ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

Fakulta dopravní
Katedra dopravní telematiky

Analýza využití COTS PLC pro koncept Železnice 4.0

Analysis of the Use of COTS PLC for the Railway 4.0 Concept

Bakalářská práce

Studijní program: Technika a technologie v dopravě a spojích

Studijní obor: Inteligentní dopravní systémy

Vedoucí práce: doc. Ing. Martin Leso, Ph.D.

Michal Trs

Praha 2024

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
Fakulta dopravní
děkan
Konviktská 20, 110 00 Praha 1

K620..... Ústav dopravní telematiky

ZADÁNÍ BAKALÁŘSKÉ PRÁCE
(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení studenta (včetně titulů):

Michal Trs

Studijní program (obor/specializace) studenta:

bakalářský – ITS – Inteligentní dopravní systémy

Název tématu (česky): **Analýza využití COTS PLC pro koncept Železnice 4.0**

Název tématu (anglicky): Analysis of the Use of COTS PLC for the Railway 4.0 Concept

Zásady pro vypracování

Při zpracování bakalářské práce se řiďte následujícími pokyny:

- Analyzujte požadavky na COTS PLC pro koncept Železnice 4.0
- Analyzujte dostupné produkty na bázi COTS PLC pro železniční aplikaci s bezpečnostními požadavky SIL4
- Analyzujte vývojové prostředí SILworX a PLC řady Himatrix
- Navrhněte a realizujte zátěžové testy umožňující ověření klíčových parametrů řady Himatrix pro využití v aplikacích konceptu Železnice 4.0
- Zhodnoťte možnosti využití programovatelného PLC řady Himatrix pro koncept Železnice 4.0

Rozsah grafických prací: dle potřeby

Rozsah průvodní zprávy: minimálně 35 stran textu (včetně obrázků, grafů a tabulek, které jsou součástí průvodní zprávy)

Seznam odborné literatury: LESO, Martin. Koncept Železnice 4.0. - vize digitální železnice v ČR. Praha, 2020.

CHLEBEK, Daniel. Analýza rozhraní železniční infrastruktury Konceptu Železnice 4.0. Praha, 2022.

Dokumentace firmy HIMA

Vedoucí bakalářské práce: **doc. Ing. Martin Leso, Ph.D.**

Datum zadání bakalářské práce: **5. října 2023**
(datum prvního zadání této práce, které musí být nejpozději 10 měsíců před datem prvního předpokládaného odevzdání této práce vyplývajícího ze standardní doby studia)

Datum odevzdání bakalářské práce: **5. srpna 2024**
a) datum prvního předpokládaného odevzdání práce vyplývající ze standardní doby studia a z doporučeného časového plánu studia
b) v případě odkladu odevzdání práce následující datum odevzdání práce vyplývající z doporučeného časového plánu studia

L. S.

Ing. Zuzana Bělinová, Ph.D.
vedoucí
Ústavu dopravní telematiky

prof. Ing. Ondřej Přibyl, Ph.D.
děkan fakulty

Potvrzuji převzetí zadání bakalářské práce.

Michal Trs
jméno a podpis studenta

V Praze dne 5. října 2023

Abstrakt

Tato bakalářská práce se zabývá možností využití komerčně dostupných programovatelných logických kontrolérů (COTS PLC) jako objektových kontrolérů v rámci konceptu Železnice 4.0. Železnice 4.0 představuje moderní přístup k digitalizaci a automatizaci železniční infrastruktury s cílem zvýšit bezpečnost, efektivitu a interoperabilitu.

Práce analyzuje technické a funkční požadavky na COTS PLC a jejich potenciální aplikace v železničních systémech. Detailně zkoumá konkrétní produkty dostupné na trhu a hodnotí jejich vhodnost pro implementaci do železniční infrastruktury. Součástí analýzy jsou i aspekty bezpečnosti, spolehlivosti, odolnosti a dalších důležitých vlastností, které jsou nutné pro železniční aplikace.

Cílem práce je posoudit, zda mohou COTS PLC nahradit tradiční současně používanou technologii a jakým způsobem by mohly přispět k realizaci vize Železnice 4.0. Výsledky analýzy naznačují, že použití COTS PLC může přinést významné technické a provozní výhody, což je činí vhodnou volbou pro modernizaci železniční infrastruktury v České republice.

Klíčová slova: Železnice 4.0, COTS PLC, objektový kontrolér, inteligentní dopravní systémy, digitalizace železnice

Abstract

This bachelor thesis addresses the possibility of using commercially available programmable logic controllers (COTS PLCs) as object controllers within the Railway 4.0 Concept. The Railway 4.0 Concept represents a modern approach to digitalization and automation of railway infrastructure with the aim of improving safety, efficiency and interoperability.

The paper analyses the technical and functional requirements for COTS PLCs and their potential applications in railway systems. It examines in detail specific products available on the market and evaluates their suitability for implementation in railway infrastructure. The analysis also includes aspects of safety, reliability, durability and other important features required for railway applications.

The aim of the work is to assess whether COTS PLCs can replace traditional technology currently in use and how they could contribute to the realisation of the Railway 4.0 vision. The results of the analysis suggest that the use of COTS PLCs can bring significant technical and operational benefits, making them a suitable choice for the modernisation of railway infrastructure in the Czech Republic.

Keywords: The Railway 4.0 Concept, COTS PLC, Object Controller, Intelligent Transport Systems, Digitalization of railway

Poděkování

Rád bych touto formou poděkoval panu doc. Ing. Martinu Lesovi, Ph.D., za vedení této práce, za veškeré rady a za čas, který mi věnoval. Dále bych chtěl poděkovat panu Ing. Chludilovi z firmy Casablanca INT, který mi byl velice nápomocen při návrhu, tvorbě a realizaci testovacích programů. V další řadě děkuji své rodině a přátelům za jejich podporu během celého studia.

Čestné prohlášení

Nemám závažný důvod proti užívání tohoto školního díla ve smyslu § 60 Zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon).

Prohlašuji, že jsem předloženou práci vypracoval samostatně a uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací a Rámcovými pravidly používání umělé inteligence na ČVUT pro studijní a pedagogické účely v Bc. a NM studiu.

V Praze dne 5. srpna 2024 .....

Podpis

Obsah

Seznam obrázků

Seznam tabulek

Seznam symbolů a zkratek

CENELEC	Evropský výbor pro normalizaci v elektrotechnice
COTS	Výrobek připraven k použití (Commercial Off-The-Shelf)
CRC	Cyklická kontrola redundance (Cyclic Redundancy Check)
ČVUT	České vysoké učení technické v Praze
EOV	Elektronický ohřev výhybek
FBD	Funkční blokový diagram
FD	Fakulta dopravní
I/O	Vstupy/výstupy (Inputs/Outputs)
ITS	Inteligentní dopravní systémy (Inteligent Transport Systems)
ITS-R	Železniční inteligentní dopravní systémy (ITS – Railway)
IZZ	Integrální zabezpečovací zařízení
OC	Objektový kontrolér (Object Controller)
OCEOV	Objektový kontrolér EOV
OCN	Objektový kontrolér návěstidla
OCP	Objektový kontrolér přestavníku
OCPN	Objektový kontrolér počítače náprav
OCZ	Objektový kontrolér PZZ
OLT	Online test
OS	Operační systém
PADT	Zařízení se softwarem SILworX (Programming and Debugging Tool)
PES	Programovatelný elektronický systém (Programmable Electronic System)
PLC	Programovatelný logický automat (Programmable Logic Controller)
PN	Počítač náprav
PZZ	Přejezdové zabezpečovací zařízení
SDT	Sdružení pro dopravní telematiku
SFC	Sekvenční funkční diagram (Sequential Function Chart)
SIL	Úroveň bezpečnosti integrity (Safe Integrity Level)
ST	Strukturovaný text
SWX	software SILworX
UI	Uživatelské prostředí (User Interface)
ZZ	Zabezpečovací zařízení

Úvod

Technologický pokrok v oblasti železniční dopravy přináší nové výzvy i příležitosti pro zlepšení efektivity, bezpečnosti a spolehlivosti železniční infrastruktury. Jedním z klíčových konceptů v této oblasti je Železnice 4.0, která usiluje o integraci moderních technologií a digitalizace do železničního systému. Tento koncept zahrnuje zavádění novodobých zabezpečovacích a sdělovacích technologií, které zvyšují interoperabilitu, optimalizaci a celkovou efektivitu železniční dopravy.

Vize Železnice 4.0 staví na principech inteligentních dopravních systémů pro železnici (ITS-R), které byly definovány v pozičním dokumentu pracovní skupiny SDT. [1] Cílem je vytvořit moderní, bezpečný a efektivní železniční systém, který bude schopen rychle reagovat na nové výzvy a požadavky současného světa. Klíčovým prvkem této vize je využití programovatelných logických automatů (PLC), které mohou zastávat roli objektových kontrolérů (OC).

Tato bakalářská práce se zaměřuje na analýzu využití komerčně dostupných programovatelných logických kontrolérů COTS PLC jako objektových kontrolérů v konceptu Železnice 4.0. COTS PLC jsou modulární, univerzální a snadno přizpůsobitelné systémy, které mohou být využity v různých průmyslových aplikacích, včetně železniční infrastruktury. V práci bude detailně rozebrána role COTS PLC, jejich technické a funkční požadavky, dostupné COTS produkty na trhu a specifické aplikace v železničních systémech.

Cílem této práce je posoudit, zda a jak mohou COTS PLC efektivně nahradit tradiční objektové kontroléry v železniční infrastruktuře a přispět tak k realizaci vize Železnice 4.0. Práce se také zaměří na analýzu bezpečnostních, spolehlivostních a technických aspektů COTS PLC a jejich praktického využití v reálných železničních aplikacích.

1.1 Vize konceptu Železnice 4.0

Budoucnost železniční dopravy skví, stejně jako u dalších módů dopravy, v zavádění nových moderních technologií, a to jak v zabezpečovací, tak i ve sdělovací technice. Není divu, že hlavním trendem poslední doby je digitalizace a vzájemná propojitelnost různých procesů a prvků, která přispívá k větší bezpečnosti, optimalizaci, ale i k celkové efektivitě železničního systému. Na těchto zásadách je také založený koncept Železnice 4.0, jenž vznikl na základě vize „Železniční inteligentní dopravní systémy“ (ITS-R), kterou definoval poziční dokument pracovní skupiny SDT. [1]

Koncept Železnice 4.0 přivádí nové možné řešení pro zavádění moderních technologií, naplňující vizi ITS-R, pro zvýšení bezpečnosti a pro vyšší zapojitelnost a využitelnost české železnice do dopravního systému České republiky. Toto řešení by také mělo přispět k modernizaci a implementaci vnějších prvků naší železniční infrastruktury, a to co nejvíce efektivně a za krátký časový úsek. [2]

1.2 Objektové kontroléry

Jednou z hlavních skupin technologií, kterou koncept Železnice 4.0 zavádí, a tvoří ucelený prvek nově definovaného systému, jsou objektové kontroléry (OC). Objektový kontrolér je programovatelná počítačová jednotka, která je vybavena fyzickými vstupy, výstupy a komunikačním rozhraním. OC také může disponovat základními logickými a diagnostickými funkcemi. [2]

Jak již název tohoto prvku napovídá, jedná se o zařízení, které slouží k řízení a monitorování objektů, resp. prvků vnějšího zabezpečovacího zařízení, kde konkrétní příklady možného použití budou rozebrány v jedné z kapitol této práce. K řízení prvků ZZ je použito ovládatelných výstupů, k diagnostice a monitoringu prvků pak slouží logické vstupy. OC ovládá prvky ZZ na základě komunikace s integrálním zabezpečovacím zařízením (IZZ) a na základě jeho žádostí. [2]

Objektové kontroléry se řadí mezi technologii, jenž musí splňovat nároky na elektronické zabezpečovací zařízení, které je umístěné ve venkovním prostředí, resp. ve venkovní skříni v blízkosti tratě. [2] Dalším důležitým kritériem je potřebná certifikace a schválení, které jsou nutná pro aplikaci výrobku na železniční síť.

2 COTS PLC jako objektové kontroléry

Objektové kontroléry by měly tvořit zařízení, které jsou univerzální a nezávislé na jiných výrobcích. Zároveň by se mělo jednat o zařízení, která jsou bez dalších nutných úprav připraveny ihned k použití. Zároveň by ale měla být uživatelsky modifikovatelná pro konkrétní aplikace. Pro takováto zařízení se dá použít souhrnné označení COTS (commercial off-the-shelf).

Jednou z cest řešení objektových kontroléru s COTS zařízením se ukazuje použití programovatelných logických kontrolérů (PLC) resp. programovatelných elektronických systémů (PES). PLC jsou obecné mikroprocesorové řídící systémy, které jsou již v dnešní době hojně používány, především v průmyslové automatizaci, kde se osvědčila jejich schopnost rychle a efektivně reagovat na změny v řízených procesech v reálném čase. To z nich činí klíčové komponenty v moderních automatizačních systémech. PLC jsou modulární, díky čemuž je lze snadno rozšířit o moduly a přizpůsobit je specifickým potřebám jednotlivých aplikací. Většina PLC také podporuje širokou škálu komunikačních protokolů, jako např. Ethernet/IP, CAN, Modbus TCP, Profinet, Profibus a další. [3; 4; 5; 6; 7]

V posledních letech se někteří výrobci věnují uzpůsobení těchto PLC systémů pro použití v železniční zabezpečovací technice pro funkce s nejvyššími požadavky na bezpečnost. Proto se tato práce věnuje prověření možností a způsobu použití těchto prvků ve funkci objektového kontroléru. Zda tyto prvky jsou vhodným zařízením se však ukáže až v detailním teoretickém a praktickém prověření.

COTS PLC jsou tedy programovatelné logické kontroléry, které jsou sériově vyráběny jako komerční produkty pro univerzální použití v různých odvětvích. PLC použité pro koncept Železnice 4.0 tak nemusí být nutně vyrobeny výrobcem zabývajícím se výhradní výrobou techniky pro železniční trh, ale může se jednat i o firmu, která se zabývá výrobou automatizačních prvků pro průmysl. To by mohlo znamenat značnou výhodu ve formě větší otevřenosti na železničním trhu, které by vedlo k vyšší konkurenceschopnosti výrobců těchto technologií, a tím i k poklesu pořizovacích a provozních nákladů na tyto prvky.

Použití COTS PLC může přinést ekonomické, technické a provozní výhody, což je činí potenciálně vhodnou volbou pro aplikace v rámci konceptu Železnice 4.0.

3 Požadavky na COTS PLC v železničních aplikacích

Aby bylo COTS PLC možné použít v reálném železničním provozu, je potřeba definovat základní podmínky, které by měly tyto technické prostředky splňovat. Splnění těchto požadavků je klíčová, a to nejen pro zajištění bezpečnosti a spolehlivosti železničního provozu, ale i pro dosažení potřebné efektivity a interoperability systému.

Mezi definované požadavky patří:

- Bezpečnost
- Spolehlivost
- Odolnost
- Funkční vlastnosti
- Vývojové prostředí
- Dostupnost

3.1 Bezpečnost

Bezpečnost je klíčovým požadavkem na COTS PLC pro použití v železničních aplikacích. Železniční infrastruktura spadá pod kritickou infrastrukturu ČR a je tedy nezbytné, aby COTS PLC zajišťovali bezpečný provoz na železnici a nedocházelo tak k ohrožení národní dopravy. [8] Z tohoto důvodu je nutné přísné dodržení českých i evropských norem při výrobě, ověřování a nasazování do ostrého provozu.

Jedním z nejdůležitějších aspektů bezpečnosti u železničních aplikací je dosažení určité úrovně bezpečnostní integrity SIL (Safety Integrity Level) schválené pro použití na železnici. Například u zařízení vykonávající bezpečnostně kritické funkce se požaduje dosažení SIL 4. SIL 4 je nejvyšší úroveň bezpečnostní integrity a zaručuje extrémně nízkou pravděpodobnost selhání systému, která by mohla vést k nebezpečným stavům. [12] COTS PLC tedy musí být navrženy a certifikovány tak, aby byl splněn požadavek na bezpečnostní funkce, které vedou k odolnosti proti chybám.

Dále je nutné splnění požadavků daných dalšími normami, které by se měly týkat i COTS PLC. Jedná se konkrétně o ČSN EN 50126, která stanovuje specifikace spolehlivosti, dostupnosti, udržovatelnosti a bezpečnosti (RAMS) [9; 10; 11] pro železniční systémy během jejich celého životního cyklu. Další důležitou normou je ČSN EN 50128, zaměřená na softwarové požadavky pro železniční řídicí a ochranné systémy. [9; 10; 11] Bezpečnostní požadavky na elektronické zabezpečovací systémy jsou definovány v ČSN EN 50129, která se soustředí na bezpečnostní integritu a ochranu proti selhání elektronických zařízení používaných

v železničních aplikacích. [9; 10; 11] Tyto normy vychází z evropské normy IEC 61508, která však samostatně aplikacím na železnici nevyhovuje.

3.2 Spolehlivost

Spolehlivost COTS PLC je velmi důležitým požadavkem, neboť přispívá k bezpečnosti a plynulosti železničního provozu spolu se snížením nákladů na údržbu nebo servis. Je to další aspekt, na který se hledí při posuzování produktů k dosažení určité úrovně SIL. Například u SIL 4 to znamená, že spolehlivost PLC musí být extrémně vysoká a pravděpodobnost selhání za hodinu na funkci musí být větší nebo rovná 10^{-9} a zároveň menší než 10^{-8}. [13] Zařízení používaná na železnici musí být navržena tak, aby byla schopna fungovat nepřetržitě bez výpadků po celou dobu jejich životnosti. Životnost u venkovních COTS prvků by měla být vypočítána dle spolehlivostních modelů, přičemž by docházelo k častější obměně těchto prvků (např. každých 15 let), v závislosti na životnosti celého systému.

Zvýšení spolehlivosti dále zahrnuje použití kvalitních komponent, redundantních systémů a diagnostických funkcí, které umožnují včasnou detekci a řešení potenciálních problémů. Konkrétní PLC by také měly být ověřené a otestované předešlými aplikacemi, ať už v železničním nebo alespoň v průmyslovém prostředí.

3.3 Odolnost

Důležitým faktorem pro nasazení COTS PLC do každodenního železničního provozu je odolnost vůči různým vlivům, což zajišťuje jeho spolehlivý provoz a dlouhodobou životnost. Železniční prostředí je mnohdy vystavováno náročným podmínkám, mezi které patří například extrémní teploty, vlhkost, vibrace, nebo i elektromagnetické rušení. Je tedy zásadní, aby konkrétní zařízení dokázalo těmto vlivům odolat, a to bez ztráty jeho spolehlivosti, či výpočetního výkonu, který by mohl vést k selhání systému. PLC by také měla mít kvalitní a robustní konstrukci s ochrannými kryty, které by jej chránily před prachem nebo vodou. Instalace PLC se ale uvažuje do ochranných krytů, skříní, nebo reléových místností, které samy o sobě dokážou PLC před určitými vlivy chránit.

V tomto kontextu je důležité splnění norem ČSN EN 50125, která definuje environmentální podmínky pro zařízení železničních vozidel, ČSN EN 50124, která se zabývá izolační koordinací, a ČSN EN 50121, která se týká elektromagnetické kompatibility drážních zařízení.

3.4 Funkční vlastnosti

Funkční vlastnosti představují klíčové parametry a schopnosti, které zařízení musí splňovat pro efektivní a bezpečný provoz. Tato kapitola poskytuje přehled základních funkčních vlastností PLC, které jsou důležité pro jejich fungování v náročném prostředí železniční dopravy.

3.4.1 Výpočetní výkon a rychlost zpracování dat

Při použití PLC v železničních aplikacích je potřeba dbát na vysoký výpočetní výkon. Ten je důležitý pro rychlou a efektivní reakci na změny v řízených procesech a je nezbytný pro složitější programové operace nebo programy s velkým počtem operací. V digitálním světě je klíčové zajistit robustní komunikaci, například pomocí standardů jako EULYNX, která vyžaduje rychlou odezvu a zabezpečení přenosu. Použití kryptografie pro zabezpečení komunikace je výkonově náročné, a proto je vysoký výpočetní výkon procesoru nezbytný i pro tyto operace. Rychlost zpracování dat přímo souvisí s výkonem procesoru, přičemž rychlost zpracování dat při vyšším výkonu procesoru roste.

3.4.2 Paměť

Pro správnou funkci PLC je potřeba definovat dva typy pamětí.

První typ paměti je PROM (Programmable Read-Only Memory), která slouží jako flash paměť, k ukládání uživatelských dat, nastavení a programů pro PLC. Nejčastěji se v PLC používá paměť EPROM (Erasable Programmable Read-Only Memory), která je elektricky mazatelná a její životnost dat je vysoká. EPROM umožňuje opakované mazání a přepisování dat, což je výhodné pro aplikace, kde je potřeba často aktualizovat nastavení nebo programy. Velikost této paměti by se měla pohybovat alespoň v jednotkách MB.

Druhý typ paměti je RAM (Random Access Memory), která slouží k dočasnému ukládání proměnných během běžného provozu PLC. RAM zajišťuje rychlý přístup k datům a umožňuje efektivní čtení a zápis proměnných. Tato paměť je volatilní, což znamená, že její obsah je ztracen po vypnutí napájení, ale během provozu zaručuje rychlou a spolehlivou manipulaci s daty. Pro některé aplikace může být lepší využití paměti NVRAM (Non-Volatile RAM). Tato paměť kombinuje vlastnosti RAM a nevolatilní paměti, což znamená, že si uchovává data i po vypnutí napájení. NVRAM se používá k uchovávání kritických proměnných, které musí být dostupné i po restartu systému. Velikost této paměti by se měla pohybovat alespoň v jednotkách až desítkách kB.

3.4.3 I/O rozhraní

Pro COTS PLC je I/O rozhraní, jenž tvoří vstupy, výstupy a komunikačního rozhraní, zásadními komponentami, které umožňují interakci s různými zařízeními a systémy, a tvoří tak základní prvky pro interoperabilitu.

Vstupy musí být schopny přijímat signály z různých senzorů a detektorů, které poskytují nezbytné informace pro řízení procesů. Výstupy pak na základě logiky programu ovládají akční členy. Vstupy i výstupy musí být rozšiřitelné, aby bylo možné přizpůsobit systém specifickým potřebám daných aplikací. Také by měly být podporované různé typy I/O modulů, které by se daly ke kontroléru jednoduše připojit.

Digitální vstupy a výstupy umožňují zpracování binárních signálů, což je nezbytné pro základní řídicí dvoustavové operace. Jde například o zapínání a vypínání připojených prvků periférií nebo kontrola jejich stavů na bázi binární kombinace zapojených vstupů a výstupů. Příkladem užití digitálních vstupů může být detekce polohy koncových spínačů, zatímco digitální výstupy mohou být použity pro ovládání relé nebo přímé zapínání či aktivaci připojených prvků.

Analogové vstupy a výstupy jsou důležité pro zpracování spojitých signálů, což je zásadní pro přesné monitorování a řízení procesů, jako je regulace. Analogové vstupy tak mohou být použity pro snímání hodnot z různých čidel a senzorů, jejichž změna se projeví poklesem nebo nárustem vstupního napětí. Analogové výstupy zas mohou řídit regulátory a komponenty, které jsou citlivé na změnu vstupního napětí.

Komunikační rozhraní umožňuje propojení PLC s dalšími zařízeními nebo řídicími systémy. Zajišťuje tak efektivní výměnu dat a integraci do širšího automatizačního systému. PLC by mělo podporovat více komunikačních protokolů jako Modbus, Profibus nebo Ethernet/IP, které by umožnily komunikaci s monitorovacími a ovládacími systémy, vzdálenými I/O moduly, s připojenými periferiemi anebo jinými PLC. Mělo by tak být vybaveno dostatečným počtem komunikačních portů.

COTS PLC musí disponovat rozhraními, která jsou schopná ovládat železniční periferie. V tomto kontextu lze rozlišit dva přístupy:

- **Elektrická rozhraní** – nahrazují dosavadní centralizované ovládání po kabelech, kde PLC přebírá úkoly řízení na místě a řeší je pomocí svých programovatelných I/O
- **Datová rozhraní** – minimalizují počet fyzických I/O na PLC a místo toho využívají sofistikované vnější prvky, které komunikují s PLC prostřednictvím datových protokolů.

3.5 Vývojové prostředí

Další důležitým požadavkem na COTS PLC je vývojové prostředí, který slouží jako základní nástroj pro tvorbu, konfiguraci a údržbu řídicích systémů. Tento software by měl podporovat blokové programování a bezpečný programovací jazyk, což umožňuje snadnou tvorbu a údržbu komplexních řídicích aplikací. Programovatelnost a konfigurace by měly být snadno dostupné, spolu s možnostmi pro ladění (debugging), což umožňuje rychlé a efektivní řešení problémů. Základní logika musí být dobře implementována, což zahrnuje podporu základních logických operací, jako jsou AND, OR, NOT a další. Kromě toho je podstatné, aby programovací software měl intuitivní uživatelské rozhraní, které usnadňuje práci s PLC i méně zkušeným uživatelům. Celkově by měl programovací software umožnit snadnou tvorbu, konfiguraci a údržbu řídicích systémů. V neposlední řadě je nutné schválení

3.6 Dostupnost

Dalším důležitým požadavkem na COTS PLC je dostupnost, která zahrnuje jak ekonomickou efektivitu, tak dlouhodobou podporu. Nákladovost těchto systémů musí být přiměřená, aby se zajistila jejich ekonomická výhodnost ve srovnání s proprietárními řešeními. To zahrnuje nejen pořizovací náklady, ale také náklady na údržbu a provoz. Dlouhodobá podpora je klíčová pro zajištění kontinuálního provozu a zahrnuje dostupnost náhradních dílů, pravidelné aktualizace softwaru a spolehlivý technický servis. Tím se zajišťuje, že investice do COTS PLC je udržitelná a efektivní po celou dobu životnosti systému.

4 Dostupné produkty na bázi COTS PLC pro železniční aplikace

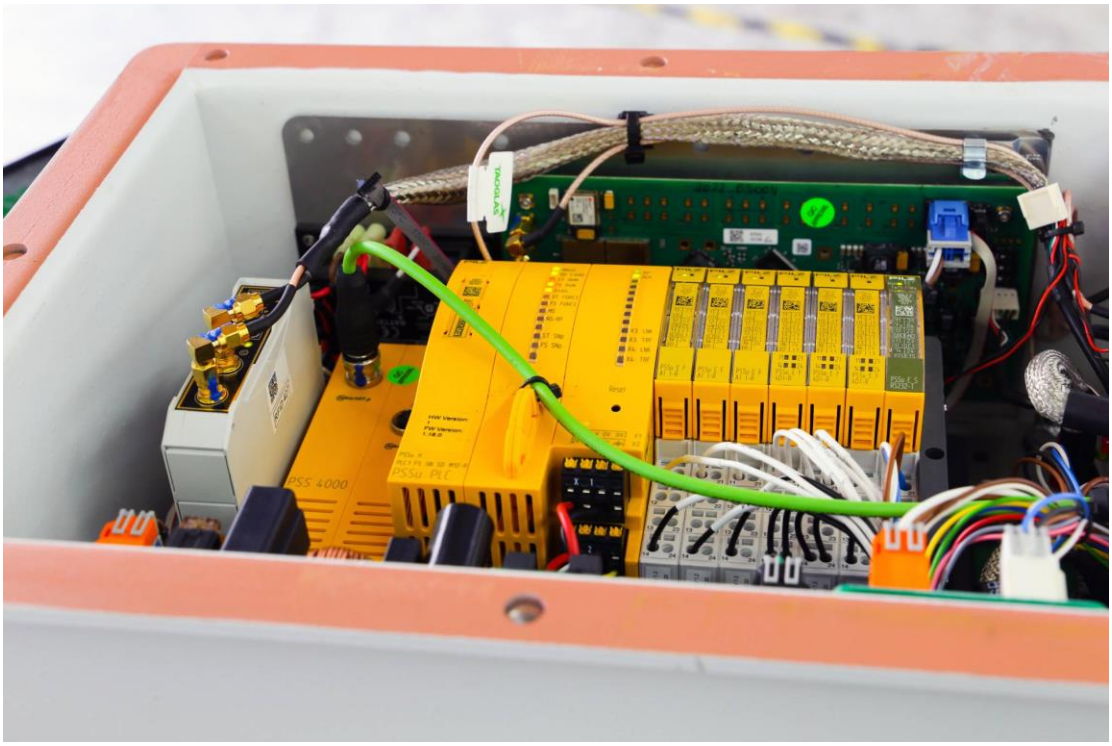
Tato kapitola se zaměřuje na dostupné produkty založené na komerčně dostupných programovatelných logických kontrolérech pro železniční aplikace. V analýze byly identifikováni dva hlavní ověření výrobci, kteří se v současné době na trhu zabývají výrobou COTS PLC kontrolérů splňující přísné normy pro aplikace na železnici. Jedná se konkrétně o firmy Pilz a HIMA jenž se proslavily výrobou PLC a jiných řídících systémů a zařízení používajících se v různých průmyslových odvětvích.

4.1 Pilz

Pilz GmbH & Co. KG je významný výrobce PLC kontrolérů a dalších technických prostředků pro zejména pro průmyslové aplikace. Společnost byla založena v Německu a má dlouhou historii inovací v oblasti bezpečnosti a automatizace. V současné době také nabízejí jeden produkt splňující parametry COTS PLC a který je vyráběn výhradně pro železniční aplikace. [16]

4.1.1 PSS 4000-R

Produkt PSS 4000-R je COTS PLC kontrolér (*obrázek 1*), který splňuje požadované normy pro železniční aplikace a který splňuje úroveň bezpečnostní integrity SIL 4. Tento produkt je založen na PLC kontroléru PSS 4000, který je osvědčeným, bezpečným a testovaným produktem firmy Pilz. Na rozdíl od této základní verze je PSS 4000-R ale speciálně upraven tak, aby jeho použití bylo vhodné pro aplikace na železnici.



Obrázek 1: Kontrolér PSS 4000-R

Do kontroléru je možné nahrávat uživatelské programy vytvořené v softwaru PAS4000, přes který je možné provádět i jeho konfiguraci a diagnostiku celého systému.

4.2 HIMA

Německá firma HIMA Paul Hildebrandt GmbH je významným výrobcem bezpečnostních řídicích systémů a PLC kontrolérů. Společnost HIMA používá zkratku PES, kterou lze v tomto případě vnímat jako synonymum k PLC. Na rozdíl od firmy Pilz mají širší portfolio produktů, které splňují přísné železniční normy a úroveň bezpečnostní integrity SIL 4 podle CENELEC. Jejich produkty jsou již nyní rozšířené v železničním průmyslu díky jejich spolehlivosti, modularitě a snadné integraci do stávajících systémů.

4.2.1 HIMatrix

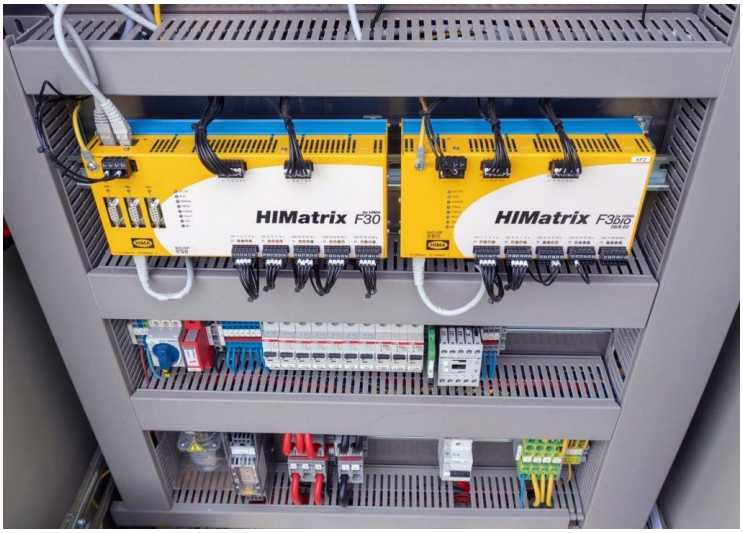
Mezi její nejrozšířenější produktovou řadu patří produktová řada HIMatrix, je zaměřena na kontroléry spadajících do programovatelných elektronických systémů (PES), jenž můžeme v kontextu s touto prací přirovnat k PLC. Ty jsou navržené pro aplikace vyžadující rychlou odezvu a vysokou spolehlivost, zvláště v malých až středně velkých projektech. Produkty této řady lze také použít pro železniční aplikace, zejména pro řízení prvků železniční infrastruktury. V zahraničí se kontroléry HIMatrix začínají běžně vyskytovat na železnici, zvláště pak u PZZ, kde by jejich výskyt měl být nejčetnější.Řada HIMatrix je vhodná právě pro tyto aplikace

zejména díky své konstrukci a kompaktnosti. Programování a konfigurace kontrolérů této řady je realizováno softwarem SILworX.

PES řady HIMatrix můžeme dle konstrukce rozdělit na následující dvě skupiny:

- Kompaktní systémy
- Modulární systémy

Kompaktní systémy integrují všechny nezbytné komponenty, jako jsou procesor, vstupy a výstupy a komunikační rozhraní, do jednoho pouzdra. Tyto systémy jsou navrženy tak, aby poskytovaly vysokou úroveň bezpečnosti v malém, ale robustním balení, což je činí ideálními pro aplikace s omezeným prostorem nebo tam, kde je potřeba rychlá instalace. Díky integrovanému designu jsou kompaktní systémy jednodušší na instalaci a uvedení do provozu. Mezi kompaktní systémy se řadí PES F30 (*obrázek 2*) a PES F35, jenž je popsán v *kapitole 5*. Do kompaktních systémů lze také zařadit kompaktní moduly, které je možné řídit vzdáleně. Tento kompaktní systém je možné propojit s dalšími systémy HIMatrix přes SafeEthernet. [17]



Obrázek 2: Kontrolér HIMatrix F30 s I/O modulem F3DIO

Modulární systém HIMatrix F60 (*obrázek 3*) se skládá z různých modulů, které mohou být přidávány nebo vyměňovány. Tento systém zahrnuje rackovou skříň F60 GEH, do které lze zasunout až 8 modulů, jako jsou například napájecí moduly F60 PS, procesorové moduly F60 CPU a variantní moduly se vstupy a výstupy. Tato konstrukce umožňuje snadnou adaptaci a rozšíření systému o další moduly podle specifických potřeb aplikace. Napájecí modul PS musí být vždy umístěn do slotu č. 1 a procesorový modul CPU musí být umístěn vždy do slotu č. 2. Sloty

č. 3-8 jsou volné pro instalaci variantních modulů se vstupy a výstupy dle potřeby. Tento modulární systém je také možné propojit s dalšími systémy HIMatrix přes SafeEthernet. [17]

Obrázek 3: HIMatrix F60

4.2.2 HIMax

HIMax je výkonný bezpečnostní kontrolér navržený pro aplikace vyžadující maximální bezpečnost (*obrázek 4*), spolehlivost a dostupnost. Jde o kontrolér s modulární architekturou, který je možné v případě potřeby jednoduše rozšířit o další moduly. Narozdíl od kontrolérů řady HIMatrix je aplikace produktu HIMax doporučena pro použití v projektech většího rozsahu. Na železnici je tento kontrolér využíván především jako ovládací prvek staničního ZZ, kde spolu s produkty řady HIMatrix tvoří jeden ucelený systém. Tento kontrolér lze, stejně jako u řady HIMatrix, programovat a konfigurovat skrze software SILworX.

Obrázek 4: Kontrolér HIMax s I/O modulem řady HIMatrix

4.2.3 HIJunctionBox

HIMA HIJunctionBox (*obrázek 5*) je ucelený výrobek, který je poskládán z různých produktů firmy HIMA dle požadavků zákazníka. Lze ho tak nakonfigurovat např. s kontroléry řady HIMatrix. Ochranná skříň tohoto produktu je dimenzována na stupeň krytí IP66 a je vybavena potřebnými nosnými prvky, ke kterým je připevněn potřebný hardware. Produkt je navržen k okamžité montáži na potřebné místo, kdy je po připojení ke kabeláži ihned schopný fungovat. Není tak nutné dalších projekčních a montážních prací, díky čemuž může dojít k značným úsporám při realizaci projektů. Velmi podobným způsobem by mohl být koncipován systém objektového kontroléru konceptu Železnice 4.0.

Obrázek 5: HIJunctionBox s kontrolérem HIMax

5 HIMatrix F35-03

Tato kapitola je zaměřena na důležité vlastnosti a funkce kontrolérů F35 z řady HIMatrix (*obrázek 6*). Tyto kontroléry jsou, dle mé analýzy, v železničních aplikacích nejrozšířenější, a to vzhledem k jejich kompaktnosti a řadě důležitých funkcí, jimiž kontroléry F35 disponují. Jeden kontrolér HIMatrix F35-03 byl včetně kompletní licence k prostředí SILworX zapůjčen do Dopravního sálu FD ČVUT společností HIMA Slovakia s.r.o., panem Michalem Dysko.

Obrázek 6: HIMatrix F35

5.1 Bezpečnost

HIMatrix F35 splňuje všechny potřebné bezpečnostní požadavky pro železniční aplikace, jež byly definovány v *kapitole 3.1*. Splnění těchto požadavků potvrzuje certifikát německého sdružení TÜV.

Kontroléry F35 mohou být použity u aplikací, které si žádají patřičné „fail-safe“ řešení při vzniku různých nebezpečných situací. To přispívá ke snížení následků takovéto situace a zajišťuje tak bezpečnost a ochranu lidských životů, majetku i životního prostředí.

5.1.1 Self-testy

Kontrolér F35 je schopný vykonávat samotestování (self-test), přičemž se jedná o diagnostické testy pro ověření funkčnosti a bezchybnosti vykonávaných procesů. Operační systém procesorového systému provádí rozsáhlé self-testy při spuštění i během provozu kontroléru. V rámci self-testu se kontrolují následující prvky:

- Procesory
- Paměti

- Watchdog časovač
- Individuální I/O kanály
- Napájení

5.1.2 Bezpečné stavy

Pokud operační systém během self-testu detekuje jakékoliv chyby, které by mohly způsobit vznik nebezpečné provozní situace, tak se systém uvede do bezpečného stavu.

Hlavní kontrolér tak vypne vadné připojené zařízení (další moduly nebo kontroléry) nebo své I/O kanály. U neredundantních systémů pak může dojít k disfunkci určitých částí kontroléru, nebo dokonce k jeho nouzovému vypnutí. Při nouzovém vypnutí kontroléru dojde k přepnutí I/O kanálů do bezpečného „de-energized“ stavu, což znamená, že všechny výstupy budou nastaveny na hodnotu 0. Při návrhu systému je tedy nutné s tímto faktem počítat a zajistit u dané aplikace redundanci nebo jiné protiopatření, které přepne systém do uživatelem definovaného bezpečného stavu.

V případě chyby v jednom z nahraných programů se vadný program přestane zpracovávat. Běh ostatních programů to však neovlivní, pokud není jejich fungování přímo závislé na vadném programu. Při vytváření programů je nutné na tuto reakci myslet a případně vytvářet vzájemné závislosti a kontroly programů, které uživateli umožní definovat jejich bezpečný stav.

5.2 Základní parametry

Zdrojové napětí pro kontrolér HIMatrix F35 je 24 VDC přičemž maximální proudová spotřeba je 9 A při maximálním zatížení a proudová spotřeba při nečinnosti je 0,5 A. Kontrolér je opatřen pamětí, která je určená pro ukládání dat a programů kontroléru s hodnotu 5 MB, přičemž 64 kB z ní je rezervováno pro CRC. Kontrolér splňuje stupeň krytí IP20 a je provozuschopný při teplotách 0…+60°C.

5.3 I/O rozhraní

I/O rozhraní systému HIMatrix F35-03 zahrnuje 24 digitálních vstupů, 8 digitálních výstupů, 2 čítače a 8 analogových vstupů, které zajišťují širokou škálu možností pro připojení a ovládání různých zařízení.

Digitální vstupy jsou navrženy pro zpracování signálů v rozmezí 0 až 30 V a zahrnují ochranu proti přepětí. Každý vstup je schopen detekovat vysoké a nízké úrovně napětí, což umožňuje přesné monitorování stavu připojených zařízení. Digitální výstupy mohou dodávat až 2 A na

kanál, což je vhodné pro ovládání různých akčních členů, jako jsou relé nebo solenoidy. Výstupy jsou navíc chráněny proti přetížení a zkratu, což zajišťuje bezpečný provoz systému.

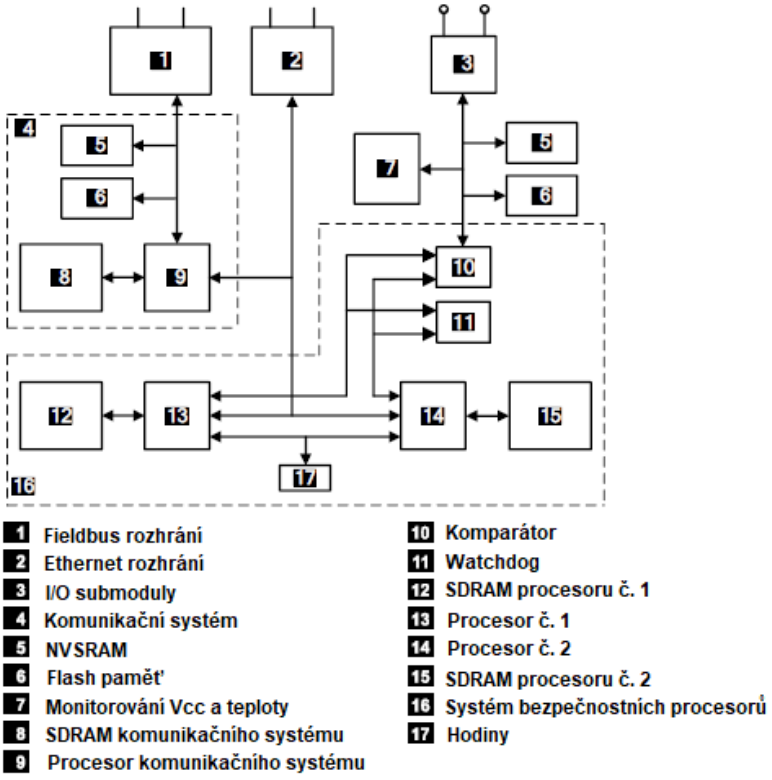
Analogové vstupy podporují měření napětí v rozmezí 0 až 10 V nebo proudu až do 20 mA. Tyto vstupy mají vysokou přesnost díky vysoké vzorkovací frekvenci, což je ideální pro aplikace vyžadující přesná měření.

Čítače jsou schopny zpracovávat vstupní frekvence až do 100 kHz a podporují různé režimy provozu, včetně dekódování Grayova kódu. Tyto režimy se dají nastavovat pomocí systémových proměnných ve vývojovém prostředí SILworX.

Komunikační rozhraní tohoto kontroléru využívá pro Ethernetové komunikaci 4 RJ-45 konektory a pro Fieldbus komunikaci až 3 konektory označené jako „FB“. Fieldbus konektor č. 3 je uzpůsoben pro sériovou komunikaci RS485 pro protokol Modbus nebo pro ComUserTask. [Odkaz F35]

5.4 Procesorový systém

Kontrolér HIMatrix F35-03 využívá procesorový systém verze 03. Tento procesorový systém tvoří dva synchronní mikroprocesory, jenž tvoří spadají do systému bezpečnostních procesorů (CPU modul), a jeden komunikační procesor patřící do komunikačního systému (COM modul). Schéma architektury tohoto procesorového systému je vyobrazeno na *obrázku 7*.



Obrázek 7: Architektura procesorového systému

Dva mikroprocesory řídí a kontrolují operace celého kontroléru, včetně vyhodnocování vstupů a výstupů, a zajišťují tak, že všechny bezpečnostní funkce a procesy jsou prováděny správně a spolehlivě. Zpracování dat ze vstupů a výstupů probíhá v reálném čase, což umožňuje rychlou reakci při změně jejich stavů. Oba mikroprocesory mají svou vlastní SDRAM a jejich běh je synchronizován. Synchronizace je kontrolována pomocí hardwarových komparátorů, které detekují jakoukoliv nesrovnalost nebo odchylku. V případě vyhodnocení chyby je kontrolér uveden do bezpečného stavu.

Komunikační procesor kontroluje a provádí bezpečnou výměnu dat mezi připojenými zařízeními, dalšími kontroléry nebo moduly, a řídí komunikaci s PADT. Zajišťuje, že data jsou přenášena bezpečně a spolehlivě.

5.4.1 Průběh procesů

Kontroléry využívá cyklický průběh procesů k zajištění konzistentního, spolehlivého a bezpečného provozu. Systém tak může nepřetržitě reagovat na změny ve vstupních podmínkách, které se s časem mění. Tento cyklický proces se skládá z několika hlavních fází, které jsou opakovaně prováděny v přesně stanoveném pořadí.

Každý cyklus začíná čtením vstupních dat ze senzorů a dalších vstupních zařízení. Následně se provádí zpracování nahraného uživatelského programu, kde se na základě vstupních dat a předem definovaných logických podmínek vypočítávají nové hodnoty pro výstupy. Po dokončení zpracování programu se nové hodnoty zapisují do výstupních zařízení, čímž se aktualizují stavy akčních členů a dalších komponent. Poslední fází je provádění diagnostických a komunikačních úloh, které zahrnují například výměnu dat s jinými systémy a provádění diagnostických testů.

5.5 Multitasking

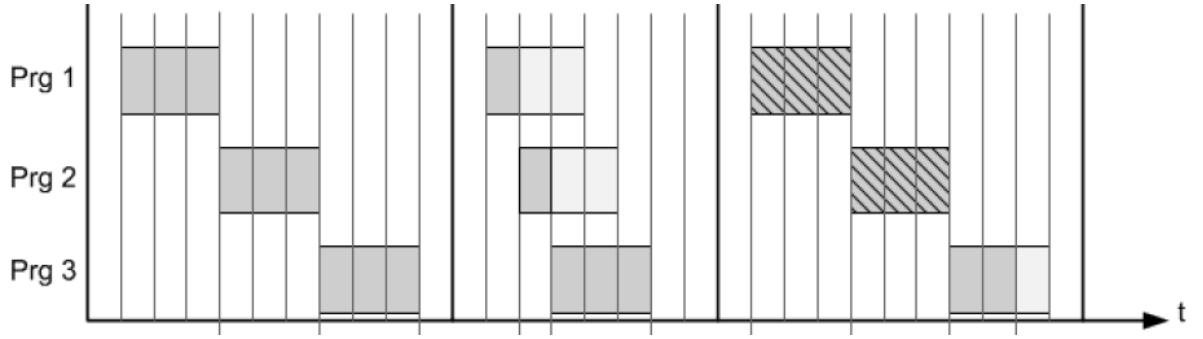
Multitasking je schopnost kontroléru HIMatrix F35 zpracovávat více programů současně. Tento přístup umožňuje rozdělit projekt na jednotlivé podfunkce, které mohou být nezávisle spuštěny a zastaveny.

F35 je schopen zpracovat až 32 programů v rámci procesorového modulu. Každý uživatelský program může mít jinou prioritu, což ovlivňuje pořadí jejich zpracování. Uživatelské programy s vyšší prioritou jsou zpracovávány před programy s nižší prioritou. Pokud mají programy stejnou prioritu, systém je zpracovává v rostoucím pořadí podle jejich ID. Tento systém řízení priorit zajišťuje, že kritické úlohy jsou zpracovávány efektivně, aniž by došlo ke zbytečným zpožděním méně důležitých úloh.

Pro kontrolér F35 existují tři různé operační módy multitaskingu, které umožňují uživatelům optimalizovat využití procesorového času podle specifických požadavků jejich aplikací. Všechny jednotlivé módy jsou detailně rozebrány v oficiální HIMA dokumentaci.

5.5.1 Mód 1

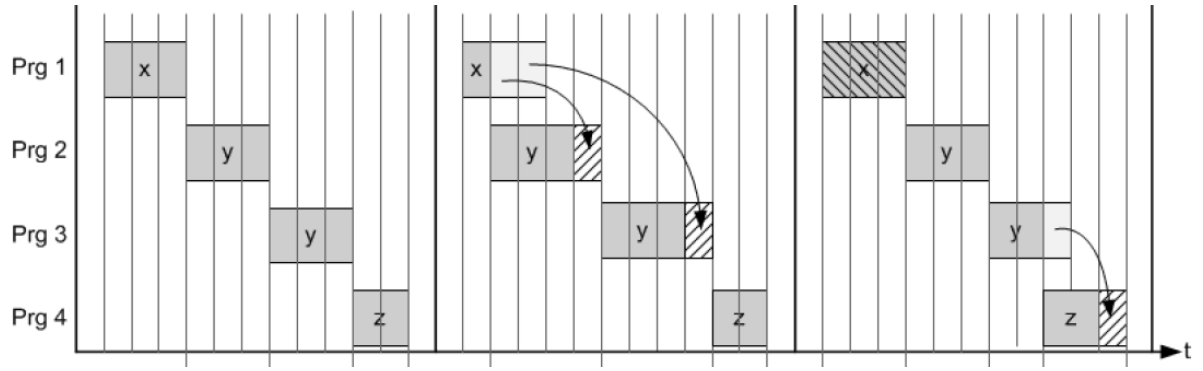
V módu 1 (*obrázek 8*) se systém snaží využít veškerý dostupný čas co nejefektivněji tím, že co nejdříve dokončí všechny úkoly. Jakmile je jeden uživatelský program kompletně zpracován, ihned se začne zpracovávat další program. Pokud má procesor po dokončení všech úloh stále čas k dispozici, tento čas nečeká na další cyklus, ale rovnou se pustí do zpracování úloh dalšího cyklu.



Obrázek 8: Multitasking mód 1

5.5.2 Mód 2

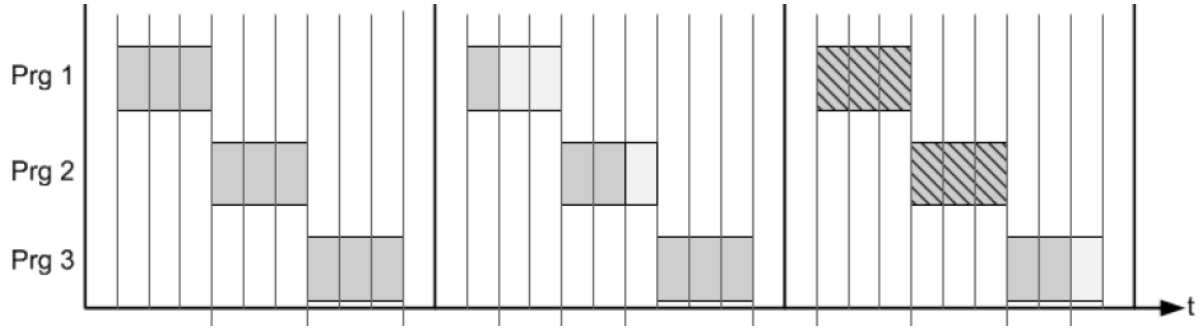
V módu 2 (*obrázek 9*) systém zajišťuje rovnováhu mezi efektivitou a stabilitou. Čas, který není využit programy s nižší prioritou, je přerozdělen mezi programy s vyšší prioritou. To znamená, že programy s vyšší prioritou dostávají více času na zpracování, což zajišťuje, že kritické úlohy jsou dokončeny včas.



Obrázek 9: Multitasking mód 2

5.5.3 Mód 3

V módu 3 (*obrázek 10*) systém čeká, dokud není dosažena maximální délka cyklu programu, a poté začne zpracovávat další program. Tento přístup zajišťuje, že všechny programy mají dostatek času k dokončení své práce, a poskytuje konzistentní a předvídatelný čas cyklů. Tento mód se dá použít k ověření, zda systém může zajistit správné provádění programu i v nejhorších podmínkách.



Obrázek 10: Multitasking mód 3

5.6 Výchozí nastavení HIMatrix F35

Výchozí nastavení je výrobcem určené nastavení daného zařízení, které přišlo z výroby, nebo pokud na zařízení byl proveden tovární reset.

U zařízení HIMatrix 35 výchozí nastavení znamená, že v paměti zařízení není uložen žádný uživatelský program ani nastavení, zařízení není nakonfigurované a nejsou nadefinované žádné proměnné, krom systémových proměnných. Tento stav by mělo zařízení mít vždy před každou novou aplikací, kdy je nutné systém znovu nakonfigurovat, naprogramovat a nastavit.

IP adresa CPU řady F35 je přednastavená na 192.168.0.99, maskou podsítě 255.255.252.0 a bránou 0.0.0.0, a defaultní IP adresa COM procesoru je 192.168.0.100 se stejnou maskou podsítě. Defaultně nastavené ID (SRS) zařízení F35 je 60000 a musí být vždy před každou aplikací změněno na unikátní číslo pro danou síť.

5.7 Tovární reset

Tovární reset kontroléru HIMatrix F35 se provádí následujícím způsobem.

Zařízení musí být před resetem vypnuté, respektive odpojené od napájení. Pro reset zařízení je nutné lokalizovat kulatý otvor, který se nachází na horním panelu, vedle Ethernet portů. V tomto otvoru se nachází tlačítko, které je nutné stlačit a držet jej stlačené po celou dobu restartu systému. Ke stlačení se použije vhodný úzký nástroj, který je nejlépe vyroben z nevodivého materiálu. Po stlačení a držení tlačítka je možné zařízení zapnout. Tlačítko se musí držet stále stlačené do dokončení inicializace systému, tedy po dobu cca 20 sekund. Po inicializaci systému je možné tlačítko pustit a systém by měl být uveden do defaultního nastavení. Kontrolní LED diody při defaultním nastavení by měly mít následující stavy:

- 24V DC – svítí zeleně
- RUN – bliká zeleně

- FAULT – bliká oranžově

5.8 Licence pro systém HIMatrix

Pro každý kontrolér řady HIMatrix je potřeba zvlášť dokoupit potřebné licence, které mohou být nutné pro přístup ke klíčovým funkcím daného zařízení. Dostupné licence je možno získat přes webové stránky firmy HIMA, a to po online registraci konkrétního zařízení a zaplacení poplatku za každou poptávanou licenci. Každá licence je vázaná pouze na konkrétní zařízení, a to prostřednictvím unikátního systémového ID.

Pokud kontrolér nemá žádnou platnou licenci, dojde na kontroléru HIMatrix ke stálému svícení ERROR LED světla.

5.8.1 SMR licence

Tato licence umožňuje přístup k následujícím systémovým funkcím:

- Záznam sekvence událostí (SOE)
- Multitasking
- Reload (opětovné načítání)

5.8.2 Licencování komunikačních protokolů

K použití komunikačních protokolů mohou být vyžadovány zvláštní licence. To se netýká následujících komunikačních protokolů, které jsou nastaveny jako defaultní:

- HIMA safeethernet Ethernet
- SNTP server/client

Ostatní komunikační protokoly mohou být používány bez licence po dobu 5000 provozních hodin, což se týká i RS485. Po uplynutí 5000 provozních hodin komunikace pokračuje, dokud není řadič zastaven. Poté nelze uživatelský program spustit bez platné licence pro komunikační protokoly použité v projektu (neplatná konfigurace).

6 Využití programovatelného PLC HIMatrix F35

V rámci konceptu Železnice 4.0 je možné programovatelné COTS PLC využít jako objektové kontroléry různých vnějších zabezpečovacích a funkčních prvků železniční infrastruktury. V konceptu jsou tak OC koncipovány do skupiny, které jsou vytvořeny pro jednotlivé vnější

prvky, resp. objekty, zvlášť dle jejich funkčních požadavků. Protože PLC systémy jsou poměrně drahé, v poměru na počet vstupů a výstupů, předpokládá se v konceptu Železnice 4.0 převážné použití tzv. „chytrých periferií“. Znamená to použití specificky vyvinutých elektronických řešení vnějších periferií (závory, výstražníky, návěstidla, případně přestavníky) s vlastními řídícími a diagnostickými funkcemi, komunikují

cí s objektovým kontrolérem datovou komunikací např. CANbus nebo RS485. Tato periferie musí být realizována na úrovni bezpečnosti SIL4, dle ČSN EN 50129.

Cílem této kapitoly je posouzení využití programovatelného PLC HIMatrix F35 v závislosti na stanovených funkčních požadavcích daných skupin OC, jenž byly stanoveny v bakalářské práci Daniela Chlebka. Konkrétně se jedná o tyto skupiny objektových kontrolérů:

- OCN
- OCP
- OCZ
- OCPN a OCEOV

6.1 OCN

Z výstupů práce D. Chlebka se předpokládá použití ovládání návěstních svítilen návěstidel pro koncept Železnice 4.0 v podobě LED technologie, realizované jako chytré periferie.

F35 disponuje až třemi Fieldbus konektory, které umožňují realizaci požadovaných rozhraní. Počet připojených návěstních svítilen musí být řešen až v konkrétní aplikaci. Lze však předpokládat, že bude možné realizovat až desítky periferií.

6.2 OCP

Přestavníky (ovládání výhybky) krom vlastního napájení vyžadují 6 digitálních výstupů, které by byly použity např. pro ovládání stykačů, softstartérů, relé a detektorů proudu, 2 digitální vstupy, které by sloužily pro dohledy, v případě výhybek pro vysoké rychlosti také dodatečné digitální vstupy pro kontrolní kontakty polohy výměnové části výhybky, a 1 analogový vstup k detekci proudu. F35 má 8 digitálních výstupů, 24 digitálních vstupů a 8 analogových vstupů. Do jednoho kontroléru F35 by tak bylo možné napojit pouze jeden přestavník.

K F35 by se však v tomto případě daly připojit až dva I/O moduly F2 DO 8 01 nebo jeden I/O modul F2 DO 16 01, jenž by vytvořily systém o počtu 24 digitálních výstupů, 24 digitálních

vstupů a 8 analogových vstupů. Do tohoto systému by tak bylo možné připojit ve výsledku až 3 přestavníky.

Přestavníky by bylo možné ovládat také jako chytré periferie, je však nutné zvážit ekonomickou efektivitu tohoto řešení.

6.3 OCZ

Rozhraní objektových kontrolérů pro ovládání přejezdů se bude lišit dle počtu a složení jednotlivých prvků. S ohledem na možný vysoký počet periferií (závory, výstražná světla) na PZZ se předpokládá využití „chytrých periferií“. U místního bodu přejezdu je nutné také uvažovat I/O moduly např. pro indikační a ovládací skříňku. F35 disponuje až třemi Fieldbus konektory, které umožňují realizaci požadovaných rozhraní. Pro toto rozhraní by měl postačovat jeden kontrolér F35.

6.4 OCPN a OCEOV

Počítače náprav a elektronické ohřevy výhybek budou samy o sobě tvořit objektové kontroléry, které budou připojeny přes switch přímo do optické sítě IZZ. U těchto aplikací není potřeba F35 zřizovat.

7 Vývojové prostředí SILworX

Vývojový software SILworX se používá k programování, konfiguraci a monitorování systémů produktové řady HIMatrix a dalších systémů jiných produktových řad firmy HIMA.

Všechny informace pro vytvoření této uživatelské příručky byly čerpány z manuálu firmy HIMA „SILworX First Steps Manual“.

Ve zmíněném manuálu se často objevuje zkratka PADT (Programming and debugging tool), kterou je označován počítač nebo laptop, který je opatřen softwarem SILworX, má platnou licenci a je určen k programování a konfiguraci systémů firmy HIMA.

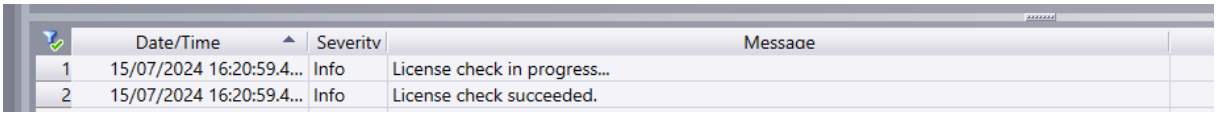
V této psané příručce je pro označení konkrétních funkcí a prvků prostředí SWX a OS Microsoft Windows používána *kurzíva*.

7.1 Licence SILworX

Pro správný chod, kompatibilitu a funkci softwaru SILworX je potřeba mít zakoupenou platnou licenci. Ta je vázaná buď na přenosné zařízení (USB flash disk) a je možné ji použít na více PC, nebo je vázaná na konkrétní PC, na kterém je potřeba licenci aktivovat přímo v SWX, dle instrukcí manuálu. Dále je pro SWX verze V11 a vyšší vyžadován OS Microsoft Windows 10.

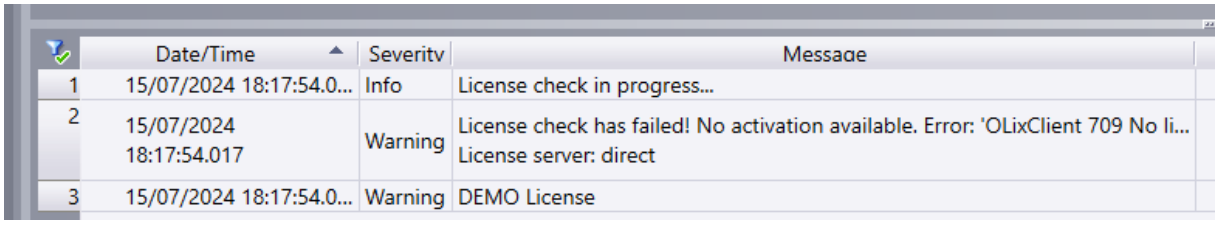
Ještě před spuštěním SWX je tedy potřeba do PC zasunout USB flash disk s platnou licencí, nebo mít aktivovanou licenci na konkrétním PC. Bez licence se spustí pouze DEMO verze SWX, ve které nejsou zpřístupněny potřebné klíčové funkce, a nelze uskutečnit online s pojení z žádným zařízením. Dalším aspektem DEMO verze je nekompatibilita s jakoukoliv jinou verzí a licencí SILworX. Při vytvoření projektu v DEMO verzi tak není možné otevřít tento projekt v plné verzi a naopak.

Zpráva o správném načtení platné licence by se měla zobrazit v logbooku ve spodní části UI. (viz obrázek 11).



Obrázek 11: Zpráva o načtení platné licence

Pokud by se licenci nepodařilo úspěšně ověřit, automaticky by došlo ke spuštění DEMO verze softwaru (viz obrázek 12).



	Date/Time	Severity	Message
1	15/07/2024 18:17:54.0...	Info	License check in progress...
2	15/07/2024 18:17:54.017	Warning	License check has failed! No activation available. Error: 'OLixClient 709 No li... License server: direct
3	15/07/2024 18:17:54.0...	Warning	DEMO License

Obrázek 12. Spuštění DEMO verze programu SILworX

7.1.1 Základní SILworX licence

Základní licence pro SWX je nabízená v následujících variantách:

- **Plná licence** – podporuje všechny řídicí systémy HIMA a pokrývá všechny požadavky
- **HIMatrix licence** – podporuje všechny systémy v řady HIMatrix F
- **Udržovací licence** – poskytuje přístup k datům všech řídicích systémů HIMA
- **Serverová licence** – centrální správa licencí SILworX pro připojené PADT

7.1.2 Volitelné SILworX licence

Volitelné SWX licence je možné propojit s jakoukoliv základní licencí. Umožňují rozšíření různých funkcí SWX, které nejsou v základní licenci dostupné. Jedná se o tyto licence:

- C-Code Functions block
- Safety Simulator
- Typical Import
- Smart Safety Test
- API
- COMPARATOR PLUS
- API NOGUI
- Plug-In
- Mulitfile project

7.1.3 Leasingové pakety SILworX

HIMA nabízí i možnost sjednání leasingových paketů, které vždy zahrnují základní plnou licenci SWX a k tomu různé kombinace volitelných licencí v závislosti na daném paketu. Leasingový paket je možné sjednat na pět let a platba za službu probíhá ročně. V aktuální době je možnost sjednat tyto pakety:

- SILworX Bronze
- SILworX Silber
- SILworX Gold
- SILworX Platinum

7.2 Instalace a odinstalace SILworX

Software SILworX může být instalován pouze na PC s operačním systémem Microsoft Windows. Lze jej nainstalovat z USB flash disku, nebo z jiného nosiče, na kterém je uložena složka s instalačním souborem. Složka by měla mít název „SILworX Vx.x.x“, kde x označuje verzi SWX. Aktuální verze k 15.7. 2024 je V14.0.0 R2908. Soubor je potřeba z nosiče uložit do konkrétního PC. Následně je možné v souboru otevřít složku „Installation“, ve které by se měl nacházet instalační soubor SILworX_setup.exe. Po otevření tohoto souboru se spustí instalace softwaru, kde je potřeba dodržovat zobrazované instalační instrukce.

SWX je defaultně nainstalován do adresáře C:\Program Files\HIMA\ SILworX_Vx.x.x, kde x označuje instalovanou verzi SWX.

Odinstalaci softwaru SILworX je možné provést přes *Ovládací panely* v sekci *Programy a funkce*.

7.3 Otevření a založení nového projektu

SWX se dá otevřít přímo z adresáře, pomocí vytvořeného zástupce na ploše, nebo pomocí zástupce umístěném v nabídce Start ve složce HIMA.

Po otevření SWX je možné založit nový projekt pomocí panelu nabídek (menu bar), a to konkrétně klikem na miniaturu, nebo po otevření nabídky *Project* a výběrem funkce *New...*. Dojde k zobrazení okna, ve kterém se vybere umístění souboru a jeho název.

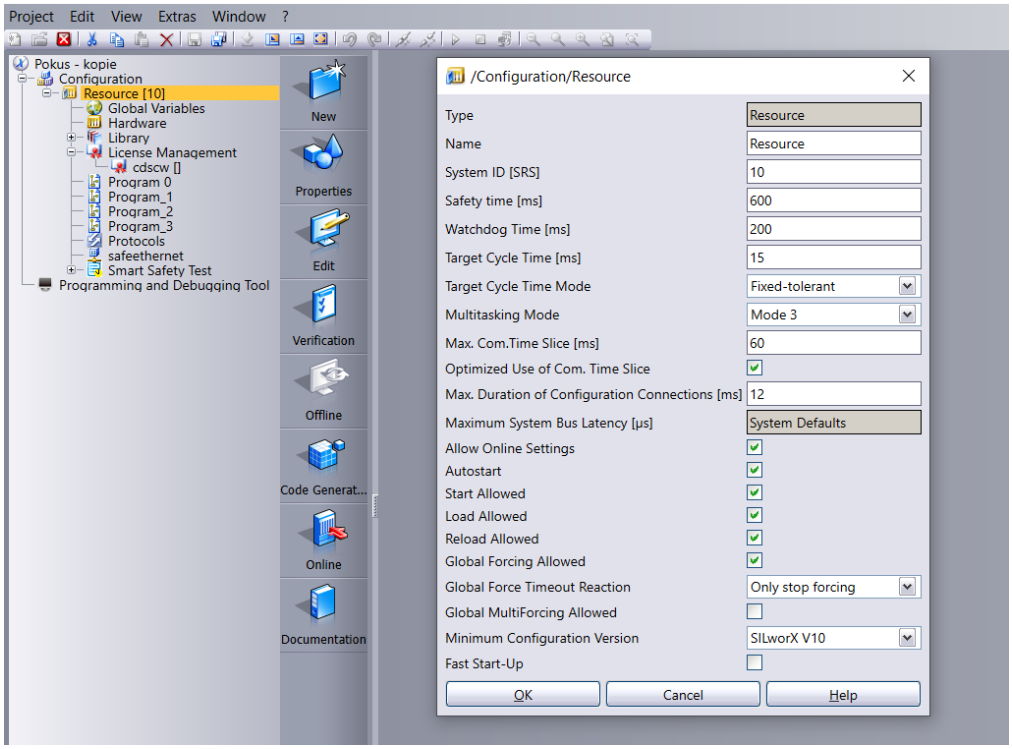
Již existující projekt je možné otevřít obdobným postupem, tedy pomocí panelu nabídek (menu bar), a klikem na miniaturu, nebo po otevření nabídky *Project* a výběrem funkce *Open...*.

7.4 Nastavení vlastností projektu

Důležitým korkem je nastavení vlastností, které budou ovlivňovat chod systému. Mezi nastavované vlastnosti patří ID zařízení, multitasking, globální nastavení hardwaru, nebo i různá časová omezení pro chod programů, cyklů a časovačů atd.

Prvek *Resource* představuje kontrolér, který zpracovává jeden nebo více programů. Tvoří také stěžejní prvek, pod který spadají všechny vlastnosti, programy, nastavení komunikace a hardwarové přiřazení. Je-li potřeba v rámci jedné aplikace zapojení více kontrolérů, lze tak v projektu vytvořit další *Resource*, a to pravým kliknutím na prvek *Configuration* a následným vybrání prvku *Resource* z nabídky.

Nastavení vlastností je zpřístupněno přes klik na prvek *Resource,* který se zobrazí ve stromové struktuře po rozkliknutí kmenového prvku *Confuguration.* Dále se zvolí možnost *Properties*, která je vyobrazena na akčním panelu vedle stromové struktury. Následně dojde k zobrazení okna vlastností, které lze upravovat, viz obrázek 13 .



Obrázek 13: Vlastnosti projektu

Mezi nejdůležitější nastavované vlastnosti patří:

- System ID – Unikátní identifikační číslo kontroléru (uvádí se číslo mezi 1 a 65535)

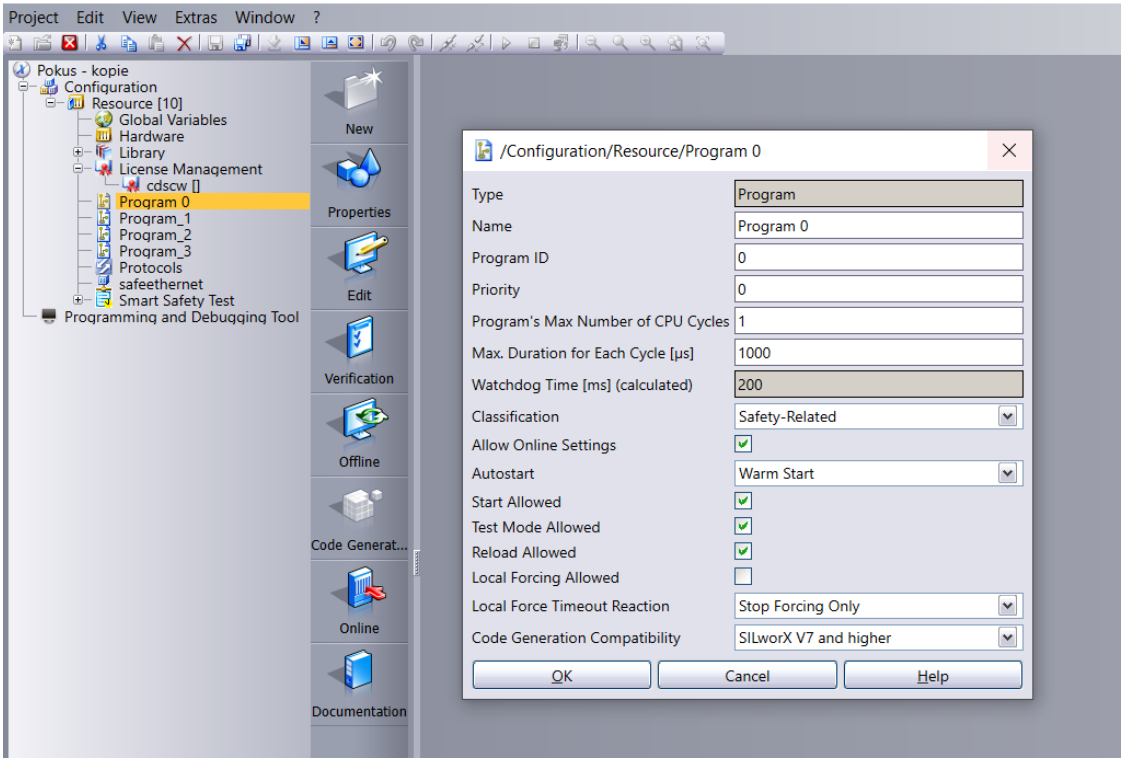
- Safety time – Maximální doba, po kterou může systém čekat na kritické bezpečnostní operace; pokud tato doba uplyne bez očekávané reakce, systém se přepne do bezpečného stavu
- Watchdog Time – Časový interval pro kontrolu správné funkce systému; pokud systém neprovede očekávaný úkon v tomto intervalu, systém se resetuje nebo se přepne do bezpečného stavu
- Target Cycle Time – Celkový čas pro provedení jednoho cyklu operací
- Multitasking Mode – Výběr jednoho ze tří možných módů multitaskingu
- Allow Online Settings – Možnost změny konfigurace systému online bez nutnosti restartu

7.5 Nastavení vlastností programů

U každého programu zvlášť je nutné nastavení programových vlastností. Mezi nastavované vlastnosti patří název programu, ID programu, priorita programu, počet cyklů, nebo i různá časová omezení apod.

Nastavení vlastností je zpřístupněno přes klik na konkrétní *Program* ve stromové struktuře. Dále se zvolí možnost *Properties*, která je vyobrazena na akčním panelu vedle stromové struktury. Následně dojde k zobrazení okna vlastností, které lze upravovat, viz obrázek 14 .

Je zejména důležité vyplnit pole názvu *Name*, pole *Program ID,* kde se uvede unikátní číslo programu*,* a pole *Priority*, které udává prioritu před ostatními programy. Do tohoto pole se vyplní číslo 0 pro nejvyšší prioritu a vyšší číslo pro menší prioritu.

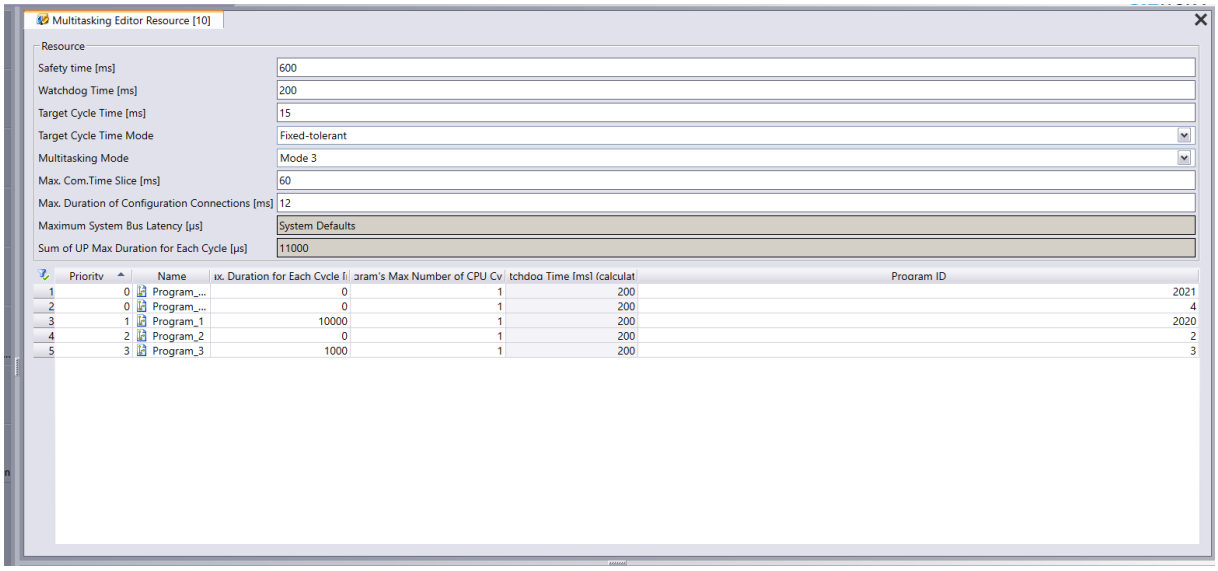


Obrázek 14: Vlastnosti programu

7.6 Multitasking editor

Multitasking editor (*obrázek 15*) slouží k přehlednému zobrazení posloupnosti vykonání jednotlivých programů v rámci multitaskingu. Umožňuje také změnu některých vlastností systému, které mají na multitasking nějaký efekt.

Editor se otevře dvojklikem na prvek *Resource*, přičemž dojde k otevření nové záložky. V horní části jsou vypsány vlastnosti systému, které se mohou měnit. Ve spodní části je zobrazen seznam programů s jejich nastavenými vlastnostmi, jenž lze také měnit skrze tento editor. Programy by měly být uvedeny v pořadí, ve kterém budou dle nastavených vlastností zpracovány.



Obrázek 15: Multitasking editor

7.7 Definování proměnných

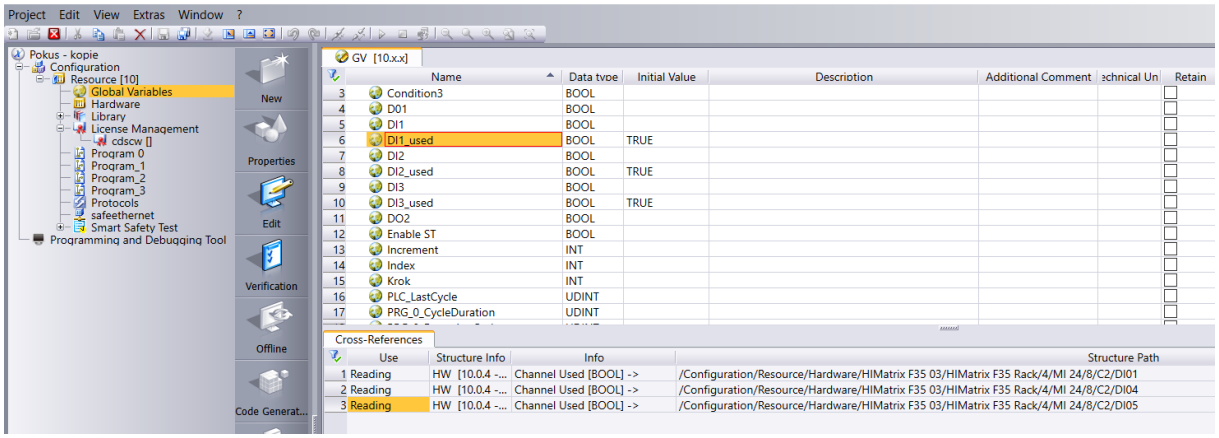
Proměnné v SWX slouží k ukládání a přenosu dat mezi různými částmi programu a hardwarovými komponentami systému HIMatrix. Jsou klíčové pro definování vstupů, výstupů a interních stavů, což umožňuje efektivní řízení a monitorování procesů. V SWX jsou tři důležité skupiny proměnných, které jsou popsány níže.

7.7.1 Globální proměnné

Globální proměnné je potřeba vytvořit pro následující případy:

- Hardware – k uložení a následnému zpracování hodnot ze vstupů a výstupů
- Komunikace – k výměně dat mezi jednotlivými kontroléry pomocí různých protokolů
- Programování – k výměně dat mezi jednotlivými programy nebo funkčními bloky v rámci uživatelského programu

Pro jejich vytvoření nebo úpravu je ve stromové struktuře potřeba otevřít záložku *Global Variables* (*obrázek 16*). V záložce jsou parametry proměnných zobrazeny formou tabulky. Novou globální proměnnou lze vložit pravým klikem myši do pracovní plochy, a následným vybrání *New Global Variable.* Jednotlivé parametry proměnných lze změnit kliknutím do vybrané buňky a jejich přepsáním, nebo vybráním předdefinovaného parametru z nabídky. Dále je možné tabulku s globálními proměnnými exportovat nebo importovat z/do CSV souboru. Tato možnost je dostupná při pravém kliku do pracovní plochy a zvolení možnosti *Import Table Contetnt from CSV* nebo *Save Table Contetnt as CSV.*



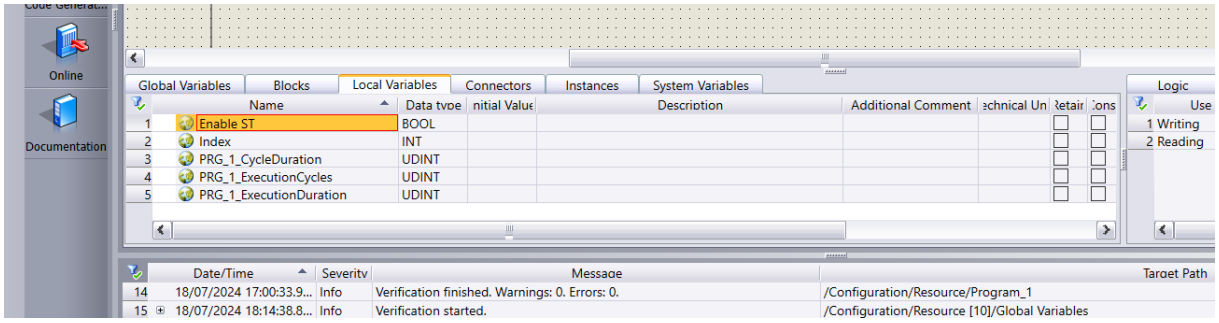
Obrázek 16: Záložka globálních proměnných

7.7.2 Lokální proměnné

Lokální proměnné jsou dostupné pouze v editačním okně programu a logického bloku.

Tyto proměnné lze užít jen v rámci jednoho programu a nelze je použít např. pro zápis hodnot na vstupech a výstupech zařízení, a jiné akce, u kterých je nutné použití globálních proměnných.

Vzhledem k tomu, že proces vytváření a úpravy lokálních proměnných se provádí sice obdobným způsobem jako u globálních proměnných, ale za to v záložce *Local Variables* (*obrázek 17*) při rozkliknutí konkrétního programu, je lepší si tyto proměnné definovat jako globální a využít je v daném programu jako lokální. Tento způsob využití globálních proměnných není oficiální, ale je osvědčený.

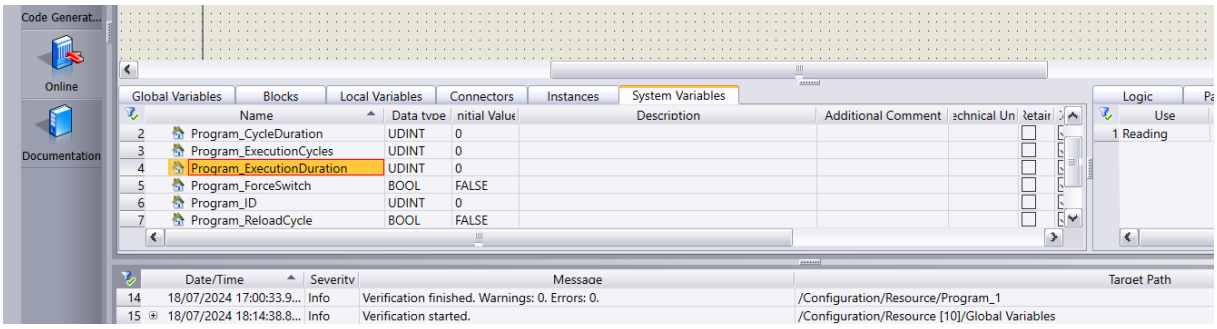


Obrázek 17: Záložka lokálních proměnných

7.7.3 Systémové proměnné

Proměnné tohoto typu jsou předem definované a jejich parametry nelze upravovat. Dají se však, stejně jako ostatní typy proměnných, přetažením vložit do programu a využít je ke čtení jejich aktuálních hodnot, které se v průběhu času mění. Záložka *System Variables*

(*obrázek 18*) je téže dostupná v editačním okně každého programu, jako je tomu u lokálních proměnných.

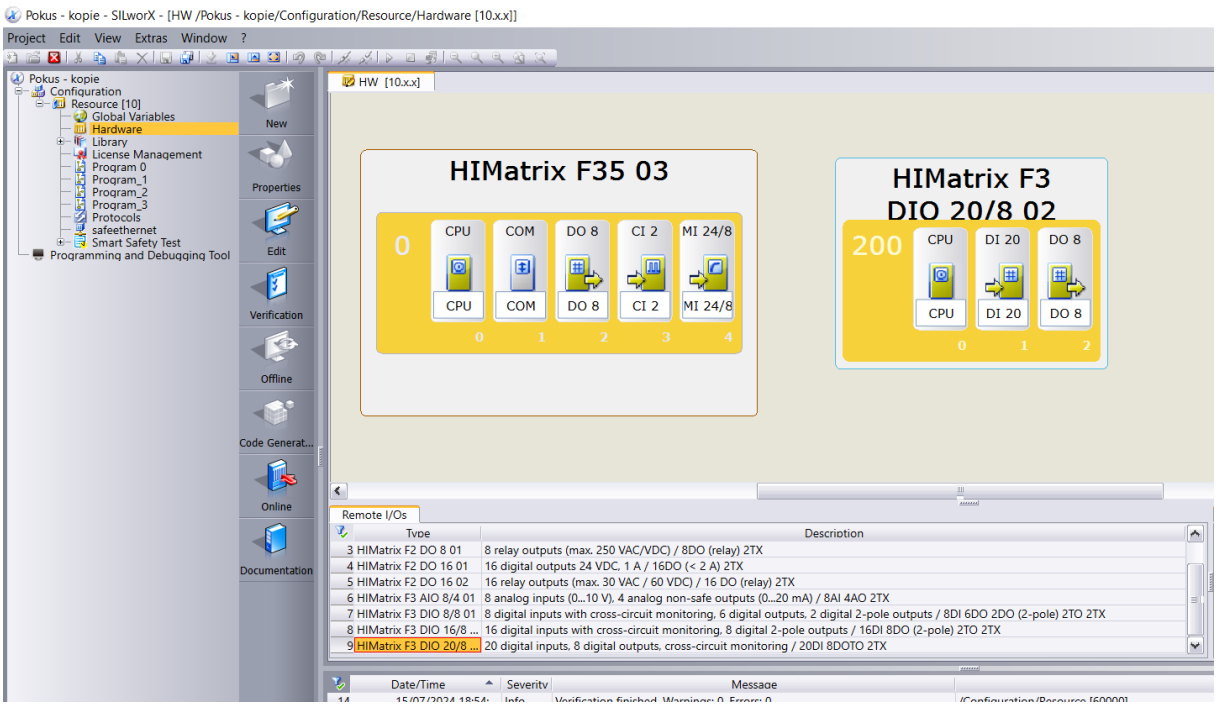


Obrázek 18: Záložka systémových proměnných

7.8 Vložení a konfigurace zařízení HIMatrix

SILworX umožňuje uživateli přidávat a nastavovat konkrétní zařízení HIMatrix a vzdálené I/O jednotky, čímž zajišťuje správnou konfiguraci systému.

Ve stromové struktuře SWX se dvojklikem otevře záložka *Hardware*. Zobrazí se okno, ve kterém se vybere konkrétní zařízení, které se bude konfigurovat (např. HIMatrix F35 03). Vybrané zařízení se zobrazí na ploše záložky. (viz obrázek 19)



Obrázek 19: Záložka Hardware

7.8.1 Vložení dalších modulů

Pokud jsou k zařízení připojené další konfigurovatelné moduly, např. dálkově ovládané I/O, je možné tyto moduly vybrat z prostředního okna *Remote I/Os*, a operací drag-and-drop je přesunout na plochu záložky. U modulárních systémů HIMatrix F60, je také nutné stejným způsobem vybrat a modifikovat konkrétní obsazení jednotlivých modulů systému. Po dokončení konfigurace je před dalším pokračováním nutné projekt uložit.

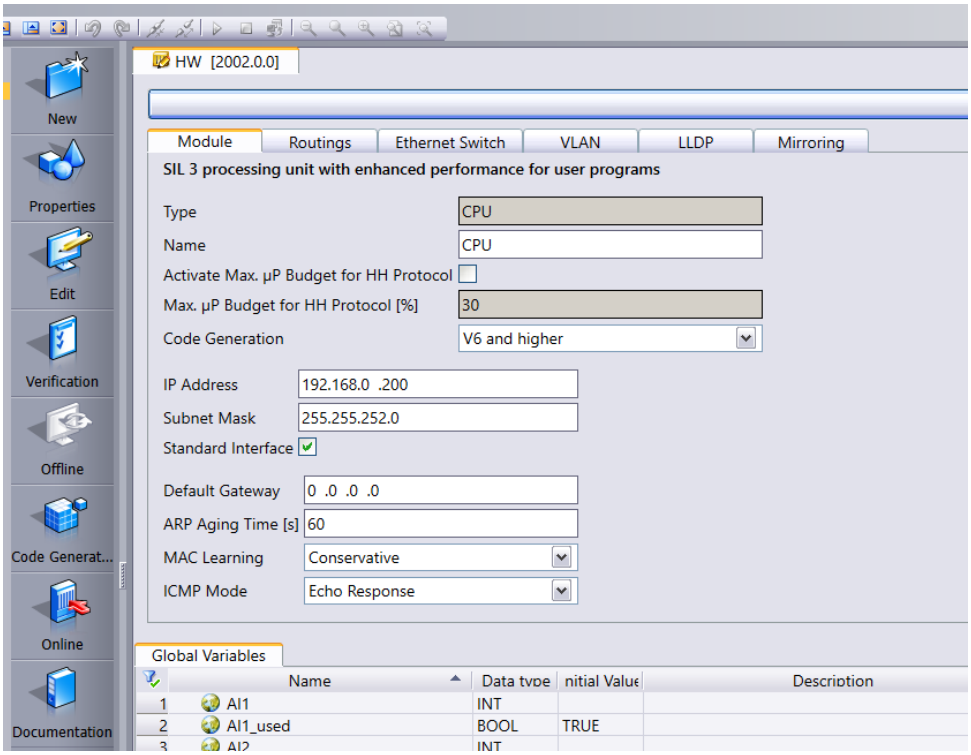
Přidané moduly lze z plochy odstranit označením a následným stiskem klávesy Delete na klávesnici.

7.8.2 Nastavení IP adresy zařízení

Pro zajištění komunikace s PADT, jinými zdroji nebo vzdálenými I/O moduly musí být procesorovému modulu i komunikačnímu modulu přiřazena IP adresa, která je jedinečná v celé síti. Obecně platí, že pro IP adresy by neměly být používány výchozí hodnoty.

Dvojklikem na ikonu procesorového modulu CPU se zobrazí záložka *Module* s nastavením modulu viz obrázek 20. Do políčka *IP Adress* je možné napsat zvolenou unikátní IP adresu pro tento modul, např. 192.168.0.200. Dále lze aktivovat funkci *Standard Interface*, která slouží k automatickému předvyplnění IP adresy při Online přihlašování k systému. Ostatní parametry by měly být dle doporučení výrobce ponechány bez změny.

Stejným postupem se změní i IP adresa komunikačního modulu.



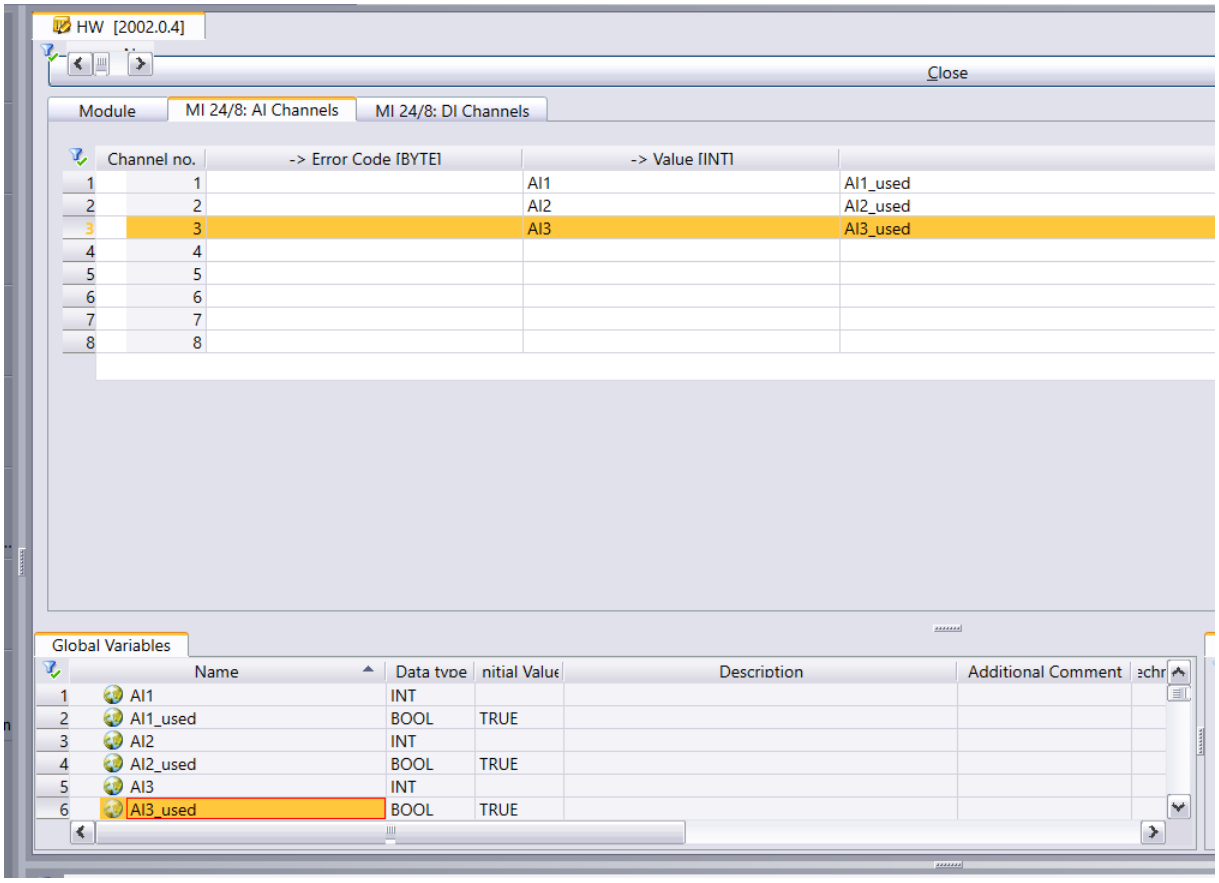
Obrázek 20: Nastavení IP adresy

7.8.3 Přiřazení proměnných k hardwaru

Pro použití fyzických vstupních a výstupních hodnot hardwaru v logice musí být vstupy a výstupy propojeny s globální proměnnou odpovídajícího datového typu.

Před přiřazením proměnných k určitému hardwaru se v záložce *Hardware* klikne na modul vstupů či výstupů a v záložce *Module* se nastaví potřené parametry pro tento modul. U modulu smíšených vstupů MI 24/8 je například vhodné nastavit režim analogových vstupů na *FS1000* nebo *FS2000*, podle toho, zda hodnoty vstupů budou dosahovat rozmezí 0-1000 nebo 0-2000. Následně je možné pokračovat zvolením libovolné záložky, např. *MI 24/8: AI Channels*, kde se na konkrétní kanál do políčka *Value [INT]* dají přetáhnout vytvořené globální proměnné z okna *Global Variables.*

Všechny kanály navíc musí být explicitně aktivovány. U konkrétního kanálu je potřeba vytvořit zvláštní proměnnou typu *BOOL*, která bude mít výchozí hodnotu nastavenou na TRUE. Tato proměnná se pak přiřadí k určenému kanálu do políčka *Channel Used [BOOL].*



Obrázek 21: Nastavení analogových vstupů

7.9 Vytváření programů

Uživatelské programy musí obsahovat logiku potřebnou k řízení a monitorování procesů ve spojení s jedním nebo více kontroléry. Maximální počet programů pro daný kontrolér je 32. Jejich počet může být ale omezen dle jejich složitosti, resp. velikosti a volné paměti kontroléru.

Všechny vytvořené jednotlivé programy tvoří pro daný kontrolér jeden rámcový uživatelský program, jenž v systému běží v cyklech. Pořadí a proces vykonávání jednotlivých programů udává jejich nastavení priority a nastavení režimu multitaskingu systému.

Program lze vytvořit pravým klikem na *Resource,* zvolením *New* a následným výběrem *Program.*

7.10 Programování v SILworX

K programování systémů v SWX lze použít funkční blokové diagramy (FBD), sekvenční funkční diagramy (SFC), strukturovaný text (ST) a v rámci možností lze použít i programovací jazyk C++, a to v podobě speciálních funkčních bloků.

Jelikož je FBD výchozím programovacím jazykem pro uživatelské programy, je nutné pro použití ostatních programovacích jazyků použít FBD Editor. Všechny podporované programovací jazyky jsou implementovány do speciálních funkčních bloků, které se následně vkládají do programu přes FBD Editor.

7.10.1 Programování FBD

Programování pomocí FBD je základní možností k programování logiky uživatelských programů pro systémy HIMatrix.

SILworX nabízí mnoho standardních funkcí a funkčních bloků, které lze použít k vytváření programů.

Ve stromové struktuře se zvolí konkrétní *Program*, a klikne se na funkci *Edit* na akčním panelu. Dojde k otevření nové programové záložky a zobrazení FBD Editoru.

FBD Editor je v zásadě rozdělen do následujících oblastí: pracovní plocha, navigační panel a panel objektů.

Pracovní plocha je určena pro vkládání a propojování funkčních bloků, proměnných a dalších objektů. Ve výchozím nastavení je rozdělena do devíti pracovních podoken, přičemž zpracování programu se provádí zleva doprava a postupně odshora dolů.

Navigační panel zobrazuje oddálený pohled na pracovní plochu a slouží tak pro jednoduchou navigaci a orientaci u rozsáhlých programů.

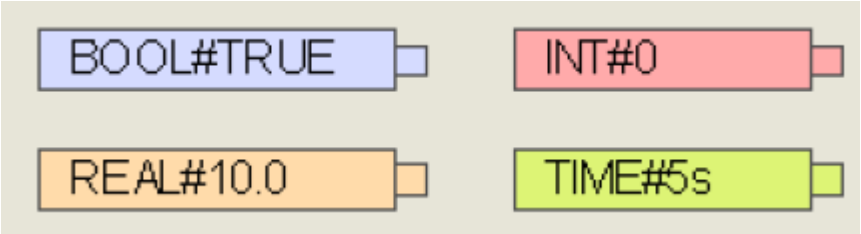
Na panelu objektů jsou vyobrazeny všechny skupiny objektů, které lze přetažením vložit do pracovní plochy. Každý z objektů se na pracovní ploše zobrazí jinak v závislosti na jeho názvu, funkci a počtu vstupů a výstupů. Některé objekty lze rozšiřovat a tím i zvyšovat počty vstupů a výstupů v závislosti na funkci objektu. Vstupy a výstupy objektu jsou označeny čtverci, a slouží k připojení propojovací čáry, kterou lze jednotlivé objekty spojovat a vytvářet tak vzájemné závislosti. Při najetí na vstup/výstup by se měl místo kursoru objevit černý kříž, který umožňuje kreslit propojovací čáry, a to přidržením levého tlačítka a potáhnutím myší.

Funkční bloky se nachází na panelu objektů v záložce *Blocks*. Každý z funkčních bloků je označen názvem, typem knihovny a u některých bloků je i popisný symbol.

Mezi další vkládané objekty spadají např. globální proměnné, lokální proměnné, systémové proměnné, nebo třeba pomocné konektory a OLT pole.

Pomocné konektory je možné vložit přes pravé tlačítko myši a výběrem *Create FBD Input Connector*. Slouží k propojení různých částí programu a k zajištění přenosu hodnot mezi funkčními bloky, což je vhodné pro přehlednou organizaci objektů u složitých programů. Namísto pomocných konektorů lze samozřejmě vytvořit globální nebo lokální proměnnou, která bude mít stejnou funkci.

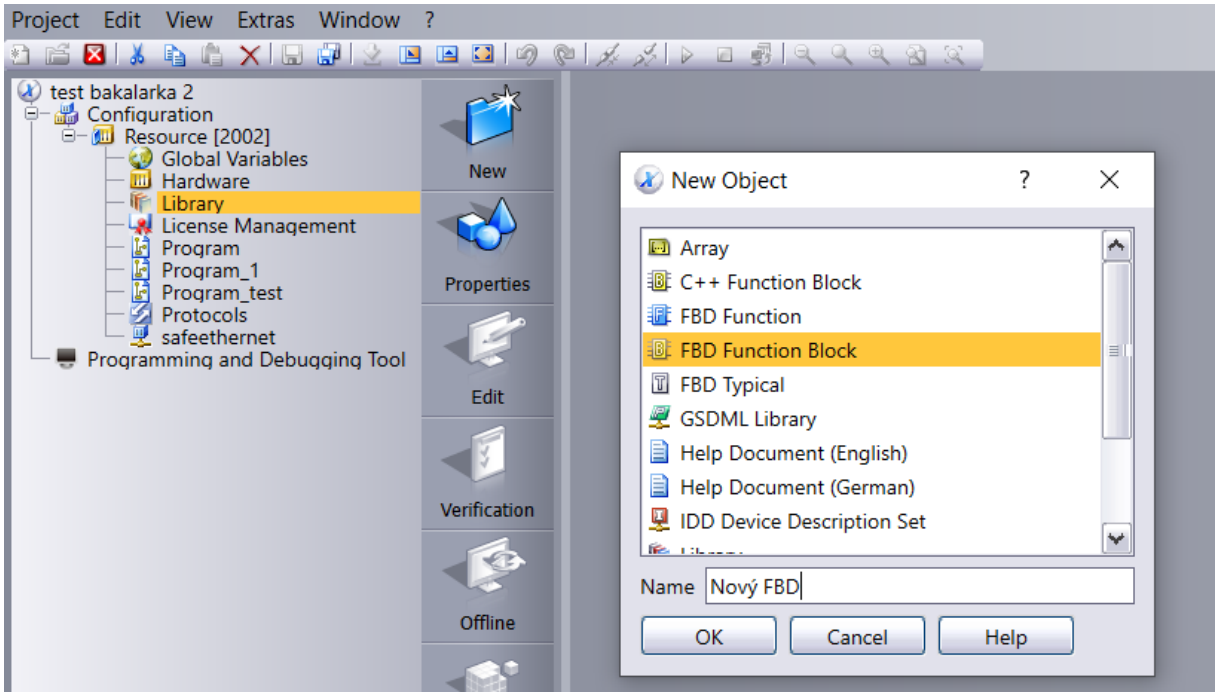
Klikem pravého tlačítka myši do pracovní plochy jsou dále přístupné další užitečné funkce, jako vložení komentářového pole nebo hodnotového pole, do kterého je možné zapsat libovolnou hodnotu typu BOOL, INT, REAL nebo TIME. Pole se automaticky dle zvoleného typu přepne a změní svou barvu. Modrou barvou jsou označeny prvky typu BOOL, červeně prvky INT, oranžově prvky REAL a žlutě zbarvené jsou prvky typu TIME. Správné zadání pro hodnotové pole je vyobrazeno na obrázku 22.



Obrázek 22: Hodnotová pole

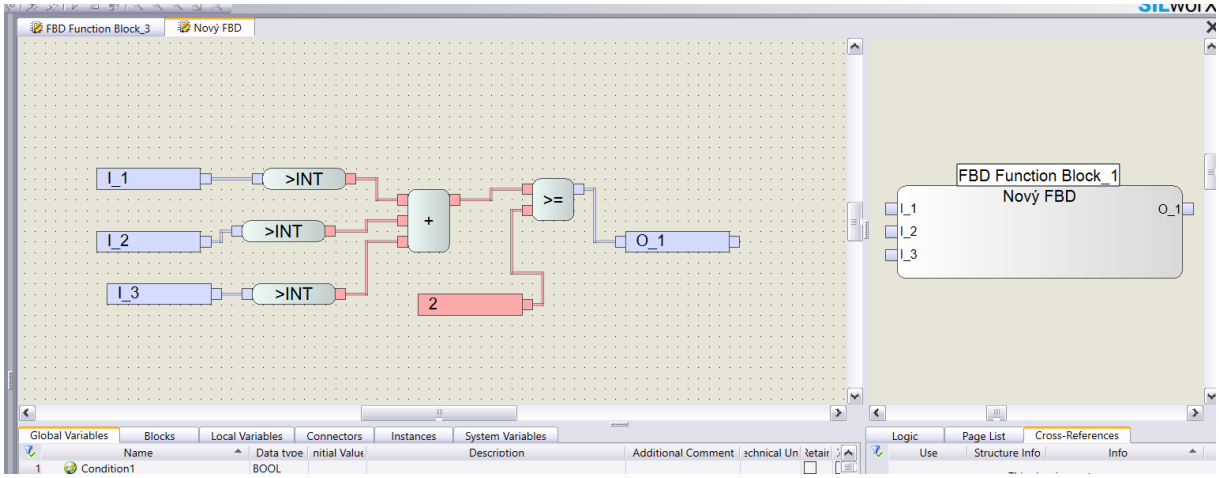
Další užitečnou funkcí jsou OLT pole, která slouží pro zobrazení aktuálních hodnot proměnných a jiných objektů vložených do programu. Tyto pole se dají vytvořit pro každý vložený objekt s nezapojeným výstupem, a to pravým klikem na zvolený výstup a výběrem možnosti *OLT field*. Dojde k vytvoření pole, které lze levým klikem myši vložit na libovolné místo, přičemž dojde k automatickému spojení s objektem. OLT pole zobrazuje hodnoty pouze v offline simulaci, nebo při zobrazení online průběhu programu.

Logické segmenty a různé programové operace je možno seskupit do uživatelsky definovaných funkčních bloků a použít je ve více programech. Lze tak docílit větší efektivity programu a celkového zjednodušení programovacího procesu. Nový funkční blok, jako i jiné programové funkce, se vytvoří přes pravý klik na prvek *Library* ve stromové struktuře, následným výběrem funkce *New* a dále vybráním *FBD Function Block* (viz ***obrázek 23***). Dojde k vytvoření nového prvku s umístěním pod knihovnou projektu ve stromové struktuře. Pravým klikem na tento prvek a vybráním *Properties* lze upravit jeho vlastnosti. Prvek lze následným dvojklikem otevřít a editovat jej stejným způsobem, jako samotný program pomocí FBD programování. Vedle pracovní plochy je vizualizována podoba nového FBD. Nový FBD se po uložení automaticky objeví v seznamu funkčních bloků *Blocks* u programů, a lze jej pak vložit do pracovní plochy přetažením, jako každý jiný prvek.



Obrázek 23: Vytvoření nového prvku knihovny projektu

Na *obrázku 24* je vyobrazen ukázkový uživatelsky definovaný FB, který je koncipovaný jako komparátor. V tomto případě data ze vstupů převádí na celé čísla, jenž následně sečte a porovnává, zda je součet větší nebo rovný dvěma. Pokud ano, tak je výstupní stav změněn z FALSE na TRUE.



Obrázek 24: Pracovní a vizualizační plocha při vytváření nového FBD bloku

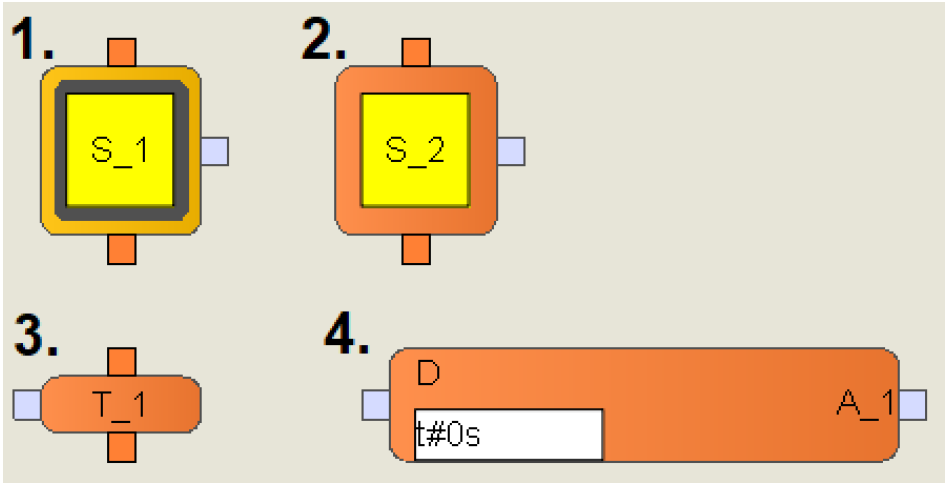
7.10.2 Programování SFC

SFC (Sequential Function Chart) programování se v SWX používá k vytváření sekvenčních řídících procesů, které jsou rozděleny do jednotlivých kroků a přechodů.

Pro vytvoření SFC programu se postupuje stejným způsobem jako při vytváření programů s FBD. Rozdílné je pouze použití určitých funkčních bloků určených pro SFC. Tyto bloky jsou dostupné v FBD Editoru v záložce *Blocks*, jsou označeny rozdílnou ikonou a ve sloupci *Library Type* mají uvedeno *SFC Objects*. Objekty stačí ze záložky přesunout na pracovní plochu a dále je možné pokračovat dle pravidel SFC programování.

Na *obrázku 25* jsou vyobrazeny všechny dostupné SFC objekty, a to konkrétně:

1. Initstep
2. Step
3. Transition
4. Action



Obrázek 25: SFC objekty v SILworX

7.10.3 Programování ST

ST (Structured Text) programování se v SILworX používá k vytváření složitých řídicích logik pomocí textového programování. Programování s využitím ST je možné přes speciální funkční blok, který se následně vloží do FBD Editoru.

Funkční ST blok lze vytvořit obdobným způsobem jako nový funkční blok (viz *kapitola 7.10.1*), tedy definováním nového typu bloku ve stromové struktuře přes prvek *Library*. Při výběru nového prvku se tentokrát zvolí možnost *ST Function Block*.

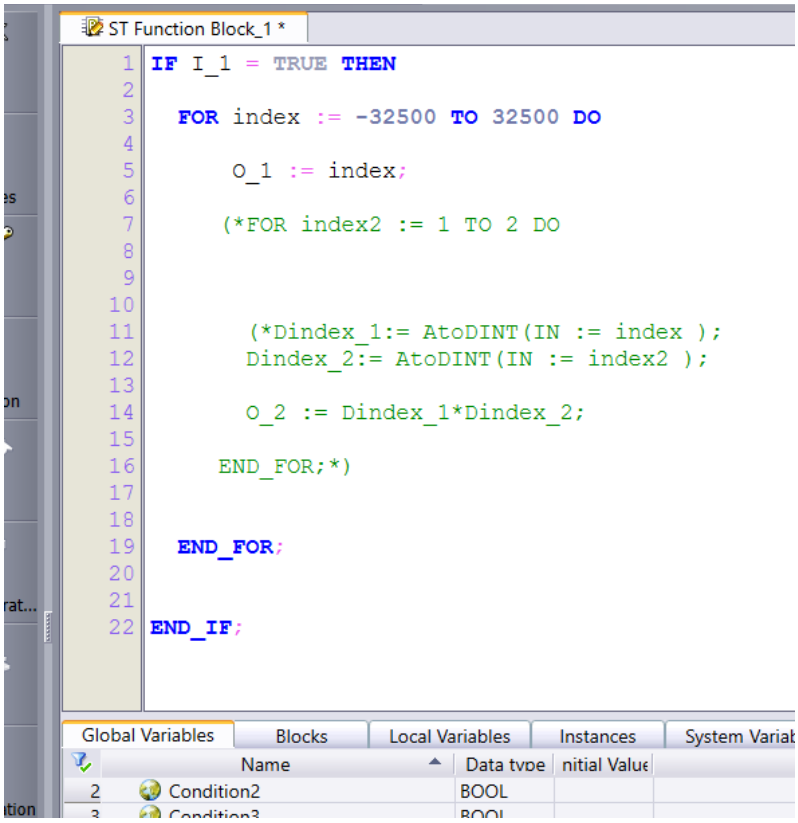
Po vytvoření lze opět nastavit potřebné vlastnosti tohoto bloku a následně jej otevřít. Dále je nutné vytvoření potřebných proměnných, které se budou v bloku vyskytovat, zejména vstupy a výstupy.

Až poté je možné zahájit programování ST bloku dle pravidel pro ST. Programování probíhá přímo v SWX psaním kódu do pracovní plochy.

Po uložení ST bloku dojde k jeho zobrazení v záložce *Blocks* v FBD Editoru a lze jej přetáhnout na pracovní plochu, kde ho je možné zakomponovat do programu.

ST umožňuje používat příkazy jako jsou *IF, THEN, ELSE, FOR, WHILE, CASE* apod. To umožňuje vytvářet složitější logické podmínky a cykly. Zápis funkce se provádí ve formě strukturovaného textu s použitím metod a způsobů vyššího programovacího jazyka včetně deklarací, podmínek, knihoven atd. Pokud by při vytváření funkce v ST bylo potřeba použití nějakého funkčního bloku, lze jej jednoduše přetáhnout ze záložky *Blocks* do pracovní plochy. Následně dojde k jeho převedení do textové podoby. Takto lze postupovat i při komponování jiného objektu z panelu objektů.

Příklad funkce psané ve strukturovaném textu je vyobrazen na *obrázku 26*.



Obrázek 26: Příklad funkce psané v ST

7.10.4 Programování C++

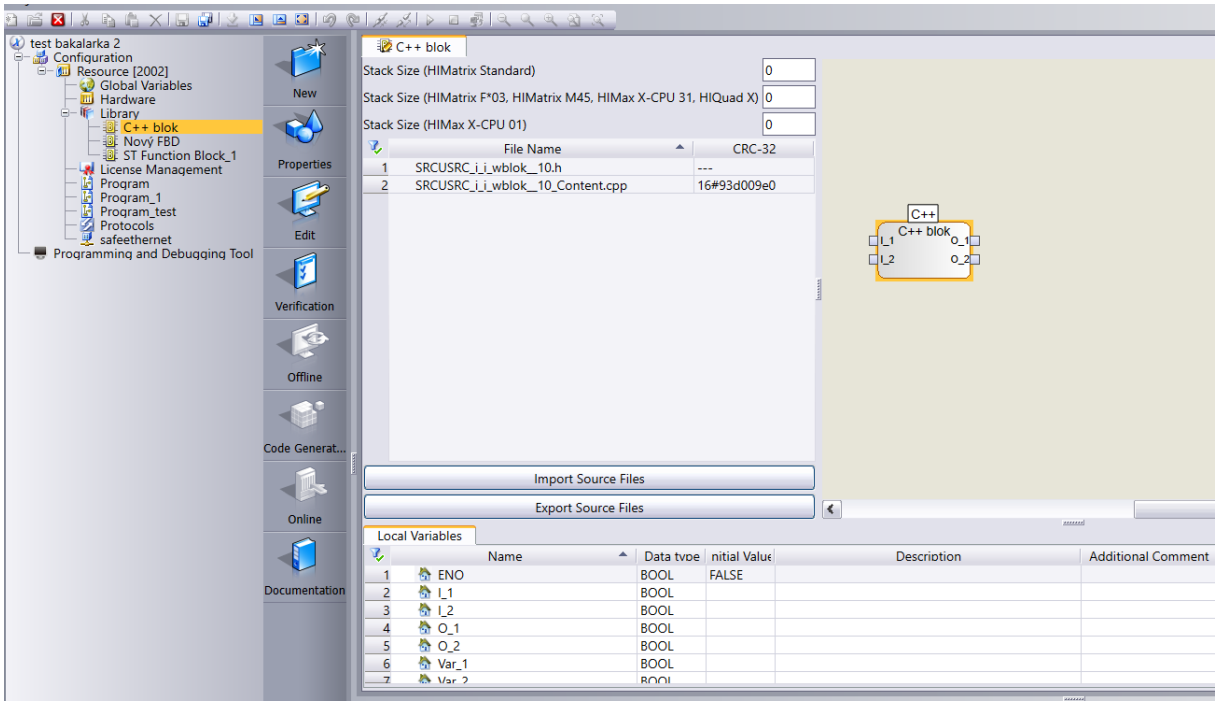
C++ je výkonný programovací jazyk, který umožňuje vytvářet optimalizovaný a efektivní kód, který je vhodné použít pro složité výpočty a algoritmy. Programování v jazyce C++ je, jako

v minulé podkapitole, možné přes speciální funkční blok, který se následně vloží do FBD Editoru.

Funkční C++ blok lze vytvořit ve stromové struktuře přes prvek *Library*. Při výběru nového prvku se ale zvolí možnost *C++ Function Block*.

Po jeho vytvoření lze nastavit potřebné vlastnosti tohoto bloku, jako například maximální velikost grafického zobrazení bloku, nebo zvolení patřičné přístupové varianty (Normal – strukturu FB lze zobrazit a editovat, Read-Only – lze zobrazit strukturu FB a nelze ho editovat, Know-How Protection – strukturu FB nelze zobrazit ani editovat).

Následně lze vytvořený blok otevřít, přičemž dojde k zobrazení záložky C++ bloku, ve které jsou uvedeny názvy vygenerovaných souborů pro C++, které slouží jako programovací soubory viz *obrázek 27*. Jedná se konkrétně o hlavní soubor s příponou .cpp a o hlavičkový soubor s příponou .h.



Obrázek 27: Záložka C++ funkčního bloku

Dalším krokem je vytvoření potřebných proměnných, které se budou v bloku vyskytovat, zejména vstupy a výstupy. Počet nadefinovaných vstupů a výstupů mj. také udává velikost grafického zobrazení FB v FBD Editoru.

Pro každý C++ FB je defaultní paměťová velikost nastavena na 4 kB. Pokud C++ FB bude vyžadovat větší velikost, je nutné tento parametr nastavit v horní části záložky. Velikostní

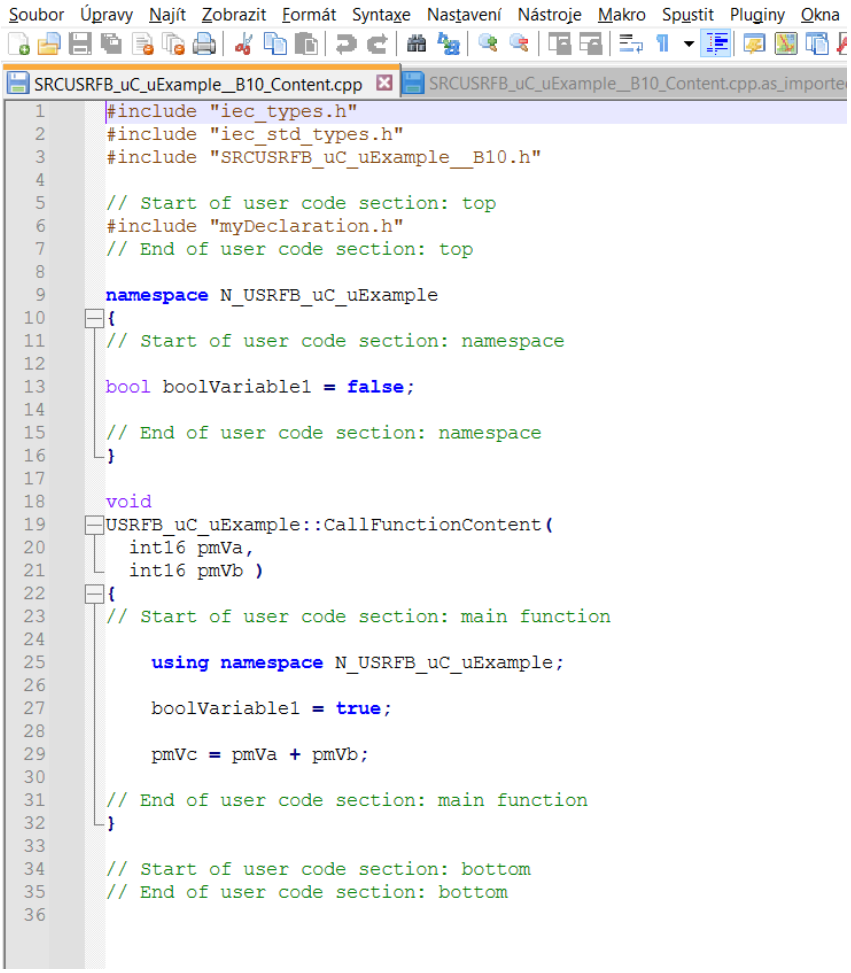
parametr se zapisuje v Bytech a jeho rozmezí se pohybuje od 0 do 1 048 064 Bytů, kde 0 je defaultní nastavení. Pro HIMatrix F35 se vyplňuje políčko *Stack Size (HIMatrix Standard).*

Pro úpravu a naprogramování těchto souborů je potřeba jejich export do zvolené složky na PC, a to kliknutím na tlačítko Export Source Files.

Po jejich exportaci do zvolené složky je soubory možné otevřít v libovolném C++ programovacím softwaru (např. Notepad++).

Souborům není možné měnit název, jelikož jsou vázány na název funkčního C++ bloku v SWX.

Dále je možné zahájit editaci a programování zmíněných souborů dle pravidel pro programovací jazyk C++. V rámci .cpp souboru je mj. možné přidat další uživatelské hlavičkové soubory .h, pokud je to nutné pro správnou funkci programu. Příklad jednoduchého C++ kódu včetně jeho správné struktury je vyobrazen na *obrázku 28*.



Obrázek 28: Příklad C++ kódu v softwaru Notepad++

Po dokončení programování je možné upravené soubory, spolu s případnými uživatelskými hlavičkovými soubory, importovat zpět do SWX, a to kliknutím na tlačítko Import Source Data a vybráním složky. Při výběru konkrétní složky se v ní uložené soubory nezobrazí, protože jde pouze o nastavení zdrojové složky. SILworX příslušné soubory automaticky rozpozná a dojde k jejich importaci.

Po úspěšném importu se C++ blok uloží, přičemž dojde k jeho zobrazení v záložce *Blocks* v FBD Editoru, ze které jej lze přetáhnout na pracovní plochu, kde ho je možné zakomponovat do programu jako funkční blok.

Je potřeba mít na paměti, že se na programování s využitím tohoto jazyka v SWX vážou určitá specifická pravidla a omezení, která jsou detailně vysvětlena v oficiálním manuálu SILworX určenému této problematice. zdroj: SILworX_C-FBS_Manual_PU00009642.PDF

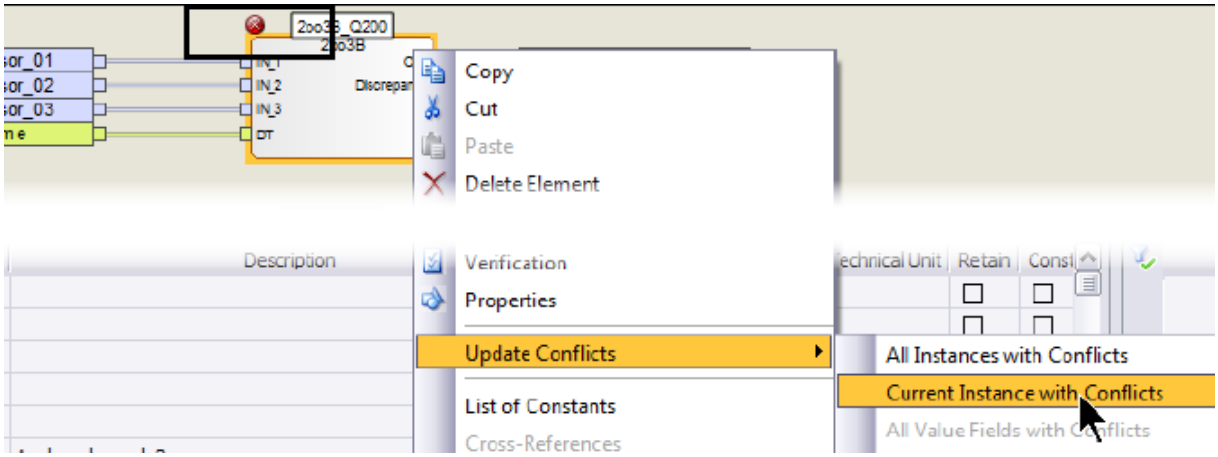
Doporučuje se používat pouze podporované jazykové prvky C++ pro zajištění kompatibility a bezpečnosti aplikací:

- Konstanty s podporovanými datovými typy
- Lokální proměnné s podporovanými datovými typy
- C++ globální proměnné s podporovanými datovými typy
- Literály
- Komentáře
- Deklarace funkcí a volání funkcí
- If..else
- switch
- while, do..while
- for
- sizeof
- assignments
- return
- break
- continue
- extern
- #include
- #ifdef, #if .. #elif .. #else .. #endif
- #define
- #undef (pouze pro samostatně definované prvky)

7.11 Řešení konfliktů

Při změně rozhraní u funkčních bloků nebo u hodnotových polí může dojít ke konfliktům. Konflikty jsou vyznačeny červenou ikonou s křížkem.

Konflikty lze vyřešit kliknutím pravého tlačítka myši na konfliktní objekt (*obrázek 29*), kde se dále zvolí možnost *Update Conflicts* a následně se vybere *Current Instance with Conflicts* nebo *All Instances with Conflicts*. Je možné že se propojovací čáry mezi objekty budou muset překreslit.



Obrázek 29: Proces řešení konfliktů

7.12 Offline simulace

K ověření správné funkce naprogramovaného programu a k simulaci jeho různých stavů je možné použít offline simulaci, kterou SWX disponuje. Není tak potřeba fyzické spojení s reálným systémem.

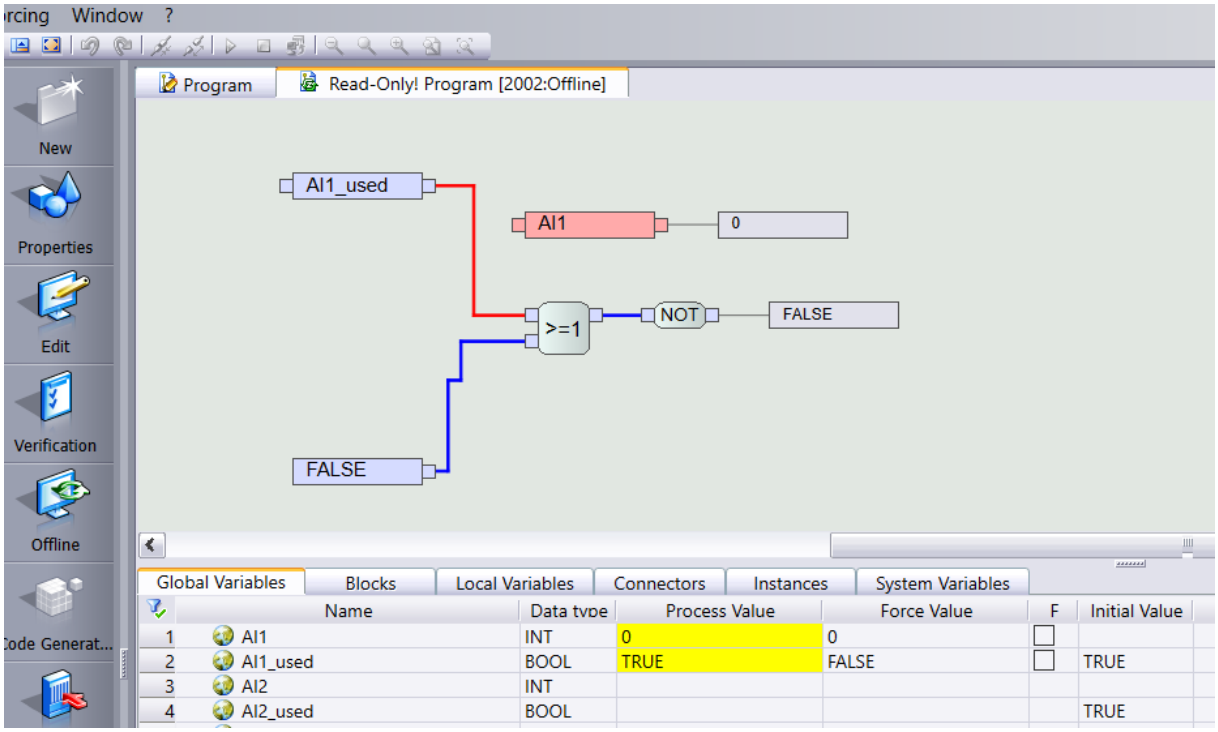
Pro vygenerování kódu offline simulace se ve stromové struktuře vyberte program, pro který se bude offline simulace spouštět, např. *Program_1*. Dále se na akčním panelu vybere funkce Offline a v dialogovém okně se vybere *OK*. Generátor kódu začne generovat kód pouze pro logiku vybraného programu.

Pokud generování kódu offline simulace selže, zobrazí se v logbooku varování „Warning“ nebo chyba „Error“. Při více chybových hláškách najednou se zobrazí jen jednořádkový záznam, který je možné rozvinout symbolem +, a zobrazit tak jednotlivé popisy a kódy chyb.

Pokud byl kód vygenerován bez chyb, logika programu se otevře jako nová záložka offline simulace.

Pro spuštění offline simulace je nutné v horním nabídkovém panelu vybrat *Online, Programs, Start Program (Cold Start with Test Mode Option).* Následně je offline simulace aktivní.

Aktivní proměnné budou mít během simulace v objektovém okně označený atribut *Process Value* žlutou barvou viz *obrázek 30*.

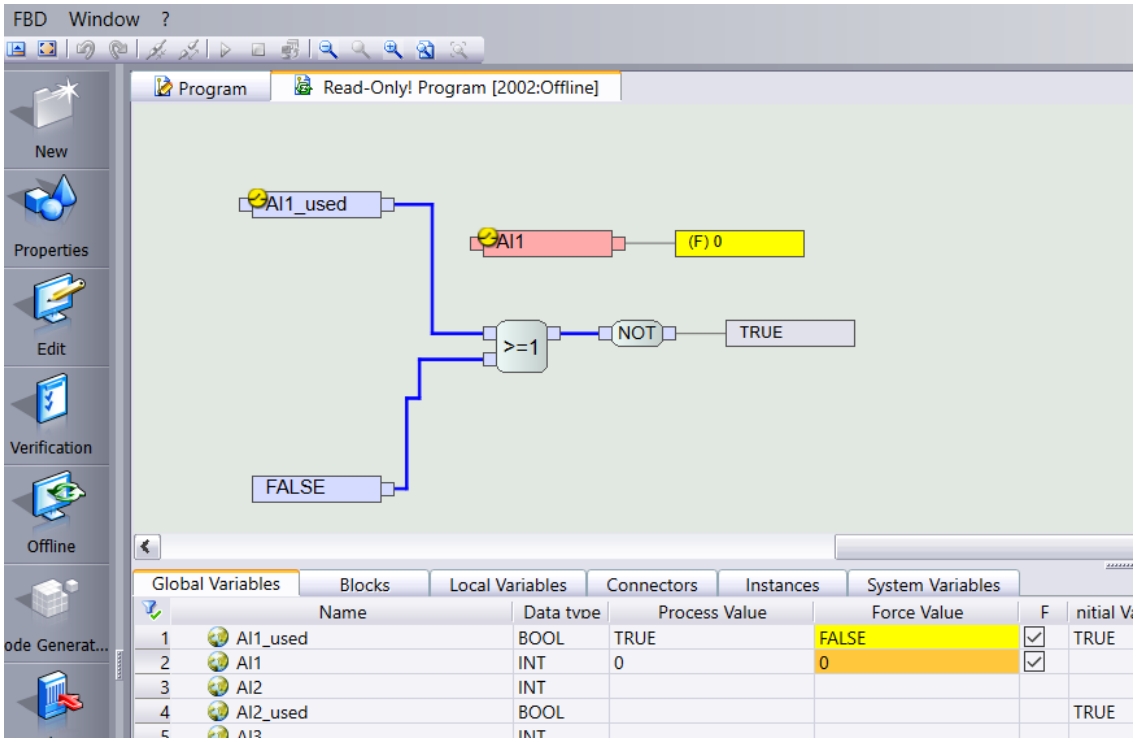


Obrázek 30: Záložka offline simulace

Pro změnu hodnot u proměnných během simulace lze využít nucené změny hodnoty, neboli forcing (*obrázek 31*). Toho lze docílit přes rozkliknutí zvolené proměnné v objektovém okně, následnou úpravou hodnoty v sekci *Force Value* a zaškrtnutím políčka *F*, kterým se vynucení aktivuje.

V simulaci je nucená změna hodnoty indikována následovně:

- na levé straně, nad blokem proměnných, se zobrazí ikona žlutého přepínače.
- barva vytvořeného OLT pole se změní z šedé na žlutou a před čtenou hodnotou je doplněn o písmeno F



Obrázek 31: Aktivní forcing

Simulaci lze ukončit prostým zavřením záložky offline simulace.

7.13 Generování kódu uživatelského programu

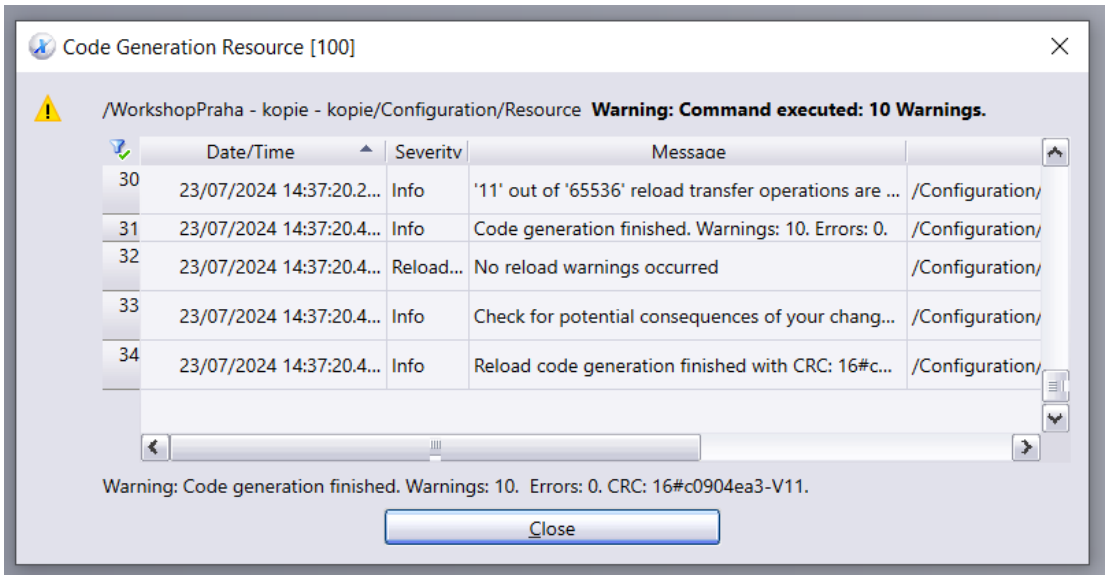
Před nahráním uživatelského programu do kontroléru musí být vygenerován jeho speciální kód. Generování kódu ověřuje nastavení konfigurace, syntaxi logiky a převádí data SILworX do strojově čitelného kódu.

Pro provedení generování kódu se ve stromové struktuře klikne pravým tlačítkem myši na *Resource*, pro který má být kód generován, a z kontextového menu se vyberte možnost *Code Generation*. Dojde k zobrazení výběrového okna, kde se dají aktivovat následující dvě funkce:

- CRC Comparison – slouží k cyklické kontrole redundance programového kódu, tato možnost by dle výrobce měla být vždy zvolená
- Prepare Reload – používá se k přípravě systému na aktualizaci uživatelského programu bez přerušení jeho běhu, tato možnost se objeví pokud program byl již v minulosti nahrán do systému a pokud se předpokládá, že zařízení bude v režimu RUN

Zobrazí se dialogové okno s průběhem generování kódu.

Po dokončení generování kódu se může zobrazit okno s chybami nebo varováními, které se během generování vyskytly viz *obrázek 32*.



Obrázek 32: Okno s varovným seznamem

Pokud došlo k zobrazení varování a při generování nedošlo k chybě, lze takto vygenerovaný programový kód použít.

Pokud jsou v projektu zjištěny chyby, nelze takto vygenerovaný programový kód použít. Zjištěné chyby je třeba odstranit ručně.

Cyklická kontrola redundance (CRC) je metoda detekce chyb používaná k ověřování integrity dat. Po každém vygenerování se ke kódu přiřadí unikátní CRC číslo, které reprezentuje kontrolní součet vytvořeného kódu. Při každém dalším generování stejného kódu se ověří jeho integrita a CRC číslo, které musí zůstat stejné.

Pro úspěšné vygenerování kódu je nutné proces generování provést alespoň dvakrát!

Až po zobrazení následující hlášky v logbooku je zaručena platnost kódu: *The CRC comparison from the dual code generation was successful. The generated code is valid.*

7.14 Propojení systému HIMatrix s PC

Propojení systému HIMatrix s PC je nutné pro jejich vzájemné online spojení.

Propojení zařízení HIMatrix s PC se realizuje pomocí Ethernet kabelu. Jeden konec kabelu se zasune do vybraného Ethernet portu zařízení, druhý konec se zasune do Ethernet portu PC. Následovně je možné modul zapnout připojením ke zdroji elektrické energie, dle jeho

technických specifikací. Při zapnutí modulu a navázání komunikace s PC se na využitém portu modulu rozsvítí zelená LED dioda signalizující aktivní připojení.

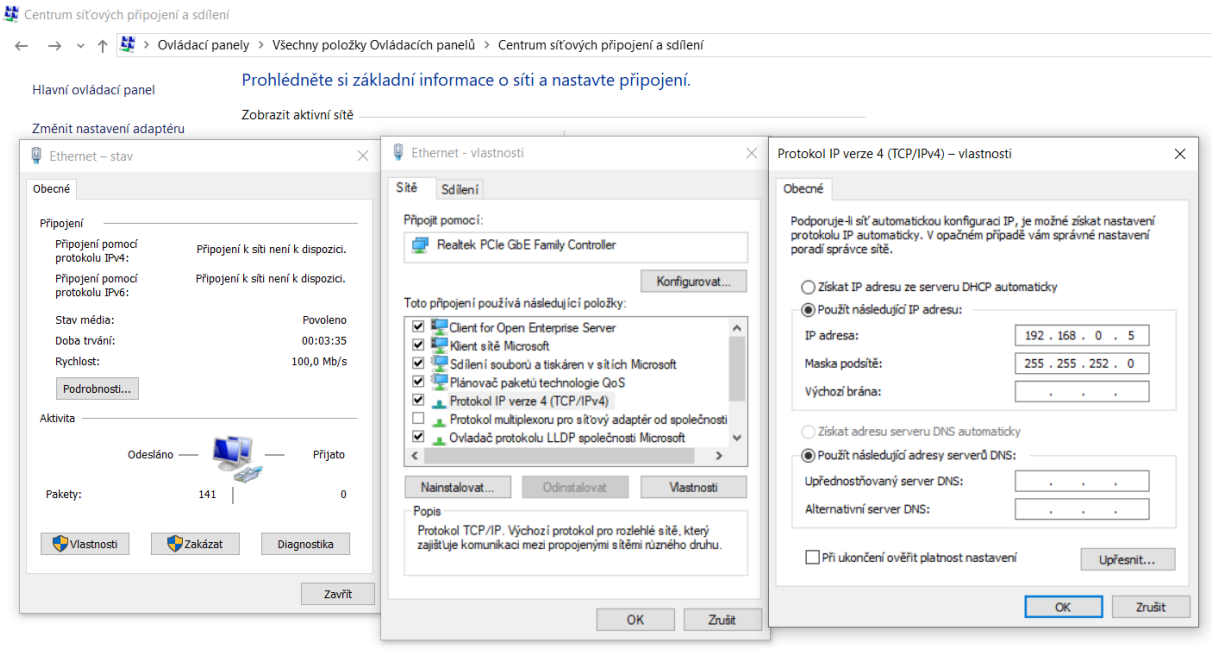
7.15 Online spojení se systémem HIMatrix

Online spojení je důležité pro přímý přenos aktuálních dat, informací a parametrů ze zařízení do prostředí SILworX na PC. Toto spojení také umožňuje provádění změn v nastavení, nahrání konfigurací, uživatelského programu atd.

7.15.1 Nastavení statické IP adresy PC

Pro úspěšnou komunikaci mezi HIMatrix modulem a PC je nezbytné zajistit, aby obě zařízení byly ve stejném síťovém rozsahu. Pokud zapojujeme PC přímo s modulem, bude nutné si na PC nastavit statickou IP adresu se stejnou maskou podsítě, který používá systém HIMatrix (*obrázek 33*).

Toho lze docílit nastavením v Ovládacích panelech OS Windows, konkrétně v sekci *Síť a internet*. Dále se otevře *Centrum síťových* připojení a sdílení, kde se zvolí *Změnit nastavení adaptéru*. Zde se pravým tlačítkem myši klikne na správný Ethernet port, do kterého je modul připojen a vybereme možnost *Vlastnosti*, která je dostupná po přihlášení správce PC. V seznamu zobrazených položek se rozklikne položka *Protokol IP verze 4 (TCP/IPv4)* a dojde k otevření jejich vlastností. Zde je možné změnit automaticky přidělenou IP adresu na manuálně zadanou (např. 192.168.0.5), stejně jako masku podsítě (např. 255.255.252.0).

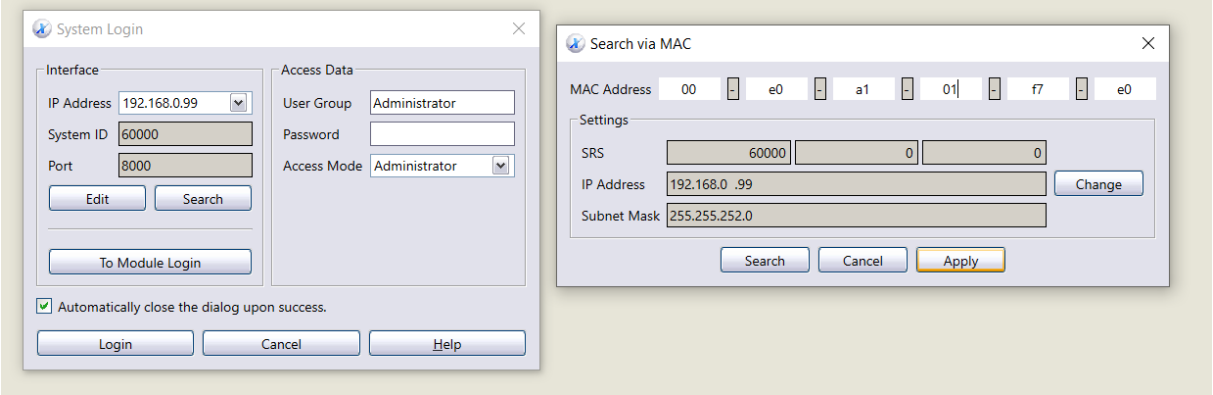


Obrázek 33: Nastavení IP adresy u OS Windows

7.15.2 Navázání spojení v SILworX

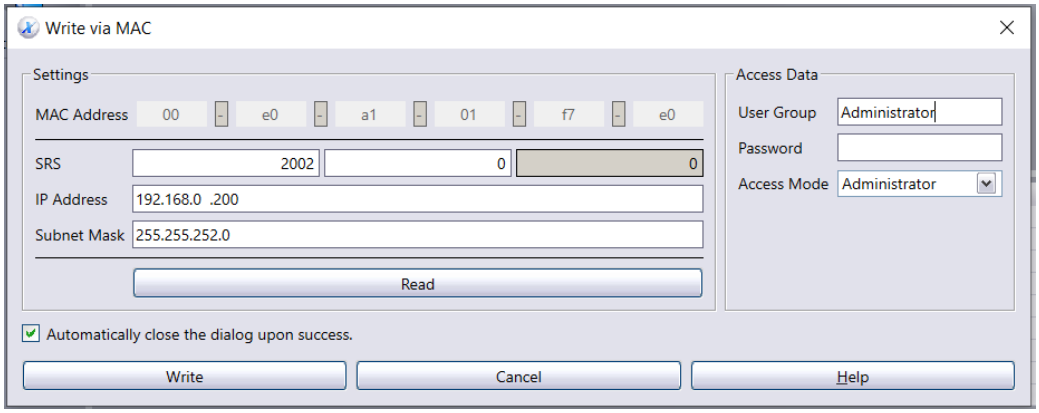
Pro zajištění plné funkcionality a správné konfigurace systému HIMatrix je nezbytné navázat online spojení pomocí softwaru SILworX. Tento proces umožňuje přístup k aktuálním nastavením zařízení a jejich úpravu přímo z prostředí SILworX, čímž se zjednodušuje správa a diagnostika systému.

Ve stromové struktuře otevřeme záložku *Hardware,* pro zobrazení konfigurace systému, a klikneme na ikonu *Online*. Otevře se okno *System login,* ve které se klikne na tlačítko *Search* (*obrázek 34*). Následně se zobrazí okno *Search via MAC*, do kterého se doplní poslední tři části MAC adresy připojovaného zařízení, které jsou uvedené na jeho krytu (např. 00-E0-A1-01-F7-E0). Dále se klikne na tlačítko *Search* a všechna pole v sekci *Settings* by se měla automaticky vyplnit. Vyplněné parametry jsou aktuálně nastavené parametry podsítě na straně systému.



Obrázek 34: Okna System Login a Search via MAC s defaultním nastavení

Pokud se systém konfiguruje nově, pokračuje se stiskem *Change*, přičemž se otevře další okno *Write via MAC* (*obrázek 35*). Zde se změní všechny parametry tak, aby se shodovali s nastavenými parametry projektu. Dále je nutné vyplnit přihlašovací údaje v sekci *Access Data*, kde je defaultně nastavený přístup pro administrátora bez hesla. Pomocí klávesové zkratky “Ctrl+A“ se přihlašovací údaje sami předvyplní. Pokračuje se stisknutím tlačítka *Write*, přičemž by mělo dojít k uložení nastavení do připojeného zařízení.



Obrázek 35: Přepis původní IP adresy zařízení

Pokud je předchozí krok splněný, nebo systém byl již konfigurován, je možné potvrdit předvyplněná data tlačítkem *Apply*, což uloží aktuální parametry do sekce *Interface*. Dále je nutné vyplnit přihlašovací údaje v sekci *Access Data*, kde je defaultně nastavený přístup pro administrátora bez hesla. Pomocí klávesové zkratky Ctrl+A se přihlašovací údaje sami předvyplní a následně je možné pokračovat stisknutím tlačítka *Login*, kterým dojde k otevření záložky online zobrazení připojeného systému.

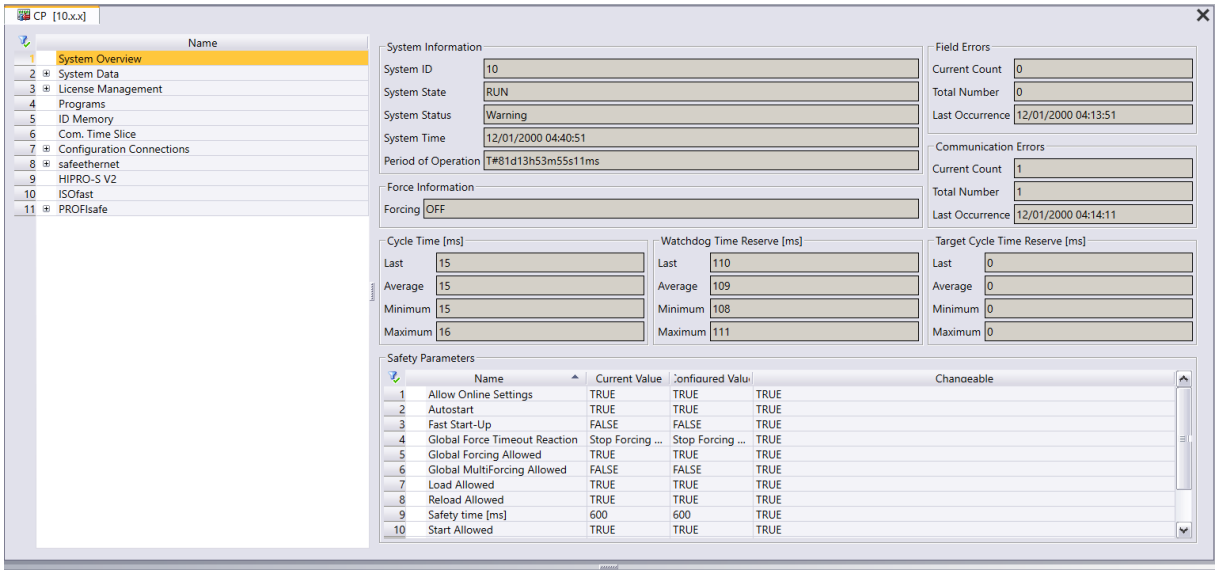
Pro každé další online spojení se systémem by již mělo stačit přímé přihlášení přes okno *System Login* bez dalšího potřebného nastavení.

7.16 Online prostředí

Po navázání online spojení systému s PADT se v SWX lze dostat do různých diagnostických a stavových oken a záložek. Tyto okna a záložky tvoří online prostředí softwaru SILworX.

7.16.1 Kontrolní panel

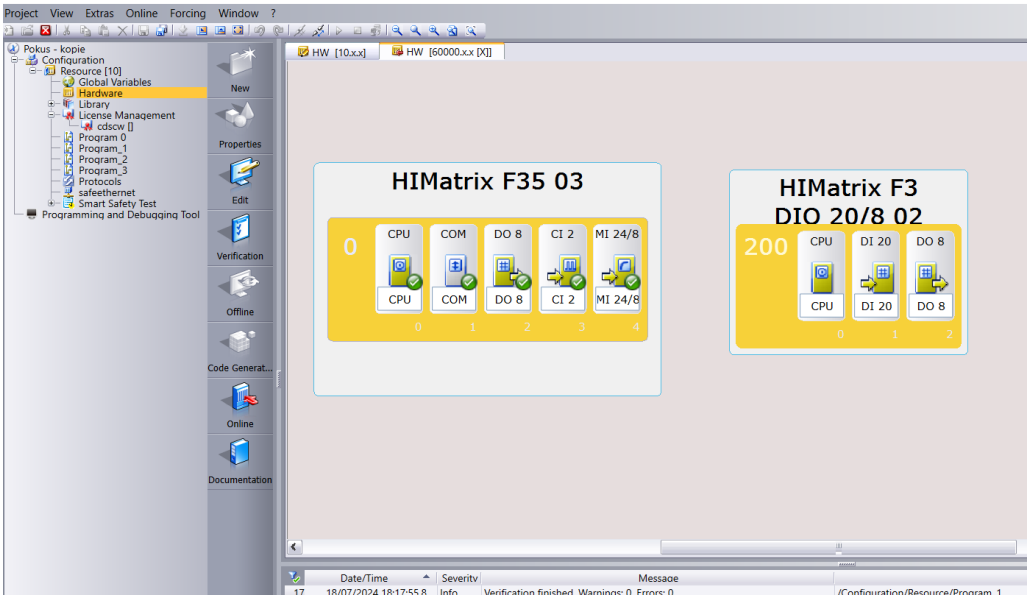
Mezi nejvýznamnější online záložku patří Kontrolní panel (Control panel), patrný na *obrázku 36*, jenž lze zobrazit klikem na *Resource* a následně na funkci *Online* na akčním panelu. Tato záložka obsahuje spoustu informací jako: základní nastavené parametry systému, aktuální online stav a status zařízení, časomíry cyklů a časových funkcí, správu licencí, stavy a časy běhu jednotlivých programů a stavy komunikačních protokolů.



Obrázek 36: Kontrolní panel

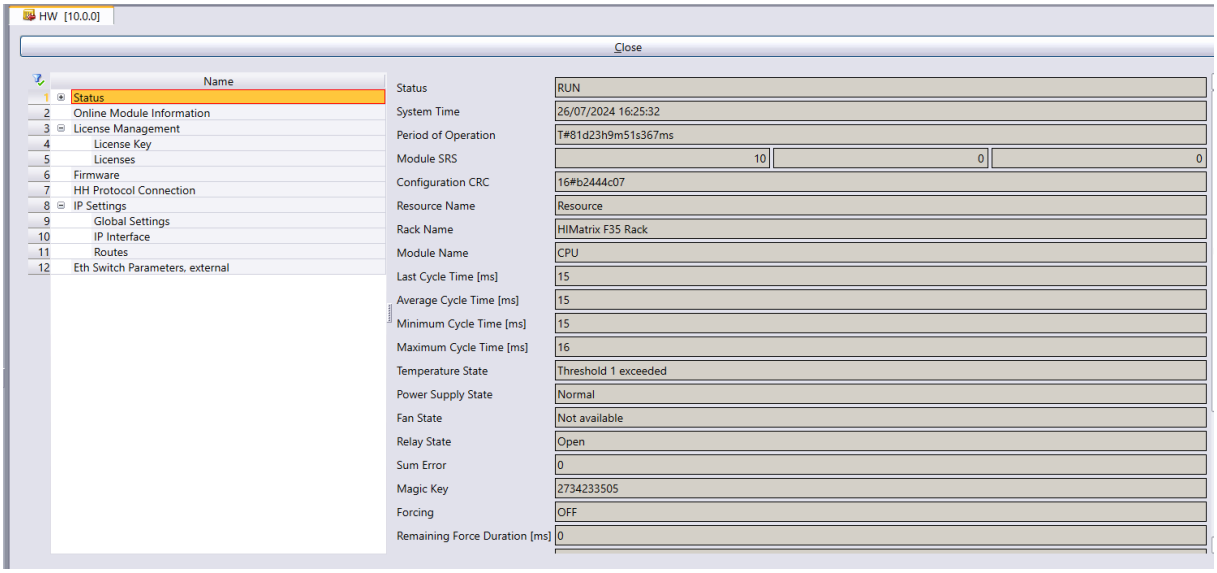
7.16.2 Online přehled hardwaru

Podobné informace týkající se čistě hardwaru je možné zobrazit klikem na *Hardware* následovaný opětovným stiskem funkce *Online*. V tomto online prostředí je zobrazena zvolená konfigurace systému, včetně online statusů jednotlivých modulů HIMatrix systému viz *obrázek 37*.



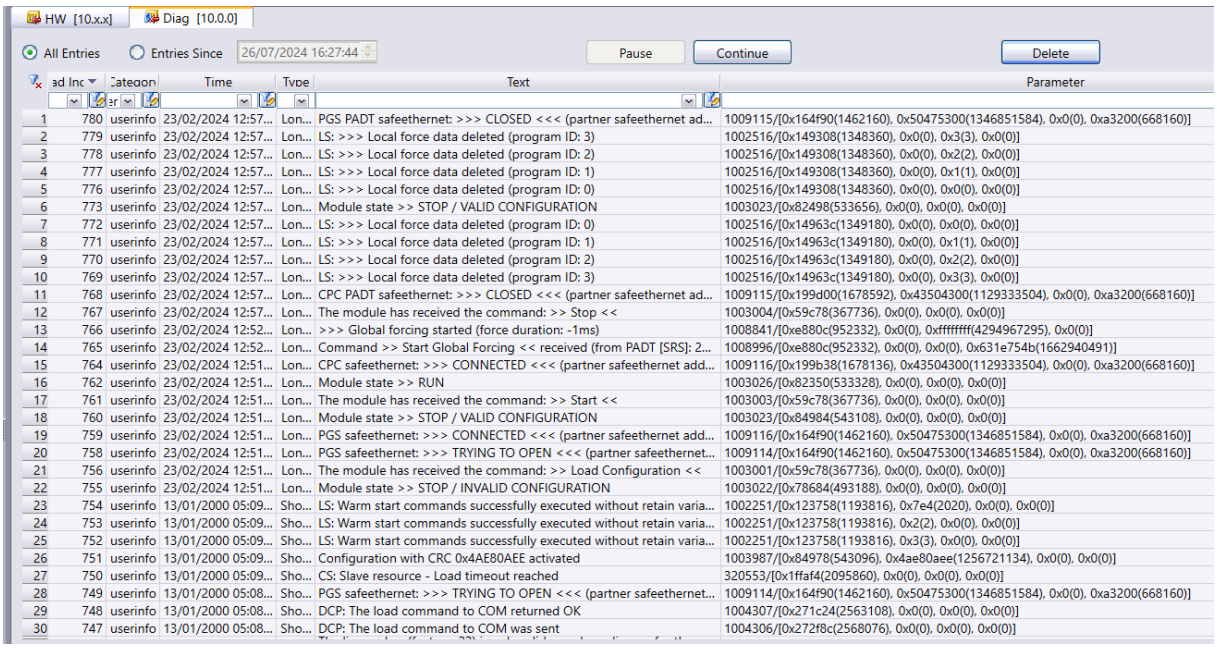
Obrázek 37: Online prostředí hardwaru

Podrobné statusy jednotlivých modulů systému je možně zobrazit dvojklikem na vybraný modul viz *obrázek 38*.



Obrázek 38: Hardware rozkliknutí

U procesorového CPU a komunikačního COM modulu je také možné zobrazit podrobnou diagnostiku jejich procesů. Lze tak učinit pravým klikem na zvolený modul vybráním možnosti *Diagnostics*, přičemž dojde k zobrazení nové karty s diagnostikou (viz *obrázek 39*). Pro zobrazení všech procesů je nutné zvolit možnost *All Entries*. Záznam procesů je možné libovolně zastavovat a mazat dle potřeby.



Obrázek 39: Diagnostika hardwaru

7.16.3 Online průběh programů

Online prostředí je dále užitečné pro sledování online dění v programech. Po vybrání konkrétního programu ve stromové struktuře a klikem na funkci *Online* se otevře záložka s online stavem vybraného programu. Prostředí vypadá a funguje stejně jako při offline (*kapitola 7.12)* simulaci, ale veškeré procesy se dějí na zařízení.

V OLT polích programu by se měly zobrazovat aktuální online hodnoty jednotlivých objektů.

Pravým tlačítkem myši lze vyvolat nabídku, ve které jsou další funkce. Nejužitečnější funkce z této nabídky je Activate Automatic OLT Fields, po jejímž zvolení dojde k zobrazení dočasných OLT polí, která se zobrazí nad každým výstupem všech objektů. Lze tak sledovat aktuální hodnoty i v mezilehlých částí programu.

7.17 Ovládání online funkcí

V SWX je možné získat přístup ke speciálním online funkcím, a to pokud je se systémem navázané online připojení.

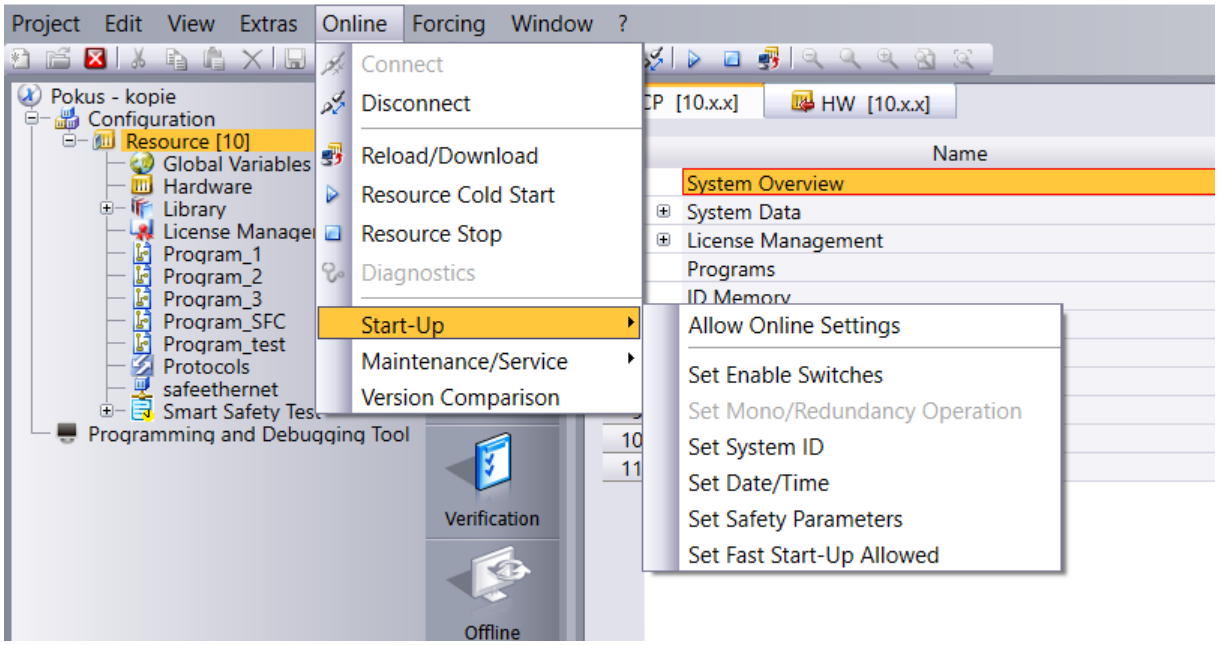
K ovládání těchto funkcí slouží speciální prvky, které lze zobrazit při kliknutí na *Resource* a následným klikem na *Online* prvek na akčním panelu. Dojde k přidání *Online* lišty a *Forcing* lišty, které by se měly zobrazit mezi ostatními lištami v levé horní části obrazovky. Dále by mělo dojít k zpřístupnění některých ikon na ikonovém panelu pod lištami.

V tomto stavu se musí online prostředí nacházet vždy, pokud je potřeba ovládat a nastavovat funkce popsané v této kapitole.

7.17.1 Start-Up

Po prvním navázání online spojení se systém bude nejspíše nutné nastavit některé správné parametry a funkce. Toho lze docílit přes *Online* lištu v levé horní části obrazovky, kde se vybere podmenu *Start-Up.* Toto podmenu nabízí několik možností, které se dají nastavit (*obrázek 40*). Důležité je nastavit tyto možnosti:

- Set System ID – dodatečné nastavení ID čísla kontroléru
- Set Date/Time – nastavení datumu a času
- Set Safety Parameters – nastavení bezpečnostních parametrů



Obrázek 40: Možnosti Start-Up podmenu

7.17.2 Force Editor

Pokud je potřeba zobrazit aktuální hodnoty globálních proměnných, je tak možné učinit skrze *Force Editor.* Ten lze otevřít pouze ve zvoleném online prostředí, a to přes prvek *Forcing* v horní liště SWX. Lze ho také zpřístupnit při offline simulaci zvoleného programu.

Po otevření Force Editoru se dá přepínat mezi různými záložkami proměnných, vstupů a výstupů, nebo mezi proměnnými, které jsou vyfiltrované dle použití v jednotlivých programech.

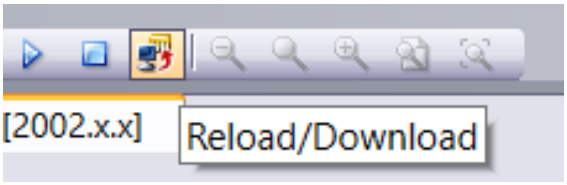
Hlavní funkcí Force Editoru je globální nucená změna hodnot, přičemž se v jednotlivých záložkách a oknech dá nastavit konkrétní hodnota, která bude při aktivaci nucení změněna.

Aktivace nucení změní hodnoty všech vybraných položek naráz. Aktivace a stejně tak deaktivace této funkce je možná přes lištu *Forcing,* výběrem *Start Global Forcing* nebo *Stop Global Forcing*.

7.17.3 Nahrání programu do zařízení

Po prvním navázání online spojení se systém bude nejspíše nacházet v STOP stavu. Tento stav je potřebný k nahrání prvního programu do systému. Dále musí být vygenerován programový kód.

Nahrání programu je možné přes tlačítko *Reload/Download* v horním ikonovém panelu viz *obrázek 41* nebo přes *Online* lištu v levé horní části obrazovky. Pro zpřístupnění obou těchto možností je nutné být v online prostředí prvku *Resource*. Následně dojde k zobrazení nového okna, ve kterém je doporučeno aktivovat funkci *Create Project Archive after Loading*. Tato funkce archivuje uživatelský program do souboru s příponou .PA3, který se uloží do zvoleného úložiště v dalším kroku.



Obrázek 41: Nahrání programu do zařízení

Pokračuje se zvolením *Download*, což by mělo začít nahrávat program do systému.

Možnost *Reload* by bylo možné použít v případě, kdy byl programový kód vygenerován s aktivovanou funkcí *Prepare Reolad* a pokud by zařízení bylo ve stavu RUN.

7.17.4 Spuštění a zastavení programu

Po úspěšném nahrání programu do zařízení je možné program spustit. To lze udělat pomocí horní ikonový lišty vybráním symbolu ▶, nebo přes horní menu lištu pod *Online* a výběrem funkce *Resource Cold Start*.

Program se následně v systému spustí a stav systému se změní ze STOP na RUN.

Stav RUN by měl systém udržovat stále, dokud nedojde k jeho záměrnému přepnutí a zastaveni programu. I v tom případě, kdy dojde ke krátkodobému výpadku napájení a resetu systému, systém se automaticky uvede do stavu RUN bez nutné interakce.

Program je samozřejmě v případě nutnosti zastavit, přičemž dojde ke změně stavu systému z RUN na STOP. To lze učinit pomocí horní ikonové lišty vybráním symbolu ■, nebo přes horní menu lištu pod *Online* a výběrem funkce *Resource Stop.*

7.18 Smart Safety Test

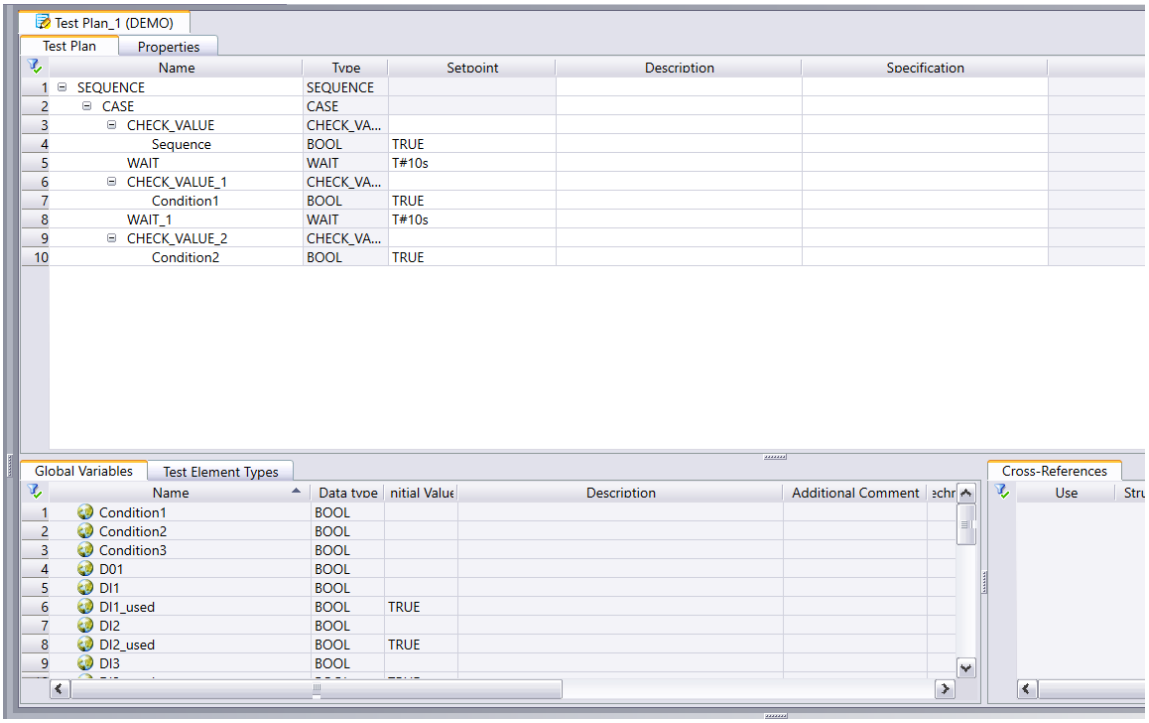
Smart Safety Test je dodatečný nástroj pro SWX, který slouží k ověřování a hodnocení uživatelských programů v bezpečnostních automatizačních systémech. Hlavním účelem Smart Safety Testu je umožnit uživatelům efektivně a opakovaně testovat své programy, aniž by museli měnit jejich kód. Tento nástroj se dá

Smart Safety Test umožňuje uživatelům vytvářet komplexní testovací plány, které mohou být prováděny offline i online. Testovací plány mohou obsahovat různé prvky, jako jsou nastavení hodnot (SET_VALUE), kontrola hodnot (CHECK_VALUE), příprava na testy (PREPARATION), čekací doby (WAIT) a přerušovací body (CHECKPOINT). Každý z těchto prvků slouží k simulaci různých scénářů a podmínek, které mohou nastat v reálném provozu, a k ověření, že systém reaguje správně. Například pomocí prvku SET_VALUE lze nastavit konkrétní hodnoty vstupních proměnných, zatímco prvek CHECK_VALUE umožňuje ověřit, zda výstupní proměnné dosahují očekávaných hodnot. Přerušovací body zase umožňují uživatelům zastavit testovací plán v určitém okamžiku a provést manuální zásahy nebo kontroly, což je užitečné při komplexním ladění a testování.

Jedním z hlavních přínosů Smart Safety Testu je jeho schopnost generovat podrobné testovací zprávy ve formátu PDF. Tyto zprávy dokumentují všechny kroky provedené během testování, včetně nastavených a naměřených hodnot, a poskytují vizuální indikaci úspěchu nebo neúspěchu jednotlivých testů. Tento proces je zvláště užitečný pro inspekční orgány a prokazování, že změny v uživatelském programu nezpůsobily nové chyby v dříve testovaných částech. Tímto způsobem mohou uživatelé zajistit, že jejich programy zůstávají bezpečné a spolehlivé i po provedení úprav nebo aktualizací.

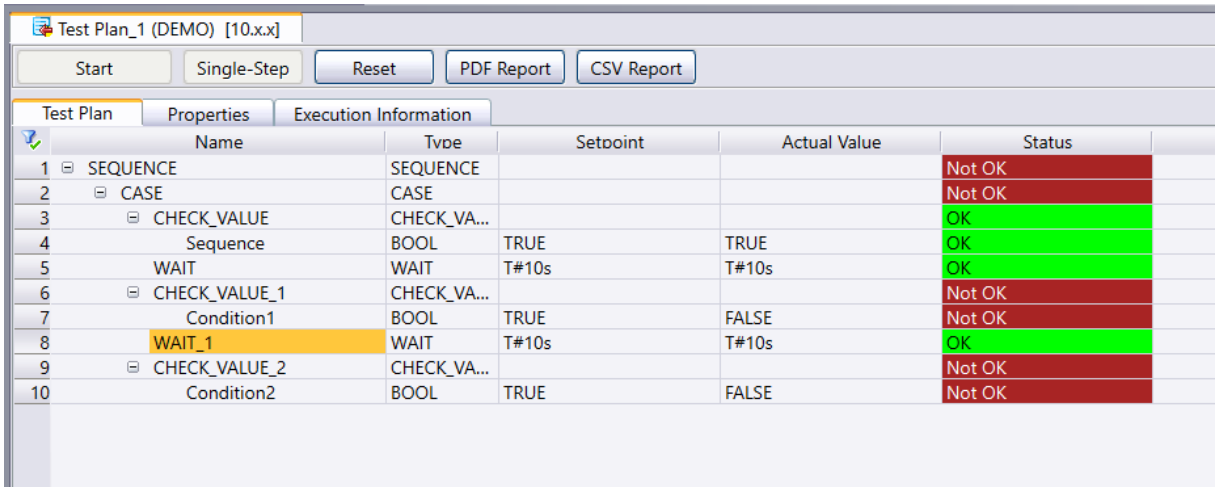
Smart Safety Test vyžaduje pro plné využití svého potenciálu speciální softwarovou licenci. Bez této licence mohou uživatelé vytvářet a provádět pouze omezený počet testovacích plánů s omezeným počtem prvků. Plná licence však umožňuje využívat všechny funkce nástroje bez omezení, což je zásadní pro organizace, které provádějí časté a rozsáhlé testování svých bezpečnostních systémů.

Dalším důležitým aspektem Smart Safety Testu je jeho integrace s prostředím SILworX, což umožňuje snadnou správu a úpravy testovacích plánů. Nejprve se vytvoří nový prvek *Smart Safety Test* ve stromové struktuře přes prvek *Resource* (***obrázek 42***). Uživatelé dále mohou vytvářet nové testovací plány přímo pod tímto prvkem a hotové plány lze importovat i z CSV souborů. To zvyšuje flexibilitu a usnadňuje sdílení a jejich opětovné použití. Kromě toho mohou uživatelé využívat různé typy testovacích prvků a globálních proměnných, které lze jednoduše přetahovat do testovacích plánů.



Obrázek 42: Obrázek Smart Safety test editace

Přepnutím vytvořeného testovacího plánu do offline simulace nebo do online režimu zapříčiní jeho spuštění, přičemž se vytvořené úlohy budou provádět postupně dle nastavených parametrů jednotlivých prvků. U každého prvku je pak vidět, zda testem prošel, nebo ne (*obrázek 43*). Po dokončení testu lze vytvořit evaluační PDF soubor s výsledkem testu.



Obrázek 43: Vyhodnocení Smart Safety testu

7.19 ComUserTask

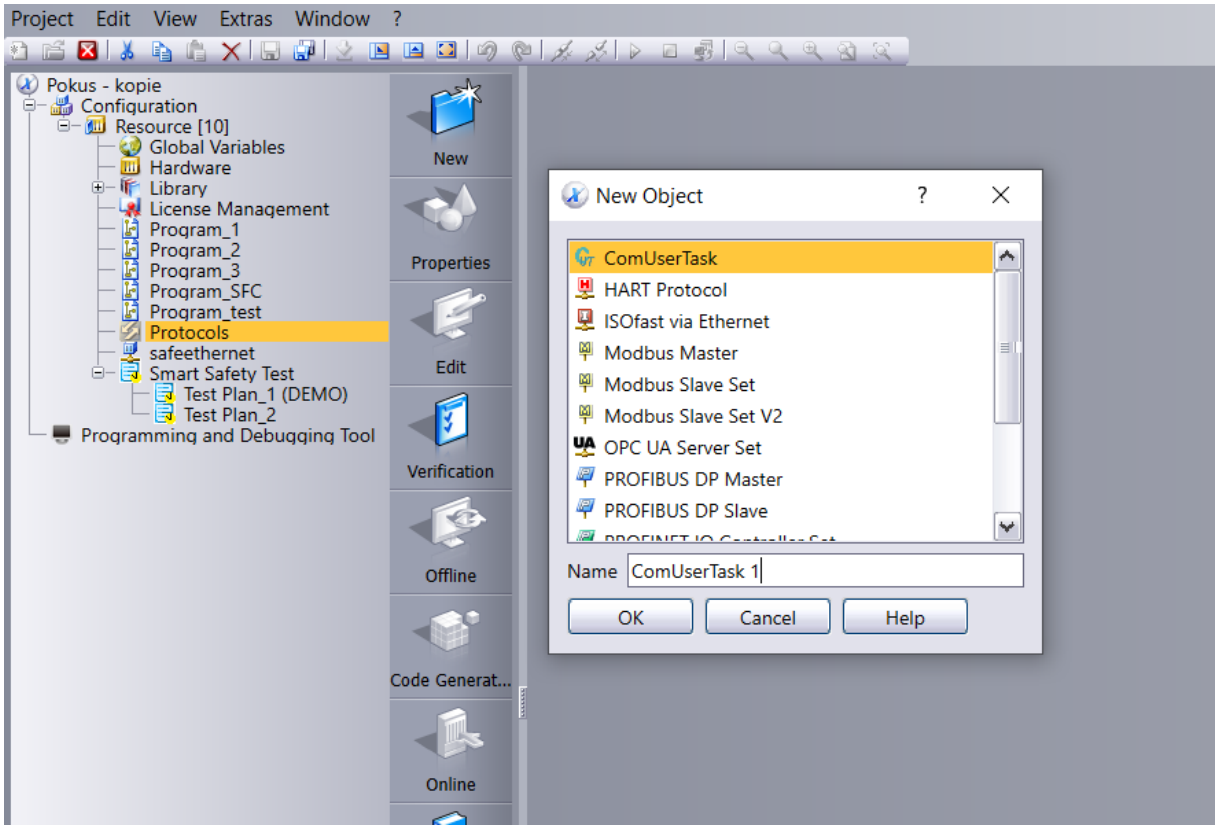
ComUserTask je nástroj, který umožňuje uživatelům psát C programy, které běží na komunikačním COM modulu a umožňují připojení k různým komunikačním rozhraním,

zejména RS485, TCP a UDP. Hlavním cílem je rozšířit možnosti komunikace a integrace kontrolérů HIMA s jinými systémy a zařízeními. Všechny informace pro vytvoření této kapitoly byly čerpány z oficiálního manuálu pro ComUserTask, kde jsou podrobně popsány instalační a programovací postupy včetně příkladů. (odkaz!!!)

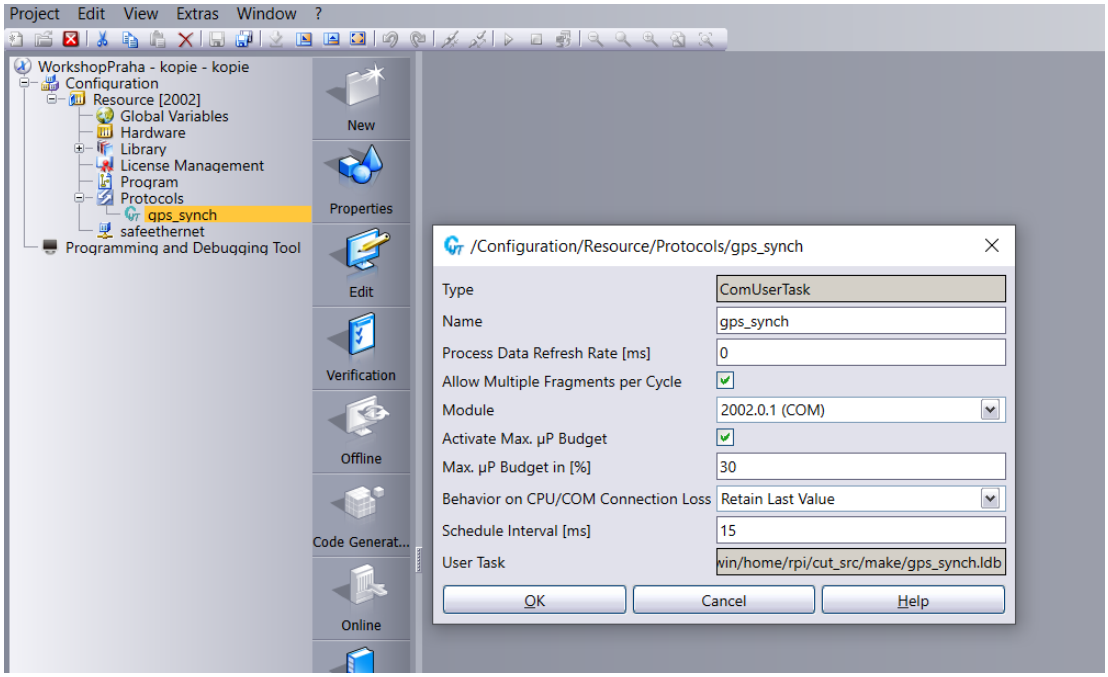
Pro správnou konfiguraci ComUserTask je nutná znalost programování v jazyce C a porozumění komunikačním protokolům, pro které má být tento nástroj nastaven.

Instalace nástroje ComUserTask zahrnuje několik kroků. Nejprve je nutné nainstalovat doplňkové nástroje, které zahrnuje GNU C kompilátor a Cygwin, což je simulované unixové prostředí běžící na OS Windows. Instalace prostředí Cygwin začíná spuštěním instalačního programu. Během instalace je doporučeno deaktivovat antivirový program, aby nedošlo k problémům při instalaci. Po dokončení instalace prostředí Cygwin se pokračuje instalací GNU kompilátoru a nastavením systémových proměnných, aby kompilátor správně fungoval.

Používání ComUserTask začíná v SWX vytvořením nového *ComUserTask* prvku ve stromové struktuře pod prvkem *Protocols* (viz *obrázek 44*). Ve vlastnostech nového prvku je zapotřebí vybrat správný COM modul a nastavit další potřebné parametry (viz *obrázek 45*). V SWX se dále definují globální vstupní a výstupní proměnné, které musí odpovídat proměnným používaných v C kódu. Následně lze vytvořit SWX program pomocí funkčních bloků, který může fungovat např. jako časovač.



Obrázek 44: Vytváření nového ComUserTask prvku



Obrázek 45: Nastavení vlastností ComUserTask prvku

Vlastní programování ComUserTask se provádí ve zvoleném textovém editoru v programovacím jazyce C. To zahrnuje definování vstupních a výstupních proměnných, psaní hlavní úlohy a implementaci potřebných funkcí pro komunikaci a zpracování dat.

Vytvořený C kód se zkompiluje pomocí GNU kompilátoru v prostředí Cygwin na spustitelný kód v podobě .ldb souboru, který se načte do *ComUserTask* prvku v SWX. Tento kód se poté nahrává do kontroléru HIMA, kde běží jako součást komunikačního modulu.

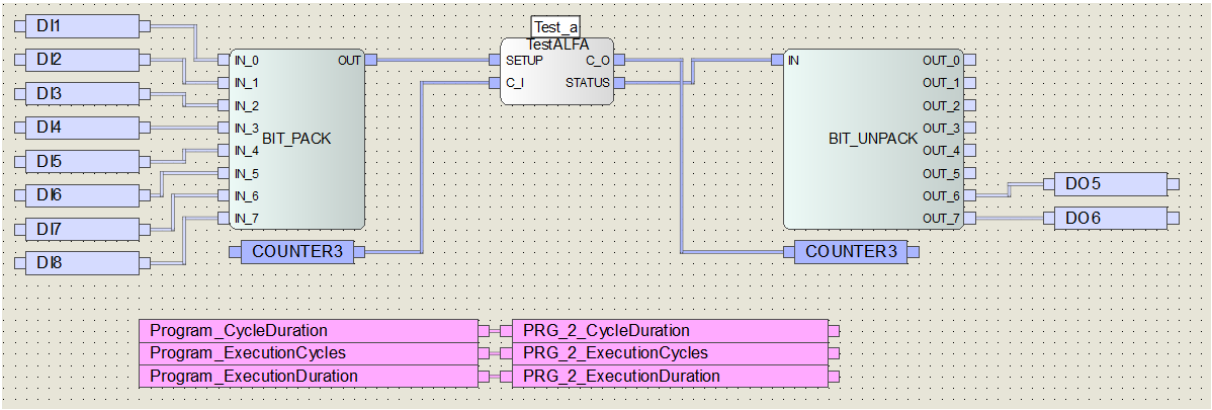
K používání ComUserTask je třeba speciální SWX licence a softwarový aktivační kód, který se získává od výrobce. Tento kód je nutný pro aktivaci softwaru a umožňuje přístup k plné funkcionalitě nástroje.

8 Návrh a realizace zátěžových testů

Pro ověření klíčových parametrů kontroléru F35 řady HIMatrix bylo nutné navrhnout a provést zátěžové testy, které měly zjistit, zda je tento kontrolér vhodnou volbou pro koncept Železnice 4.0. Všechny testy byly provedeny přímo na reálném kontroléru F35. Návrh a realizace testů probíhaly ve spolupráci s firmou Casablanca INT pod vedením Ing. Jiřího Chludila.

8.1 Testovací program

Byla navržena základní struktura testovacího programu tak, aby bylo možné změnou kódu jeho funkčního C++ bloku změnit funkci zátěžového testu. Byl vytvořen ve vývojovém prostředí SILworX a skládá se ze tří částí (*obrázek 46*).



Obrázek 46: Testovací program v SILworX

První část tvoří násobitel, který je vytvořený pomocí funkčního bloku BIT_PACK, jehož funkcí je na základě kombinací vstupů vytvořit výstupní číslo v hexadecimální soustavě, které je pak ve funkčním C++ bloku převedeno do desítkové soustavy. Na vstupy násobitele jsou přivedeny proměnné, které jsou spojeny s digitálními vstupy kontroléru F35. Výstup násobitele je pak přiveden do funkčního C++ bloku, který tvoří další část testovacího programu. Násobitel je ovládán pomocí fyzických vstupů, na které se přivádí napětí v úrovni „logická 1“. Je tak možné vytvářet různé kombinace vstupů. Při testování se používalo osm hodnot násobitele. Hodnota násobitele byla nastavena přivedením napětí na vstup DI1 až DI8. Jelikož byl při testu měněn pouze jeden vstup, výstup násobitele pro každý test byl násoben hodnotami 1, 2, 4, 8, 16, 32, 64 a 128.

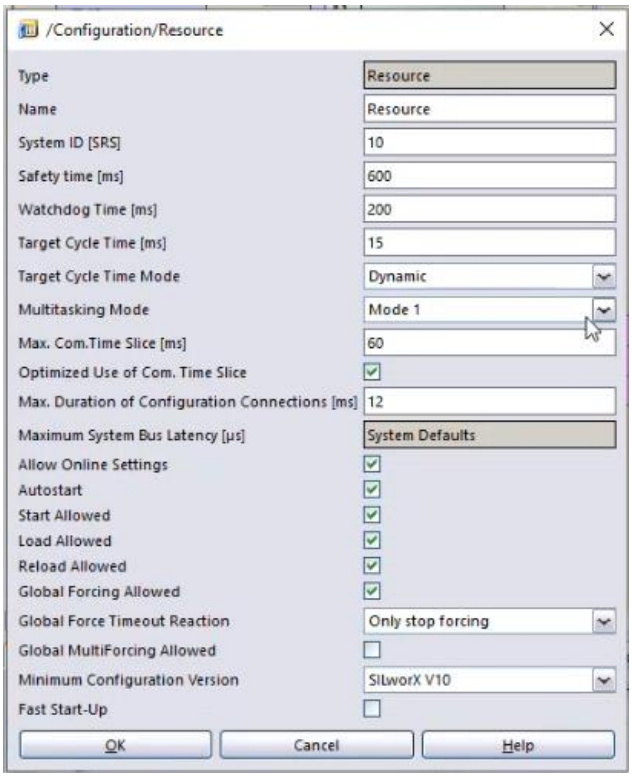
Druhá část testovacího programu je tvořena funkčním C++ blokem, jehož kód je navržen jako zátěžový test. Tento blok je navržen tak, aby testoval zatížení a stabilitu systému F35 prostřednictvím opakovaných aritmetických a logických operací. Tyto operace probíhají ve smyčce, jejíž počet iterací je závislý na výstupu násobitele. Pro tento funkční blok byly

vytvořeny dva C++ programy, které budou použity ve třech testech, jež budou popsány a vyhodnoceny v dalších podkapitolách. Při nahrávání vytvořených C++ programů do prostředí SILworX bylo zjištěno, že pro správnou funkci by každý program měl mít svou vlastní složku v adresáři.

Třetí část testovacího programu se zabývá vyhodnocením C++ programů, přičemž byly sledovány tři systémové proměnné: Doba cyklu (Cycle Duration), Počet vykonaných cyklů (Execution Cycles) a Doba vykonání (Execution Duration). Z těchto proměnných a nastavených vlastností kontroléru bylo možné spočítat interval operací, které dokáže HImatrix F35 za čas provést.

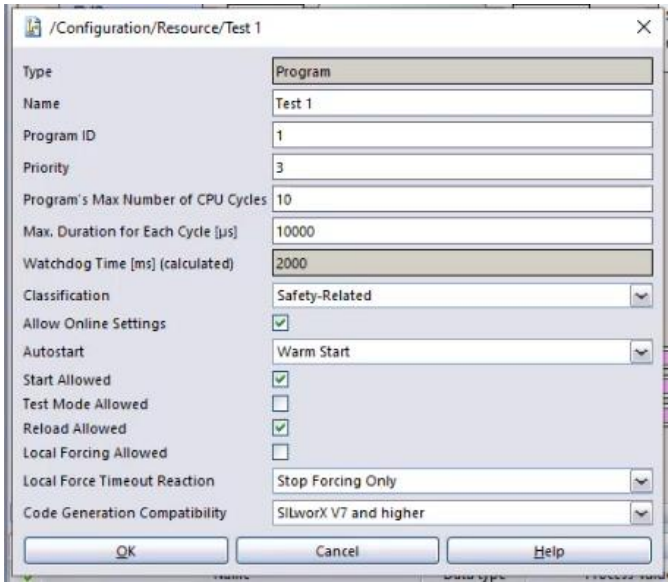
8.1.1 Nastavené parametry pro testy

Všechny nastavené parametry kontroléru a jednotlivých programů byly pro všechny tři testy identické. Přesné nastavení vlastností kontroléru je vyobrazeno na *obrázku 47* a nastavení vlastností programů je vyobrazeno na *obrázku 48*.



Obrázek 47: Nastavení vlastností kontroléru

Z uvedených parametrů vlastností kontroléru je pro nás důležité zejména nastavení cílené doby jednoho cyklu (Target Cycle Time), včetně jeho módu, který se v rámci testu volil mezi fixním a dynamickým. Dalším měněným parametrem byl režim multitaskingu, který se přepínal mezi všemi třemi režimy.



Obrázek 48: Nastavení vlastností programů

U všech programů byla nastavená maximální doba trvání jednoho cyklu na 10000 µs a priorita programů byla nastavena vždy na 3, aby se nastavili stejné podmínky pro všechny testy.

8.2 Test 1

První zátěžový test byl zaměřen na zpracování C++ programu ALFA, který byl složen z čistě aritmetických operací. Vnitřní násobící koeficient byl nastaven na hodnotu 10000 a násobil se s převedenou výslednou hodnotou funkčního bloku BIT_PACK. Výsledná hodnota této operace určovala maximální počet for cyklů, které musely být v rámci programu spočítány. Navrhnuté operace jsou zobrazené v úryvku kódu programu ALFA na *obrázku 49*, kde proměnná pmVA měla počáteční hodnotu 999, a pmVB měla počáteční hodnotu 100. Tato sestava operací se ve zmíněném for cyklu opakuje a celkem čítá 32 operací.

```
for (pmVCounter = 0; pmVCounter < (pmVSETUP*10000); pmVCounter = pmVCounter + 1) {
    pmVC = pmVA + pmVB;
    pmVC = pmVA - pmVB;
    pmVC = pmVA / pmVB;
    pmVC = pmVA * pmVB;
```

Obrázek 49: Úryvek kódu programu ALFA

Pro první test byly vytvořeny dvě tabulky. Do *tabulky č. 1* se zapisovaly hodnoty, které byly naměřeny při nastavení dynamického režimu cílené doby jednoho cyklu, kdežto do *tabulky č. 2* se zapisovaly hodnoty naměřené při fixním režimu. Měřeny byly hodnoty následujících veličin: Doba cyklu, Počet potřebných cyklů, Doba vykonání operací. Tyto hodnoty se naměřily pro každou hodnotu danou násobitelem a pro každý režim multitaskingu.

Test 1 - Dymamický režim									
Režim multitaskingu	Násobitel	1	2	4	8	16	32	64	128
Režim 1	Doba cyklu [µs]	3000	3000-4000	4000-5000	7000	11000	23000	46000	90000
	Počet potřebných cyklů	1	1	1	1	1	2	4	8
	Doba vykonání [µs]	750	1420	2400	4700	9100	17800	35000	71000
Režim 2	Doba cyklu [µs]	3000	3000-4000	4000-5000	7000	11000	23000	46000	90000
	Počet potřebných cyklů	1	1	1	1	1	2	4	8
	Doba vykonání [µs]	750	1420	2400	4700	9100	17800	35000	71000
Režim 3	Doba cyklu [µs]	12000	12000	12000	12000	12000	24000	50000	99000
	Počet potřebných cyklů	1	1	1	1	1	2	4	8
	Doba vykonání [µs]	740	1280	2400	4600	9000	17800	35000	70600

Tabulka 1: Hodnoty naměřené při dynamickém režimu cílené doby jednoho cyklu, test 1

Test 1 - Fixní režim									
Režim multitaskingu	Násobitel	1	2	4	8	16	32	64	128
Režim 1	Doba cyklu [µs]	15000	15000	15000	15000	15000	30000	60000	120000
	Počet potřebných cyklů	1	1	1	1	1	2	4	8
	Doba vykonání [µs]	730	1280	2340	4600	9050	17700	35200	70500
Režim 2	Doba cyklu [µs]	15000	15000	15000	15000	15000	30000	60000	120000
	Počet potřebných cyklů	1	1	1	1	1	2	4	8
	Doba vykonání [µs]	730	1280	2340	4600	9050	17700	35200	70500
Režim 3	Doba cyklu [µs]	15000	15000	15000	15000	15000	30000	60000	120000
	Počet potřebných cyklů	1	1	1	1	1	2	4	8
	Doba vykonání [µs]	730	1280	2360	4600	9000	17800	35200	70500

Tabulka 2: Hodnoty naměřené při fixním režimu, test 1

8.3 Test 2

Druhý zátěžový test byl zaměřen na zpracování C++ programu BETA, který byl složen z čistě logických operací. Vnitřní násobící koeficient byl nastaven na hodnotu 5000 a násobil se s převedenou výslednou hodnotou funkčního bloku BIT_PACK. Výsledná hodnota této operace určovala maximální počet for cyklů, které musely být v rámci programu spočítány. Navrhnuté operace jsou zobrazené v úryvku kódu programu BETA na *obrázku 50*, kde proměnná pmVA měla počáteční hodnotu 999, a pmVB měla počáteční hodnotu 100. Tato sestava operací se ve zmíněném for cyklu opakuje a celkem čítá 21 operací.

```
for (pmVCounter = 0; pmVCounter < (pmVSETUP*5000); pmVCounter = pmVCounter + 1) {
    pmVC = pmVA & pmVB;
    pmVC = pmVA | pmVB;
    pmVC = pmVA xor pmVB;
```

Obrázek 50: Úryvek kódu programu BETA

Pro druhý test byly vytvořeny dvě tabulky. Do *tabulky č. 3* se zapisovaly hodnoty, které byly naměřeny při nastavení dynamického režimu cílené doby jednoho cyklu, kdežto do *tabulky č. 4* se zapisovaly hodnoty naměřené při fixním režimu. Měřeny byly hodnoty následujících veličin: Doba cyklu, Počet potřebných cyklů, Doba vykonání operací. Tyto hodnoty se naměřily pro každou hodnotu danou násobitelem a pro každý režim multitaskingu.

Test 2 - Dymamický režim									
Režim multitaskingu	Násobitel	1	2	4	8	16	32	64	128
Režim 1	Doba cyklu [µs]	3000	4000	4000	5000	7000	12000	23000	45000
	Počet potřebných cyklů	1	1	1	1	1	1	2	4
	Doba vykonání [µs]	430	740	1280	2400	4600	9000	17800	35300
Režim 2	Doba cyklu [µs]	3000	4000	4000	5000	7000	12000	23000	45000
	Počet potřebných cyklů	1	1	1	1	1	1	2	4
	Doba vykonání [µs]	430	740	1280	2400	4600	9000	17800	35000
Režim 3	Doba cyklu [µs]	13000	13000	13000	13000	13000	13000	25000	50000
	Počet potřebných cyklů	1	1	1	1	1	1	2	4
	Doba vykonání [µs]	440	800	1280	2400	4550	9000	17800	35300

Tabulka 3: Hodnoty naměřené při dynamickém režimu cílené doby jednoho cyklu, test 2

Test 2 - Fixní režim									
Režim multitaskingu	Násobitel	1	2	4	8	16	32	64	128
Režim 1	Doba cyklu [µs]	15000	15000	15000	15000	15000	15000	30000	60000
	Počet potřebných cyklů	1	1	1	1	1	1	2	4
	Doba vykonání [µs]	440	740	1280	2400	4550	9000	17700	35300
Režim 2	Doba cyklu [µs]	15000	15000	15000	15000	15000	15000	30000	60000
	Počet potřebných cyklů	1	1	1	1	1	1	2	4
	Doba vykonání [µs]	440	740	1280	2400	4550	9000	17700	35300
Režim 3	Doba cyklu [µs]	15000	15000	15000	15000	15000	15000	30000	60000
	Počet potřebných cyklů	1	1	1	1	1	1	2	4
	Doba vykonání [µs]	440	740	1280	2400	4550	9000	17700	35000

Tabulka 4: Hodnoty naměřené při fixním režimu, test 2

8.4 Test 3

Třetí zátěžový test byl zaměřen na zpracování kombinace C++ programů ALFA a BETA, který tedy kombinoval aritmetické i logické operace. Tato kombinace je složena konkrétně ze dvou C++ programů ALFA a dvou programů BETA, což v součtu vychází na 4 paralelně běžící operace. Vnitřní násobící koeficient byl nastaven na hodnotu 1000 pro program ALFA a 5000 pro program BETA, které se násobily s převedenou výslednou hodnotou funkčního bloku BIT_PACK. Výsledná hodnota této operace určovala maximální počet for cyklů, které musely být v rámci programu spočítány. Navrhnuté operace jsou pro programy ALFA i BETA stejné strukturou i rozsahem kódů, včetně proměnných pmVA s pmVB. Tento test, jako jediný, testuje běh čtyř SWX programů naráz, konkrétně dvou programů ALFA a dvou programů BETA.

Pro třetí test byly vytvořeny dvě tabulky. Do *tabulky č. 5* se zapisovaly hodnoty, které byly naměřeny při nastavení dynamického režimu cílené doby jednoho cyklu, kdežto do *tabulky č. 6* se zapisovaly hodnoty naměřené při fixním režimu. Měřeny byly hodnoty následujících veličin: Doba cyklu, Počet potřebných cyklů, Doba vykonání operací. Tyto hodnoty se naměřily pro každou hodnotu danou násobitelem a pro každý režim multitaskingu.

Test 3 - Dymamický režim									
Režim multitaskingu	Násobitel	1	2	4	8	16	32	64	128
Režim 1	Doba cyklu [µs]	5000	7000	9000	17000	30000	77000	154000	305000
	Počet potřebných cyklů	1	1	1	1	1	2	4	8
	Doba vykonání [µs]	540	1280	2400	4500	9000	17700	35500	70700
Režim 2	Doba cyklu [µs]	5000	7000	9000	17000	30000	77000	155000	305000
	Počet potřebných cyklů	1	1	1	1	1	2	4	8
	Doba vykonání [µs]	740	1280	2400	4550	9000	17800	35600	70500
Režim 3	Doba cyklu [µs]	43000	43000	43000	43000	43000	85000	170000	340000
	Počet potřebných cyklů	1	1	1	1	1	2	4	8
	Doba vykonání [µs]	540	1290	2400	4500	9000	17700	35300	70650

Tabulka 5: Hodnoty naměřené při dynamickém režimu cílené doby jednoho cyklu, test 3

Test 3 - Fixní režim									
Režim multitaskingu	Násobitel	1	2	4	8	16	32	64	128
Režim 1	Doba cyklu [µs]	15000	15000	15000	17000	29000	77000	150000	300000
	Počet potřebných cyklů	1	1	1	1	1	2	4	8
	Doba vykonání [µs]	740	1280	2360	4760	9050	17800	35300	71000
Režim 2	Doba cyklu [µs]	15000	15000	15000	16000	30000	77000	153000	305000
	Počet potřebných cyklů	1	1	1	1	1	2	4	8
	Doba vykonání [µs]	740	1280	2370	4700	8910	17700	35400	70600
Režim 3	Doba cyklu [µs]	42000	43000	43000	43000	43000	43000	170000	340000
	Počet potřebných cyklů	1	1	1	1	1	2	4	8
	Doba vykonání [µs]	740	1280	2360	4500	8950	17800	35400	70700

Tabulka 6: Hodnoty naměřené při fixním režimu, test 3

8.5 Vyhodnocení testů

Po naměření hodnot se data vyhodnotila. Vzhledem k tomu, že z *tabulek 2* a *4* vychází stejné hodnoty, které jsou oproti *tabulkám 1* a *3* méně vhodné, je možné je z dalšího porovnání vyloučit a prohlásit, že při zpracování jednoho programu je lepší a efektivnější využít dynamického režimu cílené doby jednoho cyklu.

U zbývajících dvou tabulek dále můžeme vypočítal reálný procesní čas, který pomůže stanovit výhodnost použití módu multitaskingu. Pro sjednocení výpočtů se bude počítat s nejvyšším násobitelem o hodnotě 128. Tento čas se vypočítá dle následujícího vzorce:

$$t_p = \frac{Dc * Ns}{10^6}$$

Kde:

- t_p = reálný procesní čas [s]
- Dc = doba cyklu [µs]
- Ns = násobitel [-]

U prvního testu se výpočet provede např. takto:

$$t_p = \frac{9000 * 128}{10^6} = 11{,}52\ s$$

	Test 1	Test 2
Režim multitaskingu 1	11,520	5,760
Režimmultitaskingu 2	11,520	5,760
Režim multitaskingu3	12,672	6,400

Tabulka 7: Reálný procesní čas

V *tabulce 7* je na první pohled jistá podobnost dat. Hodnoty v tabulkách druhého testu se zhruba rovnají polovině hodnot z prvního testu a nelze jednoduše zjistit, zda kontrolér F35 dosahuje lepších výsledků při zpracování aritmetických operací nebo logických operací. To lze vyhodnotit získáním počtu provedených procesů za čas. Pro sjednocení výpočtů se bude opět počítat s nejvyšším násobitelem o hodnotě 128. K výpočtu lze použít vzorec:

$$P_p = \frac{Po * Ns}{Dc}$$

Kde:

- P_p = počet procesů za čas [procesy/µs]
- Po = Počet operací [-]
- Ns = násobitel [-]
- Dc = doba cyklu [µs]

U prvního testu se výpočet provede např. takto:

$$P_p = \frac{(32 * 10000) * 128}{90000} = 455{,}111\ procesů/\mu s$$

V *tabulce 8* jsou uvedeny vypočítané procesy za čas pro test 1 a pro test 2, vždy pro daný mód multitaskingu.

	Test 1	Test 2
Režim multitaskingu 1	455,111	298,667
Režimmultitaskingu 2	455,111	298,667
Režim multitaskingu3	413,737	268,800

Tabulka 8: Vypočtené hodnoty pro test 1 a 2

Z tabulky lze vyhodnotit, že F35 je při zpracovávání jednoho programu stejně efektivní v režimech multitaskingu 1 a 2, a přitom více efektivní nežli u režimu multitaskingu 3. Lze tak vyhodnotit, že kontrolér lépe zpracovává aritmetické operace nežli logické operace.

Výpočet pro získání počtu provedených procesů za čas byl proveden také pro data z obou tabulek testu 3, čímž vznikla *tabulka 9*.

	Dynamický režim	Fixní režim
Režim multitaskingu 1	356,7213115	362,6666667
Režim multitaskingu 2	356,7213115	356,7213115
Režim multitaskingu 3	320	320

Tabulka 9: Vypočtené hodnoty z obou tabulek testu 3

Na základě dat z této tabulky lze prohlásit, že v případě běhu několika různých programů najednou ve více cyklech je lepší použít režim multitaskingu 1 nebo 2, vzhledem k jejich větší efektivitě. Z tabulky dále vyplívá, že nejefektivnější by měl kontrolér při fixním režimu cílené doby jednoho cyklu, a to při využití prvního módu multitaskingu.

9 Závěr

Tato bakalářská práce se zabývala možnostmi využití komerčně dostupných programovatelných logických automatů (COTS PLC) jako objektových kontrolérů v rámci konceptu Železnice 4.0. Byla provedena detailní analýza technických a funkčních požadavků na COTS PLC, stejně jako jejich dostupnosti a vhodnosti pro implementaci do železniční infrastruktury.

V rámci této bakalářské práce jsem se detailně seznámil s kontrolérem HIMAtrix F35 a vývojovým prostředí SILworX. Byly provedeny zátěžové testy kontroléru, které ukázaly poměrně velkou výpočetní výkonnost systému. Dále jsem se věnoval možnostem aplikace kontrolérů HIMatrix jako OC v rámci konceptu Železnice 4.0

Na základě získaných výsledků lze doporučit pokračování v dalším výzkumu a testování konkrétních modelů COTS PLC F35 v reálných podmínkách železničního provozu. Tento výzkum by měl zahrnovat podrobnější analýzu bezpečnostních aspektů a dlouhodobé spolehlivosti těchto zařízení.

Seznam použité literatury

[1] ITS-Railway: Poziční dokument SDT k dalšímu rozvoji telematiky v železniční dopravě. In: *Sdružení pro dopravní telematiku* [online]. 2015 [cit. 2024-07-02]. Dostupné z: https://www.sdt.cz/dokumenty/2015_Pozicni_dokument_SDT_rozvoj_ITS_na_zeleznici.pdf

[2] LESO, Martin. Koncept Železnice 4.0. - vize digitální železnice v ČR. In: *Železnice 4.0* [online]. 2022 [cit. 2024-07-02]. Dostupné z: https://zeleznice40.cz/wp-content/uploads/2022/03/Leso_Zeleznice4_0_Final.pdf

[3] ETHERNET/IP. In: *FCC průmyslové systémy* [online]. c2024 [cit. 2024-07-04]. Dostupné z: https://www.fccps.cz/rub-technicky-magazin/ethernetip#title

[4] CAN FD - STRUČNÝ ÚVOD. In: *FCC průmyslové systémy* [online]. c2024 [cit. 2024-07-04]. Dostupné z: https://www.fccps.cz/rub-technicky-magazin/can-fd-%C2%A0strucny-uvod#title

[5] MODBUS TCP. In: *FCC průmyslové systémy* [online]. c2024 [cit. 2024-07-04]. Dostupné z: https://www.fccps.cz/rub-technicky-magazin/modbus-tcp#title

[6] PROFINET. In: *FCC průmyslové systémy* [online]. c2024 [cit. 2024-07-04]. Dostupné z: https://www.fccps.cz/rub-technicky-magazin/profinet#title

[7] PROFIBUS. In: *FCC průmyslové systémy* [online]. c2024 [cit. 2024-07-04]. Dostupné z: https://www.fccps.cz/rub-technicky-magazin/profibus#title

[8] Kritická infrastruktura. In: *Hasičský záchranný sbor České republiky* [online]. c2024 [cit. 2024-07-05]. Dostupné z: https://www.hzscr.cz/clanek/web-krizove-rizeni-a-cnp-kriticka-infrastruktura-kriticka-infrastruktura.aspx

[9] ČSN EN 50126-1 ED.2 (333502). In: *Technické normy ČSN* [online]. 2011 [cit. 2024-07-05]. Dostupné z: https://www.technicke-normy-csn.cz/csn-en-50126-1-ed-2-333502-181386.html

[10] ČSN EN 50128 (342670). In: *Technické normy ČSN* [online]. 2022 [cit. 2024-07-05]. Dostupné z: https://www.technicke-normy-csn.cz/csn-en-50128-342670-182823.html

[11] ČSN EN 50129 ED.2 (342675). In: *Technické normy ČSN* [online]. 2022 [cit. 2024-07-05]. Dostupné z: https://www.technicke-normy-csn.cz/csn-en-50129-ed-2-342675-182841.html

[12] SIL, průmysl procesní techniky. In: *Festo* [online]. c2024 [cit. 2024-07-05]. Dostupné z: https://www.festo.com/cz/cs/e/reseni/odvetvi/prumysl-procesni-techniky/chemicky-prumysl/funkcni-bezpecnost-sil-id_4237/

[13] Functional Safety – SIL. In: *Endress+Hauser* [online]. [cit. 2024-07-06]. Dostupné z: https://portal.endress.com/wa001/dla/5000639/2936/000/01/CP01008Z11EN_0313_SIL-Brochure_X4_.pdf

[14] About us. In: *HIMA: Smart Safety* [online]. c2024 [cit. 2024-07-11]. Dostupné z: https://www.hima.com/en/about-hima

[15] HIMatrix. In: *HIMA: Smart Safety* [online]. c2024 [cit. 2024-07-11]. Dostupné z: https://www.hima.com/en/products-services/himatrix

[16] Pilz. In: *Pilz* [online]. 2024 [cit. 2024-08-05]. Dostupné z: https://www.pilz.com/en-INT/company/profile

[17] HIMatrix Module Overview. In: *HIMA: Smart Safety* [online]. c2024 [cit. 2024-07-11]. Dostupné z: https://www.hima.com/en/products-services/Products/himatrix/HIMatrix%20Module%20Overview

[18] HIMatrix Module Overview. In: HIMA: Smart Safety [online]. c2024 [cit. 2024-07-11]. Dostupné z: https://www.hima.com/en/products-services/Products/himatrix/HIMatrix%20Module%20Overview

Seznam příloh