Portál pre odborné publikovanie ISSN 1338-0087

Diagnostické funkcie PLC

Bubniak Patrik · Elektrotechnika

26.02.2014

Článok sa zameriava na vysvetlenie základných princípov diagnostiky PLC pomocou funkčných blokov v prostredí Unity Pro. V článku je uvedená konfigurácia aplikačnej a systémovej diagnostiky s jednoduchými ukážkami diagnostiky pomocou odvodených funkčných blokov (DFB – Derived function block).

Úvod

Diagnostika ako taká, je veľmi potrebná, či už vo vývojovom procese alebo v procese výroby, pretože zabezpečuje predchádzanie úrazom alebo poškodeniu zariadenia a v konečnom dôsledku aj minimalizuje prestoje výrobných liniek. Diagnostika ako termín určuje procedúry a testy vykonávané za cieľom zistenia chýb, porúch zariadenia. Diagnostické procesy sa nesnažia zistenú vadu opraviť, ich podstata je skôr informačná a testovacia, zistenú informáciu posúvajú ďalej buď procesom, ktoré sú určené na reakciu na danú zistenú vadu alebo poruchu alebo iba ako informáciu pre používateľa.

V tomto článku sa budeme venovať stručnému úvodu do diagnostiky programovateľných logických automatov (PLC – Programmable logic controller) som zameraním sa hlavne na ich diagnostické funkcie softvéru Unity Pro XL od spoločnosti Schneider Electric a na konci článku si ukážeme niekoľko príkladov diagnostických blokov.

Diagnostika a PLC

Význam diagnostiky ako cudzieho slova je proces určovania choroby podľa špecifických príznakov pomocou diagnostických metód. V našom prípade je diagnostika proces určovania chýb, porúch a stavov buď procesu alebo zariadenia. Všeobecne môžeme diagnostiku rozdeliť na aplikačnú a systémovú. Programovateľné logické automaty obsahujú prídavné funkcie alebo systémové bity a slová, vďaka ktorým je lokalizácia porúch jednoduchšia. Systémové poruchy môžeme podľa závažnosti rozdeliť do dvoch kategórií:

1. Fatálne – závažné poruchy.
2. Nefatálne – menej závažné poruchy.

Keď sa objaví fatálna porucha, spôsobí prepnutie PLC do režimu STOP a nastavenie

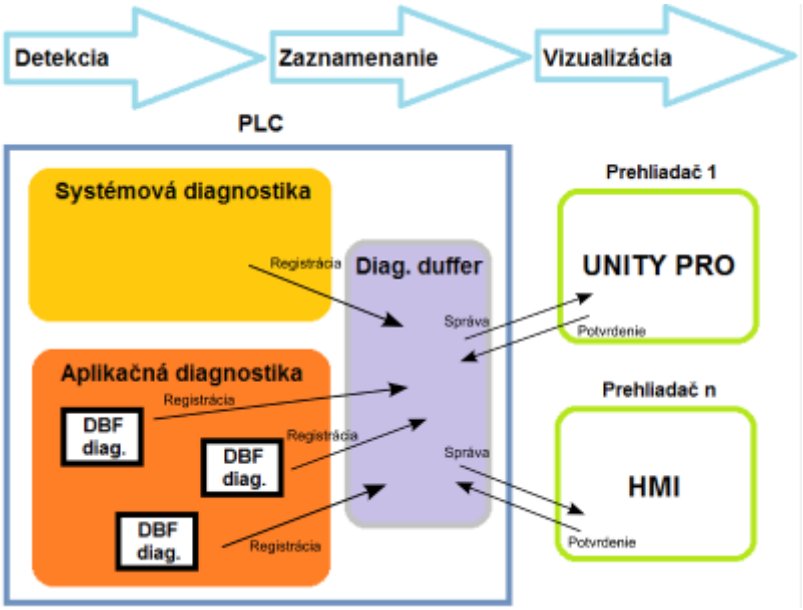
jeho výstupov do preddefinovaných stavov. Jedná sa o tzv. blokujúcu poruchu ako napr. presiahnutie watchdogu, vykonanie inštrukcie HALT, krok grafcetu je neurčený a pod. Riadiaci systém vyžaduje určitý zásah programátora na odstránenie poruchy resp. reštartovanie PLC čo fyzicky neodstráni príčinu poruchy.

Na rozdiel od fatálnych porúch, menej závažné poruchy (neblokujúce poruchy) nebránia PLC v chode a po detekcii takejto poruchy, môže PLC v režime RUN pokračovať vo vykonávaní programu, avšak je dôležité tieto poruchy neignorovať a čo najskôr ich opraviť. Môžu to byť chyby ako delenie nulou, pretečenie indexu, chyba pri počítaní s pohyblivou rádovou čiarkou a iné. Okrem systémovej diagnostiky poznáme ešte užívateľom definované poruchy (aplikačná diagnostika) Užívateľ má väčšinou možnosť sám si zadefinovať vlastné poruchy a chybové hlásenia pre špecifické situácie. Existujú tri typy používateľom definovaných porúch (alarmov):

1. Fatálne poruchy.
2. Menej závažné poruchy.
3. Špeciálna inštrukcia, ktorá posiela správu do programovej konzoly alebo počítača.

Pre každý alarm musí byť určený kód a správanie sa. Rovnaký kód nesmie byť priradený aj fatálnej aj menej závažnej poruche. Väčšinou je to však programovo ošetrené, aby nedošlo k takejto možnosti. Špeciálna inštrukcia, ktorá posiela správu do programovej konzoly alebo počítača (MSG) sa používa pre vypisovanie správ na displeji programovej konzoly. Pre diagnostiku v PLC je veľmi dôležitý „diagnostický buffer“ (obr. 1). Diagnostic Buffer je miesto kam sa ukladajú všetky chybové hlásenia a všetky zaznamenané udalosti. Tieto záznamy sa ukladajú chronologicky a s časovou značkou, kedy boli zaznamenané. Obvykle sa ukladajú chybové kódy a ich popisy. Predstavuje to jednoduchý a prehľadný zoznam porúch zaznamenaných počas procesu, ktorý je možné zobraziť na HMI, programovacom nástroji a pod. Výhodou je:

1. Timestamping v zdroji poruchy.
2. Poradie porúch.
3. Zapamätanie prechodných porúch.
4. Rovnaká správa pre všetky vizualizačné prostriedky.
5. Multi-action management (jediné potvrdenie).
6. Poruchy aplikácie ošetrované z jedného bodu.



Obr. 1 Princíp zápisu a čítania informácii z/do diagnostického buffra.

Unity Pro

Unity Pro je programovací software spoločnosti Schneider Electric pre PAC Modicon. Tento softvér používa 5 IEC 61131-3 programovacích jazykov – ladder diagram (LD), Instruction list (IL), Structured text (ST), sequential function chart (SFC) a function block diagram (FBD). Unity Pro obsahuje PLC simulátor pre validáciu užívateľom vytvoreného programu pred nasadením na zariadenie. Súčasťou simulátora je aj testovacie a diagnostické programovacie centrum.

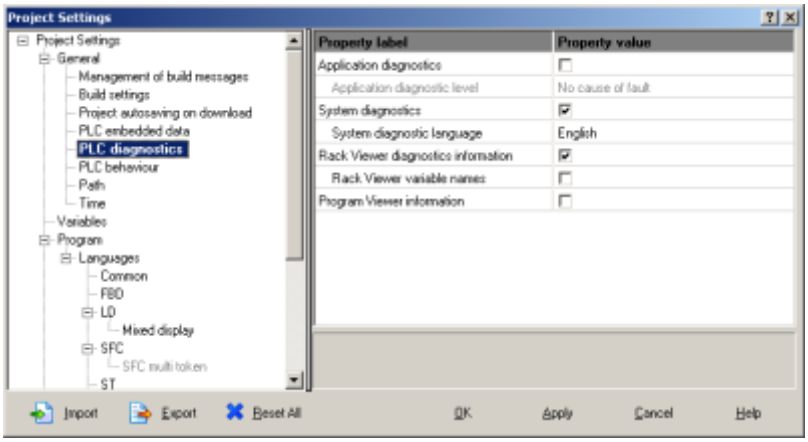
V podstate môžeme povedať, že tento software obsahuje „všetko v jednom“ programovom balíku, čím značne zjednodušuje vývojový proces programu pre užívateľa, keďže užívateľ nájde všetko na jednom mieste. Unity Pro je určený pre radu Modicon (Modicon Quantum , Modicon Premium a Modicon M340) a má širokú aplikáciu hlavne v riadení priemyselnej výroby, strojov a infraštruktúre.

Nastavenie diagnostiky v Unity Pro

V nastaveniach projektu (Project Settings) v časti PLC diagnostics (obr. 1) sa špecifikuje aplikačná a systémová diagnostika. Pri zapnutej systémovej diagnostike je možné zvoliť jazyk v ktorom sa budú hlásenia zobrazovať. Aplikačná diagnostika umožňuje okrem využitia diagnostických blokov DFB aj:

- „No cause of fault“ – Nehľadá príčinu vzniknutej poruchy.
- „Local diagnostic“ – Príčina poruchy sa vyhľadáva v sekcii v ktorej bol zaznamenaný alarm.
- „Global diagnostic“ – Príčina poruchy sa vyhľadáva v celej aplikácii.

Ak je zapnutá lokálna alebo globálna diagnostika, tak sa pre ne zobrazuje maximálne 16 príčin ich vzniku.



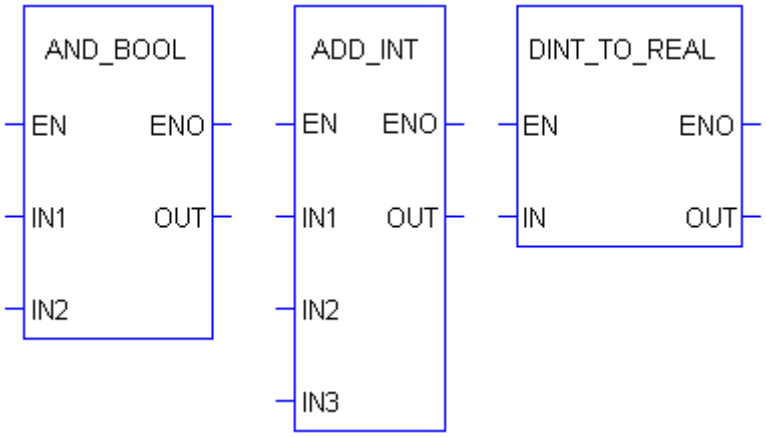
Obr. 2 Záložka nastavenia diagnostiky v Unity Pro

Funkcie a funkčné bloky Unity Pro

V Unity Pro sa používajú rôzne druhy funkčných blokov. Poznáme štyri základné typy:

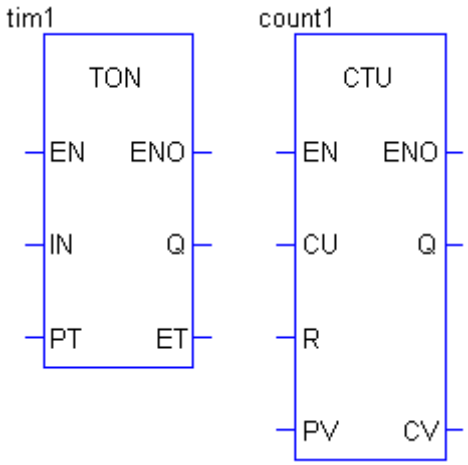
- Základná funkcia (EF-Elementary Function)
- Základný funkčný blok (EFB-Elementary Function Block)
- Odvodený funkčný blok (DFB-Derived Function Block)
- Procedúra

Základná funkcia je funkcia, ktorá nemá vnútorný stav a má len jeden výstup. V grafických jazykoch (FBD a LD) je reprezentovaná blokom s vstupmi vľavo a výstupmi vpravo (Obr. 3). Meno funkcie (resp. typ funkcie) je zobrazené v strede bloku. Ak majú vstupy rovnakú hodnotu, tak hodnota na výstupe je rovnaká pre každé vykonávanie funkcie.



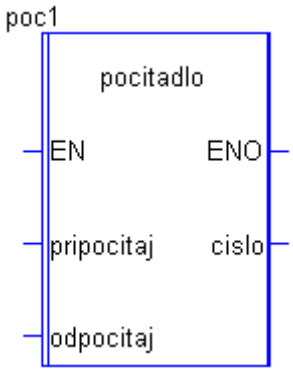
Obr. 3 Ukážky základných funkcií v Unity Pro v jazyku LD (AND_BOOL – operácia AND nad boolovkými premennými, ADD_INT – inštrukcia sčítania premenných dátových typov INT, DINT_TO_REAL – inštrukcia pretypovania dátového typu DINT na REAL)

Základný funkčný blok je funkčný blok, ktorý má na rozdiel od základnej funkcie vnútorný stav. Ak majú vstupy rovnakú hodnotu, hodnota na výstupoch sa môže líšiť pre viaceré vykonávanie funkcie. Meno funkčného bloku (resp. typ funkčného bloku) je zobrazené v strede bloku. Je reprezentovaný vstupmi na ľavej strane a výstupmi na pravej strane bloku. Nad názvom funkčného bloku sa nachádza inštancia (názov dátového bloku) pre daný funkčný blok.



Obr. 4 Ukážky základných funkčných blokov v Unity Pro v jazyku LD (TON – časovač s inštanciou tim1, CTU – počítadlo s inštanciou count1)

Odvodený funkčný blok má rovnaké vlastnosti ako základné funkčné bloky. Vytvára ho užívateľ v programovacích jazykoch FBD, LD, IL a ST za účelom sprehľadnenia programu ako aj minimalizácie pamäťových nárokov pre program.

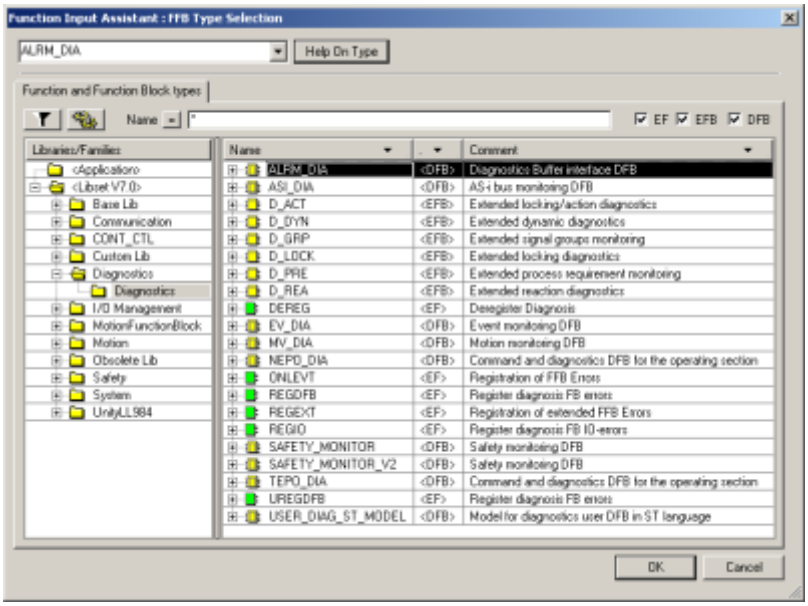


Obr. 5 Ukážka jednoduchého vlastného počítadla (DFB) s názvom pocitadlo a inštanciou poc1. Vstupy pripocitaj a odpocitaj slúžia na navyšovanie a znižovanie hodnoty výstupu cislo.

Procedúra je funkcia s viacerými výstupmi. Podobne ako funkcie, procedúry nemajú vnútorný stav. Nevracajú hodnotu, ale na rozdiel od funkcie podporujú aj premenné VAR_IN_OUT typu, t.j. premenná môže byť programom čítaná aj zapisovaná. Graficky sa nijak neodlišuje od základnej funkcie. Nakoľko sa jedná o rozšírenie IEC 61131-3, tak je ich potrebné povoliť v nastaveniach.

Diagnostická knižnica Unity Pro

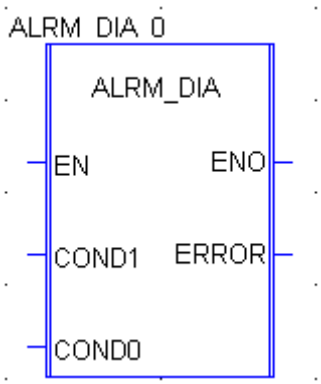
Diagnostické funkcie v Unity Pro sa nachádzajú v knižnici (Obr. 6) a je ich možné použiť v rôznych programovacích jazykoch. V článku si opíšeme len niektoré z nich a uvedieme príklad ich použitia.



Obr. 6 Zoznam diagnostických funkčných blokov v Unity Pro

ALRM_DIA

Ako prvý príklad si uvedieme diagnostický funkčný blok ALRM_DIA (Obr. 7). Slúži na ukladanie chýb (porúch) do diagnostického buffra.



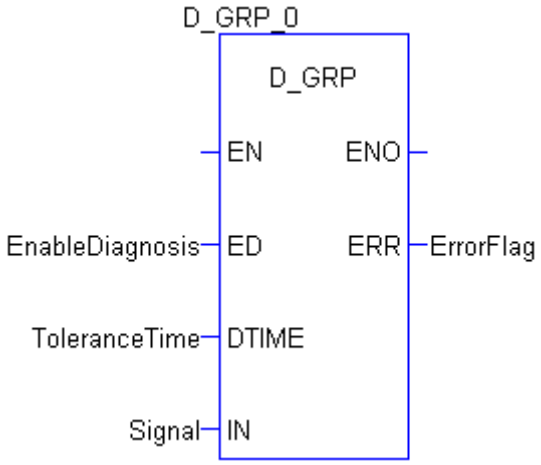
Obr. 7 Príklad inštancie funkčného bloku ALRM_DIA

Prepnutie vstupu COND1 na 0 alebo prepnutie vstupu COND0 na 1 ukladá chybu (poruchu) do diagnostického buffra. Ak sú oba vstupy prepnuté, tak sa uloží jediná chyba, ktorá sa zruší po tom, čo sa oba vstupy vrátia na správnu hodnotu. Vstupný parameter COND1 je typu EBOOL. Číslo „1“ v názve charakterizuje jeho funkciu, vstupný bit je v stave 1 a monitoruje sa jeho zmena. V prípade zmeny z 1 na 0, ALRM_DIA generuje chybové hlásenie do diagnostického buffra. Ak sa hodnota COND1 zmení z hodnoty 0 na 1, tak ALRM_DIA nehlási žiadnu novú chybu.

Vstupný parameter COND0 je takisto typu EBOOL a má podobnú funkciu ako COND1, s tým rozdielom, že jeho prednastavenou hodnotou je 0 a monitoruje sa zmena na 1. Reakcia na zmenu je rovnaká ako pri COND1. V prípade, že je vygenerovaná chyba a hodnota COND1 sa zmení z 1 na 0, ALRM_DIA nehlási žiadnu novú chybu. Výstupný parameter ERROR je typu EBOOL. Má hodnotu 0, keď nie je prítomná žiadna chyba. V prípade chyby sa bit zmení na 1.

D_GRP

Funkčný blok D_GRP sa používa na monitorovanie skupiny signálov.

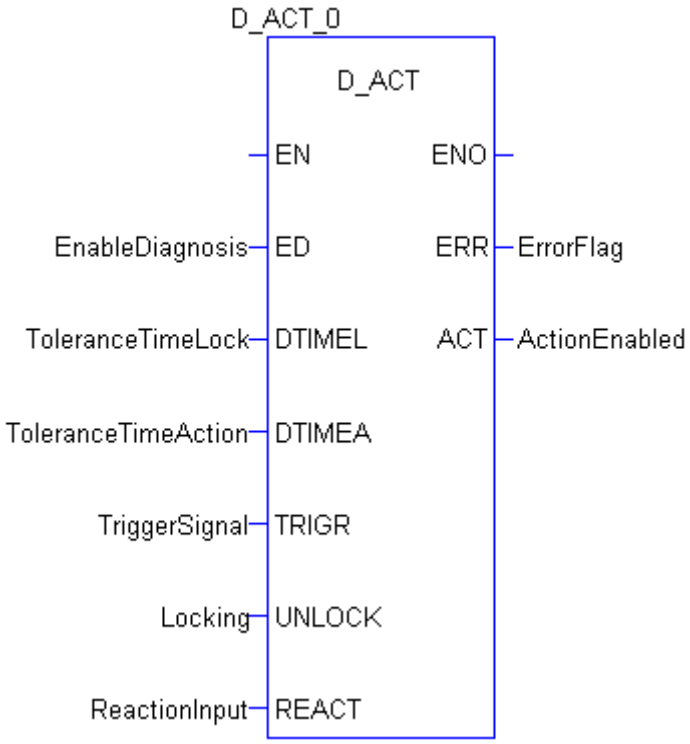


Obr. 8 Príklad inštancie funkčného bloku D_GRP

Monitorovanie sa vykonáva cyklicky. Diagnostika a cyklus sa aktivujú vstupným signáli ED. Pokiaľ je stav logická 1 na vstupe IN dlhšie ako definovaný čas DTIME, tak sa vygeneruje porucha. Premenné sú typu BOOL, okrem premennej DTIME, ktorá je pochopiteľne typu TIME.

D_ACT

Ako posledný príklad si predstavíme niečo zložitejšie, funkčný blok D_ACT (Obr. 9), ktorý poskytuje kombináciu „zamykacích a akčných diagnostík“ (locking and action diagnostics).



Obr. 9 Príklad inštancie funkčného bloku D_ACT

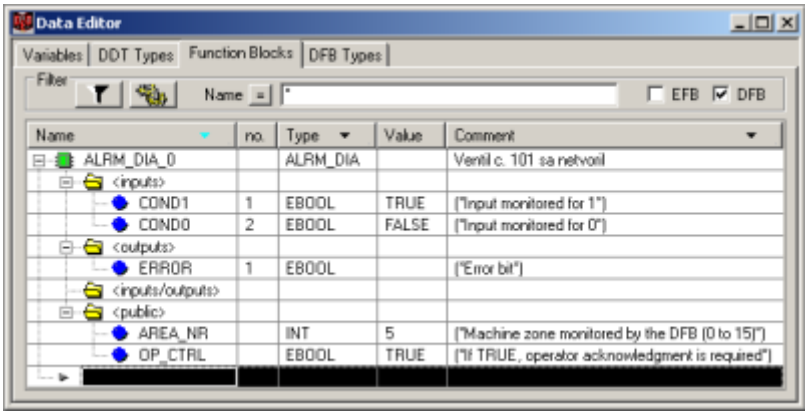
Zamykacia diagnostika sa aktivuje v prípade aktívneho vstupu TRIGR. V určitých sieťach však signál z TRIGR vstupu nemusí začínať vykonávanie akcie, ale kombinovane so zámkami v procese. Toto môže spôsobiť, že výstup ACT dostane signál

s určitým oneskorením. Na toto slúži zamykacia diagnostika, ktorá kontroluje, či UNLOCK je zapnutý v čase DTIMEL, keď je zapnutý TRIGR. Ak áno, tak výstup ACT dostane signál a môže sa vykonávať činnosť. TRIGR musí byť vtedy aktívny počas celého vykonávania akcie. V prípade, že sa UNLOCK neobjaví za stanovený čas, objaví sa porucha, ACT samozrejme nie je aktívna a výstup ERR sa nastaví. Okrem toho sa chyba zaznamená do diagnostického buffra.

Akčná diagnostika je inicializovaná po aktivovaní výstupu ACT. Toto inicializuje určitú preddefinovanú činnosť procesu, napr. výstup môže byť nastavený pre prepnutie motora do pohotovostného režimu. Táto činnosť musí spustiť reakciu, ktorá sa väčšinou objavuje s oneskorením. Ak sa reakcia neuskutoční v danom čase DTIMEA, nastane chyba, ktorá sa nastaví na ERR a zapíše do diagnostického buffra. Monitorovanie sa uskutočňuje cyklicky, ktoré sa nastavuje pomocou signálu ED. Okrem premenných DTIMEL A DTIMEA, ktoré sú typu TIME, sú všetky premenné typu BOOL.

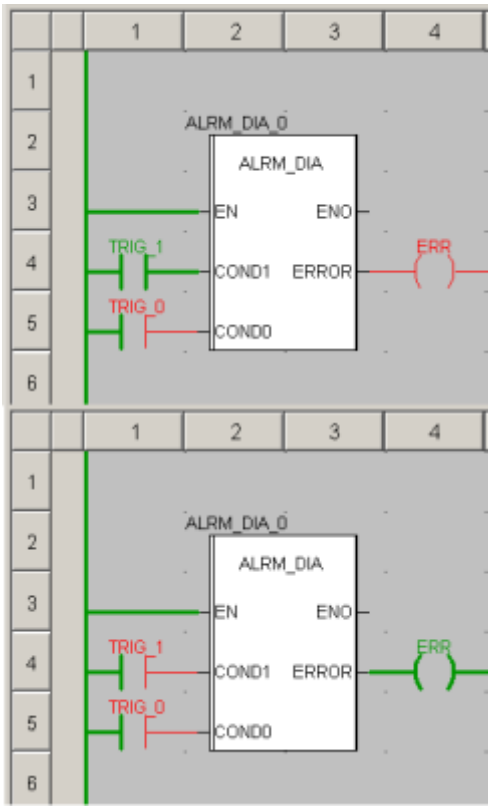
Praktický príklad

Uvedieme si príklad použitia funkčného bloku ALRM_DIA. Konfigurácia funkčného bloku je na obr. 10. Inštancia ALRM_DIA_0 má prednastavené inicializačné hodnoty vstupov COND0 a COND1 pre prípad, ak by k nim neboli pripojené vstupy v programe. Pri nezadefinovanom stave by sa mohlo nechcene vygenerovať alarmové hlásenie. Parameter AREA_NR definuje skupinu zariadení do ktorého môžeme funkčný blok zaradiť (resp. vygenerované hlásenia). Tento parameter sa využíva aj vo vizualizačnom nástroji VijeoDesigner napr. pri filtrácii porúch. Posledný parameter OP_CTRL slúži na zapnutie a vypnutie nutnosti potvrdzovania vzniknutého alarmu. V tomto príklade si uvedieme ukážku s nutnosťou potvrdenia operátorom.



Obr. 10 Konfigurácia funkčného bloku ALRM_DIA

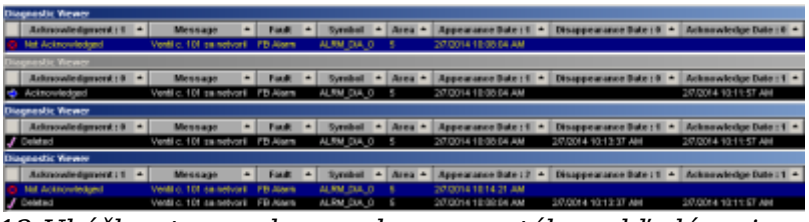
Po vytvorení sekcie (programu) v jazyku LD bol vytvorený nasledovný ukážkový program (Obr. 11). Bity TRIG_0 a TRIG_1 slúžili na simuláciu zmeny hodnoty stavov na vstupe diagnostického funkčného bloku. Signál ERR sa použil len na indikáciu a extra zápis do pomocnej premennej. V horne j časti je ukážka kedy nevznikol alarm (TRIG_0 = FALSE, TRIG_1 = TRUE). V spodnej časti došlo k zmene stavu na vstupe COND1 z TRUE na FALSE, preto sa vygeneroval alarm (Obr. 12) a na výstupe je ERROR = TRUE.



Obr. 11 Ukážky online sledovania programu v Unity Pro v jazyku LD.

Na obrázku nižšie (Obr. 12) je niekoľko stavov alarmov. Prvý príklad zobrazuje vznik alarmu s hlásením „Ventil c. 101 sa neotvoril“, ktorý bol zapísaný do diagnostického buffra funkčným blokom (FB Alarm) s inštanciou ALRM_DIAG_0 (Symbol). Okrem toho bola zaznamenaná časová značka jeho vzniku. Všimnime si, že alarmové hlásenie je komentár inštancie funkčného bloku (Obr. 10), ktoré je spoločné aj pre stav COND0 a COND1.

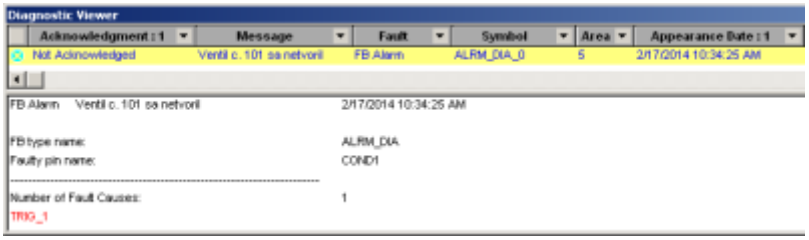
Druhý príklad zobrazuje rovnaký alarm, ktorý bol potvrdený operátorom (viď. stav Acknowledged). Potvrdenie bolo vykonané s využitím nástroja Diagnostic Viewer priamo v Unity Pro. Tretí príklad zobrazuje alarm po potvrdení a zrušení stavu, ktorý ho vyvolal, t.j. TRIG_1 sa zmenil zo stavu FALSE na TRUE. Posledný, štvrtý príklad zobrazuje opätovné vzniknutie daného alarmu. Diagnostic Viewer naďalej zobrazuje okrem aktívneho alarmu aj prednedávnom potvrdený a zaniknutý alarm.



Obr. 12 Ukážky stavov alarmov bez zapnutého vyhľadávania príčiny

Ako posledný príklad si uvedieme vygenerovanie alarmu pri nastavenej lokálnej diagnostike. V takomto prípade sa zobrazuje aj príčina poruchy (obr. 13) pri vyznačení niektorého z alarmov. Ako vidno zobrazuje sa početnosť (Number of Fault Causes) a aj vstup, ktorý alarm vygeneroval (tag / premenná TRIG_1). Môže to byť užitočné ak sa používajú obe vstupné podmienky diagnostickej funkcie. V prípade, že tag TRIG_1 by bol zapisovaný v inej sekcie, tak pre zistenie pôvodu poruchy je potrebné zapnúť

globálnu diagnostiku.



Obr. 13 Ukážky stavov alarmov so zapnutou lokálnou diagnostikou

Záver

V praxi sa veľmi často používa diagnostika pomocou nastavovania hodnôt bitov (samostatných bitov, bitov slov alebo dvojslov), ktoré sú vo vizualizácii vyhodnotené ako alarm resp. priame vyhodnotenie prekročenia limít analógového signálu. Takéto riešenie generuje alarmové hlásenie vo vizualizácii bez toho, aby sa to zapisovalo do diagnostického buffra. Článok sa zameriaval na stručné oboznámenia čitateľa s možnosťami diagnostiky PLC pomocou funkčných blokov v Unity Pro, ktoré umožňuje zapísať správy do diagnostického buffra. Tieto správy je možné čítať priamo v PLC, HMI, PC a pod. ako ich aj potvrdzovať a preto predstavuje efektívnejší spôsob diagnostiky.

Poďakovanie

Tento príspevok vznikol vďaka podpore Nadácie Tatra banky – E-Talent z Nadácie Tatra banky č. 2013et030.

Literatúra

1. http://www.mikroe.com/old/books/plcbook/app_c/app_c.htm
2. http://www.schneider-electric.com

Spoluautorom článku je Ladislav Körösi.