VŠB – Technická univerzita Ostrava

Fakulta elektrotechniky a informatiky

Katedra kybernetiky a biomedicínského inženýrství

Testovací aplikace pro ovládání dané zadní svítilny s LIN sběrnicí

Test Application for Control of Rear Lamps with LIN Bus

2020 **Bc. Lukáš Šnajdr**

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra kybernetiky a biomedicínského inženýrství

Zadání diplomové práce

Student: **Bc. Lukáš Šnajdr**

Studijní program: N2649 Elektrotechnika

Studijní obor: 2612T041 Řídicí a informační systémy

Téma: Testovací aplikace pro ovládání dané zadní svítilny s LIN sběrnicí
Test Application for Control of Rear Lamps with LIN Bus

Jazyk vypracování: čeština

Zásady pro vypracování:

V automobilovém průmyslu se v současnosti používají moderní světlomety, které obsahují sběrnice pro komunikaci s řídicí jednotkou vozidla. Cílem práce je vytvoření testovací aplikace pro ovládání automobilové zadní svítilny s LIN sběrnicí pomocí vývojového prostření LabVIEW, která ověří její funkčnost. Náplní této práce je výběr vhodného HW a vývoj SW.

Body zadání:
1. Seznámení a pochopení principů vývojového prostředí NI.
2. Výběr sběrnicové karty a seznámení s jejími funkcemi.
3. Výběr dalšího potřebného HW pro ovládání zadní svítilny.
4. Seznámení a pochopení principů komunikace sběrnice LIN, analýza sběrnice.
5. Návrh testovací aplikace pro možné univerzální využití.
6. Implementace testovací aplikace ve vývojovém prostření LabVIEW.
7. Ověření funkčnosti testovací aplikace.
8. Závěrečné zhodnocení.

Seznam doporučené odborné literatury:

[1] VLACH, Jaroslav, Josef HAVLÍČEK a Martin VLACH. *Začínáme s LabVIEW.* 1. vyd. Ilustrace Viktorie Vlachová. Praha: BEN - technická literatura, 2008, 247 s. ISBN 978-80-7300-245-9.
[2] BRESS, Thomas J. *Effective labview programming.* 1st ed. Allendale: NTS Press, 2013, 701 s. ISBN 19-348-9108-8.

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **doc. Ing. Petr Bilík, Ph.D.**

Datum zadání: 01.09.2019

Datum odevzdání: 30.04.2020

doc. Ing. Jiří Koziorek, Ph.D.
vedoucí katedry

prof. Ing. Pavel Brandštetter, CSc.
děkan fakulty

Prohlášení studenta

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Odrách dne: *15. května 2020*

.....
podpis studenta

Prohlášení zástupce spolupracující právnické nebo fyzické osoby

Souhlasím se zveřejněním této diplomové práce dle požadavků čl. 26, odst. 9 Studijního a zkušebního řádu pro studium v magisterských programech VŠB-TU Ostrava

V Ostravě dne: *15. května 2020*

.....

podpis zástupce

Poděkování

Rád bych poděkoval vedoucímu diplomové práce doc. Ing. Petru Bilíkovi, Ph.D. a konzultantovi Ing. Jiřímu Frýdlovi za odbornou pomoc a konzultaci při vytváření této diplomové práce.

Abstrakt

Cílem této diplomové práce je sestavení HW a vytvoření testovací aplikace pro ovládání zadní svítilny, pomocí programovacího prostředí LabVIEW od firmy National Instruments. Testovací aplikace je univerzální pro různé typy zadních svítilen obsahující LIN sběrnici a veškerý HW je ovládán právě pomocí testovací aplikace.

Klíčová slova

LIN (Local Interconnect Network), LabVIEW, ovládání zadní svítilny.

Abstract

The aim of this diploma thesis is to build HW and create a test application for the control of the rear lamp, using the LabVIEW programming environment from National Instruments. The test application is universal for different types of rear lamps containing LIN bus and all hardware is controlled by just using test applications.

Key words

LIN (Local Interconnect Network), LabVIEW, control of the rear lamp.

Obsah

Seznam použitých symbolů a zkratek

Zkratka	Význam
LED	Light Emitting Diode (elektroluminiscenční dioda)
LIN	Local Interconnect Network
CAN	Controller Area Network
LSB	least significant bit (nejméně významný bit)
BCM	Body Control Modules (centrální řídící jednotka karoserie)
HCM	Headlamp Control Modules (Řídící jednotka světlometu)
ECU	Electronic Control Unit (Elektronická řídicí jednotka)
LDM	LED Driver Modules (Řídicí modul LED)
NI	National Instrument
IVI	Interchangeable Virtual Instruments (zaměnitelné virtuální přístroje)
MSO	Mixed Signal Oscilloscope (Smíšený signál osciloskopu)
ID	Identifikátor
I/O karta	Vstupně-výstupní karta

Seznam ilustrací

1 Úvod

Testování světlometů a svítilen je důležité především při vývoj i při sériové výrobě, než svítilna projde různými testy až k zákazníkovi. K testování během výrobního procesu slouží různé testery, které jsou naprogramované právě na potřebné testy. Tyto testery vyhodnocují funkčnost, ale i stav pro dané testy, které jsou podmínkou.

Moderní světlomety obsahují řídící jednotku světlometu tzv. HCM (Headlamp Control Modules), která přijímá data od centrální řídící jednotky karoserie tzv. BCM (Body Control Modules), kdy data jsou přijímána pomocí komunikační sběrnice LIN nebo CAN. Řídící jednotka světlometu pak na základě přijatých dat provádí spínání, napájení a diagnostiku jednotlivých funkcí světlometů. Ovládací funkce světlometu je pak možné rozdělit na analogové a digitální, kdy analogové funkce jsou funkce spínané pomocí napětí baterie tedy kontaktu a digitální právě pomocí řídící jednotky světlometu.

Cílem této diplomové práce je vytvořit aplikaci pro ovládání svítilen s komunikační sběrnicí LIN a simulovat tak plnohodnotné ovládání svítilny i mimo vozidlo.

Jedním z vhodných programovacích prostředí pro vytváření testovacích aplikací je grafické programovací prostředí LabVIEW od firmy National Instrument. Toto programovací prostředí se používá pro vytváření testovacích aplikací vyžadující testování, měření a kontrolu umožňující rychlé hardwarové a datové nahlédnutí. Grafické programování pomáhá vizualizovat jednotlivé aspekty aplikace, včetně hardwarové konfigurace, naměřených dat a ladění.

2 Světlomety a svítilny

Světlomety i svítilny jsou v dnešní době nejen povinnou součástí výbavy vozu zajišťující viditelnost a bezpečnost, ale také součást designu. Rozdíl mezi světlometem a svítilnou, je že světlomet se nachází na přední straně automobilu a svítilna na zadní straně automobilu.

2.1 Typy světlometů a svítilen

Světlomety a svítilny jsou rozdělovány především podle zdroje použitého světla. U funkce potkávacího a dálkového světla je více typů zdroje světla, kterými jsou halogenové žárovky, xenonové výbojky, LED a laser (pouze pro dálkové světla), zatím co u ostatních funkcí jsou zdrojem světla jen klasické žárovky nebo LED. [1],[2],[3],[4]

2.1.1 Klasické světlomety a svítilny

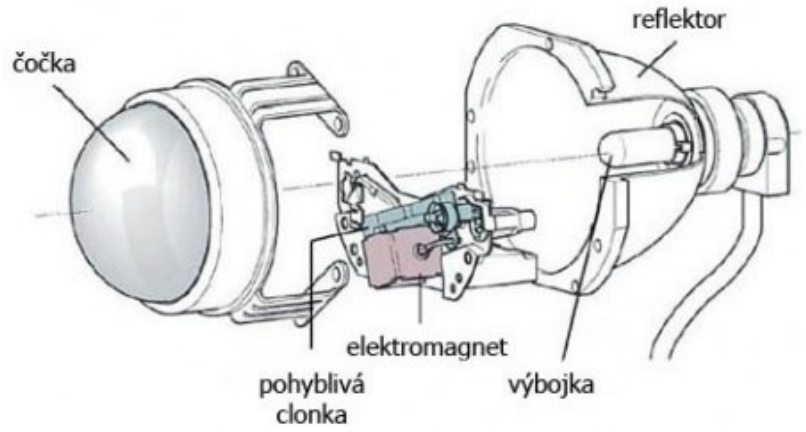
Klasické světlomety využívají jako zdroj světla halogenové žárovky pro potkávací a dálkové světla klasické žárovky pro všechny ostatní funkce. Klasické žárovky jsou používány pro světlomety i svítilny. Zdrojem světla u klasické žárovky je rozžhavené wolframové vlákno uvnitř skleněné baňky, ve které je vakuum. U halogenové žárovky je skleněná baňka navíc naplněná halogenem, to umožňuje halogenové žárovce větší účinnost a životnost. Nerozšířenějším typem halogenové žárovky je typ H7 a H4. Hlavními výhodami těchto světlometů je malá cena, jednoduchost a možná výměna žárovky, ale nevýhodou je životnost, velká spotřeba elektrického proudu, menší osvícení. U novějších halogenových světlometů jsou klasické žárovky nahrazeny jiným zdrojem světla, tedy LED a postupně jsou i celkově tyto typy nahrazovány LED světlomety. [1],[2],[3],[4]

Obrázek 2.1: *Halogenové světlomet v kombinaci s LED [4]*

2.1.2 Xenonové světlomety

V xenonových výbojkách je světlo vyzařováno obloukem mezi elektrodami ve skleněné baňce, která je naplněna xenonem. Výhodou je široký paprsek světla a nízký odběr elektrického proudu, nevýhodou je ale cena a s ní spojený servis. Xenonové výbojky se používají jen pro funkce potkávacího nebo dálkového světla. Ve světlometech se používají buď v kombinaci s halogenovou žárovkou, nebo tzv. Bi-xenonové světlomety. Bi-xenonové světlomety umožňují použití jedné xenonové výbojky pro dálkové i potkávací světla, kdy v uvnitř světlometu je umístěna elektromagnetická clonka, která se při přepínání mezi těmito funkcemi posouvá a výbojka tak neustále svítí. Konstrukce Bi-xenonového světlometu je znázorněna na obrázku 2.3. [1],[2],[3],[4],[5],[6]

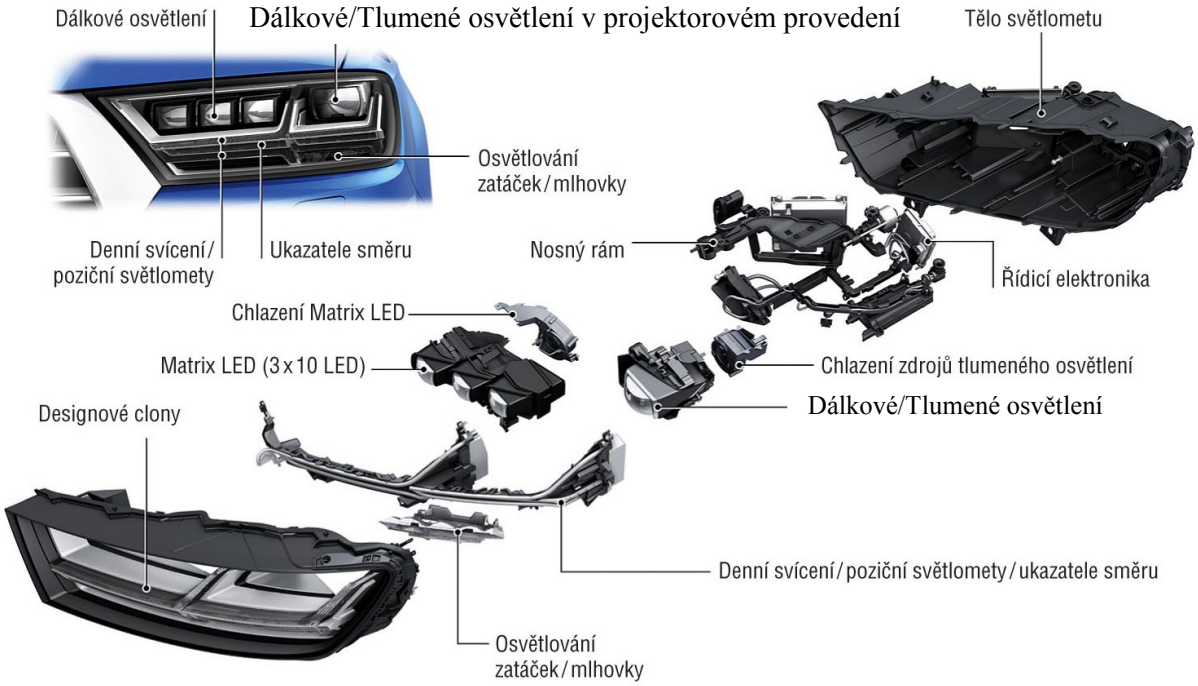
Obrázek 2.2: *Xenonový světlomet v kombinaci s halogenovou žárovkou [5]*



Obrázek 2.3: *Konstrukce Bi-xenonového světlometu [6]*

2.1.3 **LED světlomety a svítilny**

Zdrojem světla jsou elektroluminiscenční diody tzv. LED. LED je polovodivá součástka s přechodem typu P-N, která při průchodu proudu tímto P-N přechodem emituje světlo. LED jsou výhodné především díky malým rozměrům a účinnosti, která je cca 10 krát větší než klasická žárovka, dalšími výhodami jsou, různá barva světla, vysoká úroveň jasu, životnost, malé napájecí napětí. Nevýhodou je však vysoká výrobní cena celého světlometu, neopravitelnost, nerozebíratelnost světlometu a při vyšších výkonech i nutnost chlazení. LED jsou díky svým vlastnostem použitelné pro všechny světelné funkce světlometu i svítilny. Zatím co dříve byly využívány v kombinaci s halogenovými žárovkami nebo xenonovými výbojkami pro funkci dálkového světla popřípadě potkávacího světla, dnes se používají LED i pro tuto funkci a jsou tak kompletovány samostatně, nebo nově v kombinaci s laserem. LED světlomety tak v dnešní době nahrazují předešlé zdroje světla. Pro funkci dálkového a potkávacího světla jsou používány LED v projektorovém provedení tzv. BI-LED modul, který je zobrazen i na obrázku 2.4. Princip je v tomto případě podobný jako u Bi-xenonových světlometů akorát se uvnitř nenachází elektromagnetická clonka, ale jsou rozsvěcovány jednotlivé LED dle potřeby. [1],[2],[3],[4]

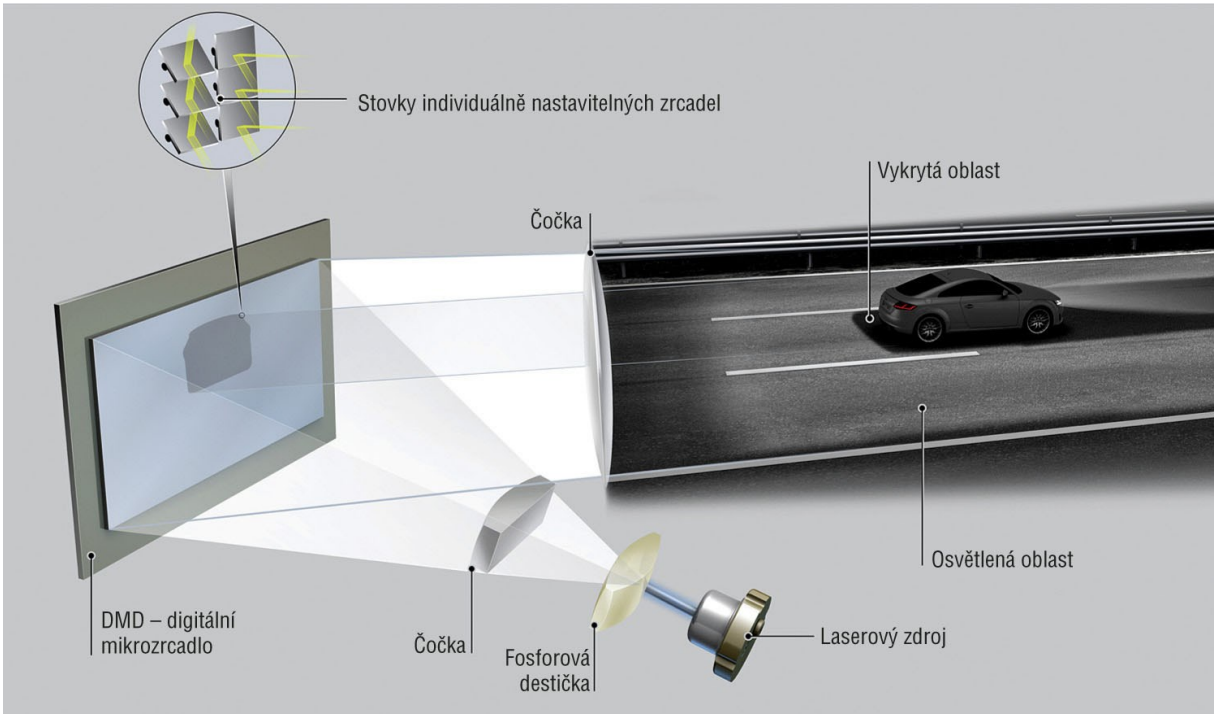


Obrázek 2.4: *LED světlomet a jeho kontrukce [7]*

Obrázek 2.5: *LED svítilna [8]*

2.1.4 **Laserové světlomety**

U novodobých světlometů je laser používán pouze pro funkci dálkového světla jako doplněk LED světlometů. Výhodami laseru jsou vysoká světelná účinnost a větší osvícení vozovky, bodový světlený tok, nevýhodami jsou však bezpečnostní riziko, omezené podmínky pro zapnutí, vyšší napětí a nutnost vícestupňové regulace. Paprsek laseru je vyzařován přes konvertor s čočkou, který mění vyzařovanou vlnovou délku, na soustavu zrcadel, od nichž se odráží přes čočku na vozovku, tak jak je znázorněno na obrázku 2.6. [1],[2],[3],[4]



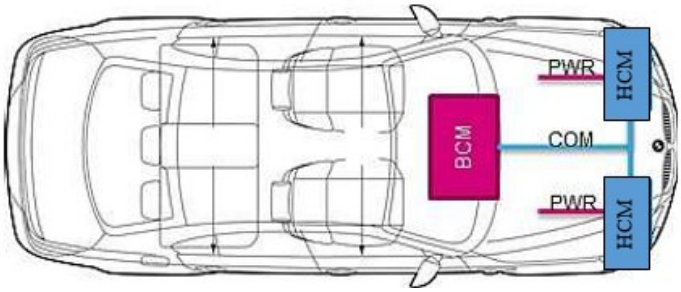
Obrázek 2.6: *Konstrukce Laserového světlometu [7]*

2.2 Funkce

Mezi základní světelné funkce pro světlomety patří potkávací (tlumené), dálkové, mlhová, směrová, obrysová světla a pro svítilny to jsou brzdová, obrysová, mlhová, směrová a parkovací světla. [1]

2.3 Ovládání

O ovládání moderního světlometu se stará řídící jednotka světlometu tzv. HCM (Headlamp Control Modules) společně s centrální řídící jednotkou karoserie tzv. BCM (Body Control Modules). Kdy jednotka BCM je hlavní jednotka, která se řídí všechny funkce karoserie, jako jsou například veškeré světla, okna, zámky dveří a další, získává tak data od různých senzorů jako např. kamera, natočení volantu, rychlost a podle těchto dat pak vyhodnocuje ovládání všech funkcí karoserie i funkcí světlometu tak aby byl zajištěna co nejlepší viditelnost a bezpečnost. BCM jednotka je tak centrální jednotka typu Master ke které jsou připojené jednotlivé tzv. ECU (Electronic Control Unit) jednotky typu Slave například právě jednotka HCM, které pak obsahují elektronické obvody pro ovládání, spínání, napájení a diagnostiku jednotlivých funkcí. BCM jednotka ovládá jednotlivé ECU jednotky pomocí komunikace po sběrnici LIN nebo CAN.



Obrázek 2.7: *Topologie pro ovládání světlometů [9]*

Jednotka HCM je jednotka, která je součástí světlometů a přijímá data od BCM jednotky po sběrnici LIN nebo CAN, provádí tak spínání digitálních funkcí světlometů. Uvnitř této jednotky se nachází elektronické obvody pro ovládání, spínání, napájení a diagnostiku jednotlivých funkcí. Součástí ovládání světlometu jsou i elektronické obvody, pro diagnostiku detekce výpadku nebo změn teploty, které pak upravují napájení LED, jedná se o LED řídící moduly tzv. LDM (LED Driver Modules).

Obrázek 2.8: *Řídící jednotka světlometu (HCM) [11]*

Ovládací funkce světlometu je možné rozdělit na analogové a digitální.

2.3.1 **Analogové funkce**

Analogové funkce jsou funkce, které jsou ovládány klasicky přivedením napětím baterie automobilu, pomocí kontaktu přes relé či jiné spínací prvky. Mezi tyto funkce patří například brzdová a směrová světla. Vlastnosti těchto funkcí však můžou být ovlivňovány řídící jednotkou světlometu, jako například intenzita vyzařovaného světla a animace.

2.3.2 **Digitální funkce**

Digitální funkce jsou spínány jednotkou HCM, pomocí komunikace po sběrnici LIN nebo CAN od centrální jednotky BCM. Celá komunikace po sběrnice je řízená a ovládána přímo pomocí souboru databáze dané svítilny nebo světlometu. Databáze světlometu/svítilny je v podstatě zdrojový kód pro komunikaci po sběrnici s daným světlometem nebo svítilnou, obsahuje definovanou sadu příkazů, jejich parametry a také základní parametry komunikační sběrnice, jako jsou například rychlost a verze sběrnice. Digitální funkce jsou tak ovládány pomocí příkazů z databáze svítilny nebo světlometu, jejichž funkce jsou dále popsány ve specifikaci světlometu/svítilny od výrobce. [9],[10],[11]

3 LIN sběrnice

LIN je nízkonákladová sériová komunikační sběrnice, využívána především v automobilovém průmyslu. Tato sběrnice vznikla za účelem doplnění CAN sběrnice, která byla pro některé funkce automobilů příliš nákladná. LIN sběrnice je tedy především využívána pro jednodušší funkce, kde není zapotřebí vyšší rychlosti ovládání a bezpečnosti automobilů, jako je například ovládání svítilen, řízení klimatizace, zámků, oken, zrcátek a další.

3.1 Historie

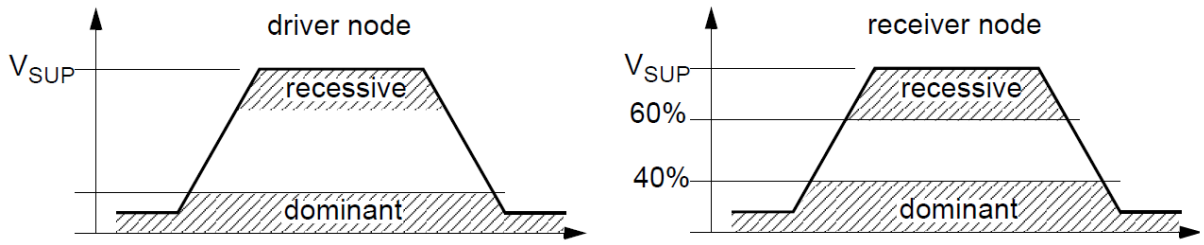
První specifikace byla vydána v červenci 1999, jako verze LIN 1.0 po té došlo k dalším 2 aktualizacím především ve fyzické vrstvě, až vznikla verze LIN 1.3. Vlivem evolučního růstu vznikla verze LIN 2.0, kde byly opraveny chyby předešlé verze a dodány nové funkce pro konfiguraci a diagnostiku. Vlivem dalšího evolučního růstu vznikly nové verze LIN 2.1 a LIN 2.2, které obsahují nové funkce a další rozšíření a jsou zpětně kompatibilní z předešlými verzemi LIN 2.x.

3.2 Vlastnosti

Komunikace po LIN sběrnici je typu Master-Slave konkrétně jedna jednotka typu Master a jedna a více jednotek typu Slave, přičemž maximální počet LIN Slave jednotek je 16. Jedná se o nízkonákladovou sběrnici založenou na běžném rozhraní UART / SCI. Pro komunikací využívá jedno vodičově provedení s napěťovou hladinou 12 V. Rychlost komunikace je až 20 kb/s a velikost zprávy může být 2, 4 nebo 8 Bajtů. LIN také obsahuje transportní vrstvu a podporu diagnostiky. [12],[16],[17]

3.3 Fyzická vrstva

LIN sběrnice byla navržena především pro automobilový průmysl na základě standardu ISO 9141, který byl vyvinut pro diagnostické účely automobilů. LIN je obousměrná sběrnice komunikující po jednom vodiči, což snižuje náklady, ale zvyšuje pravděpodobnost elektromagnetického rušení, proto se používají tři základní hodnoty rychlosti sběrnice, kterými jsou 2400, 9600 a 19200 kb/s. Úrovně signálu jsou vztaženy k napětí v automobilu, generovanému baterií 12 V. Logická 0 (dominantní hodnota) je implementována jako GND a logická 1 (recesivní hodnota) jako napětí na baterii automobilu tedy přibližně 12 V. Pro vysílač je logická 1 od 80 % do 100% napětí na baterii a logická 0 od 0 % do 30 %. U přijímače je rozmezí hodnot větší pro logickou 1 od 60 % do 100% napětí na baterii a logická 0 od 0 % do 40 %. [12],[13],[16],[17]



Obrázek 3.1: *Fyzická vrstva úrovně signálů [13]*

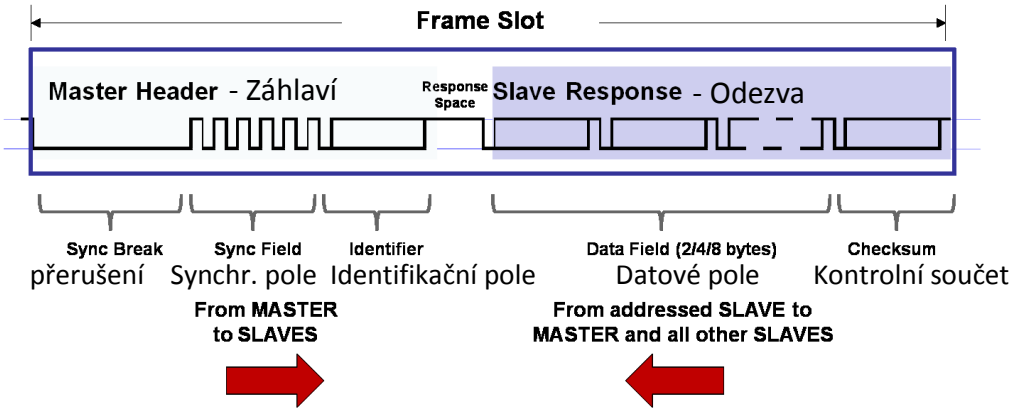
3.4 LDF (LIN Description File)

LDF je soubor databáze dané komponenty, ve kterém jsou definované jednotlivé parametry komunikace, jako jsou například rychlost a verze komunikační sběrnice, informace a parametry ECU jednotek, rámců a signálů. LDF je tedy textový soubor definující celou sadu příkazů (signálů) pro komunikaci po sběrnici LIN. Celý tento soubor je určen jen pro stanovenou HW komponentu, například pro jeden typ světlometu. [12],[14]

3.5 LIN rámec

Komunikace po LIN sběrnici je typu Master-Slave, komunikace probíhá pomocí zpráv, které jsou posílány jednotkou typu master. Jednotlivé jednotky typu Slave čekají na pokyny jednotky typu Master. Master jednotka je jen jedna a řídí celou komunikaci. Jednotlivé zprávy můžou být určeny pro zápis nebo pro vyčtení dat.

Každá zpráva po LIN sběrnici je odesílána v rámečkovém slotu. Tento rámec obsahuje 2 hlavní části, kterými jsou záhlaví (Master Header) a odezva (Slave Response), ty se pak rozdělují na další pole.



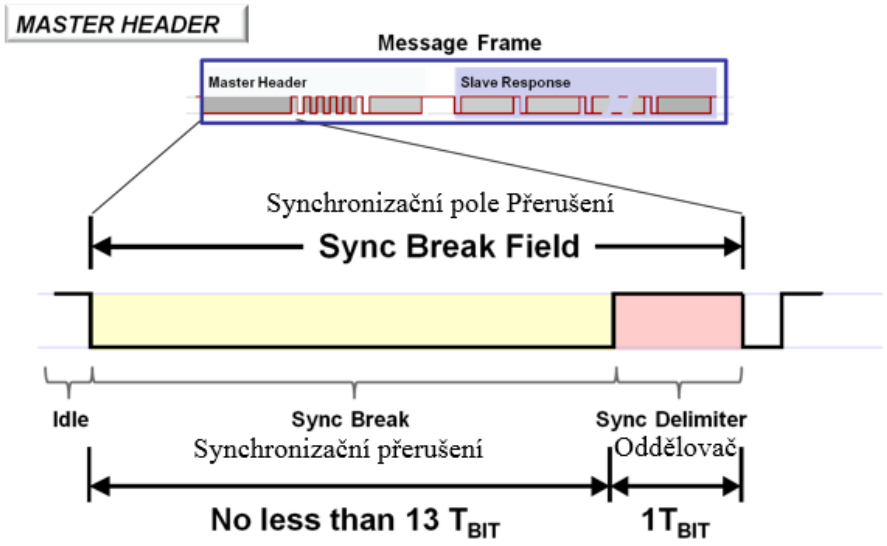
Obrázek 3.2: *Rámec zprávy [15]*

3.5.1 Záhlaví (Master Header)

Záhlaví je část, která je vždy přenášená jednotkou typu Master a obsahuje synchronizační pole přerušení (Sync Break), synchronizační pole (Sync Field) a identifikační pole (identifier).

Synchronizační pole přerušení (Sync Break Field)

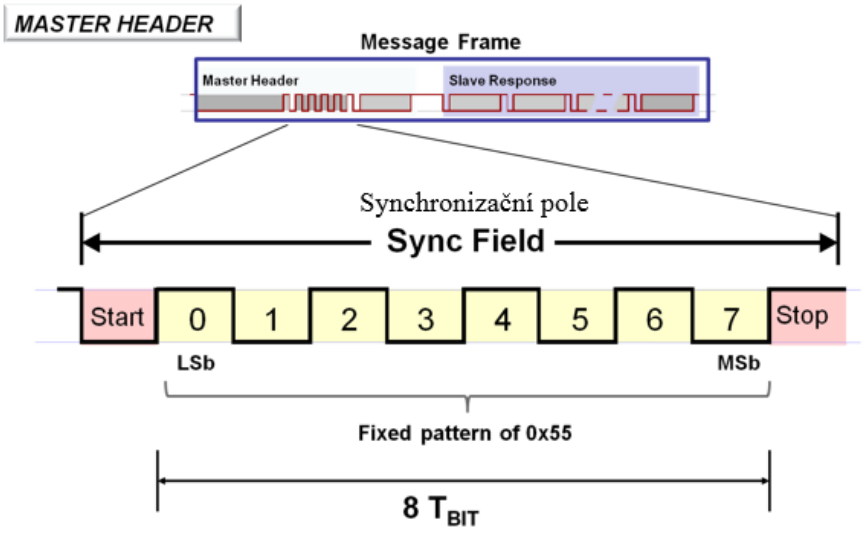
Jedná se o podmínku nutnou pro začátek každé zprávy, která slouží jako oznámení pro všechny Slave jednotky. Přerušení obsahuje minimálně 13 dominantních bitů (Logická 0), po té následuje oddělovač (break delimiter) obsahující minimálně 1 nominální bit (logická 1).



Obrázek 3.3: *Přerušení [15]*

Synchronizační pole (Sync Field)

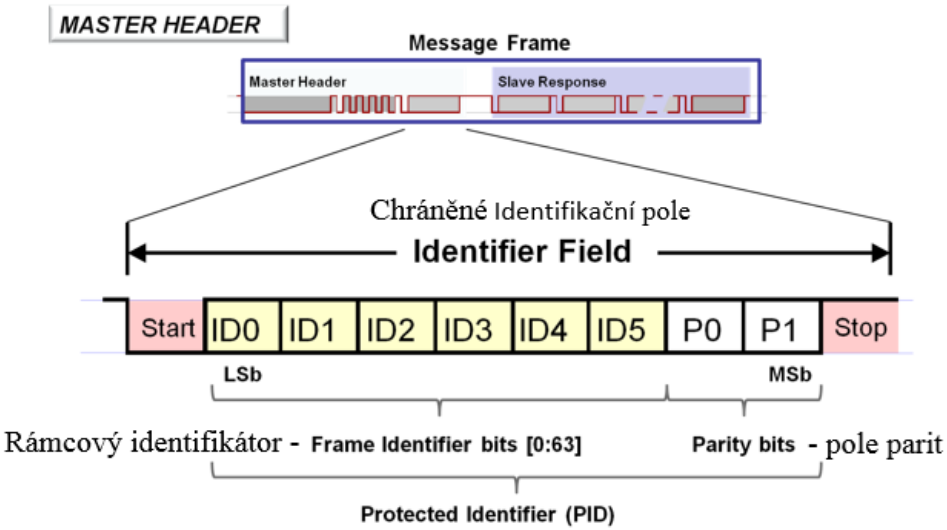
Synchronizační pole slouží pro synchronizaci přenosové rychlosti mezi jednotkou Master a Slave, Kdy jednotka Slave automaticky detekuje přenosovou rychlost a přizpůsobí se této přenosovou rychlosti. Synchronizační pole je definováno jako bajtové pole s datovou hodnotou 0x55.



Obrázek 3.4: *Synchronizační pole [15]*

Chráněné identifikační pole (PID)

Chráněné identifikační pole je rozděleno do dvou dílčích polí rámcový identifikátor (Frame identifier) a pole parit (Parity). Pro rámcový identifikátor (ID) je vyhrazeno 6 bitů, tedy 0 až 5 bit a může nabývat hodnot 0 až 63. Dle této hodnoty jsou rámce rozděleny do 3 kategorií. Hodnoty 0 až 59 slouží pro rámce přenášející signály, 60 a 61 pro diagnostické data a 62 a 63 pro budoucí vylepšení LIN protokolu. Tento identifikátor obsahuje informace odezvě pro Slave jednotku, tedy jestli bude data přijímat nebo zapisovat a o velikosti dat. Identifikátor 0 až 31 má velikost dat 2 B, 32 až 47 má 4 B a 48 až 63 má 8 B. Pro pole parit jsou vyhrazeny 2 bity.



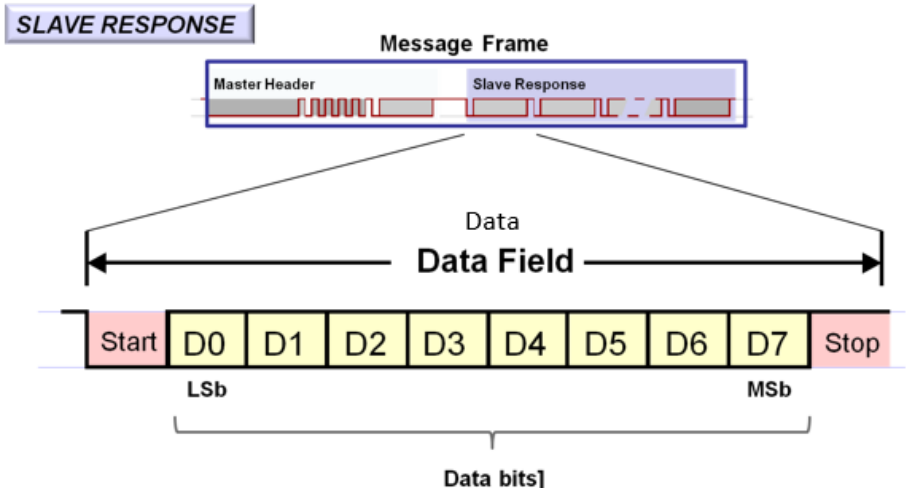
Obrázek 3.5: *Chráněné identifikační pole [15]*

3.5.2 Odezva (Slave Response)

Odezva je část určená pro Slave jednotku, která buď z této části přímá data, nebo je zapisuje. Odezva obsahuje pole data a kontrolní součet (Cheksum).

Data

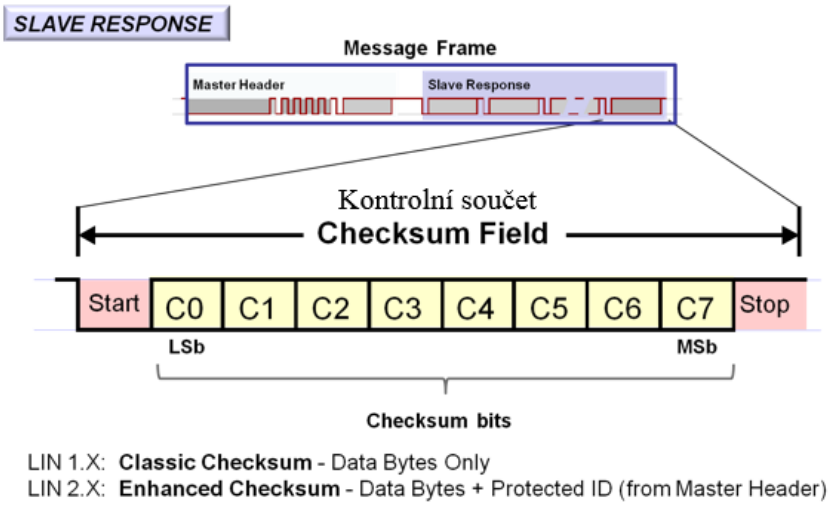
Na základě předchozího chráněné identifikační pole byly stanoveny parametry pro data a nyní se v poli data provede přenos dat jednotlivých signálu. Velikost dat je určena identifikátorem a může mít hodnotu 2, 4 nebo 8 B. V jednom rámci může být přenášeno několik signálu, přičemž jsou nejprve přenášeny LSB data.



Obrázek 3.6: *Data [15]*

Kontrolní součet (Cheksum)

Kontrolní součet slouží ke kontrole datového pole a celé odezvy. Pro verzi LIN 1.X je používán klasický kontrolní součet, kde jsou sečteny všechny data a pro verzi LIN 2.X rozšířený kontrolní součet, kde jsou sečteny všechny data a je přičtena i hodnota chráněného identifikátoru. Výjimkou jsou rámce s hodnotou identifikátoru 60 až 63, u kterých musí být použitý vždy kontrolní součet. [12],[15],[16]



Obrázek 3.7: *Kontrolní součet [15]*

4 LabVIEW

LabVIEW je grafický programovací jazyk, pro vytváření aplikací vyžadující testování, měření a kontrolu umožňující rychlé hardwarové a datové nahlédnutí. Grafické programování pomáhá vizualizovat jednotlivé aspekty aplikace, včetně hardwarové konfigurace, naměřených dat a ladění. Tato vizualizace usnadňuje integraci měřicího hardwaru, představuje komplexní logiku v diagramu, vyvíjí algoritmy analýzy dat a navrhuje uživatelská rozhraní pro technické inženýrství. LabVIEW se řídí základním principem datového toku tzv. dataflow, což je propojení jednotlivých bloků, tak aby byl vytvořen logický průběh, kdy každý blok se vykoná až má k dispozici všechny vstupní data. [18],[20]

4.1 NI VISA

NI Visa je softwarová vrstva, která vytváří unifikované API pro komunikaci mezi přístroji a PC v LabVIEW, bez ohledu na to pomocí kterého rozhraní je propojení realizováno, poskytuje tak nezávislost na typu rozhraní. Propojení může být například realizováno pomocí GPIB, USB, sériového rozhraní a Ethernet. [19],[21],[22]

4.1.1 Komunikace pomocí VISA

Komunikace je tvořena příkazy nebo registry. Každý přístroj, který umožňuje ovládání pomocí LabVIEW má vytvořený programovací manuál, ve kterém jsou vypsány jednotlivé příkazy nebo registry a jejich činnost a parametry. Některé příkazy jsou definovány standardem IEEE488.2 a musí je obsahovat každý přístroj, jsou to například *IDN?, *RST, a další. *IDN? Je příkaz pro zobrazení základních informací o přístroji a *RST pro resetování přístroje. Příkazy, které obsahují na konci znak „?“ jsou příkazy, po kterých je očekáváná odpověď od přístroje, která musí být po zápisu přečtena pomocí bloku VISA Read.

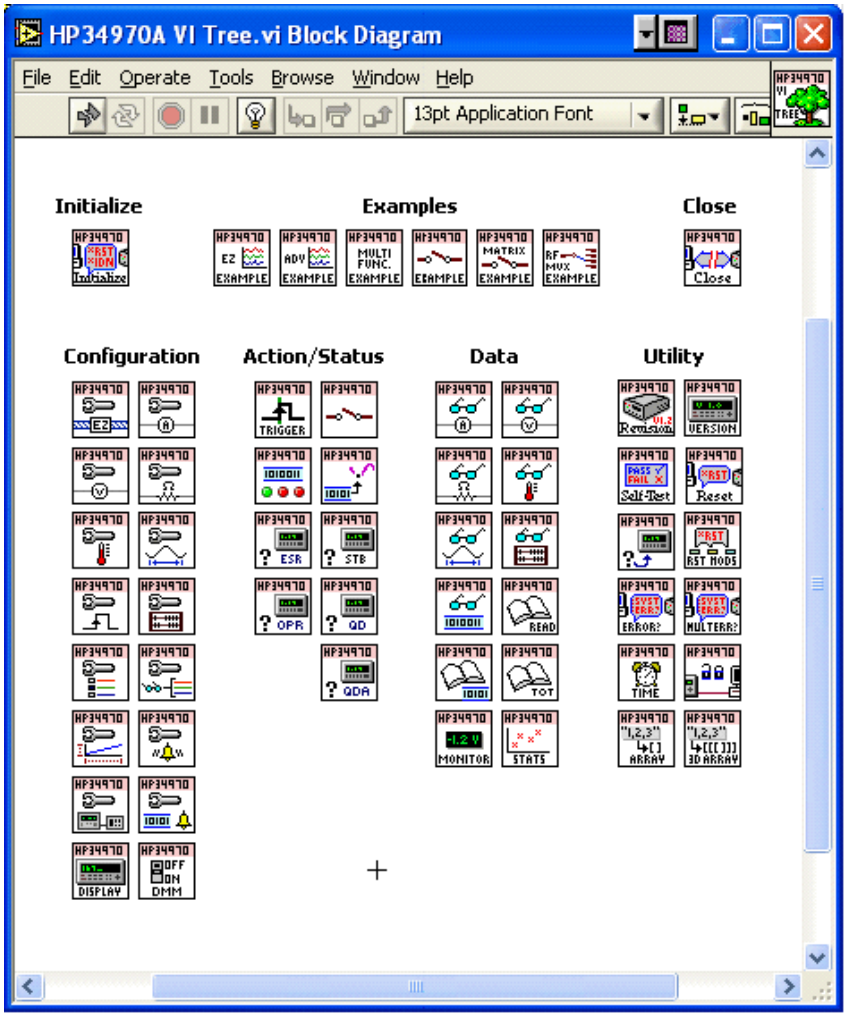
Obrázek 4.1: *Základní kód pro komunikaci s přístrojem pomocí příkazů [23]*

Na obrázku 4.1 je základní kód pro komunikaci s přístrojem, konkrétně pro příkaz *IDN?, s následujícím postupem

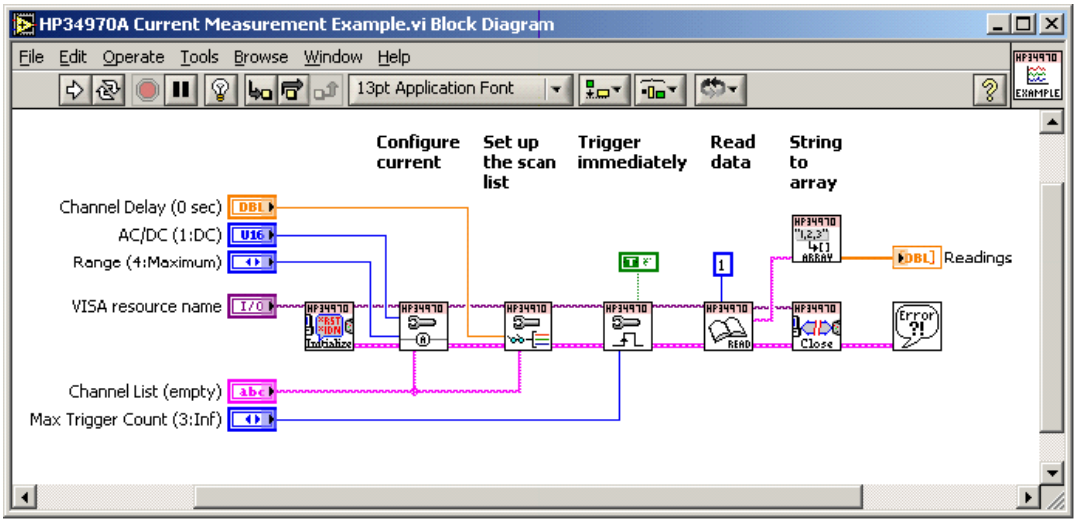
1. Vytvoření VISA relace, pomocí bloku VISA Open.
2. Zápis zprávy, pomocí bloku VISA Write.
3. Přečtení zprávy, pomocí bloku VISA Read.
4. Ukončení VISA relace, pomocí bloku VISA Close. [19],[21],[22],[23],[24]

4.1.2 Plug and Play ovladače

Pro ovládání programově řízených přístrojů se používají výrobcem vytvořené ovládací přístroje. Tyto přístrojové ovladače pak představují funkce a architekturu ovládání přístroje. Proto, aby byla sjednocena struktura těchto přístrojových ovladačů byla založena VXI plug&play Systems Alliance, která stanovila hlavní požadavky pro přístrojové ovladače. Hlavními požadavky jsou plná kontrola nad funkcemi, otevřený kód, modulární hierarchické uspořádání, jednotný design a implementace, kontrola chyb, dokumentace, nápověda a kontrola verzí. Pokud jsou tyto požadavky splněny, jedná se o Plug and play přístrojový ovladač. Plug and play ovladače jsou většinou na webových stránkách výrobce přístroje ke stažení a do LabVIEW se musí instalovat. Po instalaci ovladače v LabVIEW používat a nachází se v paletě funkcí Instrument I/O/Instr Drivers/ kde jsou zabaleny ve složce s názvem přístroje. Zde se nachází i subVI VI Tree.vi, ve kterém jsou jednotlivé ovladače rozděleny do jednotlivých oblastí podle vnitřní struktury, těmi jsou inicializace, konfigurace, status, data, utility a ukončení relace.



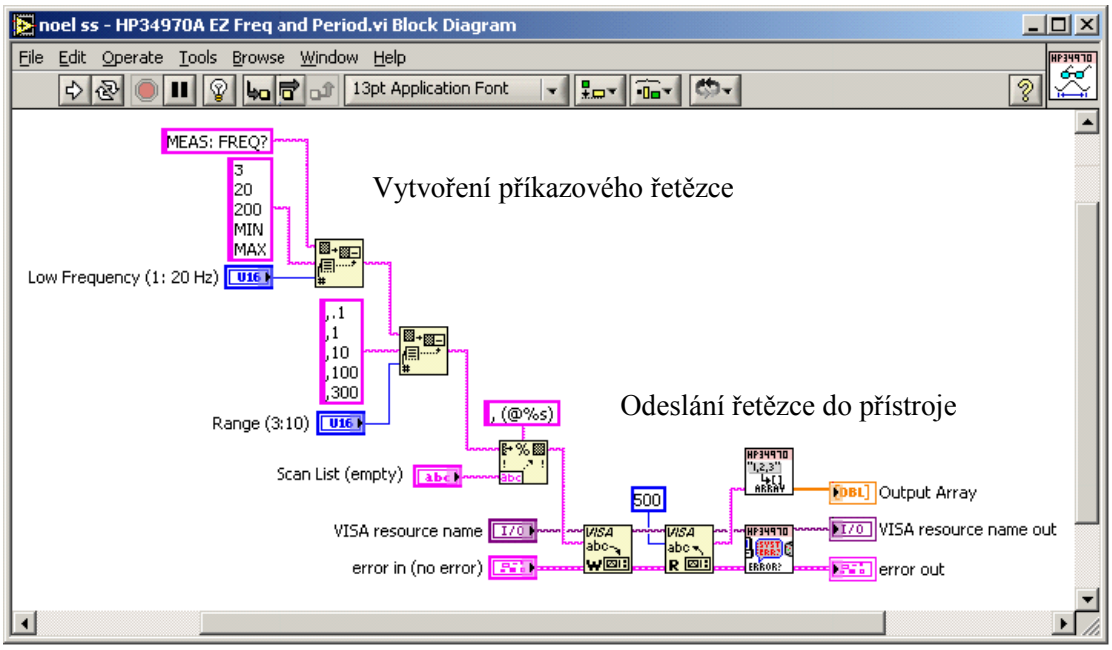
Obrázek 4.2: *Ukázka vzorového VI Tree.vi pro přístroj HP34970A [24]*



Obrázek 4.3: *Blokový digram vzorového příkladu pro měření proudu HP34970A [24]*

Na obrázku 4.3 je příklad použití jednotlivých plug and play ovladačů pro měření proudu u přístroje HP34970A. Postup tohoto příkladu je:

1. Inicializace
2. Konfigurace proudu
3. Nastavení scan listu
4. Trigger konfigurace
5. Přečtení dat
6. Ukončení relace



Obrázek 4.4: *Příklad příkazového řetězce pro konfiguraci frekvence a periody pro HP34970A [24]*

Obrázek 4.4 ukazuje zdrojový kód jednoho ovladače, konkrétně pro konfiguraci frekvence a periody pro přístroj HP34970A. Ve zdrojovém kódu jsou použity standartní funkce LabVIEW a VISA k vytvoření příkazových řetězců a následné odeslání do přístroje.

4.1.3 IVI ovladače

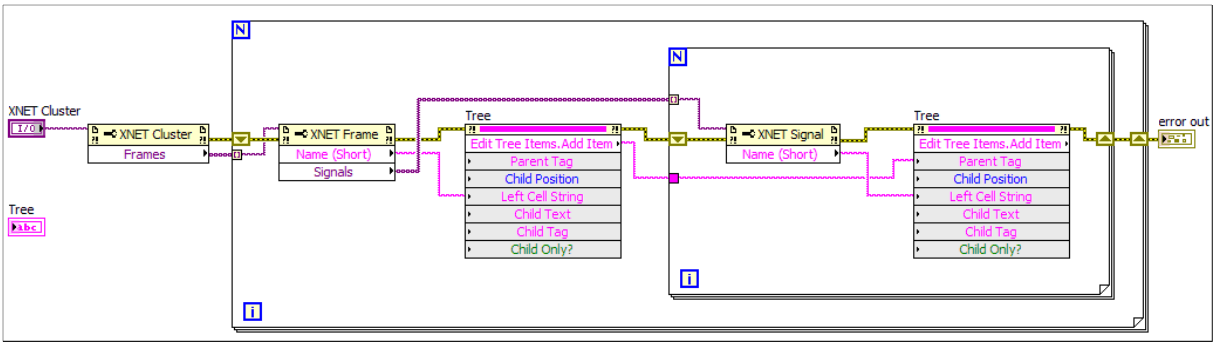
IVI ovladače vznikly za účelem dosáhnutí nezávislosti na konkrétním modelu přístroje a umožnit tak aplikacím změnu modelu přístroje bez zásahu do zdrojového kódu. IVI ovladače jsou specifikovány konsorciem IVI Foundation, které sdružuje mnoho významných výrobců měřících přístrojů, softwaru a systémových integrátorů. IVI ovladače jsou zaměřeny na poskytnutí lepšího výkonu, usnadnění přenositelnosti kódu aplikace na jiný přístroj dané třídy a ušetření času a nákladů spojených s vývojem aplikace. Ve srovnání s Plug and play ovladači, přišla tak IVI Foundation s dalším rozšířením přístrojových ovladačů o zaměnitelnost přístrojů v rámci dané třídy přístrojů, kdy struktura těchto ovladačů je právě postavena na specifikaci pro VXI Plug and Play ovladače. [19],[21],[22],[24]

4.2 NI X-NET.

NI-XNET je SW ovladač LabVIEW, který umožňuje vytvořit síťovou komunikaci, po sběrnici typu LIN a CAN. U ovladačů NI-XNET je komunikace řízená přímo pomocí souboru databáze dané komponenty, kdy se samotný ovladač stará o časování a realizaci celé komunikace, uživatel má poté možnost zapisovat nebo vyčítat hodnoty příkazů (signálů). Díky tomu je programování i realizovaní aplikace jednodušší a je umožněno tak univerzální použití pro více typů hw komponent, kdy má každá svou vlastní databázi. Ovladače NI-XNET, taky sjednocují všechny své HW komponenty, jako jsou jednotlivé karty sběrnic a zařízení jako PXI, NI CompactDAQ a NI CompactRIO, tak aby bylo zajištěno použití těchto ovladačů pro všechny stejné. Při sestavování bloků je nutností dodržení předepsaného postupu, kdy prvním blokem musí být vždy blok pro vytvoření relace, potom následuje blok pro čtení nebo zápis, dle vytvořené relace a nakonec blok pro ukončení relace. Mezi prvním a posledním blokem se můžou nacházet další bloky pro různé úpravy či nastavení parametrů. [19],[25],[26]

4.2.1 Databáze komponenty

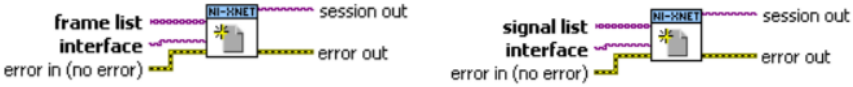
Databáze komponenty je soubor konkrétní HW komponenty, ve kterém jsou definované jednotlivé parametry a příkazy komunikace, jako jsou například rychlost a verze komunikační sběrnice, informace a parametry ECU jednotek, rámců a signálů. S databází komponenty pak spolupracují ovladače NI-XNET pro realizaci komunikace po sběrnici LIN nebo CAN. Pro použití databáze komponenty jsou použity odkazy v paměti nazvané „alias“. Tyto odkazy se vytváří automaticky po zadání cesty k souboru databáze komponenty, pro další použití. Databáze komponenty je rozdělená na jednotlivé pod části, v tomto pořadí klastr, ECU, rámec a signál. Každá databáze komponenty obsahuje jeden nebo více klastrů, které představují kolekci HW jednotek sdílené přes jeden kabelový svazek. ECU je elektronicky řízená jednotka a představuje tedy právě jednu HW jednotku, přičemž v každém klastru se může nacházet jedna nebo více ECU jednotek. Rámec je v podstatě čtená nebo zapisovaná zpráva obsahující jednotlivé předepsané pole, na základě kterých komunikace funguje. Součástí každého rámce jsou signály, které mají v poli data přiřazenou svou velikost a počáteční a konečný bit. V programu LabVIEW je pro každou pod část databáze komponenty k dispozici uzel vlastností, který umožňuje z každé pod části zapsat nebo vyčíst jednotlivé údaje.



Obrázek 4.5: *Příklad kódu pro vyčtení názvů jednotlivých pod částí databáze do stromu [27]*

4.2.2 XNET Vytvoření relace

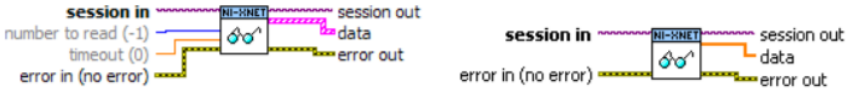
K vytvoření relace slouží blok XNET create session.vi, tento blok umožňuje nastavit relaci do režimu čtení nebo zápis a nastavit formát vstupních dat a výsledných zapisovaných nebo čtených dat. Vstupní data můžou být ve formátu signálů, rámců a výstupní ve formátu číselného pole hodnot signálů, pole hodnot rámce nebo grafické pole. Vstupními parametry jsou interface, seznam signálů/rámec a chybový cluster. Interface je název používané sběrnice, přičemž každý interface lze přiřadit pouze k jedné relaci. Výstupními parametry je založená relace a chybový cluster.



Obrázek 4.6: *XNET vytvoření ralace [25]*

4.2.3 XNET Čtení

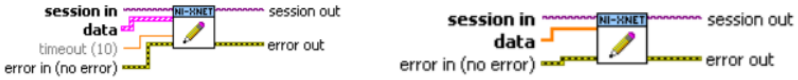
Pro čtení slouží blok XNET read.vi, který jednorázově vyčítá hodnoty ve formátu stanoveném při vytvoření relace.



Obrázek 4.7: *XNET čtení [25]*

4.2.4 XNET Zápis

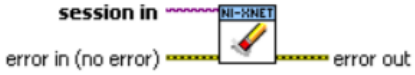
Pro zápis slouží blok XNET write.vi, který jednorázově zapisuje hodnoty ve formátu stanoveném při vytvoření relace.



Obrázek 4.8: *XNET zápis [25]*

4.2.5 XNET Ukončení relace

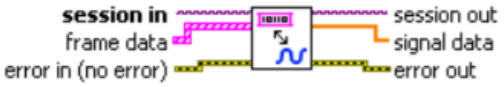
K ukončení relace slouží blok XNET clear.vi, pomocí kterého je ukončena jakákoliv relace a interface je tak uvolněn pro vytvoření nové relace.



Obrázek 4.9: *XNET ukončení relace [25]*

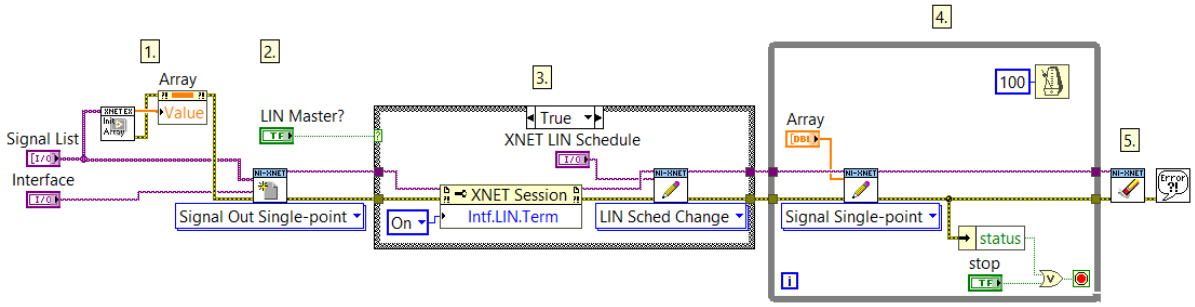
4.2.6 XNET Změna formátu

Změna formátu výsledné hodnoty lze provést pomocí bloku XNET convert.vi. Jedná se především o změnu formátu z tvaru signálu a jejich hodnot na rámec, nebo opačně. [19],[25]



Obrázek 4.10: *XNET Změna formátu [25]*

4.2.7 XNET Vzorový příklad



Obrázek 4.11: *Vzorový příklad zdrojového kódu pro komunikaci po LIN sběrnici [28]*

Na obrázku 4.11 je zobrazen Vzorový příklad zdrojového kódu pro komunikaci po LIN sběrnici, pomocí zapisování dat ve formátu signálů s následujícím postupem

1. Nastavení výchozích hodnot do vstupního pole hodnot signálů
2. Vytvoření relace XNET v režimu zápis s formátem vstupních dat ve formě signálů
3. Nastavení rozhraní na Master nebo Slave dle přepínače „LIN Master?“. V případě Master následné nastavení LIN schedule.
4. Zapisování hodnot signálů ve smyčce do stisknutí tlačítka stop
5. Ukončení relace XNET

Tento příklad je součástí vzorových příkladů v LabVIEW nazvaný „LIN Signal Output Single Point.vi“. [28]

5 Požadavky na testovací aplikaci

Pro tuto diplomovou práci byly stanoveny jednotlivé požadavky, které musí testovací aplikace splňovat.

- Univerzálnost – použití pro jakoukoliv zadní svítilnu s LIN sběrnicí
- Výběr HW, tak aby bylo možné další případné rozšíření
- HW musí být kompatibilní s LabVIEW
- Komunikace po LIN sběrnici
- Spínání analogových funkcí zadní svítilny
- Ovládání zdroje pro napájení zadní svítilny
- Dekódování komunikace po LIN sběrnici pomocí osciloskopu
- Nastavení sekvence – uživatelem vytvořený seznam událostí s určitou posloupností, podle kterých může být svítilna i HW automaticky ovládán

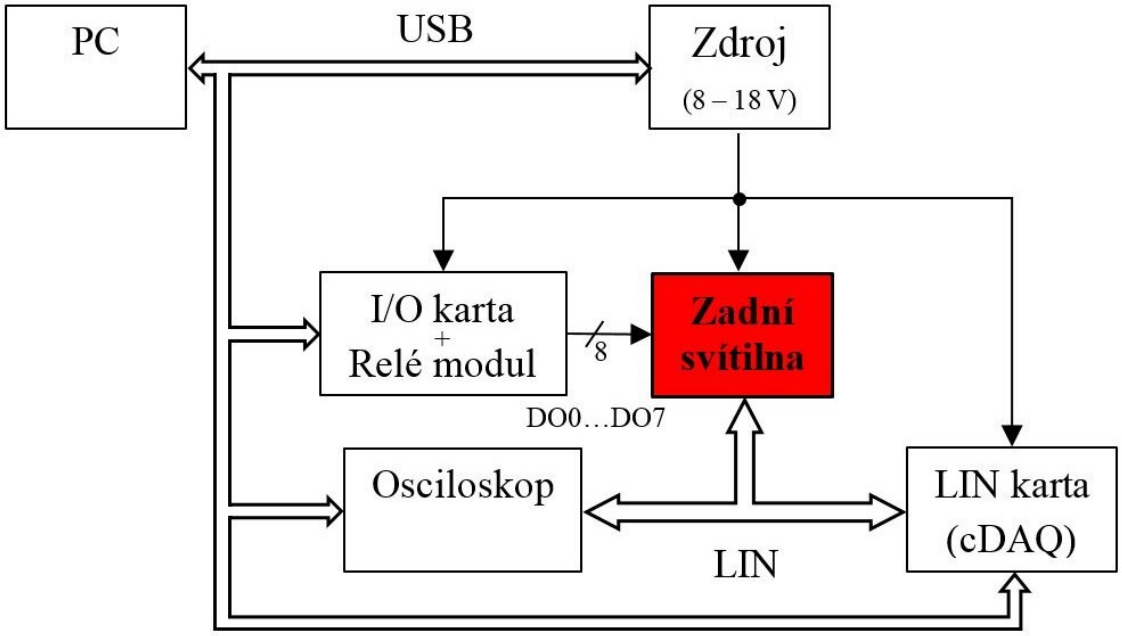
Testovací aplikace bude sloužit pro ovládání zadní svítilny s LIN sběrnicí a koncepce HW, kdy ovládání svítilny je důležité především pro otestování její funkčnosti a dalších případných testů. Ovládání takové svítilny je realizováno pomocí klasických analogových funkcí, a také digitálních funkcí, ke kterým slouží LIN sběrnice. Do koncepce HW patří laboratorní zdroj, I/O karta, LIN karta a osciloskop. Posledním bodem je nastavení sekvence, což je automatické ovládání svítilny a koncepce HW dle předepsaného postupu.

Obrázek 5.1: *Obrázek zadní svítilny*

Na obrázku 5.1 je zobrazena zadní svítilna, pomocí které byla aplikace vyvíjená a testována.

6 Koncepce hardware

Pro popisovanou práci byl vybrán hardware, tak aby bylo umožněno ovládání celého testu jen pomocí vytvořené aplikace v programu LabVIEW. Jednotlivé komponenty hardwaru jsou vybaveny rozhraním USB umožňující připojení k počítači a jsou plně kompatibilní s LabVIEW. Jednotlivými komponenty jsou zdroj, vstupně-výstupní karta s doplňujícím HW, osciloskop a LIN karta umístěná v CompactDAQ šasi.



Obrázek 6.1: *Blokové schéma HW*

Blokové schéma HW znázorňuje zapojení jednotlivých komponent HW. Jak je vidět na obrázku 6.1, PC zajišťuje hlavní funkci testování zadní svítilny pomocí testovací aplikace s připojenými jednotlivými komponenty, kterými jsou Laboratorní zdroj, vstupně-výstupní karta (I/O karta), Osciloskop, LIN karta, k těmto komponentám je připojena testovaná zadní svítilna. Laboratorní zdroj slouží pro napájení zadní svítilny, LIN karty a relé modulu. Vstupně-výstupní karta je propojená s relé modulem a společně tak tvoří HW pro spínání analogových funkcí zadní svítilny. LIN karta připojená k PC umožňuje komunikaci se zadní svítilnou po sběrnici LIN. Poslední komponentou HW je osciloskop, který dekóduje komunikaci po LIN sběrnici a posílá data do testovací aplikace, kde jsou tyto data porovnávaná s odeslanými pomocí LIN karty. Osciloskop je tedy zde pro kontrolu odeslaných nebo přijatých příkazů, jedná se o požadavek firmy.

6.1 CompactDAQ šasi cDAQ-9174

Pro komunikaci po LIN sběrnici byl vybrán CompactDAQ šasi model cDAQ-9174 v kombinaci s LIN kartou od firmy National Instrumets. CompactDAQ šasi je systém pro sběr dat, který obsahu více slotů pro připojení různých I/O modulů pro různé aplikace, a dále je vybavený rozhraním USB, Ethernet nebo WiFi umožňující připojení k počítači. Konktrétně vybraný model cDAQ-9174 obsahuje 4 sloty pro připojení I/O karet a je vybaven rozhraním USB. Hlavní výhodou CompactDAQ šasi je především možnost připojení více karet, jejichž data jsou pak přenášeny pomocí jednoho rozhraní. Toto šasi bylo vybráno především pro budoucí možnost rozšíření o další karty. [28],[30]

Obrázek 6.2: *CompactDAQ šasi cDAQ-9174 [30]*

6.2 LIN Karta NI-9866

LIN Karta NI-9866 Umožňuje systému CompactDAQ připojení ke komunikační sběrnici LIN. Jedná se modul řady C pro vývoj aplikací s ovladačem NI-XNET. Karta obsahuje 1 port RS232 pro rozhraní LIN s možností připojení až 16 zařízení. Maximální délka kabelu pro rozhraní LIN je 40 m. Pro správnou komunikaci je nutné připojit externí napájení LIN sběrnice v rozmezí 8 až 18 V DC. [31],[32]

Obrázek 6.3: *LIN karta NI-9866 [31]*

6.3 Laboratorní Zdroj Manson HCS3402

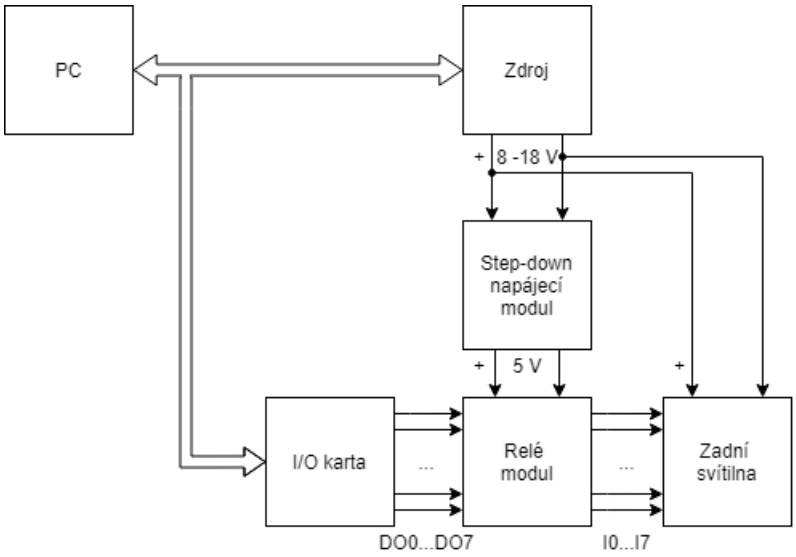
Pro napájení svítilny byl vybrán laboratorní zdroj Manson HCS3402, který umožňuje připojení k počítači pomocí sběrnice USB pro programování parametrů. Pro programování lze využít software od výrobce Manson, nebo přes programové prostředí LabVIEW, pomocí ovladačů vytvořených od výrobce. Výstupní napětí zdroje je 1-32 V a proud 0-20 A, což je dostatečné pro napájení svítilny. [35]

Obrázek 6.4: *Zdroj Manson HCS3402 [35]*

6.4 I/O karta USB-6002

Jedná se o multifunkční I/O zařízení, které poskytují různé kombinace analogových, digitálních I/O a funkcí časovače a čítače. USB 6002 poskytuje 8 analogových vstupů, 2 analogové výstupy a 13 digitálních I/O ve 3 portech které je možné nastavit na vstupní nebo výstupní. Analogové vstupy a výstupy mají rozsah $\pm 10\,V$ a obsahují 16-bit A/D (D/A) převodník s celkovou vzorkovací frekvencí maximálně 50 kS/s pro analogové vstupy a maximálně 5 kS/s na kanál pro analogové výstupy. Digitální vstupy a výstupy pracují na principu TTL logiky, tedy v rozsahu 0-5 V. Maximální výstupní proud pro digitální výstupy na kanál je 4 mA. [33],[34]

Obrázek 6.5: *I/O karta USB-6002 [33]*



Obrázek 6.6: *Blokové schéma pro I/O kartu a další HW pro spínání analogových funkcí*

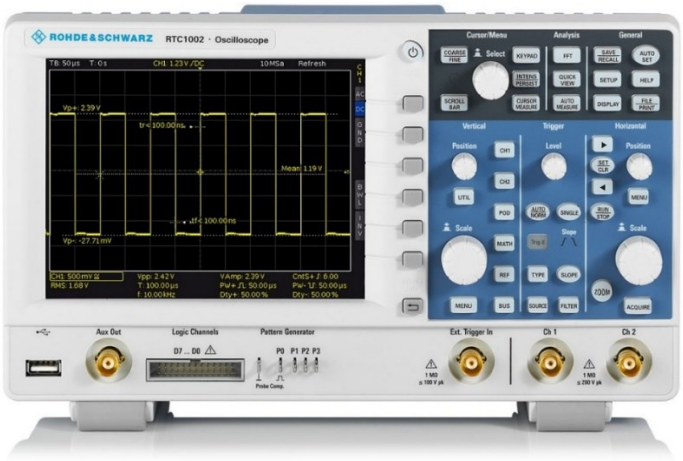
Jak je vidět na obrázku 6.6 tato karta byla vybrána především pro možnost spínání analogových funkcí svítilen přes aplikaci, při použití dalšího hardwaru, a pro další možné rozšíření. Dalším hardwarem pro zajištění spínání analogových funkcí je Arduino relé modul, který obsahuje 8 relé modulů ovládaných TTL logikou, a pro napájení relé modulů Step-down napájecí modul LM2596 Buck. LM2596 Buck umožňuje konstantní nastavení nižšího výstupního napětí v rozsahu 1,25-37 V při vstupním napětím 4-40 V, přičemž musí být vyšší než výstupní napětí. Výstupní proud tohoto modulu je až 3 A. Tento další hardware musel být vybrán, protože digitální výstupy I/O karty mají malý výstupní proud 4 mA (pro jeden digitální výstup) a nedokážou tak samostatně sepnout relé. [36],[37]

Obrázek 6.7: *Arduino relé modul [36]*

Obrázek 6.8: *Step-down napájecí modul LM2596 Buck [37]*

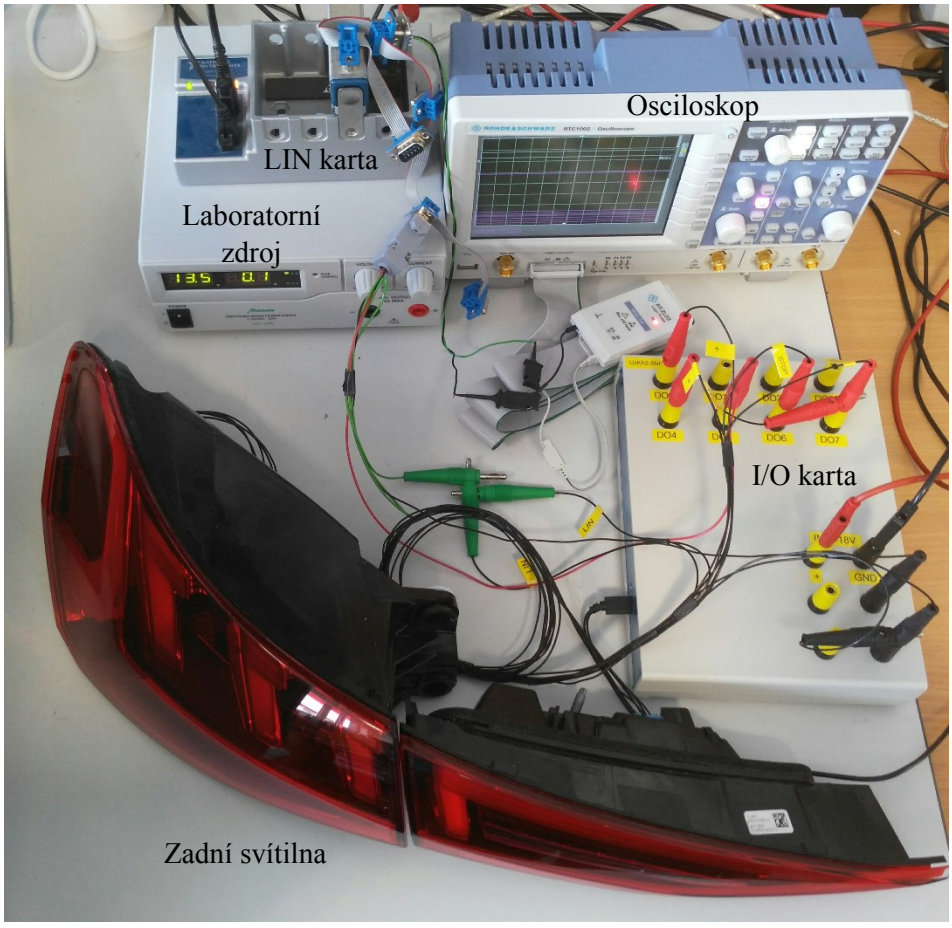
6.5 Osciloskop R&S RTC1000 RTC1K-52

Digitální osciloskop Rohde & Schwarz RTC1000 RTC1K-52 je vysoce citlivý osciloskop s možností vykonávat řadu funkcí. Hlavními vlastnostmi jsou šířka pásma 50 MHz, vzorkovací frekvence 2 GHS a hloubka paměti 2 MS. Osciloskop obsahuje 2 klasické kanály a 8 logických kanálů. Mimo standartní funkce umožňuje další, kterými jsou logická analýza s možností MSO, generátor různých typů signálů, dekódování sériové sběrnice typu I^2C, SPI, UART, LIN, CAN a funkcí digitálního voltmetru. Připojení k PC je umožněno pomocí komunikační sběrnice USB nebo Ethetnet.



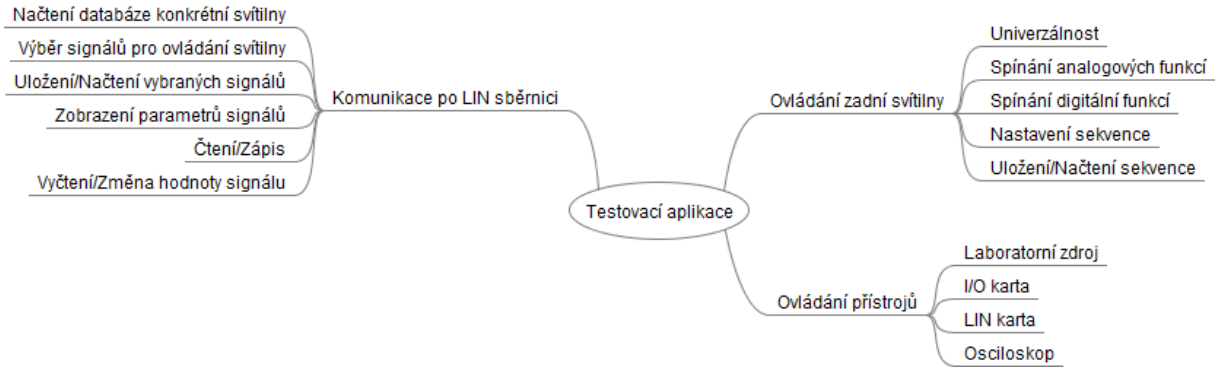
Obrázek 6.9: *Osciloskop Rohde & Schwarz RTC1000 RTC1K-52 [38]*

Tento osciloskop byl vybrán především pro funkci dekódování sběrnice LIN, s možností vyčtení výsledku do aplikace. Pro správnou funkci dekódování sběrnice LIN se k osciloskopu musely dokoupit HW a SW doplňující balíčky MSO RTC-B1 a CAN/LIN RTC-K3. MSO RTC-B1 je HW a SW balíček obsahující speciální a měřící sondu pro měření digitálních kanálů a SW licenci pro zpřístupnění funkcí pro měření digitální kanálů na osciloskopu. CAN/LIN RTC-K3 je SW balíček obsahující licenci pro zpřístupnění funkcí pro dekódování CAN a LIN sběrnice na osciloskopu. [38],[39]



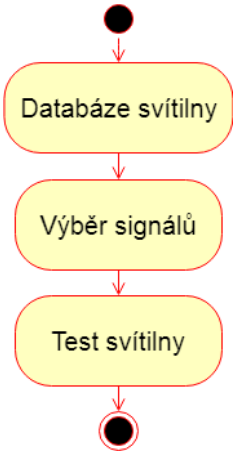
Obrázek 6.10: *Foto pracoviště*

7 Návrh aplikace



Obrázek 7.1: *Struktura funkcí testovací aplikace*

Na obrázku 7.1 je zobrazena struktura funkcí testovací aplikace, která znázorňuje, co testovací aplikace umožňuje. Je to tedy především ovládání zadní svítilny, které je dle požadavků univerzální (pro jakoukoliv svítilnu s LIN sběrnicí), je umožněno spínání analogových a digitálních funkcí a nastavení sekvence s možností načtení a uložení. Komunikace po LIN sběrnici, kdy v aplikaci má uživatel možnost načtení databáze konkrétní svítilny, výběr signálů z databáze svítilny pro ovládání svítilny, jejich následné uložení nebo načtení, zobrazení parametrů signálů, zvolení režimu čtení nebo zápis a změna nebo vyčtení hodnoty signálu. Ovládání přístrojů tedy laboratorního zdroje, vstupně-výstupní karty, LIN karty a Osciloskopu.

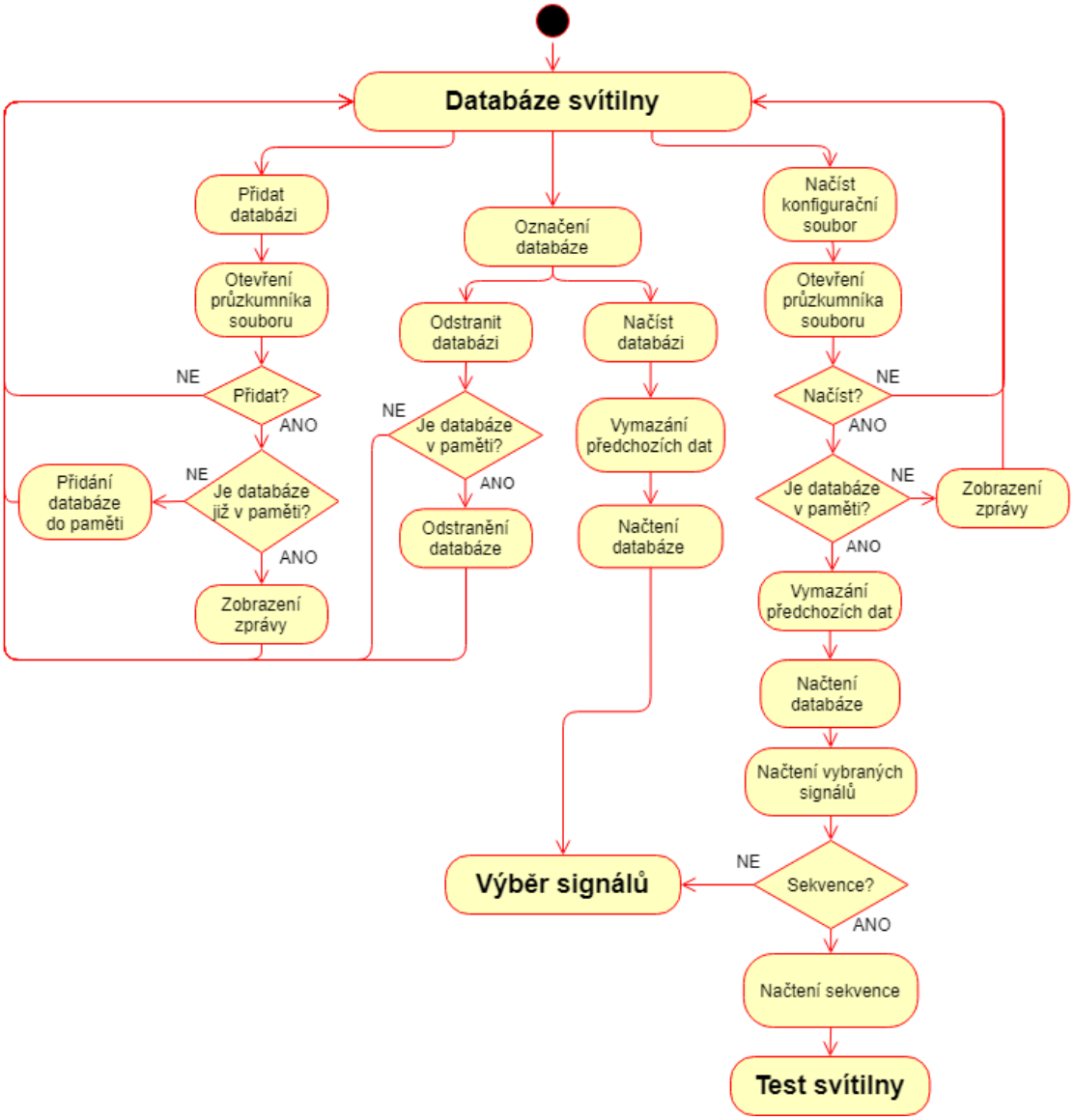


Obrázek 7.2: *Základní stavový diagram*

Základní stavový diagram na obrázku 7.2 znázorňuje jednotlivá okna aplikace, kterými bude aplikace postupně procházet. V aplikaci je tedy nutné nejprve v prvním okně databáze svítilny vybrat příslušnou databázi svítilny, podle které se bude aplikace ovládat svítilnu. Poté následuje druhé okno výběr signálu, ve kterém se provede výběr potřebných signálů z databáze, pro ovládání svítilny, přičemž aplikace je navržena tak aby byl výběr signálu rozdělen na signály pro čtení a signály pro zápis. Po vybrání signálů z databáze svítilny následuje poslední okno test svítilny, které už slouží k otestování a ovládání dané zadní svítilny a je umožněno komunikace po LIN sběrnici, změna hodnot vybraných signálů, nastavení sekvence a ovládání přístrojů laboratorní zdroj, osciloskop a I/O karta. Každé okno má zvlášť navržený podrobný stavový diagram, který ukazuje jednotlivé hlavní události, jejich posloupnost a také přechody mezi jednotlivými okny.

Vedlejšími událostmi, které navíc obsahuje každé okno je zobrazení nápovědy a přepínání mezi jazyky aplikace. Podporovanými jazyky uživatelského rozhraní jsou čeština a angličtina.

7.1 Stavový digram okno databáze svítilny

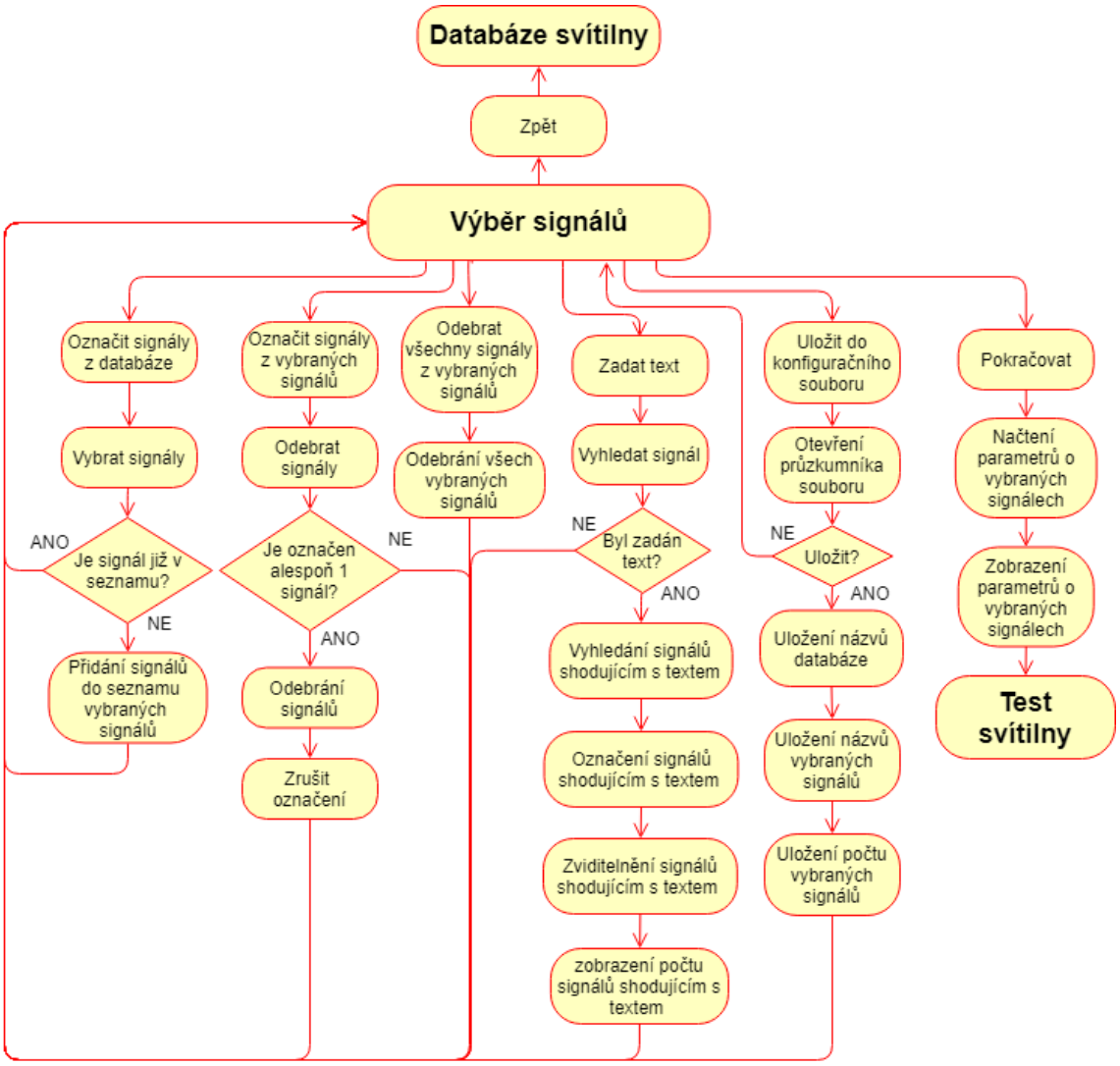


Obrázek 7.3: *Stavový diagram pro okno databáze svítilny*

Okno databáze svítilny, dle stavového diagramu pro okno databáze svítilny na obrázku 7.3 umožňuje celkem 4 události, kterými jsou přidat, načíst anebo odstranit databázi a načtení konfiguračního souboru. Události načíst databázi a načíst konfigurační soubor umožňují přechod aplikace do dalšího okna. Každá událost je spuštěna stisknutím určeného tlačítka. Po spuštění události přidat databázi je otevřen průzkumník souboru, kde je možné vybrat soubor databáze svítilny a po úspěšném vybrání následuje kontrola, zda se již tato databáze nachází v paměti, pokud ne je přidána do paměti a pokud ano je zobrazena zpráva „databáze se již v paměti nachází“. V události odstranit databázi se jen provede kontrola, zda se daná databáze nachází v paměti a pokud ano je následně odstraněna. Událost načíst databázi vymaže předchozí data databáze svítilny a provede načtení nové a přechod do dalšího okna výběr signálu.

Před vykonáním těchto dvou událostí je však nutné nejprve danou databázi svítilny označit v seznamu. Poslední událost načíst konfigurační soubor otevře průzkumníka souboru a umožní vybrání souboru a po úspěšném vybrání je provedena kontrola zda se databáze svítilny nachází v paměti. Pokud ano dojde k vymazání předchozích dat databáze svítilny, načtení nové, uložených vybraných signálů a sekvence, jestliže se v souboru nachází a přejití do dalšího okna test svítilny, jestliže se souboru sekvence nenachází, dojde k přejití do okna výběr signálů. V případě, že se databáze nenachází v paměti, dojde k ukončení události a zobrazí se zpráva „chybějící databáze, nejprve databázi přidejte do paměti“.

7.2 Stavový digram okno Výběr signálů

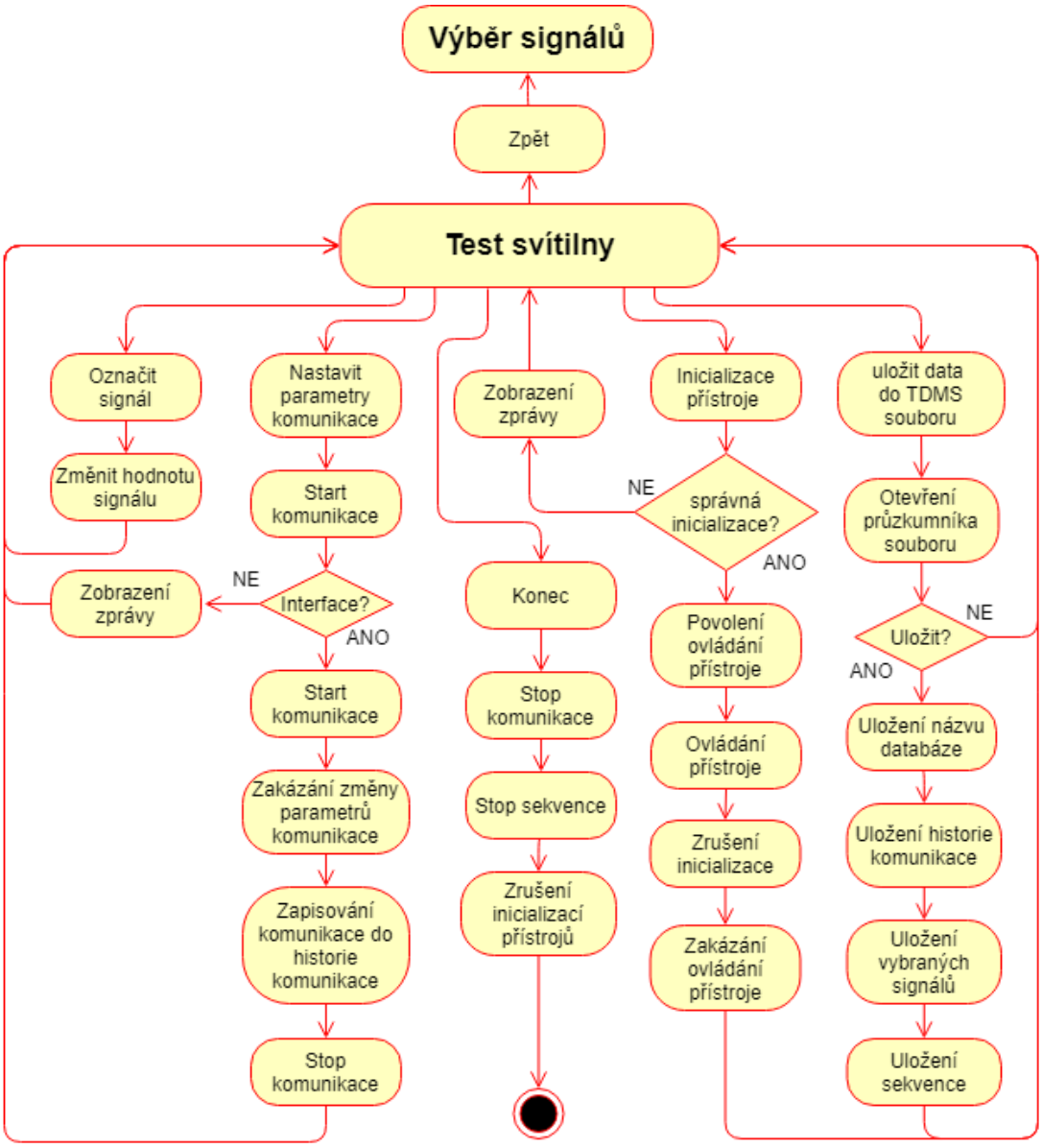


Obrázek 7.4: *Stavový diagram pro okno výběr signálů*

Okno výběr signálů, dle stavového diagramu pro okno výběr signálů na obrázku 7.4 umožňuje celkem 7 událostí, kterými jsou vybrat, odebrat, odebrat všechny, vyhledat signály, uložit do konfiguračního souboru, zpět a pokračovat. Událost pokračovat umožňuje přechod aplikace do dalšího okna test svítilny, po načtení a zobrazení parametrů signálů a událost zpět do předchozího okna databáze svítilny. Událost vybrat signály umožňuje přidání označených signálů z databáze svítilny do vybraných signálů, pro ovládání svítilny.

Událost odebrat signály provede odebrání všech signálů ze seznamu vybraných signálů a událost odebrat všechny signály resetuje celý seznam vybraných signálů. Pro vyhledání signálu v databázi svítilny slouží událost vyhledat signál, která na základě zadaného textu vyhledá signály obsahující text a označí, zviditelní je v seznamu a také vypíše počet nalezených shodujících se signálů. Při události uložit do konfiguračního souboru dojde k uložení názvu databáze svítilny a uložení počtu a názvů vybraných signálů. Událostí vybrat, odebrat, odebrat všechny, vyhledat signály se nachází v tomto okně dvakrát, jednou pro signály určené pro čtení a podruhé pro signál určené pro zápis.

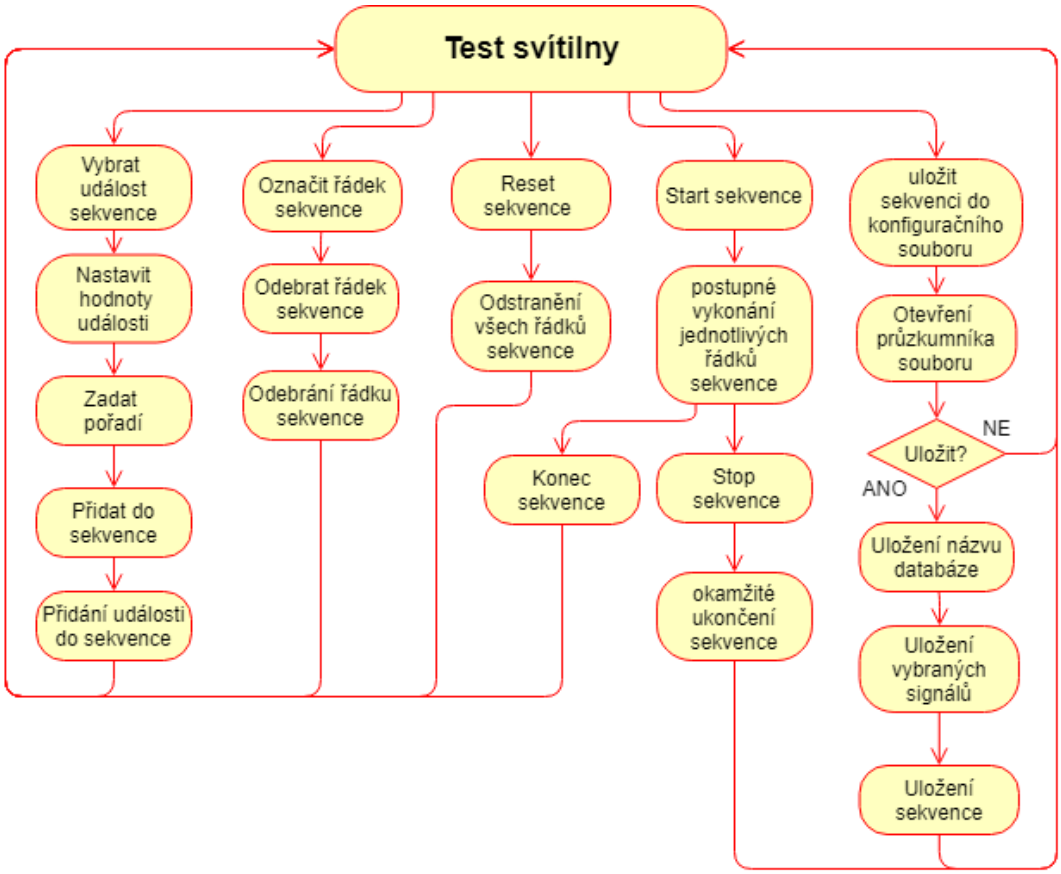
7.3 Stavový digram okno Test svítilny



Obrázek 7.5: *první část Stavového diagramu pro okno test svítilny*

Okno test svítilny, dle první části stavového diagramu pro okno test svítilny na obrázku 7.5 umožňuje celkem 8 událostí, kterými jsou: zpět, změnit hodnotu signálu, start a stop komunikace, konec, inicializace a zrušení inicializace přístroje a uložení dat do TDMS souboru.

Pomocí události zpět se aplikace přesune do předchozího okna výběr signálů. Událost změnit hodnotu signálu umožňuje po označení signálu změnit jeho hodnotu, dále je možné vytvořit komunikaci po sběrnici LIN pomocí události start komunikace, kdy je nutné předem nastavit vhodné parametry komunikace, které po vytvoření komunikace už nelze měnit a nastavit interface. Změna hodnoty je možná i za běhu komunikace. Komunikace je ukončena událostí stop komunikace. Jednotlivé přístroje, kterými jsou zdroj, I/O karta a osciloskop lze inicializovat pomocí inicializace přístroje a poté je ovládat. Při zrušení inicializace se ukončí softwarové ovládání přístroje a dojde k zakázání ovládání přístroje. Jednotlivé data testu lze uložit do TDMS soboru pomocí události uložit data do TDMS souboru, kdy dojde k uložení názvu databáze svítilny, historie komunikace, vybraných signálů a sekvence. Celá aplikace e ukončena pomocí události konec, kdy se nejprve automaticky ukončí sekvence a komunikace.



Obrázek 7.6: *druhá část Stavového diagramu pro test svítilny*

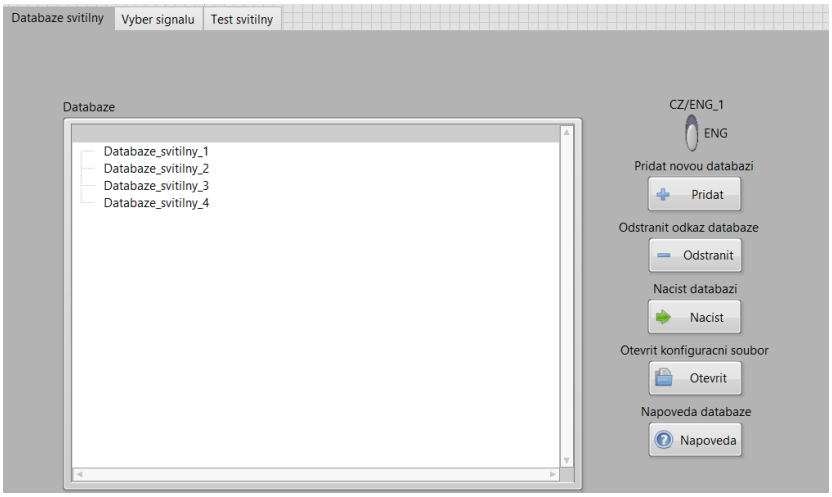
Okno test svítilny, dle druhé části stavového diagramu pro okno test svítilny na obrázku 7.6 umožňuje dále celkem 6 událostí, kterými jsou přidat událost do sekvence, označit řádek, reset, start a stop sekvence a uložit sekvenci do konfiguračního souboru. Událost přidat do sekvence umožňuje po nastavení parametrů vložit do sekvence další událost jako řádek do tabulky, a pomocí události odebrat řádek sekvence se označený řádek z tabulky odstraní. Reset sekvence provede odstranění všech řádků sekvence. Po připravení sekvence je možné sekvenci spustit, kdy sekvence vykonává jednotlivé řádky tabulky sekvence, tak že provádí zadané události za uživatele. Po dokončení posledního řádku sekvence je dojde k automatickému ukončení sekvence, pro případné manuální okamžité ukončení slouží událost stop sekvence. Celá sekvence lze také uložit do konfiguračního souboru pomocí události uložit sekvenci do konfiguračního souboru, kde se uloží i název databáze svítilny a vybrané signály.

8 Popis vytvořené aplikace

Aplikace je rozdělena dle základního stavového diagramu na obrázku 7.2 do třech hlavních oken, kterým jsou databáze svítilny, výběr signálu a test svítilny. V prvním okně databáze svítilny jsou uživateli zobrazeny, všechny odkazy na databáze svítilen uložené v paměti a uživateli je umožněno přidat další nebo naopak odebrat. Po zvolení databáze svítilny se aplikace přepne do druhého okna, kde jsou v jednotlivých stromech zobrazeny signály pro čtení a zápis. Uživateli je umožněno vybrat libovolné signály z každého stromu pro testování a ovládání dané svítilny dle specifikace. Po výběru signálu následuje poslední okno test svítilny, které slouží k ovládání a testování dané svítilny, změnu hodnot jednotlivých signálů, zahájení a ukončení komunikace, ovládání jednotlivých přístrojů a definování sekvence.

8.1 Databáze svítilny

Databáze svítilny je první okno, které se zobrazí po spuštění aplikace. Toto okno je naprogramované dle stavového diagramu databáze svítilny na obrázku 7.3.



Obrázek 8.1: *Okno Databáze svítilny*

Jak je vidět na obrázku 8.1 ve stromu databáze jsou načteny jednotlivé odkazy na všechny databáze v paměti. Přepínač CZ/ENG slouží k přepínání mezi jazyky aplikace. Pomocí tlačítka Přidat novou databázi je po splnění podmínek odkaz na databázi přidán do paměti a také do stromu. Podmínkou pro přidání databáze je že jméno databáze ani cesta k databázi se nesmí shodovat s žádnou databází v paměti. Při nesplnění této podmínky bude uživateli zobrazena na obrazovce vyskakovací okno se zprávou „Databáze se již v paměti nachází“. Tlačítko odstranit odkaz databáze slouží k odstranění odkazu databáze z paměti a stromu databáze. Pro pokračování do dalšího okna jsou k dispozici tlačítka načtení databáze a otevření konfiguračního souboru. Pro načtení databáze pomocí tlačítka načtení databáze je nejprve nutné, aby uživatel vybral správnou databázi ve stromu databáze, a potom pokračoval stiskem tlačítka načtení databáze. Po stisku tohoto tlačítka přejde aplikace do dalšího okna, kde se zobrazí signály databáze. Pro načtení databáze pomocí tlačítka otevřít konfigurační soubor není nutné vybírat databázi ve stromu databáze, stačí jen po jeho stisknutí vybrat správný konfigurační soubor, ve kterém je již uložen název dané databáze. Podmínkou je však, aby název databáze v konfiguračním souboru byl i v paměti.

Podle toho jestli se v konfiguračním souboru nachází i uložená sekvence, tak následně aplikace přejde do okna test svítilny, v opačném případě pouze do následujícího okna výběr signálů. Posledním tlačítkem je nápověda, která zobrazí uživateli ve vyskakovacím okně nápovědu k danému oknu.

8.1.1 Struktura konfiguračního souboru

Konfigurační soubor obsahuje celkem 4 sekce databáze, signály_rx, signály_tx a sekvence. V každé sekci jsou uložené jednotlivé klíče, které se musí v aplikaci vyčíst. Sekce signály_rx je určena signálům pro čtení a Sekce signály_tx signálům pro zápis.

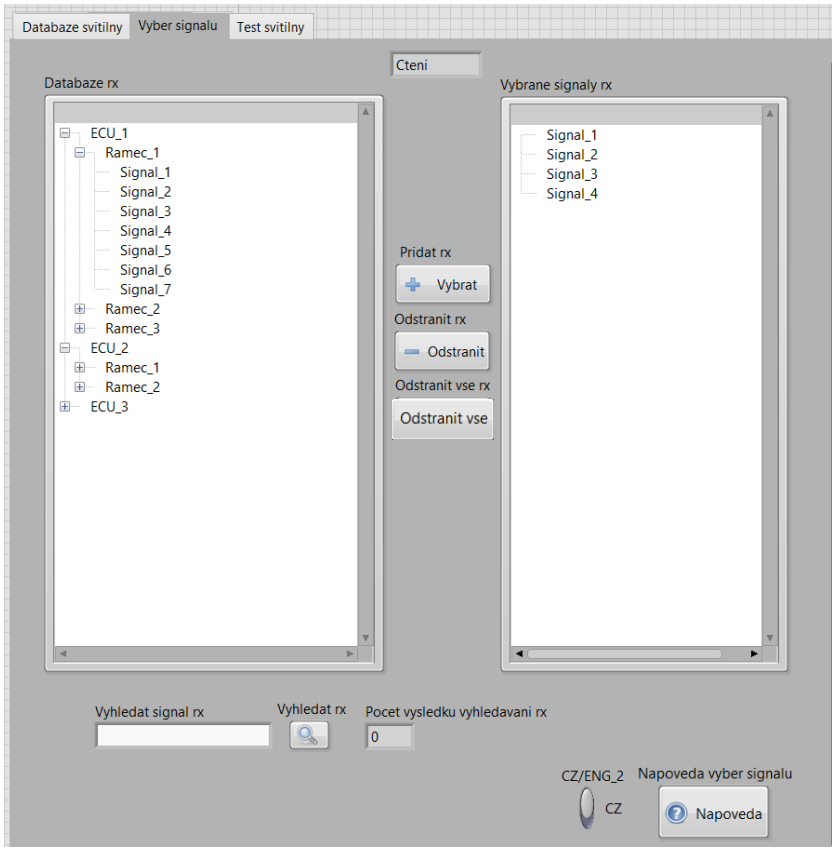
```
[Databaze]
Nazev_databaze = "databaze"

[Signaly_rx]
pocet_signalu_rx = 2
vybrane_signaly_rx = "signal1;signal2"

[Signaly_tx]
pocet_signalu_tx = 3
vybrane_signaly_tx = "signal1;signal2;signal3"

[Sekvence]
Pocet_radku_sekvence = 4
sekvence_radek_0 = "poradi-udalost-parametr1/signal1-hodnota1-"
sekvence_radek_1 = "poradi-udalost-parametr1/signal1;parametr2/signal2;-hodnota1;hodnota1;-"
sekvence_radek_2 = "poradi-udalost-parametr1/signal1-hodnota1-"
sekvence_radek_3 = "poradi-udalost-parametr1/signal1;parametr2/signal2;-hodnota1;hodnota1;-"
```

Obrázek 8.2: *Struktura konfiguračního souboru*

Jak je vidět na obrázku 8.2 sekce databáze obsahuje jeden klíč, ve kterém je uložen název databáze svítilny. Další sekce signály_rx a signály_tx obsahují dva klíče počet signálů a názvy vybraných signálů, kde jednotlivé signály jsou odděleny znakem „;“. Poslední sekce sekvence obsahuje klíč počet řádků sekvence, podle kterého je určen počet dalších klíčů, ve kterých je zapsán právě jeden řádek sekvence a všechny řádky tak vytvářejí tabulku sekvence. V řádku sekvence jsou jednotlivé sloupce odděleny znakem „-“ a jednotlivé položky patřícího do stejného sloupce „;“. Na obrázku jsou uvedeny pouze vzorové názvy uložených dat, skutečné názvy záleží na typu svítilny a jsou také firemním tajemstvím.

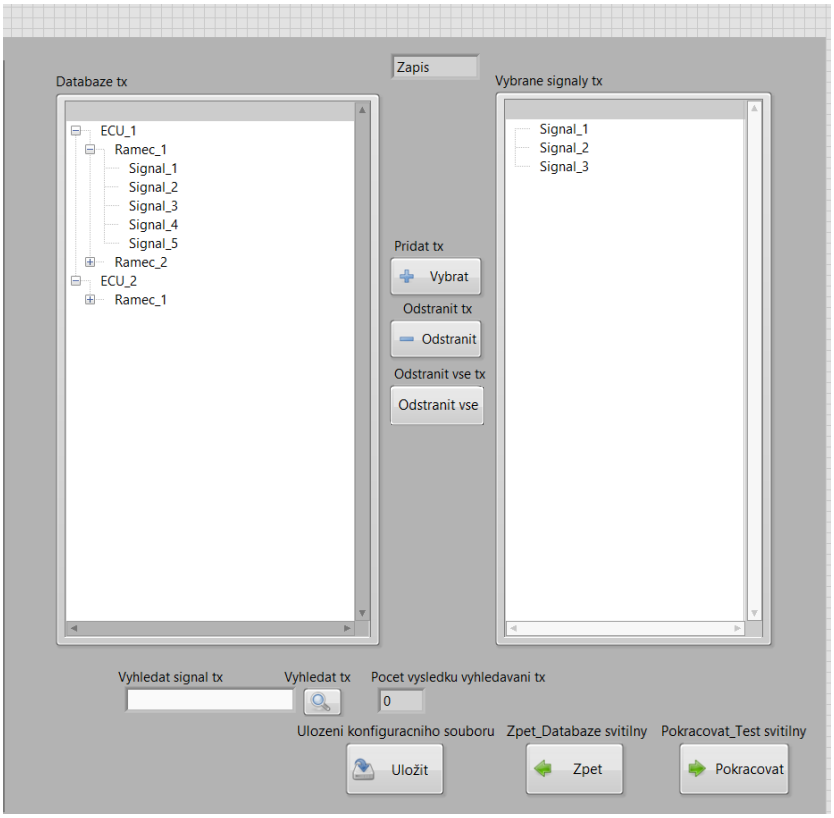
8.2 Výběr signálů

Výběr signálů je druhé okno, následující po prvním okně databáze svítilny. Toto okno je naprogramováno dle stavového digramu pro okno výběr signálů na obrázku 7.4 a slouží pro výběr signálů z databáze svítilny, které jsou potřebné pro testování a ovládání svítilny. Celé okno je graficky rozděleno na dvě části čtení a zápis dle signálů ve stromech.



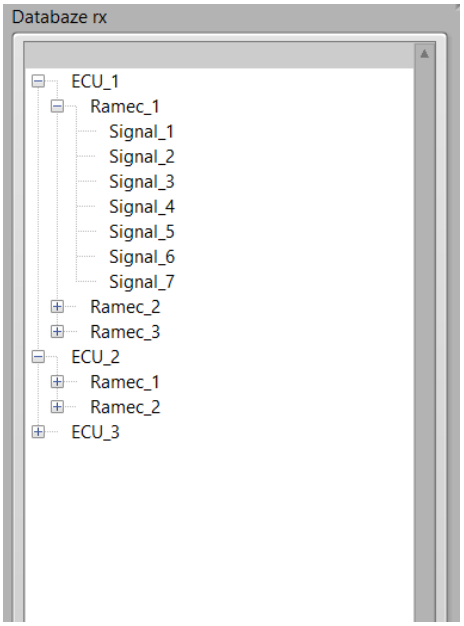
Obrázek 8.3: *Část okna Výběr signálů pro čtení*

Po přejití do tohoto okna pomocí tlačítka načtení databáze v předchozím okně se do jednotlivých stromů databaze_rx a databaze_tx načte struktura z vybrané databáze svítilny s určenými signály pro čtení do databaze_rx a pro zápis do databaze_tx tak jak lze vidět na obrázku 8.3 a 8.4. Při přejití do tohoto okna pomocí tlačítka otevření konfiguračního souboru v předchozím okně je do stromů databaze_rx a databaze_tx načtena databáze svítilny z konfiguračního souboru, a dále se také načtou jednotlivé vybrané signály do stromů vybrané signály rx a vybrané signály tx. Pomocí tlačítek přidat jsou vybrané signály ze stromu databaze_rx nebo databaze_tx zkopírovány do určeného stromu vybrané signály rx nebo vybrané signály tx. Tlačítka odstranit slouží k odstranění označených signálů ze stromu vybrané signály rx nebo vybrané signály tx, a pro odstranění všech signálů lze použít tlačítka odstranit vše. Jednotlivé signály lze vyhledat po zadání jejich části nebo celého textu do textového řádku vyhledat signály a stisknutí tlačítka vyhledat. Signály se shodujícím textem budou označeny v příslušném stromu žlutou barvou. Vyhledávání signálů nerozlišuje malé a velké písmena. Reset vyhledávání lze provést vyhledáním prázdného textu. Vybrané signály lze uložit do konfiguračního souboru, pomocí tlačítka uložení konfiguračního souboru. Tlačítkem CZ/ENG_2 lze přepínat jazyky a tlačítko nápověda slouží k zobrazení nápovědy, která zobrazí uživateli ve vyskakovacím okně nápovědu k danému oknu. V Pravém dolním rohu se pak nachází tlačítka Zpět, pro přepnutí do předchozího okna databáze svítilny a Pokračovat, pro přepnutí do následujícího okna test svítilny a vybrané signály z databáze svítilny se zobrazí v dalším okně i se všemi parametry.



Obrázek 8.4: *Část okna Výběr signálů pro zápis*

8.2.1 Struktura načtené databáze svítilny



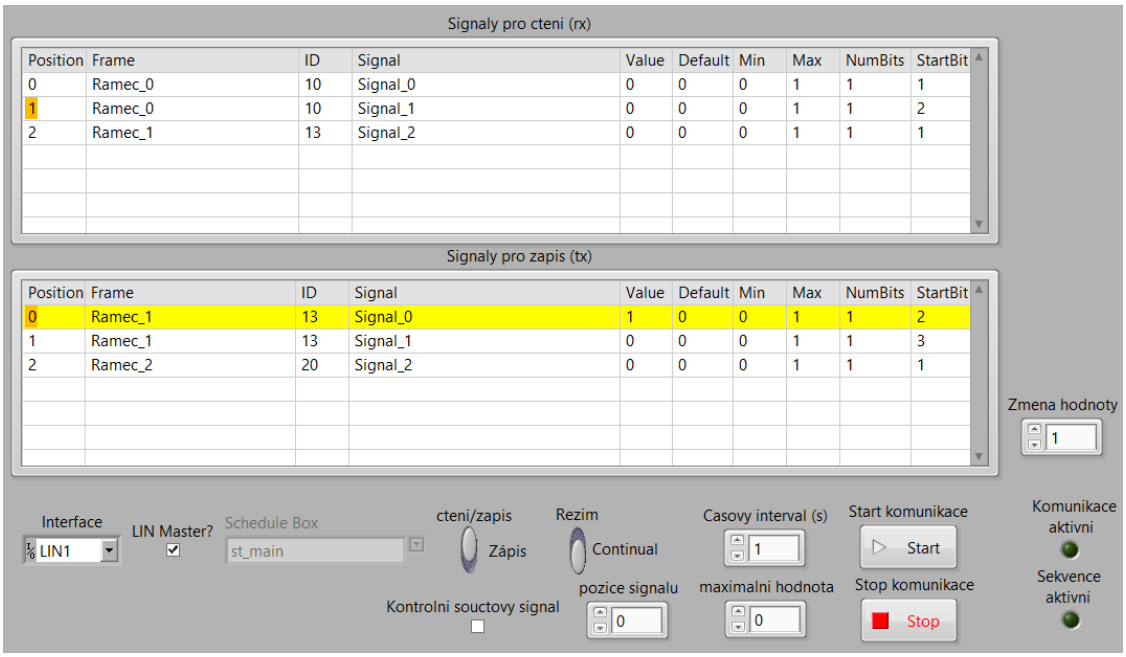
Obrázek 8.5: *Struktura načtené databáze svítilny*

Jak je vidět na obrázku 8.5 první položkou načtené databáze svítilny je jednotka ECU, po té následují podsložky s názvem rámce a nakonec podsložka s názvy signálů.

8.3 Test svítilny

Test svítilny je poslední a také hlavní okno celé aplikace, které je naprogramováno dle 2 stavových digramů pro okno test svítilny viz obrázek 7.5 a 7.6. Toto okno slouží především pro ovládání komunikace se zadní svítilnou, změnu hodnot jednotlivých signálů, zobrazení historie komunikace, ovládání jednotlivých přístrojů a pro nastavení (definování) sekvence. V pravém dolním rohu, se pak nachází tlačítko zpět do výběru signálů, které po stisknutí přepne aktuální okno aplikace na předchozí a tlačítko konec, pro ukončení celé aplikace.

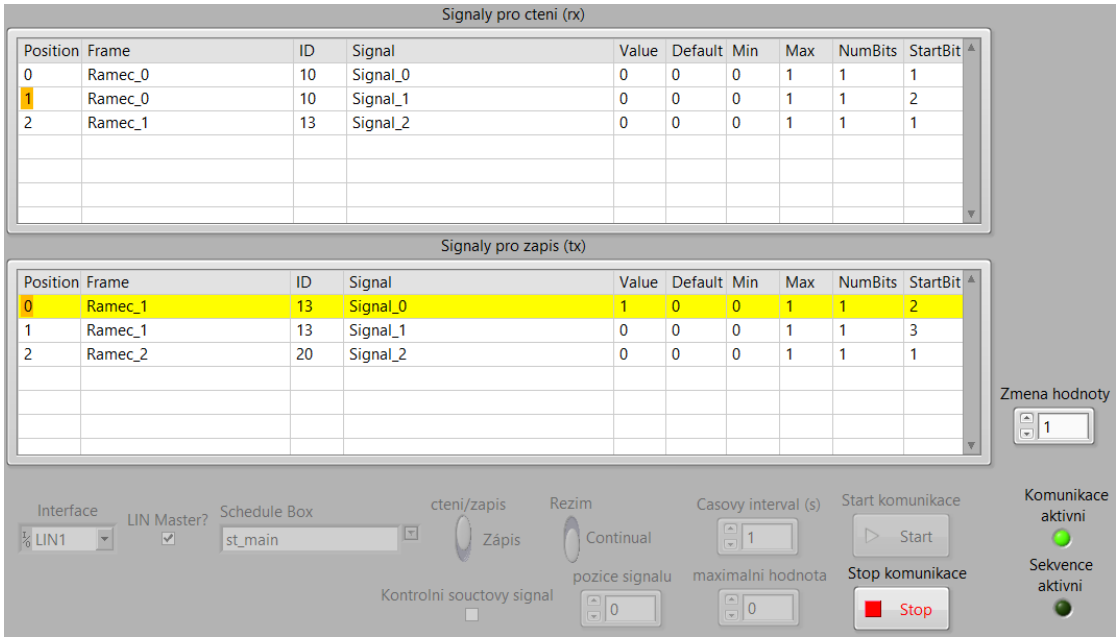
8.3.1 Ovládání komunikace



Obrázek 8.6: *Ovládání komunikace výchzí režim*

Jak lze vidět na obrázku 8.6 ovládání komunikace se nachází v levé horní části okna test svítilny a umožňuje komunikaci se zadní svítilnou. K ovládání komunikace jsou k dispozici dva list boxy signály pro čtení a signály pro zápis, číslicové vstupy změna hodnoty, časový interval, pozice signálu a maximální hodnota, I/O box interface, vypínače LIN master a kontrolní součtový signál, textový schedule box, přepínače čtení/zápis a Continual/Single, tlačítka start komunikace a stop komunikace a kontrolka komunikace aktivní signalizující běh komunikace. List boxy signály pro čtení a signály pro zápis zobrazují názvy a parametry vybraných signálů. Číslicový vstup změna hodnoty slouží pro změnu hodnoty označeného signálu v list boxu signály pro zápis. I/O box interface umožňuje výběr správného názvu sběrnice LIN karty. Vypínačem LIN master se nastavuje, zda je mód LIN karty master a zároveň se tímto vypínačem povolí i nastavení schedule v schedule boxu. Přepínač čtení/zápis slouží pro nastavení módu komunikace, tedy zda se bude jednat o čtení nebo zápis signálů. Přepínačem continual/single se nastavuje kontinuální nebo single mód komunikace. Číslicový vstup časový interval nastavuje časový interval pro kontinuální komunikaci. Vypínač kontrolní součtový signál slouží pro nastavení, zda je při komunikaci nutný kontrolní součtový signál a číslicovým vstupem pozice signálu se nastavuje jeho pozice a vstupem maximální hodnota se nastavuje jeho maximální hodnota.

Tlačítko start komunikace zahájí komunikaci, pokud byl zadán interface a zakáže změnu všech ovládacích prvků kromě číslicového vstupu časový interval, změna hodnoty, jednotlivých list boxů a tlačítka stop, které umožňuje zastavení komunikace a povolení všech ovládacích prvků.

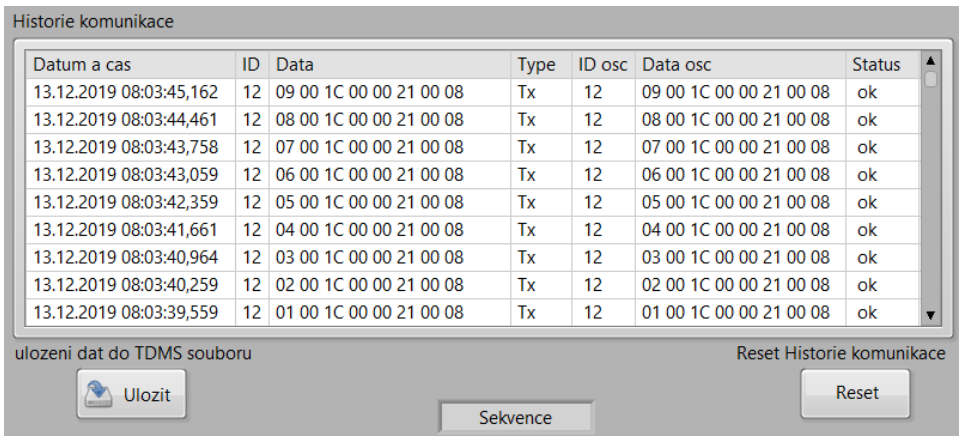


Obrázek 8.7: *Ovládání komunikace aktivní režim*

Jak je vidět na obrázku 8.6 a 8.7 první sloupec každého list boxu je pozice signálu v list boxu, Druhý sloupec obsahuje název rámce signálů. Třetím sloupcem je číslo identifikátoru rámce, čtvrtým název signálů. Pátý sloupec je vyhrazen pro aktuální hodnotu signálů a další sloupce zobrazují další informace o signálech, kterými jsou výchozí, minimální a maximální hodnoty signálů, počet bitů a počáteční bit.

8.3.2 Historie komunikace

Historie komunikace se nachází v pravé horní části okna test svítilny. Historie komunikace je list box zobrazující historii komunikace a naměřená data osciloskopu v čase. Pod list boxem se nachází tlačítko uložení dat do TDMS souboru, které uloží historii komunikace, jednotlivé názvy vybraných signálů s informacemi, a název databáze svítilny do TDMS souboru a tlačítko reset historie komunikace, které slouží k vymazání všech dat v list boxu historie komunikace.

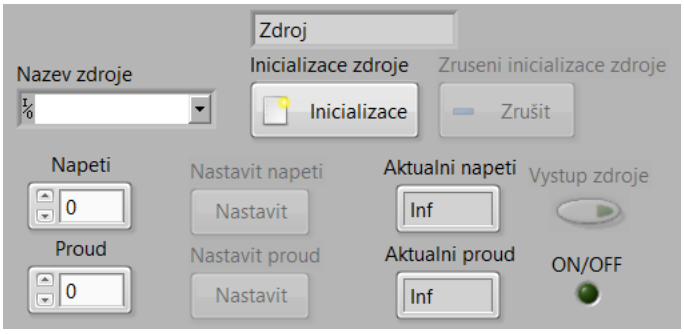


Obrázek 8.8: *Historie komunikace*

Jak je vidět na obrázku 8.8 první sloupec historie komunikace zobrazuje datum a čas, druhý sloupec ID, tedy číslo identifikátoru odeslaného rámce zprávy. Třetí sloupec zobrazuje přímo data LIN zprávy, které byly při komunikaci odeslány. Poté následují dva sloupce zobrazující identifikátor odeslaného rámce zprávy a data změřené pomocí osciloskopu, pokud bylo zapnuté měření osciloskopu. Poslední sloupec status zobrazuje, zda se odeslaná data shodují s naměřenými. Data zobrazená v obrázku jsou pouze vzorové a znázorňují pouze tak příklad zápisu dat do historie komunikace, skutečná data jakékoliv historie komunikace svítilny jsou firemním tajemstvím.

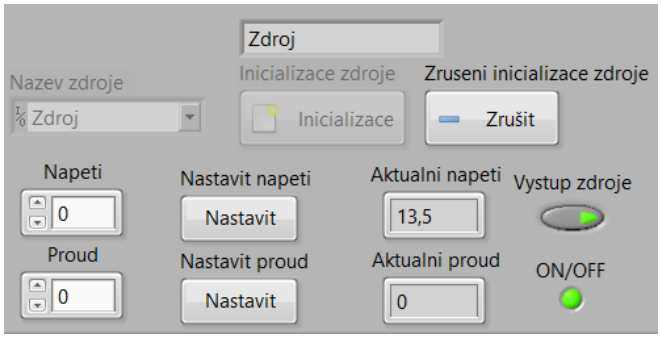
8.3.3 Ovládání Zdroje

Ovládání zdroje se nachází ve spodní levé části okna test svítilny v záložce zdroj/ I/O karta.



Obrázek 8.9: *Ovládání zdroje výchozí režim*

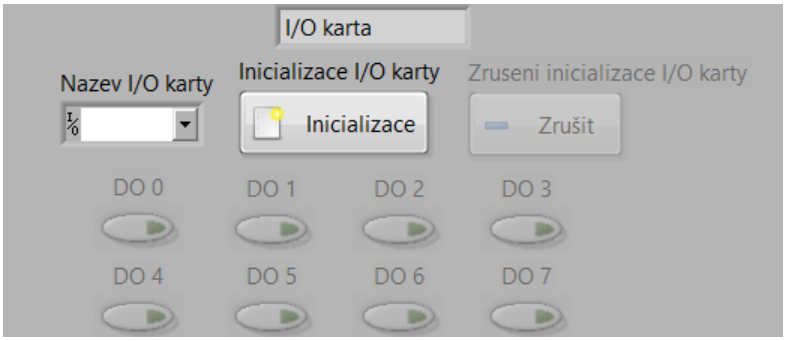
Jak je vidět na obrázku 8.9 ovládání zdroje jsou k dispozici textový box název zdroje, tlačítko inicializace zdroje, zrušit inicializaci zdroje, dva číslicové vstupy, dvě tlačítka pro nastavení zadané hodnoty, dva zobrazovací číselné výstupy, přepínač ON/OFF a kontrolka ON/OFF. Ve výchozím stavu jsou povoleny pouze textový box název zdroje, tlačítko inicializace zdroje, ostatní ovládací prvky jsou v zakázaném módu. Zdroj je tedy nutné nejprve inicializovat, až poté je povoleno ovládání. Textový box název zdroje umožňuje výběr správného vstupního názvu zdroje, který je k počítači připojen. Stiskem tlačítka inicializace zdroje je zdroj inicializován a povolí se ostatní ovládací prvky. Tlačítkem zrušit inicializaci zdroje dojde ke zrušení inicializace a ovládací prvky jsou opět v zakázaném módu. Číslicový vstup nastavit napětí, slouží k zadání hodnoty napětí, která má být nastavena. Stisknutím tlačítka nastavit napětí se nastaví napětí zdroje na požadovanou hodnotu. Nastavení požadovaného proudu funguje obdobně, akorát se musí využít číslicový vstup nastavit proud a tlačítko nastavit proud. Vedle tlačítek nastavit proud a nastavit napětí se nachází zobrazující číselné vstupy aktuální napětí a aktuální proud zobrazující aktuální hodnoty napětí a proudu, fungující pouze pokud je zdroj ve stavu ON. Přepínač ON/OFF slouží k nastavení stavu zdroje a kontrolka ON/OFF zobrazuje aktuální stav zdroje.



Obrázek 8.10: *Ovládání zdroje aktivní režim*

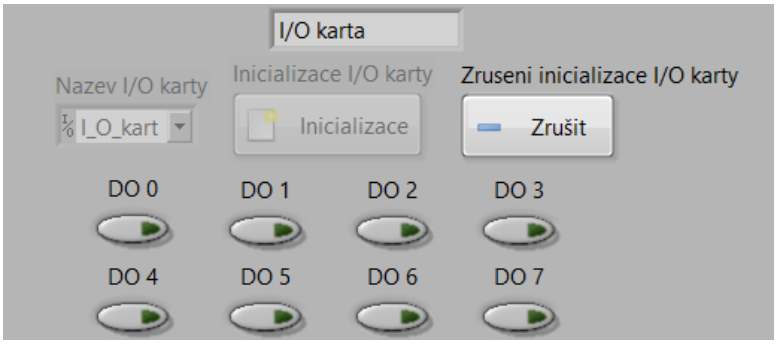
8.3.4 Ovládání I/O karty

Ovládání I/O karty se nachází ve spodní levé části okna test svítilny v záložce zdroj/ I/O karta.



Obrázek 8.11: *Ovládání I/O karty výchozí režim*

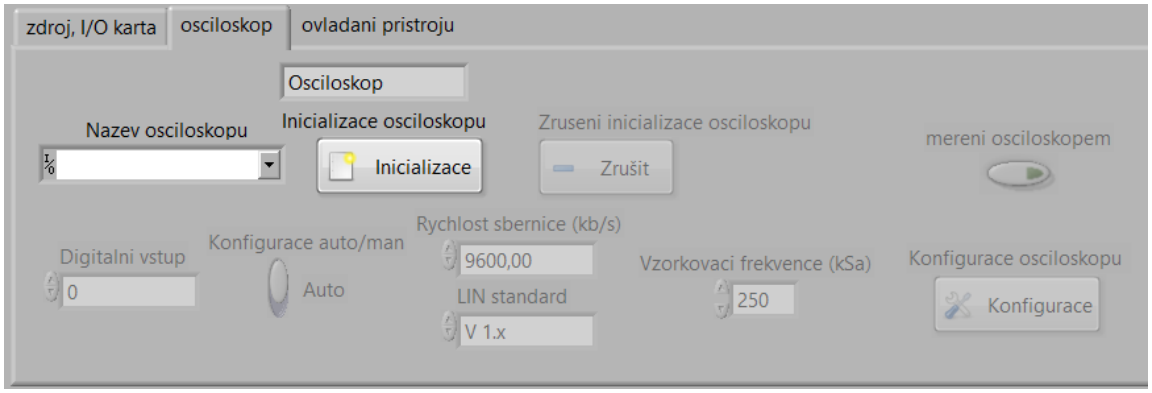
Jak je vidět na obrázku 8.11 k ovládání I/O karty jsou k dispozici textový box název I/O karty, tlačítko inicializace I/O karty, zrušit inicializaci I/O karty, 8 Vypínačů ON/OFF s názvy DO0 až DO7. Ve výchozím stavu jsou povoleny pouze textový box název I/O karty, tlačítko inicializace I/O karty, ostatní ovládací prvky jsou v zakázaném módu. Pro povolení ovládacích prvků je tedy nutná inicializace I/O karty. Textový box název I/O karty, umožňuje výběr všech připojených přístrojů, pro správnou funkci musí být vybrán správný název I/O karty a stisknuto tlačítko inicializace I/O karty, které provede inicializaci I/O karty, a povolí další ovládací prvky a zároveň zakáže textový box název I/O karty a tlačítko inicializace I/O karty. Tlačítko zrušit inicializaci I/O karty slouží k ukončení komunikace s přístrojem, přičemž se ovládací prvky nastaví zpět do výchozího stavu. Vypínače DO0 až DO7, slouží pro spínání jednotlivých analogových funkcí zadní svítilny.



Obrázek 8.12: *Ovládání I/O karty aktivní režim*

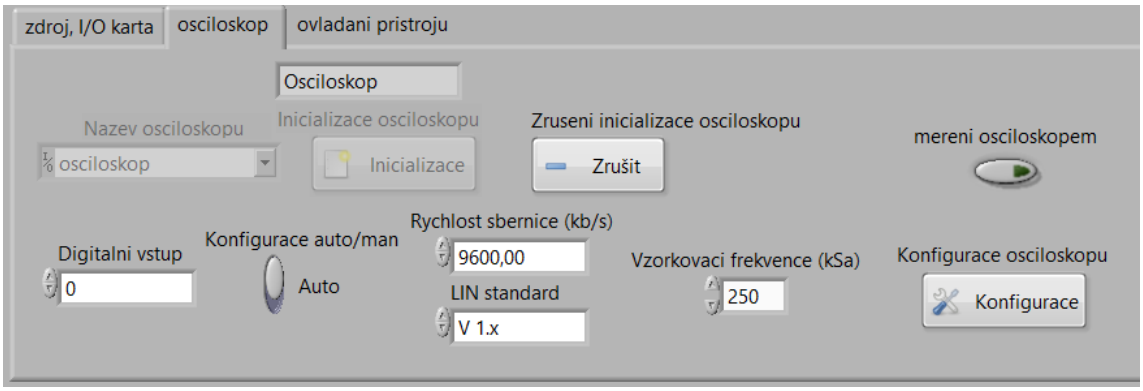
8.3.5 Ovládání osciloskopu

Ovládání osciloskopu se nachází ve spodní levé části okna test svítilny v záložce osciloskop.



Obrázek 8.13: *Ovládání osciloskopu výchozí režim*

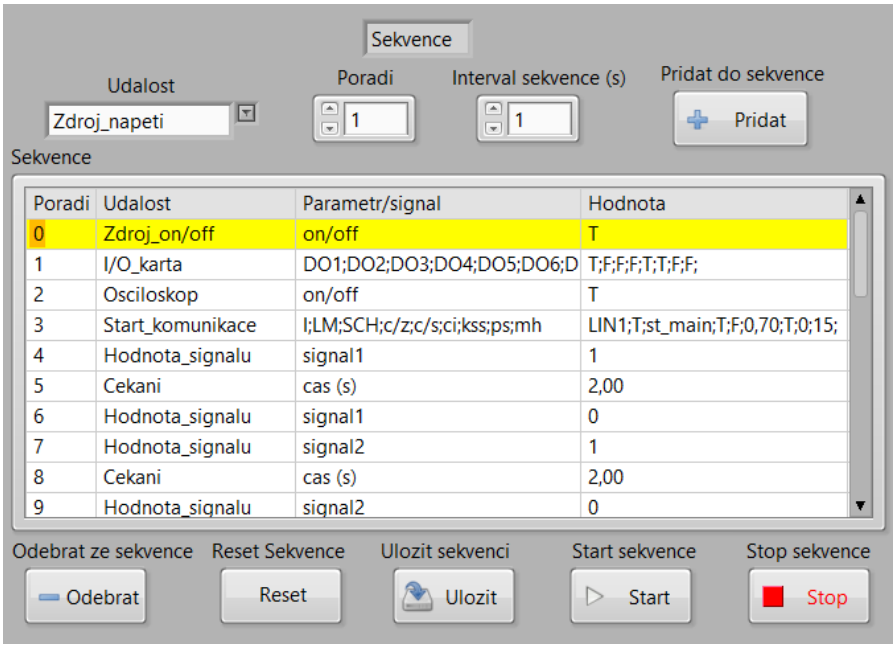
Jak je vidět na obrázku 8.13 k ovládání osciloskopu jsou k dispozici textový box název osciloskopu, tlačítka inicializace osciloskopu, zrušit inicializaci osciloskopu, konfigurace osciloskopu, přepínač konfigurace auto/man, vypínač měření osciloskopem a číslicové vstupy digitální vstup, rychlost sběrnice (kb/s), LIN standart a vzorkovací frekvence. Ve výchozím stavu jsou povoleny pouze textový box název osciloskopu, tlačítko inicializace osciloskopu, ostatní ovládací prvky jsou v zakázaném módu. Textový box název osciloskopu, umožňuje výběr všech připojených přístrojů, pro správnou funkci musí být vybrán správný název osciloskopu a stisknuto tlačítko inicializace osciloskopu, které provede inicializaci osciloskopu, a povolí další ovládací prvky a zároveň zakáže textový box název osciloskopu a tlačítko inicializace osciloskopu. Po inicializaci je vhodné osciloskop nakonfigurovat, tak aby umožnil dekódování probíhající komunikace po LIN sběrnici. Konfigurace se provede, tak že se zadá do číslicového vstupu digitální vstup digitální vstup osciloskopu, který je k měřené sběrnici připojen, dále se přepínačem konfigurace auto/man zvolí, zda mají být parametry rychlost sběrnice a LIN standard zadány automaticky z načtené databáze svítilny nebo manuálně. Poté je nutné zadat vhodnou vzorkovací frekvenci do číslicového vstupu vzorkovací frekvence, se kterou bude osciloskop následně měřit. Posledním krokem konfigurace je stisknutí tlačítka konfigurace osciloskopu, které provede konfiguraci osciloskopu dle zadaných konfiguračních parametrů. Vypínač měření osciloskopem slouží pro povolení měření osciloskopem, při běhu komunikace, kdy naměřená data se zapisují do list boxu historie komunikace.



Obrázek 8.14: *Ovládání osciloskopu po inicializaci*

8.3.6 **Ovládání sekvence**

Ovládání sekvence se nachází v pravém dolní části okna test svítilny.

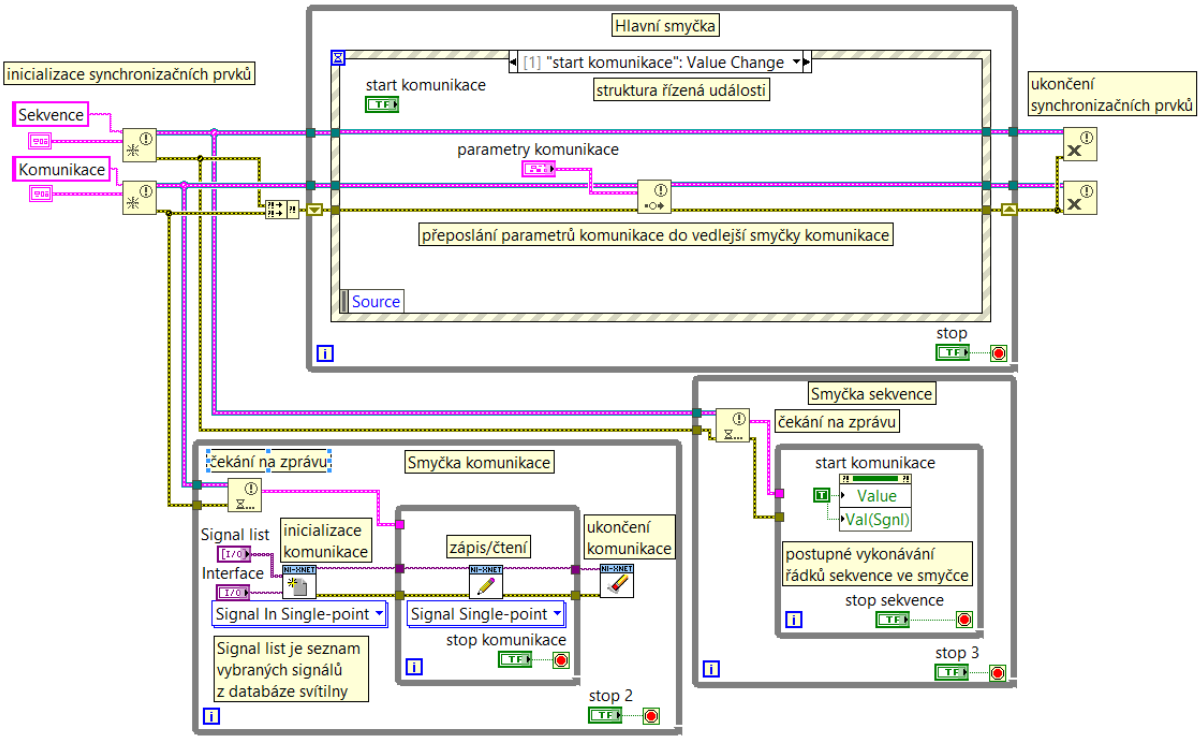


Obrázek 8.15: *Ovládání sekvence výchozí režim*

Jak lze vidět na obrázku 8.15 k ovládání sekvence jsou k dispozici textový box událost, číslicové vstupy pořadí a interval sekvence a tlačítka přidat do sekvence, odebrat ze sekvence, reset sekvence, uložit sekvenci, start sekvence a stop sekvence. Ve výchozím režimu jsou povoleny všechny ovládací prvky sekvence. Textový box událost umožnuje celkem výběr z 9 událostí, kterými jsou start komunikace, nastavení hodnoty signálu, stop komunikace, ovládání I/O karty, stav osciloskopu (on/off), nastavení parametrů zdroje, tedy napětí, proudu a stavu (on/off) a poslední událostí je čekání. Princip přidání události do sekvence je takový, že uživatel nejprve zvolí událost v textovém boxu událost, poté nastaví příslušné hodnoty parametrů určené události přímo v jednotlivých ovládacích částech přístrojů nebo komunikace, nastaví pořadí a stiskem tlačítka přidat do sekvence se do sekvence přidá vybraná událost s příslušnými nastavenými parametry. V případě události čekání je nutné nastavit pouze interval sekvence. Událost sekvence je vždy vkládána jako nový řádek do list boxu sekvence na určené pořadí, tak aby se jednotlivá pořadí neshodovala, kdy po vložení řádku jsou pořadí řádků aktualizovány dle pozice v list boxu. Odebrání označeného řádku sekvence je možné provést pomocí tlačítka odebrat ze sekvence a pro odstranění všech řádků tlačítko reset sekvence. Tlačítkem uložit sekvenci lze provést uložení sekvence, vybraných signálů a názvu databáze svítilny do konfiguračního souboru. Tlačítko start sekvence slouží k jednorázovému provedení celé sekvence, kdy nejprve jsou zakázány všechny ovládací prvky kromě tlačítka stop sekvence (aktivní režim) a poté se postupně provádí sekvence řádek po řádku, s pevným časovým rozdílem 400 ms. Vykonávaný řádek je navíc označen žlutou barvou, tak aby byl uživatel informován o průběhu sekvence. Po provedení všech řádků je sekvence automaticky ukončena, pro okamžité ukončení je možné použít tlačítko stop sekvence. Ukončením sekvence se všechny ovládací prvky povolí a jsou tak nastaveny do výchozího režimu.

9 Popis kódu aplikace

Kód aplikace obsahuje 3 smyčky typu WHILE, kdy se jedná především o architekturu Master/Slave. Hlavní smyčka v režimu Master obsahuje strukturu řízenou událostmi. První vedlejší smyčka, v režimu Slave slouží pro komunikaci po LIN sběrnici a druhá pro sekvenci. Obě vedlejší smyčky jsou spuštěny pouze hlavní smyčkou v reakci na událost start komunikace nebo start sekvence, pomocí dvou synchronizačních prvků mezi smyčkami notifikace (oznámení).



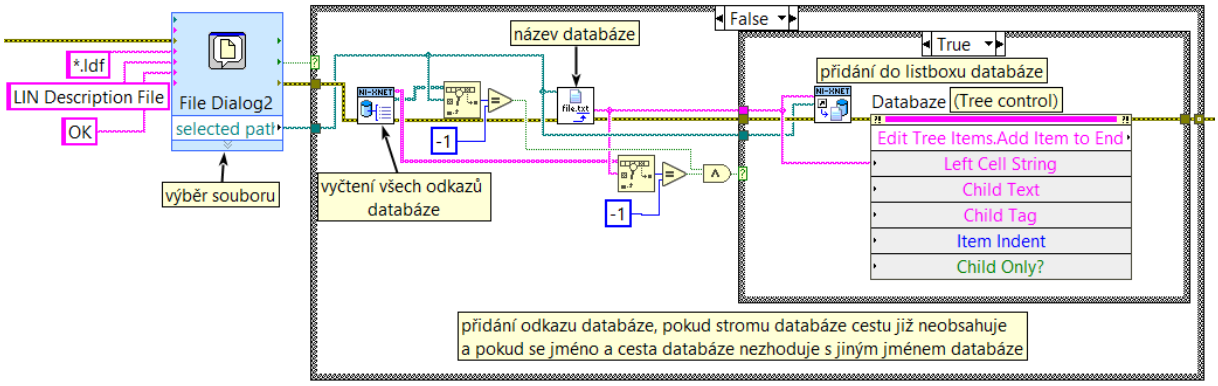
Obrázek 9.1: *Použitá architektura aplikace*

9.1 Hlavní smyčka

Jak lze vidět na obrázku 9.1 uvnitř hlavní smyčky se nachází struktura řízená událostmi, která reaguje na jednotlivé události vykonané uživatelem nebo programem, jako je například stisknutí některého tlačítka. Ve struktuře se nachází několik událostí, které jsou specifikovány a naprogramovány podle jednotlivých stavových diagramů na obrázcích 7.3, 7.4, 7.5 a 7.6. Při první iteraci hlavní smyčky se nastaví výchozí hodnoty všech proměnných, poté už jsou ve struktuře událostí vykonávány jednotlivé požadované události vykonané uživatelem nebo programem po celou dobu běhu aplikace.

Přidání databáze svítilny

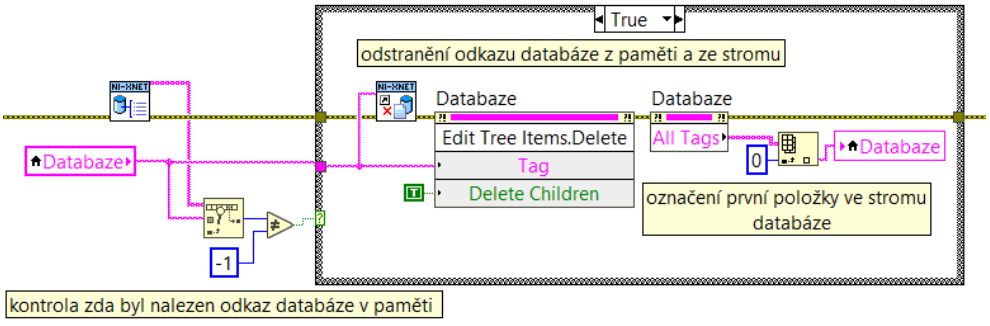
V kódu události pro přidání databáze svítilny je podle stavového diagramu na obrázku 7.3 nejprve otevřen průzkumník souboru pro výběr databáze svítilny. Po úspěšném výběru databáze svítilny následuje vyčtení všech odkazů databáze z paměti a provede se kontrola, zda se název a cesta databáze svítilny neshoduje z některou databází již v paměti, a pokud se neshoduje, je databáze přidána do paměti a stromu databáze (tzv. Tree Control) se stejným názvem jako v souboru. V opačném případě se uživateli zobrazí zpráva „databáze se již nachází v seznamu“. Úspěšným výběrem databáze svítilny se rozumí potvrzení souboru v průzkumníku souboru.



Obrázek 9.2: *Kód pro přidání databáze svítilny*

Odstranění databáze svítilny

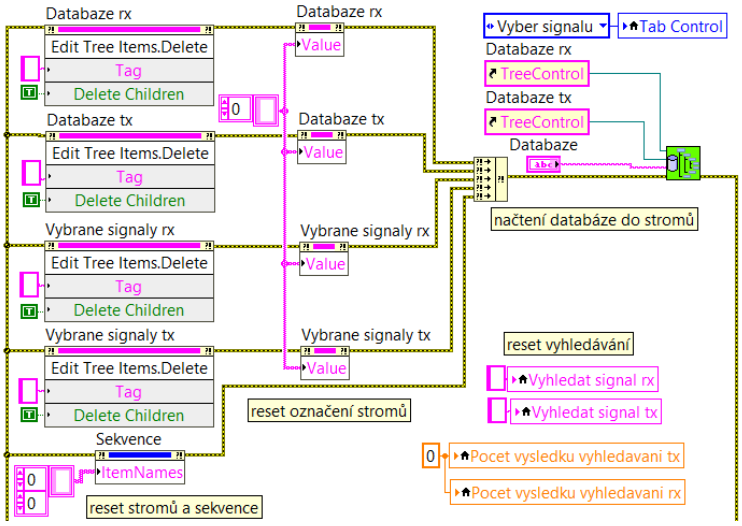
Pro odstranění databáze svítilny podle stavového diagramu na obrázku 7.3 se v kódu této události provede kontrola, zda se nachází v odkaz databáze paměti, při úspěšném provedení této kontroly se poté databáze vymaže z paměti i ze stromu databáze a označí se první položka ve stromu databáze. V případě neúspěšné kontroly se odstranění neprovede.



Obrázek 9.3: *Kód pro odstranění databáze svítilny*

Načtení databáze svítilny

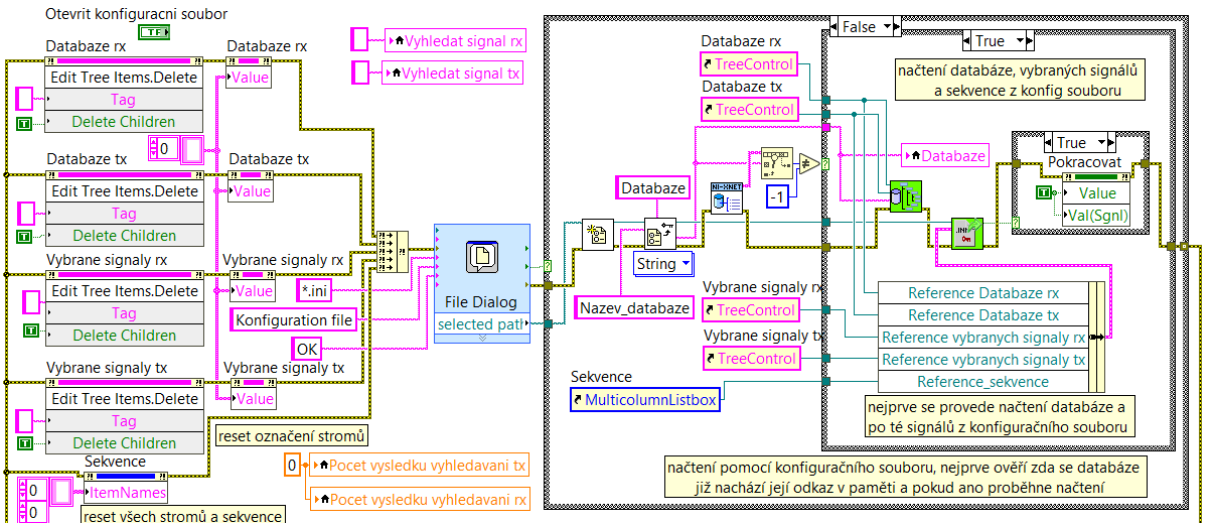
V kódu této události se podle stavového diagramu na obrázku 7.3 nejprve provede reset všech stromů a jejich označení a poté následuje načtení databáze svítilny do stromu pomocí subVI nacteni_polozek_databaze_do_stromu.vi. V tomto subVI je realizováno vyčtení názvů ECU jednotek, rámců pro čtení a zápis a jednotlivých signálů, zápis do stromů se provede až po vyčtení všech položek, pomocí uzlu vlastností pro přidání položek stromu. Kód pro výše uvedené subVI nachází v příloze D.



Obrázek 9.4: *Kód pro načtení databáze svítilny*

Otevření konfiguračního souboru

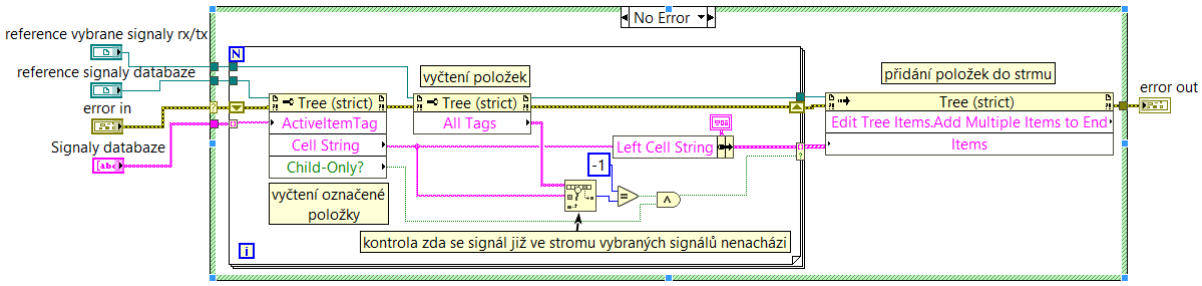
Princip i kód je podobný jako při události Načtení databáze svítilny, akorát je zde přidáno vyčtení dat z konfiguračního souboru. Kdy po resetování všech stromů a jejich označení je otevřen průzkumník souboru, ve kterém uživatel může vybrat konfigurační soubor. Po úspěšném vybrání konfiguračního souboru, se zahájí vyčítání dat z konfiguračního souboru, kdy nejprve je provedeno vyčtení názvu databáze svítilny pro její načtení do stromů, poté jsou vyčteny jednotlivé signály do stromů vybraných signálů, pomocí subVI nacteni_signalu_z_konfig_souboru.vi. V tomto subVI jsou také jednotlivé signály kontrolovány, zda se nachází v databázi svítilny.



Obrázek 9.5: *Kód pro Otevření konfiguračního souboru*

Přidat signál z databáze svítilny do vybraných signálů

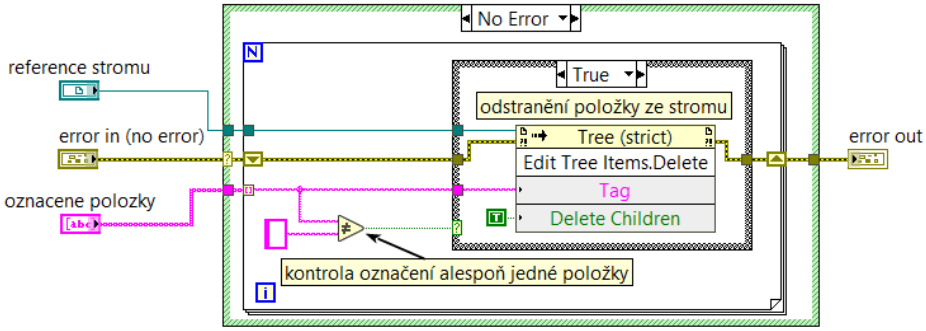
Jedná se o událost, při které jsou označené signály ze stromu signály rx/tx přidány do stromu vybrané signály rx/tx, tak jak je naznačeno ve stavovém diagramu na obrázku 7.4. V kódu je tato událost nazvaná dle tlačítek přidat rx a přidat tx. Pro tuto událost bylo vytvořeno subVI pridani_signalu_do_stromu.vi. Hlavními vstupními proměnnými jsou reference na strom vybrané signály rx/tx a na strom databáze rx/tx a textové pole označených položek. Prvním krokem v tomto subVI je postupné vyčtení označených signálů ze stromu signály rx/tx ve smyčce, pomocí uzlu vlastností. Po té je ve smyčce provedena kontrola zda se signál již nenachází ve vybraných signálech a pokud nenachází, jsou všechny signály splňující tuto podmínku po skončení smyčky přidány do stromu vybrané signály rx/tx, pomocí uzlu vlastností pro přidání více položek stromu.



Obrázek 9.6: *Kód pro přidání signálu*

Odstranit vybraný signál

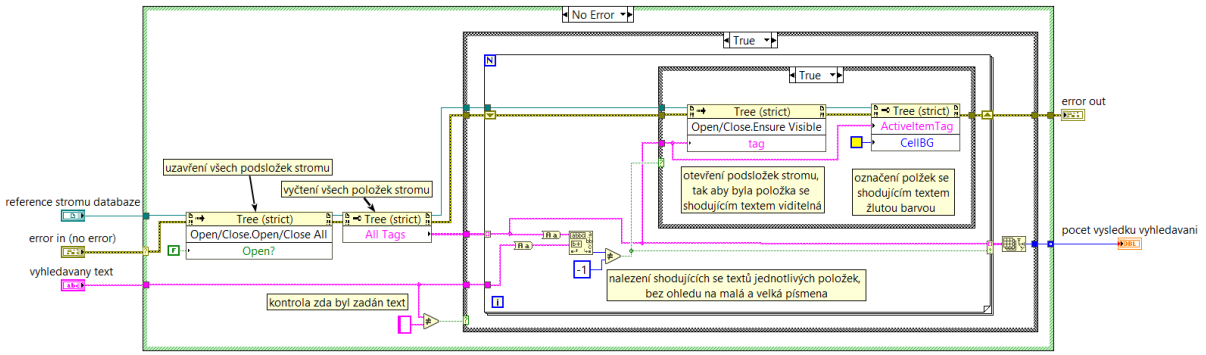
Jedná se o událost, při které jsou označené signály ze stromu vybrané signály rx/tx odstraněny, tak jak je naznačeno ve stavovém diagramu na 7.4. V kódu je tato událost nazvaná dle tlačítek odstranit rx a odstranit tx. Pro tuto událost bylo vytvořeno subVI odstraneni_polozky_ze_stromu.vi. Hlavními vstupními proměnnými je reference na strom vybrané signály rx/tx a textové pole označených položek. V kódu tohoto subVI se provede kontrola, zda byl označen alespoň jeden signál a následně se postupně ve smyčce odstraní jednotlivé označené signály, pomocí uzlu vlastností odstranění položek. Po odstranění se pomocí uzlu vlastností resetuje označení položek ve stromu vybrané signály rx/tx.



Obrázek 9.7: *Kód pro odstranění vybraného signálu ze stromu vybraných signálů*

Vyhledání shodujících se položek ve stromu

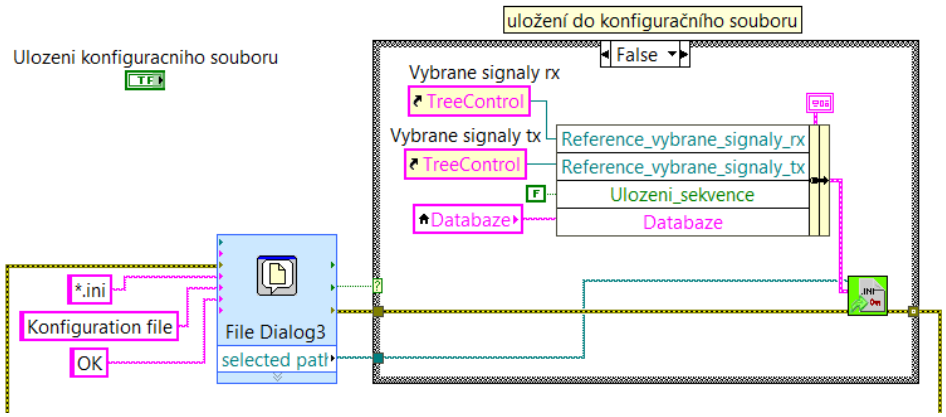
Tato událost slouží pro vyhledání shodujících se položek ve stromu databáze tx/rx se zadaným textem a jejich označení, tak jak je naznačeno ve stavovém diagramu na obrázku 7.4. V kódu je tato událost nazvaná dle tlačítek vyhledat rx/tx. Pro tuto událost bylo vytvořeno subVI vyhledat. vi. Hlavními vstupními proměnnými je reference na strom databáze rx/tx a vyhledávaný text, a výstupní proměnná číselného typu počet výsledků vyhledávání. V kódu tohoto subVI se nejprve provede uzavření všech podsložek ze stromu a vyčtení všech položek stromu pomocí uzlu vlastností. Poté se provede kontrola, zda byl zadán text do proměnné vyhledávaný text, a pokud ano tak se ve smyčce jednotlivě porovnává shodnost všech položek stromu s vyhledávaným textem, bez ohledu na velká a malá písmena. Položky se shodujícím se textem budou následně zviditelněny a označeny žlutou barvou pomocí uzlu vlastností, ostatní položky budou označeny bílou barvou. Pokud nebude zadán žádný text do proměnné vyhledávaný text, budou všechny položky označeny bílou barvou, dojde tak k resetování vyhledávání.



Obrázek 9.8: *Kód pro vyhledání shodujících se položek ve stromu*

Uložení dat (název databáze, vybrané signály a sekvence) do konfiguračního souboru

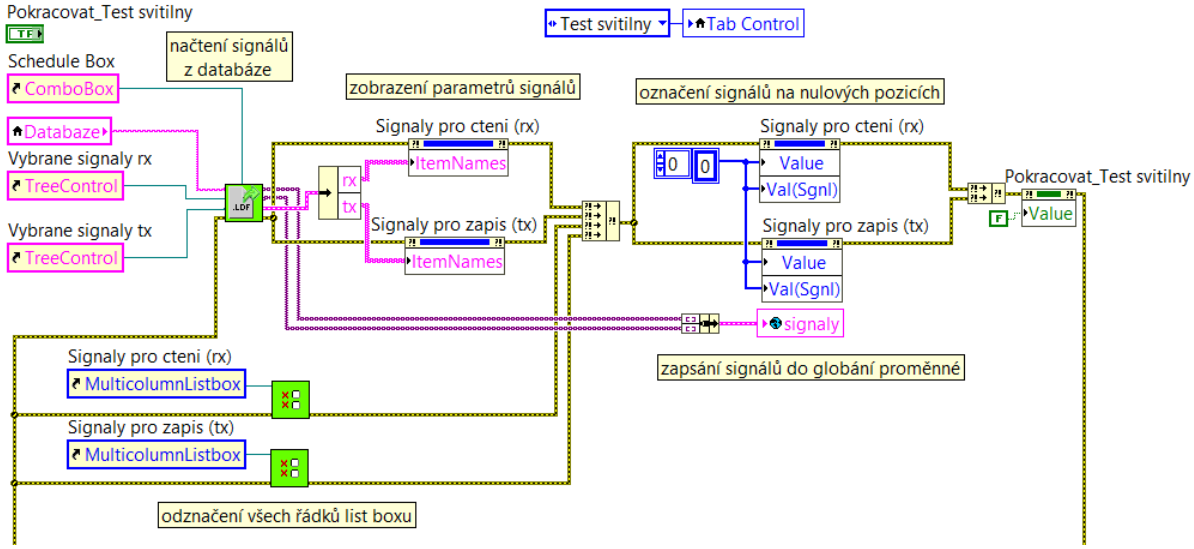
Uložení do konfiguračního souboru je v programu použito dvakrát, jednou pro uložení názvu databáze svítilny a vybraných signálů a podruhé pro uložení sekvence, kdy jsou uloženy i vybrané signály a název databáze svítilny. Princip uložení do konfiguračního souboru je zobrazen ve stavovém diagramu na obrázku 7.4 a 7.5. Uložení do konfiguračního souboru je tedy řešeno pomocí subVI ulozeni_konfig_souboru.vi, které je naprogramováno podle výše uvedených stavových diagramů a podle struktury konfiguračního souboru na obrázku 8.2, kdy je nejprve uložen název databáze svítilny, po té názvy a počty vybraných signálů a nakonec sekvence, která musí být aktivována vstupním přepínačem. Kód pro toto subVI se nachází v příloze G.



Obrázek 9.9: *Kód pro uložení dat do konfiguračního souboru*

Pokračovat do okna Test svítilny

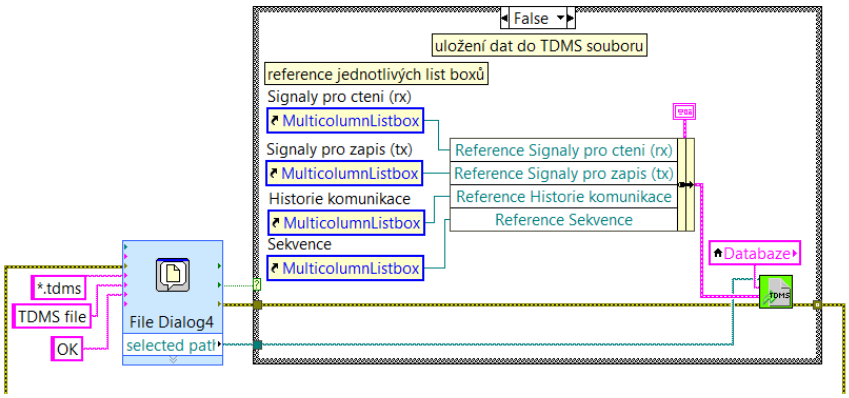
Pokračovat do okna test svítilny je událost pro přejití do okna test svítilny, tak jak je naznačeno ve stavovém diagramu na obrázku 7.4. V kódu této události se provede načtení signálů a parametrů signálů z databáze svítilny, podle názvů signálů. Informace jsou následně zapsány do list boxů signály pro čtení/zápis. Po té se provede odznačení všech řádků list boxů, a nakonec označení signálů na nulových pozicích a zapsání signálů do globální proměnné signály.



Obrázek 9.10: *Kód pro událost pokračovat do okna Test svítilny*

Uložení historie komunikace do TDMS souboru

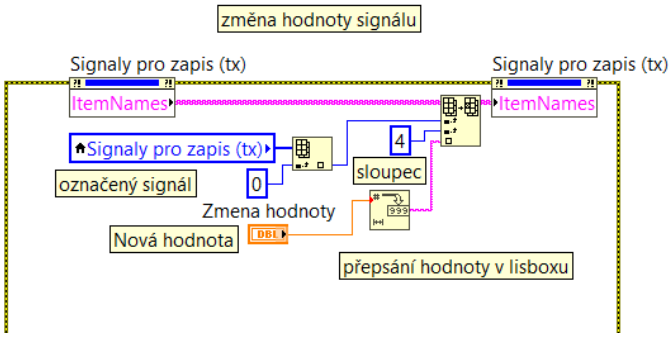
Princip uložení historie komunikace do TDMS souboru je zobrazen ve stavovém diagramu na obrázku 7.5. Uložení do TDMS souboru je v kódu provedeno tak, že na titulní stranu TDMS souboru je uložen název databáze svítilny, datum a čas, počet signálů pro čtení/zápis. Na druhou stranu jsou uloženy signály a jejich informace, na třetí historie komunikace a na poslední stranu celá sekvence. Data jsou uložena tak, jak jsou zobrazena v jednotlivých list boxech. Uložení dat je realizováno pomocí subVI ulozeni_do_TDMS_souboru.vi a kód tohoto subVI se nachází v příloze H.



Obrázek 9.11: *Kód pro uložení dat do TDMS souboru*

Změna hodnoty signálu

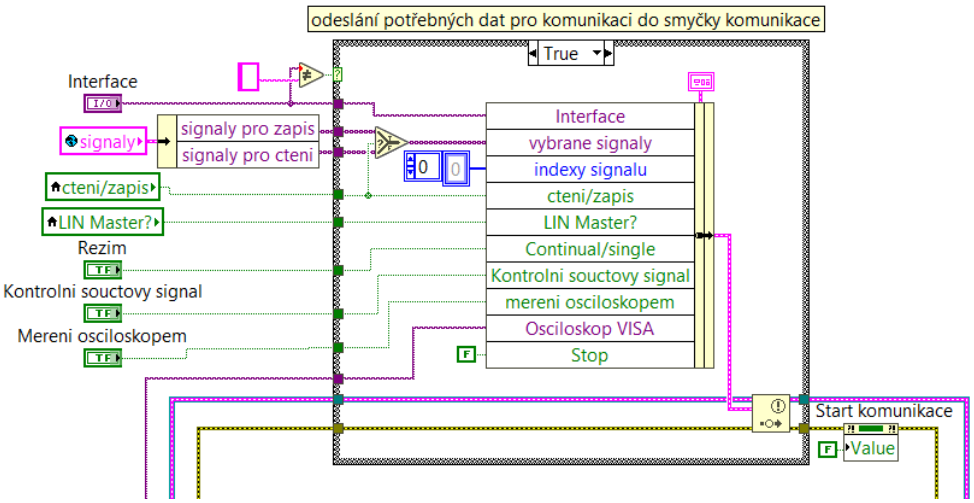
Princip změny hodnoty signálu je zobrazen ve stavovém diagramu na obrázku 7.5. Změna hodnoty je v kódu provedena, pomocí přepsání hodnoty v poli, tak aby byla přepsána hodnota označeného signálu na požadovanou.



Obrázek 9.12: *Kód pro změnu hodnoty*

Start komunikace se svítilnou

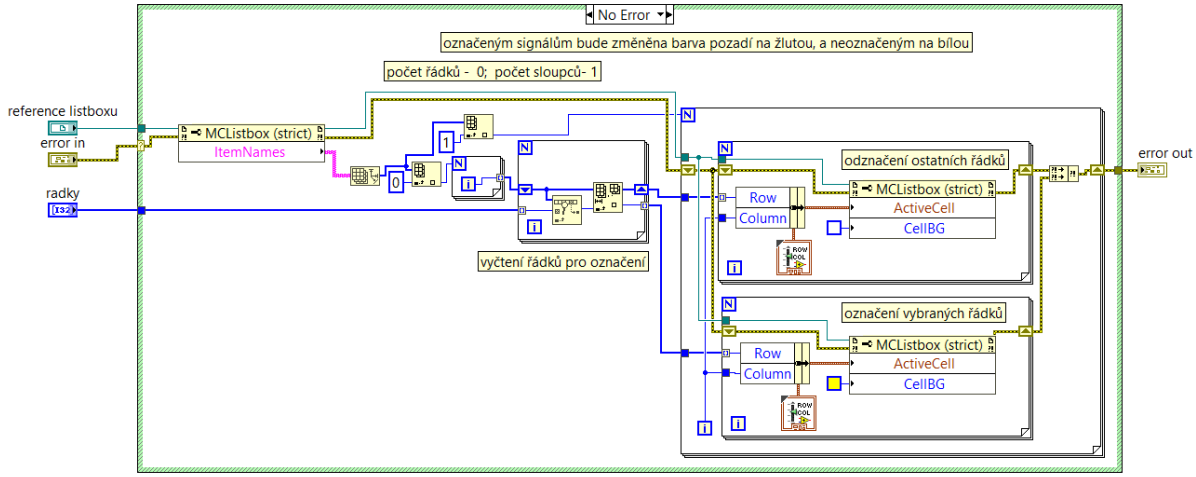
V kódu této události jsou přeposlána data potřebná pro komunikaci do vedlejší smyčky komunikace, pomocí synchronizačního prvku notifikace. Princip této události je zobrazen ve stavovém diagramu na obrázku 7.5.



Obrázek 9.13: *Kód pro start komunikace*

Označení řádku list boxu

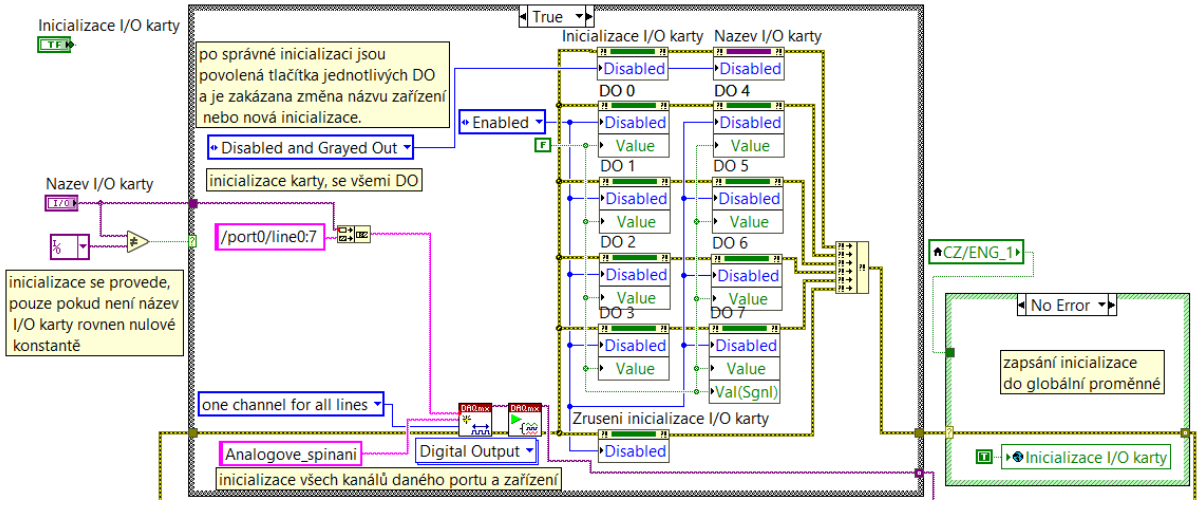
Označení řádku list boxu je řešeno pomocí subVI oznaceni_radku_listboxu.vi, ve kterém jsou pomocí uzlu vlastností ve smyčce, dle počtu sloupců a řádků označeny vybrané řádky žlutou barvou a ostatní bílou. Toto subVI je v kódu využito celkem třikrát pro označení signálů v list boxech signály pro čtení a zápis a pro list box sekvence. Označení řádků list boxu je zavoláno vždy při změně hodnoty list boxu.



Obrázek 9.14: *Kód pro označení řádku list boxu*

Inicializace přístroje

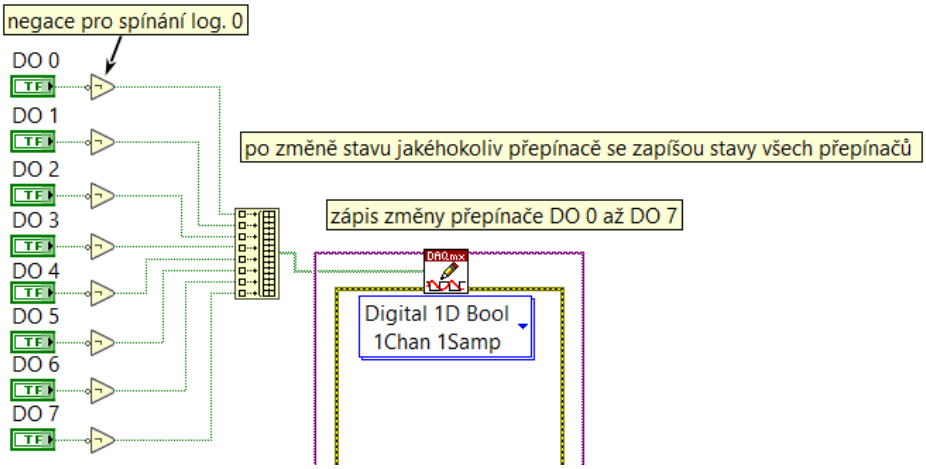
Inicializace přístroje je událost, která se v aplikaci vyskytuje třikrát, pro I/O kartu, zdroj a osciloskop. Princip kódu této události je stejný pro všechny, jsou akorát použity jiné bloky pro inicializaci, kdy každý přístroj má svůj předepsaný blok. Princip je takový že nejprve se provede kontrola, zda byl zadán název přístroje, poté se provede inicializace, povolení ovládacích prvků a nastavení výchozích hodnot pomocí uzlů vlastností a nakonec zapsání inicializace do globální proměnné, pokud inicializace proběhla v pořádku, v opačném případě vypsání zprávy „chybná inicializace“ do vyskakovacího okna. V případě I/O karty se jedná o založení DAQmx kanálu pro všechny digitální výstupy, kterých je 8.



Obrázek 9.15: *Kód pro inicializaci I/O karty*

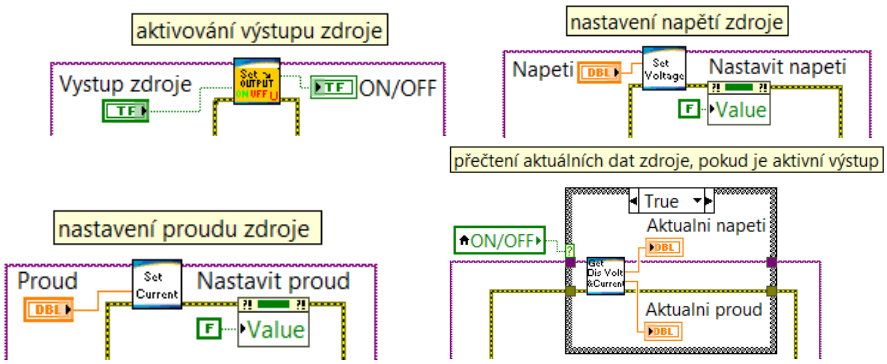
Ovládání přístrojů

Do této události patří konfigurace přístrojů a vyčítání naměřených hodnot, ve kterých jsou použity předepsané Plug and Play ovladače od výrobce. Událostmi jsou nastavení napětí, proudu, výstupu a přečtení aktuálních hodnot zdroje a konfigurace a přečtení naměřených hodnot osciloskopu. Spínání digitálních kanálů I/O karty je realizováno pomocí DAQmx.



Obrázek 9.16: *Kód pro ovládání I/O karty*

Jak lze vidět na obrázku 9.16 spínání digitálních kanálů I/O karty je realizováno v jedné události pro všechny přepínače DO 0 až DO 7, které nastavují hodnotu jednotlivých digitálních kanálů I/O karty, tak že při změně jakéhokoliv přepínače jsou zapsány do I/O karty změny všech přepínačů. Hodnoty jednotlivých přepínačů jsou negovány, protože Arduino relé modul je spínán logickou 0.



Obrázek 9.17: *Kódy pro ovládání zdroje*

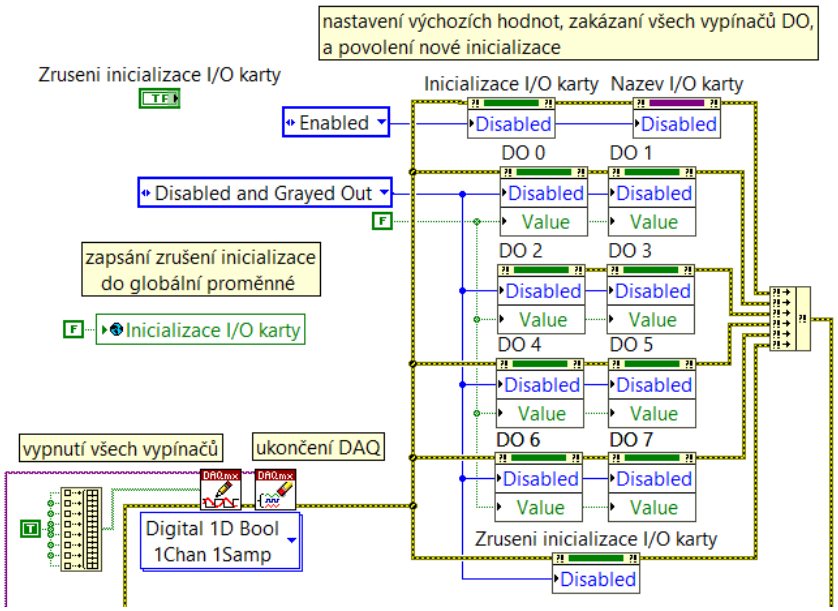
Jak lze vidět na obrázku 9.17 Nastavení napětí, proudu, výstupu zdroje je rozděleno do jednotlivých událostí s reakcí na předepsané tlačítko. Přečtení aktuálních hodnot zdroje je realizováno v události čekání, kdy není ve struktuře událostí vykonávána žádná jiná událost.

Konfigurace osciloskopu je samostatná událost pro tlačítko konfigurace osciloskopu, která nakonfiguruje osciloskop pro dekódování LIN sběrnice, dle nastavených parametrů. Konfigurace osciloskopu pro dekódování LIN sběrnice je prováděná pomocí několik za sebou řazených ovládacích bloků v předepsané posloupnosti, podle manuálu přístroje, kdy nejprve je vypnut kanál Ch1 po té je provedena konfigurace zvoleného digitálního kanálů, komunikační sběrnice, triggeru a časové základny. Kód pro konfiguraci osciloskopu se nachází v příloze E.

Přečtení naměřených hodnot osciloskopu je realizováno ve vedlejší smyčce komunikace, a je zavoláno vždy po zápisu nebo přečtení zprávy komunikace pokud je měření osciloskopem povoleno.

Zrušení inicializace přístroje

Zrušení inicializace přístroje je událost, je událost pro ukončení komunikace s přístrojem a nastavení výchozího stavu. Tato událost se v aplikaci vyskytuje třikrát pro I/O kartu, zdroj a osciloskop. Princip kódu této události je stejný pro všechny, jsou akorát použity jiné bloky pro ukončení komunikace s přístrojem, kdy každý přístroj má svůj předepsaný blok. Princip je takový, že nejprve se provede ukončení komunikace s přístrojem, potom následuje zakázání a povolení ovládacích prvků do výchozího stavu a zapsání do globální proměnné inicializace určeného přístroje.



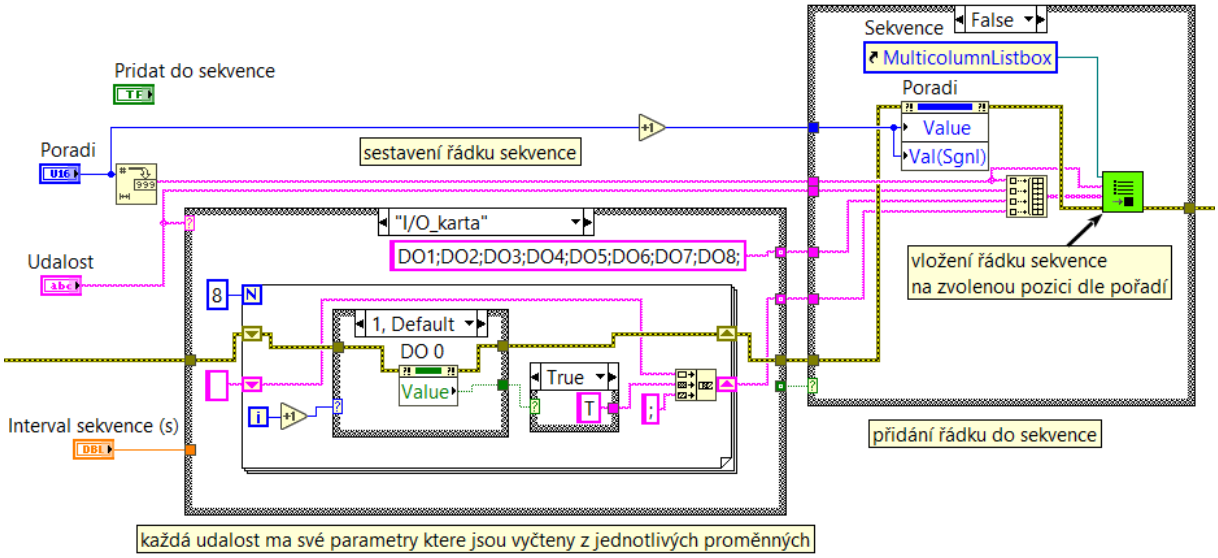
Obrázek 9.18: *Kód pro zrušení inicializace I/O karty*

Přepínání mezi jazyky

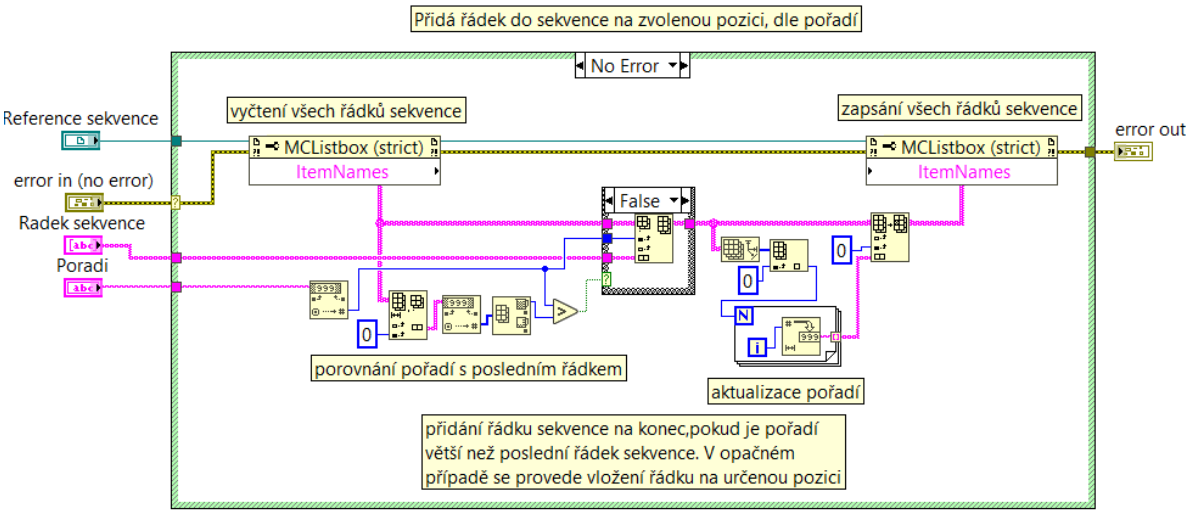
Přepínání mezi jazyky je událost na tlačítko CZ/ENG_1, 2, 3. V kódu je přepínání mezi jazyky realizováno pomocí uzlů vlastností pro každý prvek postupně ve smyčce, kdy pro český jazyk je zobrazen popisek nazvaný „label“ a popisek nazvaný „caption“ je skrytý, pro anglický jazyk je to opačné.

Přidat událost do sekvence

Přidat událost do sekvence je událost, kdy je do list boxu sekvence přidán řádek sekvence se zvolenou událostí a parametry. Princip této události je zobrazen ve stavovém diagramu na obrázku 7.6. V kódu je to provedeno tak, že podle zvolené události jsou vyčteny jednotlivé hodnoty určených proměnných a ty jsou následně zapsány do řádku sekvence, který je poté vložen na zvolenou pozici. Vkládání řádků je hlídáno, tak aby pořadí řádků nebylo stejné a byla dodržena posloupnost, pomocí subVI pridani_radku_do_sekvence.vi.



Obrázek 9.19: *Kód pro přidání události do sekvence*

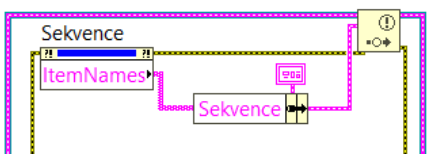


Obrázek 9.20: *subVI pridani_radku_do_sekvence.vi.*

Jak je vidět na obrázku 9.20 v tomto subVI se nejprve provede porovnání pořadí s posledním řádkem sekvence a podle je nový řádek vložen na zvolenou pozici nebo na poslední pozici, poté následuje aktualizace pořadí, která upraví pořadí všech řádků sekvence, tak aby odpovídaly pozici v list boxu.

Start sekvence

V kódu této události jsou přeposlána data potřebná pro sekvenci do vedlejší smyčky sekvence, pomocí synchronizačního prvku notifikace. Princip této události je zobrazen ve stavovém diagramu na obrázku 7.6.



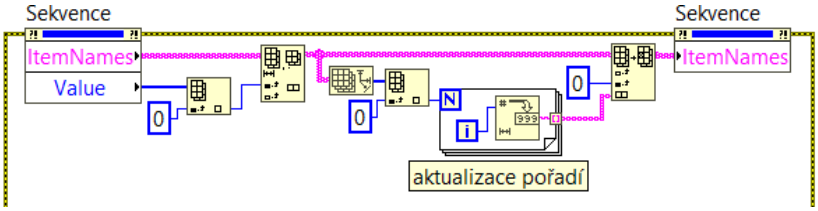
Obrázek 9.21: *Kód pro start sekvence*

Reset sekvence a historie komunikace

Reset list boxu je proveden tak že po stisknutí tlačítka reset sekvence nebo historie komunikace je pomocí uzlu vlastností resetovaný požadovaný list box, podle stisknutého tlačítka.

Odebrání události ze sekvence

Princip této události je zobrazen ve stavovém diagramu na obrázku 7.6. Odebrání události ze sekvence je realizováno, pomocí uzlu vlastností kdy je odstraněn označený řádek list boxu sekvence a následně je aktualizováno pořadí řádků list boxu, podobně jako při přidání řádku do list boxu.



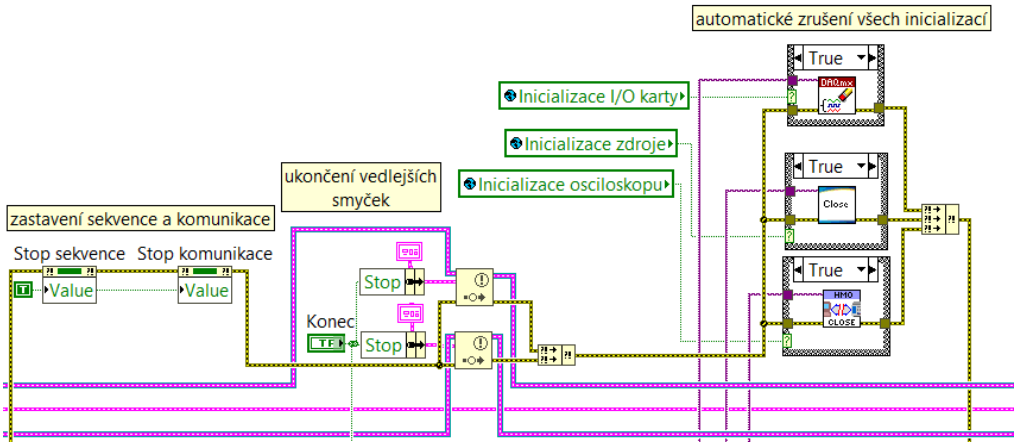
Obrázek 9.22: *Kód pro odebrání události ze sekvence*

Nápověda

V aplikaci se nachází celkem 3 nápovědy, tedy pro každé okno jedna. Uvnitř kódu těchto událostí je pouze předáno textová konstanta s textem nápovědy do funkce pro zobrazení vyskakovacího okna.

Konec

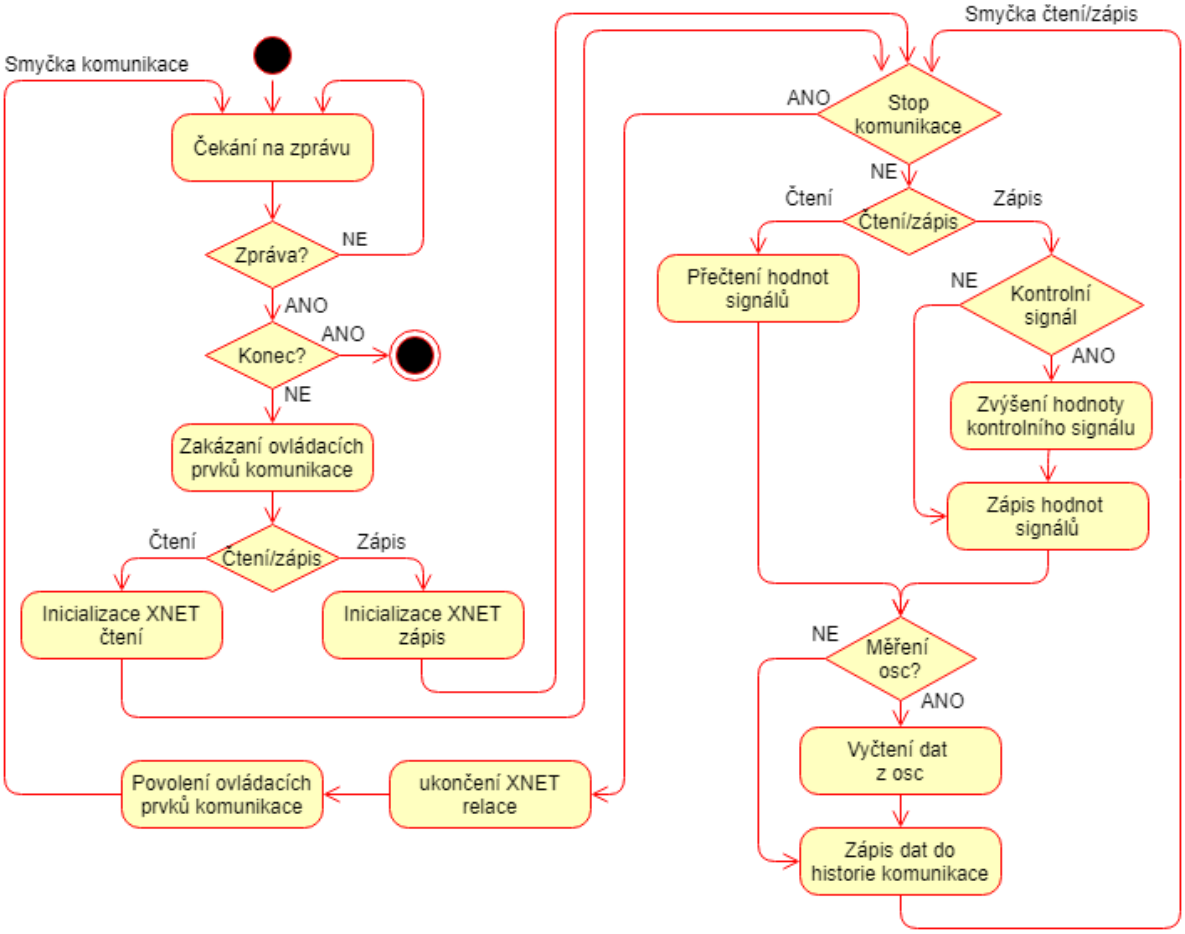
Tato událost slouží pro ukončení celé aplikace. Nejprve je tedy ukončena smyčka čtení/zápis a smyčka událostí, poté je ukončena i smyčka komunikace a sekvence, pomocí přeposlání parametru stop opět pomocí synchronizačního prvku notifikace. Posledními kroky jsou automatické ukončení relací jednotlivých přístrojů, pokud byly inicializovány a ukončení hlavní smyčky. Po ukončení hlavní smyčky ještě před ukončením celé aplikace následuje ukončení synchronizačních prvků notifikace a blok pro zobrazení případných chyb.



Obrázek 9.23: *Kód pro ukončení celé aplikace*

9.2 Vedlejší smyčka komunikace

Vedlejší smyčka komunikace slouží ke komunikaci se svítilnou po sběrnici LIN. Ve smyčce jsou využity především bloky NI-XNET.



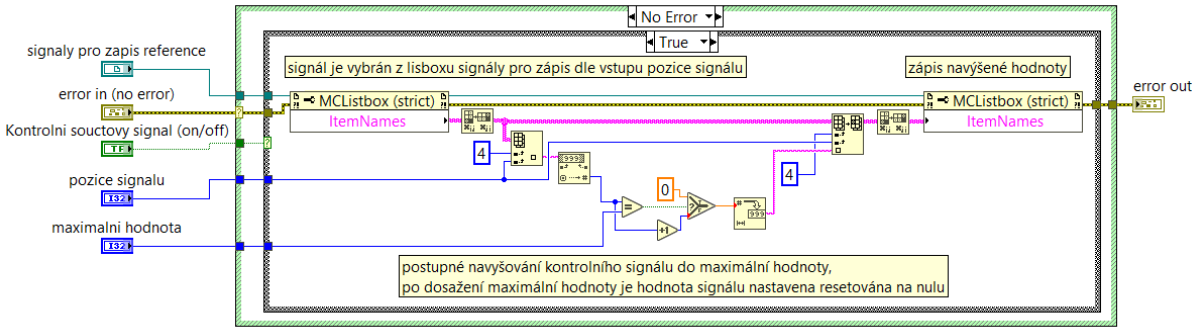
Obrázek 9.24: *Vývojový diagram pro smyčku komunikace*

Jak lze vidět ve vývojovém diagramu na obrázku 9.24 tato smyčka čeká na data z hlavní smyčky, data jsou do této smyčky přeposlány v události start komunikace nebo konec, pomocí synchronizačního prvku notifikace, kdy při události konec je přeposlán pouze parametr pro ukončení celé této smyčky. Ve zprávě při události start komunikace jsou přeposlány vstupní data potřebné pro komunikaci, kterými jsou interface, signály pro čtení/zápis, čtení/zápis komunikace, LIN Master, režim komunikace, kontrolní součtový signál, měření osciloskopem a interface osciloskopu. V této smyčce se tedy nejprve provede vyčtení zprávy ze synchronizačního prvku notifikace, poté se provede zakázání ovládacích prvků komunikace, kromě časového intervalu a stop komunikace a provede se zapnutí kontrolky komunikace aktivní. Po zakázání ovládacích prvků následuje vytvoření NI-XNET relace v režimu čtení nebo zápis, dle zadaného parametru čtení/zápis a nastavení Schedule, pokud je LIN Master aktivní v subVI inicializace komunikace. Potom následuje další smyčka čtení/zápis, která provádí komunikaci, dokud není stisknuto tlačítko stop komunikace, které zastaví tuto smyčku a umožní následné ukončení vytvořené NI-XNET relace a nastaví ovládací prvky opět do výchozího stavu a nakonec ukončí i smyčku komunikace. Ve smyčce čtení/zápis se provádí čtení nebo zápis dle zadaného vstupního parametru čtení/zápis.

Smyčka čtení/zápis obsahuje v režimu zápisu jednotlivé subVI v tomto pořadí kontrolní signál, zápis, měření osciloskopem a historie komunikace, a v režimu čtení subVI čtení, měření osciloskopem a historie komunikace. V případě neznámé chyby je smyčka komunikace ukončena okamžitě i s celou aplikací, pokud se jedná o známé chyby, jsou tyto chyby zobrazeny ve vyskakovacím okně a není ukončená smyčka komunikace ani celá aplikace. Známými chybami jsou chybný interface, kdy byl zadán špatný název interface LIN karty, nepřipojené napájení sběrnice a chybný název schedule. Tyto chyby jsou řešeny pomocí subVI chyby komunikace. Zdrojový kód celé smyčky je zobrazen v příloze I a další kapitoly popisují jednotlivé subVI vedlejší smyčky komunikace.

9.2.1 subVI kontrolní signál

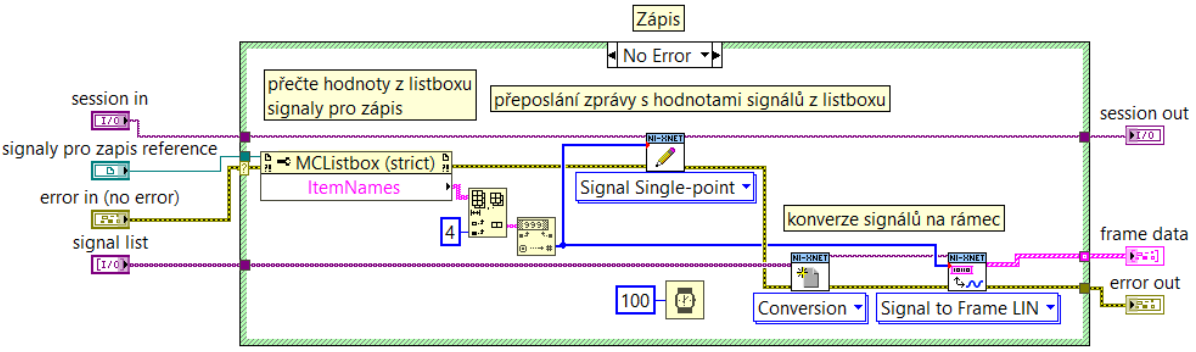
Vstupními parametry pro toto subVI jsou reference na signály pro zápis, přepínač kontrolní součtový signál, vstupní chybový cluster a dva číslicové vstupy pozice signálu a maximální hodnota. SubVI kontrolní signál funguje tak, že postupně při každém spuštění přičítá 1 k hodnotě zadaného signálu, až do maximální hodnoty a poté resetuje hodnotu na 0. Kontrolní signál je zadán, pomocí číslicového vstupu pozice signálu. Tato změna je prováděna v list boxu, z kterého jsou hodnoty jednotlivých signálů při komunikaci vyčítány.



Obrázek 9.25: *Kód pro subVI kontrolní signál*

9.2.2 subVI zápis

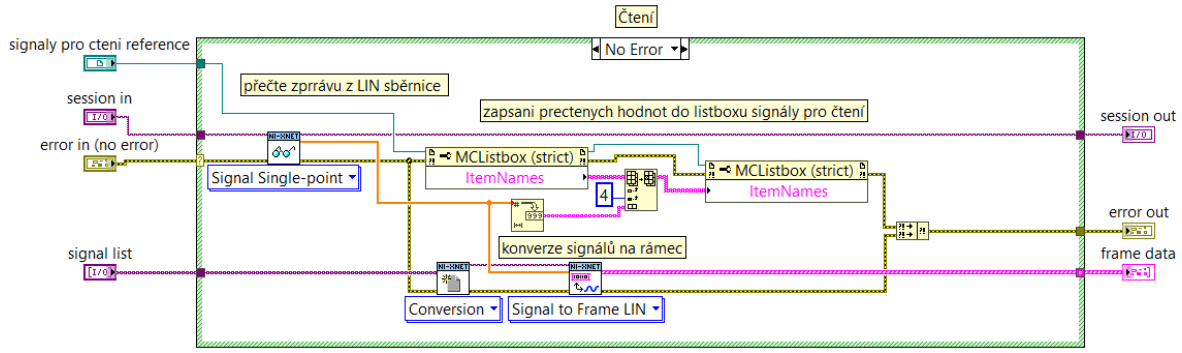
Toto subVI slouží pro zápis hodnot signálu. V kódu se nejprve vyčtou hodnoty signálů pro zápis z list boxu signály pro zápis a ty se pak následně zapíšou na sběrnici, pomocí bloku XNET write.vi. Po zápise jsou následně signály se zapsanými hodnoty převedeny na rámec, pomocí bloku XNET convert.vi, tak aby byl umožněn zápis do historie komunikace.



Obrázek 9.26: *Kód pro subVI zápis*

9.2.3 subVI čtení

Toto subVI slouží pro čtení hodnot signálu. V kódu se nejprve vyčtou hodnoty signálů, pomocí bloku XNET read.vi, a ty se následně zapíšou do list boxu signály pro čtení na pozici hodnot jednotlivých signálů. Po vyčtení jsou následně signály se zapsanými hodnoty převedeny na rámec, pomocí bloku XNET convert.vi, tak aby byl umožněn zápis do historie komunikace.



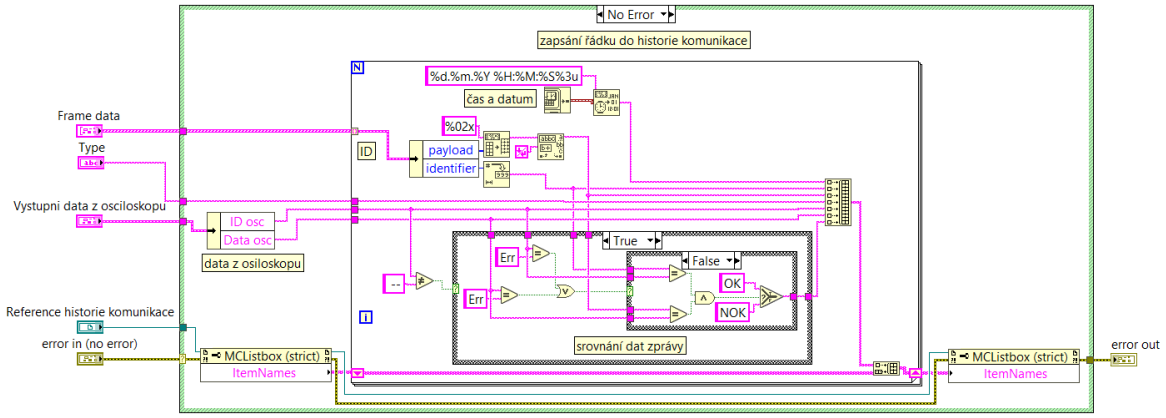
Obrázek 9.27: *Kód pro subVI čtení*

9.2.4 subVI měření osciloskopem

SubVI měření osciloskopem provádí vyčítání naměřených hodnot z osciloskopu, konkrétně tedy aktuálně dekódovaný rámec komunikace. V tomto subVI jsou použity především ovladače k osciloskopu vytvořené výrobcem. V kódu je nejprve zjištěn počet změřených rámců, na základě toho je spuštěna smyčka pro vyčtení rámců komunikace, která je ukončena po správném vyčtení celého rámce, po přečtení neúspěšném přečtení všech zpráv nebo při chybě. V této smyčce jsou postupně vyčítány informace o rámci, kterými jsou status rámce, ID rámce, počet B, data. Výstupem tohoto subVI jsou výstupní data osciloskopu, ve kterých je ID a data rámce, v případě že došlo k chybě měření je ve výstupu zapsána zpráva „Err“. Kód tohoto subVI se nachází v příloze F.

9.2.5 subVI historie komunikace

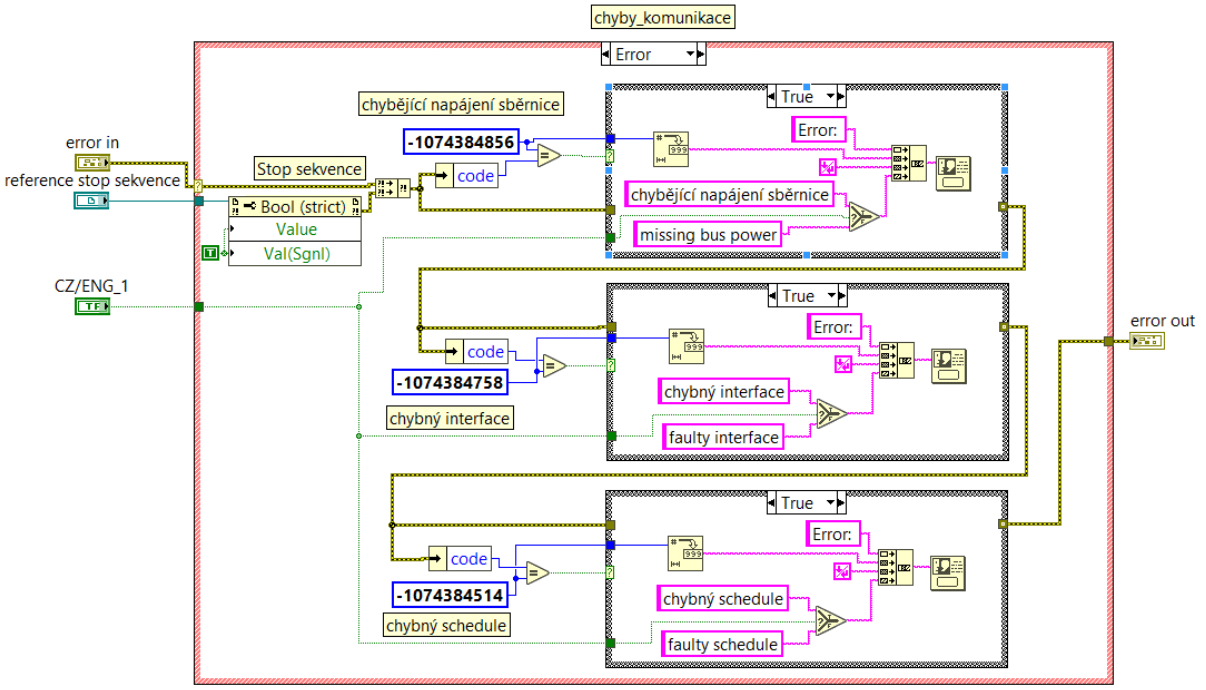
Toto subVI uspořádá data do jednotlivých sloupců řádku a srovná ID a data zprávy z komunikace se změřenými data osciloskopu a následně zapíše řádek do list boxu historie komunikace.



Obrázek 9.28: *Kód pro subVI historie komunikace*

9.2.6 **subVI chyby komunikace**

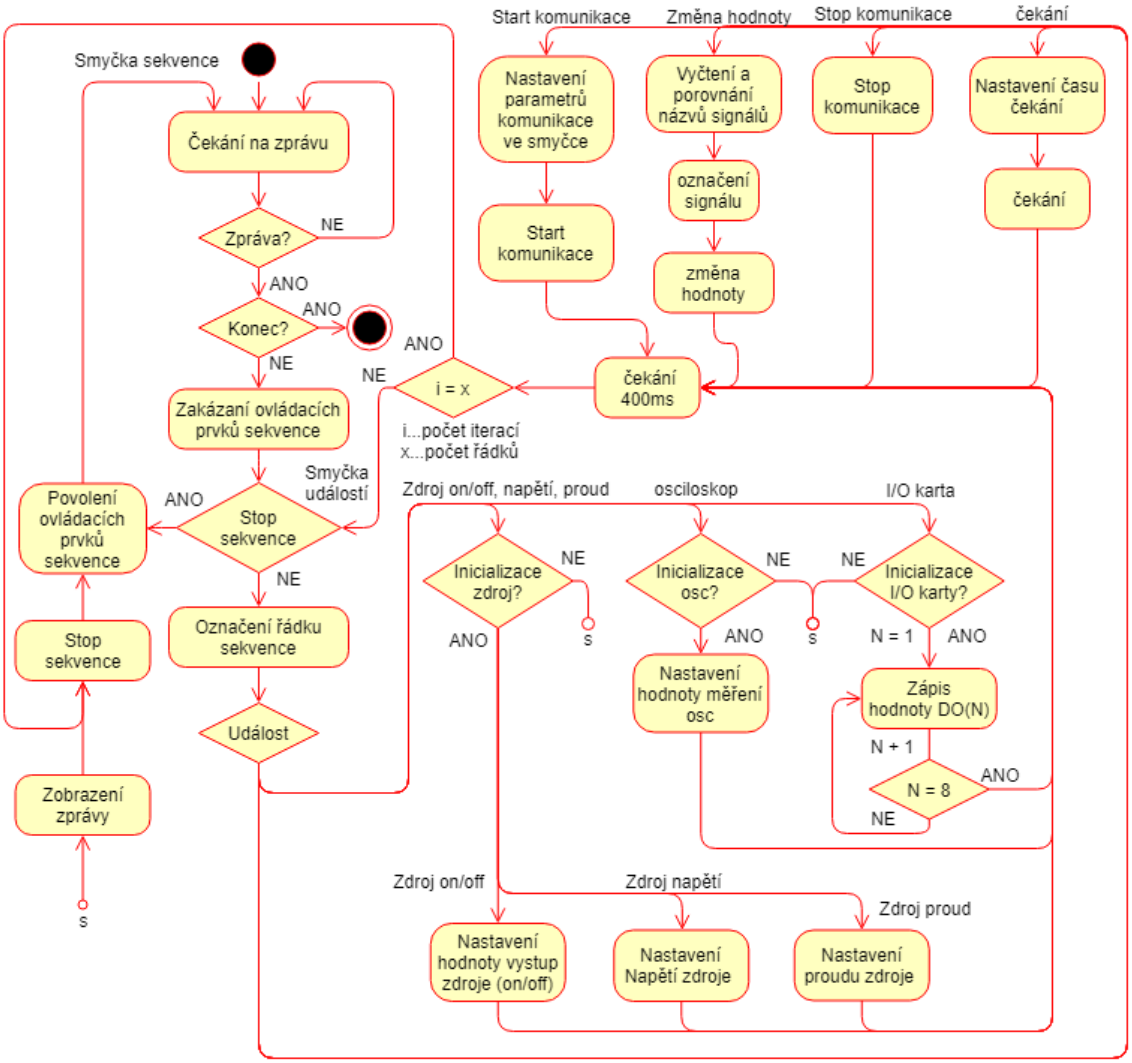
Toto subVI zobrazí a odstraní známé chyby komunikace, kterými jsou chybný interface, kdy byl zadán špatný název interface LIN karty, nepřipojené napájení sběrnice a chybný název Schedule. V kódu jsou v případě chyby porovnávány jednotlivé kódy chyba dle nich je následně chyba vyhodnocena a zobrazena ve vyskakovacím okně.



Obrázek 9.29: *Kód pro subVI chyby komunikace*

9.3 Vedlejší smyčka sekvence

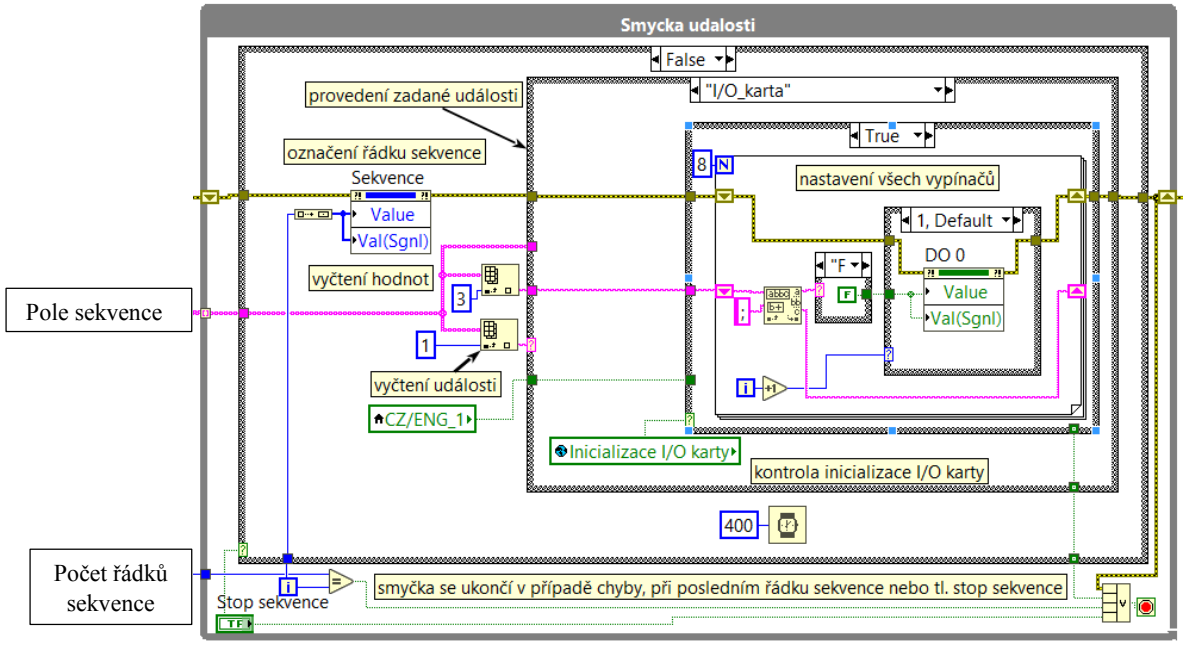
Vedlejší smyčka sekvence vykonává definovanou sekvenci uživatelem po stisknutí tlačítka start sekvence. Jednotlivé události sekvence jsou v kódu vykonávány pomocí uzlů vlastností, jedná se tak například o programovou změnu hodnoty vypínače, na kterou již zareaguje standardně hlavní smyčka a provede danou událost.



Obrázek 9.30: *Vývojový diagram pro smyčku sekvence*

Jak lze vidět ve vývojovém diagramu na obrázku 9.30 tato smyčka čeká na data z hlavní smyčky, data jsou do této smyčky přeposlány v události start sekvence nebo konec, pomocí synchronizačního prvku notifikace, kdy při události konec je přeposlán pouze parametr pro ukončení celé této smyčky. Ve zprávě při události start sekvence jsou přeposlána vstupní data potřebné pro sekvenci, které jsou zapsány v poli sekvence. V poli sekvence jsou úplně stejná data jako v list boxu sekvence. V této smyčce se tedy nejprve provede vyčtení zprávy ze synchronizačního prvku notifikace, poté se provede zakázání ovládacích prvků sekvence, kromě tlačítka stop sekvence a provede se zapnutí kontrolky sekvence aktivní. Poté následuje smyčka událostí, ve které jsou postupně vykonávány jednotlivé řádky sekvence pomocí natavení uzlu vlastností s pevným časovým intervalem 400 ms. Smyčka událostí je ukončena při posledním řádku sekvence, při chybě nebo po stisknutí tlačítka stop sekvence.

Po ukončení smyčky událostí jsou nakonec povoleny všechny ovládací prvky sekvence. V případě neznámé chyby je smyčka sekvence, smyčka událostí ukončena ihned i s celou aplikací, pokud je chyba známa bude chyba zobrazena ve vyskakovacím okně a ukončena pouze smyčka událostí, nastane tak ukončení sekvence. Známými chyby jsou chybějící inicializace jednotlivých přístrojů, které je potřeba k vykonání událostí těchto přístrojů, pokud jsou v sekvenci. Zdrojový kód pro celou smyčku komunikace se nachází v příloze J.



Obrázek 9.31: *Řešení kódu smyčky událostí pro I/O kartu*

Na obrázku 9.31 je zobrazeno řešení kódu smyčky událostí pro I/O kartu, tedy pro spínání analogových funkcí, ostatní události jsou řešeny obdobně. V kódu jsou ve smyčce událostí postupně vyčítány jednotlivé řádky z pole sekvence, kdy v každé iteraci smyčky je z řádku vyčtená požadovaná událost, která je následně provedena. Událost je většinou provedena pomocí programové změny požadovaného ovládacího prvku přes uzel vlastností, například změna hodnoty vypínače. Z řádku pole sekvence jsou tak vyčteny hodnoty ovládacích prvků, které jsou převedeny požadovaný datový typ a stanoví tak hodnotu přímo daného ovládacího prvku přes uzel vlastností. U jednotlivých přístrojů je navíc před samotným provedením prováděna kontrola, zda byl přístroj inicializován tak aby událost byla provedena, jen pokud byl přístroj inicializován, pomocí globální proměnné, ve které je uložena tato informace.

10 Závěr

V této práci byl v teoretické části proveden rozbor typu světlometů a svítilen, princip jejich ovládání především pomocí sběrnice LIN. Dále byl také proveden rozbor grafického programovacího prostředí LabVIEW a jeho doplňkových softwarových nástrojů pro ovládání přístrojů tzv. NI-VISA a pro komunikaci po sběrnici LIN, CAN a Flexray tzv. NI-XNET.

V praktické části byl dle požadavků firmy vybrán koncept HW, který zajišťuje ovládání zadní svítilny. Do konceptu HW patří PC, Laboratorní zdroj, I/O karta, Osciloskop, LIN karta a testovaná zadní svítilna.

Pomocí vývojové prostředí LabVIEW byla vytvořena firmou požadovaná aplikace, která obsahuje 3 okna databáze svítilny, výběr signálů a test svítilny. Je tak zajištěno dodržení posloupnosti ovládání svítilny a uživatelský interface je navržen tak, aby byl přehledný.

Při vytváření této diplomové práce jsem narazil na problémy s neznalostí komunikační sběrnice LIN a s tím spojeným programováním v LabVIEW při použití NI-XNET bloků. Musel jsem si tedy danou problematiku před vytvářením této práce nastudovat.

Dle požadavků tak vznikl SW, který se bude používat ve firmě při vývoji pro ovládání svítilen, a který bude v zaměstnání dále rozšířen pro testování svítilen a případně i světlometů.

Při vytváření této diplomové práce jsem se dozvěděl řadu informací o ovládání a testování moderní automobilové svítilny, především o ovládání pomocí komunikace po sběrnici LIN a také použití programovacího prostředí LabVIEW pro tuto oblast.

Literatura

[1] *Lighting Systems* [online]. Francie: Valeo, 2015 [cit. 2020-04-29]. Dostupné z: https://www.dalroad.com/wp-content/uploads/2015/06/lighting_systems_from_light_to_advanced_vision_technologies_technical_handbook_valeoscope_en_998542_web.pdf

[2] Bielawny, Andreas & Schupp, Thorsten & Neumann, Cornelius. (2016). Automotive Lighting Continues to Evolve. Optics and Photonics News. 27. 36-43. 10.1364/OPN.27.11.000036.

[3] VAVERKA, Lukáš. *Žárovka, výbojka, diody a laser: Jak fungují různé typy světlometů* [online]. ČR: Mladá fronta, 2019 [cit. 2020-05-03]. Dostupné z: https://autobible.euro.cz/zarovka-vybojka-diody-laser-funguji-ruzne-typy-svetlometu/

[4] LAŽANSKÝ, Milan. *Do aut se teď montují čtyři typy světlometů. Vyznáte se v nich?* [online]. ČR: CZECH NEWS CENTER, 2016 [cit. 2020-05-03]. Dostupné z: https://www.autorevue.cz/do-aut-se-ted-montuji-ctyri-typy-svetlometu-vyznate-se-v-nich

[5] SAJDL, Jan. *Xenonové světlomety* [online]. ČR: autolexicon.net., 2020 [cit. 2020-05-03]. Dostupné z: https://www.autolexicon.net/cs/articles/xenonove-svetlomety-vybojky/

[6] SAJDL, Jan. *Bi-Xenonové světlomety* [online]. ČR: autolexicon.net., 2020 [cit. 2020-05-03]. Dostupné z: https://www.autolexicon.net/cs/articles/bi-xenonove-svetlomety-vybojky/

[7] HANKE, Petr. Audi Matrix LED & Laser [online]. CZ: Automotorevue, 2015 [cit. 2020-04-29]. Dostupné z: https://www.automobilrevue.cz/rubriky/clanky/technika/audi-matrix-led-laser-jeste-ucinnejsi_44442.html

[8] Audi s8 TFSI. In: *Rychle, komfortně a s lehkostí: Audi S8 TFSI* [online]. ČR: Porsche Česká republika, 2020 [cit. 2020-05-04]. Dostupné z: https://www.audi.cz/media/GalleryThumbnails_Slider_Image_Component/58016-559240-slider-342444/dh-1460-a9ad28/6b9fd893/1585826270/1920x1080-as8-d-181010-2-oe.jpg

[9] BARTHEL, JOCHEN a GERT RUDOLPH. *Lighting technology for the modern automobile* [online]. USA: AspenCore, 2019 [cit. 2020-05-03]. Dostupné z: https://www.edn.com/lighting-technology-for-the-modern-automobile/

[10] TARANOVICH, STEVE. *What is the body control module in modern automobiles?* [online]. USA: AspenCore, 2017 [cit. 2020-05-03]. Dostupné z: https://www.edn.com/what-is-the-body-control-module-in-modern-automobiles/

[11] *Elektronika* [online]. ČR: Varroc Group, 2019 [cit. 2020-05-03]. Dostupné z: https://www.varroclighting.com/product/SitePages/Electronics.aspx

[12] *LIN Specification Package* [online]. Revision 2.2A. Germany: LIN Consortium, 2010 [cit. 2019-12-12]. Dostupné z: https://www.cs-group.de/wp-content/uploads/2016/11/LIN_Specification_Package_2.2A.pdf

[13] *LIN Physical Layer* [online]. USA: Microchip Technology, 2019 [cit. 2019-12-12]. Dostupné z: https://microchipdeveloper.com/lin:protocol-physical-layer

[14] *LIN Description File (LDF)* [online]. USA: Microchip Technology, 2019 [cit. 2019-12-12]. Dostupné z: https://microchipdeveloper.com/lin:protocol-app-ldf

[15] *LIN Message Frame* [online]. USA: Microchip Technology, 2019 [cit. 2019-12-12]. Dostupné z: https://microchipdeveloper.com/lin:protocol-dll-lin-message-frame

[16] *Introduction to the Local Interconnect Network (LIN) Bus* [online]. USA: National Instruments, 2019 [cit. 2019-12-12]. Dostupné z: http://www.ni.com/cs-cz/innovations/white-papers/09/introduction-to-the-local-interconnect-network--lin--bus.html

[17] SUTORÝ, Tomáš. *LIN* [online]. Brno: Vysoké učení technické v Brně, 2004 [cit. 2019-12-12]. Dostupné z: http://www.elektrorevue.cz/clanky/04012/index.html

[18] VLACH, Jaroslav, Josef HAVLÍČEK a Martin VLACH. *Začínáme s LabVIEW*. 1. vyd. Ilustrace Viktorie Vlachová. Praha: BEN - technická literatura, 2008, 247 s. ISBN 978-80-7300-245-9.

[19] BRESS, Thomas J. *Effective labview programming.* 1st ed. Allendale: NTS Press, 2013, 701 s. ISBN 19-348-9108-8.

[20] WITTASSEK, Tomáš. *VIRTUÁLNÍ INSTRUMENTACE I*. Ostrava: VŠB – Technická univerzita Ostrava, 2012. ISBN CZ.1.07/2.2.00/15.0113.

[21] BILÍK, Petr. *VIRTUÁLNÍ INSTRUMENTACE 2*. Ostrava: VŠB – Technická univerzita Ostrava, 2012. ISBN CZ.1.07/2.2.00/15.0113.

[22] BILÍK, Petr. *SYSTÉMY PRO MĚŘENÍ A SBĚR DAT*. Ostrava: VŠB – Technická univerzita Ostrava, 2012. ISBN CZ.1.07/2.2.00/15.0113.

[23] *NI-VISA Overview* [online]. USA: National Instruments, 2020 [cit. 2020-04-06]. Dostupné z: https://www.ni.com/cs-cz/support/documentation/supplemental/06/ni-visa-overview.html

[24] *Developing LabVIEW Plug and Play Instrument Drivers* [online]. USA: National Intruments, 2018 [cit. 2020-04-06]. Dostupné z: http://www.ni.com/tutorial/3271/en/

[25] *XNET: NI-XNET Hardware and Software Manual*. National Instruments. USA, 2014. Dostupné také z: http://www.ni.com/pdf/manuals/372840h.pdf

[26] *NI-XNET CAN, LIN, and FlexRay Platform Overview* [online]. USA: National Instruments, 2019 [cit. 2020-05-04]. Dostupné z: https://www.ni.com/cs-cz/innovations/white-papers/09/ni-xnet-can--lin--and-flexray-platform-overview.html

[27] Advanced Database Example Using Property Nodes. In: *Database Programming* [online]. USA: National Instruments, 2019 [cit. 2019-12-12]. Dostupné z: http://zone.ni.com/images/reference/en-XX/help/372841W-01/advanceddatabaseexample.gif

[28] LIN Signal *Output* Single Point, example. National Instruments, 2014, *LIN Signal Output Single Point.vi*, součást instalace NI-XNET

[29] *NI cDAQ™-9174*. National Instrument. USA, 2013. Dostupné také z: https://www.ni.com/pdf/manuals/374045a.pdf

[30] *CDAQ-9174: compactDAQ Chassis* [online]. USA: National Instruments, 2019 [cit. 2019-12-12]. Dostupné z: http://www.ni.com/cs-cz/support/model.cdaq-9174.html

[31] *C Series LIN Interface Module* [online]. USA: National Instruments, 2019 [cit. 2019-12-12]. Dostupné z: http://www.ni.com/cs-cz/shop/select/c-series-lin-interface-module

[32] *NI 9866*. National Instruments. USA, 2012. Dostupné také z: http://www.ni.com/pdf/manuals/373710b.pdf

[33] *Multifunction I/O Device* [online]. USA: National Instrument, 2019 [cit. 2019-12-12]. Dostupné z: http://www.ni.com/cs-cz/shop/select/multifunction-io-device

[34] *NI USB-6002*. National Instrument. USA, 2019. Dostupné také z: http://www.ni.com/pdf/manuals/374371a.pdf

[35] *Remote Programming Lab.Grade Switching Mode Power Supply: HCS-3402 USB* [online]. Hongkong: Manson Engineering Industrial, 2017 [cit. 2019-12-12]. Dostupné z: https://www.manson.com.hk/product/hcs-3402-usb/

[36] *Relé modul 8 kanálů s optickým oddělením* [online]. CZ: GM electronic, 2019 [cit. 2019-12-12]. Dostupné z: https://www.gme.cz/rele-modul-8-kanalu-s-optickym-oddelenim

[37] *LM2596 Buck* [online]. Havlíčkův Brod: ECLIPSERA, 2019 [cit. 2019-12-12]. Dostupné z: https://arduino-shop.cz/arduino/1332-lm2596-buck-step-down-napajeci-modul-dc-4-0-40-1-3-37v-led-voltmetr.html

[38] *R&S®RTC1000 oscilloscope* [online]. Philippines: Rohde & Schwarz, 2019 [cit. 2019-12-12]. Dostupné z: https://www.rohde-schwarz.com/ph/product/rtc1000-productstartpage_63493-515585.html

[39] *R&S®RTC1000 Digital Oscilloscope User Manual* [online]. Germany: Rohde & Schwarz GmbH & Co., 2018 [cit. 2020-05-04]. 1335.7352.02. Dostupné z: https://scdn.rohde-schwarz.com/ur/pws/dl_downloads/dl_common_library/dl_manuals/gb_1/r/rtc/RTC_UserManual_en_04.pdf

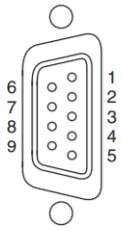
[40] USB-6002 pinout. In: *USB-6002* [online]. USA: National Instruments, 2019 [cit. 2019-12-12]. Dostupné z: http://zone.ni.com/images/reference/en-XX/help/370466AH-01/loc_6001_6002_6003_pinout.gif

Seznam příloh

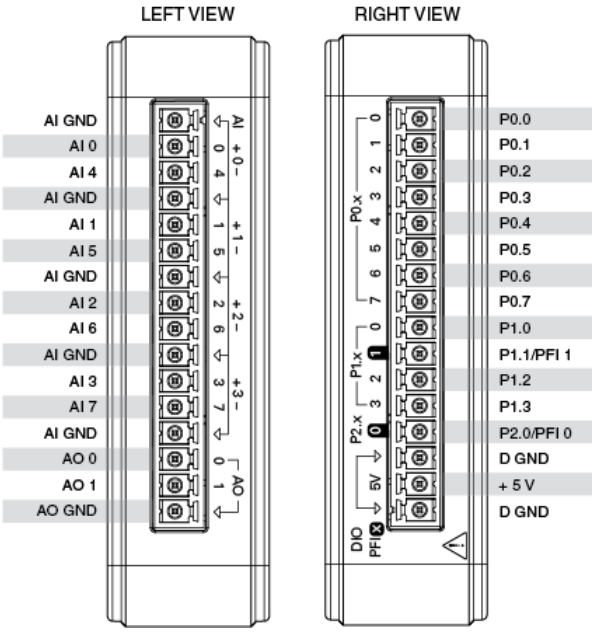
CD příloha

I. Testovaci_aplikace.vi
II. SubVI

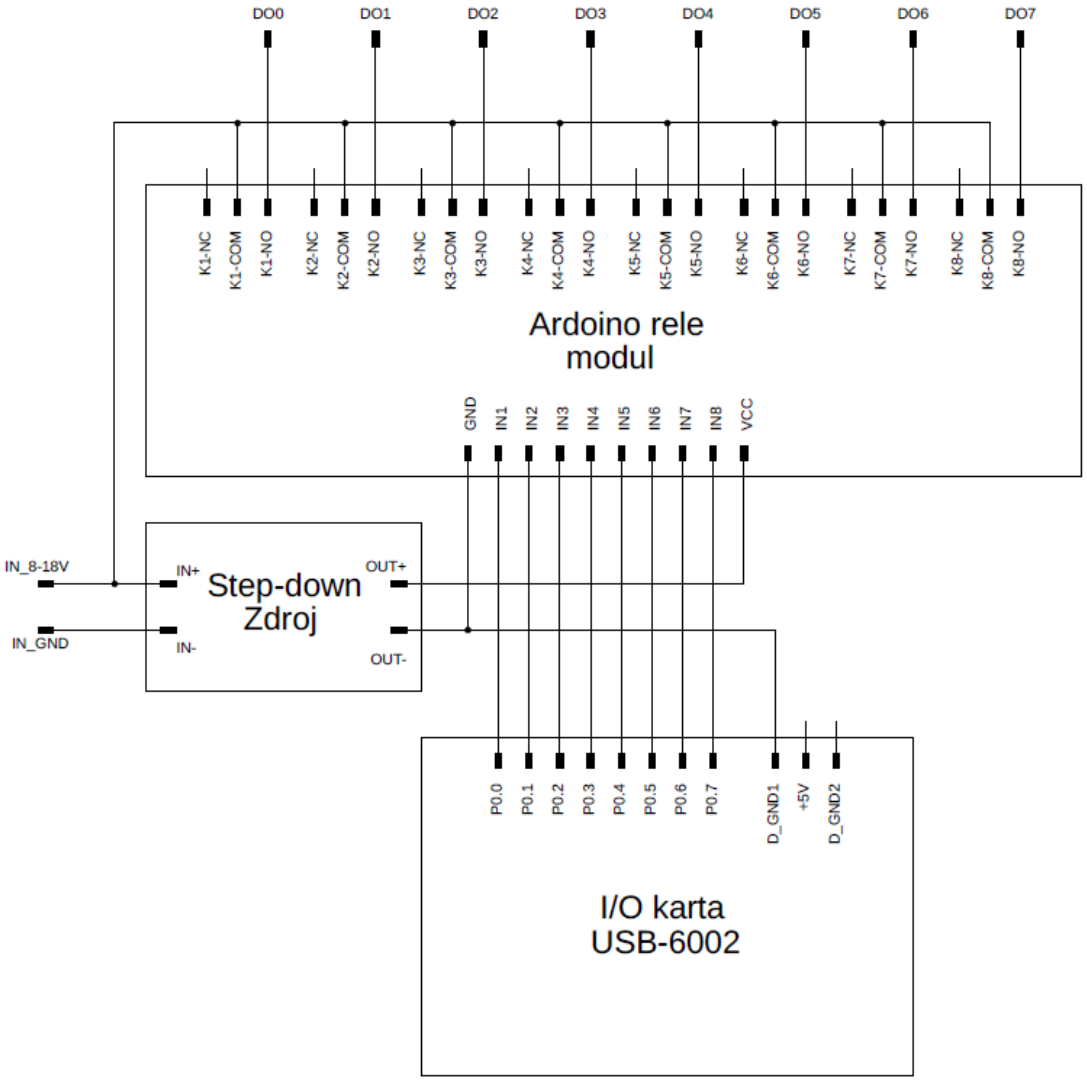
Příloha A: *Rozložení pinů LIN karty NI-9866 [30]*

Connector	Pin	Signal
	1	No Connection (NC)
	2	NC
	3	COM
	4	NC
	5	SHLD
	6	COM
	7	LIN
	8	NC
	9	V_{SUP}

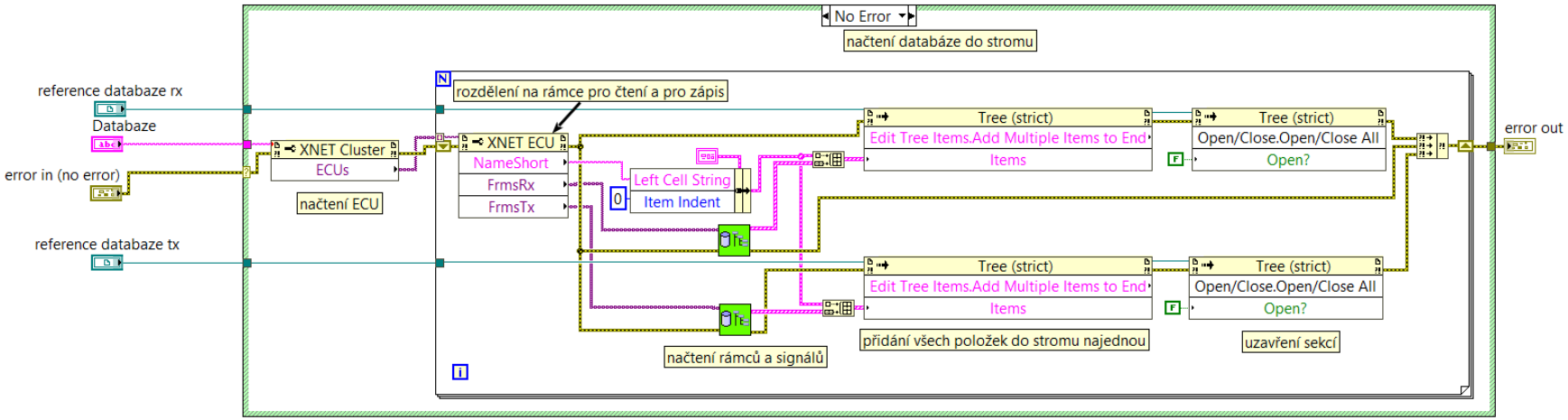
Příloha B: *I/O karta USB-6002 [40]*



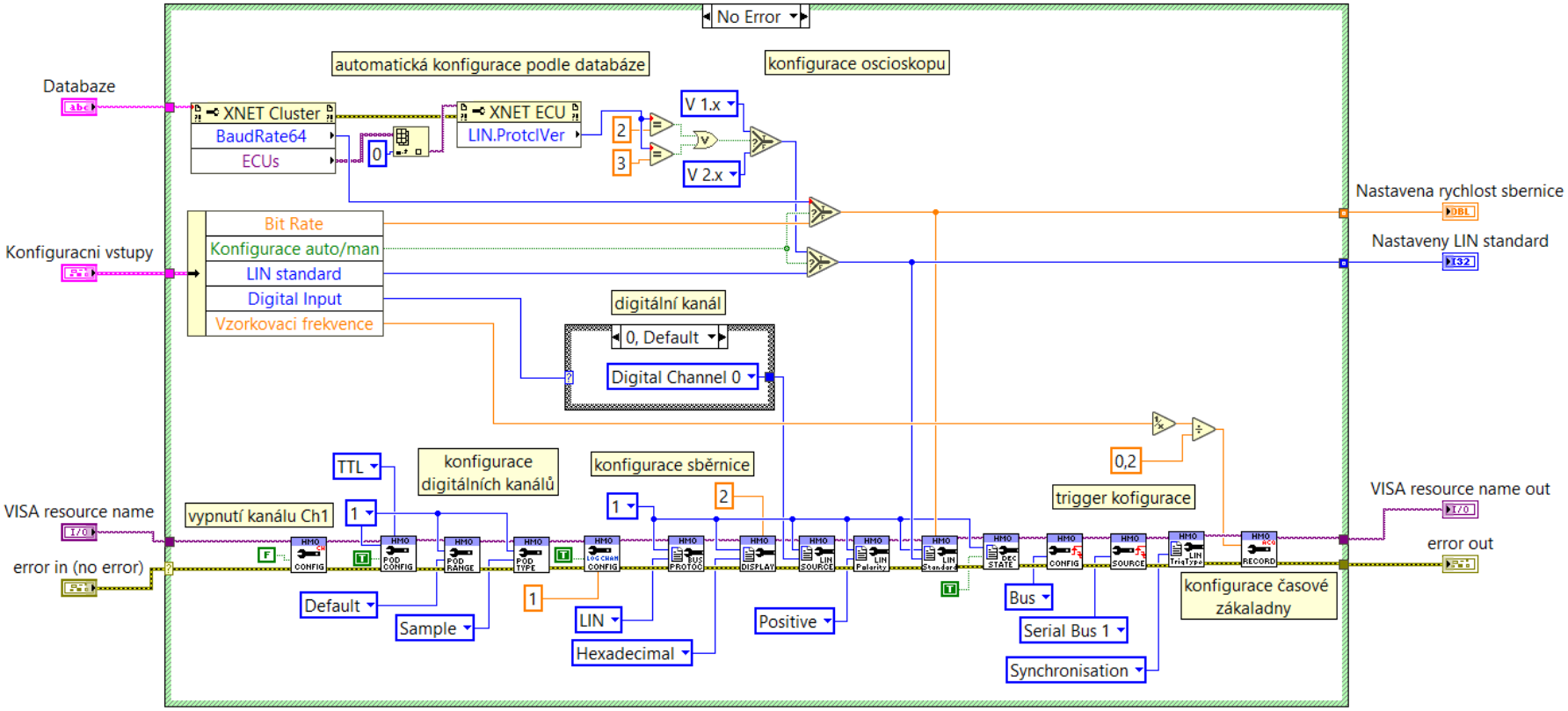
Příloha C: *Schéma zapojení pro I/O kartu a dalšího HW pro spínání analogových funkcí*



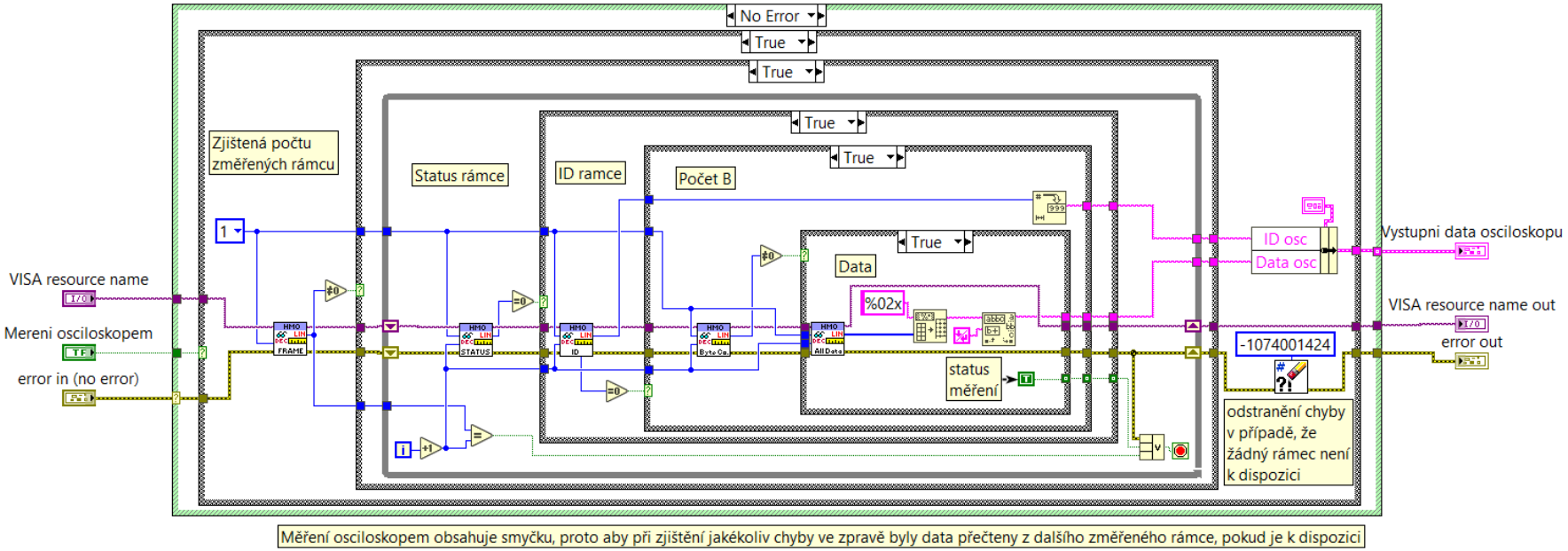
Příloha D: *Zdrojový kód pro subVI načtení položek databáze do stromu*



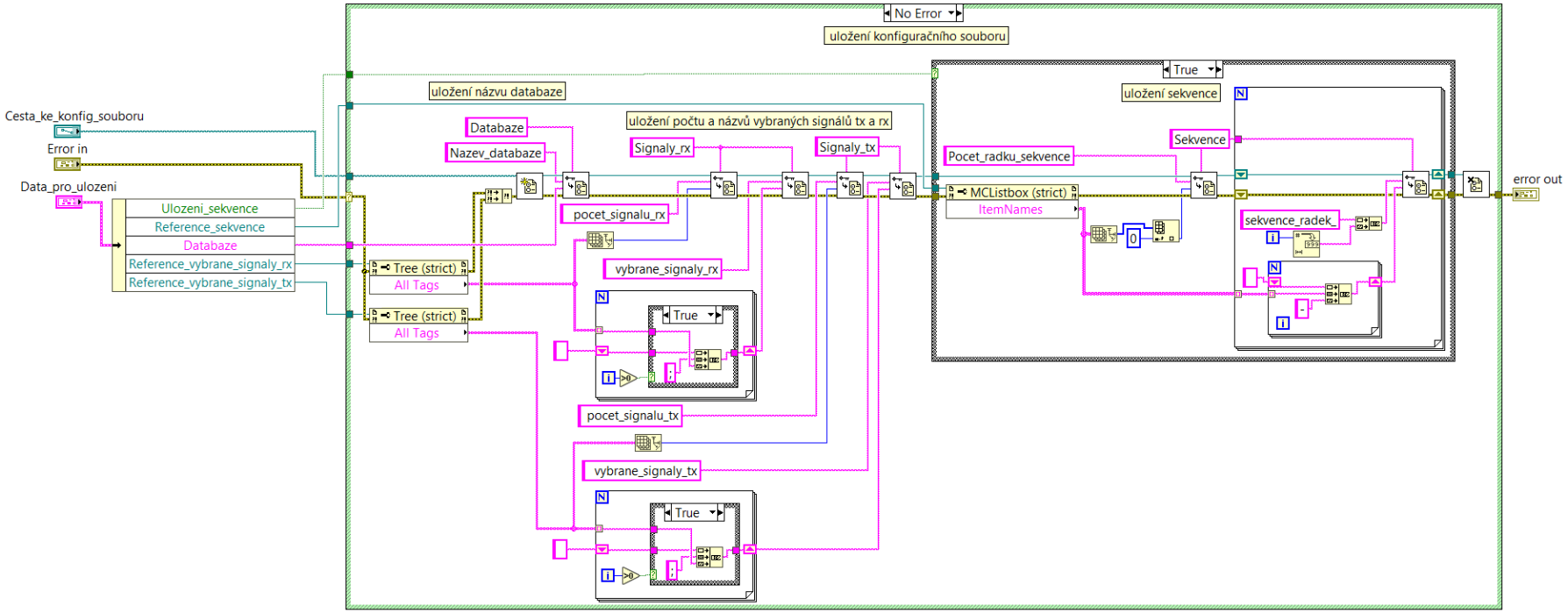
Příloha E: *Zdrojový kód pro subVI konfigurace osciloskopu*



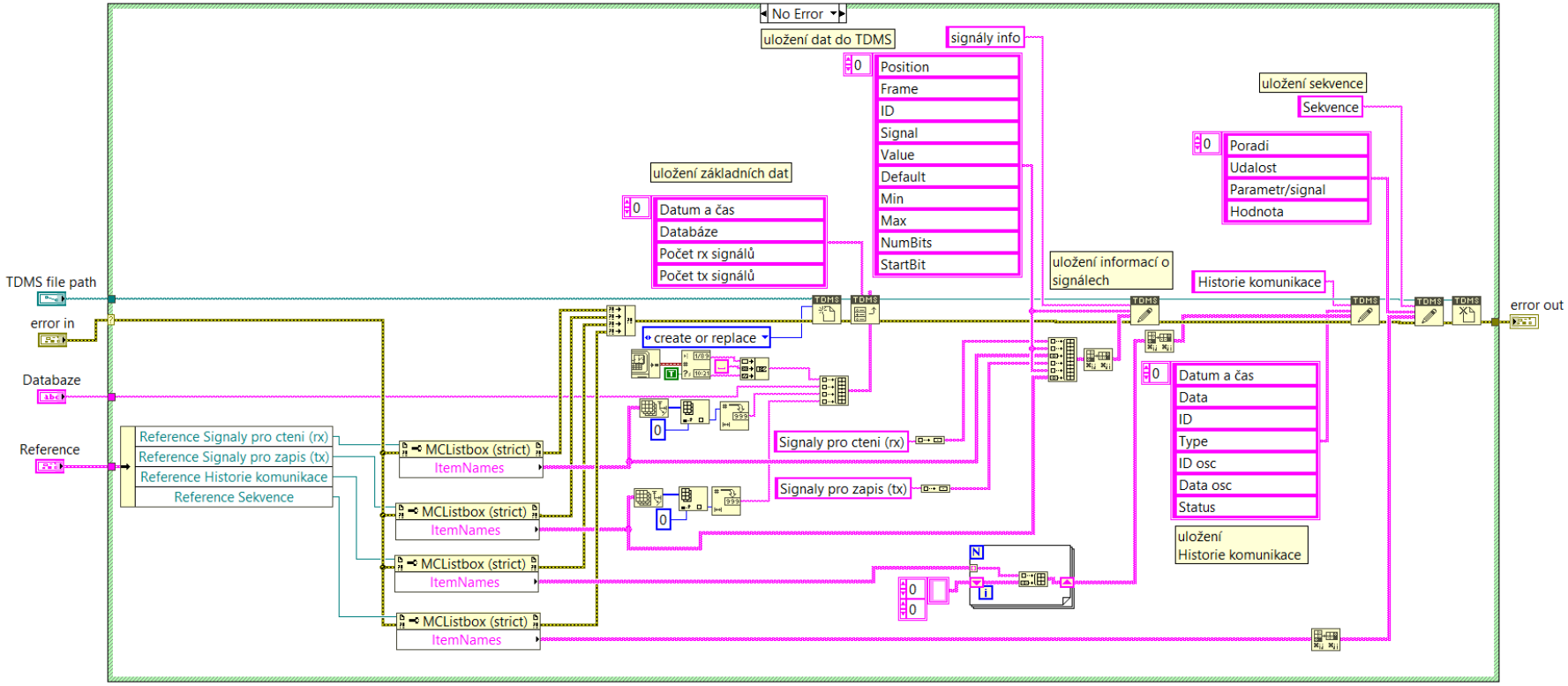
Příloha F: *Zdrojový kód pro subVI Měření osciloskopem*



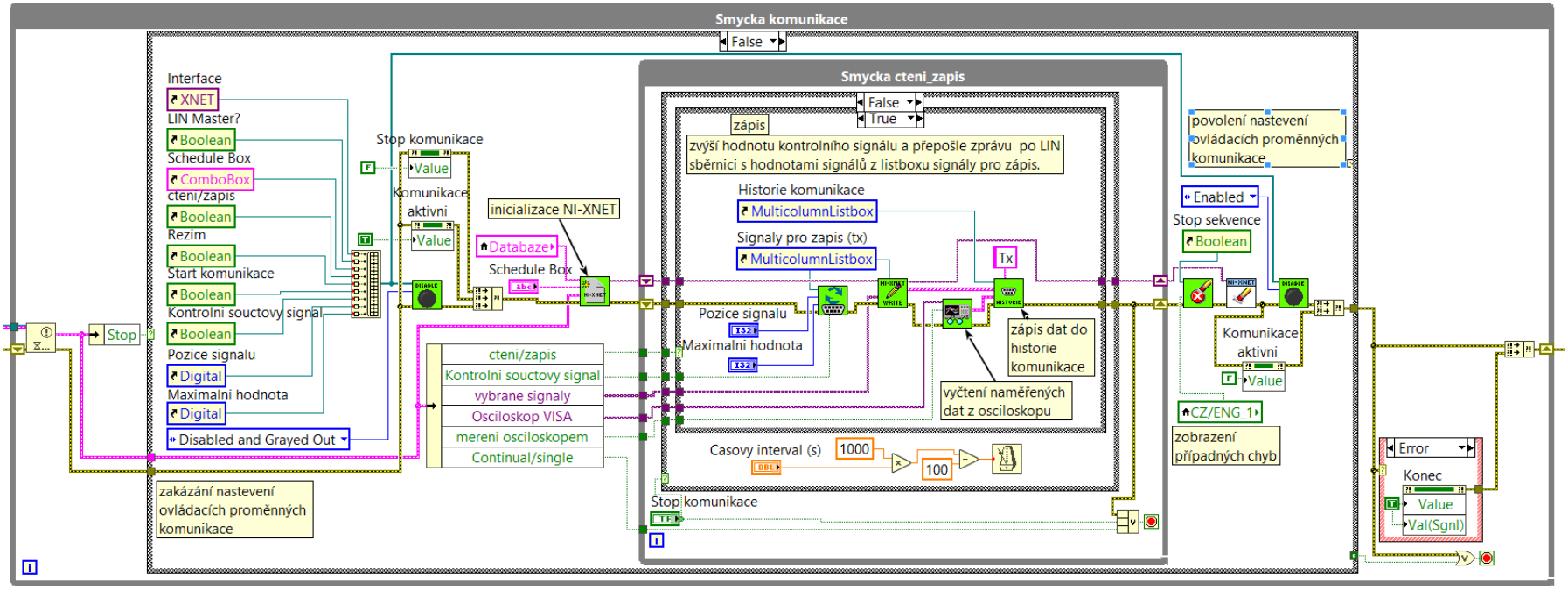
Příloha G: *Zdrojový kód pro subVI uložení konfiguračního souboru*



Příloha H: *Zdrojový kód pro subVI uložení dat do TDMS souboru*



Příloha I: *Zdrojový kód pro subVI Smyčku komunikace*



Příloha J: *Zdrojový kód pro subVI pro Smyčku sekvence*