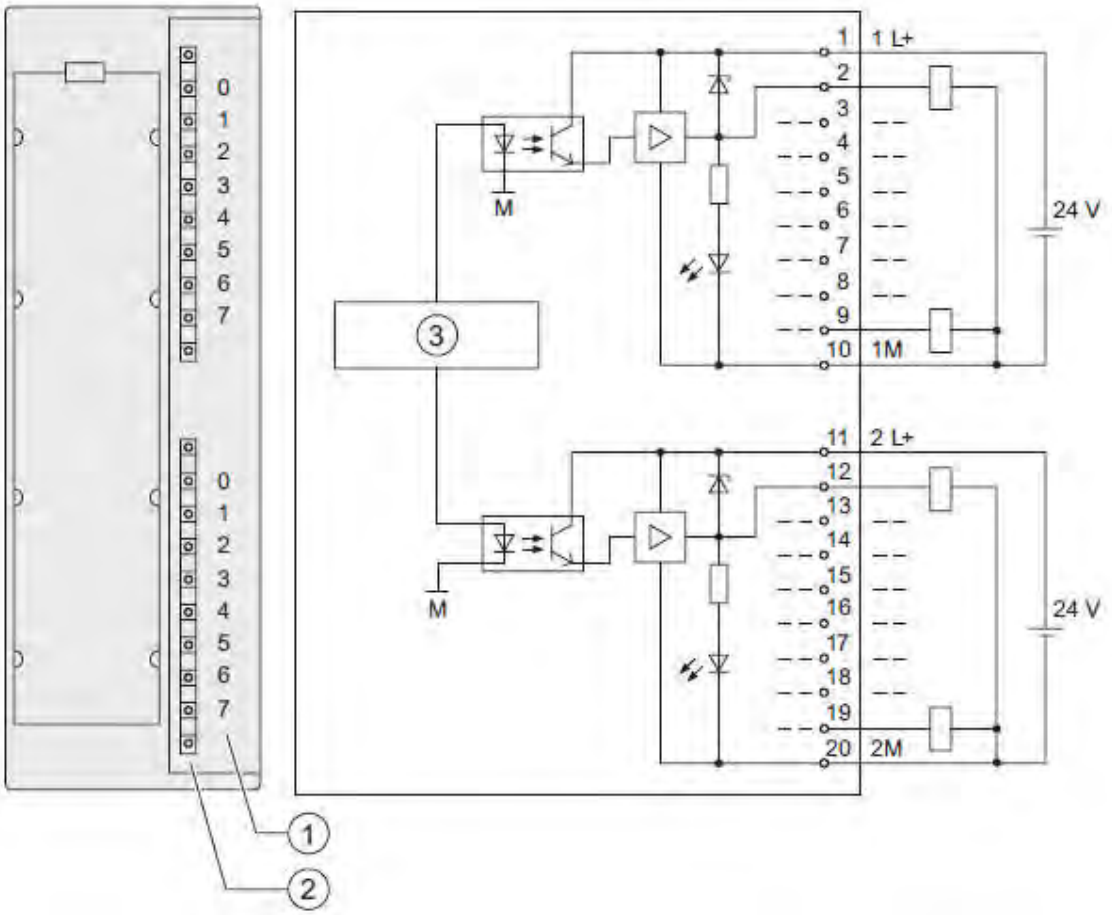
① Channel number
② Status display - green
③ Backplane bus interface

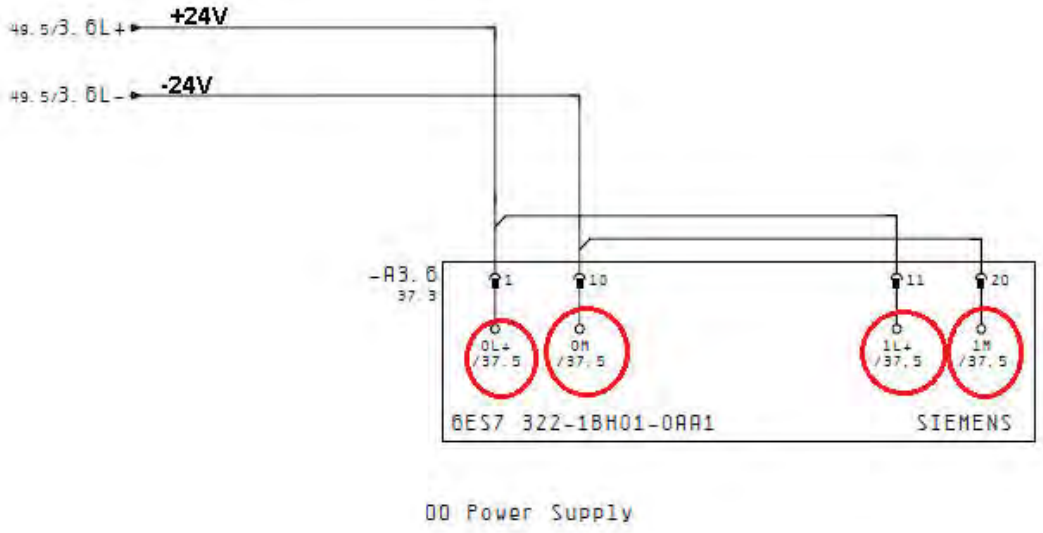
ماژول‌های *SIPLUS S7-300* را می‌توان در شرایط محیطی توسعه یافته (*extended environmental*) مورد استفاده قرار داد

- مناسب برای عملیات در بازهدمایی C ۲۵- تا C + ۶۰
- فشار مکانیکی مجاز افزایش یافته (*increased mechanical stress*)

۱۰-۱-۲- سیم‌بندی ماژول‌های دیجیتال خروجی (*DO*)

Digital output module SM 322; DO 16 x DC 24 V/ 0,5 A;(6ES7322-1BH01-0AA0)





DO Power Supply

۱۰-۱-۳- سیم‌بندی ماژول‌های آنالوگ ورودی (AI)

Analog input module SM 331; AI 8 x 13 Bit; (6ES7 331-1KF01-0AB0)

این ماژول برای اندازه‌گیری انواع سیگنال‌های زیر استفاده می‌شود

- *Voltage*
- *Current*
- *Resistance*
- *Temperature*

Wiring: 2-wire and 4-wire transducers for current measurement

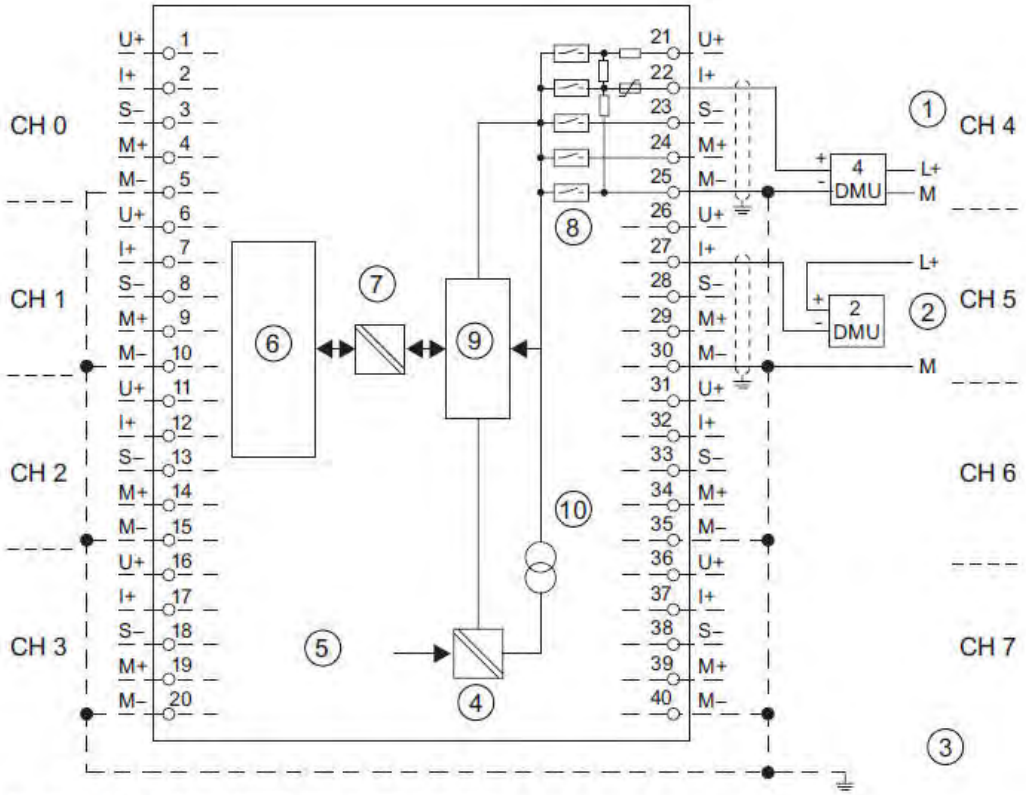
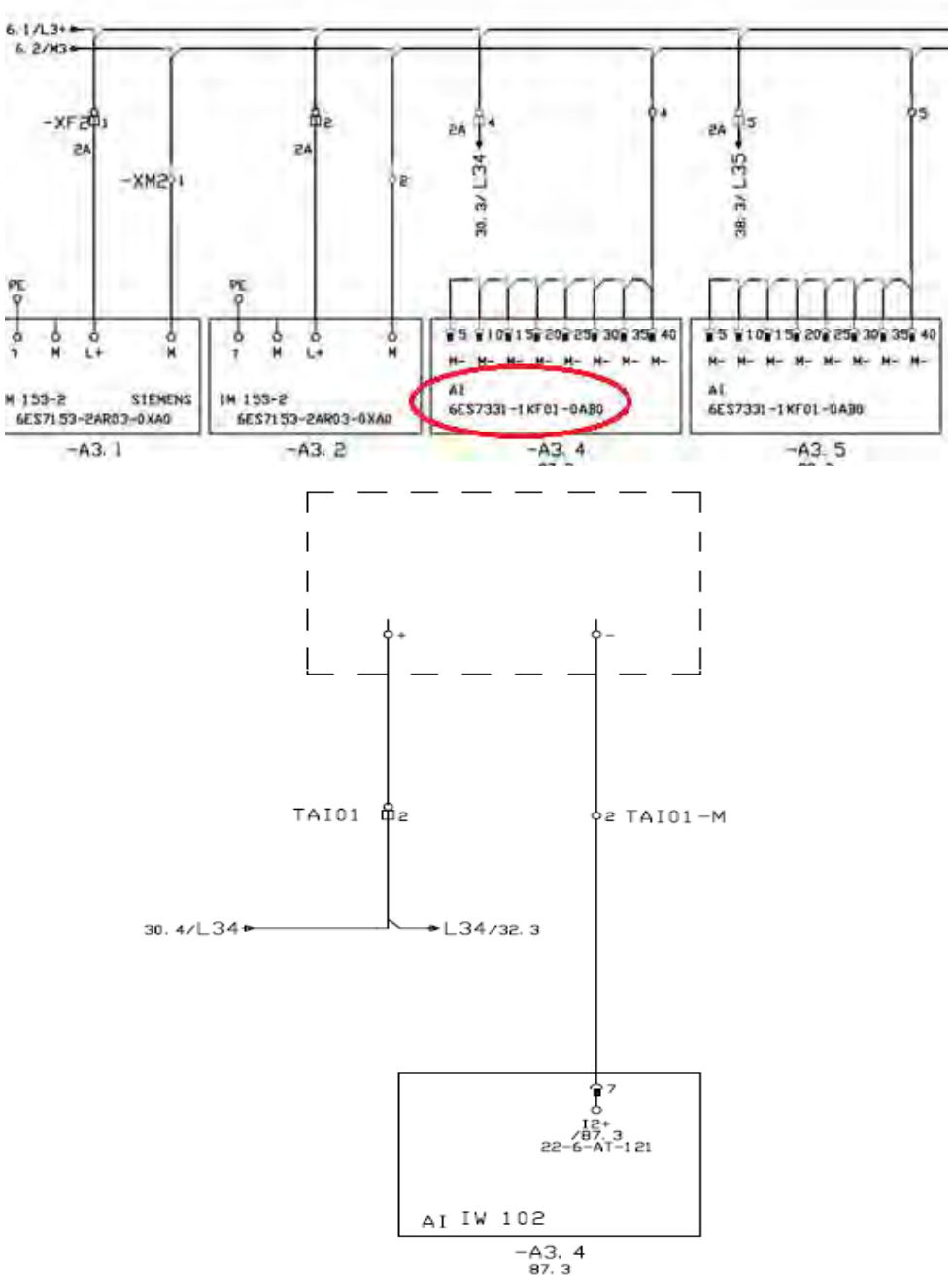


Figure 6-12 Block diagram and wiring diagram

① 4-wire transducer (0/4 mA to 20 mA or ± 20 mA)
② 2-wire transducer (4 mA to 20 mA)
③ Equipotential bonding
④ Internal supply
⑤ + 5 V from backplane bus
⑥ Logic and backplane bus interface
⑦ Electrical isolation
⑧ Multiplexer
⑨ Analog to Digital Converter (ADC)
⑩ Current source

برای اتصال سنسور جریان به *AI (6ES7 331-1KF01-0AB0)* تنها زمین (-) ۲۴ ولت به پایه‌های *M* ماژول که به هم متصل شده‌اند وصل می‌گردد.

خط ۲۴+ به سر سنور رفته و خط – برگشت از سنسور به ورودی منفی – ماژول می‌رود.



۱۰-۱-۴- سیم‌بندی ماژول‌های آنالوگ خروجی (AO)

Analog output module SM 332; AO 8 x 12 Bit; (6ES7332-5HF00-0AB0)

Wiring: Current output

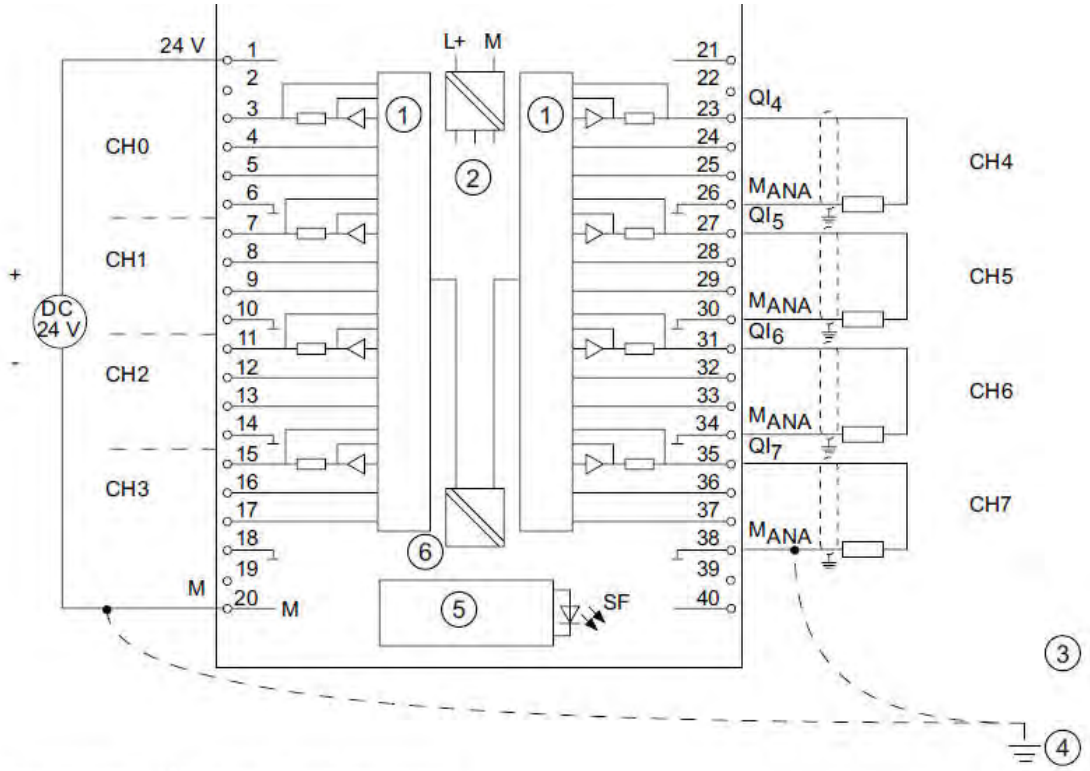


Figure 6-33 Wiring and block diagrams

Numeral	Description
①	DAC
②	Internal supply
③	Equipotential bonding
④	Functional earth
⑤	Backplane bus interface
⑥	Electrical isolation

Chanel	*Connection number*	*Chanel*	*Connection number*
Ch0	***3,6 →+,-***	***Ch4***	***23,26 →+,-***
Ch1	***7,10→+,-***	***Ch5***	***27,30 →+,-***
Ch2	***11,14 →+,-***	***Ch6***	***31,34 →+,-***
Ch3	***15,18 →+,-***	***Ch7***	***35,38 →+,-***

فصل – ۱۱ ماژول‌های *SIPLUS* زیمنس

q- کسی اطلاع داره که *PLC*های معمولی زیمنس(و نه *siplus*) یا *ABB* چقدر در مقابل خورندگی گازهایی مثل *H2S* مقاومند؟ مثلا *S7-400* تا میتونه بطور دایم زیر چه حدی از گاز برحسب *ppb* کار کنه؟یا تو چه گروهی قرار می‌گیره؟(*G1-G4*)

A- تا جایی که اطلاع دارم PLCهای زیمنس Outdoor نیست و باید در داخل تابلو کنترلی و در اتاق کنترل با شرایط تعریف شده باشه.

A- البته بنظرم این مقاومت در برابر خورندگی که مد نظر شماست هم مربوط به تابلو میشه نه خود کنترل‌کننده

A- شما میتوانید از سری cpu های siplus زیمنس استفاده کنید که در محیط های اسیدی مورد استفاده قرار می‌گیرند. که می توانند خارج از تابلو های کنترلی قرار بگیرند.

A- اما گازهای خورنده میتونند indoor هم باشند که داخل تابلو، اولین جای است که اسیب میرسونه، کنترلره. بر اساس میزان خورندگی، شرایط indoor میتونه G1 تا Gx باشه.

Ambient Climatic Conditions

You can use the S7-400 under the following ambient climatic conditions:

Table 1-10 Ambient Climatic Conditions

Climatic Conditions	Permitted Range	Remark
Temperature	0 to +60 °C	
Temperature change	Max. 10 °C/h	
Relative humidity	Max. 95 % at +25 °C	No condensation, corresponds to RH stressing level 2 in accordance with IEC 61131-2
Atmospheric pressure	1080 to 795 hPa (corresponds to a height of -1000 to 2000 m)	
Concentration of contaminants	SO_2: < 0.5 ppm; RH < 60 %, no condensation H_2S: < 0.1 ppm; RH < 60 %, no condensation	Test: 10 ppm; 4 days Test: 1 ppm; 4 days

Climatic environmental conditions

The S7-300 may be operated on following environmental conditions:

Environmental conditions	Permissible range	Comments
Temperature: horizontal mounting position: vertical mounting position:	 0°C to 60°C 0°C to 40°C	
Relative humidity	10 % to 95 %	No condensation, corresponds to relative humidity (RH) Class 2 to IEC 61131, Part 2
Barometric pressure	1080 hPa to 795 hPa	Corresponds with an altitude of -1000 m to 2000 m
Concentration of pollutants	SO2: < 0.5 ppm; RH < 60 %, no condensation H2S: < 0.1 ppm; RH < 60 %, no condensation	Test: 10 ppm; 4 days Test: 1 ppm; 4 days

SIPLUS extreme
Designed for demanding conditions

Industrial applications that have to withstand extreme environmental conditions require special product features to function cost-effectively and reliably. Sewage treatment plant digester towers or desalination plants are typical areas which illustrate the higher requirements that must be met by electronic components and accessories. SIPLUS extreme products are extremely robust and are therefore predestined for use in these difficult environmental conditions.

Environmental conditions	SIPLUS extreme		
Temperature range	From –40/–25 to +60/+70 °C[1]		
Relative humidity	100% humidity, condensation and ice formation are permitted		
Pollutant concentrations	EN60721-3-3 3C4 and ISA S71.04 G1, G2, G3, GX		
		Permanent load	Limit value [2]
	H_2S	9.9 ppm	49.7 ppm
	Cl	0.2 ppm	1.0 ppm
	HCl	0.66 ppm	3.3 ppm
	At RH < 75%, condensation allowed		
Salt mist	Salt mist test (EN 60068-2-52)		
Mechanically active substances – Dust (total suspended solids) – Dust (deposit)	EN60721-3-3 3S4 including conductive sand/dust ("Arizona dust") 4.0 mg/m³ 40 mg/(m².h)		
Biologically active substances	EN60721-3-3 3B2 mold growth, fungus (excluding fauna)		

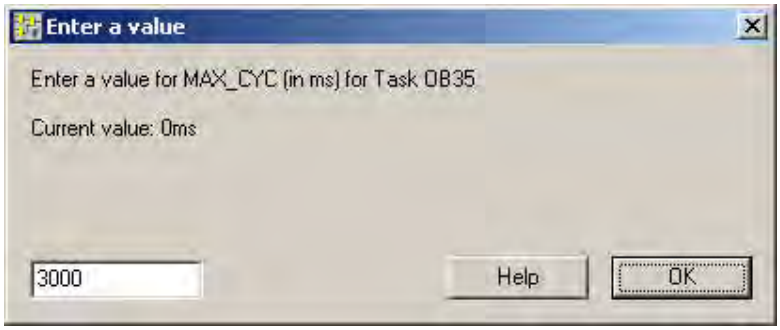
[1] For specific product families
[2] 30 min/day

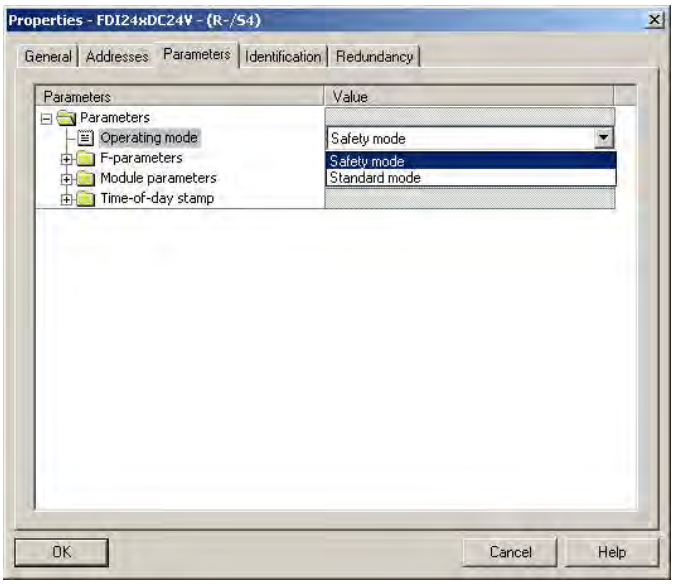
برخی از کارت های ورودی دیجیتال نوع *F* عبارتاند از:

6ES7 321-7TH00-0AB0	*Digital input module DI 16xNAMUR , with diagnostics and value status, reconfigurable online*

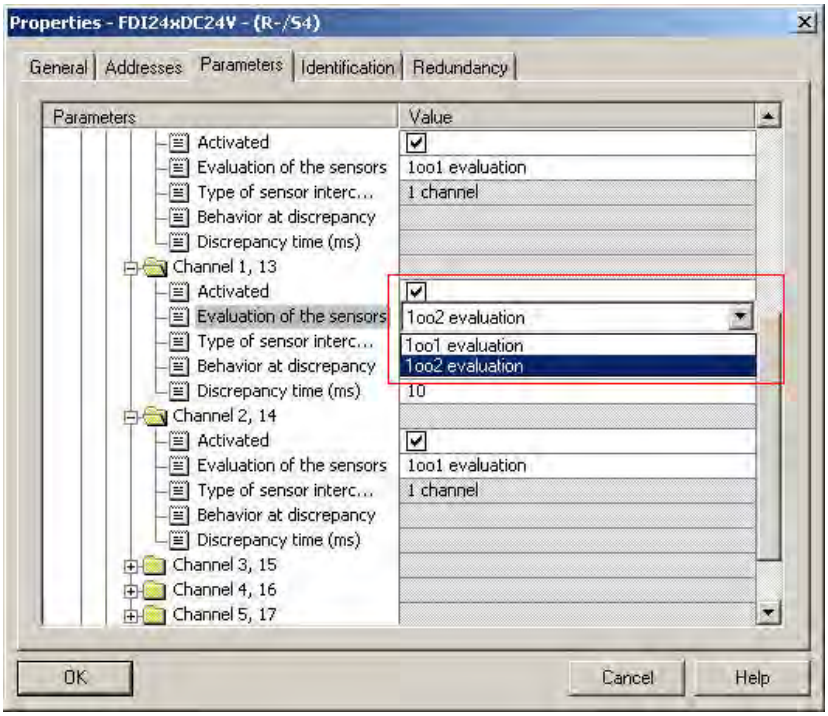
موقع درج کردن بلاک های *F* به محیط *CFC* پنجره ظاهر می شود که در آن می توان برای برنامه *f* پسورد انتخاب کرد

موقع دابل کلیکی روی یک سیگنال که به برنامه *F* مرتیط است پنجره زیر آشکار می شود.

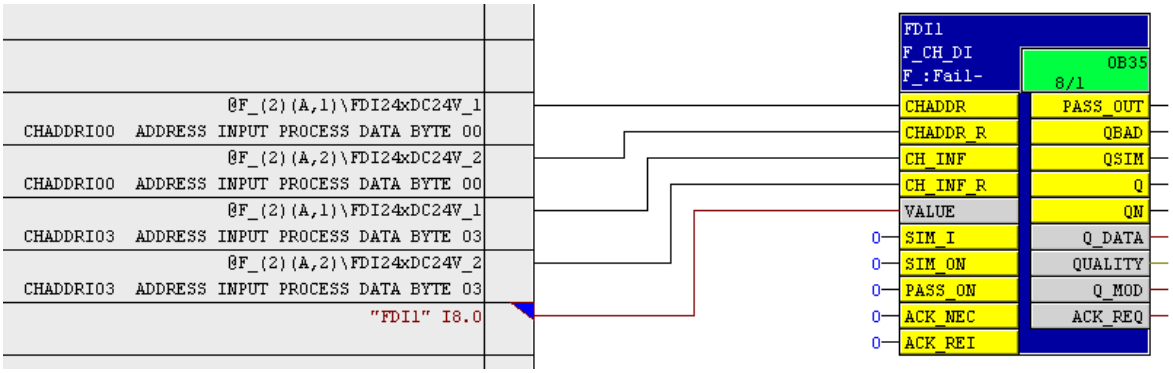




تنظیم کارت *FDI* برای مد *Safety*

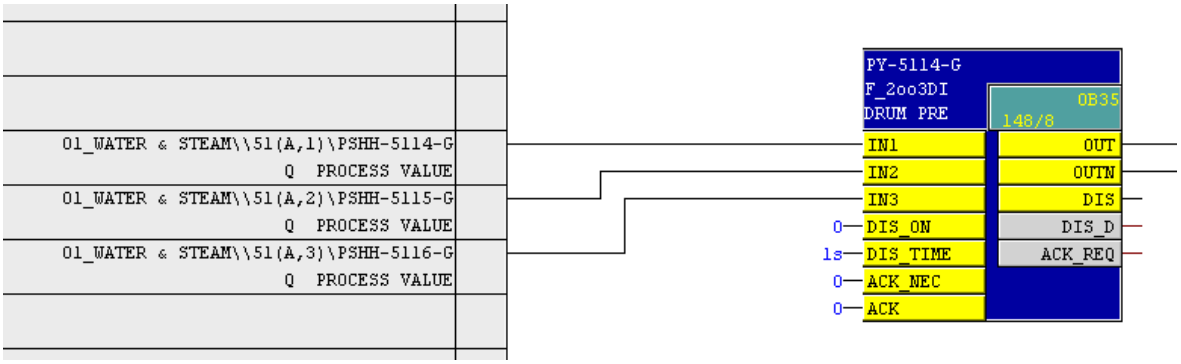


تعیین مد ارزیابی *1oo2* یا *1oo1*

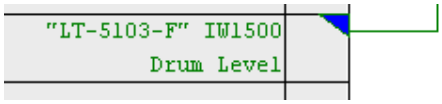


ماژول ارزیابی سه ورودی دیجیتال(*F_2oo3DI*)

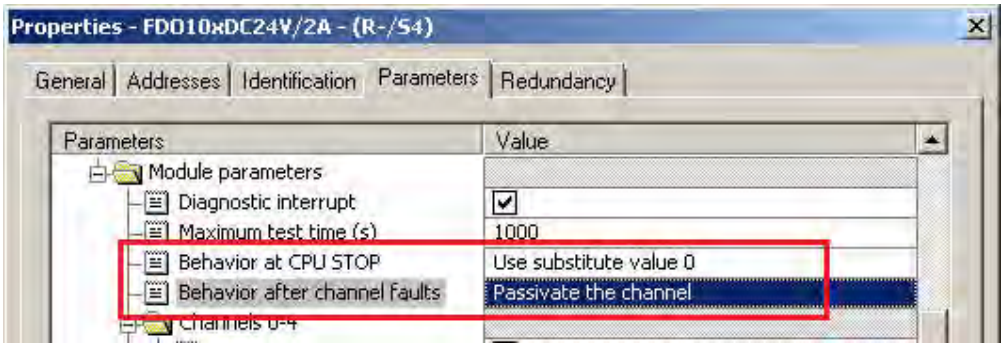
F_2oo3DI: 2oo3 evaluation of inputs of data type BOOL with discrepancy analysis
This block monitors three binary inputs for signal state 1. The OUT output is 1 when at least two inputs INx are 1. Otherwise the output is OUT 0. The OUTN output corresponds to the negated OUT output.



نکته : آدرس‌های واقع در کارت‌های *I/O* با مثلث آبی رنگ ظاهر می‌شوند



نکته : در پراپرتی یک بلاک *F-DO* و *F-DI* رفتار کانال پس از وقوع فالت به *Passivate the chanel* تعیین شده است.

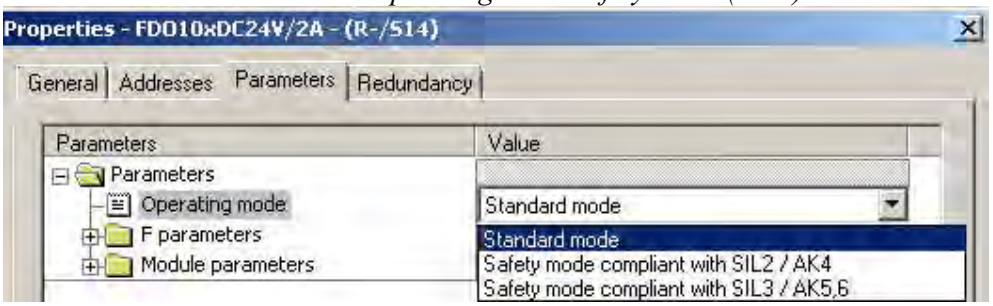


شکل ۱۱-۱-

➢ مدکاری ماژول‌های سیگنال *fail-safe*

برای ماژول‌های سیگنال *fail-safe* می‌توان مد کاری را از طریق پارامتر *Operating mode* در سربرگ *Parameters* پنجره پراپرتی ماژول سیگنال تعین کرد که در یکی از مدهای استاندارد یا *F* کار کند. ماژول‌های زیر مستثنی می‌باشند.

- *6ES7 326-2BF10-0AB0* → *Operating mode:safety mode (SIL3)*



اطلاعات مربوط به صفحات *Diagnostc* از چارت‌های @ که به صورت خودکار با کامپایل چارت ایجاد شده‌اند، گرفته می‌شود.

۱۱-۱-۱- فعال‌سازی کانال‌های *IO* (*Reintegration*)

با توجه به اینکه موقع وقوع یک فالت در برنامه یا سخت‌افزار، مقدار *failsafe* تعریف شده برای کانال‌های *I/O* در ورودی/خروجی‌ها قرار می‌گیرد. لذا برای فعال سازی مجدد کانال‌های نیاز به عمل *Depassivation* یا *Reintegration* می‌باشد.

تعویض از مقادیر قرارگرفته در خروجی‌های داریور کانال در زمان غیرفعال شدن(fail-safe) به مقادیر فرایند (reintegration of an F-I/O) یا به طور خودکار و یا پس از عمل *user acknowledgment* روی درایور *F-Channel* اجرا می‌شود.

روش *Reintegration* به موارد زیر بستگی دارد:

- عامل غیرفعال کردن کانال‌های *F-I/O* یک *F-I/O*
- پارامترهای اختصاص داده شده به درایور کانال *F*

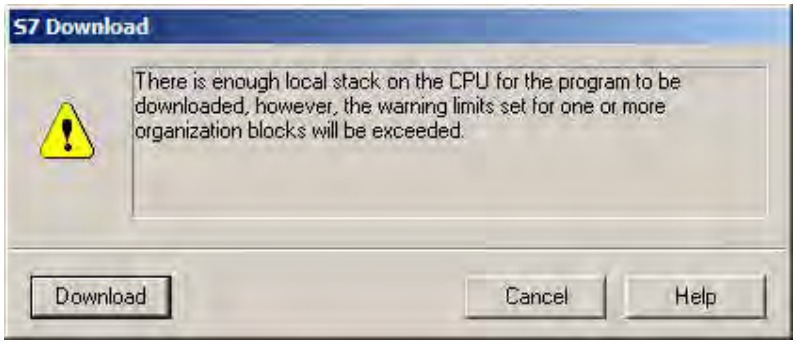
نکته : اگر با فشردن کلید *F1* بر روی یک بلاک از کتابخانه *F*، فایل *Help* باز نوشد، بایستی برنامه *Html help* ویندوز را *Update* کرد. (*Windows6.1-KB917607-x64*)

8. *max reaction time (max time between change at a physical input and reaction on the physical output).*

فصل – ۱۲ – مدیریت خطاها

۱۲-۱-۱- خطا در عبور از مقدار *Local data stack*

Error: There is enough local stack on the CPU for the program to be downloaded, however, the warning limits set for one or more organization blocks will be exceeded.



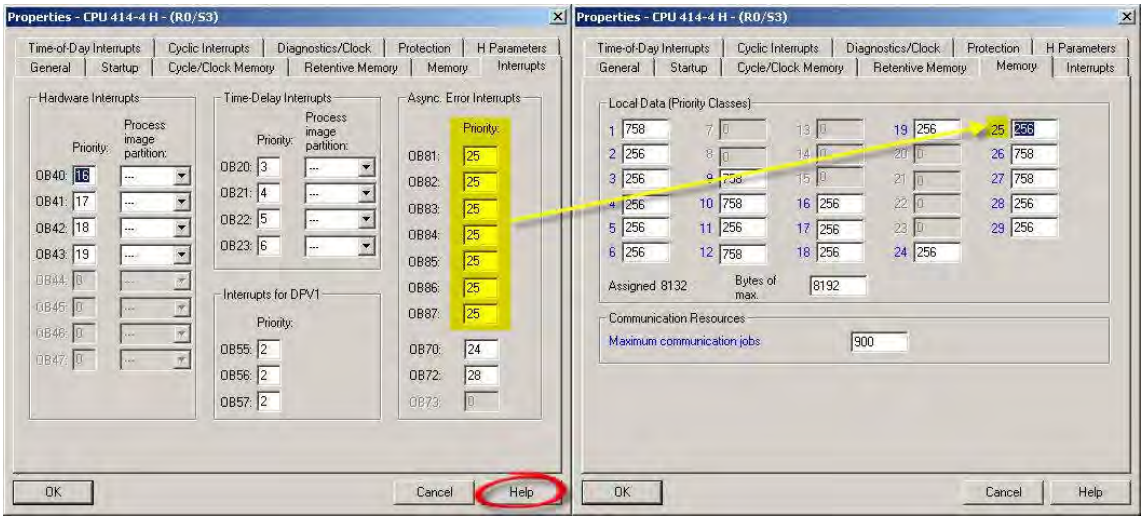
here's two ways to solve your problem:

1.) Reduce the numberof TEMP variables for the Blocks that are called from the OB's in question to reduce the local data requirements.

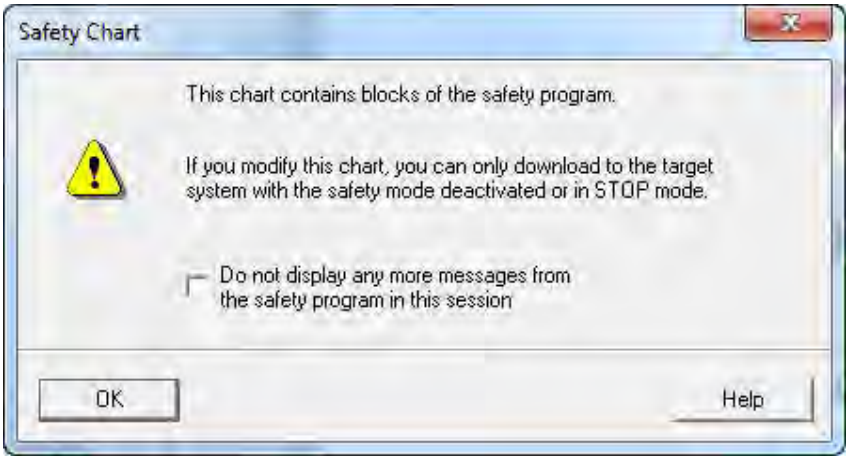
2.)Change the amount of local data that is available for the error OB's (this is what I would do).
To do so, go into HWconfig double click on your CPUand adjust the Local data size for the priority class to which your OB's belong(better explained in the attached pic).

***Please note:** Do yourself a favour and use the HELP button on the "Memory" tab and read up on the ramifications of changingthe size of the Local data!*

I hope this helps

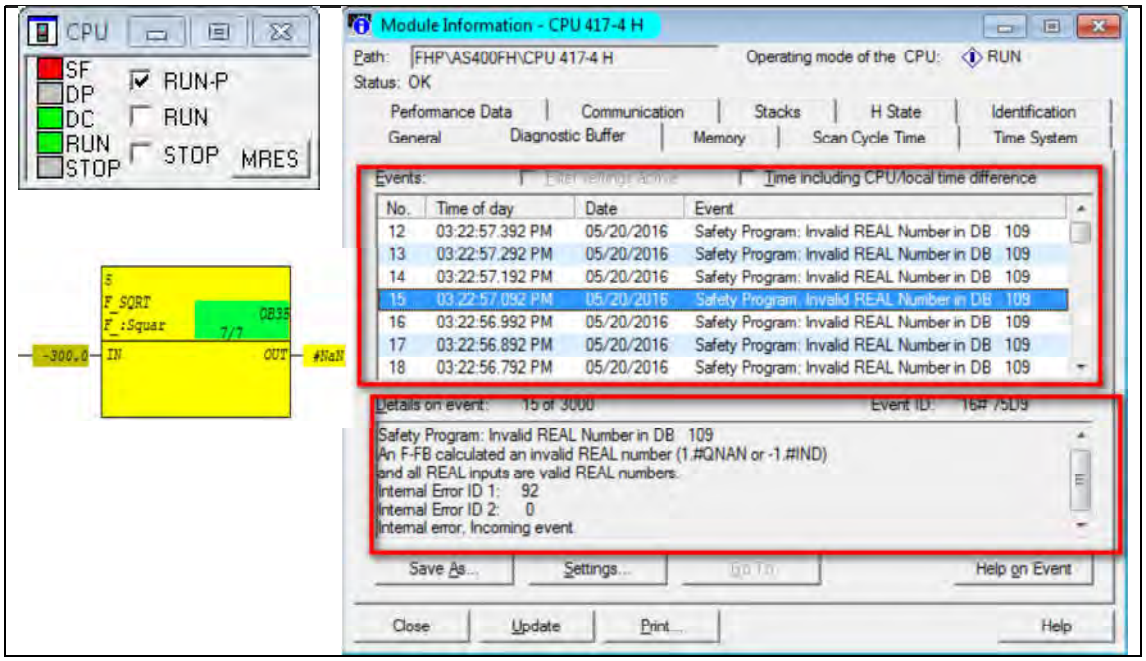


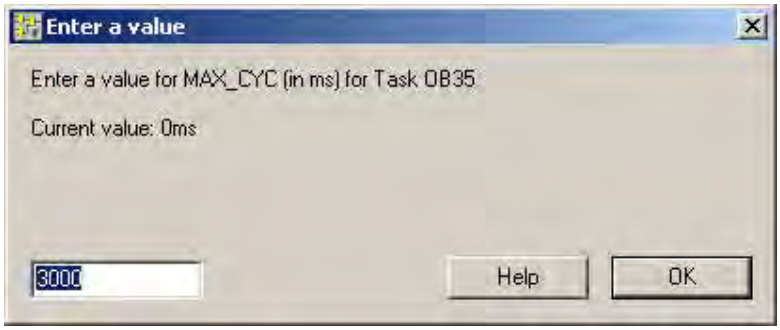
در صورت باز کردن یک چارت حاوی برنامه *F* پیغام زیر ظاهر می‌شود. به طوری که به کاربر هشدار می‌دهد که در صورت تغییر در لاجیک *F* ، بایستی برای دانلود تغییرات به *CPU* مد *Safety* در *CPU* غیرفعال شود.



۱۲-۱-۲- خطای

این خطا مربوط به بلاک‌های محاسباتی می‌باشد. به عنوان مثال در یک چارت از یک بلاک *SQRT* استفاده شده بود. این بلاک مربع یک عدد مثبت را محاسبه می‌کند. ولی از آنجایی که در ورودی این بلاک مطابق شکل زیر یک عدد منفی ظاهر می‌شد منجر به خطای *s7 safety program : invalid real number in db109* شده و چراغ *SF* ماژول *CPU* روشن می‌شود.

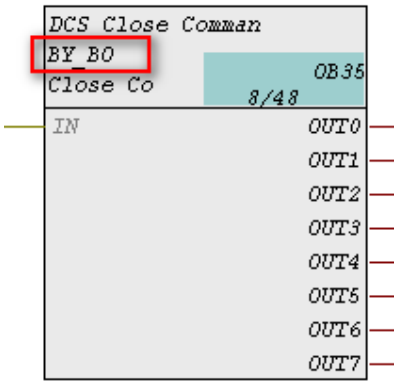




۱۲-۲- توابع استاندارد

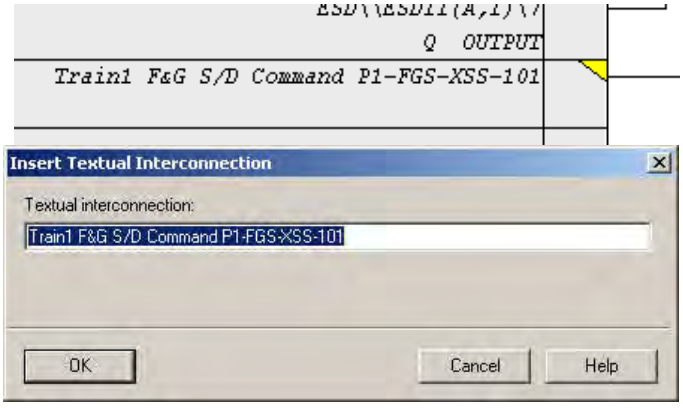
۱۲-۲-۱- فانکشن بلاک *BY_BO*

این تابع یک بایت (*BYTE*) ورودی را به هشت بیت (*BOOL*) جدا می‌کند.



۱۲-۳- علائم در چارت‌های CFC

۱. رنگ زرد برای اتصالات از نوع *Texual Interconnection* می‌باشد.



۲. رنگ سفید برای *Chart I/O* می‌باشد.