

**Technická Univerzita v Košiciach**  
**Katedra elektrotechniky a mechatroniky**

**LADENIE STRIEDAVÉHO POHONU**  
**SINAMICS S120**

2019/2020

Tomáš MERVERA

Andrea VITKOVSKÁ

Sebastián BUJŇÁK

Filip MÝTNÍK

Ladislav HRIC

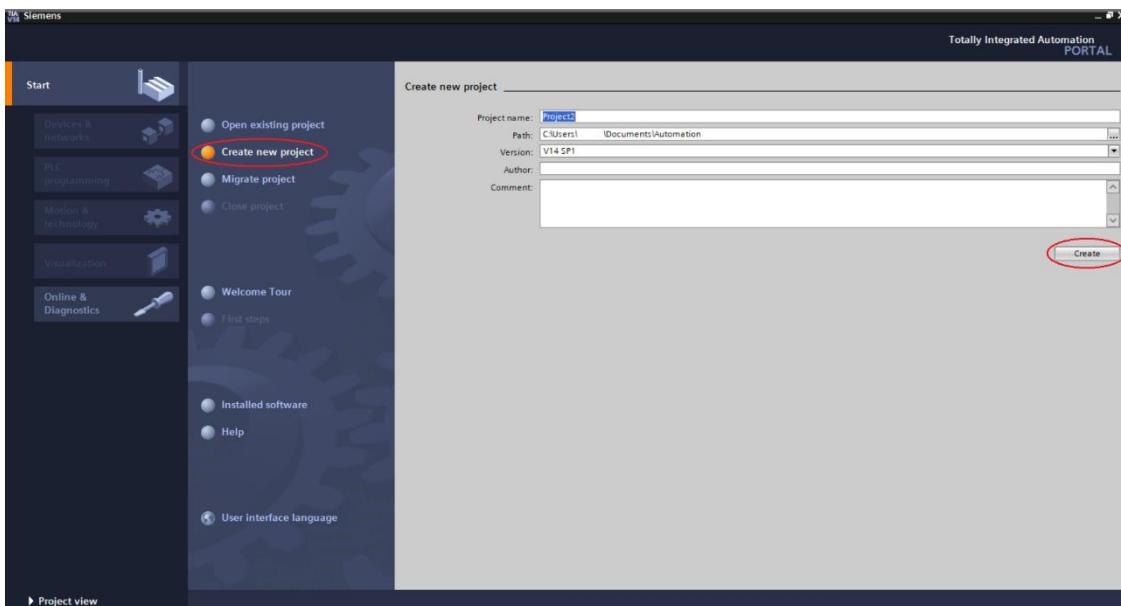
## **Obsah**

1. Vytvorenie nového projektu.....	3
2. Hardvérová konfigurácia .....	4
3. Tvorba programu.....	18
3.1. Nastavenie meniča .....	18
3.2. Tvorba programu na PLC .....	21
4. Tvorba užívateľského rozhrania .....	31

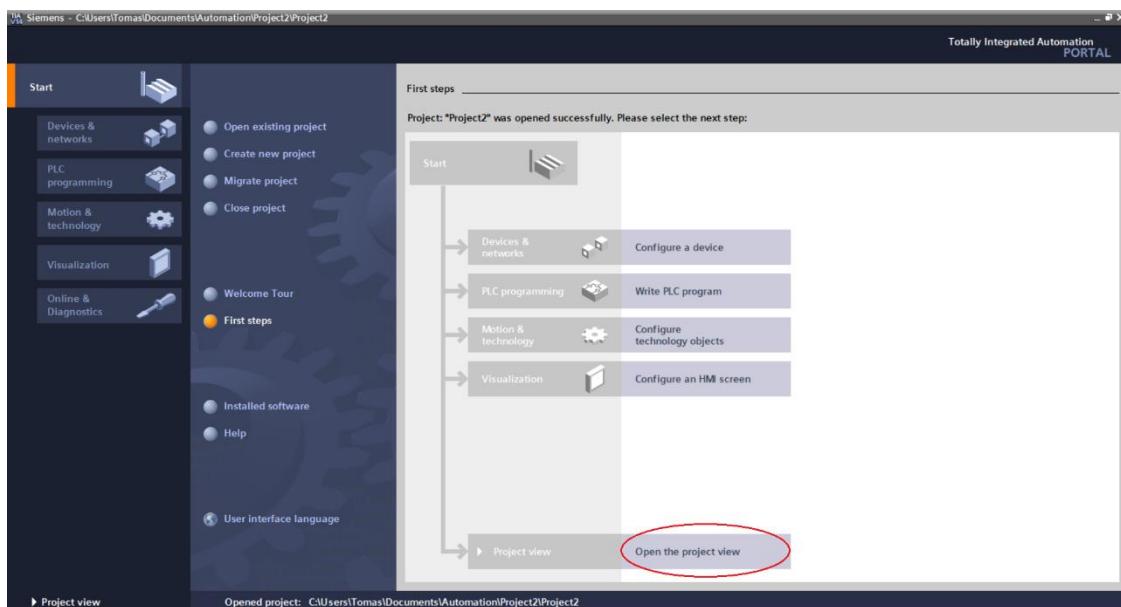
V tomto manuáli sa budeme venovať tomu, ako prijímať a odosielat dátu do meniča prostredníctvom PLC, ktoré bude obsluhované HMI zariadením. Súčasťou manuálu však nie je konfigurácia meniča. Rozoberieme si konfiguráciu jednotlivých zariadení na strane PLC, vytvorenie programu pre PLC a vytváranie užívateľského rozhrania. V tejto práci je použitý program TIA Portal od spoločnosti Siemens.

## 1. Vytvorenie nového projektu

1. Ako prvé je potrebné vytvoriť nový projekt. Po otvorení programu TIA Portal klikneme na možnosť „Create new project“. Zvolíme si názov projektu „Project name“ a cestu „Path“, kde si uložíme daný projekt.

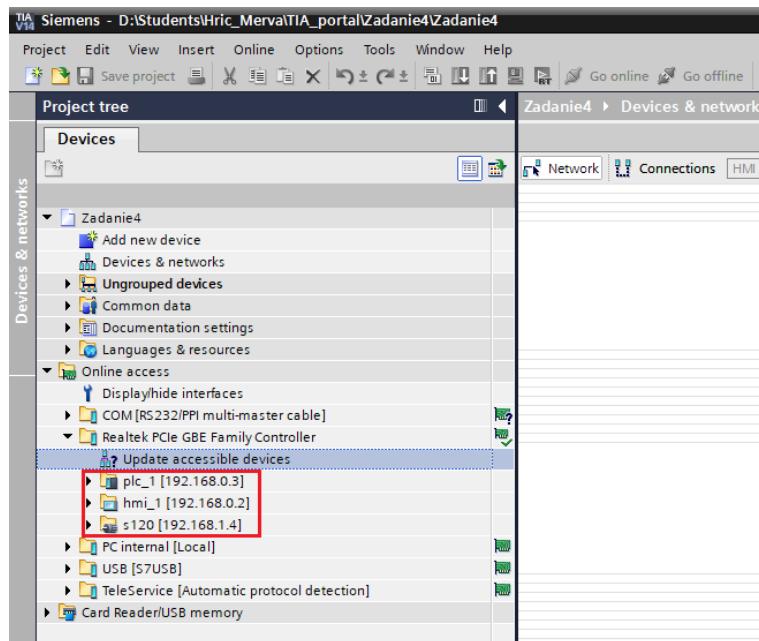


2. Následným kliknutím na tlačidlo „Create“ sa vytvorí náš projekt a zobrazí sa nasledujúca obrazovka. Pre prístup k projektu klikneme na tlačidlo „Open the project view“.



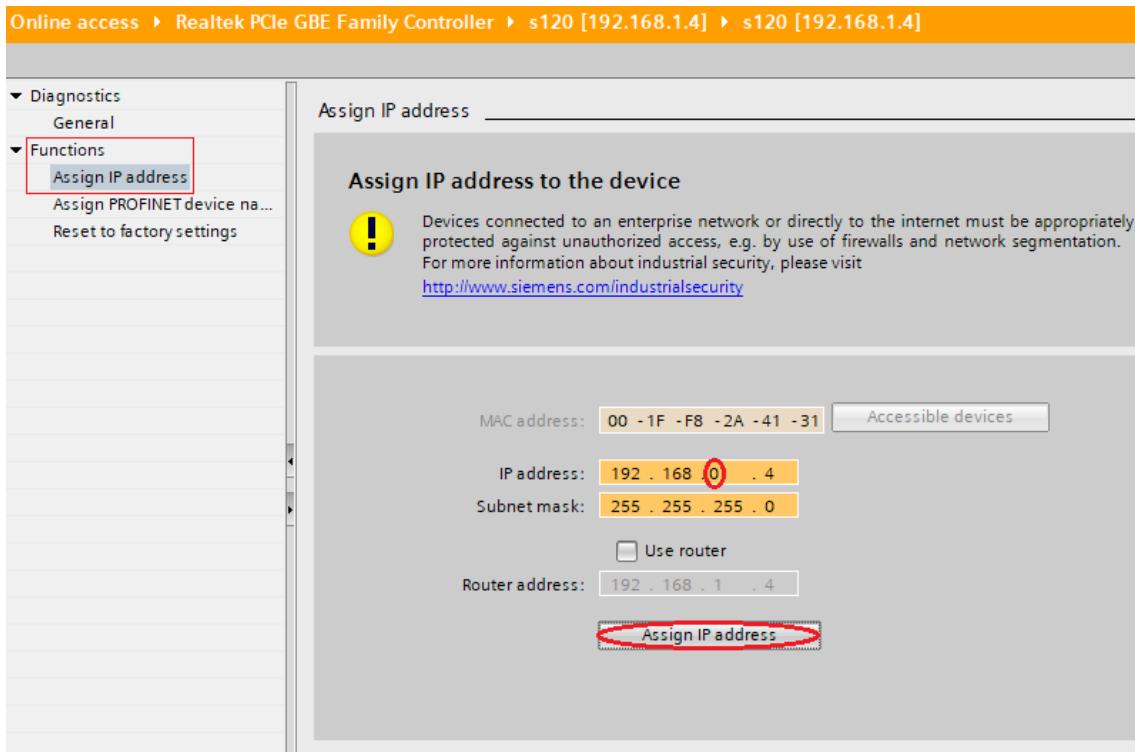
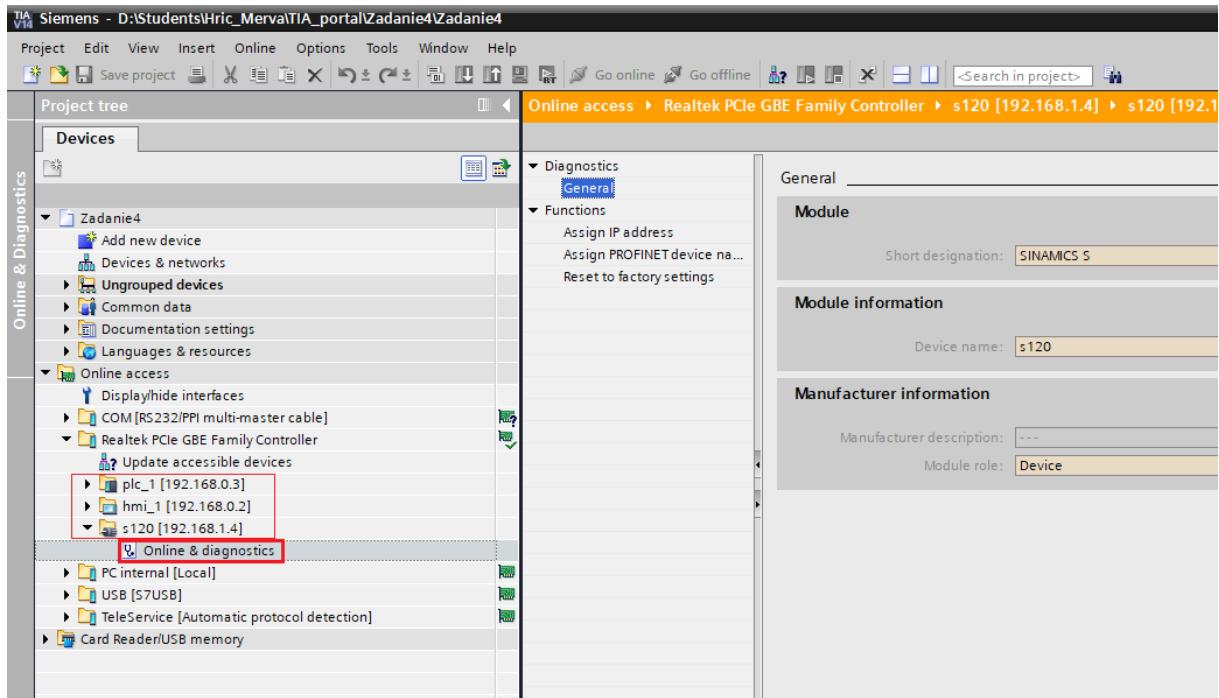
## 2. Hardvérová konfigurácia

1. Po úspešnom vytvorení nového projektu je potrebné zabezpečiť softvérové prepojenie jednotlivých zariadení (v našom prípade to sú PLC, HMI a menič) do spoločnej siete. V okne „Project tree“ rozklikneme priečinok „Online access“, nájdeme názov našej sieťovej karty („Realtek PCIe GBE Family Controller“) a po rozkliknutí priečinky sieťovej karty zvolíme dvojkliknutím „Update accessible devices“. Táto funkcia nám preskenuje sieť a nájdzie všetky zariadenia, ktoré sú fyzicky pripojené k nášmu počítaču (nezávisle od toho, v akej sieti sa nachádzajú).



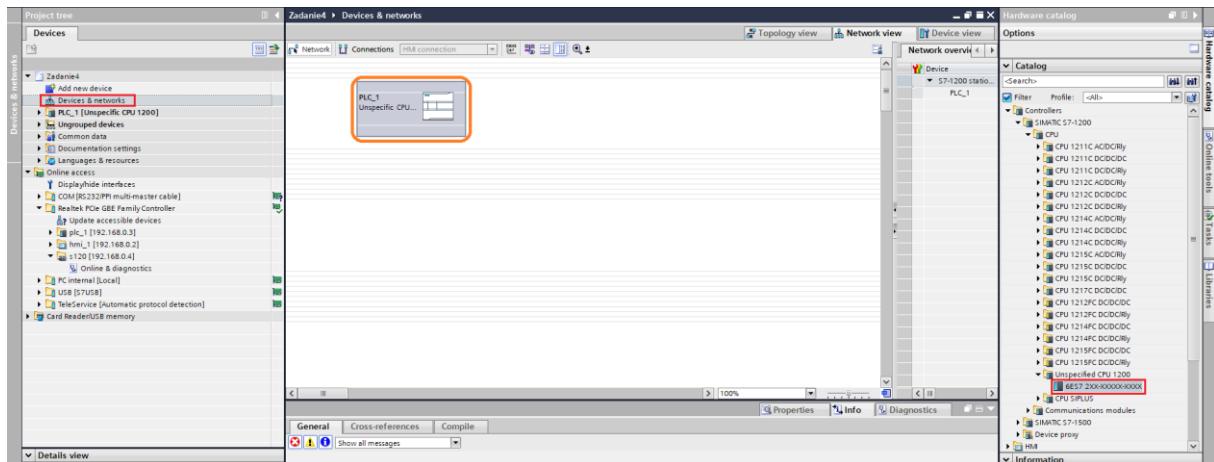
V našom prípade je k nášmu počítaču pripojené PLC s názvom („PROFINET device name“) *plc\_1*, používateľské rozhranie *hmi\_1* a menič *s120*. V prípade, že funkcia „Update accessible devices“ nenájde nejaké zariadenie, tak dané zariadenie nie je pripojené fyzicky k sieti. Ak sa zariadenie ide použiť prvý krát, tak sa síce zobrazí, ale namiesto IP adresy bude MAC adresa zariadenia a názov bude „Accessible device“.

2. Jednotlivým nájdeným zariadeniam je kvôli komunikácii potrebné priradiť správne IP adresy a názvy („PROFINET device name“). Spoločná sieť je definovaná prvými troma číslami IP adresy, ktoré musia byť pre všetky zariadenia jednej siete rovnaké. Ak má niektoré zo zariadení IP adresu nastavenú na inú sieť (IP adresa siete sa líši), je nutné ju prepísat. Štvrté číslo IP adresy musí byť pre každé zariadenie jedinečné. V našom prípade sa odlišuje IP adresa meniča *s120*, teda je potrebné ju zmeniť. V priečinke „Online access“ rozklikneme „Realtek PCIe GBE Family Controller“ a zvolíme menič „*s120*“. IP adresu meniča zmeníme voľbou „Online & diagnostics“ -> „Functions“ -> „Assign IP address“.

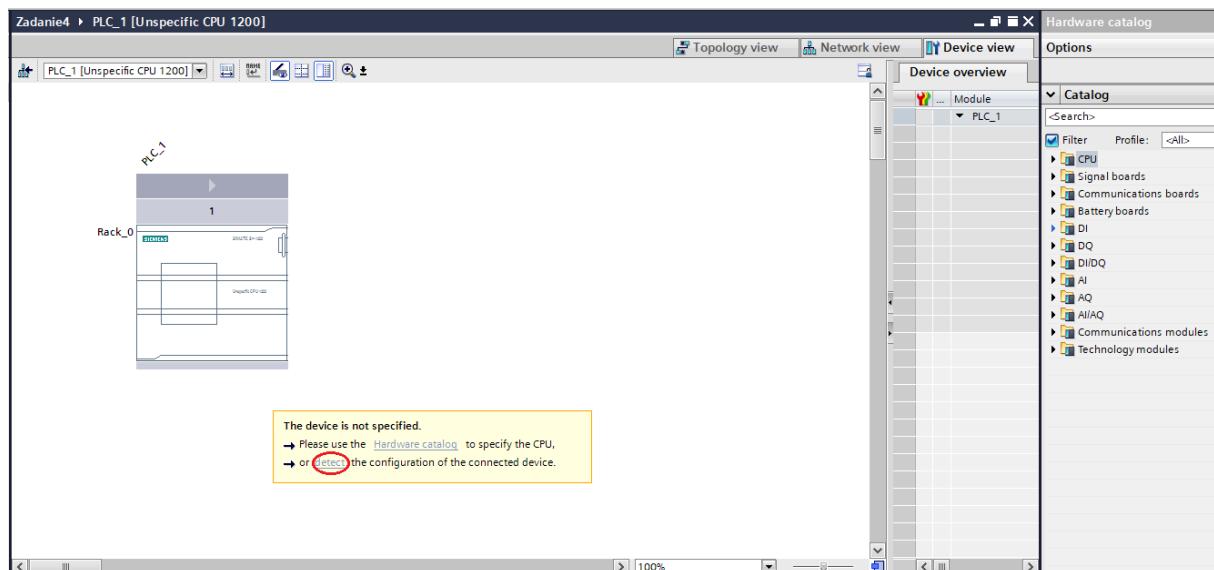


Voľbu novej IP adresy potvrdíme tlačidlom „Assign IP address“. V prípade, že je potrebné zmeniť aj meno zariadenie, vyberieme možnosť „Functions“ -> „Assign PROFINET device name“.

3. Ďalším krokom je samotné softvérové prepojenie jednotlivých zariadení. Vykonáva sa to v okne „*Devices and networks*“, ktoré nájdeme v okne „*Project tree*“. V okne „*Hardware catalog*“, na pravej strane obrazovky, vyhľadáme konkrétné zariadenia, ktoré chceme pripojiť do siete. Ako prvé sme do siete pridali PLC controller umiestnený pod položkou „*Controllers*“. Po rozkliknutí typu PLC zariadenia možno zvoliť presný typ jeho procesora alebo je možné zvoliť nešpecifikovaný procesor „*Unspecified CPU 1200*“, ako to bolo aj v našom prípade. Vybrané PLC vložíme do „*Devices and networks*“ dvojkliknutím alebo presunutím prvku potiahnutím a následným vložením do okna.

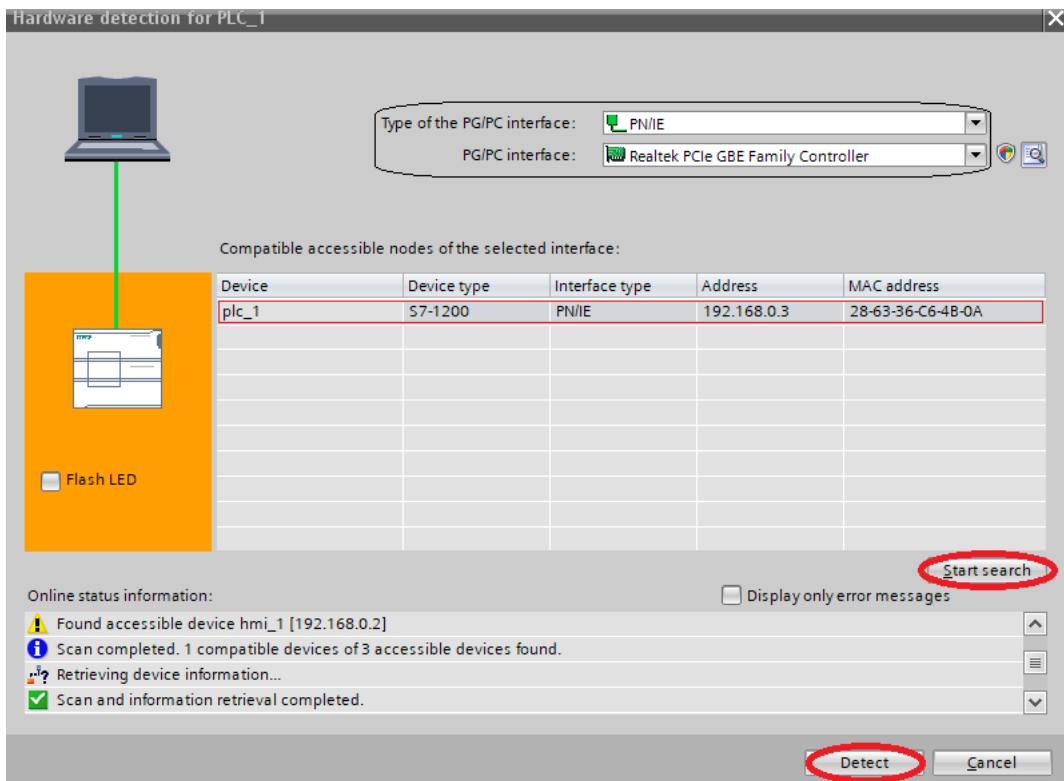


Dvojklikom na blok zvoleného PLC (PLC\_1) sa otvorí okno „*Device view*“, v ktorom môžeme nastavovať vlastnosti PLC. Najprv však musíme zvoliť typ procesora pre dané PLC, nakoľko sme si zvolili „*Unspecified CPU 1200*“ v predchádzajúcom kroku. Bud’ použijeme opäť „*Hardware catalog*“, alebo automatickú detekciu zariadenia prostredníctvom voľby „*detect*“.

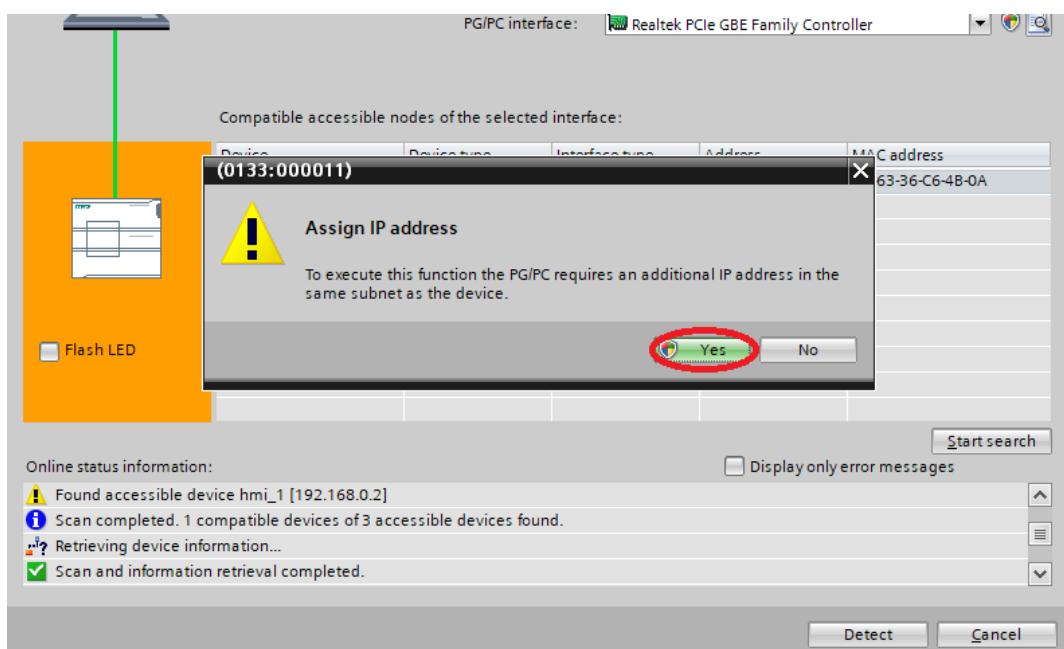


Po zvolení „*detect*“ sa otvorí okno „*Hardware detection*“. Ako prvé je nutné skontrolovať nastavenie komunikačného rozhrania. Pod pojmom „*Type of the PG/PC interface*“ sa myslí typ pripojenia počítača k zariadeniu. Vyberieme si možnosť „*PN/IE*“ (PROFINET/Industrial Ethernet). Ako „*PG/PC interface*“ sa musí vybrať sieťová karta počítača, v našom prípade

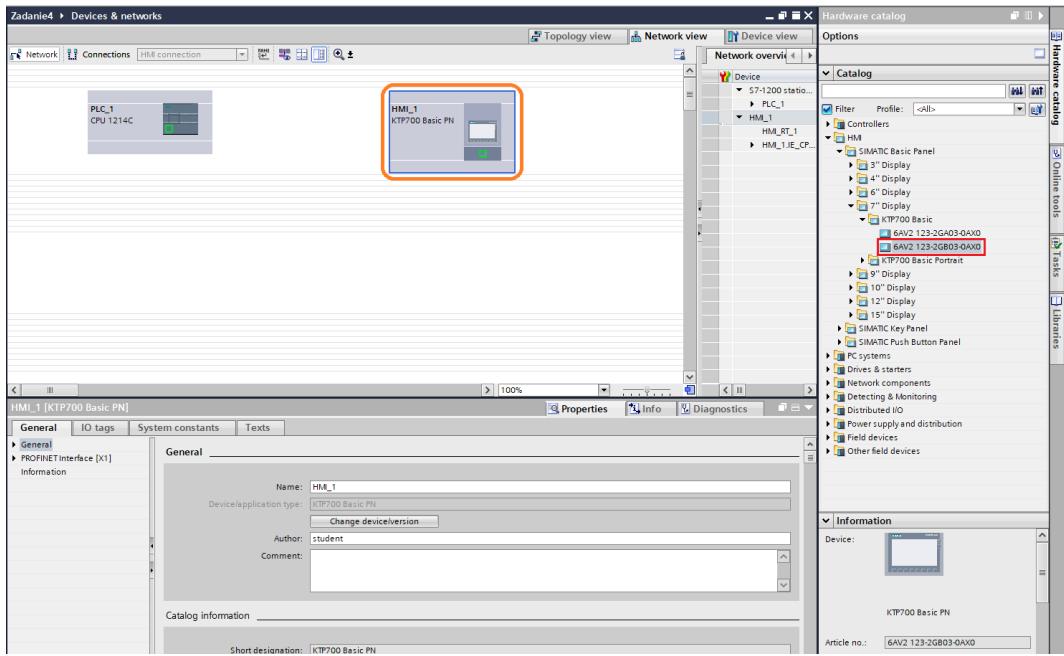
„Realtek PCIe GBE Family Controller“. Po správnom nastavení komunikačného rozhrania dámé vyhľadať PLC zariadenia v našej sieti pomocou tlačidla „Start search“. Po skončení vyhľadávania a nazbierania informácií o PLC, si nájdené PLC vyberieme pomocou tlačidla „Detect“.



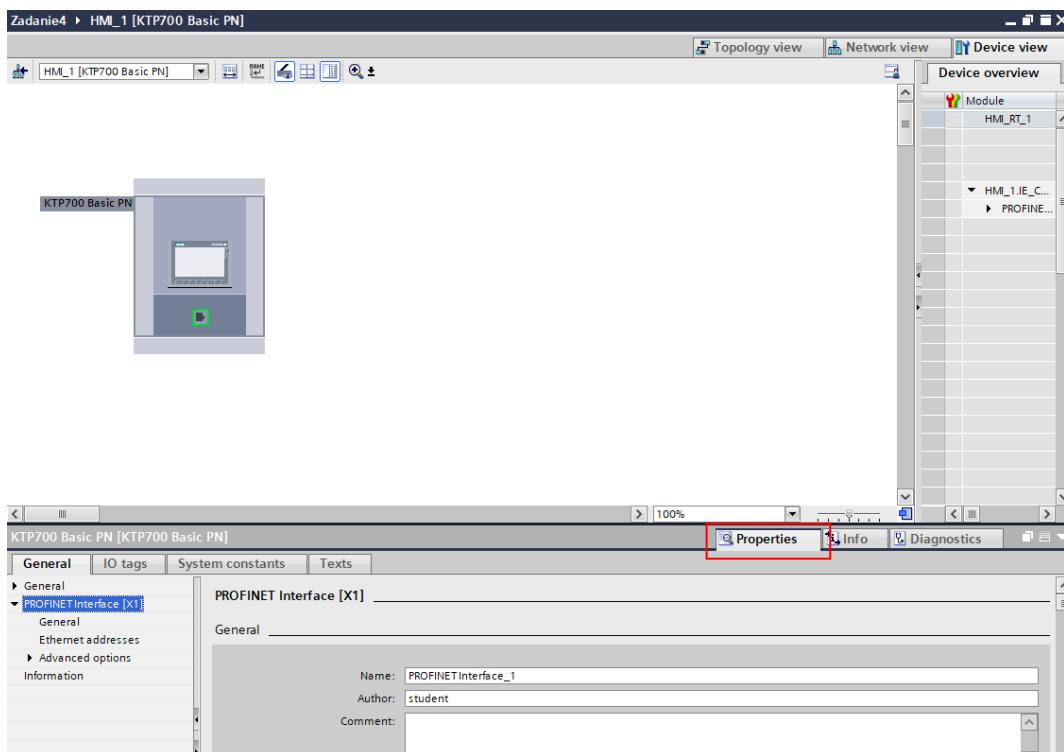
Pokiaľ nemáme pevne zvolenú IP adresu počítača, TIA Portal nám zvolí IP adresu tak, aby sme boli v rovnakej sieti so všetkými zariadeniami. Zvolenú IP adresu je potrebné potvrdiť tlačidlom „Yes“ ako je to znázornené na nasledujúcom obrázku:



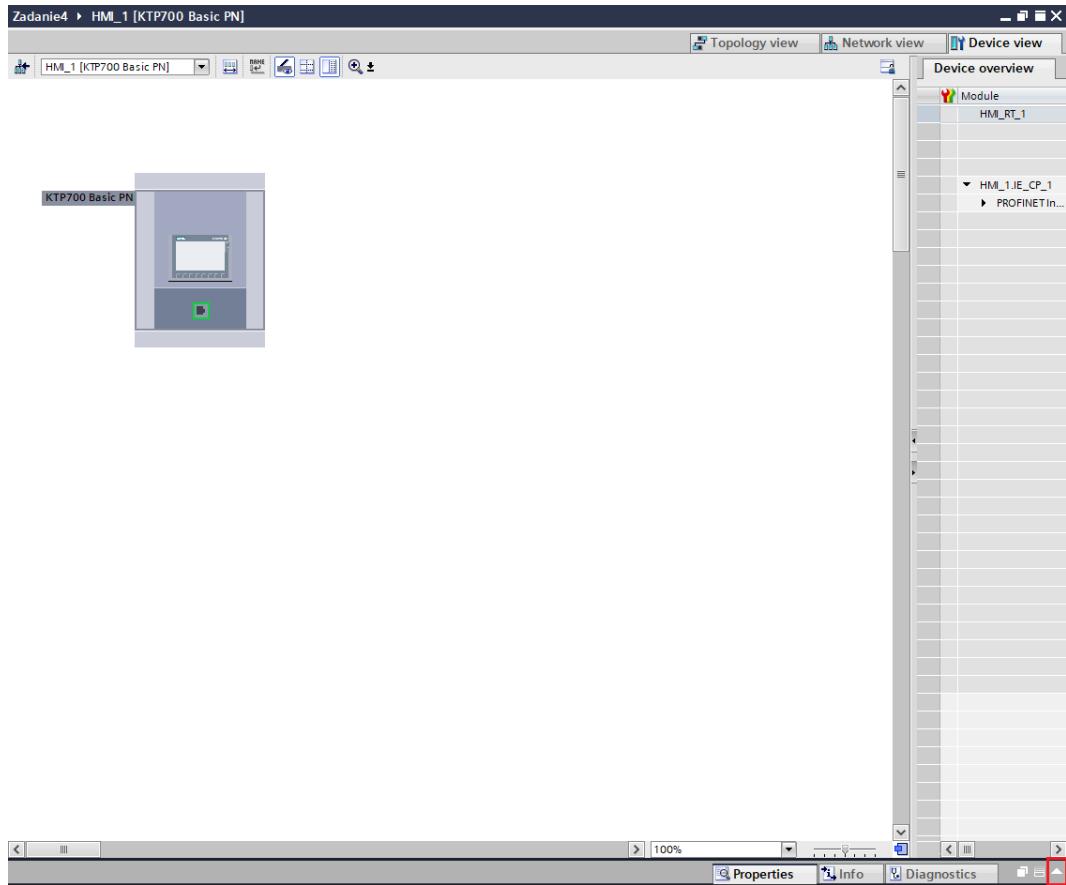
4. Podobne ako v prípade PLC zariadenia vložíme do okna „*Devices and networks*“ aj HMI panel „*hmi\_1*“. Najprv je potrebné zvoliť typ HMI panelu. Urobíme tak v okne „*Hardware catalog*“, kde pod kategóriou „*HMI*“ zvolíme typ odpovedajúci nášmu HMI panelu. („*SIMATIC Basic Panel*“ – „7“ *Display*“ – „*KTP700 Basic*“ – „*6AV2 123-2GB03-0AX0*“).



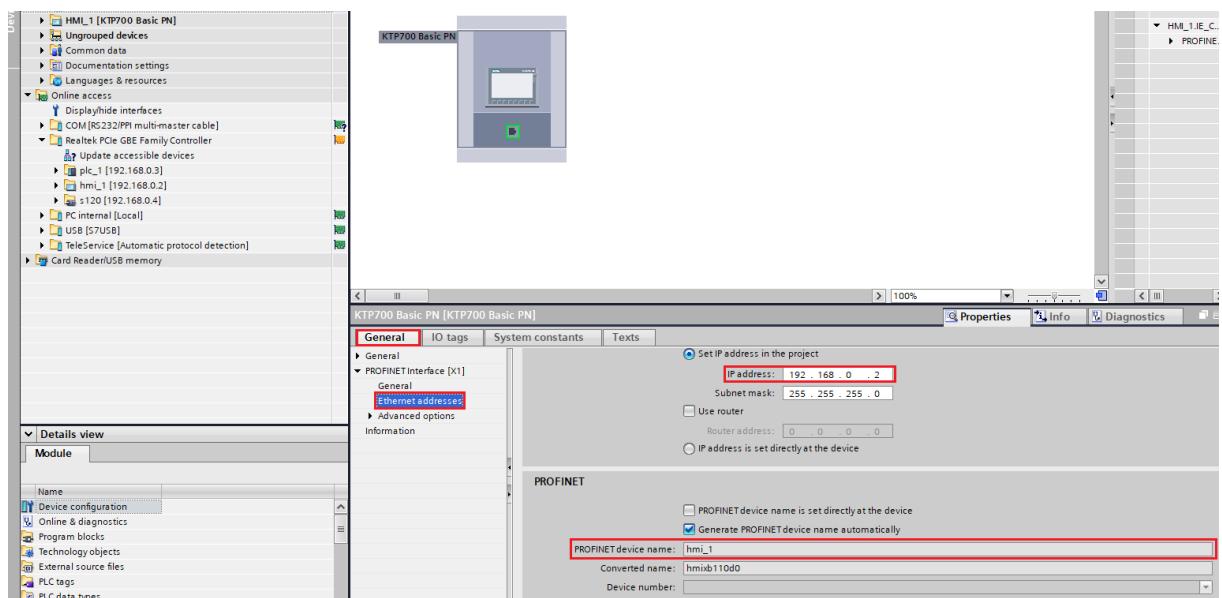
Dvojkliknutím na blok zvoleného HMI panelu (*HMI\_1*) sa otvorí okno „*Properties*“, v ktorom môžeme nastavovať komunikačné vlastnosti.



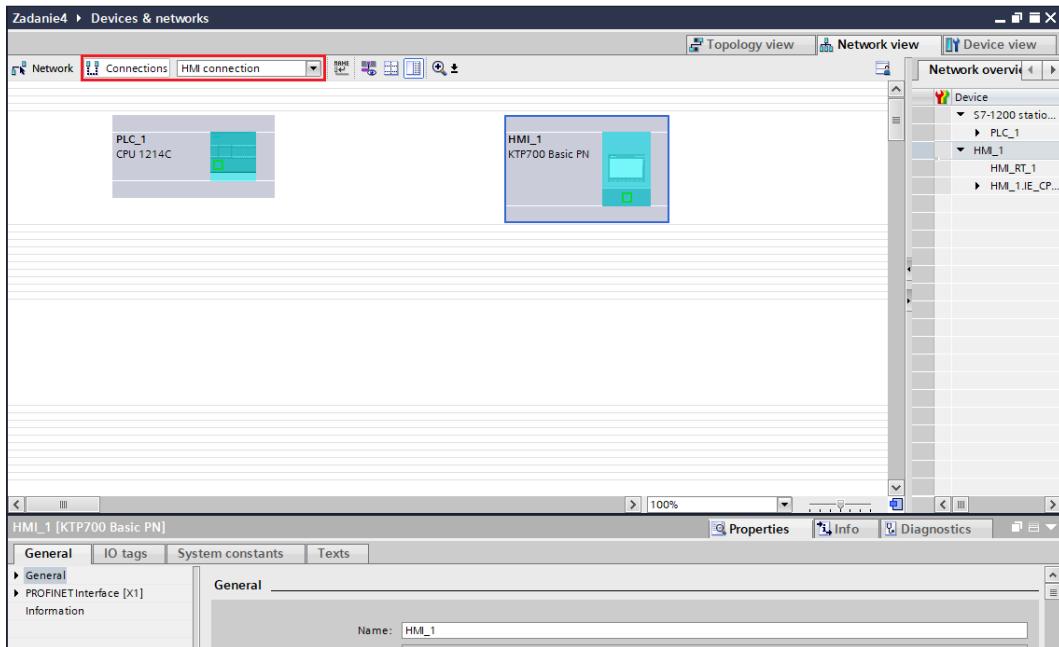
Je možné, že po otvorení karty „Device view“ nebude rozbalené okno „Properties“. V tom prípade ho stačí rozbaliať na lište šípkou v pravom dolnom rohu. Ak by sa okno „Properties“ zobrazilo prázdne, je potrebné kliknúť znova na zariadenie (v zmysle, že TIA Portal potrebuje vedieť, vlastnosti ktorého zariadenia má zobraziť).



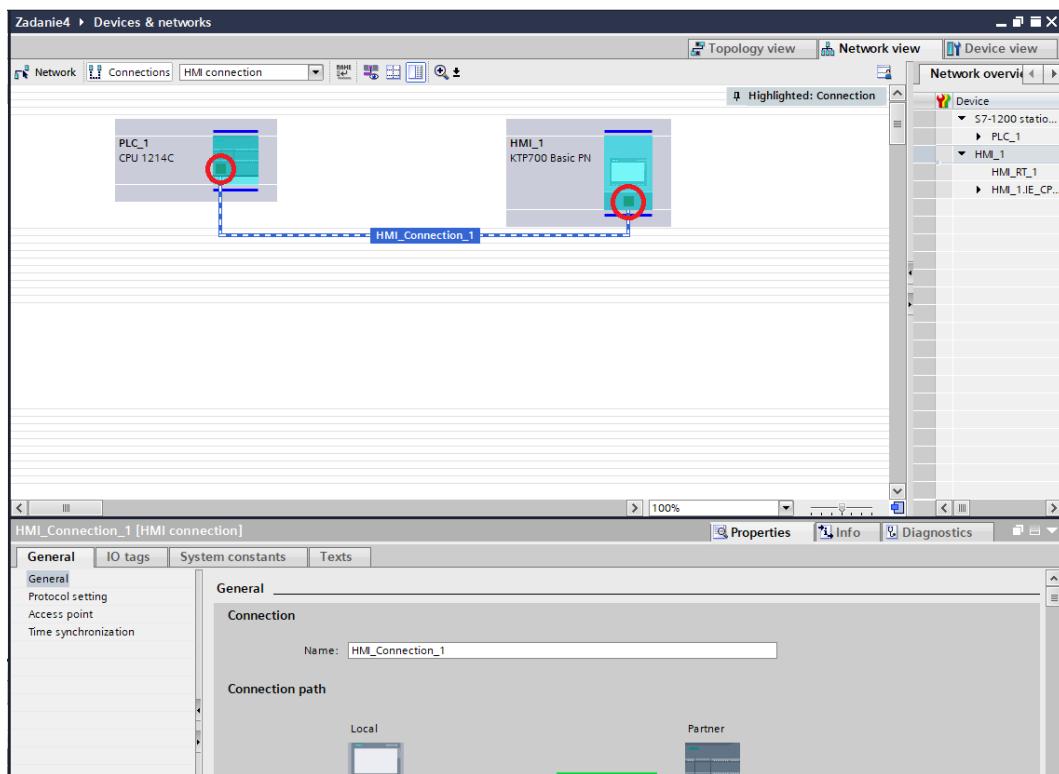
Tu máme možnosť zmeniť napr. IP adresu prvku alebo jeho názov, atď. K jednotlivým vlastnostiam sa dostaneme na karte „General“ už v spomínanom okne „Properties“. IP adresu a meno zariadenia je potrebné nastaviť rovnaké aké má aj v skutočnosti nastavené (vid. Hardvérová konfigurácia bod 2.).



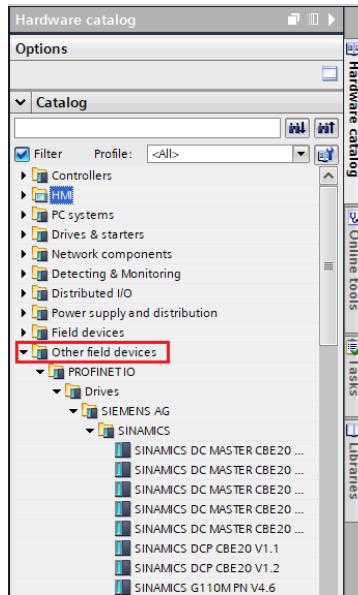
Posledným krokom k vzájomnému prepojeniu PLC a HMI panelu je vytvorenie samotného prepojenia v okne „Devices & networks“ na karte „Network view“. Pre správne prepojenie adres (len v prípade spojenia medzi HMI panelu a PLC) si v záložke „Connections“ zvolíme tiež typ prepojenia „HMI connection“.



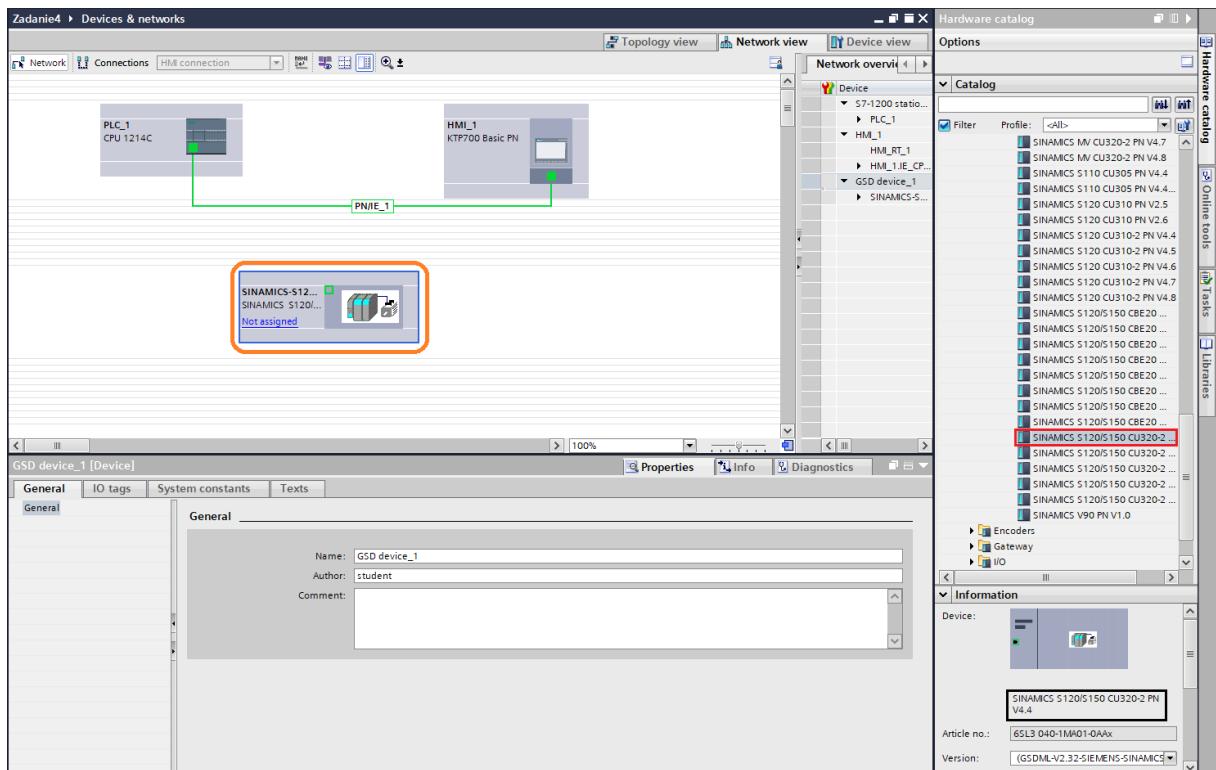
Prepojenie vykonáme kliknutím na zelený štvorček jedného zo zariadení a následným potiahnutím a kliknutím na zelený štvorček druhého zariadenia.



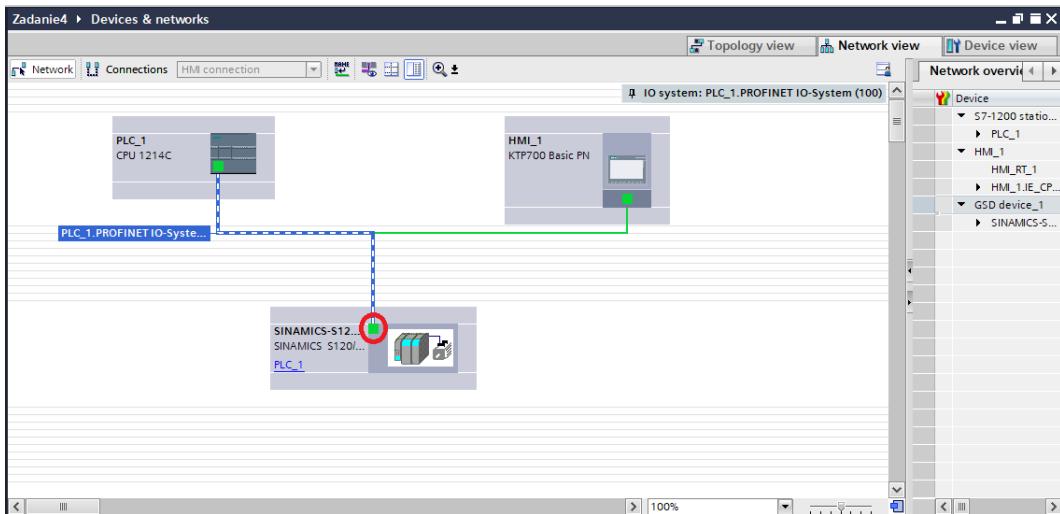
5. Posledným zariadením, ktoré musíme softvérovo pripojiť k PLC a HMI je menič Sinamics S120 „s120“. Meniče komunikujúce prostredníctvom PROFINETu sa nachádzajú v okne „Hardware catalog“ pod priečinkom „Other field devices“ -> „PROFINET IO“ -> „Drivers“ -> „SIEMENS AG“ -> „SINAMICS“.



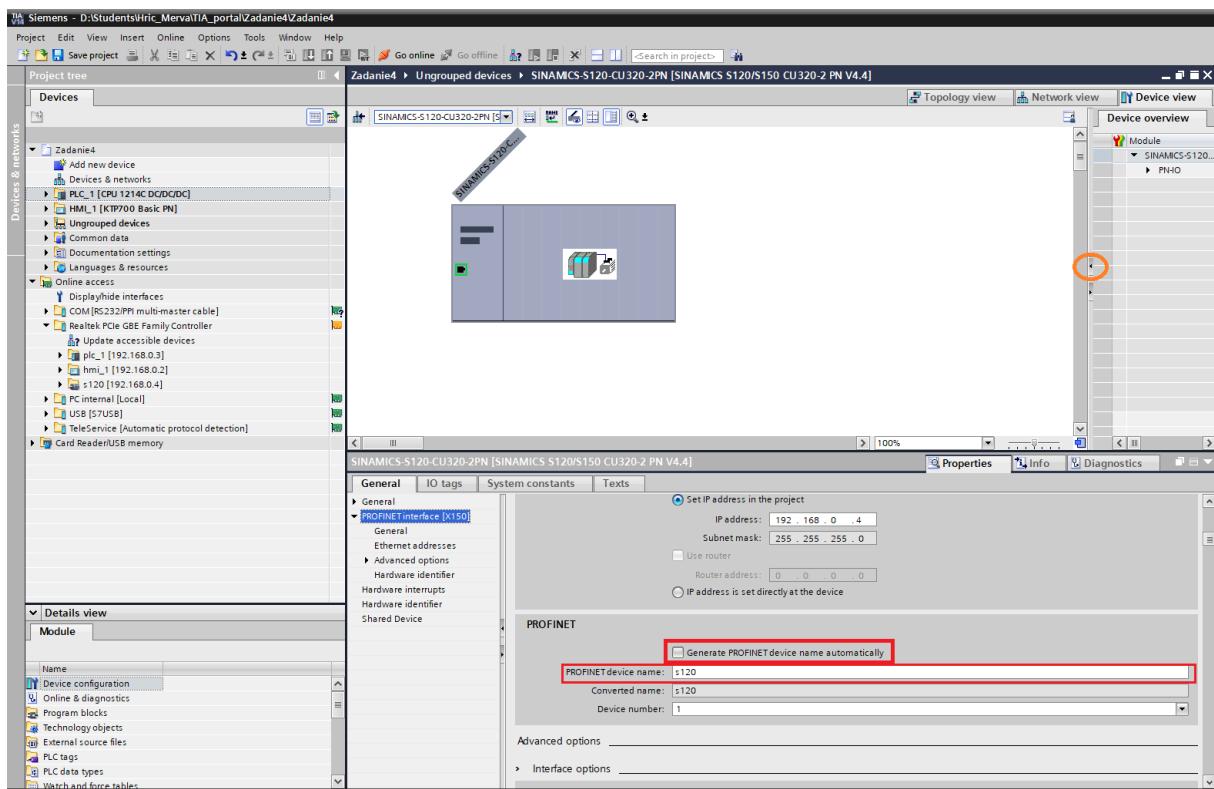
Rovnako ako PLC, HMI panel aj menič („SINAMICS S 120/S 150 CU320-2 PN V4.4“) vložíme do prostredia „Devices & networks“.



Menič následne pripojíme na rovnakú siet, na ktorej je aj PLC a HMI panel, kliknutím na zelený štvorček meniča a potiahnutím a následným kliknutím na už vzniknutý prepoj medzi PLC a HMI panel.

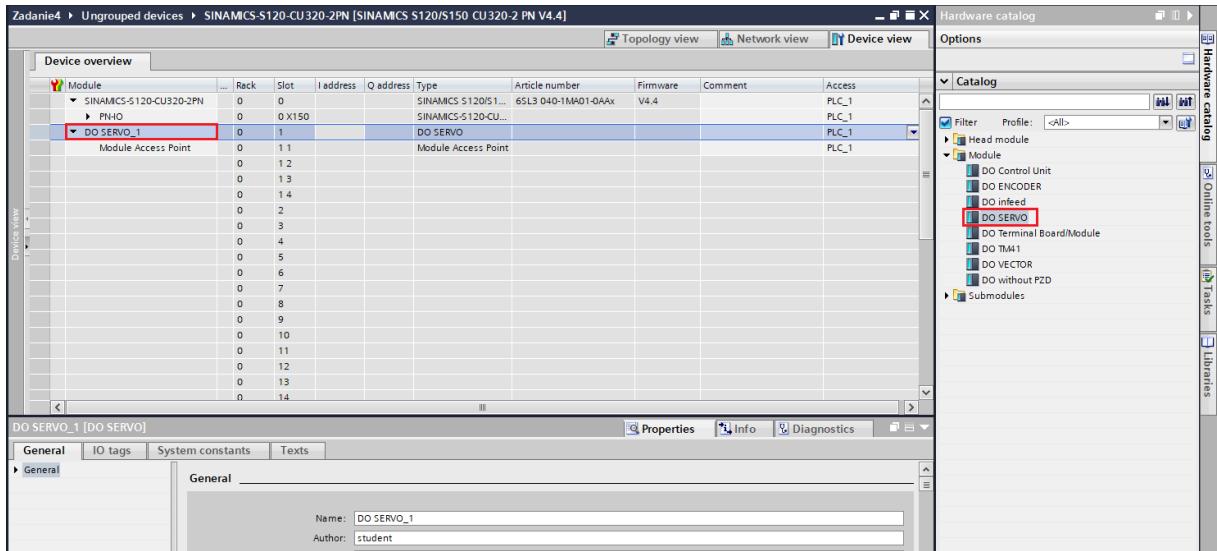


Dvojkliknutím na blok zvoleného meniča (SINAMICS-S120...) sa otvorí okno „Device view“, v ktorom môžeme meniť vlastnosti meniča rovnako ako tomu bolo pri predošlých zariadeniach. Odškrtnutím možnosti „Generate PROFINET device name automatically“ povolíme zmenu PROFINET názvu, ktorý bol priradený meniču automaticky. Názov sa opäť musí zhodovať s názvom, ktorý je nastavený na meniči (vid. Hardvérová konfigurácia bod 2.).

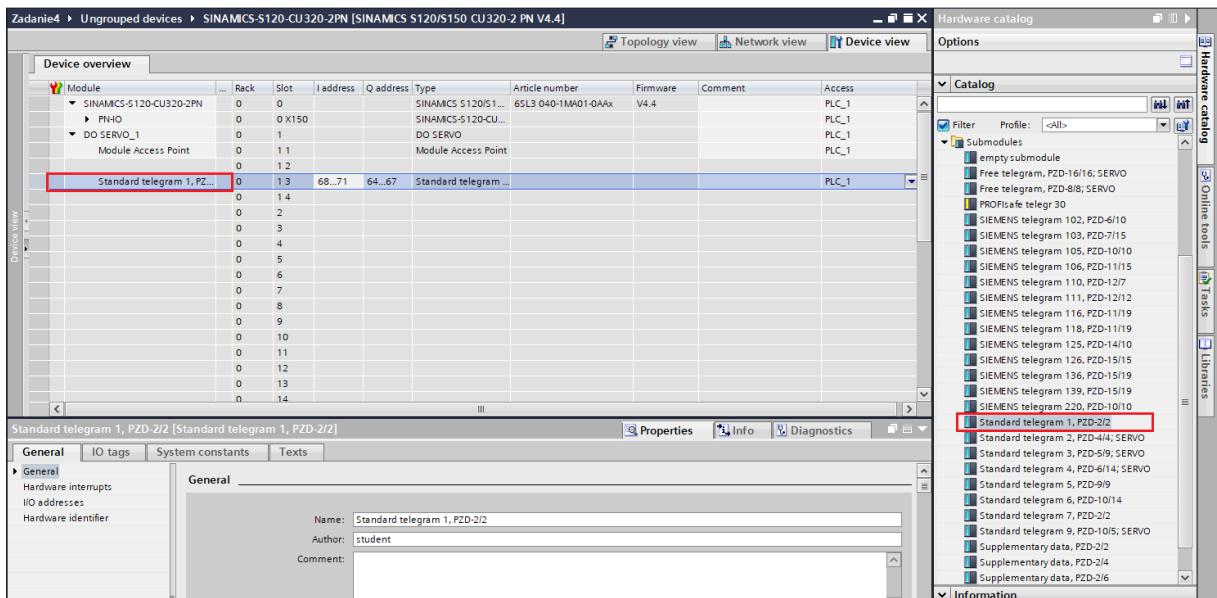


- Konečnou podmienkou spustenia komunikácie medzi PLC a meničom je zadanie telegramu, prostredníctvom ktorého budeme prenášať informácie medzi PLC a meničom. Pod kartou „Device view“ je napravo karta „Device overview“. Ak neviete nájsť „Device overview“, hľadajte malé šipky napravo, pomocou ktorých túto kartu roztvoríte. Po roztvorení karty je potrebné zvoliť „Module“ z hardvérového katalógu. V našom prípade bol menič nastavený

ako „Servo“ a preto aj v TIA Portal-i musíme vybrať rovnaký typ, teda modul „DO SERVO“. Kliknutím a potiahnutím ho vložíme do otvoreného okna „Device overview“.

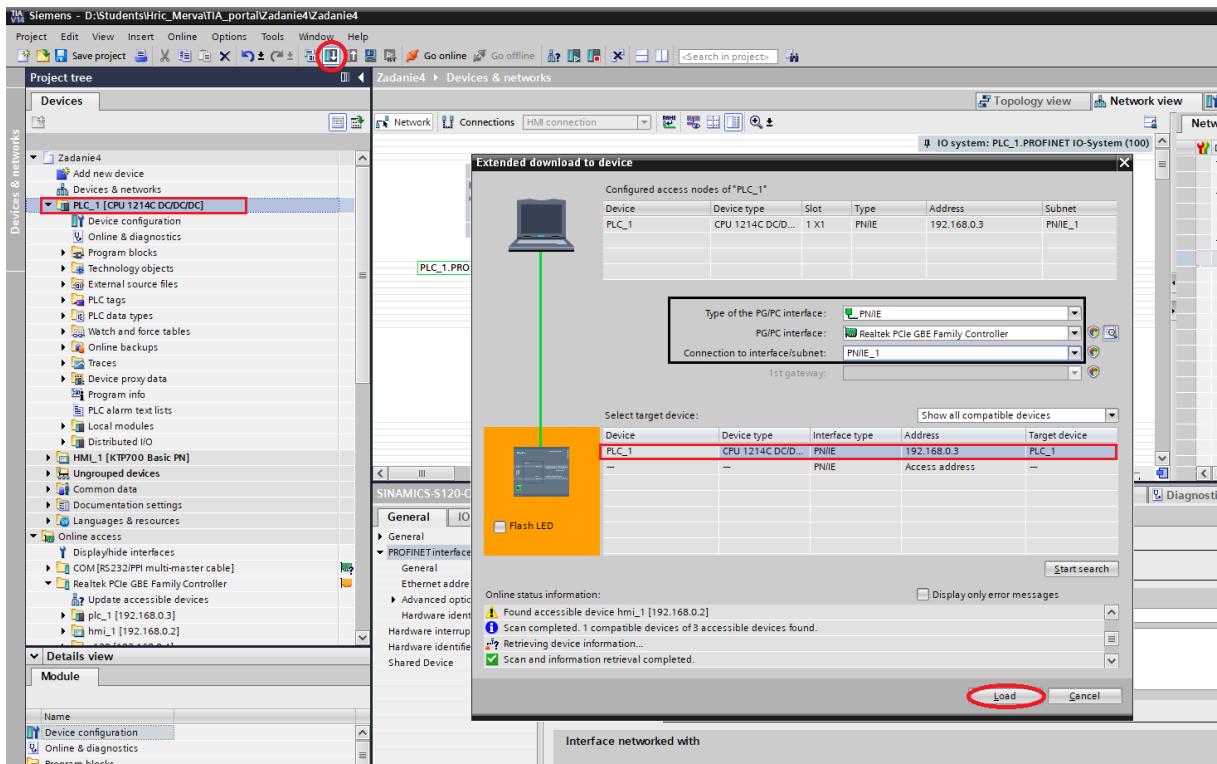


Následne podobne ako modul vložíme pod štruktúru žiadany telegram z priečinka „Submodules“ → „Standard telegram 1, PZD-2/2“. Jedná sa o najjednoduchší rýchlosťny telegram, ktorý sa skladá z 2 riadiacich a z 2 stavových slov.



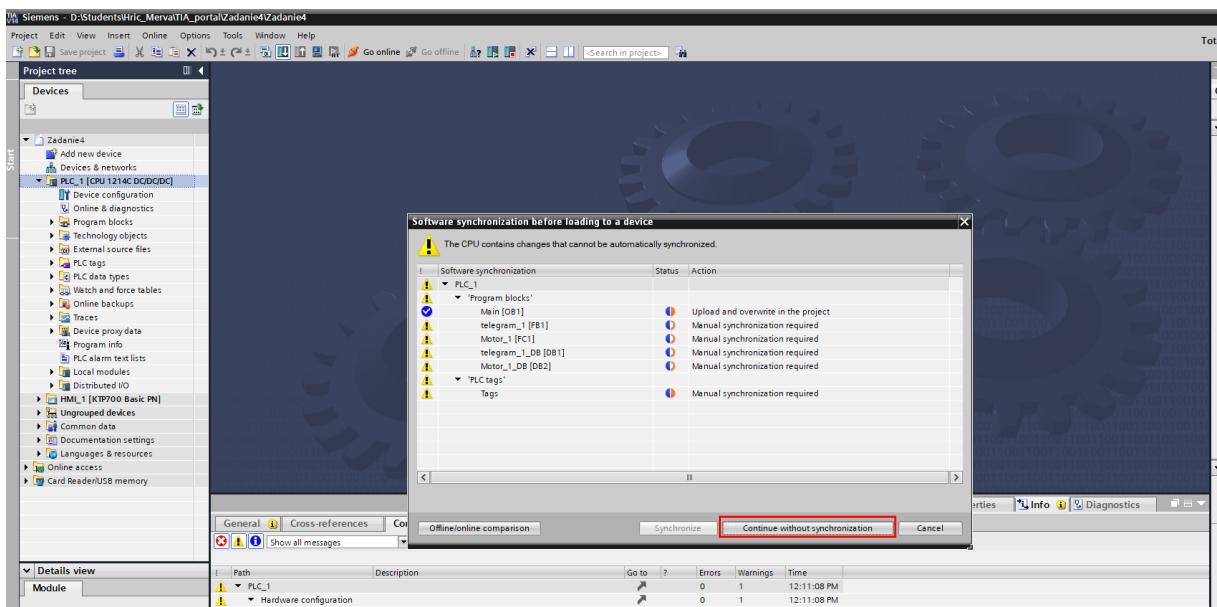
7. Po nakonfigurovaní všetkých zariadení zapojených v sieti je potrebné nahrať hardvérovú konfiguráciu do PLC a HMI panelu. Nahrávanie konfigurácie do HMI panelu si ukážeme v poslednej kapitole, teraz budeme nahrávať konfiguráciu iba do PLC zariadenia. V okne „Project tree“ si kliknutím zvolíme „PLC\_1“ a na paneli nástrojov klikneme na ikonu „Download to device“ . Zobrazí sa nové okno „Extended download to device“, kde skontrolujeme nastavenie komunikačného rozhrania (ako to bolo aj v „Hardvérová konfigurácia“ bod 2). Kliknutím označíme „PLC\_1“ (zariadenie, do ktorého ideme nahrávať

program) a potvrdíme nahrávanie stlačením tlačidla „Load“. V prípade, že v zozname nie je žiadne PLC zariadenie, je potrebné opäť vyhľadať PLC pomocou tlačidla „Start search“.

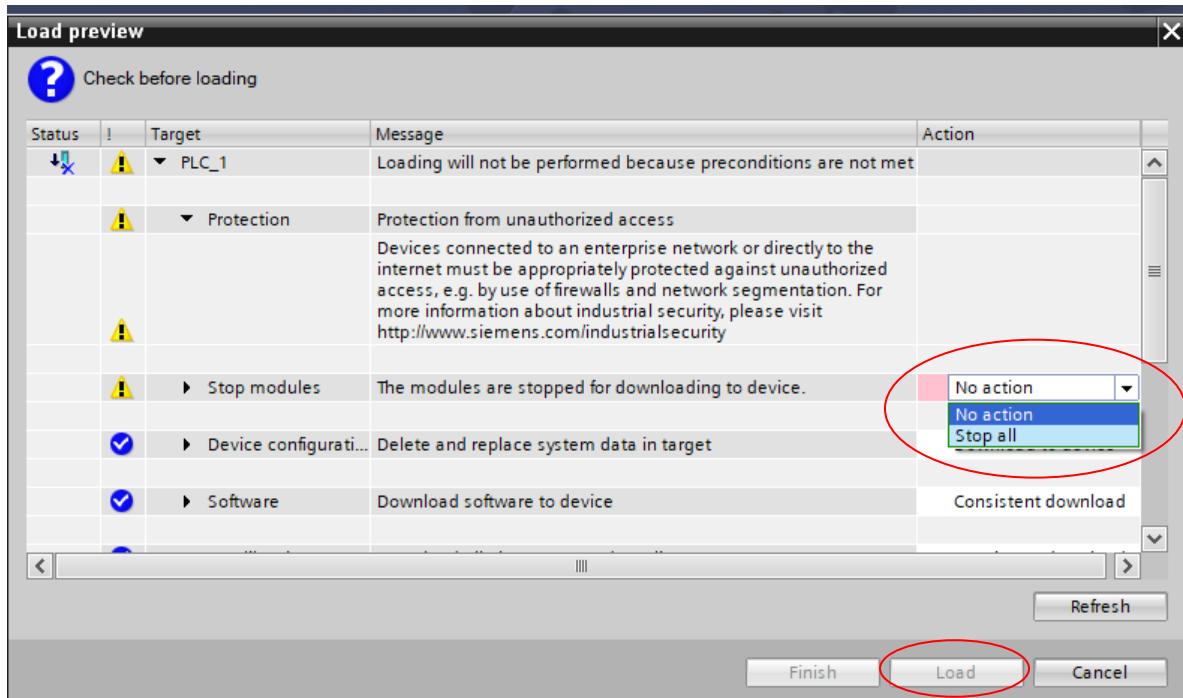


Je možné, že po potvrdení nahrávania sa budú zobrazovať okná, ktoré je potrebné potvrdzovať, aby došlo k nahratiu projektu. Nasledujúce obrázky zobrazujú práve tieto spomínané okná, ktoré sa môžu zobraziť:

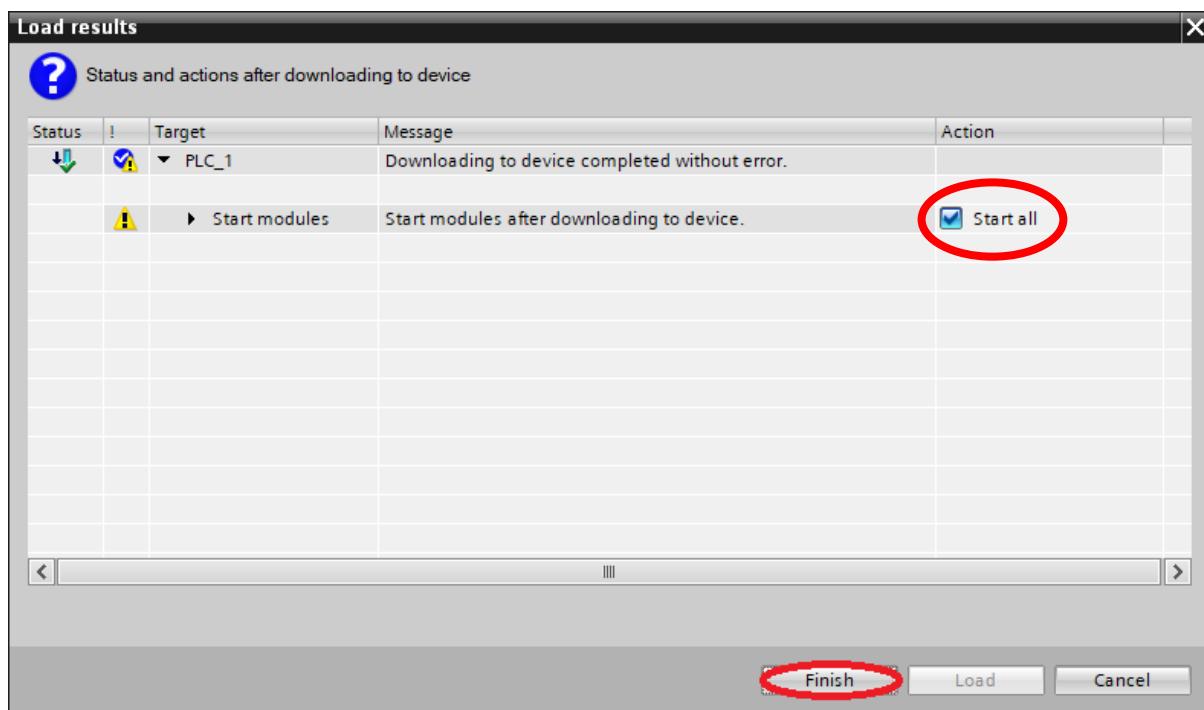
Nasledujúce okno upozorňuje nato, že na PLC zariadení je nahratý nejaký program/konfigurácia a je potrebné ju prepísať. Potvrdíme tlačidlo „Continue without synchronization“



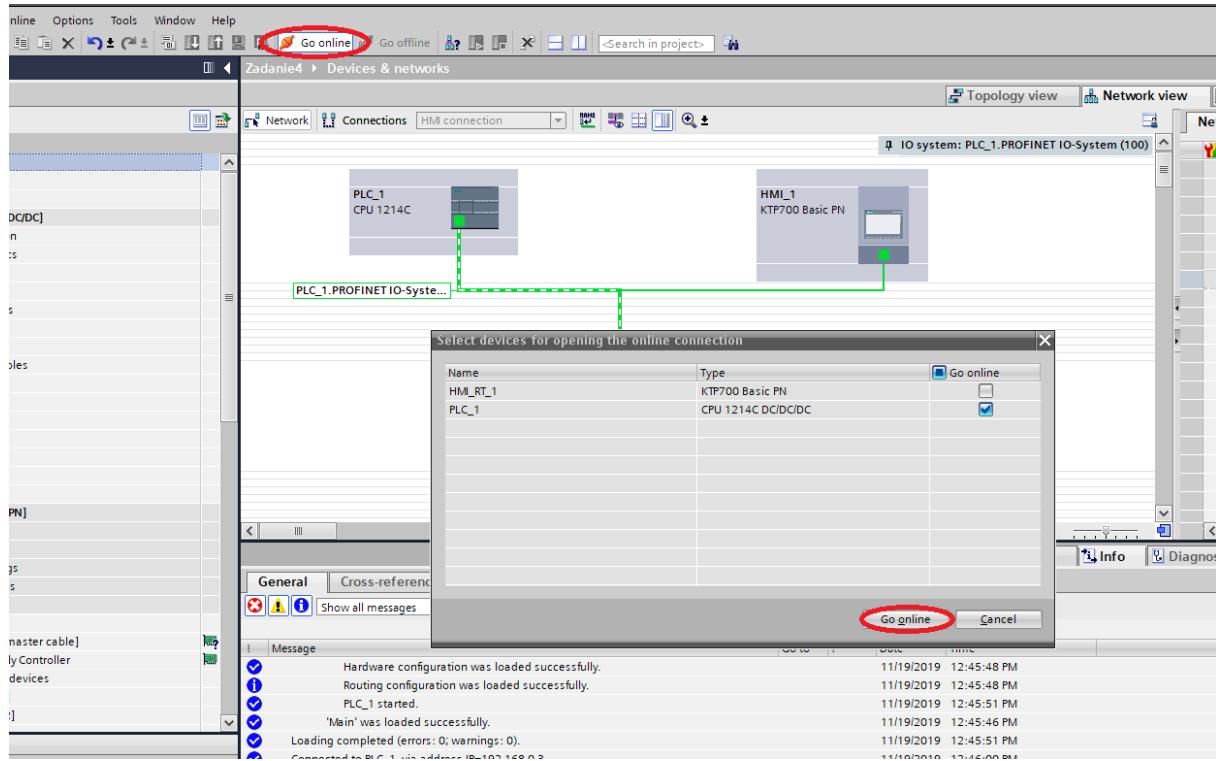
Okno „Load preview“ je posledná kontrola pred samotným nahratím projektu do PLC zariadenia. Pre umožnenie nahratia projektu, je potrebné zastaviť rôzne moduly (ako aj napríklad CPU). V riadku „Stop modules“ a stĺpci „Action“ je potrebné vybrať možnosť „Stop all“. Dokým sa nevyberie táto možnosť, spomínaná bunka bude vyfarbená na červeno a nebude možné potvrdiť nahrávanie pomocou tlačidla „Load“.



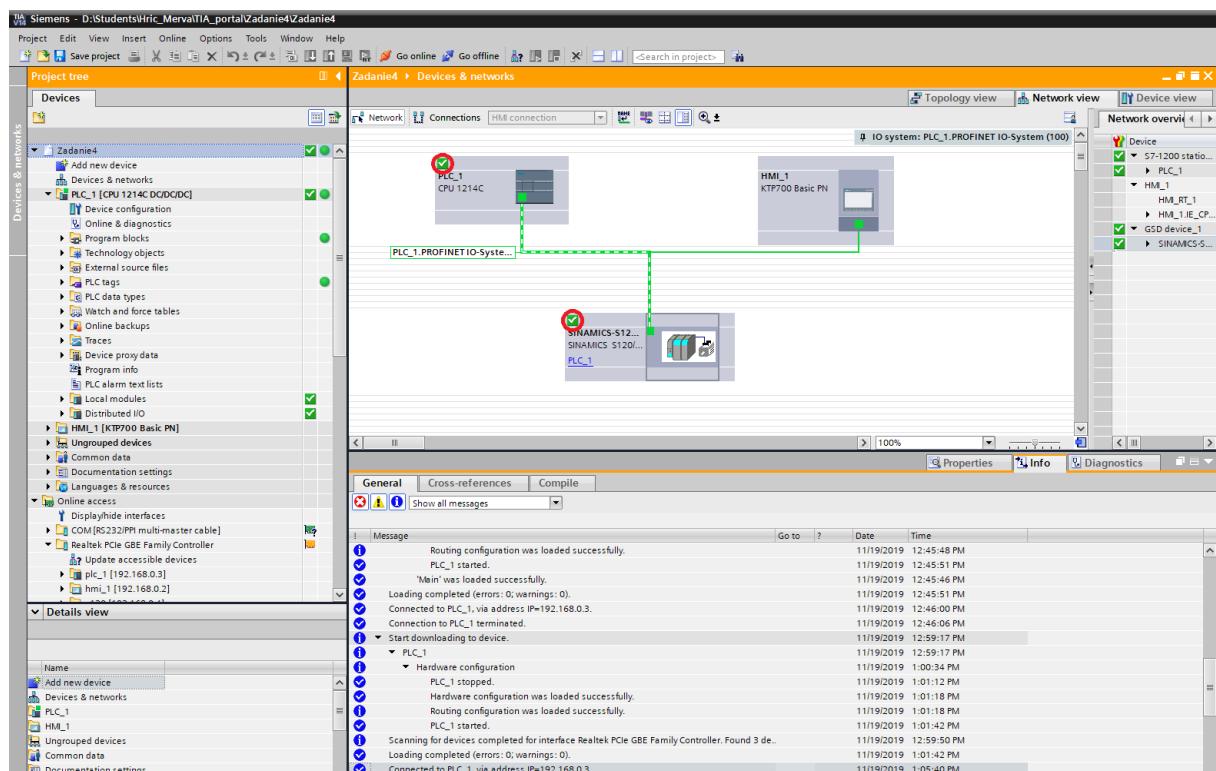
Po úspešnom nahratí projektu je potrebné zapnúť moduly, ktoré sme vyplňovali v predchádzajúcim kroku. Zapnutie modulov vykonáme zaškrtnutím možnosti „Start all“.



8. Správne nastavenie hardvérovej konfigurácie je možné overiť prostredníctvom funkcie „Go online“, ktorou sa vieme pripojiť na PLC a sledovať stav PLC v reálnom čase.



Ak je všetko nastavené správne pri PLC „PLC\_1“ a menič „s120“ sa objaví zelená fajka rovnako ako na nasledujúcom obrázku.



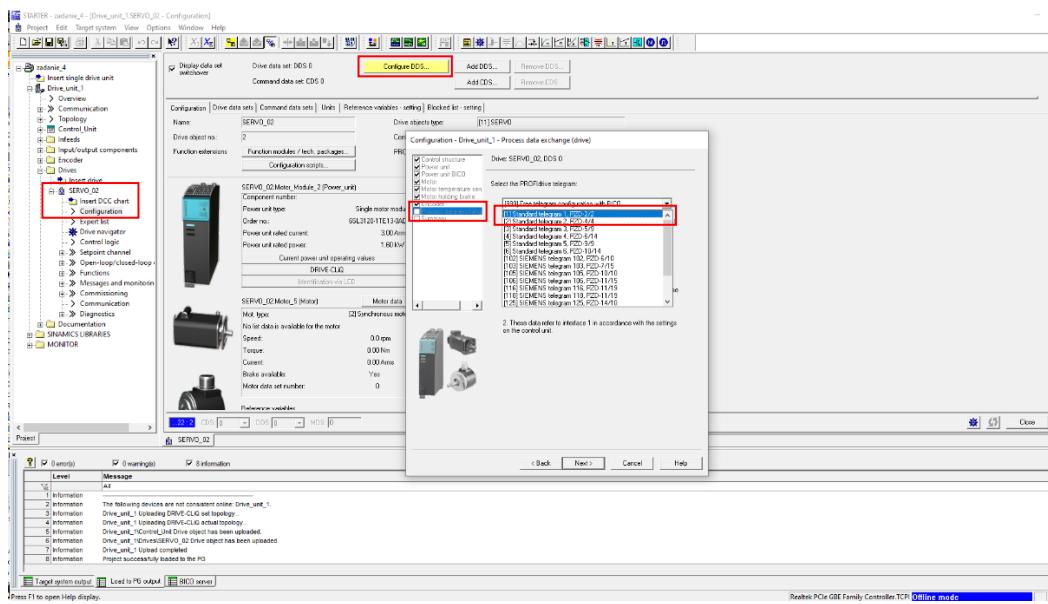
Kedže HMI panel sa správa ako rovnocenné zariadenie k PLC, neuvidíme pri ňom zelenú fajku. Rovnako ak by sme sa pripojili online na HMI panel, uvideli by sme zelenú fajku pri všetkých zariadeniach okrem PLC (najprv by sme museli nahrať konfiguráciu aj do HMI panelu, čo sme zatiaľ neurobili). Na PLC bude zelená fajka **iba vtedy**, ak PLC vidí všetky zariadenia.

### 3. Tvorba programu

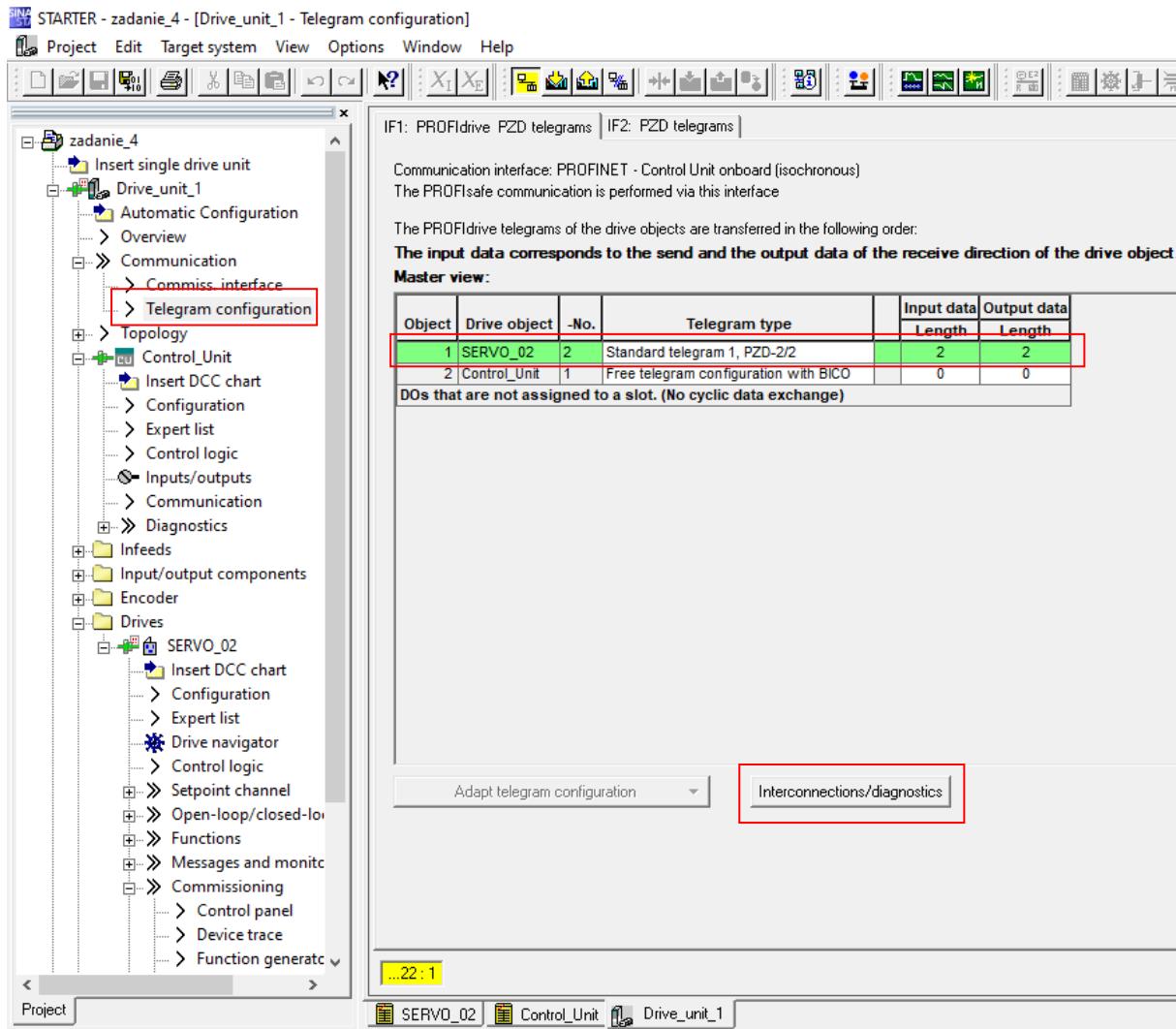
Samotné prijímanie a posielanie dát do meniča, prípadne preposlanie týchto dát do HMI panelu alebo iných periférií, zabezpečuje program nachádzajúci sa v PLC. V tejto kapitole si ukážeme podrobný postup, ako sa dá vytvoriť spomínaný program. Pre zjednodušenie návodu vytvoríme program s využitím najjednoduchšieho telegramu pre riadenie rýchlosť motoru, ktorý sa skladá z 2 slov, ktoré odosielá PLC do meniča (prvé slovo je riadiace, druhé je želaná rýchlosť), a z 2 slov, ktoré PLC prijíma od meniča (prvé slovo je stavové a druhé je skutočná rýchlosť).

#### 3.1. Nastavenie meniča

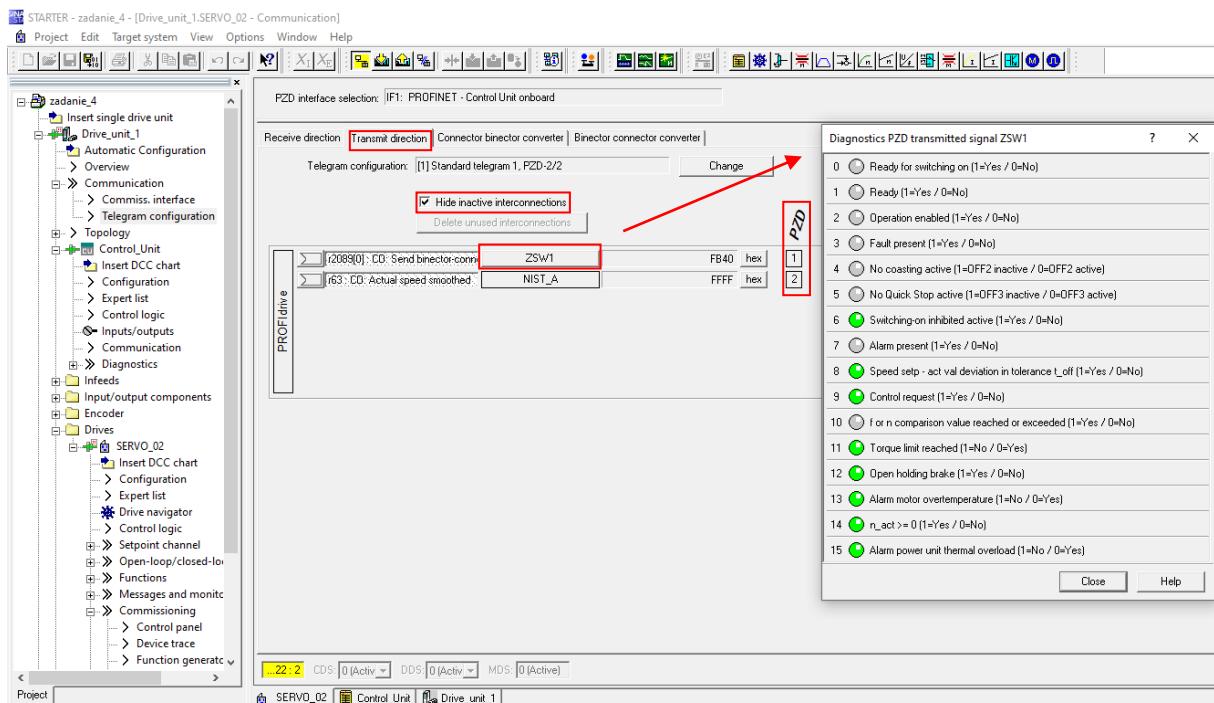
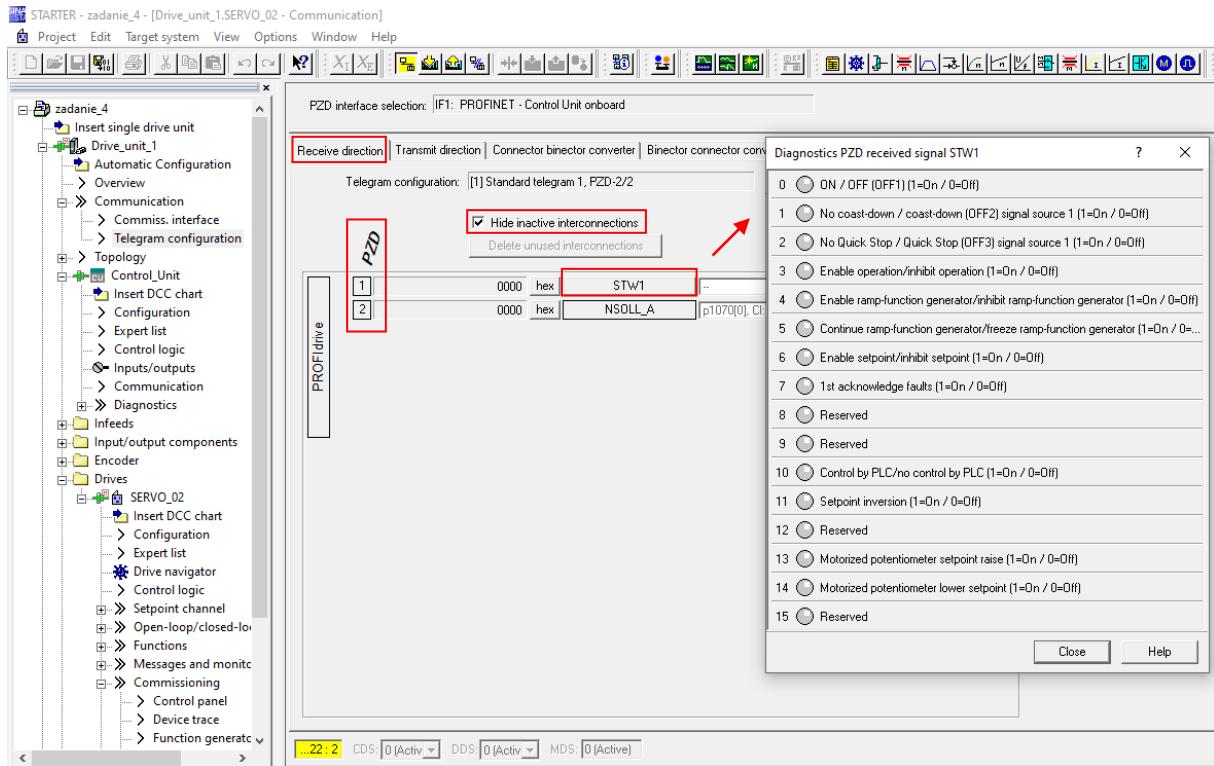
1. Ako prvé je potrebné nastaviť komunikačný telegram na strane meniča v programe Starter, z dôvodu aby menič vedel akú štruktúru údajov má prijímať a odosielat. Ako už bolo spomenuté, použijeme najjednoduchší telegram pre riadenie rýchlosť motoru „Standard telegram 1, PZD-2/2“. V programe Starter si doplníme konfiguráciu meniča o tento telegram. V „Project tree“ si zvolíme daný „Motor module“ a vyberieme okno „Configuration“. Po otvorení okna klikneme na tlačidlo „Configure DDS...“ a preklikáme sa až na stránku „Process data exchange“. Z danej ponuky PROFIdrive telegramov si vyberieme spomínaný „Standard telegram 1, PZD-2/2“. Potvrdením voľby a následným nahratím konfigurácie do meniča je nastavenie meniča ukončené.



2. V programe Starter si ešte potrebujeme pozrieť štruktúru jednotlivých slov telegramu, ktorú budeme potrebovať vytvoriť neskôr na strane PLC. Z tohto dôvodu si v „Project tree“ rozklikneme pod „Drive\_unit\_1“ možnosť „Communication“ -> „Telegram configuration“. V tabuľke, ktorá sa nám zobrazí, sú vymenované jednotlivé druhy telegramov pre riadiacu jednotku a pre motorové moduly. Všimnime si, že telegram pre komunikáciu s riadiacou jednotkou posiela a prijíma nula slov, čo znamená, že PLC nebude obsluhovať riadiacu jednotku a ani z nej nebude prijímať žiadne údaje. V našom prípade nás zaujíma telegram pri „SERVO\_02“. Zvolíme si riadok „SERVO\_02“ (označí sa na zeleno) a následne klikneme na tlačidlo „Interconnections/diagnostics“.



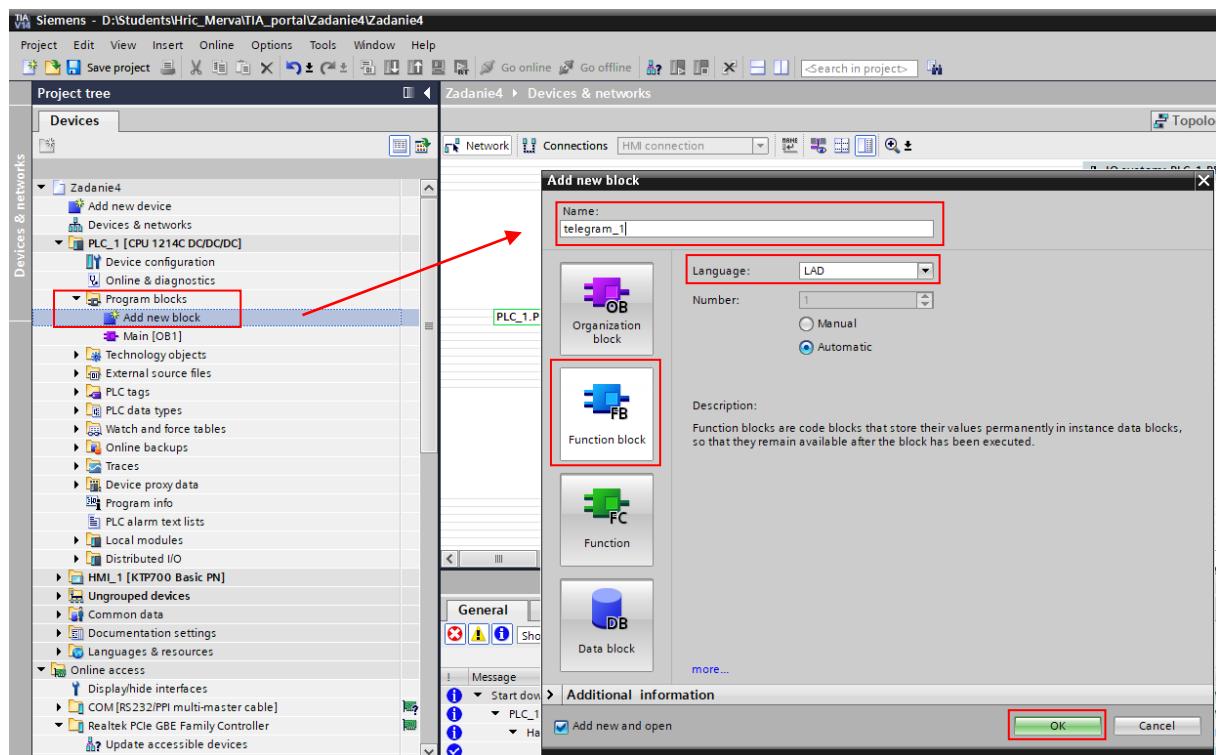
3. Následne sa nám otvorí nová obrazovka. V záložke „Receive direction“ sú zobrazené slová, ktoré posielá PLC do meniča. V záložke „Transmit direction“ sú zobrazené slová, ktoré odosielá menič do PLC. Po zaškrnutí možnosti „Hide inactive interconnections“ sa zobrazenie slov trochu zjednoduší a vďaka číslam pod nápisom „PZD“ viete zistiť presný počet párov bytov (16 bitov), z ktorých sa prisluhajúce slovo skladá. Ak chceme vidieť presnú štruktúru jednotlivých riadiacich a stavových slov, stačí kliknúť na dané slovo, ktorého štruktúru chceme zistiť, a zobrazí sa nám nové okno s danou štruktúrou. Tento proces je znázormený na nasledujúcich dvoch obrázkoch:



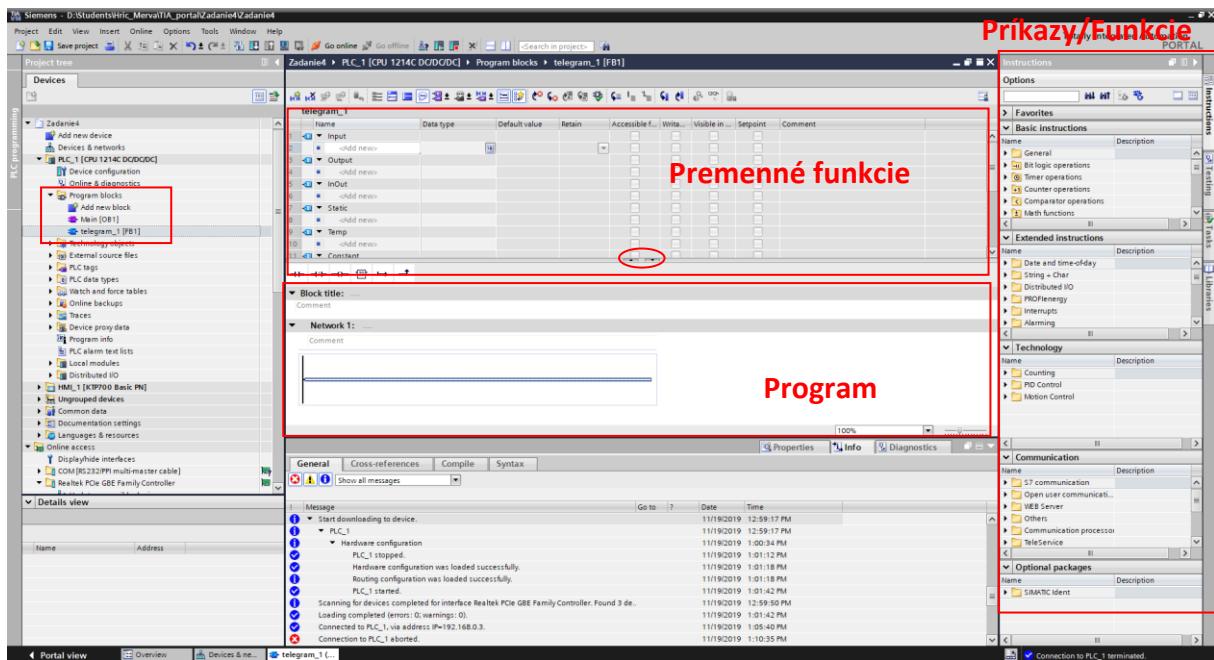
Po nastavení komunikácie na strane meniča sa vrátimo do programu TIA Portal k nášmu projektu. Hardvérovú konfiguráciu všetkých zariadení máme už urobenú a tak sa pustíme do tvorby programu.

### 3.2. Tvorba programu na PLC

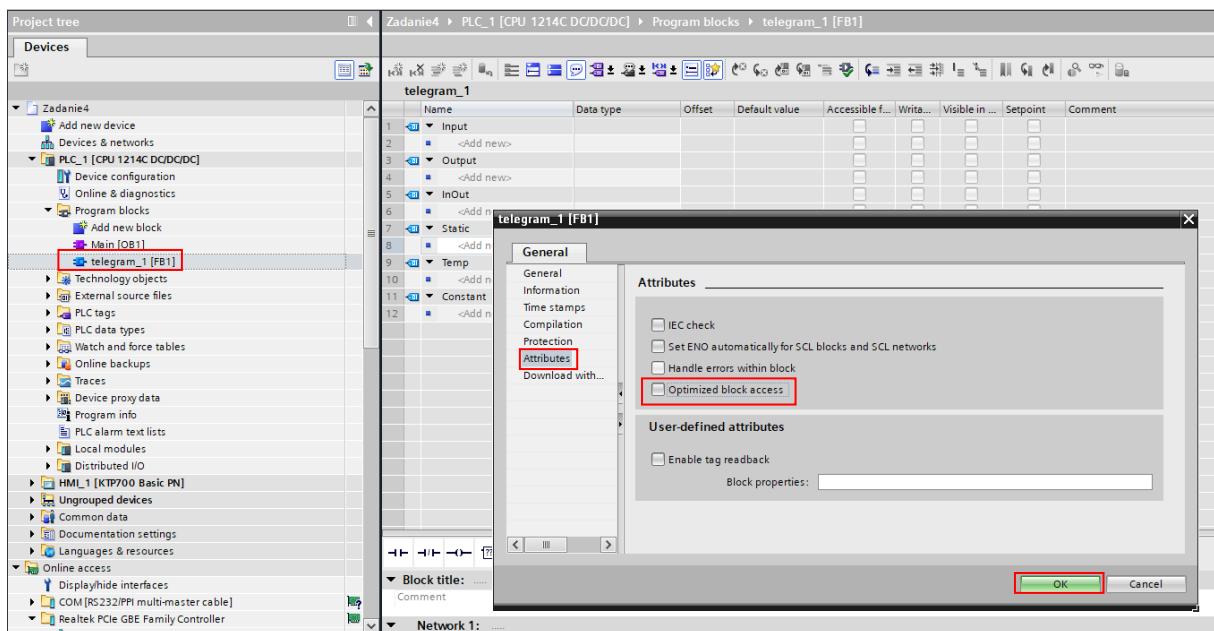
- Pre priehľadnosť a univerzálnosť programu si celý program rozložíme do viacerých blokov/funkcií, ktoré vieme v budúcich projektoch opäťovne použiť (analógia k programovaniu v jazyku C). Ako prvé si vytvoríme „*Function block*“ rozkliknutím na možnosť „*Add new block*“, ktorá sa nachádza v strome nášho projektu pod priečinkom „*Program blocks*“. „*Function block*“ je blok kódu, ktorý si uchováva hodnoty premenných v príslušnom „*Data block*“-u aj po vykonaní/ukončení funkcie. Pod pojmom „*Data block*“ si vieme predstaviť databázu premenných. Po kliknutí na možnosť „*Add new block*“ si vyberieme spomínany „*Function block*“ s názvom „*telegram\_1*“ a ako programovací jazyk si zvolíme „*LAD*“.



- Potvrdením sa nám zobrazí v priečinku „*Program blocks*“ okrem pôvodného bloku „*Main*“ aj novovytvorený blok „*telegram\_1*“. V prípade, že sa nám neotvorilo okno daného bloku, klikneme dvakrát na blok „*telegram\_1*“. Po otvorení okna môžeme vidieť „*Network 1*“, kde budeme písat náš program, panel s funkciami/príkazmi „*Instructions*“ a taktiež premenné bloku „*telegram\_1*“. Ak sa Vám nezobrazuje tabuľka s premennými, je potrebné ju rozkliknúť pomocou šípok.



3. Ešte pred začatím programovania je potrebné odškrtnúť optimalizovanie adries nášho „*Function block*“-u z dôvodu, aby sme naše štruktúry/premenné mali v presnom poradí aké pre komunikáciu potrebujeme (v zmysle poradie adries). V „*Project tree*“ klikneme na nás blok „*telegram\_1*“ pravým tlačidlom a vyberieme možnosť „*Properties*“. V novootvorenom okne v menu „*Attributes*“ odškrtneme možnosť „*Optimized block access*“.



4. Následne si potrebujeme vytvoriť dve štruktúry symbolizujúce komunikačný telegram. V tabuľke premenných si v skupine „Static“ vytvoríme štruktúru „Receive\_Data“ a štruktúru „Send\_Data“.

Do štruktúry „Receive\_Data“ sa zapisujú dve slová, ktoré menič odosielá do PLC. Stavové slovo „SW1“ je opäť štruktúra, ktorá sa skladá zo 16 bitov, pričom každý bit ma osobitný význam (vid. Kapitola 3.1 bod 3). Druhé slovo „Actual\_velocity“ predstavuje aktuálnu rýchlosť motora, pričom jej dátový typ je integer (16 bitov).

Príkazy pre menič zapisujeme do štruktúry „Send\_Data“, ktorá sa posiela z PLC do meniča. Skladá sa z jedného riadiaceho slova „CW1“ (vytvoreného opäť pomocou štruktúry) a z jedného slova predstavujúce želanú rýchlosť „Ref\_velocity“ (dátový typ je opäť integer). V prípade riadiaceho slova si nastavíme niektoré premenné na počiatočnú hodnotu „true“ z dôvodu, aby sme pri používaní meniča zapínali menič iba nultým bitom „ON/OFF1“.

	Name	Data type	Offset	Default value	Accessible f...	Writ...	Visible in ...	Setpoint	Comment
1	► Input								
2	► Output								
3	► InOut								
4	◄ Static								
5	■ □ Receive_Data	Struct	...		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
6	■ □ SW1	Struct	...		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
7	■ □ Speed_Setup	Bool	8. bit	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
8	■ □ Control_Request	Bool		false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
9	■ □ bit10	Bool		false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
10	■ □ Torque_Limit	Bool		false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
11	■ □ Open_Holding...	Bool		false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
12	■ □ Alarm_Motor...	Bool		false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
13	■ □ bit14	Bool		false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
14	■ □ Alarm_Poweru...	Bool	15. bit	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
15	■ □ Ready_for_Swit...	Bool	0. bit	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
16	■ □ Ready	Bool		false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
17	■ □ Operation_Ena...	Bool		false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
18	■ □ Fault_Present	Bool		false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
19	■ □ No_Coasting_A...	Bool		false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
20	■ □ No_Quick_Stop...	Bool		false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
21	■ □ Switching_Inhi...	Bool		false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
22	■ □ Alarm_Present	Bool	7. bit	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
23	■ □ Actual_Velocity	Int		0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
24	■ □ Send_Data	Struct	...		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
25	► Temp								
26	► Constant								

	Name	Data type	Offset	Default value	Accessible f...	Writa...	Visible in ...	Setpoint	Comment
1	► Input								
2	► Output								
3	► InOut								
4	► Static								
5	► Receive_Data	Struct	...		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
6	► Send_Data	Struct	...		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
7	► CWI	Struct	...		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
8	bit8	Bool	8. bit	...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
9	bit9	Bool		...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
10	control_by_PLC	Bool		true	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
11	setpoint_invers...	Bool		false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
12	bit12	Bool		false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
13	mot_pot_1	Bool		false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
14	mot_pot_2	Bool		false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
15	bit15	Bool	15. bit	...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
16	OFF1	Bool	0. bit	...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
17	OFF2	Bool		true	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
18	OFF3	Bool		true	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
19	Enable_operati...	Bool		true	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
20	Enable_rfq	Bool		true	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
21	Contin_rfq	Bool		true	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
22	Enable_setpoint	Bool		true	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
23	ack_faults	Bool	7. bit	...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
24	Ref_velocity	Int		0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
25	► Temp								
26	► Constant								

5. Po vytvorení telegramu je potrebné si vytvoriť vstupy „Input“ a výstupy „Output“, ktoré budú vstupovať a vystupovať do nami vytvoreného „Function block“-u „telegram\_1“. Taktiež dve „Temp“ premenné, ktoré budú slúžiť na zaznamenanie komunikačných chýb (neúspešne odoslanie dát a pod.).

	Name	Data type	Offset	Default value	Accessible f...	Writa...	Visible in ...	Setpoint	Comment
1	► Input				<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
2	Device_ID	HW_MODULE	...	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
3	ONOFF	Bool		false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
4	Ack_faults	Bool		false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
5	Velocity	Int	...	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
6	► Output				<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		
7	Ready_for_switching...	Bool	...	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
8	Ready	Bool	...	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
9	Alarm_present	Bool	...	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
10	Fault_present	Bool	...	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
11	Act_velocity	Int	...	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
12	► InOut				<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		
13	<Add new>				<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		
14	► Static				<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		
15	► Receive_Data	Struct	...		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
16	► Send_Data	Struct	...		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
17	► Temp				<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		
18	return_int	Int	...		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		
19	Warning_value	Word	...		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		
20	► Constant				<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		
21	<Add new>				<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		

Popis premenných:

I. Input:

- a. **Device\_ID** -> Vstup reprezentujúci adresu telegramu meniča. Priradíme k nej hodnotu v budúcej časti. (dátový typ: HW\_SUBMODULE)
- b. **ONOFF** -> Vstup reprezentujúci zapínanie a vypínanie meniča. Hodnotu z tohto vstupe budeme v programe zapisovať do premennej „Send\_Data.CW1.OFF1“, ktorá sa odosiela do meniča. (dátový typ: bool)
- c. **Ack\_faults** -> Vstup reprezentujúci potvrdzovanie chýb a alarmov. Hodnotu z tohto vstupe budeme v programe zapisovať do premennej „Send\_Data.CW1.ack\_faults“, ktorá sa následne odošle do meniča. (dátový typ: bool)
- d. **Velocity** -> Vstup reprezentujúci želanú hodnotu rýchlosťi. Hodnota z tohto vstupe budeme v programe zapisovať do premennej „Send\_Data.Ref\_velocity“. (dátový typ: integer)

II. Output:

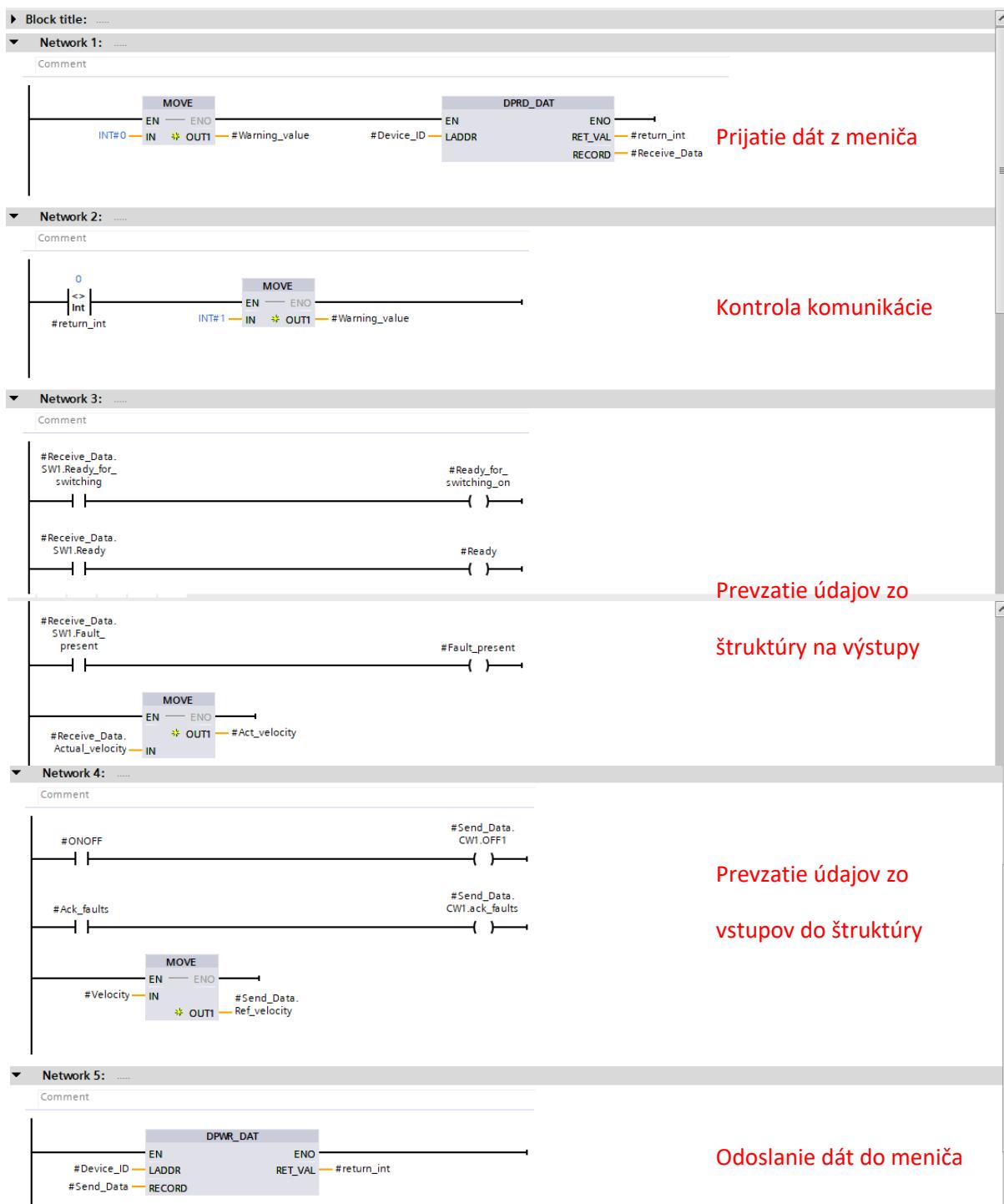
- a. **Ready\_for\_switching\_on** -> Výstup reprezentujúci signál z premennej „Receive\_Data.SW1.Ready\_for\_switching\_on“. (dátový typ: bool)
- b. **Ready** -> Výstup reprezentujúci signál z premennej „Receive\_Data.SW1.Ready\_for\_switching\_on“. (dátový typ: bool)
- c. **Alarm\_present** -> Výstup reprezentujúci signál z premennej „Receive\_Data.SW1.Alarm\_present“. (dátový typ: bool)
- d. **Fault\_present** -> Výstup reprezentujúci signál z premennej „Receive\_Data.SW1.Fault\_present“. (dátový typ: bool)
- e. **Act\_velocity** -> Výstup reprezentujúci aktuálnu rýchlosť motora prebratý z premennej „Receive\_Data.Act\_velocity“. (dátový typ: integer)

III. Temp:

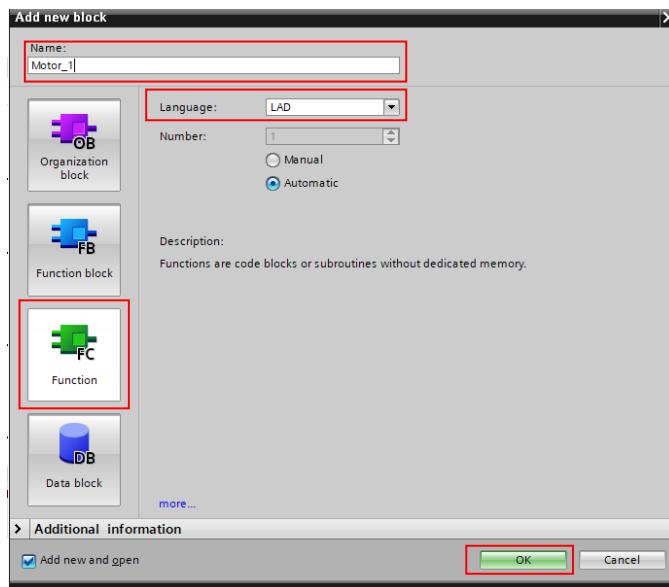
- a. **return\_int** -> Návratová hodnota z funkcie pre prijímanie dát z meniča. (analógia k programovaciemu jazyku C, dátový typ: integer)
- b. **Warning\_value** -> Príznak o vyskytnutí chyby v komunikácii. (dátový typ: word - 16 bitov)

6. Po vytvorení všetkých potrebných premenných ideme vytvoriť program. Pseudo-kód programu je nasledovný:

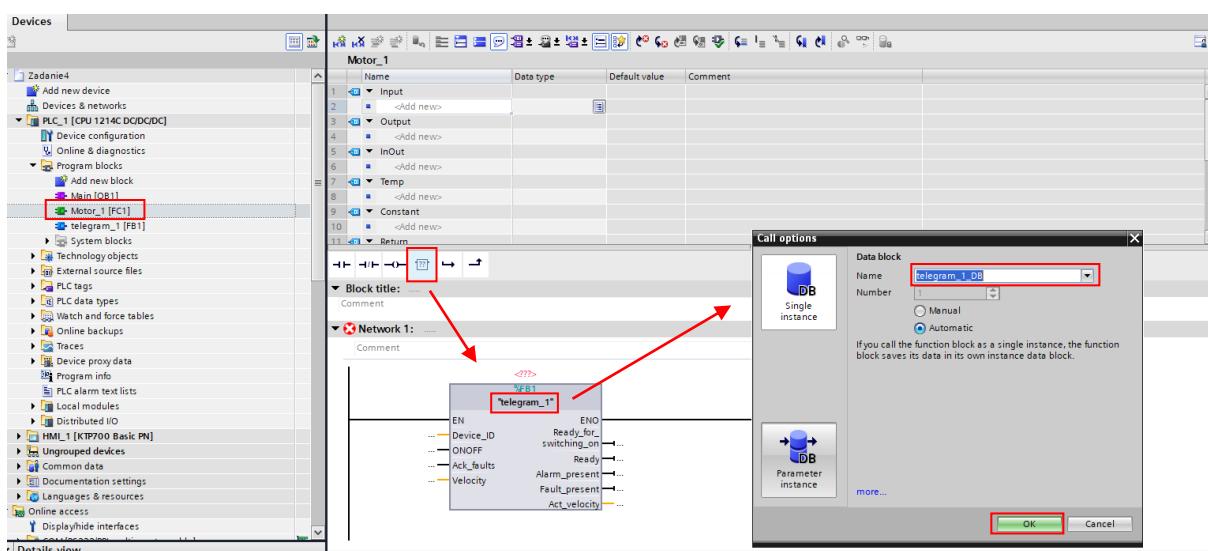
- Prijať údaje z meniča pomocou funkcie „DPRD\_DAT“ a zapísat ich do štruktúry „Receive\_Data“.
- Skontrolovanie komunikácie pomocou „return\_int“ a „Warning\_value“
- Priradiť pre nás dôležité údaje zo štruktúry „Receive\_Data“ na výstupy „Function block“-u.
- Priradiť údaje zo vstupov „Function block“-u do štruktúry „Send\_Data“.
- Odoslať štruktúru „Send\_Data“ do meniča pomocou funkcie „DPWR\_DAT“



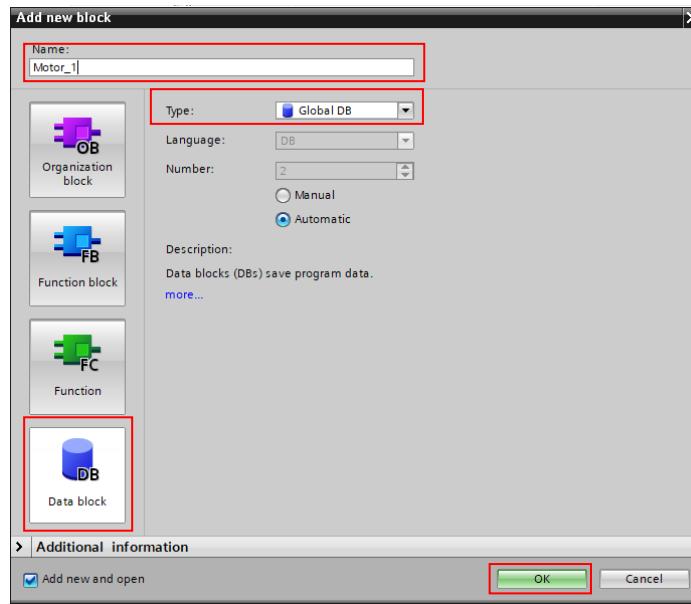
7. Takto vytvorený „Function block – telegram\_1“ vieme použiť pre ovládanie motora. Je potrebné si uvedomiť, že v PLC sa vykonáva iba to, čo je v „Main“ bloku. Preto sa nám naša funkcia zatiaľ nevykoná, keďže sme ju ešte nezavolali v „Main“-e. Pre priehľadnosť kódu si vytvoríme nový programový blok typu „Function“ s názvom „Motor\_1“, v ktorom si zavoláme náš „Function block - telegram\_1“. Následne blok „Motor\_1“ zavoláme do „Main“ bloku. Výhodou tohto na prvý pohľad komplikovaného spôsobu je, že ak by sme mali viac motorov, nemiešali by sme jednotlivé kódy motorov a v každom bloku „Motor\_X“ vieme zavolať rovnakú funkciu „telegram\_1“.



8. Po potvrdení bloku „Function“ sa nám otvorí okno funkcie „Motor\_1“. Ako prvé si zavoláme náš blok „telegram\_1“. Z ponuky základných príkazov (spínací kontakt, rozpínací kontakt,...) si vyberieme nedefinovaný blok. Po pridaní nedefinovaného bloku do „Network 1“, klikneme na meno tohto bloku a napišeme „telegram\_1“. Týmto zavoláme nami vytvorený blok. Po zavolaní nášho bloku sa zobrazí okno, ktoré nám vytvorí databázu pre uchovávanie premenných z bloku „telegram\_1“. Túto databázu si pomenujeme „telegram\_1\_DB“.



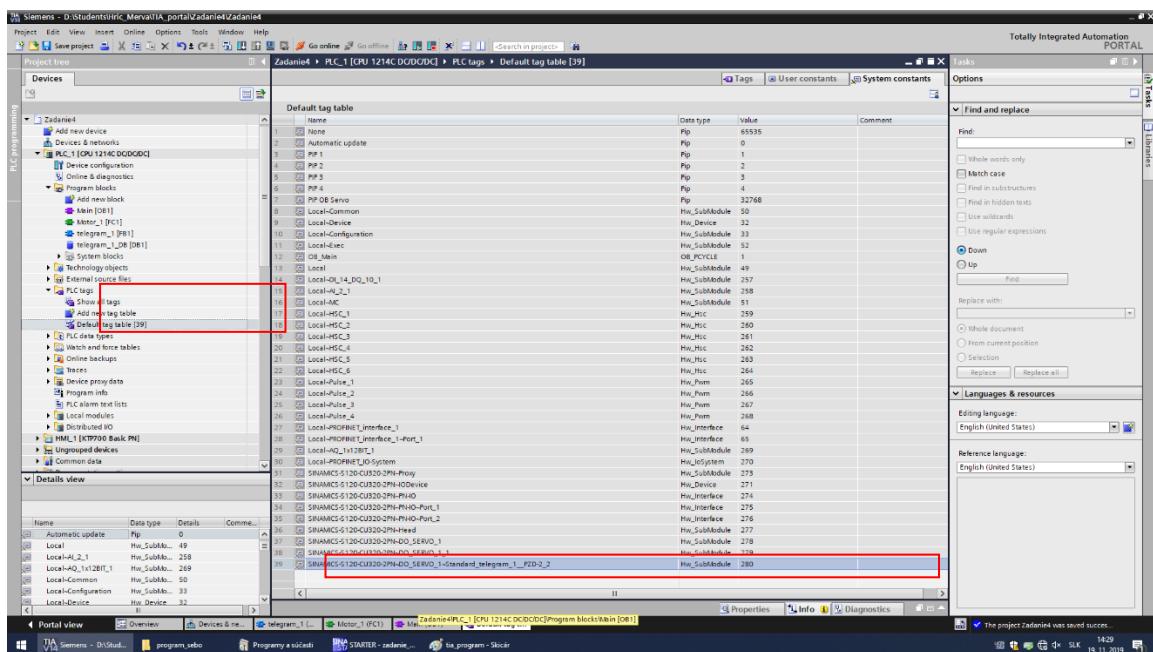
9. Následne si vytvoríme novú databázu globálnych premenných, ktoré budú jednako ovládať funkčný blok „telegram\_1“ a zároveň interagovať s premennými z HMI panela. Rozklikneme opäť možnosť „Add new block“ v „Project tree“ a vyberieme si „Data block“ a pomenujeme ho „Motor\_1\_DB“.



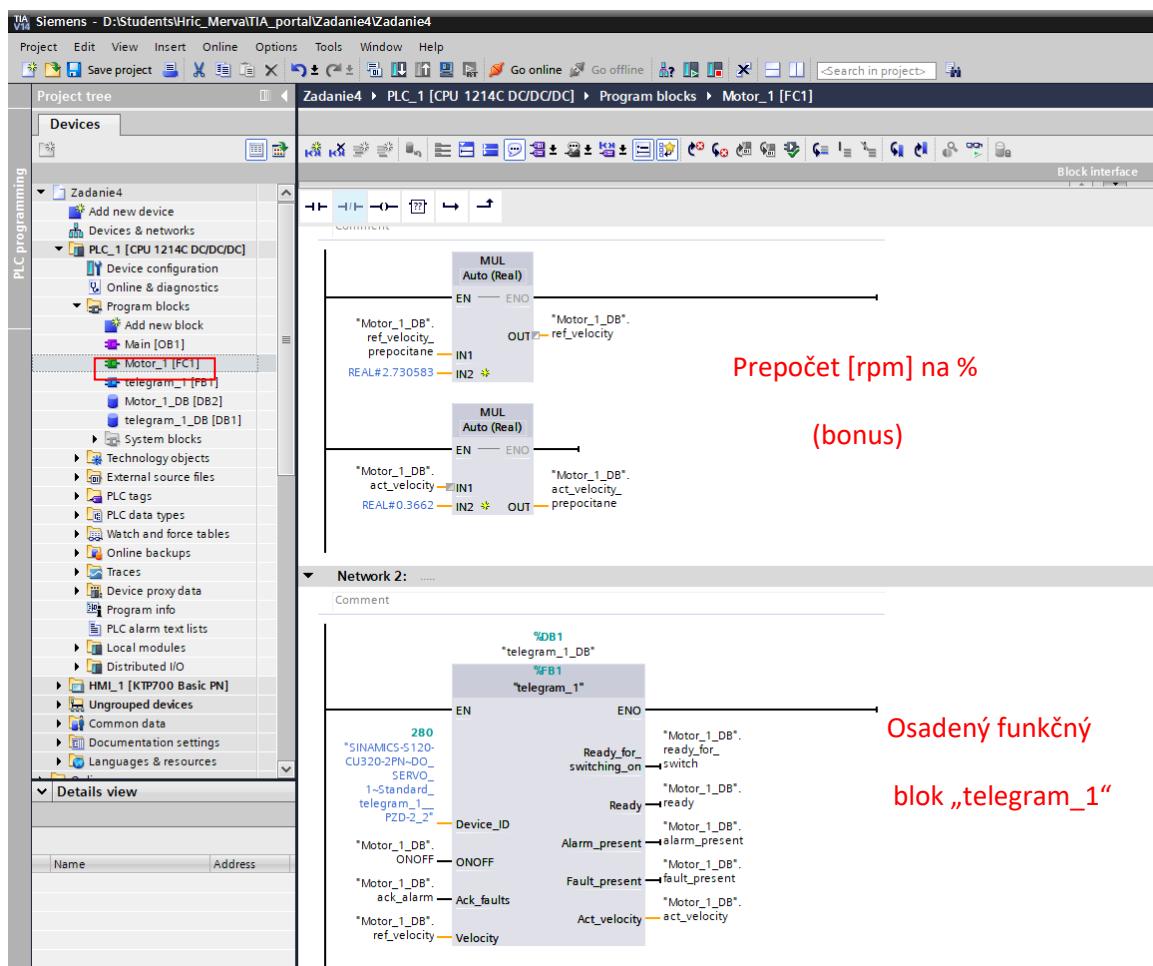
Následne si otvoríme novovzniknutú databázu „Motor\_1\_DB“ a vytvoríme si globálne premenné, ktoré privedieme na jednotlivé vstupy a výstupy bloku „telegram\_1“, podľa nasledujúceho obrázku:

Name	Data type	Start value	Retain	Accessible f...	Write...	Visible in ...	Setpoint	Comment
1 Static								
2 ONOFF	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
3 ref_velocity	Int	0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
4 act_velocity	Int	0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
5 ready_for_switch	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
6 ready	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
7 alarm_present	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
8 fault_present	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
9 ack_alarm	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
10 ref_velocity_prepocitane	Real	0.0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
11 act_velocity_prepocitane	Real	0.0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	

Po vytvorení premenných si ideme osadiť vstupy a výstupy bloku „telegram\_1“. Ako prvé si potrebujeme skopírovať adresu telegramu daného motor modulu. V „Project tree“ si otvoríme priečinok „PLC tags“ a rozklikneme možnosť „Default tag table“. Adresa telegramu bude na poslednom riadku tabuľky pod názvom „SINAMICS-S120-CU320-2PN-DO\_SERVO\_1-Standard\_telegram\_1\_PZD-2\_2“. Pre skopírovanie je potrebné označiť celý riadok.



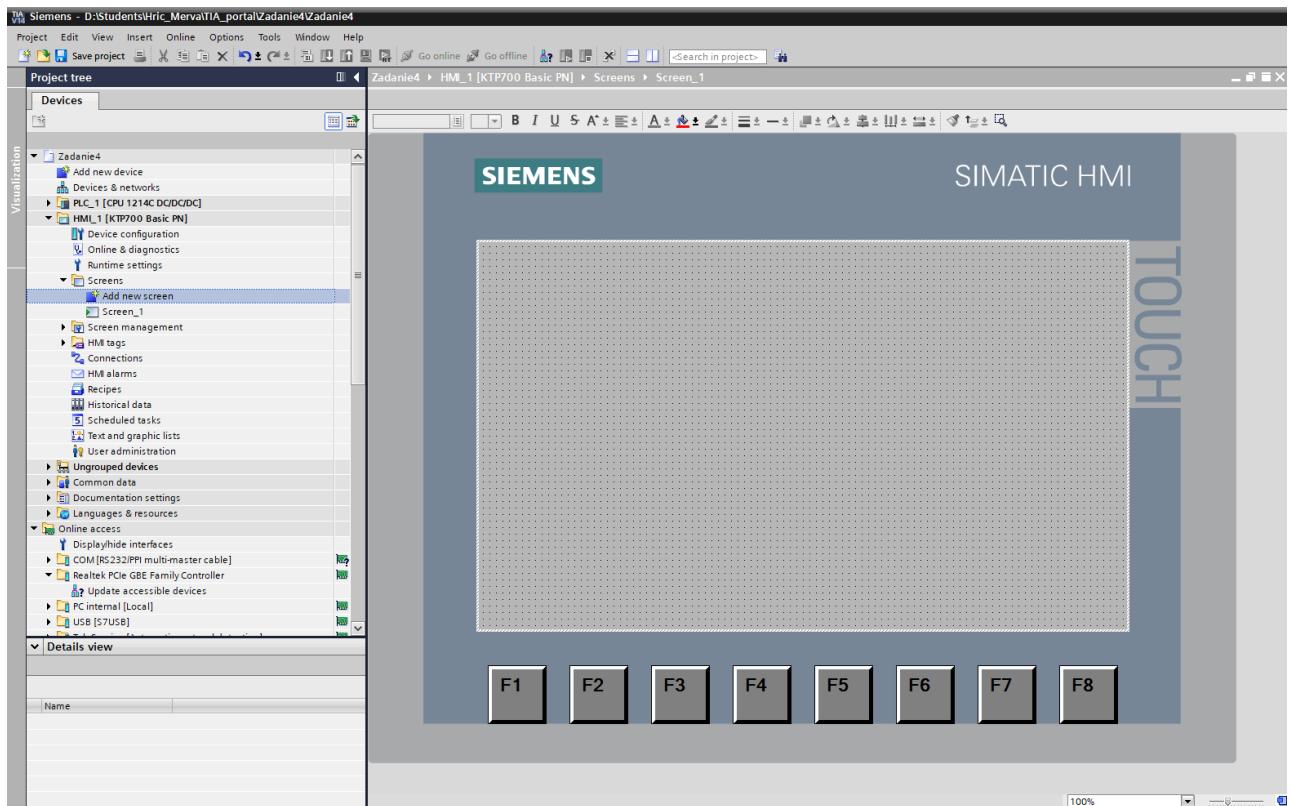
Po skopírovaní sa presunieme do funkcie „*Motor\_1 [FC1]*“ a vložíme to na vstup „*Device ID*“. Ostatné vstupy a výstupy osadíme z databázy globálnych premenných „*Motor\_1\_DB*“. Finálna funkcia „*Motor\_1 [FC1]*“ vyzerá nasledovne:



10. Posledný krok je zavolanie funkcie „Motor\_1 [FC1]“ do „Main“ bloku. (rovnaký spôsob ako v kapitole 3.2 bod 8).

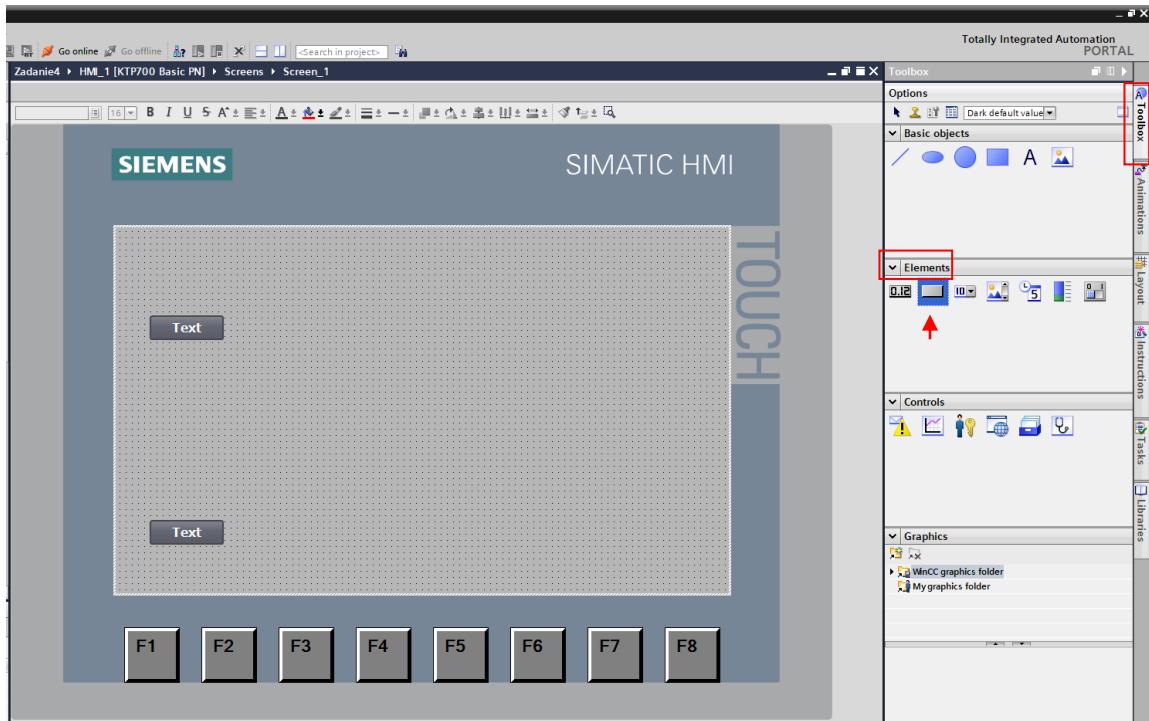
## 4. Tvorba užívateľského rozhrania

1. V okne „Project tree“ vyhľadáme zariadenie HMI panel, v našom prípade "HMI\_1". Zariadenie si rozklikneme a vyberieme možnosť "Screens". V tomto priečinku sú zobrazené všetky obrazovky používateľského rozhrania. Pokiaľ nie je vytvorená ani len jedna obrazovka, nie je možné nahrať projekt do HMI panelu (z tohto dôvodu sme v predchádzajúcich krokoch nenahrávali hardvérovú konfiguráciu do HMI panelu). Novú obrazovku pridáme rozkliknutím možnosti „Add new screen“ a následne sa nám vytvorí prázdna obrazovka pod názvom „Screen\_1“. Zobrazené tlačidlá F1-F8 predstavujú fyzické tlačidlá umiestnené na HMI paneli.

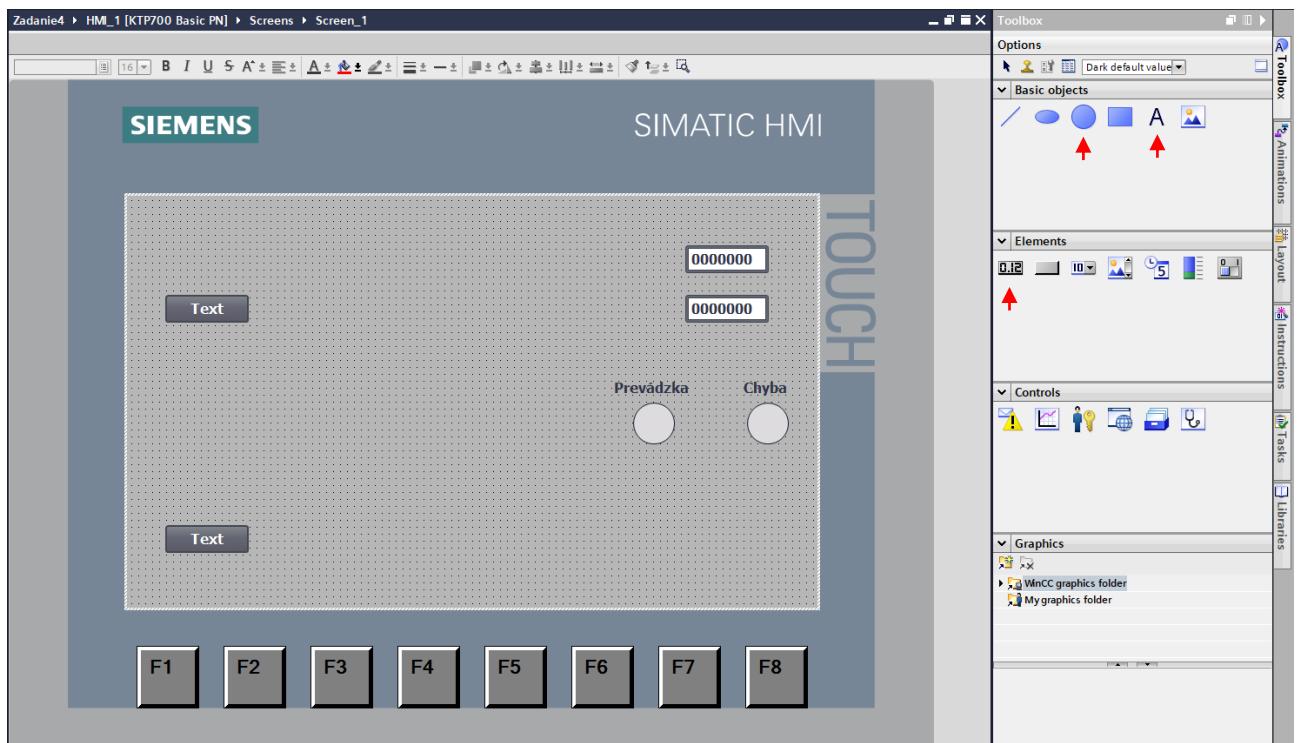


2. Pre ovládanie motora pomocou HMI panelu si pripravíme jednoduché GUI, ktoré sa bude skladať z tlačidla pre zapnutie/vypnutie meniča, z tlačidla pre potvrdenie alarmov a chýb, zo signalizácie rôznych stavov meniča a z dvoch displejov, pričom jeden bude slúžiť pre zadávanie želanej rýchlosťi motora a druhý pre zobrazovanie skutočnej rýchlosťi motora.

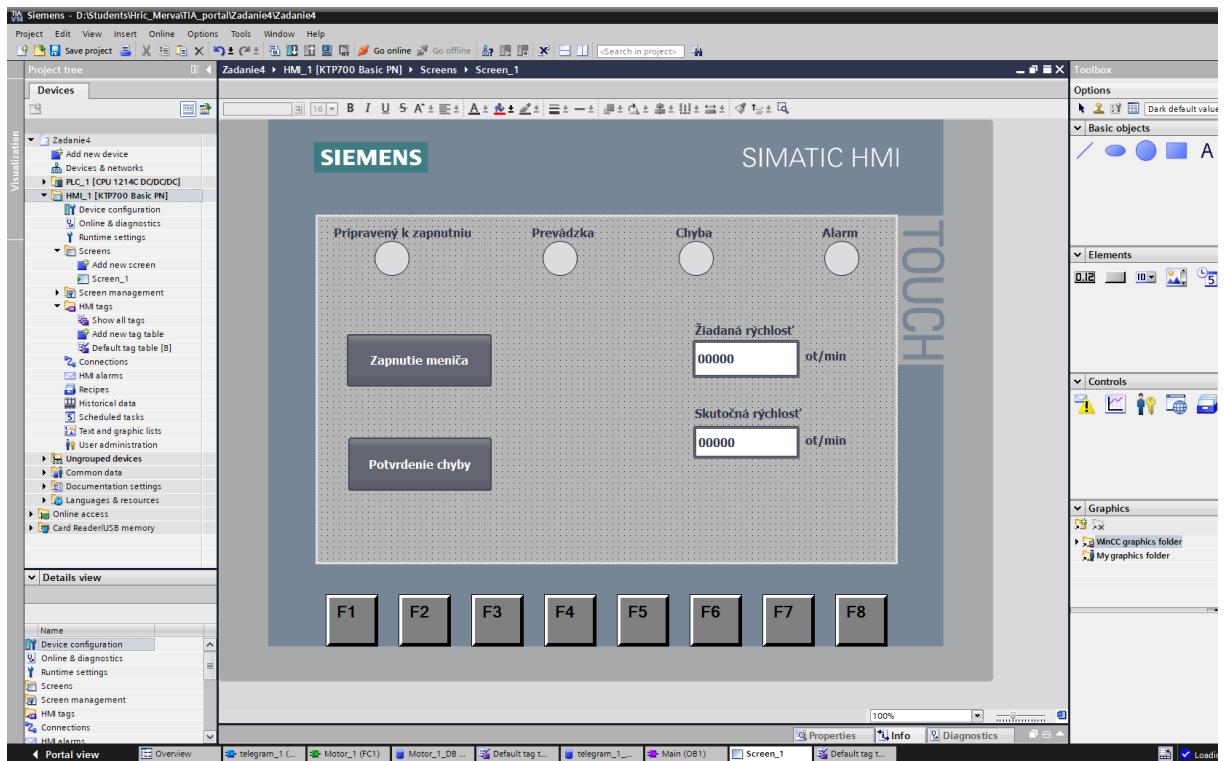
Začneme tvorbou tlačidiel pre zapnutie/vypnutie meniča a pre potvrdenie alarmov a chýb. Z pravého okna „Toolbox“ si zo záložky „Elements“ vyberieme prvok tlačidlo. Klikneme na tlačidlo a umiestníme ho na požadované miesto na obrazovke. V prípade potreby zmeny nápisu na tlačidle dvakrát klikneme do vnútra tlačidla, čo nám umožňuje prepísanie pôvodného textu.



Pre zadávanie želanej a zobrazovanie skutočnej rýchlosť použijeme prvok „Display“. Pre potreby signalizácie si do GUI pridáme LED-ky. LED-ku vieme vytvoriť pomocou prvku „Circle“, ktorý sa nachádza v záložke „Basic objects“. Pridanie textu je možné pomocou prvku „Text“.



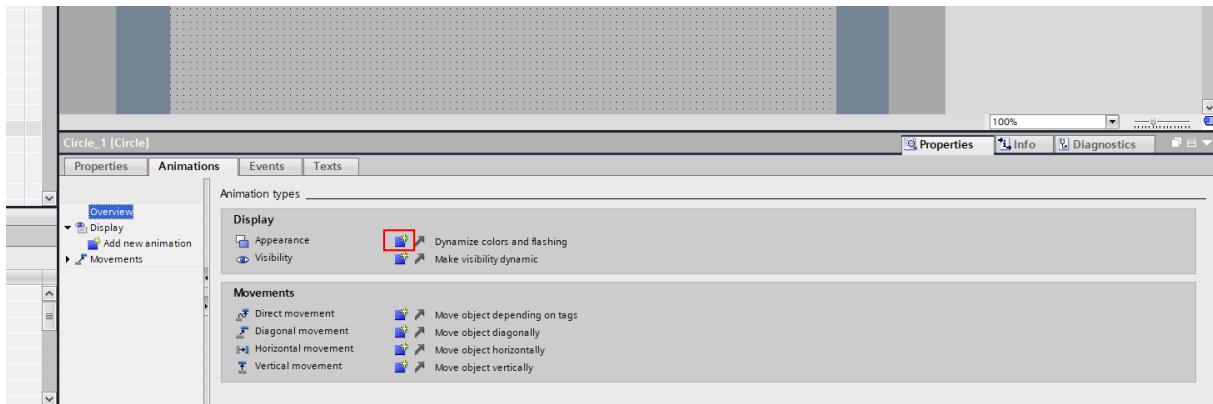
Nami vytvorené GUI má finálnu podobu nasledovnú:



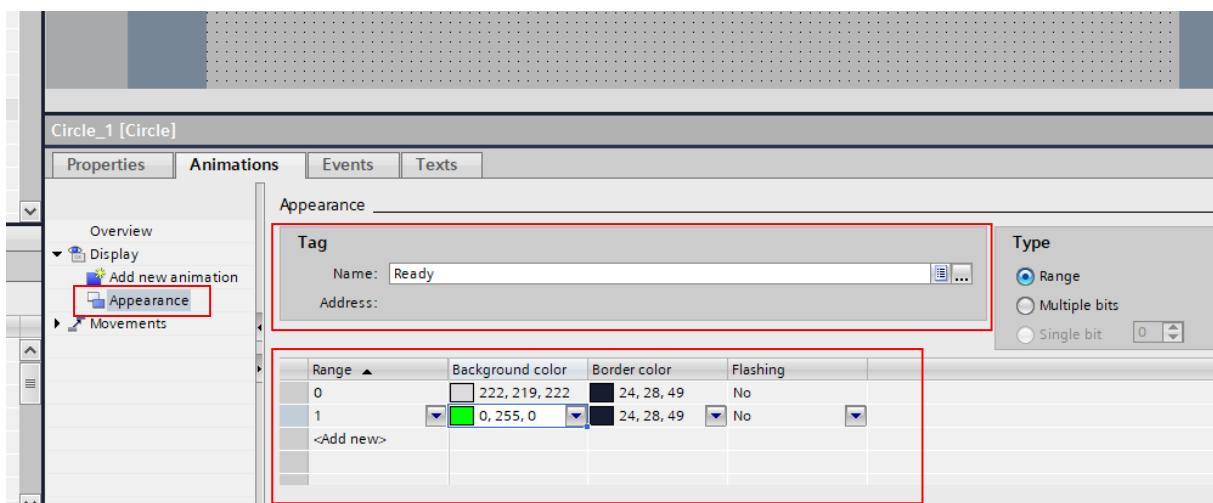
3. Nami vytvorené GUI je zatiaľ nefunkčné. Po nahratí programu do HMI panelu sa GUI sice zobrazí ale nedokáže nič riadiť/zobrazovať. Preto je potrebné si vytvoriť HMI tag-y. V prípade HMI panelu tag-y sú premenné, ktoré sa primárne používajú pre výmenu informácií medzi PLC a HMI panelom. V okne „Project Tree“ si nájdeme naše HMI zariadenie a rozklikneme si záložku „HMI tags“. Následne si vytvoríme taký počet tag-ov, koľko máme prvkov na displeji. Otvoríme si teda „Default tag table“ a klikneme na „Add new“. Ak sa jedná o tlačidlo alebo signalačnú LED-ku, za dátový typ volíme „Bool“. Ak sa jedná o displej či zadávacie pole, volíme napríklad „Real“. Aby HMI panel dokázal riadiť, prípadne snímať PLC premenné, je potrebné priradiť ku HMI tag-om prislúchajúci "PLC tag". V stĺpci „PLC tag“ klikneme na tri body a vyberieme z menu „PLC\_1“ -> „Program blocks“ -> „Motor\_1\_DB“ prislúchajúce premenné. Pre lepšiu odozvu HMI panelu pri zadávaní príkazov užívateľom sme znížili „Acquisition cycle“ z 1 s na 100 ms.

Name	Data type	Connection	PLC name	PLC tag	Address	Access mode	Acquisition cycle	Source comment
Ack_alarm	Bool	HMI_Connection_1	PLC_1	Motor_1_DB.ack_alarm		<symbolic access>	100 ms	
Act_velocity	Real	HMI_Connection_1	PLC_1	Motor_1_DB.act_velocity_preco		<symbolic access>	100 ms	
Alarm_present	Bool	HMI_Connection_1	PLC_1	Motor_1_DB.alarm_present		<symbolic access>	100 ms	
Fault_present	Bool	HMI_Connection_1	PLC_1	Motor_1_DB.fault_present		<symbolic access>	100 ms	
ONOFF	Bool	HMI_Connection_1	PLC_1	Motor_1_DB.ONOFF		<symbolic access>	100 ms	
Ready	Bool	HMI_Connection_1	PLC_1	Motor_1_DB.ready		<symbolic access>	100 ms	
Ready_for_switching	Bool	HMI_Connection_1	PLC_1	Motor_1_DB.ready_for_switch		<symbolic access>	100 ms	
Ref_velocity	Real	HMI_Connection_1	PLC_1	Motor_1_DB.ref_velocity_preco		<symbolic access>	100 ms	
<Add new>								

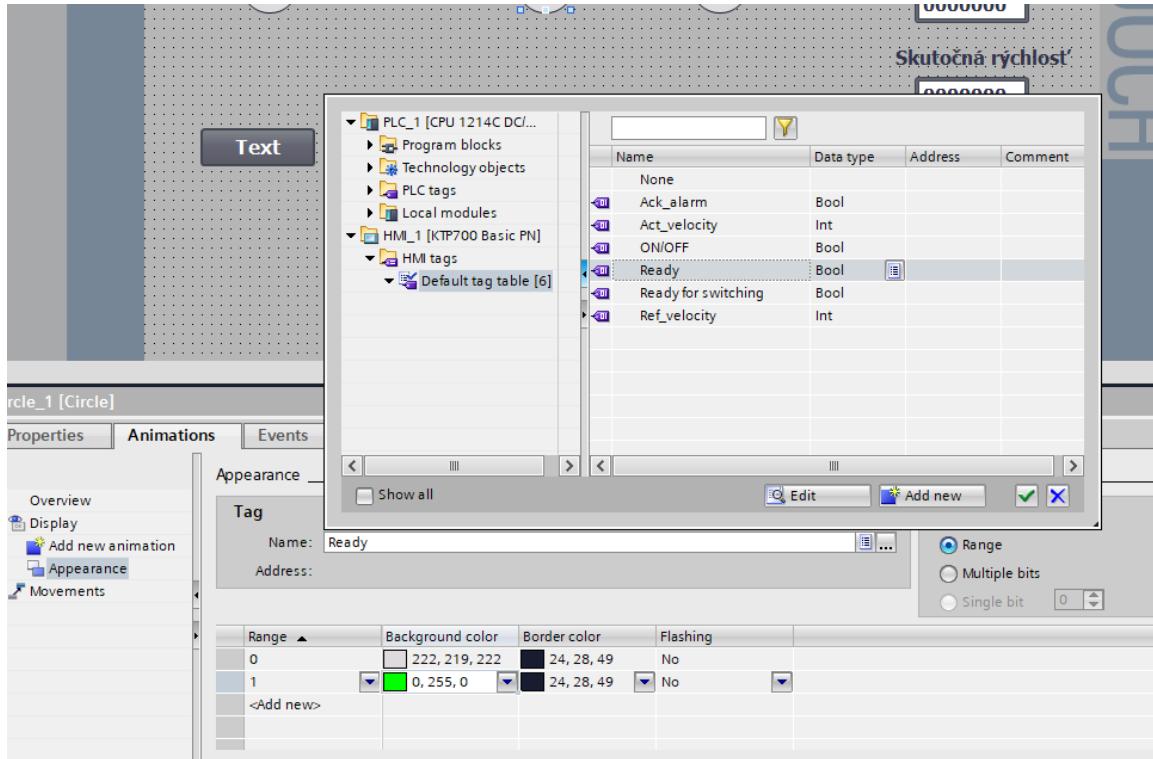
4. Po vytvorení premenných vieme nastaviť logiku pre tlačidla, displeje a signalizačné LED-ky. Pre nastavenie signalizačných LED-iek, klikneme na signalizačnú kontrolku (nami vytvorený kruh), od ktorej chceme, aby svietila pri nastavení jej prislúchajúcej premennej na určitú hodnotu. V spodnom menu „Properties“ vyberieme záložku „Animations“ a následne „Overview“. Vedľa „Appearance“ klikneme na ikonku pre vytvorenie novej animácie, ktorá bude meniť vzhľad kruhu.



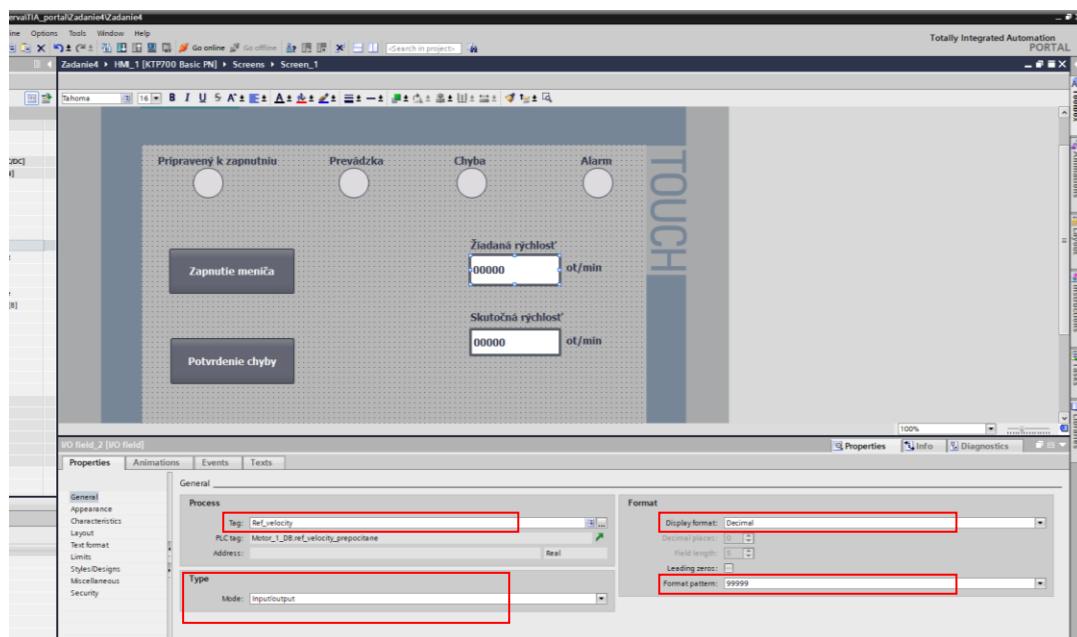
Otvorí sa nové okno „Appearance“. Do okna „Tag“ zapíšeme názov HMI tag-u (v tomto prípade tag „Ready“, ktorý nám indikuje pripravenosť meniča na zapnutie), podľa ktorého sa má daná animácia vykonávať. V tabuľke nižšie klikneme na „Add new“, a pridáme si dva nové stavy, stav 0 a stav 1. Ak má tag „Ready“ hodnotu 0, signalizačná LED-ka bude sivá. Ak má tag „Ready“ hodnotu 1, signalizačná LED-ka bude zelená. Týmto spôsobom naprogramujeme aj ostatné signalizačné LEDky.



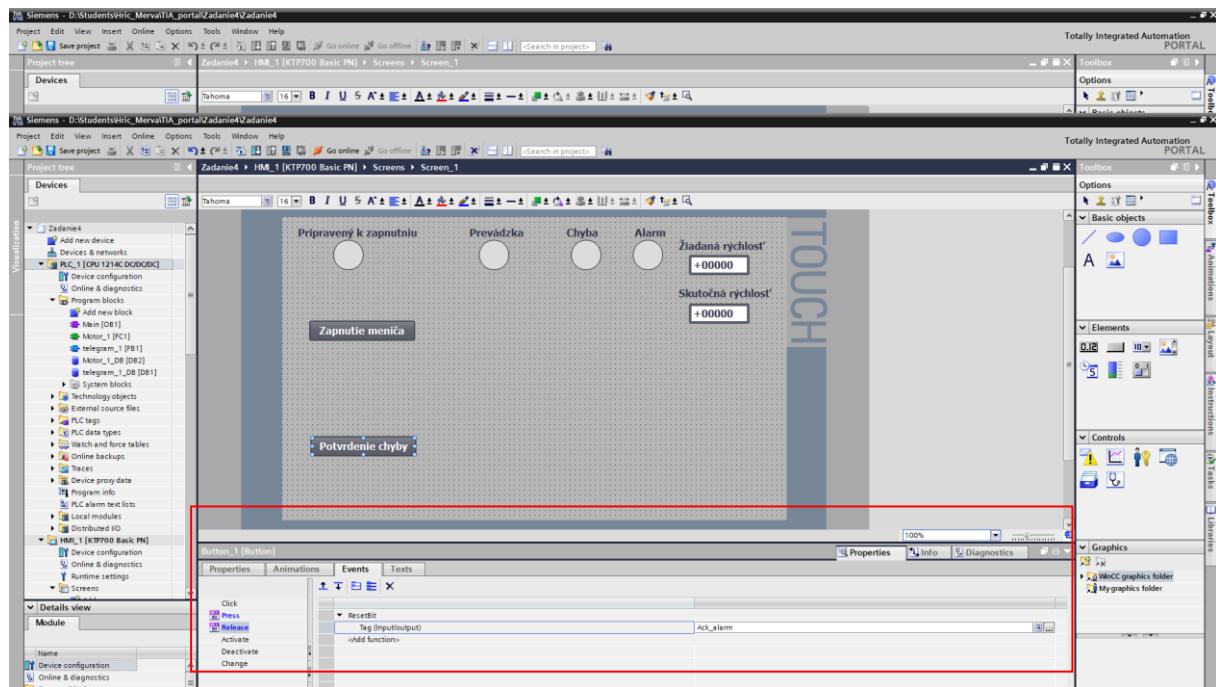
Tag vieme nastaviť aj kliknutím na tri body v riadku „Name“ a vieme si vybrať konkrétny tag zo zoznamu HMI tagov.



Podobne si nastavíme aj displeje. Po kliknutí na daný displej, si v dolnom menu „Properties“ a v záložke „General“ nastavíme názov tagu, z ktorého má displej čítať, prípadne do ktorého má zapisovať. Do okna „Type“ si zvolíme, či z daného displeja chceme iba čítať alebo iba zapisovať, prípadne aj aj. V okne „Display format“ si vyberieme v akom číselnom formáte sa nám má zobrazovať vstup/výstup.



Prejdeme k nastaveniu tlačidiel. Po kliknutí na tlačidlo v dolnom menu „Properties“ sa presunieme do záložky „Events“. Tu nastavujeme, čo sa bude diať keď vykonáme s tlačidlom nejakú akciu (teda zmenu stavu). Je potrebné si zadefinovať aký typ interakcie potrebujeme (kliknutie, zatlačenie tlačidla, uvoľnenie tlačidla,...). Zoberme si napríklad posledné tlačidlo „Potvrd’ chybu“. Jeho stlačením uvedieme príslušný tag na potvrdenie chyby na hodnotu 1. Uvoľnením tlačidla uvedieme daný tag naspäť na hodnotu 0, aby sme vedeli potvrdzovať chyby aj v budúcnosti. Postup pre naprogramovanie tejto logiky je nasledovný. V záložke „Events“ si zvolíme záložku „Press“ a zo zoznamu si zvolíme funkciu „SetBit“. Po rozkliknutí funkcie „SetBit“ si zvolíme príslušný tag „Ack\_alarm“, ktorý slúži na potvrdzovanie chýb a alarmov. Následne sa prepneeme do záložky „Release“ a zo zoznamu si zvolíme funkciu „ResetBit“. Po rozkliknutí funkcie „ResetBit“ si zvolíme rovnaký tag ako v pri funkcií „SetBit“.



Po naprogramovaní používateľské rozhrania môžeme nahrať program aj s hardvérovou konfiguráciou do HMI panelu. V okne „Project tree“ klikneme na náš HMI panel a následne v ponuke nástrojov zvolíme „Download to device“ . Objaví sa nám nasledovné okno. Postup je rovnaký ako v prípade nahrávania programu do PLC. Po správnom nastavení PG/PC rozhrania stlačením tlačidla „Start search“ nám TIA Portal nájde HMI panel, ktorý je pripojený ku počítaču. Po získaní všetkých informácií o HMI panelu si zvolíme daný HMI panel v tabuľke a nahráme projekt pomocou tlačidla „Load“. V prípade, že po stlačení tlačidla „Load“ sa zobrazia ďalšie okná, postupujeme rovnako ako v prípade nahrávania programu do PLC.

