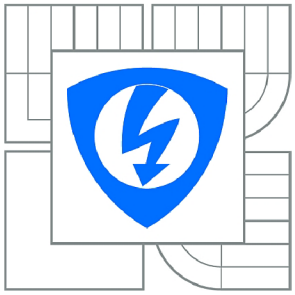
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV RADIOELEKTRONIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF RADIO ELECTRONICS

ŘÍDÍCÍ A DIAGNOSTICKÝ SYSTÉM SBĚRNICE I2C

CONTROL AND DIAGNOSTIC SYSTEM FOR I2C BUS

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

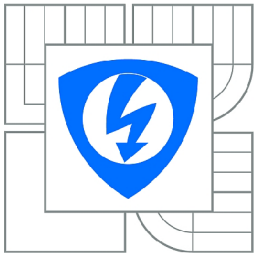
AUTOR PRÁCE — Bc. MILOŠ JUHÁS

AUTHOR

VEDOUCÍ PRÁCE — Ing. TOMÁŠ URBANEC, Ph.D.

SUPERVISOR

BRNO 2011



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav radioelektroniky

Diplomová práce

magisterský navazující studijní obor
Elektronika a sdělovací technika

Student:	Bc. Miloš Juhás	***ID:***	98171
Ročník:	2	***Akademický rok:***	2010/2011

NÁZEV TÉMATU:

Řídící a diagnostický systém sběrnice I2C

POKYNY PRO VYPRACOVÁNÍ:

Seznamte se se specifikací sběrnice I2C. Navrhněte schéma jednoduchého převodníku I2C - USB s možností funkce master, slave i pasivního sledování provozu.

Vytvořte desku plošných spojů a převodník realizujte. Pro převodník vytvořte potřebný firmware.

Pro řídící počítač vytvořte diagnostický software, který bude přehledně zobrazovat dění na sběrnici, graficky interpretovat zjištěné hodnoty jednotlivých účastníků komunikace. Software musí umožnit interaktivní nastavení a jeho uložení, záznam dat do souboru.

DOPORUČENÁ LITERATURA:

[1] NXP Semiconductors. I2C bus specification and user manual. [online], [cit. dne 26.1.2011], dostupné na http://ics.nxp.com/support/documents/interface/pdf/i2c.bus.specification.pdf

[2] Matoušek, David, USB prakticky.1. díl,S obvody FTDI, Praha, BEN - technická literatura, 2003. 1. vyd. 270 s. ISBN 80-7300-103-9

Termín zadání:	7.2.2011	***Termín odevzdání:***	20.5.2011

Vedoucí práce: Ing. Tomáš Urbanec, Ph.D.

prof. Dr. Ing. Zbyněk Raida
Předseda oborové rady

ABSTRAKT

Cieľom diplomovej práce je návrh a realizácia analyzátora zbernice I²C. Stručne je popísaný protokol zbernice I²C ako aj spôsob pripojenia zariadení ku zbernici. Ďalej sú definované základné požiadavky na analyzátor zbernice I²C. Nasleduje návrh hardwarového prevodníka I²C/USB s popisom jednotlivých blokov. V kapitole venovanej firmwaru prevodníka je vysvetlený spôsob komunikácie s počítacom, dekódovanie prijatých príkazov a princíp fungovania režimov master, slave a pasívneho sledovania zbernice. Posledná časť je zameraná na obslužný software a jeho štruktúru. Sú popísane jednotlivé rozhrania, najdôležitejšie triedy a predvolené zásuvné moduly.

KĽÚČOVÉ SLOVÁ

.NET, analyzátor, C#, diagnostický systém, FT232RL, I²C , IIC, prevodník, TWI, USB, zbernica, zásuvný modul

ABSTRACT

The objective of this diploma thesis is design and realization of I²C bus analyzer. I²C bus protocol is briefly described, together with means of connecting devices to the bus. Next the basic requirements for I²C bus analyzer are defined. Then the design of hardware I²C-to-USB converter including block description is proposed. The chapter dedicated to converter firmware describes method of communication with PC, decoding of intercepted commands and principles of master, slave and passive mode of bus monitoring. The last part is focused on operation software and its structure. Described are individual interfaces, most notable classes and default plugin modules.

KEYWORDS

.NET, analyzer, C#, diagnostic system, FT232RL, I²C , IIC, converter, TWI, USB, plugin

JUHÁS, M. *Řídící a diagnostický systém sběrnice I2C.* Brno:Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2011. 40 s., 10 s. príloh. Vedoucí diplomové práce Ing. Tomáš Urbanec, Ph.D.

PROHLÁŠENÍ

Jako autor diplomové práce na téma Řídící a diagnostický systém sběrnice I2C dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

V Brně dne …… ……
(podpis autora)

POĎAKOVANIE

Ďakujem vedúcemu diplomové práce Ing. Tomášovi Urbancovi, PhD za účinnú metodickú, pedagogickú a odbornú pomoc a ďalšie cenné rady pri spracovaní môjho semestrálního projektu.

V Brně dne …… ……
(podpis autora)

OBSAH

ZOZNAM OBRÁZKOV

ZOZNAM TABULIEK

1 ÚVOD

S rozvojom digitálnych integrovaných obvodov vznikla potreba vzájomnej komunikácie mikroprocesora s pripojenými perifériami. K tomuto účelu je výhodné využiť sériové zbernice. Či už je to I²C (Inter-Integrated Circuit), SPI (Serial Peripherial Interface), LIN (Local Interconnect Network) alebo CAN (Controller-Area Network), ktoré sa vzájomne odlišujú počtom vodičov, zložitosťou komunikačného protokolu, spoľahlivosťou, maximálnou dĺžkou vodičov alebo rýchlosťou, ich spoločnou vlastnosťou je, že šetria počet vodičov a tým nie len zjednodušujú návrh zariadenia ale aj podstatne znižujú výrobné náklady.

Analyzátor zbernice I²C je možné využiť predovšetkým pri vývoji a ladení vlastných zariadení pracujúcich so zbernicou I²C. Na trhu je dostupných množstvo rôznych analyzátorov, ktorých najväčšou nevýhodou je vysoká cena. Tá je v niektorých prípadoch tak vysoká, že by za ňu bolo možné kúpiť logický analyzátor alebo dokonca osciloskop.

Cieľom diplomovej práce je navrhnúť jednoduchý prevodník zbernice I²C na zbernicu USB (Universal Serial Bus). Prevodník bude schopný pracovať ako zariadenia I²C master, I²C slave alebo môže pasívne sledovať komunikáciu na zbernici. Bude podporovať režimy zbernice Normal mode a Fast mode a pracovať tak so zbernicami do 400 kHz. Aby mohol komunikovať s čo najväčším počtom zariadení, je dôležité umožniť mu prácu s dnes najpoužívanejšími napäťovými úrovňami 3,3 V a 5 V. Vzhľadom na to, že sa jedná o zariadenie, ktoré sa pripája k externým zberniciam, musí byť schopné ochrániť počítač proti poškodeniu v prípade poruchy na analyzovanej zbernici.

V prvej časti práce je na základe špecifikácie stručne popísaná zbernica I²C. Tento popis zahŕňa definovanie základných pojmov, spôsob pripojenia zariadení k zbernici a komunikačný protokol. Druhá časť práce je venovaná návrhu samotného analyzátora. Ten začína definovaním požiadaviek na prevodník a pokračuje návrhom hardwaru prevodníka zbernice I²C na USB. V tejto časti je stručne popísané a zdôvodnené riešenie jednotlivých blokov a výber kľúčových komponentov. Nasleduje popis firmwaru a komunikačného protokolu medzi mikroprocesorom a počítačom. Zároveň je tu popísaný spôsob dekódovania inštrukcií a jednotlivé pracovné režimy. Nasleduje stručný návod na nastavenie obvodu FT232RL a potrebnú úpravu ovládačov. Na záver je popísaný obslužný software pre počítač, predovšetkým základný koncept, rozhrania a najdôležitejšie triedy.

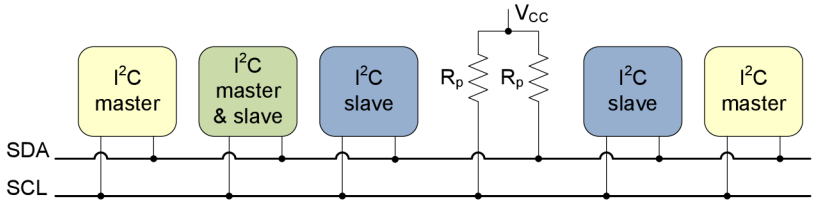
2 Zbernica I²C

Zbernica I²C je dvojvodičová zbernica typu master-slave, vyvinutá firmou Philips Semiconductors v osemdesiatych rokoch minulého storočia. Primárne bola určená na komunikáciu mikroprocesora v televíznych prijímačoch s perifériami umiestnenými na plošnom spoji. Mala tak ušetriť množstvo dátových a adresných vodičov, adresných dekodérov apod. Menší počet vodičov a súčiastok logicky vedie k nižším výrobným nákladom a zároveň vyššej odolnosti voči elektromagnetickému rušeniu ako aj elektrostatickému výboju. Dnes možno zbernicu I²C považovať za štandard, ktorý sa používa v mnohých zariadeniach. Podporujú ju popredný výrobcovia integrovaných obvodov a mikroprocesorov, používa sa na komunikáciu s displaymi, senzormi a mnohými ďalšími súčiastkami. Zbernicu samotnú je možné voľne využívať avšak na slave adresy zariadení, ktoré spravuje spoločnosť NXP Semiconductor, je potrebné zakúpiť licencie výrobcom obvodu. Aj z tohto dôvodu používajú niektorý výrobcovia zbernice označované ako TWI (Two Wire Interface) alebo TWSI (Two Wire Serial Interface), ktoré sú kompatibilné so zbernicou I²C. [1], [2]

2.1 Pripojenie zariadení k zbernici I²C

Samotná zbernica I²C pozostáva z dvoch vodičov. Dátový vodič označený SDA (Serial Data) slúži na poloduplexný prenos dát, ktorých platnosť je daná hodinovým signálom, prenášaným vodičom SCL (Serial Clock). Všetky zariadenia pripojené ku zbernici musia mať vstupno-výstupný port s otvoreným kolektorom. Vysoká úroveň H je potom určená zdvíhacím rezistorom R_p a nízka úroveň L pripojením vodiča na zem jedným zo zariadení. Žiadne z pripojených zariadení nesmie nastaviť úroveň H na zbernici nastavením vysokej úrovne na výstupnom porte. Napäťové úrovne sú definované ako $0{,}3 \cdot V_{CC}$ pre úroveň L a $0{,}7 \cdot V_{CC}$ pre úroveň H.

Ku zbernici môžu byť pripojené dva typy zariadení, zariadenie typu master a zariadenie typu slave (obr. 2.1). Zariadenia typu master a slave môžu byť tiež zlúčené do jedného púzdra a využívať spoločné vstupno-výstupné porty. Zároveň je potrebné rozlišovať zbernice s jedným zariadením typu master, tzv. single-master zbernica a zbernice s viacerými pripojenými zariadeniami typu master, tzv. multi-master zbernica. K multi-master zbernici nesmie byť pripojené zariadenie, ktoré nepodporuje dodatočné rozhodovacie funkcie. Tie pre single-master zbernice nie sú potrebné, ani vyžadované špecifikáciou I²C, [2].



Obr. 2.1: Zbernica I²C

Rýchlosť komunikácie na zbernici je daná režimom, pre ktorý je daná zbernica navrhnutá. Tento režim určuje maximálnu rýchlosť, ktorú sú zariadenia deklarujúce podporu daného režimu schopné dosiahnuť (tab. 2.1). Minimálna rýchlosť nie je stanovená, resp. sa uvádza 0 Hz a všetky zariadenia pracujúce v rýchlejšom režime musia byť spätne kompatibilné s pomalšími zariadeniami. So zvyšujúcou sa rýchlosťou samozrejme stúpajú nie len nároky kladené na vstupno-výstupné porty zariadenia ale aj požiadavky na zbernicu samotnú, ako sú časové parametre signálov, kapacita zbernice apod.

Počet zariadení slave, ktoré je možné ku zbernici pripojiť je daný počtom adresovateľných zariadení a celkovou kapacitou zbernice. Pre 7bitovú adresu je to 128 zariadení a pre 10bitovú adresu 1024 zariadení. Od tohto počtu je ešte potrebné odpočítať rezervované adresy, ktoré nemôžu byť priradené zariadeniu slave, pretože majú svoje špecifické určenie.

Tab. 2.1: Režimy zbernice I²C

Režim	Maximálna bitová rýchlosť [kb/s]	Maximálna kapacita na vodič zbernice [pF]
Standard-mode	100	400
Fast-mode	400	400
Fast-mode Plus	1000	550
High-speed mode	3400	100

2.2 Protokol zbernice I²C

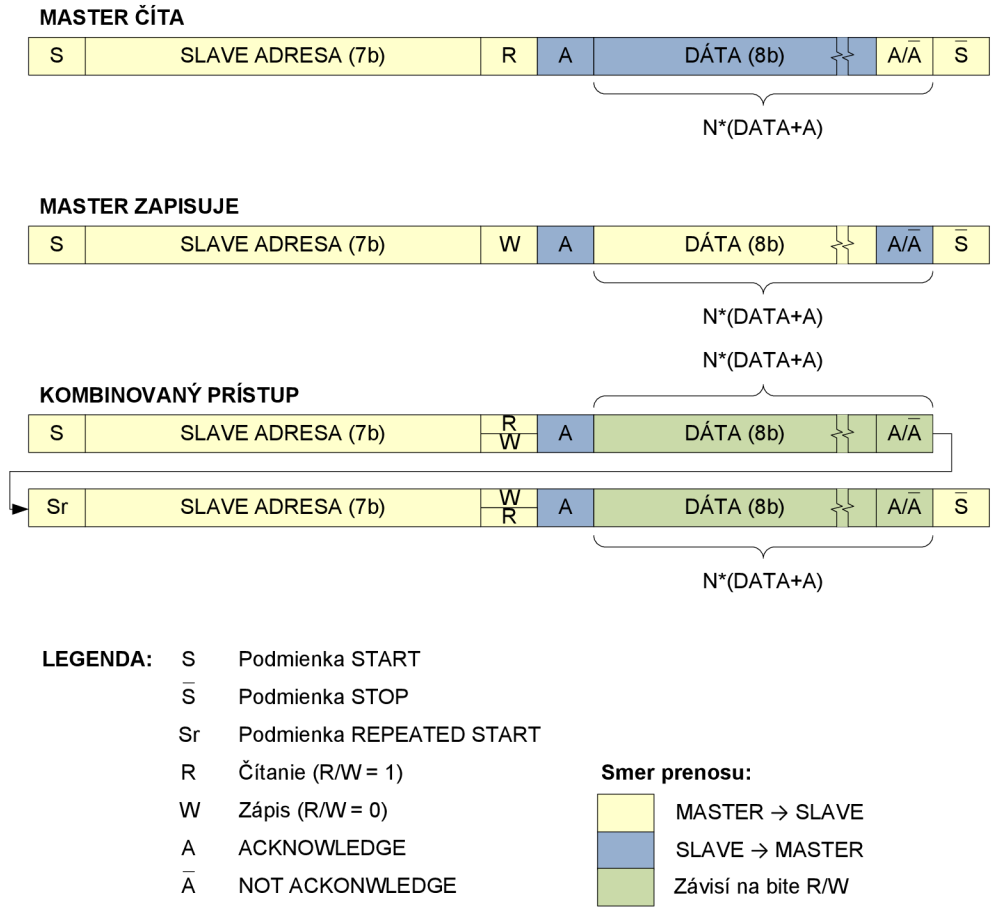
Komunikačný protokol zbernice I²C pozostáva z viacerých funkcií. Ich implementácia môže byť povinná alebo voliteľná podľa typu zariadenia (tab. 2.2). Komunikáciu na zbernici riadi vždy zariadenie typu master, ktoré zároveň generuje hodinový signál SCL. Dáta na zbernici (úroveň na linke SDA) je možné meniť len pri nízkej úrovni hodinového signálu SCL. Čítať platné dáta zo zbernice je naopak možné pri vysokej úrovni hodinového signálu SCL, kedy sa úroveň SDA nesmie meniť. Výnimku tvorí iba generovanie podmienok START a STOP.

Tab. 2.2: Prehľad funkcií zbernice I²C

	Single-master	Multi-master	Slave
Podmienka START	povinné	povinné	povinné
Podmienka STOP	povinné	povinné	povinné
ACK/NACK	povinné	povinné	povinné
Synchronizácia hodín	-	povinné	-
Arbitráž	-	Povinné	-
Podržanie hodín	voliteľné	voliteľné	voliteľné
7b slave adresa	povinné	povinné	povinné
10b slave adresa	voliteľné	voliteľné	voliteľné
General call	voliteľné	voliteľné	voliteľné
Software reset	voliteľné	voliteľné	voliteľné
START byte	-	voliteľné	-
ID zariadenia	-	-	voliteľné

Podľa smeru komunikácie z pohľadu zariadenia master je možné rozlišovať jej tri základné typy (obr. 2.2):

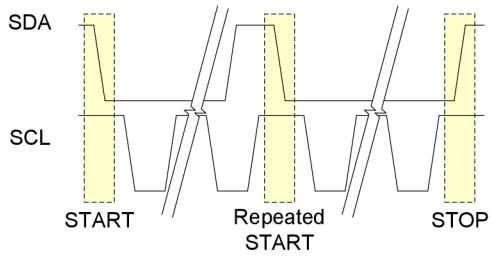
- čítanie – bit R/W je nastavený na logickú 1, master číta dáta zo zariadenia slave,
- zápis – bit R/W je nastavená na logickú 0, master zapisuje dáta do zariadenia slave,
- kombinovaný prístup – bit R/W je natavený na logickú 1 alebo 0 a za podmienkou repeated START nasleduje rovnaká adresa s negovaným bitom R/W, napr. čítanie z EEPROM, kedy je pred samotným čítaním potrebné najskôr zapísať adresu miesta v pamäti.



Obr. 2.2: Dátový rámec zbernice I²C – typy prenosu

2.2.1 Podmienky START, STOP a repeated START

Podmienky START a STOP sú generované zariadením typu master a určujú začiatok a koniec prenosu. Sú zvláštnym a jediným prípadom, kedy je možné meniť úroveň vodiča SDA pri úrovní H hodinového signálu SCL. Podmienka START je generovaná zostupnou hranou (zmena z H do L) na vodiči SDA, pri SCL = H a podmienka STOP naopak vzostupnou hranou (zmena z L do H) na vodiči SDA, pri SCL = H. Pokiaľ je medzi podmienkami START a STOP opätovne vyslaná podmienka START, jedná sa o tzv. repeated (opakovaný) START. Repeated START je možné využiť v prípade, kedy chce master komunikovať s iným zariadením a nechce uvoľniť zbernicu alebo s tým istým zariadením ale opačným smerom (čítanie/zápis).



Obr. 2.3: Podmienky START, STOP a repeated START

2.2.2 Potvrdzovanie – acknowledge, not ackonwledge

Každý byte (8 dátových bitov) poslaný na zbernicu, či už zariadením master alebo slave je nasledovaný potvrdzovacím bitom ACK (acknowledge) alebo NACK (not acknowledge).

Počas deviatej hrany hodinového signálu zariadenie, ktoré vysiela, uvoľní linku SDA a príjemca potvrdí prijatie bytu bitom ACK (úroveň L na linke SDA). V prípade, že tak príjemca neurobí, je na linke SDA vysoká úroveň, teda bit NACK, na čo môže zariadenie master reagovať ukončením komunikácie (vyslaním podmienky STOP) alebo zopakovaním operácie (podmienkou repeated START). Špecifikácia zbernice I^2C definuje päť prípadov, ktoré vedú ku generovaniu NACK:

- zariadenie s požadovanou adresou neexistuje,
- prijímač je zaneprázdnený a nemôže tak prijímať alebo vysielať dáta,
- prijímač prijal neplatné dáta,
- prijímač nemôže prijať ďalšie dáta,
- ak prijíma zariadenie typu master a nevyšle ACK, je to signál pre vysielajúce zariadenie slave, že má ukončiť komunikáciu (bol prijatý posledný byte) a master následne vygeneruje podmienku STOP.

2.2.3 Adresy zariadení

2.2.3.1 7bitová adresa

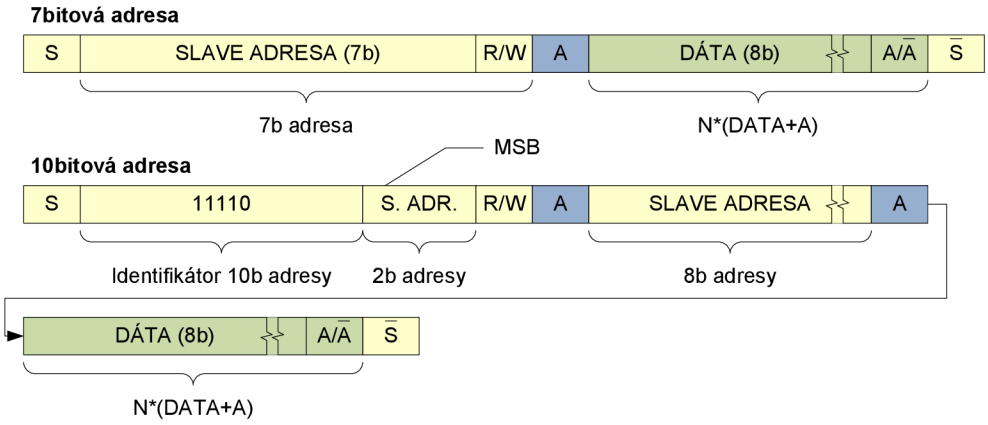
Ako už bolo naznačené vyššie, zbernica I^2C prenáša dáta po bytoch. Adresu zariadenia slave vysiela master hneď za podmienkou START a je vložená do ôsmych bitov. Prvých sedem bitov od MSB smerom k LSB je 7bitová adresa a najnižší bit určuje typ operácie (R/W). V prípade, že je bit R/W nastavený do log. 1, bude master čítať dáta zo zariadenia slave, ak je bit R/W v log. 0, master bude do zariadenia slave dáta zapisovať. Zariadenie slave, ktoré svoju adresu na zbernici deteguje ju potvrdí bitom ACK. Ten je pre zariadenie master signálom, že adresované zariadenie existuje a je pripravené na komunikáciu. Adresa pozostávajúca so siedmych bitov umožňuje, po odpočítaní rezervovaných adries (tab. 2.3), adresovať $2^7 - 16 = 112$ zariadení, čo je pre jednoduchšie aplikácie plne postačujúce. Problém nastáva napr. keď má byť v jednom zariadení použitých viacero rovnakých obvodov. Tie majú zvyčajne konfigurovateľné 3 bity, z čoho vyplýva, že maximálny počet takýchto obvodov na jednej zbernici je 8.

Tab. 2.3: Rezervované I²C slave adresy

Slave adresa	R/W bit	Popis
0000 000	0	General call
0000 000	1	START byte
0000 001	X	CBUS adresa
0000 010	X	Rezervované pre I²C kompatibilné zbernice
0000 011	X	Rezervované pre budúce použitie
0000 1XX	X	HS-mode master kód
1111 1XX	X	Rezervované pre budúce použitie
1111 0XX	X	10bitové adresovanie

2.2.3.2 10bitová adresa

Počet integrovaných obvodov s podporou zbernice I²C neustále rastie a preto bola dodatočne definovaná 10bitová adresa, ktorá umožňuje adresovať $2^{10} - 16 = 1008$ zariadení. Jej základom je 7bitová rezervovaná adresa 11110XX, ktorá zabezpečuje plnú kompatibilitu oboch typov adries. Tie môžu bez problémov pracovať na jednej zbernici, pretože sa najskôr odvysiela 7bitová adresa, ktorú bitom ACK potvrdia všetky zariadenia s 10bitovým adresovaním a zhodnými bitmi XX. Následne master odvysiela ďalší byte obsahujúci zvyšných 8 bitov adresy, ktorú potvrdí iba adresované zariadenie. Dátový rámec so 7bitovou aj 10bitovou adresou je znázornený na obr. 2.4. Špecifický je prípad použitia kombinovaného režimu, kedy sa za podmienkou repeated START odvysiela iba prvý byte adresy, teda 11110XX s bitom R/W na konci. Zariadenia so 7bitovým adresovaním túto adresu neidentifikujú ako svoju a budú ju preto ignorovať. Zariadenie slave s 10bitovým adresovaním si pamätá, že už bolo predtým adresované a vyšle bit ACK. Zvyšných 8 bitov adresy sa nevysiela, po ACK už nasledujú dáta.

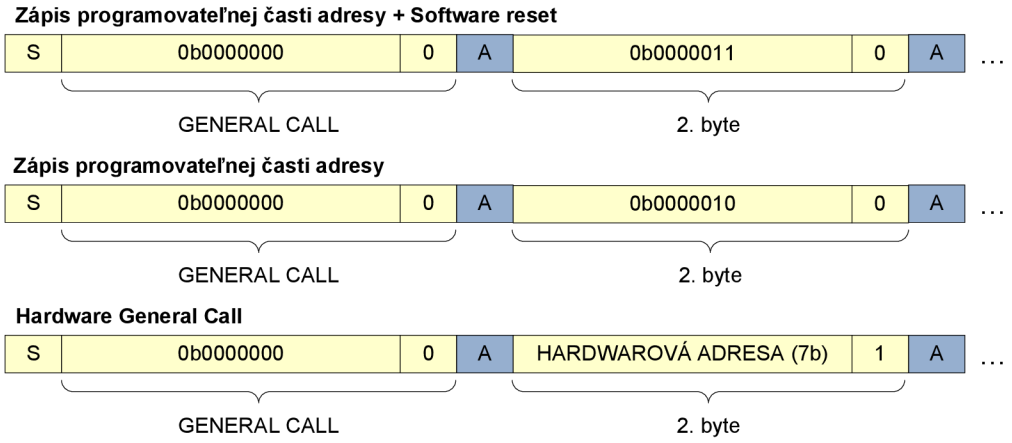


Obr. 2.4: I²C rámec so 7bitovou a 10bitovou adresou

2.2.3.3 General call

General call (všeobecné volanie) slúži na hromadné adresovanie viacerých zariadení. Master vyšle po podmienke START adresu 0b0000000 a R/W bit = 0. Všetky zariadenia slave, ktoré podporujú general call na toto volanie odpovedajú bitom ACK. Následne master vyšle ďalší byte obsahujúci príkaz (obr. 2.5). Zaujímavý je posledný z uvedených príkazov, tzv. hardware general call. Používa sa v prípade, že je zariadenie hardwarový master a nepozná adresy pripojených

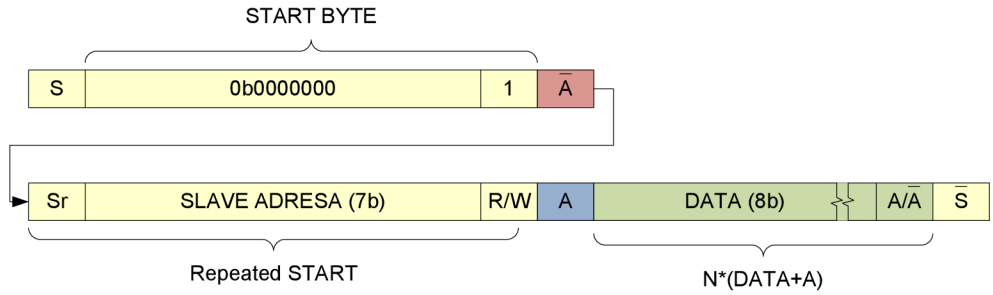
zariadení slave. Takýto master preto použije volanie adresy general call a v prípade odpovede ACK vyšle druhý byte, ktorým je jeho vlastná adresa. Inteligentné zariadenia slave na zbernici rozpoznajú toto volanie a prijmú dáta posielané hardwarovým masterom, pokiaľ sú im určené.



Obr. 2.5: Príkazy podporované general call

2.2.3.4 START byte

START byte je určený pre mikroprocesory bez hardwarovej podpory zbernice I²C. Vysiela sa ako adresa 0b0000000 s bitom R/W = 1, nasledovaná bitom NACK a podmienkou repeated START (obr. 2.6). Zariadenie tak môže vzorkovať zbernicu s nižším vzorkovacím kmitočtom a až keď deteguje jednu zo siedmych núl prepne sa na vyšší kmitočet a na základe podmienky repeated START sa zosynchronizuje. Žiadne zo zariadení na zbernici nesmie potvrdiť START byte bitom ACK.



Obr. 2.6: START byte

2.2.4 Clock stretching

Pomalšie zariadenie slave má možnosť spomaliť alebo pozastaviť prenos. Spomalenie spočíva v predĺžení každej periódy hodinového signálu podržaním úrovne L na linke SCL zariadením slave. Spomalenie nie je možné využívať v HS-mode. Pozastavenie funguje na podobnom princípe, ale úroveň L na linke SCL je predĺžená len na konci bytu, za bitom ACK. Zariadenie master tak čaká na uvoľnenie linky a zariadenie slave získa čas potrebný na spracovanie prijatých, poprípade vysielaných dát.

2.2.5 Multi-master zariadenia

V prípade, že je k zbernici pripojené iba jedno zariadenie typu master, je komunikácia jednoduchá, pretože ju tento master vždy riadi sám a nemôže tak dôjsť k žiadnemu konfliktu. Ak je ale zariadení master viac, je potrebné určiť, ktorý master môže v danom okamihu vysielať. Preto sú súčasťou špecifikácie funkcie na podporu multi-master zbernice. Jedná sa o synchronizáciu hodín a arbitráž. K multi-master zbernici nemôžu byť za žiadnych okolností pripojené zariadenia master, ktoré tieto funkcie nemajú implementované.

2.2.5.1 Synchronizácia hodín

Synchronizácia hodín je vzhľadom na spôsob pripojenia zariadení k zbernici zhodná s logickou funkciou AND. Ak aspoň jeden master nastaví úroveň L na linke SCL, výsledná úroveň je vždy L. Všetky zariadenia master musia zároveň monitorovať linku SCL a riadiť sa hodnotou skutočne nastavenou na zbernici. Ak master nastaví úroveň H uvoľnením linky SCL skôr, musí počkať na jej uvoľnenie všetkými zariadeniami master a až potom začne počítať dobu trvania úrovne H.

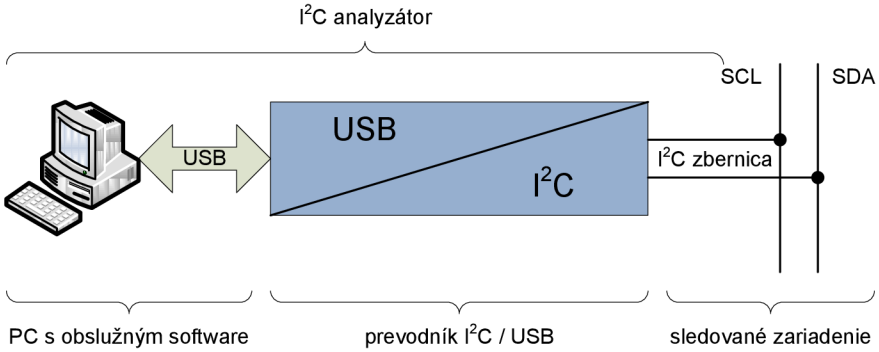
2.2.5.2 Arbitráž

Zatiaľ čo konflikty na linke SCL rieši synchronizácia hodín, na linke SDA sa používa tzv. arbitráž. Všetky zariadenia master musia monitorovať stav linky SDA a porovnávať ju s nastavenou hodnotou. Pokiaľ sa nastavená hodnota nezhoduje so stavom linky, daný master stratil arbitráž a musí okamžite uvoľniť dátovú linku. V generovaní hodinového signálu môže pokračovať do konca bytu, v ktorom stratil arbitráž, potom musí uvoľniť linku SCL a operáciu zopakovať, keď bude zbernica voľná. V praxi to znamená, že vyhráva vždy master, ktorý nastavil úroveň L na dátovej linke. Pokiaľ zariadenie master pracuje zároveň ako slave, musí sa ihneď prepnúť do tohto režimu pre prípad, že by ho víťazný master adresoval. Ak viaceré zariadenia posielajú identické dáta, môžu všetky dokončiť operáciu bez straty arbitráže. Výsledok arbitráže nie je možné dopredu určiť v nasledujúcich prípadoch:

- jedno zo zariadení master vygeneruje podmienku repeated START a druhé posiela dáta,
- jedno zo zariadení master vygeneruje podmienku STOP a druhé posiela dáta,
- jedno zo zariadení master vygeneruje podmienku repeated START a druhé podmienku STOP.

3 Analyzátor zbernice I²C

Cieľom práce je navrhnúť analyzátor zbernice I²C. Ako vidno z obr. 3.1, jadrom analyzátora je prevodník zbernice I²C/USB, ktorý preposiela dáta zo sledovaného zariadenia do počítača a späť. Samotná analýza dát už prebieha v počítači pomocou obslužného softwaru.



Obr. 3.1: Bloková schéma analyzátora zbernice I²C

Každé zariadenie musí spĺňať isté požiadavky, ktoré by mali byť definované ešte pred samotným návrhom. Z tohto dôvodu boli vopred definované základné parametre, ktoré musí prevodník spĺňať (tab. 3.1).

Tab. 3.1: Požadované parametre prevodníka I²C/USB

Napájanie	Z USB portu počítača
Napätie na zbernici	Vcc = 3,3 V, Vcc = 5 V
Rýchlosť zbernice	Standard-mode (do 100 kHz) Fast-mode (do 400 kHz)
Režimy komunikácie	Master Slave Pasívny
Sledované veličiny	Komunikácia na úrovni správ, bez časovej analýzy signálov
Ostatné	Hardwarová jednoduchosť, použitie štandardných (ľahko dostupných) súčiastok

4 Hardware prevodníka I²C/USB

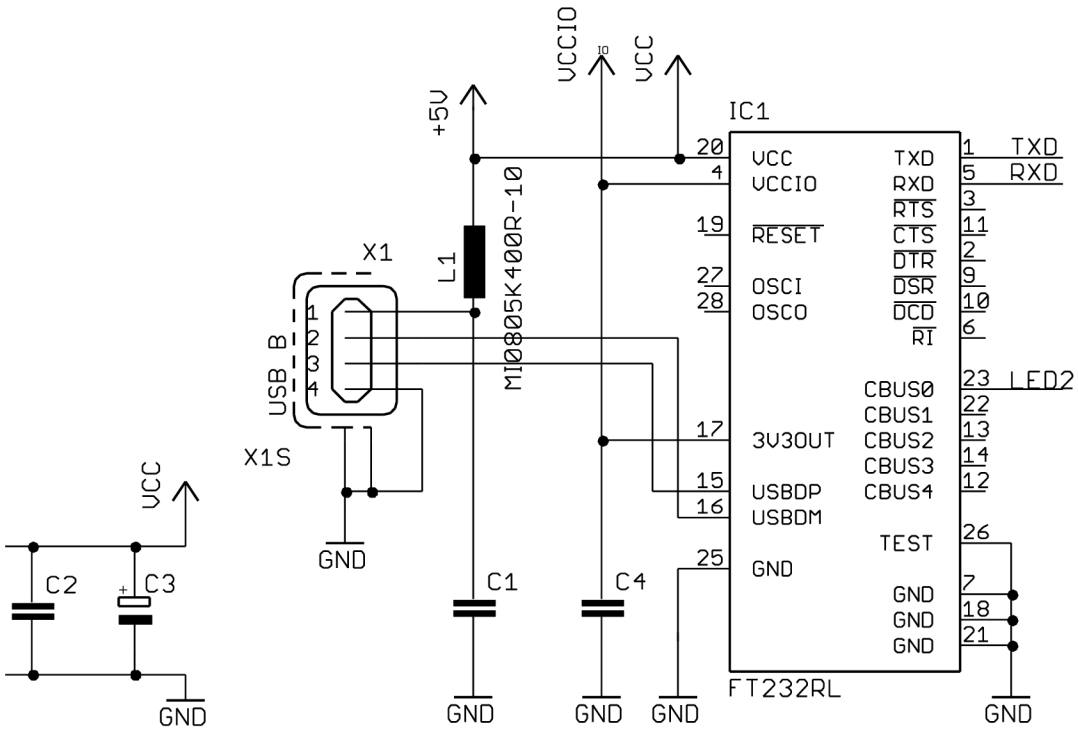
4.1 Rozhranie USB

Pripojenie zariadenia cez USB port k počítaču je jednou z kľúčových úloh, ktoré treba pri takomto návrhu vyriešiť. Rozhranie USB poskytuje viacero možností, nie všetky sú ale vhodné.

Ideálnym riešením je hardware s vlastnými ovládačmi, ktoré by mu umožňovali plne využiť jeho možnosti. Výhodou takéhoto riešenia je vysoká prenosová rýchlosť, ktorá by značne zvýšila možnosti samotného analyzátora. V prípade využitia režimu USB 2.0 hi-speed je maximálna teoretická prenosová rýchlosť až 480 Mb/s. Takto vysoká prenosová rýchlosť vedie k potrebe použiť dostatočne výkonný mikroprocesor. Ten sa značnou mierou odrazí na celkovej cene zariadenia a napísanie kvalitných ovládačov pre USB by trvalo mnohonásobne dlhšie ako návrh celého analyzátora. Toto riešenie sa k pôvodnému zámeru moc neblíži a nebude teda vhodné.

Druhou možnosťou je využitie štandardných ovládačov HID (Human Interface Device). Práca s nimi nie je príliš zložitá a nechýba ani podpora v mnohých, aj lacnejších mikroprocesoroch. Ich nevýhodou je, že sú určené pre prácu s pomalšími perifériami a spoľahlivý prevodník by pri prenosovej rýchlosti okolo 1 Mb/s bol len ťažko realizovateľný.

Poslednou variantou je využitie dostupných integrovaných obvodov na konverziu medzi USB a inou sériovou zbernicou. Dostatočný výkonom a jednoduchá implementácia viedla k využitiu prevodníka USB/UART (Universal Asynchronous Receiver Transciever), FT232RL od spoločnosti FTDI. Ten sa plne zhoduje so špecifikáciou USB 2.0 full-speed a jeho nespornou výhodou je podpora a dostupnosť ovládačov pre všetky bežne používané operačné systémy. Maximálna symbolová rýchlosť 3 Mbaudy na prenos dát z I²C zbernice postačuje a prevodník pre svoju činnosť potrebuje minimálne množstvo externých súčiastok (obr. 4.1). Obsahuje interný oscilátor s vyvedeným výstupom, ale aj pamäť eeprom na uloženie potrebných informácií ako sú napríklad USB Device ID, USB Vendor ID, nastavenia prevodníka apod. Integrované USB rezistory umožňujú pripojiť obvod priamo ku konektoru USB. Podľa katalógového listu sa doporučuje pripojenie širokopásmovej feritovej tlmivky medzi napájací pin USB konektoru a pin V_{CC} obvodu FT232RL [3]. Veľkou výhodou je tiež podpora napájania z USB portu počítača a integrovaný LDO stabilizátor napätia 3,3 V, ktorý je možné použiť na napájanie riadiaceho mikroprocesora a ďalších obvodov s celkovým odberom do 50 mA.



Obr. 4.1: Schéma zapojenia USB rozhrania prevodníka I²C/USB

4.2 Riadiaci mikroprocesor

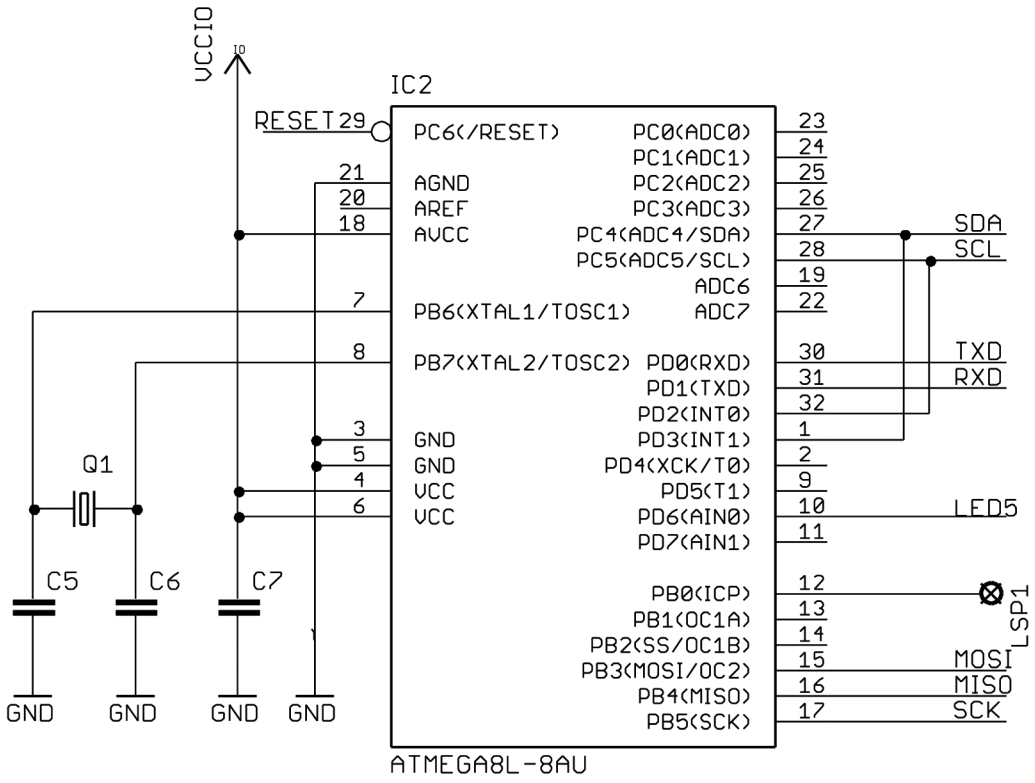
Vzhľadom na použitie prevodníka FT232RL je výhodné voliť mikroprocesor, ktorý bude mať hardwarovú podporu UART a poprípade aj zbernice I²C.

Pre prototyp prevodníka bol zvolený mikroprocesor rady AVR, konkrétne ATmega8L od firmy Atmel, [4]. Písmeno L značí, že sa jedná o nízko príkonovú variantu tohto mikroprocesora s napájacím napätím 3,3 V. Táto varianta má zároveň polovičný maximálny hodinový kmitočet, teda 8 MHz. Ten možno získať z interného oscilátora, ktorý ale nie je pre danú aplikáciu dostatočne presný a je preto potrebné použiť externý kryštál.

Mikroprocesor je vybavený jednotkou USART, ktorú je možné bez ďalších súčiastok prepojiť s prevodníkom FT232RL. Táto jednotka podporuje plne duplexný sériový asynchrónny prenos dát v dvoch režimoch. Prvým je štandardný režim so vzorkovaním prichádzajúcich dát šestnástimi vzorkami na jeden bit a maximálna symbolová rýchlosť je tak 1/16 hodinového kmitočtu. Symbolová rýchlosť je v tomto prípade rovná prenosovej rýchlosti, pretože na jeden symbol pripadá práve jeden bit. Z toho vyplýva, že maximálna prenosová rýchlosť pre hodinový kmitočet 8 MHz bude rovná 500000 b/s, čo už na prvý pohľad nepostačuje na prenesenie potrebného množstva dát v reálnom čase. To je závislé na formáte rámcov, v ktorých budú dáta prenášané a je možné ho jednoducho odhadnúť ako dvojnásobok hodinového kmitočtu na I²C zbernici. Po pripočítaní réžie prenosu cez UART ako je start bit, stop bit, parita a prípadné ďalšie zabezpečenie sa odhadované množstvo dát, ktoré je potrebné preniesť pohybuje na úrovni okolo 1 Mb/s. Druhým režimom je tzv. double-speed mode, ten

umožňuje zdvojnásobiť symbolovú rýchlosť použitím iba ôsmich vzoriek na bit, čo pre 8 MHz hodinový kmitočet predstavuje najvyššiu dosiahnuteľnú prenosovú rýchlosť 1000000 b/s. Tá by už mohla postačovať na prenesenie dát z I²C zbernice a zároveň na pridanie riadiacich informácií do prenášaných rámcov. Spoľahlivosť prenosu v tomto režime je značne závislá na presnosti oscilátora ako aj zvolenej symbolovej rýchlosti.

Na druhej strane prevodníka musí byť rozhranie I²C . Využiť možno externé prerušenia, ktoré môžu byť vyvolané hranou alebo úrovňou signálu, ale aj jednotku TWI mikroprocesora. Tá je so zbernicou I²C kompatibilná a podporuje väčšinu funkcií zo špecifikácie I²C, vrátane multi-master zbernice. TWI je definované pre zbernice s kmitočtom hodinového signálu SCL do 400 kHz. Podmienka pre taktovací kmitočet mikroprocesora, ktorý musí byť v režime slave 16x vyšší ako kmitočet hodinového signlu SCL je taktiež s rezervou splnená. Zapojenie mikroproceosra je na obr. 4.2.



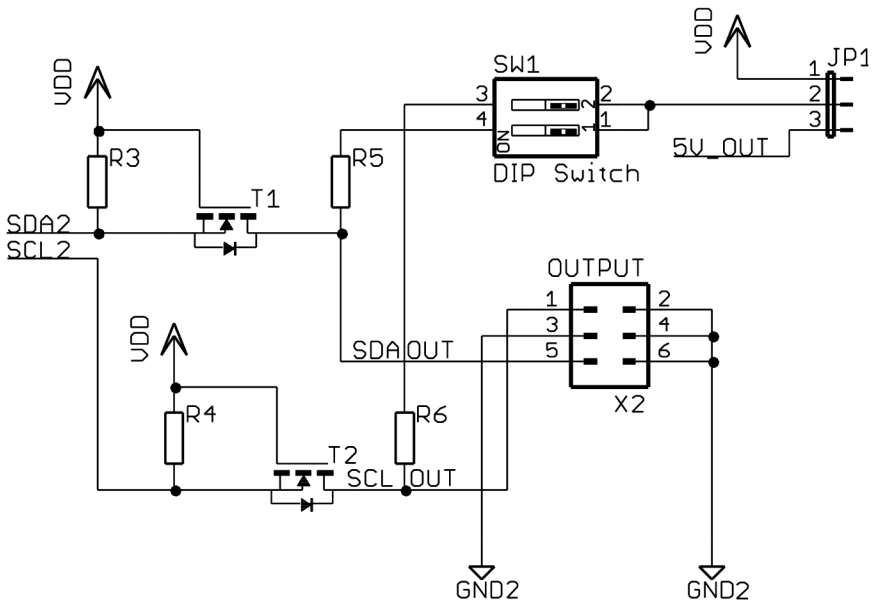
Obr. 4.2: Schéma zapojenia riadiaceho mikroprocesora prevodníka I²C/USB

4.3 Rozhranie I²C

Rozhranie I²C je zložené z niekoľkých častí. V prvom rade je potrebné zabezpečiť prispôsobenie napäťových úrovní. Možných riešení je opäť viacero, dajú sa napr. kúpiť integrované obvody určené priamo pre zbernicu I²C, no nakoniec bolo zvolené zapojenie na obr. 4.3, inšpirované aplikačnými poznámkami spoločnosti Philips Semiconductors, [5]. Na jeho správnu činnosť je potrebné vybrať vhodný tranzistor MOS-FET s kanálom typu N. Ten by mal mať čo najnižšie prahové napätie U_{GSth}, ktorého súčet s napätím na dióde nesmie prekročiť hodnotu napätia V_{DD}. Bol zvolený tranzistor 2N7002, ktorý má podľa katalógového listu

U_{GSth} = typ. 1,5 V, [6]. V prípade analýzy samotného obvodu typu slave, ktorý nie je pripojený k žiadnej zbernici, umožňuje prevodník vytvorenie vlastnej zbernice s napätím na zdvíhacom rezistore 3,3 V alebo 5 V. Úroveň napätia je možné voliť pomocou jumpra JP1 a na následné pripojenie zdvíhacích rezistorov slúži prepínač SW1.

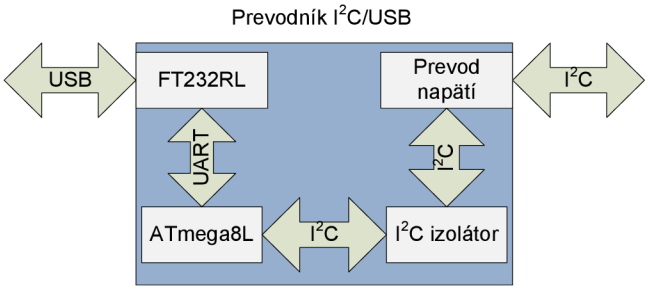
Aby sa predišlo poškodeniu počítača v prípade poruchy sledovaného zariadenia, sú mikroprocesor ako aj rozhranie USB od zbernice I²C na obr. 4.3 galvanicky oddelené. Ako oddeľovač signálov bol použitý obvod ADuM2250 od Analog Devices. Jedná sa o 5kV izolátor, priamo určený pre zbernicu I²C. Má preto dva obojsmerné kanály, jeden pre SDA a jeden pre SCL. Napájanie izolátora na výstupnej strane prevodníka je zabezpečené pomocou DC/DC meniča LME0505 spoločnosti Murrata Power Solutions a 3,3V LDO (Low Dropout) stabilizátora.



Obr. 4.3: Prispôsobenie napäťových úrovní na zbernici I²C prevodníka I²C/USB

4.4 Prepojenie jednotlivých blokov

Jednotlivé časti prevodníka už boli stručne popísané v predchádzajúcich podkapitolách. Kompletná schéma zapojenia prototypu je uvedená v prílohe A. Ako vidno z blokovej schémy na obr. 4.4, prevodník obsahuje viacero zberníc. Mikroprocesor je s počítačom prepojený pomocou sériového rozhrania UART, ktoré je v prevodníku FT232RL konvertované na USB. Na druhej strane je zbernica I²C medzi mikroprocesorom a obvodom na prispôsobenie napäťových úrovní galvanicky oddelená obvodom ADuM 2250. Obe interné I²C zbernice majú linky SCL a SDA pripojené cez zdvíhací rezistor na napätie 3,3 V. Tretia I²C zbernica je na výstupe prevodníka a má pripojiteľné zdvíhacie rezistory s možnosťou voľby logickej úrovne medzi 3,3 V a 5 V. Pokiaľ tieto rezistory nie sú pripojené, je vysoká úroveň závislá na sledovanej zbernici avšak pre správnu činnosť prevodníka je potrebné napätie minimálne 3,3 V.



Obr. 4.4: Bloková schéma prevodníku I²C/USB

Prevodník taktiež obsahuje 5 nízko príkonových LED diód. Červená dióda signalizuje pripojené napájanie, zelené diódy aktivitu na zbernici I²C a komunikáciu s počítačom a posledná, žltá LED dióda je pripojená k mikroprocesoru a jej využitie je závislé od firmwaru. Môže signalizovať napr. zahltenie vyrovnávacej pamäte, spojenie s obslužným softwarom alebo môže byť využitá pri debugovaní.

5 Firmware prevodníka I²C/USB

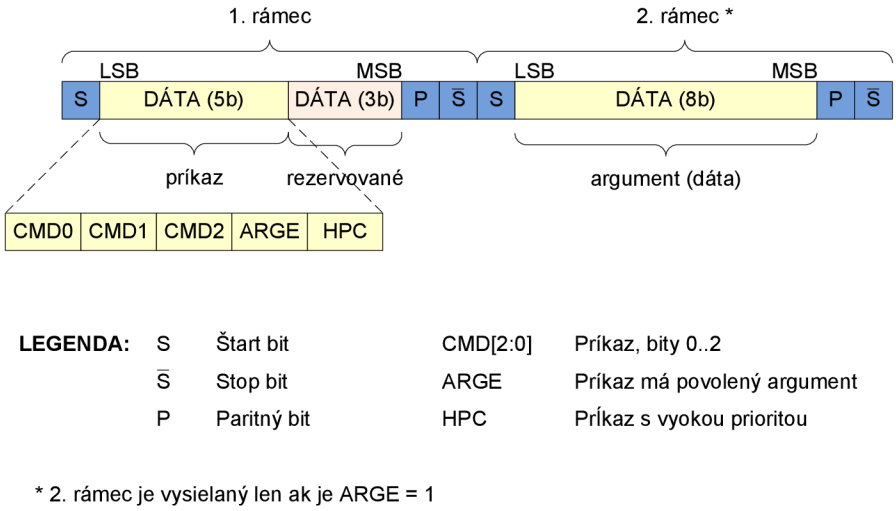
Aby bolo možné čo najefektívnejšie využiť možnosti navrhnutého hardwaru a splniť tak požiadavky na prevodník samotný, musí byť firmware prevodníka napísaný v jazyku symbolických adries. Je potrebné si uvedomiť, že mikroprocesoru trvá vykonanie jednej inštrukcie s použitým taktovacím kmitočtom 125 ns, čo je vzhľadom na požiadavky dosť dlhá doba a preto nie je vhodné spoliehať sa na to, že bude kód z vyššieho programovacieho jazyka po preklade do strojového kódu dostatočne optimálny a rýchly.

5.1 Komunikácia medzi počítačom a mikroprocesorom

Mikroprocesor prijíma príkazy z počítača prostredníctvom rozhrania UART. K dosiahnutiu maximálnej možnej prenosovej rýchlosti je využitý double speed mode, v ktorom je každý bit vzorkovaný iba ôsmimi vzorkami. Aby bol prenos dostatočne spoľahlivý, musí byť využitý externý kryštál, ktorý je oproti internému RC oscilátoru mikroprocesora výrazne presnejší. Z dôvodu voľby vhodného zabezpečenia proti chybám pri prenose bol spravený echo test s 200 bytmi prenesenými z počítača do mikroprocesora a späť, pri symbolovej rýchlosti 9600 baudov, a s prepojovacím USB káblom dlhým 180 cm. S interným RC oscilátorom bolo až 80% prenášaných dát poškodených a väčšina chýb bola odhalená kontrolou parity. Pri teste s externým kryštálom bolo spoľahlivo prenesených 100% dát. Všetky chyby vznikli na strane prijímača v mikroprocesore a boli spôsobené malým počtom vzoriek, teda využitím double speed módu v kombinácii s interným RC oscilátorom a v neposlednej rade aj zvolenou symbolovou rýchlosťou, ktorú v mikroprocesore vzhľadom na frekvenciu hodinového signálu nie je možné presne nastaviť.

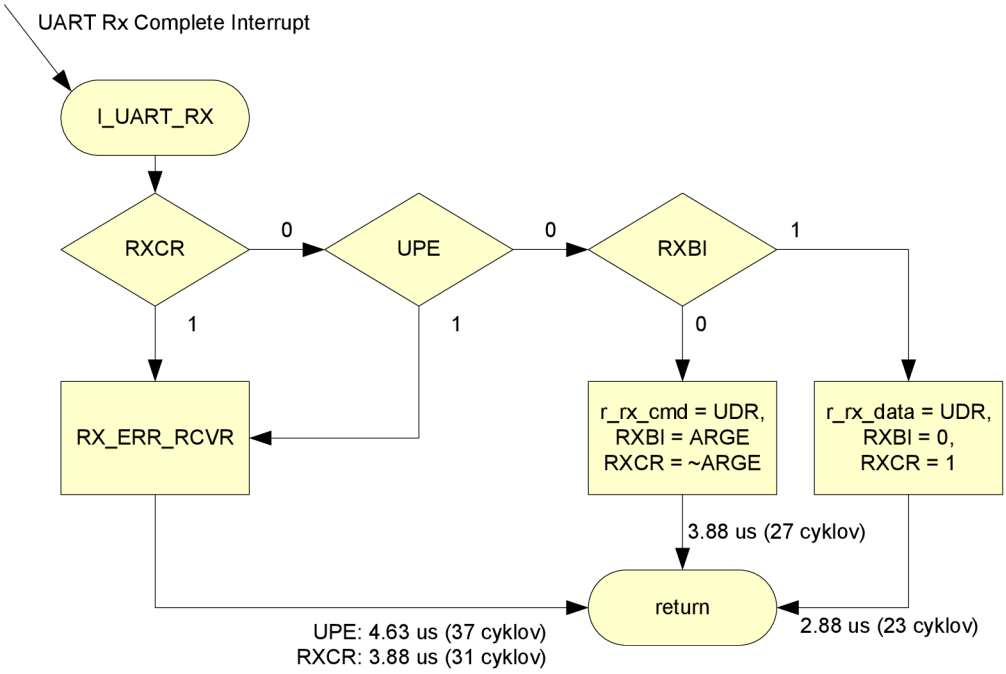
Na základe predchádzajúcich výsledkov sú dáta zabezpečenie iba párnou paritou. Jeden rámec pozostáva zo start bitu, ôsmych dátových bitov, jedného paritného bitu a jedného stop bitu (viz. obr. 5.1). To je spolu 11 bitov a výsledný čas potrebný na prenesenie rámca pri bitovej rýchlosti 1000000 b/s je potom 11 µs. Toto je zároveň interval medzi dvomi prerušeniami od prijímača UART pri vyslaní viacerých rámcov za sebou a je preto dôležité zabezpečiť aby doba, po ktorú je vykonávaná obsluha prerušenia neprekročila 11 µs.

Príkaz pre mikroprocesor pozostáva v závislosti na hodnote bitu ARGE (Argument Enable) z jedného alebo dvoch bytov, ktoré sa prenášajú tak ako je to znázornené na obr. 5.1. Význam príkazu je potom závislý na nastavenom pracovnom režime, bite HPC (High Priority Command) a bitoch CMD[2:0]. Zvyšné 3 bity sú rezervované a môžu byť využité v ďalších verziách firmwaru napr. v prípade potreby lepšieho zabezpečenia proti chybám alebo na pridanie ďalších príkazov.



Obr. 5.1: Formát rámcov medzi prevodníkom a počítačom

Komunikácia v smere k mikroprocesoru je typu stop and wait, ďalší príkaz môže byť vyslaný až po potvrdení toho predchádzajúceho správou ACK, výnimkou je iba príkaz 0x00 (Dummy frame), ktorý mikroprocesor nepotvrdzuje. Príjem dát v mikroprocesore rieši obsluha prerušenia UART Rx Complete a jej vývojový diagram je na obr. 5.2. Pre správne rozpoznanie príkazov má definované príznakové bity RXCR (RX Command Ready) a RXBI (RX Byte Index). RXCR je nastavený na 1 ak bol prijatý celý príkaz a ten je pripravený na dekódovanie. Dekodér tento príznak rozpozná, a po spracovaní príkazu ho vynuluje. Ak boli prijaté dáta počas RXCR = 1, je tento stav vyhodnotený ako chyba a zavolá sa funkcia RX_ERR_RCVR (RX Error Recovery), ktorá zabezpečí správne nastavenie stavových registrov a odoslanie správy informujúcej o zahodených dátach. Každý správne prijatý príkaz je potvrdený vopred definovanou správou ACK, ktorej súčasťou môžu byť napr. dáta prijaté zo zbernice I²C alebo informácia o stave zbernice a podobne. V prípade, že došlo pri prenose k chybe, ktorú odhalila kontrola parity, volá sa podobne ako v predchádzajúcom prípade funkcia RX_ERR_RCVR. Bit RXBI pomáha určiť poradie prijatých rámcov a umožňuje tak rozhodnúť či prijatý rámec obsahuje príkaz alebo argument príkazu.



Obr. 5.2: Spracovanie príkazov v mikroprocesore

Vo vývojovom diagrame na obr. 5.2 je tiež vyznačená doba, potrebná na vykonanie inštrukcií obsluhy prerušenia v jednotlivých prípadoch a to vrátane skoku na vektor prerušenia a návratu z obsluhy prerušenia. Z týchto údajov vyplýva, že obsluha prerušenia je vykonaná vo všetkých prípadoch dostatočne rýchlo a v žiadnom z možných scenárov nepresahuje dobu prenosu dát 11 µs.

V smere od mikroprocesora k počítaču sa príkazy nepotvrdzujú. Je to z dôvodu vyššej pravdepodobnosti správneho príjmu, ktorá je daná lepším vzorkovaním. Spracovanie v počítači je tiež rýchlejšie a je možné využiť väčšiu vyrovnávaciu pamäť.

5.2 Dekódovanie príkazov prijatých z počítača

Príkazy sú z dôvodu rovnomerného rozloženia doby dekódovania rozdelené na dve základné skupiny, od ktorých sa ďalej odvíja samotné dekódovanie.

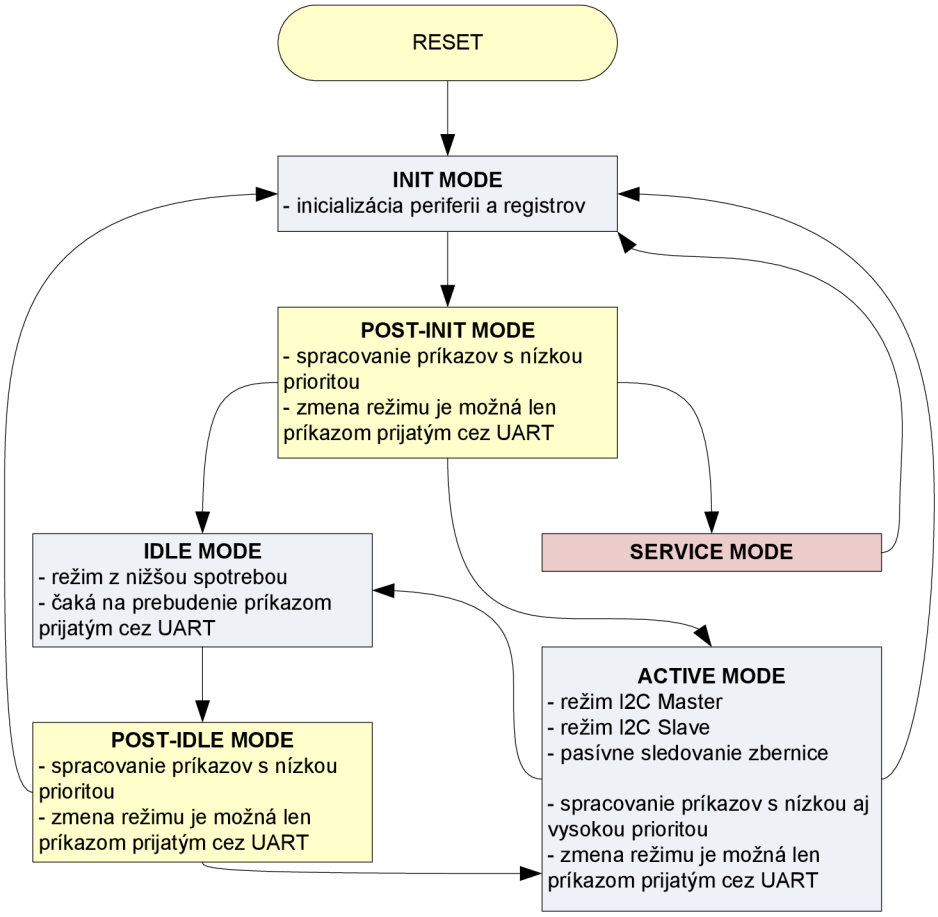
Prvou skupinou sú príkazy s vysokou prioritou, bit HPC v príkaze je rovný 1, ktoré priamo súvisia s analýzou zbernice I^2C. Tieto príkazy sú citlivé na dobu spracovania. Ich dekodér je preto optimalizovaný na rýchlosť a do programu vložený priamo bez volania funkcií.

Druhú skupinu tvoria príkazy s nízkou prioritou, tj. príkazy s bitom HPC rovným 0. Ich úlohou je predovšetkým poskytnúť počítaču požadované údaje a umožniť mu nastavenie parametrov prenosu a pracovný režim. Spoločným znakom tohto typu príkazov je, že doba ich dekódovania a spracovania neovplyvní správnu činnosť prevodníka ani inej časti analyzátora. Tieto príkazy sú spoločné pre všetky pracovné režimy a ich dekodér je napísaný ako funkcia, ktorá sa volá vždy až za dekodérom príkazov s vysokou prioritou.

Ako je naznačené na obr. 5.3, firmware prevodníka má niekoľko pracovných režimov. Tie sa môžu meniť priamo po vykonaní príslušných inštrukcií alebo príkazom prijatým cez UART. Po pripojení k USB portu počítača prejde prevodník do Init mode, kde inicializuje všetky periférie a nastaví potrebné registre. Následne sa automaticky zmení režim na Post-Init mode, ktorý slúži na spojenie prevodníka s obslužným softwarom v počítači. V Post-Init mode prevodník zotrváva do doby, kým nedostane príkaz na zmenu pracovného režimu. V tomto režime sa používa iba dekodér na spracovanie príkazov s nízkou prioritou.

Ďalším režimom je Active mode, ktorý slúži na samotnú analýzu I^2C zbernice. Môže dekódovať príkazy s vysokou aj nízkou prioritou, pričom príkazy s vysokou prioritou sú dekódované v jednotlivých sub režimoch. Sub režim je režim, v ktorom prevodník pracuje vzhľadom na zbernicu I^2C (master, slave alebo pasívne sledovanie zbernice). Každý sub režim má vlastnú sadu štvorbitových príkazov s vysokou prioritou a môže tak využiť 16 rôznych príkazov. Riadenie toku programu je pri tomto type príkazov osobitné a záleží na samotnom príkaze akým spôsobom bude vykonaný. Úlohou Active mode je teda detegovať prijatie nového príkazu a na základe bitu HPC ho odovzdať dekodéru príkazov s nízkou prioritou alebo dekodéru aktuálne nastaveného sub režimu. Po spracovaní je riadenie opäť vrátené režimu Active mode.

Prevodník tiež obsahuje režimy Idle mode a Service mode. Idle mode je súčasťou správy napájania mikroprocesora ATmega8L a slúži na zníženie spotreby v dobe kedy nie je potrebné aby mikroprocesor pracoval. Prebudenie z tohto režimu je možné vyslaním akéhokoľvek rámca cez UART, je ale vhodné použiť príkaz Dummy frame. Po prebudení prejde mikroprocesor do Post-Idle módu, kde môže spracovávať príkazy s nízkou prioritou a čaká na príkaz k zmene pracovného režimu. Service mode v tejto verzii firmwaru nemá implementovanú žiadnu funkciu a je vyhradený pre budúce použitie, napr. aktualizácia firmwaru obslužným softwarom.



Obr. 5.3: Pracovné režimy prevodníka I²C/USB

5.3 Režim I²C master

Režim master pracuje interaktívne a pomocou príkazov uvedených v tab. 5.1 má užívateľ možnosť priamo ovplyvniť priebeh komunikácie. Z počítača je vyslaný príkaz, po ktorého vykonaní je odoslaná správa ACK obsahujúca stav zbernice poprípade dáta. Po prijatí správy obslužným softwarom v počítači sa táto správa dekóduje a na jej základe sa užívateľ rozhodne akú udalosť na zbernicu odošle.

Tab. 5.1: Prehľad príkazov pre režim I²C master

Kód	Príkaz	Popis	Parameter
0x11	Send START	Odošle podmienku START	-
0x12	Send STOP	Odošle podmienku STOP	-
0x1b	Send DATA	Odošle dáta alebo adresu	Dáta (8b)
0x14	Receive DATA , ACK	Prijme dáta a potvrdí ich ACK	-
0x15	Receive DATA, NACK	Prijme dáta a potvrdí ich NACK	-
0x16	Get TWBR	Načítanie hodnoty registra TWBR (určuje periódu SCL)	-
0x1e	Set TWBR	Nastavenie registra TWBR	TWBR (8b)

Princíp činnosti prevodníka v režime master je ukázaný na výpise zdrojového kódu pre vyslanie podmienky START:

```
    ;; send START condition
_MM_START:
    ldi         r_tmp0, (1 << TWINT)|(1 << TWSTA)|(1 << TWEN)
    out         twcr,       r_tmp0
    rcall       TWI_WAIT_I
    ; cmd
    WAIT_TX_RDY
    out         udr,        r_cmd
    ; data
    WAIT_TX_RDY
    in          r_tmp0,     twsr
    out         udr,        r_tmp0
    RXCR_CLEAR
    rjmp        _ACTIVE_MODE_08
```

Po dekódovaní príkazu a jeho vykonaní čaká funkcia `TWI_WAIT_I` na dokončenie udalosti na I²C zbernici. Následne čaká makro `WAIT_TX_READY` na uvoľnenie výstupného buffera a odošle sa správa ACK. Tou je príkaz, ktorým bola operácia spustená spolu s argumentom predstavujúcim výsledok operácie, stav zbernice alebo prijaté dáta. Makro `RXCR_CLEAR` povolí prijatie ďalšieho príkazu a riadenie programu sa vráti režimu Active mode. Obdobne funguje aj generovanie ostatných I²C udalostí v režime master.

Komentovaná ukážka komunikácie medzi počítačom a obvodom PCA9536 (I/O expandér pracujúci ako I²C slave), pripojeným na 3,3V zbernicu I²C je na nasledujúcom výpise (S – dáta odoslané z PC, R – správa ACK):

```
Kombinovaný režim (Write + Read), čítanie registra 3
(Configuration Register)
-------------------------------------------------------------
S  0x11          * odošli podmienku START
R  0x11 0x08     Podmienka START odoslaná
S  0x1B 0x82     * odošli byte 0x82 (adresa write)
R  0x1B 0x18     Adresa SLA+W potvrdená bitom ACK
S  0x1B 0x03     * odošli byte 0x03 (register 3)
R  0x1B 0x28     Dáta odoslané a potvrdené bitom ACK
S  0x11          * odošli podmienku START (repeated START)
R  0x11 0x10     Podmienka repeated START odoslaná
S  0x1B 0x83     * odošli byte 0x83 (adresa read)
R  0x1B 0x40     Adresa SLA+R potvrdená bitom ACK
S  0x15          * prijmi byte a potvrď NACK
R  0x15 0x07     Prijatý byte 0x07
S  0x12          * odošli podmienku STOP
R  0x12 0x00     Podmienka STOP odoslaná
```

Tento spôsob riadenia komunikácie je v súlade so špecifikáciou zbernice I²C a väčšina zariadení by s ním nemala mať problém. Výnimku môžu tvoriť jednoduchšie zariadenia typu single-master, ktoré nekontrolujú stav liniek po ich uvoľnení a nedokážu tak korektne pracovať na zbernici s viacerými zariadeniami typu master.

5.4 Režim I²C slave

Režim I²C slave funguje na podobnom princípe ako režim master. Prevodník prijíma príkazy z počítača a tie potom vykonáva. Aby bolo možné pracovať so zbernicou interaktívne, prevodník po prijatí I²C udalosti podrží hodinový signál na nízkej úrovni (viz. 2.2.4) a to až pokiaľ užívateľ nerozhodne o ďalšej udalosti. Prevodník v tomto prípade využíva podobnú sadu (tab. 5.2) príkazov ako v režime master, len ich implementácia je odlišná.

Tab. 5.2: Prehľad príkazov pre režim I²C slave

Kód	Príkaz	Popis	Parameter
0x11	Enable	Povolí čakanie na slave adresu	-
0x12	Disable	Zastaví čakanie na slave adresu	-
0x1b	Send DATA	Odošle dáta alebo adresu	Dáta (8b)
0x14	Receive DATA , ACK	Prijme dáta a potvrdí ich ACK	-
0x15	Receive DATA, NACK	Prijme dáta a potvrdí ich NACK	-
0x1e	Set ADDR B1	Nastaví adresu (7 b / 10 b High)	Adresa (8b)
0x1f	Set ADDR B2	Nastaví adresu (10 b Low)	Adresa (8b)

Dáta do počítača sú posielané v 2 bytovom formáte, pričom prvý byte je potvrdzovací a je rovnaký ako prijatý príkaz. Druhý byte obsahuje podľa typu príkazu stav jednotky TWI alebo dáta. Výnimkou je príkaz Receive DATA, ktorý odosiela stav jednotky TWI aj prijaté dáta, teda spolu 3 byty.

5.5 Pasívny režim I²C

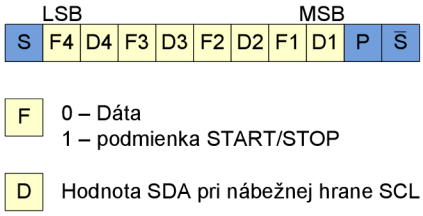
Na rozdieľ od predchádzajúcich dvoch režimov, v pasívnom režime nie je možné využiť jednotku TWI mikroprocesora a dáta je potrebné vzorkovať softwarovo. Mikroprocesor má linky SCL a SDA pripojené taktiež na externé prerušenia INT0 a INT1, ich použitie ale nie je výhodné pretože doba potrebná na volanie a návrat z prerušenia je príliš dlhá. Hrany sú preto detegované softwarovo.

Bez podmienky START nie je možné učiť význam jednotlivých bitov, prevodník preto podporuje v pasívnom režime dva spôsoby spustenia analýzy (tab. 5.3). Je to spustenie prvou podmienkou START, čo je vhodné na zachytenie samostatného rámca. Druhý spôsobom je spustenie analýzy za prvou podmienkou STOP, po ktorej je už možné určiť nie len význam bitov ale tiež rozlíšiť podmienku START od podmienky Repeated START za cenu toho, že prvý rámec bude ignorovaný.

Tab. 5.3: Prehľad príkazov pre pasívne sledovanie zbernice I²C

Kód	Príkaz	Popis	Parameter
0x11	Start after START Condition	Spustenie vzorkovania podmienkou START	-
0x12	Start after STOP Condition	Spustenie vzorkovania za podmienkou STOP	-

Po splnení spúšťacej podmienky už nasleduje samotné vzorkovanie. Na uloženie vzoriek je použitý 8bitový register, do ktorého sa dáta pomocou bitového posuvu ukladajú od LSB a posúvajú smerom k MSB (obr. 5.4). Hodnota za nábežnou hranou hodinového signálu je uložená a následne porovnaná s hodnotou pred dobežnou hranou. V prípade že sa obe hodnoty zhodujú, je za touto hodnotou uložený ešte bit 0, ktorý znamená že sa jedná o dáta, v opačnom prípade sa uloží bit 1, čo znamená že sa jedná o podmienku START alebo STOP. Do registra je takto možné uložiť 4 vzorky, ktoré sa po pretečení počítadla modulo 4 odošlú do počítača.



Obr. 5.4: Formát rámca odosielaného do PC pri pasívnom sledovaní zbernice I²C

Ak je počas merania nastavený príznak RXCR signalizujúci prijatie nového príkazu, meranie sa pri vysokej úrovni na SCL ukončí a zostávajúce dáta sa odošlú do počítača. V prípade, že je posledný I²C rámec neúplný mal by byť v počítači ignorovaný ale to už je závislé na implementácii pasívneho režimu v ovládacom softwari.

6 Ovládače prevodníka pre počítač

Aby bolo možné prevodník pripojiť k počítaču, je potrebné nahrať do EEPROM obvodu FT232RL konfiguračné údaje a vykonať drobné úpravy v samotných ovládačoch.

Na tento účel je určená utilita FT Prog, ktorá je spolu s CDM (Combined Driver Model) ovládačmi voľne dostupná na stránkach spoločnosti FTDI, *www.ftdichip.com*. Po pripojení analyzátora k počítaču a spustení programu FT Prog sa voľbou File/Template v menu otvorí súbor I2CAnalyzer.xml. Jedná sa o šablónu vytvorenú k I^2C analyzátoru, ktorá obsahuje potrebné nastavenia. Výberom položky Devices/Program v menu sa údaje nahrajú do prevodníka.

Ďalším krokom je inštalácia ovládačov. Tie je nutné pred samotnou inštaláciou najskôr upraviť tak, aby mohol obvod FT232RL pracovať v režimoch s neštandardnými symbolovými rýchlosťami. Prevodník by mohol pracovať aj so štandardnými prenosovými rýchlosťami, tie ale nie sú vhodné vzhľadom k hodinovému kmitočtu mikroprocesora. Napríklad v prípade využitia rýchlosti 921600 baudov, musí byť v mikroprocesore nastavená symbolová rýchlosť 1000000 baudov, čo predstavuje relatívnu odchylku 7,84 %. Úprava spočíva v editácii súboru ftdiport.inf, ktorý je súčasťou CDM ovládačov. Preddefinovaným symbolovým rýchlostiam pod 9600 baudov sa priradia parametre zodpovedajúce požadovaným rýchlostiam. Jedná sa o parametre subdivisor a divisor. Podrobný popis oboch parametrov, ich výpočet ako aj úprava ovládačov sú uvedené v [7]. Na požadovanú úpravu ale postačuje tab. 6.1, obsahujúca symbolové rýchlosti zobrazované v nastaveniach ovládačov a k nim priradené skutočné rýchlosti, ktorých je schopný dosiahnuť obvod FT232RL a mikroprocesor s taktovacím kmitočtom 8 MHz.

Tab. 6.1: Prehľad nastavených symbolových rýchlostí

Nastavená rýchlosť [baud]	Skutočná rýchlosť [baud]	Divisor	Sub-divisor	HEX	UBRR (U2X=1)	Rýchlosť MCU [baud]
300	200000	15	0	0F,00,00,00	4	200000
600	250000	12	0	0C,00,00,00	3	250000
1200	500000	6	0	06,00,00,00	1	500000
2400	1000000	3	0	03,00,00,00	0	1000000
4800	2000000	1	0,5	01,40,00,00	-	-
9600	9600	312	0,5	38,41,00,00	103	9615
14400	14423	208	0	D0,00,00,00	68	14492
19200	19200	156	0,25	9C,80,00,00	51	19231
38400	38400	78	0,125	4E,C0,00,00	25	38461
57600	57692	52	0	34,00,00,00	16	58824
115200	115385	26	0	1A,00,00,00	8	111111
230400	230769	13	0	0D,00,00,00	3	250000
460800	461539	6	0,5	06,40,00,00	1	500000
921600	923077	3	0,25	03,80,00,00	0	1000000

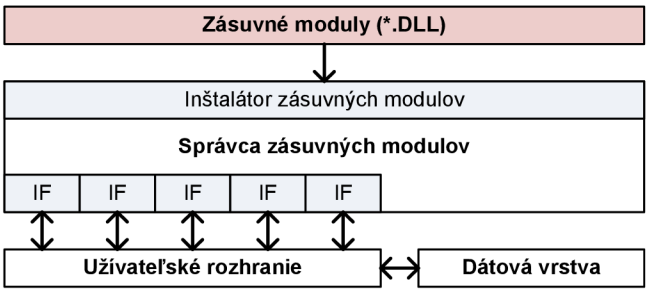
V prvých piatych riadkoch tabuľky sú premapované hodnoty, ostatné hodnoty ostali nezmenené. Stĺpec divisor a sub-divisor predstavuje hodnotu deličky, jej celú a desatinnú časť. Je to číslo ktorým je potrebné vydeliť nominálnu hodnotu symbolovej rýchlosti 3 Mbaudy aby bola dosiahnutá požadovaná rýchlosť. Táto hodnota je v tabuľke uvedená hexadecimálne tak, ako ju treba zapísať do súboru ftdiport.inf, teda byty sú v obrátenom poradí. UBRR predstavuje hodnotu tohto registra v mikroprocesore ATmega8L pre správne nastavenie symbolovej rýchlosti v double-speed móde. Nasledujúci výpis ukazuje, ktorú časť súboru ftdiport.inf treba upraviť, pričom zmeny oproti pôvodnému súboru sú zvýraznené.

```
[FtdiPort232.NT.HW.AddReg]
HKR,,"UpperFilters",0x00010000,"serenum"
HKR,,"ConfigData",1,11,00,3F,3F,0F,00,00,00,0C,00,00,00,06,00
,00,00,03,00,00,00,01,40,00,00,38,41,00,00,9C,80,00,00,4E,C0,
00,00,34,00,00,00,1A,00,00,00,0D,00,00,00,06,40,00,00,03,80,0
0,00,00,00,00,00,D0,80,00,00
```

Po týchto úpravách je možné štandardným spôsobom nainštalovať ovládače virtuálneho sériového portu. Zmeny v súbore ovládačov spôsobia v systémoch Windows Vista a Windows 7, že sa pri ich inštalácii objaví výstraha informujúca o tom, že boli zmenené súbory ovládačov. Tú je možné ignorovať.

7 Obslužný software pre počítač

Obslužný software je napísaný v jazyku C# pre .NET Framework 3.5. Užívateľovi umožní nie len komunikovať s hardwarovým prevodníkom ale aj grafickú interpretáciu dát, ich ďalšie spracovanie a uloženie do súboru. Pri návrhu bol vzhľadom na množstvo spôsobov akými je možné získané dáta zobraziť kladený dôraz na možnosť prispôsobenia programu danej úlohe. Preto je celý program rozdelený na niekoľko častí, ktoré sa pripájajú ako zásuvné moduly cez príslušné rozhranie (obr. 7.1). Toto rozhranie je potom zárukou vzájomnej kompatibility zásuvných modulov.



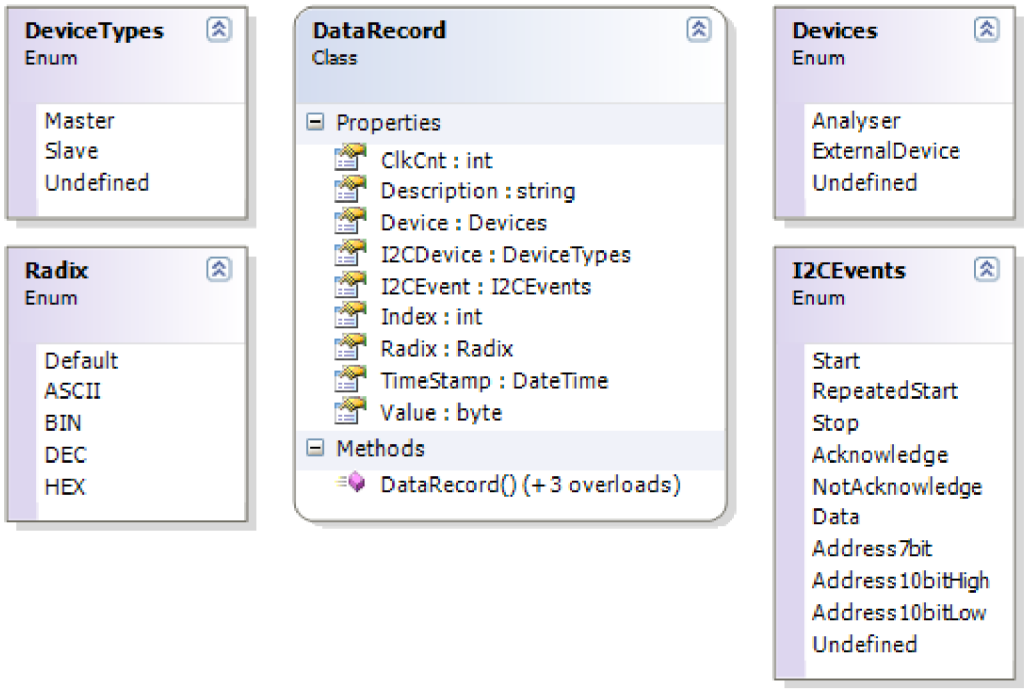
Obr. 7.1: Usporiadanie programu

7.1 Dátová vrstva

Úlohou dátovej vrstvy je správa dát a všetkého čo s nimi súvisí. Najdôležitejšími triedami dátovej vrstvy sú triedy `DataRecord` a `DataManager`.

Trieda `DataRecord` predstavuje jednu udalosť na I²C zbernici. Zoznam vlastností ako aj použité výčtové typy je vidieť na obr. 7.2. Trieda `DataManager`, ktorá uchováva kolekciu dát typu `DataRecord` je odvodená od generickej triedy `BindingList<DataRecord>`. Obsahuje všetky metódy potrebné pre prácu s uchovávanými I²C udalosťami. Na pridanie nového záznamu slúži metóda `AddEvent()`, ktorá zo vstupných parametrov vytvorí `DataRecord` a ten uloží do kolekcie pomocou metódy rodičovskej triedy `Add()`. Vyhľadávanie je realizované metódami `Find()`, `FindNext()` a privátnou metódou `FindCore()`. Metóda `Find()` uloží vstupné parametre ako privátne vlastnosti triedy, nastaví index na začiatok vyhľadávania a potom zavolá metódu `FindCore()`, ktorá jej vráti index nájdeného záznamu. Metóda `FindNext()` volá taktiež metódu `FindCore()` ale tentokrát s vlastnosťami uloženými metódou `Find()`. Index ponecháva na pôvodnej hodnote, čím sa pokračuje vo vyhľadávaní od posledného záznamu. Filtrovať záznamy je možné na úrovni I²C rámcov a to na základe zadanej hodnoty udalosti so 7 alebo 10bitovou adresou. Metóda `Filter()` odstráni všetky udalosti od podmienky START až po podmienku STOP, ktoré obsahujú zadanú adresu, poprípade je možné tento proces invertovať a odstrániť rámce, ktoré zadanú adresu neobsahujú. Zrušiť aplikované filtre je následne možné pomocou metódy `RemoveFilter()`. Trieda `DataManager` obsahuje tiež metódu `SaveToXml()` na uloženie kolekcie dát do XML súboru. Ten je možné otvoriť v inej aplikácii napr. MS Excel alebo znova načítať metódou `LoadFromXml()`. Okrem týchto

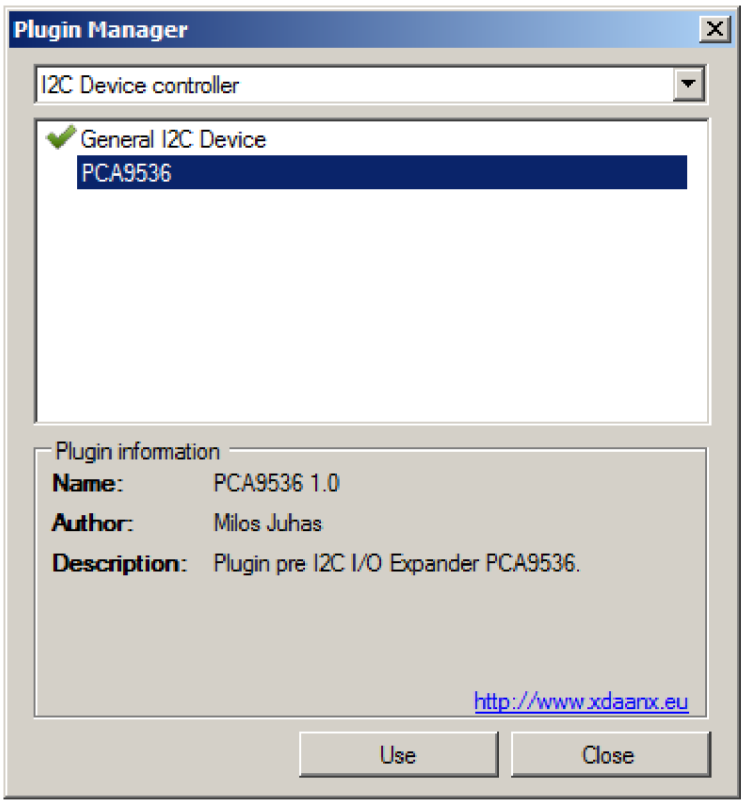
základných metód pre prácu s kolekciami obsahuje trieda DataManager metódy pre vzájomnú konverziu čísla udalosti a čísla periódy hodinového signálu.



Obr. 7.2: Trieda DataRecord

7.2 Správca zásuvných modulov

Správca zásuvných modulov je podobne ako zásuvné moduly DLL knižnica. Obsahuje nie len formulár pre výber zásuvných modulov (obr. 7.3) ale aj triedy a rozhrania potrebné na ich inštaláciu a pripojenie k programu. Po výbere kategórie zásuvného modulu z rozbalovacieho zoznamu sa z generickej triedy `Plugin<IPlugin>` načíta abecedne zobrazený aktuálny zoznam obsahujúci informácie o zásuvnom module. Súčasťou týchto informácií je tiež regulárny výraz vymedzujúci verzie programu, s ktorými je modul kompatibilný. Kompatibilita je overená a v prípade, že zásuvný modul nie je kompatibilný s aktuálnou verziou programu I²C Analyzer, zobrazí sa pri jeho názve ikona upozorňujúca na túto skutočnosť užívateľa.



Obr. 7.3: Dialógové okno pre výber zásuvného modulu

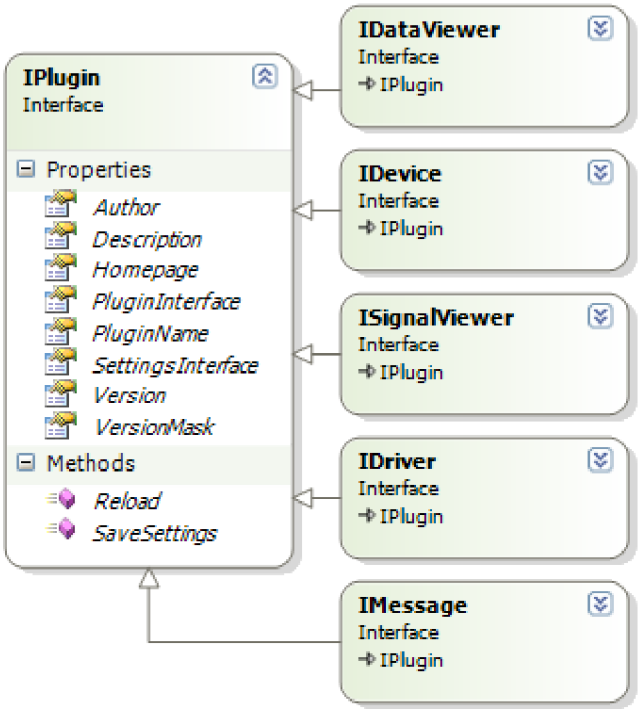
7.2.1 Rozhrania

Program podporuje spolu 7 rôznych rozhraní, ktoré slúžia na pripojenie zásuvného modulu a zároveň na jeho identifikáciu pri vyhľadávaní v DLL súboroch v adresári s programom.

7.2.1.1 Rozhranie IPlugin

Program podporuje 5 typov zásuvných modulov a pre každý typ je definované vlastné rozhranie odvodené od rozhrania `IPlugin` (obr. 7.4).

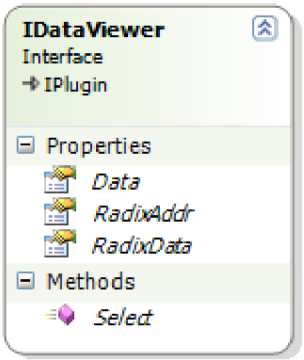
Rozhranie `IPlugin` obsahuje vlastnosti a metódy spoločné pre všetky zásuvné moduly. Jedná sa predovšetkým o vlastnosti obsahujúce názov, popis, autora a verziu programu. Vlastnosť `PluginInterface` obsahuje referenciu na komponentu s užívateľským rozhraním zásuvného modulu a vlastnosť `SettingsInterface` referenciu na komponentu s užívateľským rozhraním pre nastavenia zásuvného modulu. Pokiaľ je hodnota niektorej z týchto vlastností nastavená na `null`, znamená to, že zásuvný modul danú komponentu nemá a nebude sa teda používať. Vlastnosť `VersionMask` obsahuje regulárny výraz, ktorý sa pri kontrole kompatibility aplikuje na verziu programu I²C Analyzer a dáva tak autorovi zásuvného modulu možnosť, oznámiť triede inštalujúcej zásuvný modul do programu, s ktorými verziami je zásuvný modul kompatibilný. Metódu `Reload()` je možné využiť pokiaľ je potrebné opätovne vynútiť inicializáciu zásuvného modulu a metóda `SaveSettings()` slúži na uloženie vlastností zásuvného modulu do konfiguračného programu.



Obr. 7.4: Rozhranie IPlugin

7.2.1.2 Rozhranie IDataViewer

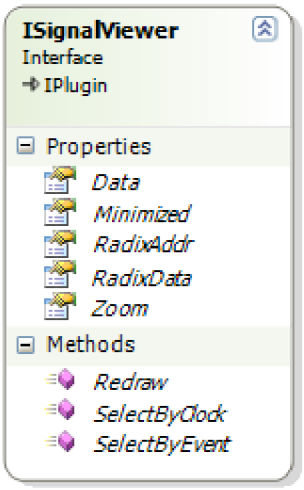
Rozhranie `IDataViewer` musia implementovať zásuvné moduly, ktoré slúžia na zobrazenie dát protokolu I²C uložených v dátovej vrstve, zvyčajne vo forme tabuľky. Na inštanciu príslušnej triedy dátovej vrstvy je možné nastaviť referenciu pomocou vlastnosti `Data`. Rozhranie tiež umožňuje globálne zmeniť nastavenie formátu zobrazenia adresy a dát a to pomocou vlastností `RadixAddr` a `RadixData` napr. z menu programu. Na označenie konkrétneho záznamu slúži metóda `Select()`, ktorá nájde svoje uplatnenie napr. pri vyhľadávaní. Žiadne ďalšie metódy na prácu s dátami nie sú potrebné, pretože zásuvný modul má možnosť reagovať na zmeny záznamov pomocou udalostí poskytovaných dátovou vrstvou.



Obr. 7.5: Rozhranie IDataViewer

7.2.1.3 Rozhranie ISignalViewer

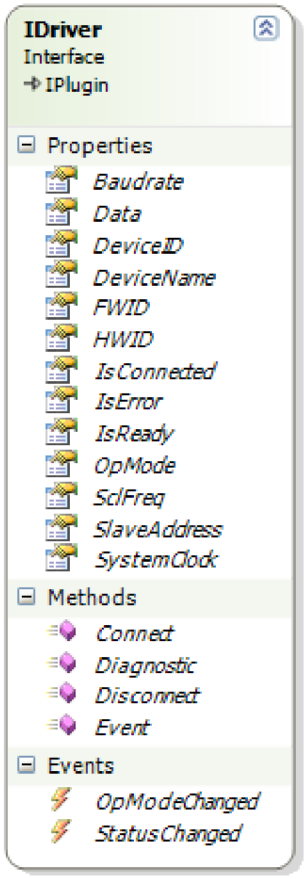
Rozhranie `ISignalViewer` je určené pre zásuvné moduly vykresľujúce signál protokolu I²C, uložený v dátovej vrstve programu. S tou modul komunikuje prostredníctvom referencie, ktorá mu bola predaná vo vlastnosti `Data`. Pokiaľ užívateľ priebeh signálu sledovať momentálne nepotrebuje je možné využiť vlastnosť `Minimized` na jeho skrytie. Podobne ako v rozhraní `IDataViewer` je tu podpora pre globálne nastavenie formátu zobrazovanej adresy a dát a pribudla ešte vlastnosť `Zoom`, ktorá slúži na zväčšenie sledovaného priebehu. Rozhranie ďalej podporuje metódu `Redraw()` na obnovenie vykresľovacej plochy. Prechod na konkrétnu časť priebehu je možné realizovať na základe poradového čísla hodinového signálu, metóda `SelectByClock()` alebo na základe čísla I²C udalosti metódou `SelectByEvent()`.



Obr. 7.6: Rozhranie ISignalViewer

7.2.1.4 Rozhranie IDriver

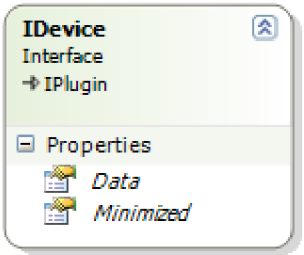
Rozhranie `IDriver` (obr. 7.7) slúži na pripojenie zásuvných modulov, ktoré komunikujú s hardwarovým prevodníkom a cez toto rozhranie tak sprostredkúvajú funkcie prevodníka ostatným častiam programu. Najväčším prínosom rozhrania `IDriver` je tak nezávislosť programu ako celku na prevodníku, ktorý je pripojený k počítaču. Program môže pracovať s rôznymi verziami toho istého prevodníka, s prevodníkom pripojeným cez rôzne porty alebo s ľubovoľným zariadením, pre ktoré bude existovať zásuvný modul spĺňajúci rozhranie `IDriver`. Môže sa dokonca jednať aj o triedu, ktorá iba simuluje I²C zbernicu. Zoznam vyžadovaných vlastností, metód a udalostí je na obr. 7.7. Medzi najdôležitejšie patria vlastnosť `IsConnected` a `IsReady`, ktoré signalizujú stav zariadenia a vlastnosť `OpMode`, ktorá umožňuje zmeniť alebo zistiť pracovný režim prevodníka. Metódy `Connected()` a `Disconnected()` slúžia na nadviazanie a ukončenie spojenia s prevodníkom a metóda `Diagnostic()` na test spojenia. Pokiaľ chce iný zásuvný modul pracovať so zbernicou I²C, má k dispozícii metódu `Event()`, ktorá je spoločná pre režim master, slave aj pasívne sledovanie zbernice. Zásuvné moduly, ktoré potrebujú byť okamžite informované o zmene stavu prevodníka môžu využiť udalosti `OpModeChanged` a `StatusChanged`.



Obr. 7.7: Rozhranie IDriver

7.2.1.5 Rozhranie IDevice

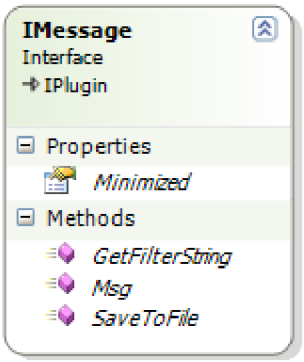
Rozhranie `IDevice` je určené pre zásuvné moduly predstavujúce zariadenie na I²C zbernici. Z pohľadu funkcionality programu má tento typ zásuvných modulov najväčší význam. Umožňuje totiž pripojiť k programu I²C Analyzer zásuvný modul prakticky pre akúkoľvek súčiastku, ktorá je pripojená k I²C zbernici. Môže tak zobrazovať jej registre, nahrať obsah pamäte EEPROM zo súboru alebo vykresliť graf s teplotou z teplotného senzora. Samotné rozhranie `IDevice` obsahuje len vlastnosť `Data` na pripojenie k dátovej vrstve a vlastnosť `Minimized`, ktorá umožňuje skryť zásuvný modul z hlavného formulára.



Obr. 7.8: Rozhranie IDevice

7.2.1.6 Rozhranie IMessage

Toto rozhranie (obr. 7.9) je určené pre zásuvný modul, ktorý spravuje hlásenia od všetkých častí programu, či už je to poznámka, varovanie alebo chyba. Dialógové okná sú pre väčšinu užívateľov iba niečím čo ich núti kliknúť na tlačítko OK aby mohli pokračovať v práci a málokedy ich čítajú. Preto ak nie je nutné aby užívateľ rozhodol o ďalšom kroku, budú hlásenia posielané cez toto rozhranie do zásuvného modulu, ktorý ich vhodným spôsobom zobrazí a poprípade aj archivuje.

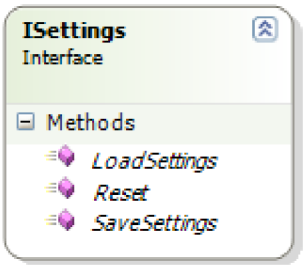


Obr. 7.9: Rozhranie IMessage

Vlastnosť `Minimized` slúži na skrytie alebo opätovné zobrazenie komponenty zásuvného modulu z hlavného formulára. Pre prípad, že by chcel užívateľ exportovať hlásenia do súboru je k dispozícii metóda `SaveToFile()`. Formát súborov, sa môže pre jednotlivé zásuvné moduly odlišovať a preto je možné pomocou metódy `GetFilterString()` získať filtrovací reťazec používaný v dialógovom okne pre uloženie súboru. Samotné hlásenie je potom odoslané metódou `Msg()`, ktorej parametrami sú typ, odosielateľ a text správy.

7.2.1.7 Rozhranie ISettings

Rozhranie `ISettings` (obr. 7.10) je potrebné implementovať do tried, ktoré slúžia na nastavenie jednotlivých zásuvných modulov. Tým, že všetky zásuvné moduly využívajú rovnaké rozhranie je zaručené, že sa s nimi pracuje vždy rovnakým spôsobom bez ohľadu na vnútornú implementáciu jednotlivých metód. Užívateľské rozhranie tieto metódy pozná a môže tak vygenerovať formulár pre nastavenie programu do ktorého pridá inštancie tried spĺňajúce rozhranie `ISettings`. Rozhranie je tvorené metódami `LoadSettings()` a `SaveSettings()`, ktoré slúžia na načítanie a uloženie nastavení do konfiguračného súboru a metódou `Reset()` na obnovenie predvolených nastavení zásuvného modulu.

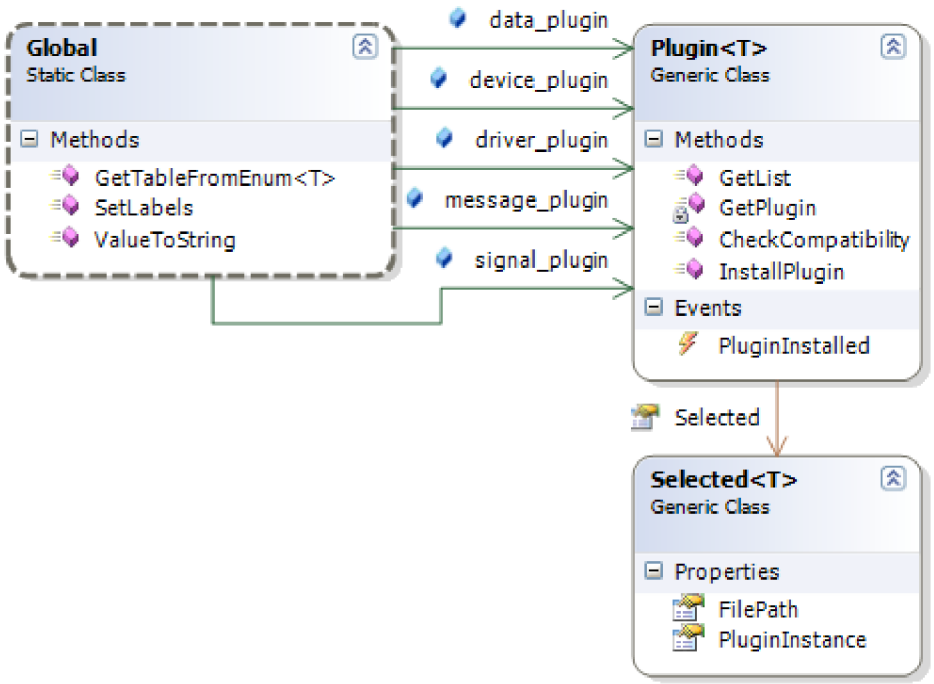


Obr. 7.10: Rozhranie ISettings

7.2.2 Inštalácia zásuvných modulov

Správca zásuvných modulov obsahuje statickú triedu `Global`, ku ktorej je možné pristupovať z viacerých častí programu. Okrem niekoľkých metód využívaných zásuvnými modulmi sú v nej ešte umiestnené inštancie generickej triedy `Plugin<T>` (kde T je rozhranie), pre všetky kategórie zásuvných modulov (obr. 7.11). Triedu `Plugin<T>` tvorí metóda `GetList()` pre načítanie zoznamu zásuvných modulov danej kategórie z adresára programu. Následne pomocou metódy `CheckCompatibility()` overí kompatibilitu zásuvných modulov zo zoznamu a označí tie, ktoré nie sú kompatibilné s danou verziou programu. V prípade, že si užívateľ vyberie nový zásuvný modul, zavolá sa metóda `InstallPlugin()`, ktorej jediným parametrom je cesta k DLL knižnici. Pokiaľ zásuvný modul v danom súbore existuje je do vlastnosti `Selected` uložená jeho inštancia a vyvolaná udalosť `PluginInstalled`. Túto udalosť využíva užívateľské rozhranie k tomu aby mohlo po inštalácii nového zásuvného modulu nastaviť jeho vlastnosti a zobraziť ho na hlavnom formulári.

Metódy `InstallPlugin()` aj `GetList()` využívajú k načítaniu zásuvného modulu z DLL knižnice a teda aj k vytvoreniu vlastnosti `Selected` privátnu metódu `GetPlugin()`. Tá najskôr overí, či daný súbor skutočne existuje. Ak áno, pokúsi sa načítať jeho assembly [8]. Z assembly je načítaný zoznam všetkých typov a pokiaľ existuje verejná trieda implementujúca dané rozhranie vytvorí sa vlastnosť `Selected` s inštanciou zásuvného modulu a cestou k súboru. Nakoniec sa táto vlastnosť vráti metóde `InstallPlugin()` alebo `GetList()`, ktoré ju využijú podľa svojich potrieb.

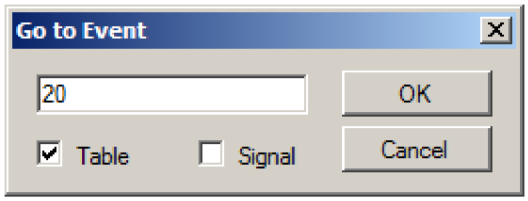


Obr. 7.11: Inštalácia zásuvných modulov

7.3 Užívateľské rozhranie

Užívateľské rozhranie samo o sebe obsahuje len základné prvky ako je hlavné menu a panel s nástrojmi. Komponenty s grafickým rozhraním jednotlivých zásuvných modulov sú pripojené až po ich nainštalovaní správcom zásuvných modulov, pričom grafické rozhranie samo zadáva požiadavku na inštaláciu naposledy použitých zásuvných modulov, ktorých názvy súborov má uložené vo svojom konfiguračnom súbore. Každý typ zásuvného modulu má vopred vyhradený priestor na formulári, ktorý sa dokáže prispôsobiť rozmerom zásuvného modulu a pokiaľ užívateľ daný modul nepotrebuje, je možné ho skryť. V priestore nevyužitom ostatnými zásuvnými modulmi je potom roztiahnutý zásuvný modul pre zobrazenie dát.

Súčasťou užívateľského rozhrania sú tiež dialógové okná, ktoré nie sú zobrazené na hlavnom formulári ale až po vybratí položky z menu, napr. obr. 7.12. Pokiaľ sú v týchto dialógových oknách zadávané hodnoty udalostí (napr. pre filter alebo vyhľadávanie), je možné ich zadať viacerými spôsobmi: binárne (s prefixom 0b), dekadicky (bez prefixu) alebo hexadecimálne (s prefixom 0x). Pokiaľ je hodnota zadaná v nesprávnom formáte alebo prekračuje povolený rozsah, farba písma sa zmení na červenú.



Obr. 7.12: Dialógové okno na zobrazenie udalosti podľa poradového čísla

Dialógové okno pre nastavenie programu si dynamicky vytvára záložky pre jednotlivé zásuvné moduly, ktoré obsahujú grafickú komponentu pre nastavenie. Po potvrdení zmien tlačítkom OK sa potom pre každý z týchto zásuvných modulov zavolá metóda `SaveSettings()`, ktorej podpora je vyžadovaná rozhraním `ISettings` a nastavenia sa uložia do konfiguračného súboru.

Užívateľské rozhranie musí takisto ošetriť pokus ukončenie aplikácie v prípade, kedy nie sú uložené dáta alebo prebieha komunikácia s prevodníkom.

7.4 Predvolené zásuvné moduly

7.4.1 DataViewer

DataViewer je predvolený zásuvný modul pre zobrazenie dát protokolu I²C v programe I²C Analyzer. Jeho základom je tabuľka `DataGridView` (obr. 7.13), ktorá na prepojenie s dátovou vrstvou využíva binding.

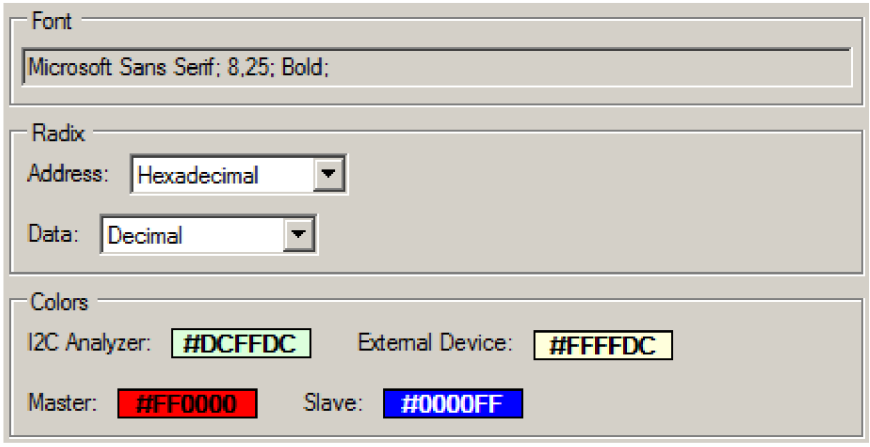
Dáta nie sú zobrazené priamo vo formáte v akom sú uložené v dátovej vrstve ale sú formátované pomocou udalosti `CellFormatting`. Pokiaľ sú k dispozícii príslušné reťazce, hlavička tabuľky a názov udalosti sú lokalizované podľa nastaveného jazyka. Čas je zobrazovaný vždy v jednotnom formáte (yyyy-dd-MM HH:mm:ss,fff) nezávisle na lokálnych nastaveniach operačného systému. Pre zobrazenie hodnoty je k dispozícii hneď niekoľko možností (ASCII,

binárne, dekadicky alebo hexadecimálne). Z kontextového menu si môže užívateľ vybrať globálne nastavenie formátu zvlášť pre adresu a zvlášť pre dáta a zároveň je podporované explicitné nastavenie pre vybranú položku. Adresa sa zobrazuje ako 8bitová hodnota udalosti spolu s bitom R/W a v zátvorke je potom zobrazená 7bitová hodnota adresy. Do stĺpca s poznámkou môže byť vložená spojená hodnota viacerých udalostí a to 16 alebo 32 bitov, Little Endian alebo Big Endian, poprípade je možné spojiť označené udalosti alebo vložiť vlastnú poznámku. Po dvojitom kliknutí na riadok s I²C udalosťou sa volá príslušná metóda `SelectByEvent()` v zásuvnom module pre zobrazenie priebehu signálu, ktorá túto položku zobrazí.

No.	Timestamp	Event	Value	Note
15	2011-05-11 03:12:06,966	Acknowledge		
16	2011-05-11 03:12:18,177	Repeated Start Condition		
17	2011-05-11 03:12:35,885	Address (7 bit) + R	0x83 (0x41)	
18	2011-05-11 03:12:35,888	Acknowledge		
19	2011-05-11 03:12:46,524	Data Byte	0b00001100	
20	2011-05-11 03:12:46,527	Acknowledge		
21	2011-05-11 03:12:48,619	Data Byte	12	
22	2011-05-11 03:12:48,622	Acknowledge		
23	2011-05-11 03:12:55,050	Data Byte	12	0xC0C
24	2011-05-11 03:12:55,053	Not Acknowledge		
25	2011-05-11 03:13:00,873	Stop Condition		

Obr. 7.13: DataViewer - predvolený zásuvný modul pre výpis dát

V nastaveniach zásuvného modulu (obr. 7.14) je tiež na výber typ a veľkosť písma ako aj farba textu a pozadia. Farbou textu je rozlíšené, či bola udalosť vyvolaná zariadením typu master alebo slave a farbou pozadia potom zdroj udalosti, teda analyzátor alebo externé zariadenie.

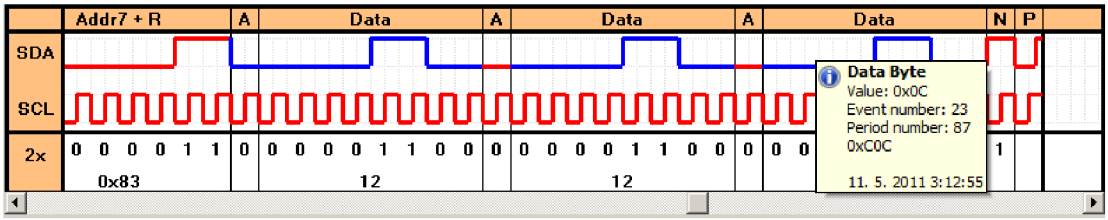


Obr. 7.14: Nastavenia zásuvného modulu DataViewer

7.4.2 SignalViewer

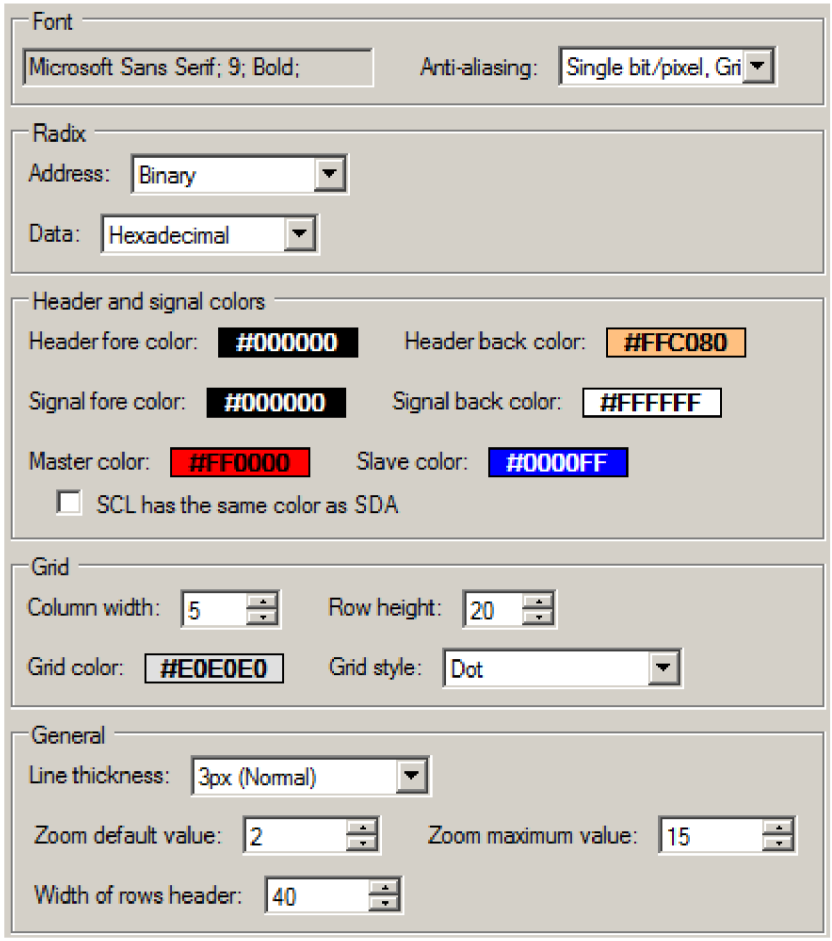
Tento zásuvný modul slúži na zobrazenie priebehu signálu protokolu I²C uloženého v dátovej vrstve programu.

Signál je vykresľovaný do pamäte ako bitmapový obrázok pomocou GDI+ (Graphics Device Interface) objektov. Na tento účel je napísaných niekoľko metód, ktoré počítajú rozmery a súradnice jednotlivých periód hodinového signálu a I²C udalostí, vrátane posunutia o záhlavie a hodnotu horizontálneho posuvníka. Z týchto údajov je potom poskladaný výsledný obrázok obsahujúci záhlavie, mriežku, popis signálu ako aj signál samotný. Po vykreslení celého signálu do pamäte sa týmto obrázkom prekreslí objekt odvodený od triedy `Panel` s nastaveným double bufferingom, čím sa užívateľovi zobrazí priebeh podobne ako je to na obr. 7.15. Po kliknutí na určitú časť vykresleného priebehu sa na základe rozmerov objektu a polohy kurzora pri kliknutí spätne spočíta, na ktorú udalosť bolo kliknuté a zobrazí sa detail. Vľavo dole je zobrazená aktuálna hodnota zväčšenia, ktorú je možné meniť z menu programu alebo klávesovou skratkou. Z kontextového menu je podobne ako v zásuvnom module DataViewer možné zmeniť globálne nastavenie pre zobrazenie adresy a dát.



Obr. 7.15: SignalViewer - predvolený zásuvný modul pre vykresľovanie signálu

V nastaveniach zásuvného modulu (obr. 7.16) má užívateľ možnosť zmeniť veľkosť a typ písma ako aj spôsob jeho vykreslenia, resp. typ anti-aliasingu. Samozrejmosťou je nastavenie farby pozadia aj popredia všetkých vykreslených objektov pre prípad, že by užívateľovi viacej vyhovovalo napr. čierne pozadie. Pri vykresľovaní je farebne rozlíšený signál vysielaný zariadením typu master od zariadenia typu slave. Keďže signál SCL generuje zariadenie master, je celý tento signál vykresľovaný vo farbe nastavenej pre zariadenie master. Toto nastavenie ale nemusí vyhovovať každému a v nastaveniach zásuvného modulu sa dá zmeniť tak aby bolo SDA a SCL vykresľované v rovnakej farbe. Je tiež možné vybrať hrúbku čiary signálu a štýl mriežky. Kompletný zoznam nastavení je vidieť z obr. 7.16.



Obr. 7.16: Nastavenia zásuvného modulu SignalViewer

7.4.3 DriverVCP

Jedná sa o predvolený zásuvný modul komunikujúci s navrhnutým prevodníkom I²C/USB pomocou virtuálneho sériového portu. Ak by táto komunikácia prebiehala v hlavnom vlákne programu, užívateľské rozhranie by počas nej prestalo reagovať a vzbudzovalo tak dojem, že program zamrzol. Z tohto dôvodu prebieha akákoľvek komunikácia s prevodníkom v novom paralelnom vlákne.

Všetky metódy používajú spoločný zámok pre prístup k zdieľaným zdrojom a pokiaľ tento zámok nie je voľný, radia sa požiadavky do fronty. Zároveň je pri aktívnej komunikácii nastavená vlastnosť `IsReady = false`, čím sa automaticky vyvolá udalosť `StatusChanged`. Ostatné časti programu tak majú hneď dve možnosti ako zistiť, či je prevodník možné používať alebo by mali počkať na dokončenie prebiehajúcej úlohy.

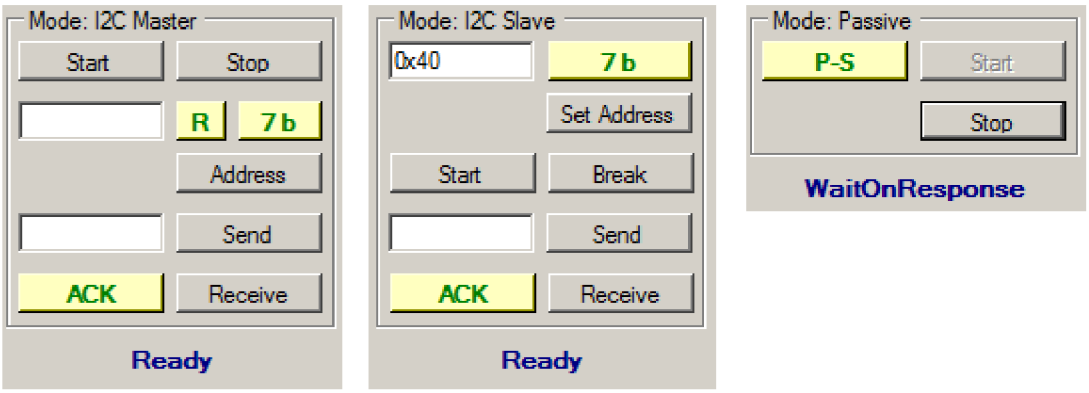
Režimy master a slave po volaní príslušnej metódy vytvoria nové vlákno, v ktorom odošlú príkaz a čakajú na odpoveď. Toto čakanie je realizované pomocou metódy `WaitOne()` s nastaveným timeoutom. Po prijatí dát z virtuálneho portu je vyvolaná udalosť `DataReceived`, v ktorej sa nastaví signál na odblokovanie metódy `WaitOne()`. Overí sa počet prijatých bytov, hodnota a ak sa jedná skutočne o odpoveď na odoslaný príkaz, pošle sa do dekodéra. Dekodér určí aká udalosť bola naozaj vyvolaná a na základe toho pridá záznam do dátovej vrstvy.

Pasívny režim funguje podobne, ale formát dát je odlišný a preto používa vlastný dekodér, ktorý vykonáva potrebné bitové operácie pre rozlíšenie dátových bitov od podmienok START/STOP. Na rozlíšenie konkrétnej udalosti potom využíva stavový automat, ktorý zároveň pridáva záznamy do dátovej vrstvy. Na rozdiel od predchádzajúceho prípadu beží čítanie v nekonečnom cykle, ktorý nie je ukončený po žiadnom timeoute ale pomocou externého signálu.

7.4.4 GeneralDevice

Predvolený zásuvný modul implementujúci rozhranie `IDevice`. Prostredníctvom metód a vlastností poskytovaných rozhraním `IDriver` komunikuje s prevodníkom a vystupuje ako zariadenie pripojené k I²C zbernici.

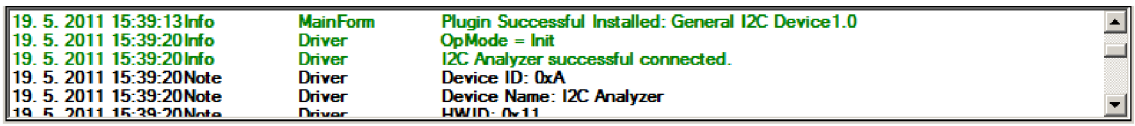
GeneralDevice je všeobecné zariadenie poskytujúce užívateľovi možnosť interaktívne komunikovať s akýmkoľvek iným zariadením na zbernici I²C v režime master alebo slave a zároveň spustiť alebo zastaviť monitorovanie zbernice v pasívnom režime. Tento zásuvný modul je vzhľadom na spôsob akým pracuje so zbernicou I²C taktiež vhodný na ladenie zariadení vyvíjaných pre túto zbernicu.



Obr. 7.17: GeneralDevice – predvolený zásuvný modul I²C zariadenia

7.4.5 MessageViewer

MessageViewer je zásuvný modul zobrazujúci hlásenia programu. Pokiaľ tento modul nie je dostupný, všetky hlásania sú zobrazované formou dialógových okien. Modul umožňuje farebne rozlíšiť chybové hlásenie, varovanie, poznámku a zobraziť toto hlásenie v spodnej časti programu spolu s časom a dátumom. Tieto hlásenia dokáže uložiť buď do textového súboru ako neformátovaný text alebo s nastavenými tabulátormi a farbou textu vo formáte RTF (Rich Text Format).



Obr. 7.18: MessageViewer – predvolený zásuvný modul na správu hlásení

8 ZÁVER

Cieľom diplomovej práce bolo zoznámiť sa so špecifikáciou zbernice I²C, navrhnúť a realizovať prevodník zbernice I²C/USB a obslužný software pre počítač.

Navrhnuté zariadenie umožňuje prácu v režime I²C Standard Mode a I²C Fast mode, s napäťovými úrovňami od 3,3 V približne do 7 V. Riadiaci mikroprocesor ATmega8L je k počítaču pripojený prostredníctvom prevodníka FT232RL, ktorého hlavnou výhodou je jednoduchá implementácia a dostupnosť ovládačov. Zbernica USB a analyzovaná zbernica I²C sú od seba galvanicky oddelené aby v prípade poruchy nemohlo dôjsť k poškodeniu počítača. Celý prevodník je napájaný z USB portu, takže nie je potrebné žiadne externé napájanie. Firmware pre mikroprocesor je napísaný v jazyku symbolických adries a umožňuje prevodníku interaktívne pracovať ako zariadenie typu master alebo slave. Taktiež má implementovanú funkciu pre pasívne monitorovanie zbernice. Z počítača je prevodník ovládaný prostredníctvom sady príkazov a ich vhodnou kombináciou je možné využiť všetky funkcie popísané v teoretickej časti práce.

Navrhnutý prevodník bol zrealizovaný a po nahratí konfiguračných údajov do EEPROM obvodu FT232RL bola odskúšaná komunikácia medzi počítačom a mikroprocesorom. Tá po pripojení externého kryštálu prebiehala bez problémov. Po dokončení obslužného softwaru pre počítač bola opakovane využitá funkcia na otestovanie spojenia, ktorá posiela a následne vyhodnocuje 10000 náhodne vygenerovaných echo rámcov, pričom výsledok bol vždy rovnaký, pri komunikácii prenosovou rýchlosťou 1 Mbaud boli všetky rámce prijaté správne.

Zároveň boli na navrhnutej testovacej doske odskúšané všetky tri pracovné režimy. Režim master a slave pracuje spoľahlivo pri rýchlostiach zbernice do 400 kHz. Pasívny režim môže s 8MHz kryštálom použitým v prototype sledovať zbernicu teoreticky do 400 kHz pokiaľ sa neprenáša paritný bit, s paritným bitom je to približne 363 kHz. Na základe týchto údajov by bolo vhodné v prípade výroby nového prevodníka použiť 16MHz kryštál v kombinácii s mikroprocesorom ATmega8A. Jedná sa o novšiu a pinovo kompatibilnú verziu stávajúceho mikroprocesora, ktorá ale bola v čase návrhu a výroby prototypu hardwarového prevodníka v ČR ťažko dostupná. Touto úpravou je možné zvýšiť prenosovú rýchlosť na dvojnásobok a výrazne tak zlepšiť vlastnosti prevodníka pri práci v pasívnom režime.

Pomocou osciloskopu bola overená strmosť hrán signálu na výstupe prevodníka pre zbernice 3,3 V a 5 V. Zo zmeraných priebehov uvedených v prílohe E vyplýva, že prevodník s rezervou spĺňa hodnoty uvedené v špecifikácii zbernice I²C.

LITERATÚRA

[1] **Embedded Systems Academy** [online]. c2010 [cit. 2010-12-21]. History of the I2C Bus. Dostupné z WWW: <http://www.esacademy.com/en/library/technical-articles-and-documents/miscellaneous/i2c-bus/general-introduction/history-of-the-i2c-bus.html>

[2] **NXP Semiconductors** . I2C-bus specification and user manual [online]. Rev. 03, 19 June 2007 [cit. 2010-12-21]. Dostupné z WWW: <www.nxp.com/documents/user_manual/UM10204.pdf>.

[3] **Future Technology Devices International Ltd.** FT232R USB UART IC [online]. Rev. 2.07, 29.7.2010 [cit. 2010-12-23]. Dostupné z WWW <http://www.ftdichip.com/Support/Documents/DataSheets/ICs/DS_FT232R.pdf>

[4] **Atmel Corporation.** ATmega8(L) [online]. Rev. 2486W, 24.2.2010 [cit. 2010-05-02]. Dostupné z WWW: <http://atmel.com/atmel/acrobat/doc2486.pdf>

[5] **Philips Semiconductors**. Bi-directional level shifter for I²C-bus and other systems [online]. 1997-08-04 [cit. 2010-12-22]. Dostupné z WWW: < http://ics.nxp.com/support/documents/interface/pdf/an97055.pdf>

[6] **Leshan Radio Company, Ltd.** L2N7002LT1 datasheet [online]. 16.7.2004 [cit. 2010-05-05]. Dostupné z WWW: <http://pdf1.alldatasheet.com/datasheet-pdf/view/71162/LRC/2N7002LT1.html>

[7] **Future Technology Devices International Ltd.** Aliasing VCP Baud Rates [online]. Rev. 1.0, 6.11.2009 [cit. 2010-05-02]. Dostupné z WWW: <http://www.ftdichip.com/Documents/AppNotes/AN_120_Aliasing_VCP_Baud_Rates.pdf>

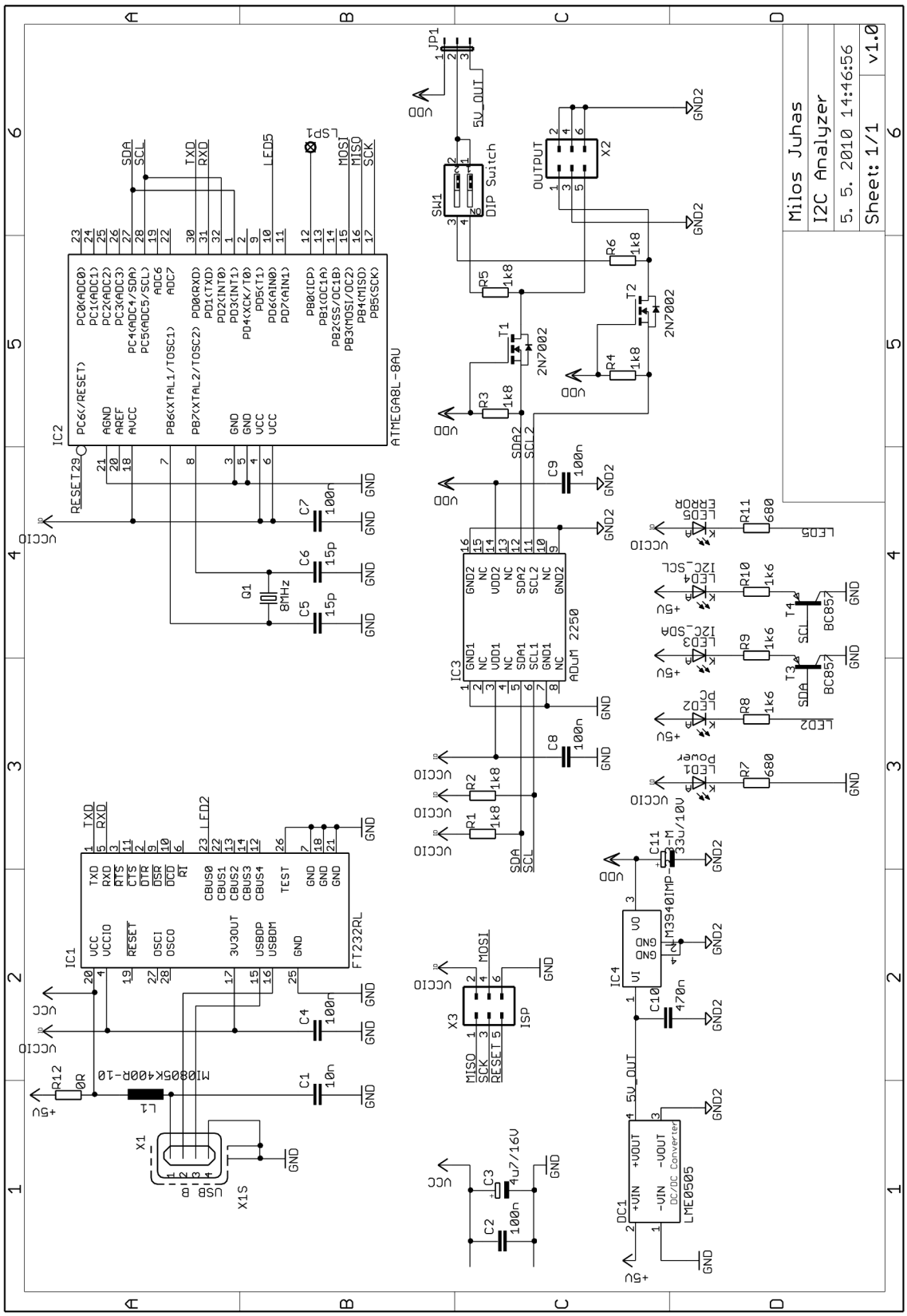
[8] **MSDN Library** [online]. c2011 [cit. 2010-05-18]. Assemblies in the Common Language Runtime. Dostupné z: <http://msdn.microsoft.com/en-us/library/k3677y81(v=VS.100).aspx>

ZOZNAM SKRATIEK

Skratka	Význam	Popis
ACK	Acknowledge	Potvrdzujúci bit alebo správa
CAN	Controller-Area Network	Sériová zbernica používaná v automobiloch
CDM	Combined Driber Model	Baliček kombinujúci dva typy ovládačov napr. k obvodu FT232RL
EEPROM	Electronically Erasable Read-Only Memory	Pamäť typu ROM s možnosťou elektického mazania
GDI+	Graphics Device Interface	Transformuje grafické objekty do výstupnách zariadení
HID	Human Interface Device	Kategória periférií pripojených k počítaču cez USB port
I^2C	Inter-integrated Circuit	Dvojvodičová sériová zbernica
LDO	Low Dropout	Stabilizátor napätia s malým úbytkom
LED	Light-Emitting Diode	Dióda emitujúca svetlo
LIN	Local Interconnect Network	Sériová zbernica používaná v automobiloch
SCL	Serial Clock	Vodič s hodinovým signálom
SDA	Serial Data	Dátový vodič zbernice I^2C
SPI	Serial Peripherila Interface	Sériová zbernica
TWBR	Two Wire Bitrate Register	Register pre nastavenie hodinového kmitočtu pre zariadenie master
TWI	Two Wire Interface	Sériová zbernica kompatibilná s I^2C
TWSI	Two Wire Serial Interface	Sériová zbernica kompatibilná s I^2C
UART	Universal Asynchronous Receiver/Transmitter	Univerzálne sériové rozhranie
UBRR	UART Baudreate Register	Register pre nastavenie rýchlosti prenosu
UPE	UART Parity Error	Príznak signalizujúci chybu v parite
USART	Universal Synchronous/Asynchronous Receiver/Transmitter	Univerzálne sériové rozhranie
USB	Universal Serial Bus	Univerzálna sériová zbernica používaná k pripojeniu zariadení k počítaču

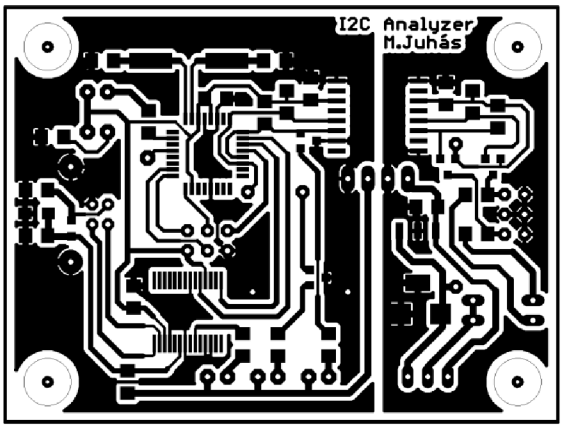
ZOZNAM PRÍLOH

A SCHÉMA ZAPOJENIA



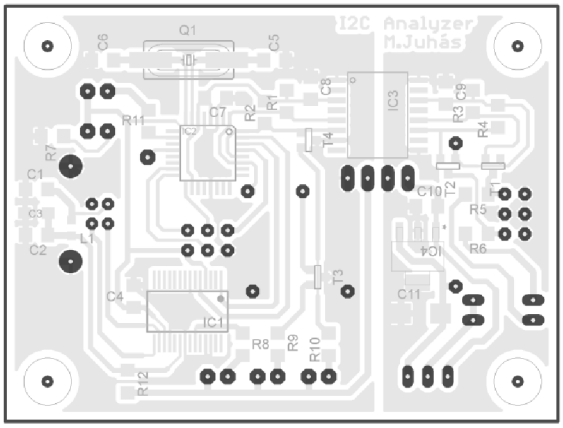
B DOSKA PLOŠNÉHO SPOJA

B.1 Neinvertovaná predloha DPS – bottom

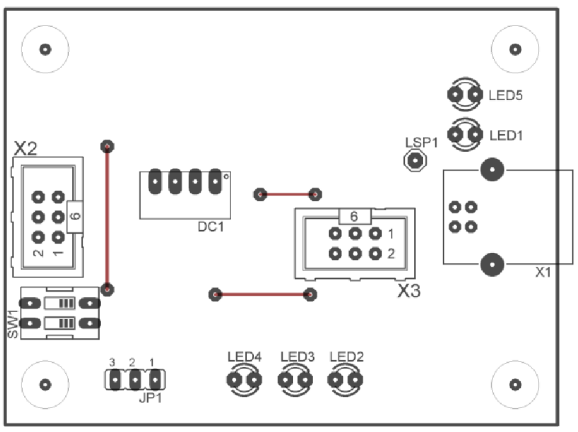


Rozmer 70,155 x 52,695 [mm], mierka M1:1

B.2 Osadzovací plán – bottom



B.3 Osadzovací plán – top



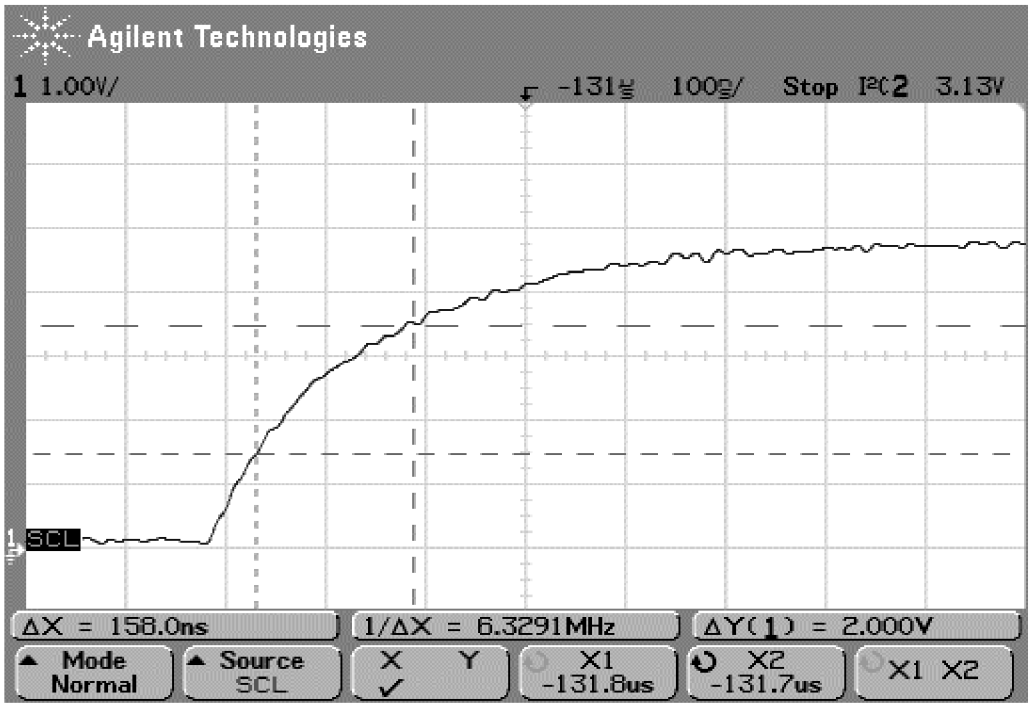
C ZOZNAM SÚČIASTOK

Označenie	Hodnota	Púzdro
R1, R2, R3, R4, R5, R6	1k8	R1206
R7, R11	680R	R1206
R8, R9, R10	1k6	R1206
R12	0R	R1206
C1	10n	C1206
C2, C4, C7, C8, C9	100n	C1206
C3	4u7/16V	Tantal A
C5, C6	15p	C1206
C10	470n	C1206
C11	33uF/16V	Tantal C
L1 (nahradená skratom)	MI0805K400R-10	SMD 0805
LED1	Red, 2mA	3 mm
LED2, LED3, LED4	Green, 2 mA	3 mm
LED5	Yellow, 2 mA	3 mm
Q1	8 MHz	HC49S
T1, T2	2N7002	SOT23
T3, T4	BC857C	SOT23
IC1	FTDI FT232RL	SSOT28
IC2	ATMEL ATmega8L-8AU	TQFP32
IC3	Analog ADuM 2250	16-SOIC Wide
IC4	LM3940IMP-3.3V	SOT223
DC1	MurataPowerSolution LME0505SC	SIP4
X1	USB-B PCB BW	USB-B do PCB
X2, X3		MLW06G
JP1		S1G2
SW1		DIP2

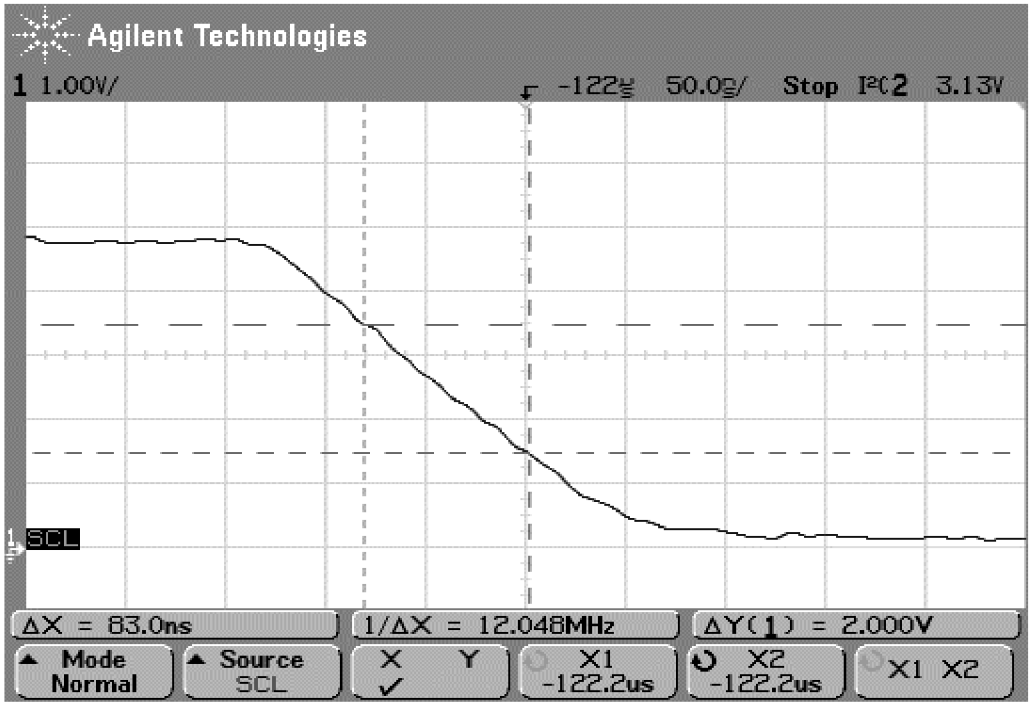
D PRÍKAZY PRE MIKROPROCESOR

Typ	Kód	Príkaz	Popis	Parameter
MASTER	0x11	Send START	Odošle podmienku START	-
	0x12	Send STOP	Odošle podmienku STOP	-
	0x1b	Send DATA	Odošle dáta alebo adresu	Dáta
	0x14	Receive DATA , ACK	Prijme dáta a potvrdí ich ACK	-
	0x15	Receive DATA, NACK	Prijme dáta a potvrdí ich NACK	-
	0x16	Get TWBR	Načítanie hodnoty registra TWBR (určuje periódu SCL)	-
	0x1e	Set TWBR	Nastavenie registra TWBR	TWBR
SLAVE	0x11	Enable	Povolí čakanie na slave adresu	-
	0x12	Disable	Zastaví čakanie na slave adresu	-
	0x1b	Send DATA	Odošle dáta alebo adresu	Dáta (8b)
	0x14	Receive DATA , ACK	Prijme dáta a potvrdí ich ACK	-
	0x15	Receive DATA, NACK	Prijme dáta a potvrdí ich NACK	-
	0x1e	Set ADDR B1	Nastaví adresu (7 b / 10 b High)	Adresa (8b)
	0x1f	Set ADDR B2	Nastaví adresu (10 b Low)	Adresa (8b)
PASSIVE	0x11	Start after START Condition	Spustenie vzorkovania podmienkou START	-
	0x12	Start after STOP Condition	Spustenie vzorkovania za podmienkou STOP	-
GENERAL	0x00	Dummy frame		-
	0x08	Echo	Echo	DATA
	0x01	Get Device ID	Načítanie ID zariadenia	-
	0x02	Get HWID	Načítanie verzie HW	-
	0x03	Get FWID	Načítanie verzie FW	-
	0x04	Get Device Name	Načítanie mena zariadenia	-
	0x05	Get System Clock	Načítanie frekvencie MCU	-
	0x06	Get Operation Mode	Načítanie aktuálneho režimu	-
	0x0e	Set Operation Mode	Nastavenie nového režimu	OP Mode ID

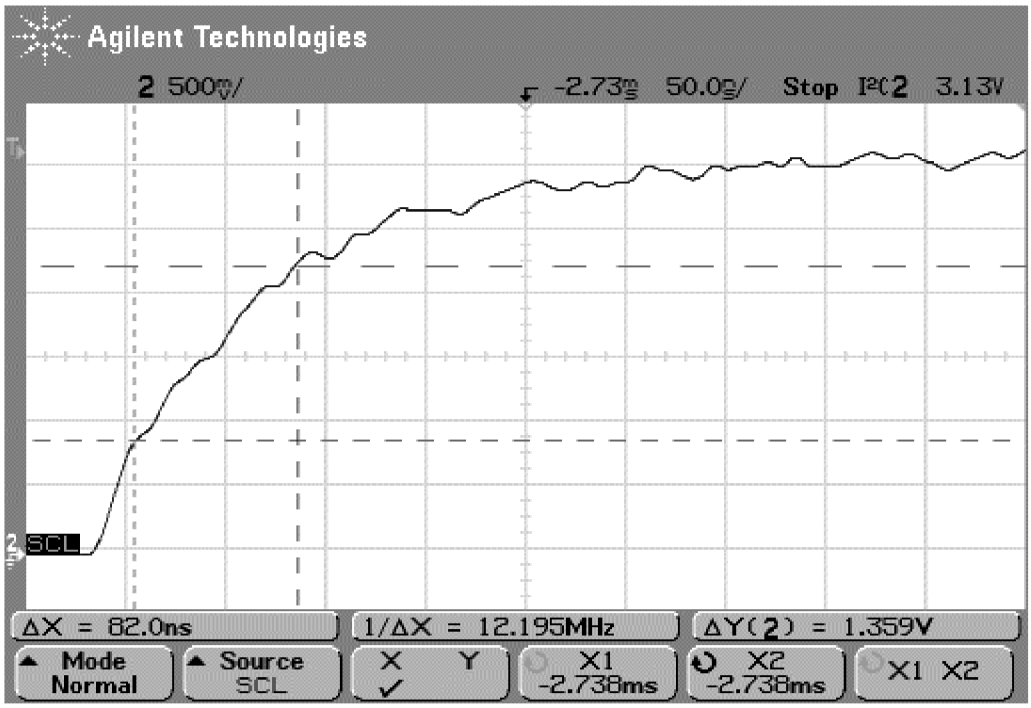
E HRANY GENEROVANÉHO SIGNÁLU



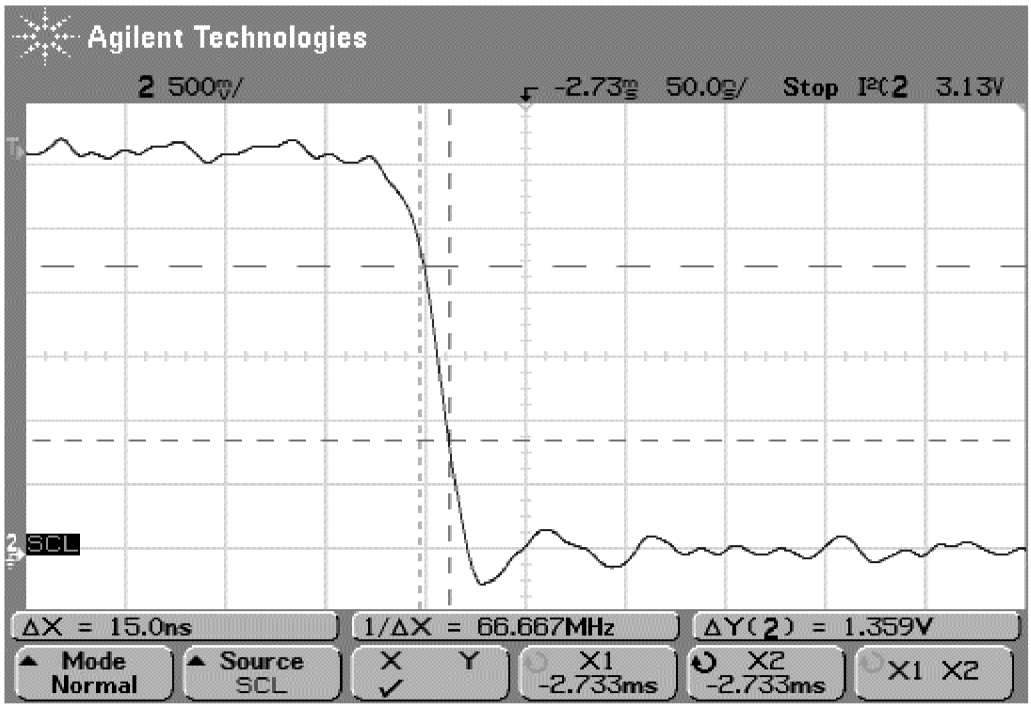
Nábežná hrana SCL (U = 5 V), t_r = 158 ns



Dobežná hrana SCL (U = 5 V), t_f = 83 ns

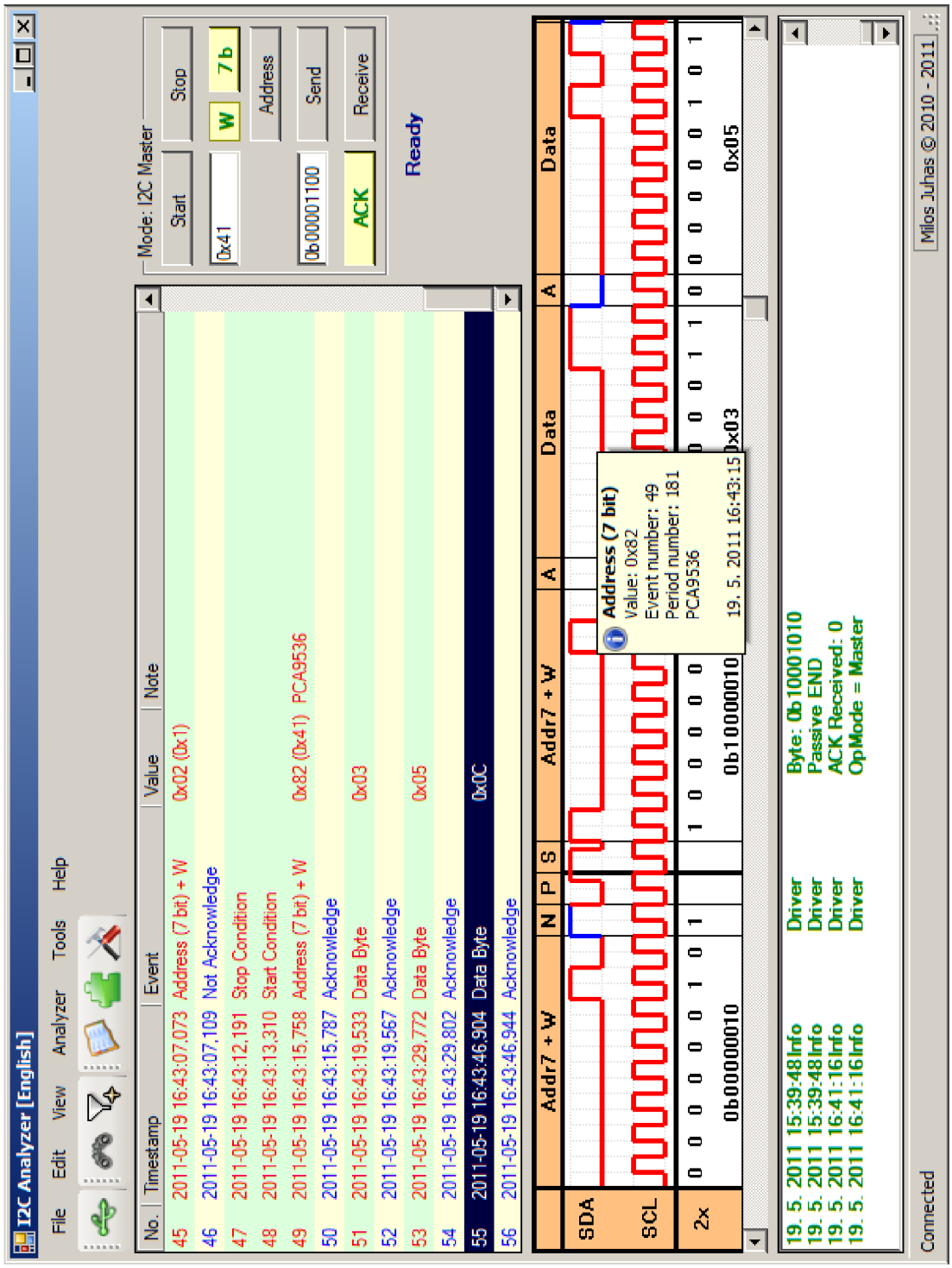


Nábežná hrana SCL (U = 3,3 V), t_r = 82 ns

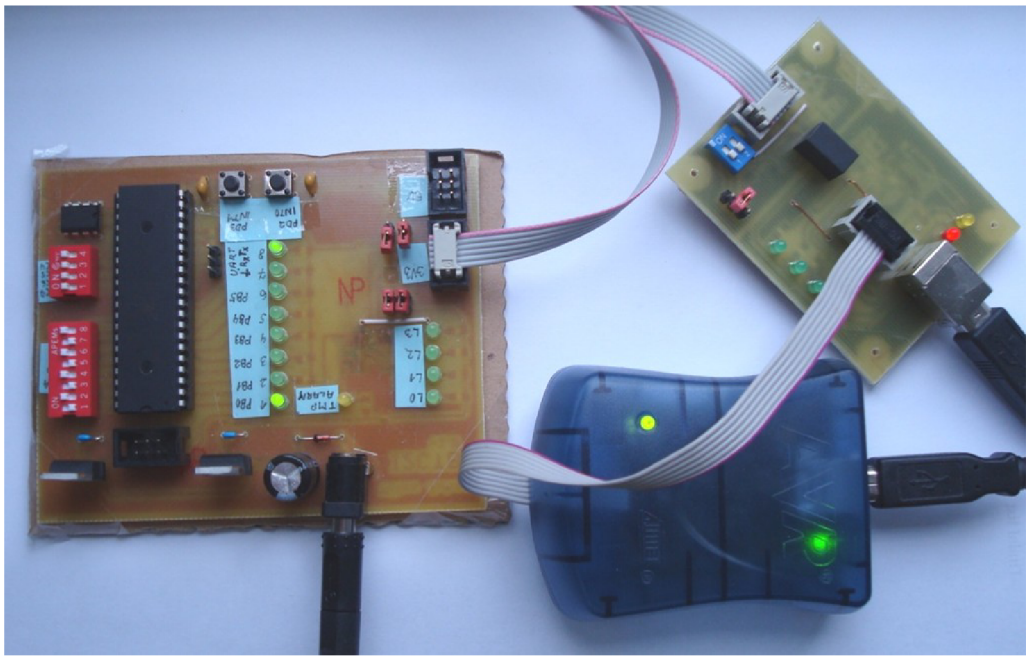


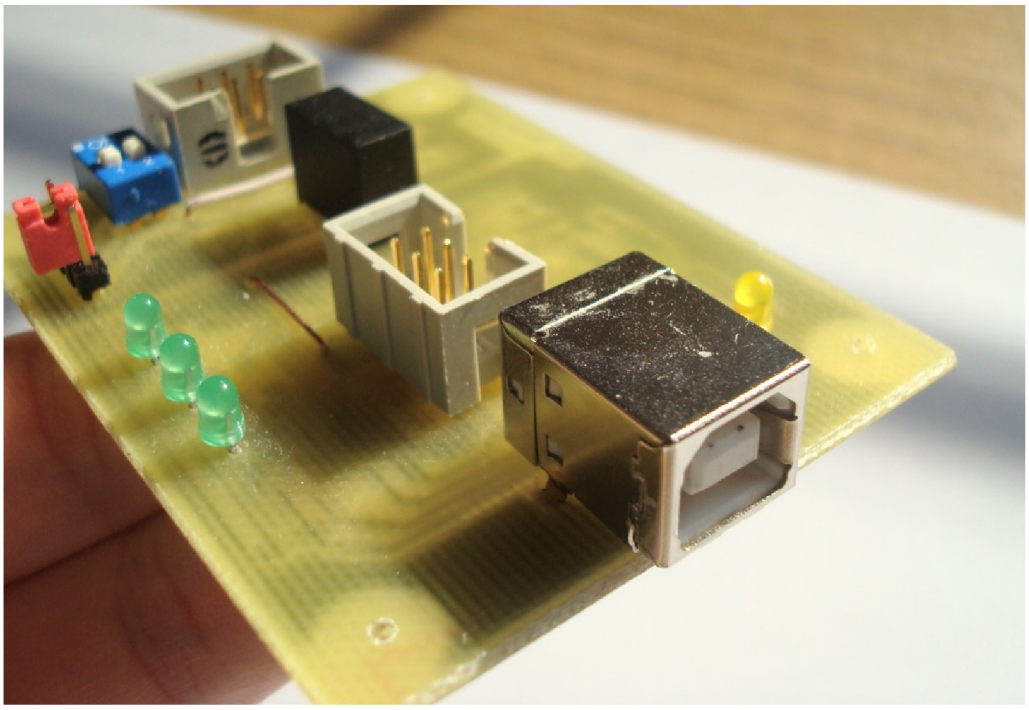
Dobežná hrana SCL (U = 3,3 V), t_f = 15 ns

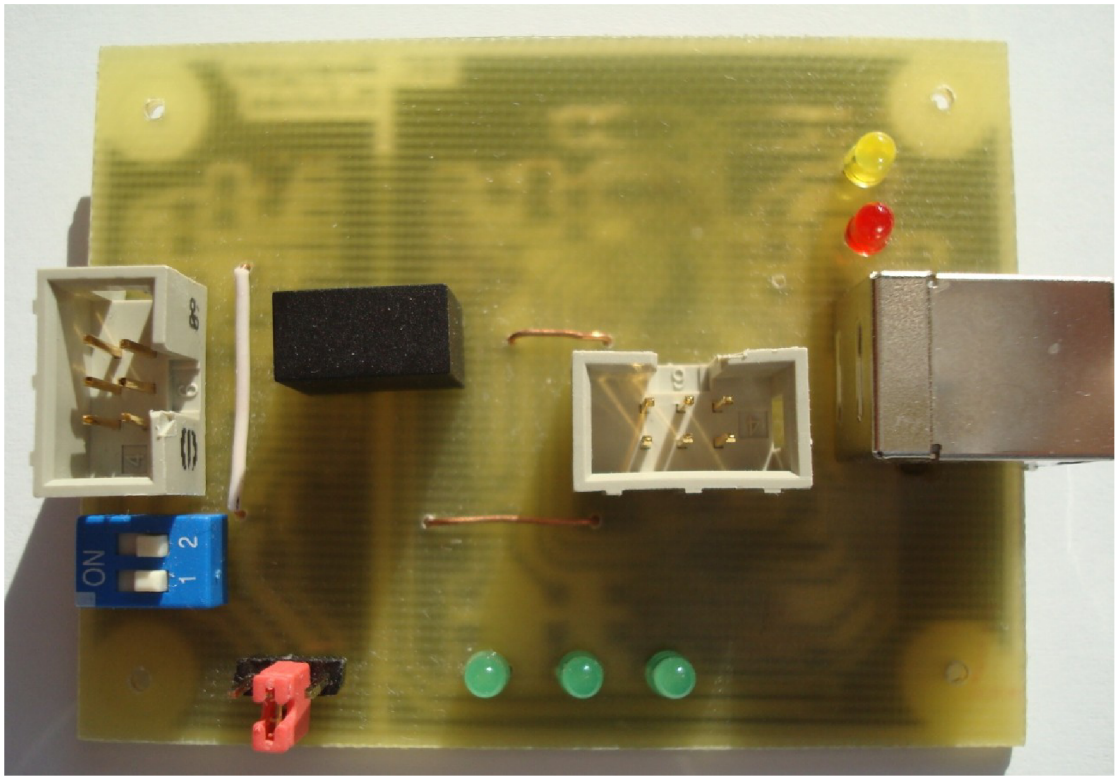
F PROSTREDIE PROGRAMU

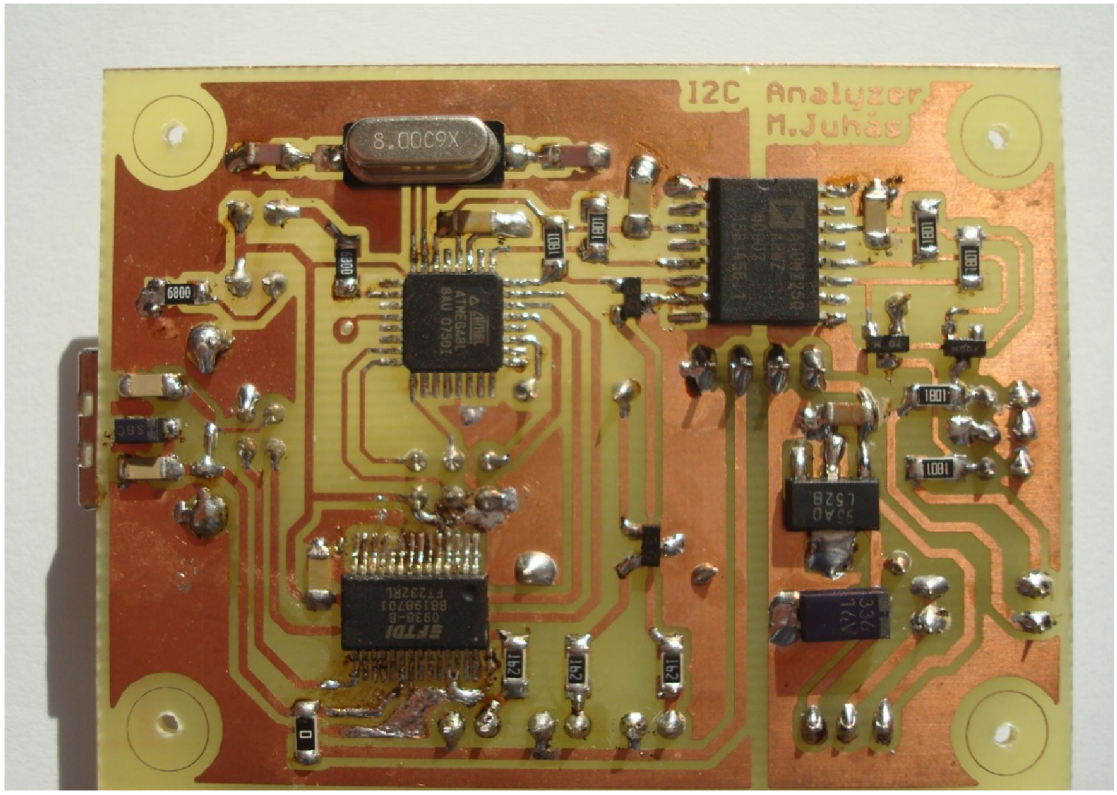


G FOTODOKUMENTÁCIA






ON
1
2


I2C Analyzer
M.Juhás
8.00C9X
ATMEGA8L
FTDI