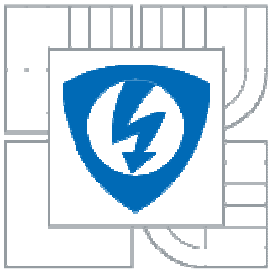
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF CONTROL AND INSTRUMENTATION

PROGRAMOVÉ VYBAVENÍ SNÍMAČE TLAKU S PROTOKOLEM HART

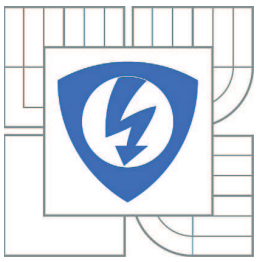
FIRMWARE FOR A PRESSURE SENSOR WITH HART INTERFACE

BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE — ONDŘEJ ČOŽÍK
AUTHOR

VEDOUCÍ PRÁCE — Ing. PETR FIEDLER, Ph.D.
SUPERVISOR

BRNO 2010



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav automatizace a měřicí techniky

Bakalářská práce

bakalářský studijní obor
Automatizační a měřicí technika

Student:	Ondřej Čožík	***ID:***	106398
Ročník:	3	***Akademický rok:***	2009/2010

NÁZEV TÉMATU:

Programové vybavení snímače tlaku s protokolem HART

POKYNY PRO VYPRACOVÁNÍ:

Navrhněte a naprogramujte programové vybavení, které umožní komunikaci senzoru tlaku s nadřazeným systémem pomocí protokolu HART. Realizované programové vybavení senzoru musí být schopno měřit požadovanou veličinu, nastavovat parametry a kalibrační konstanty a komunikovat pomocí standardu HART. Odladěné zdrojové programy detailně okomentujte. Celou práci včetně příloh dodejte také na CD.

DOPORUČENÁ LITERATURA:

ČSN, dokumentace k snímačům zadavatele

Termín zadání:	8.2.2010	***Termín odevzdání:***	31.5.2010

Vedoucí práce: Ing. Petr Fiedler, Ph.D.

prof. Ing. Pavel Jura, CSc.
Předseda oborové rady

Abstrakt

Tato bakalářská práce se zabývá návrhem programového vybavení kapacitního snímače tlaku s komunikačním protokolem HART. Nejprve je stručně popsán protokol HART, tedy princip komunikace, struktura rámce a možnosti připojení přístrojů ke sběrnici HART. Dále je popisován samotný kapacitní snímač a způsob zpracovávání měřených dat spolu s komunikací snímače pomocí protokolu HART. V poslední části práce jsou popisovány jednotlivé části praktického návrhu programu pro tento snímač, jako jsou způsob vyčítání hodnot z A/D převodníku, zápis/čtení dat z externí nebo vnitřní EEPROM či výpočet hodnoty měřeného tlaku, její teplotní kompenzace a odpovídající nastavení proudové smyčky pomocí D/A převodníku.

Abstract

The bachelor thesis deals with draft of firmware for a capacitive sensor with HART communication protocol. At first, it briefly describes the HART protocol (principles of communication, frame structure and connectivity devices on the HART bus). Subsequently attention is paid to capacitive sensor description, method of data processing and communication method of the sensor with HART interface. The final part depicts individual parts of the firmware for the capacitive sensor (such as the method of values reading from A/D converter, data reading/writing from/to external or internal EEPROM or calculation of the measured pressure value, its temperature compensation and corresponding current loop, using D/A converter.

Klíčová slova

HART, proudová smyčka, PIC16F886, AD7745, měření tlaku, měření teploty.

Keywords

HART, current loop, PIC16F886, AD7745, pressure measurement, temperature measurement.

Bibliografická citace díla

ČOŽÍK, O. *Programové vybavení snímače tlaku s protokolem HART*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2010. 59s. Vedoucí bakalářské práce Ing. Petr Fiedler, Ph.D.

Prohlášení autora o původnosti díla

„Prohlašuji, že svou bakalářskou práci na téma Programové vybavení snímače tlaku s protokolem HART jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.“

V Brně dne: 25. května 2010 ……

podpis autora

Poděkování

Děkuji vedoucímu bakalářské práce Ing. Petru Fiedlerovi, Ph.D. za odbornou pomoc při zpracování mé bakalářské práce. Velmi děkuji také Ing. Jaroslavu Kadlecovi, Ph.D, se kterým jsem při tvorbě bakalářské práce úzce spolupracoval. Bakalářská práce je vytvořena dle zadání firmy BD SENSORS s.r.o, které děkuji zejména za odbornou pomoc, poskytnutí podkladů a prostorů pro tvorbu práce.

Obsah

1. Úvod do problematiky

Cílem bakalářské práce je navrhnout programové vybavení pro kapacitní snímač tlaku komunikující pomocí protokolu HART. Základem celého snímače je mikroprocesor PIC16F886, pro který je potřeba navrhnout řízení a zmiňovaný komunikační protokol HART. Mimo univerzální příkazy protokolu, které musí být implementovány, jsou zde také příkazy, kterými lze měnit jednotlivá nastavení snímače, jako jsou rozsah, horní a dolní mez rozsahu, jednotky měření a spousty dalších nastavení. Snímač podporuje také tzv. technologické příkazy, které slouží jak pro čtení či zápis dat z / do EEPROM snímače, tak i pro čtení právě zpracovávaných dat při měření z operační paměti snímače.

Před samotným řešením problému implementace protokolu HART do mikroprocesoru bych ale chtěl alespoň nastínit samotnou funkci protokolu. Tedy takový základní popis protokolu jako takového. Bude popsáno od zapojení samotného přístroje podporující tento protokol, přes popis samotného rámce a principu komunikace až po stručné shrnutí podporovaných příkazů a jejich možností.

V další části práce se zaměřím na problematiku samotné implementace protokolu do mikroprocesoru PIC16F886. Na vývojových diagramech bude přesně ukázán postup při komunikaci (příjem/odesílání dat) protokolem HART, ale také zmíním sériové rozhraní IIC, které je používáno jak pro komunikaci mezi mikroprocesorem a EEPROM, tak i pro vyčítání hodnoty kapacity čidla pro snímání tlaku z A/D převodníku.

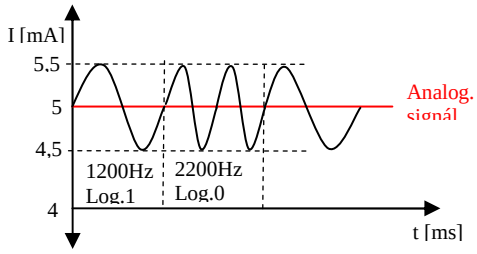
1.1 Protokol HART (Highway Addressable Remote Transducer)

Protokol HART je určen pro obousměrnou číslicovou komunikaci mezi zařízením (většinou se jedná o inteligentní snímače) a nadřazeným systémem pro kontrolu či monitorování. Nadřízeným systémem může být například obecně počítač nebo ruční terminál, nebo řídící systém mající na starosti chod celé továrny. Protože většina snímačů dnes má výstup ve formě dvouvodičové proudové smyčky 4-20mA, je tento protokol navržen tak, aby pracoval současně při tomto analogovém signálu a není potřeba tak vytvářet další komunikační trasu. Po původních vodičích proudové smyčky tak lze vést jak původní analogový, tak i číslicový signál. Tímto způsobem je tedy možné odesílat měřenou hodnotu v analogové i digitální formě a vyloučit tak chybu vzniklou převodem z analogového signálu na číslicový a zpět. Můžeme si také po stávající lince nechat zasílat informace o jiných veličinách, které je snímač schopen změřit. Pro případ snímače tlaku to nemusí být jenom samotný tlak, ale

například i teplota. V současnosti se protokol hojně využívá zejména pro inteligentní snímače.

1.1.1 Popis činnosti

Přenos číslicového signálu po analogové lince je založen na superpozici frekvenčně klíčovaných signálů, tzv. FSK(Frequency Shift Keying) na původní analogový signál 4-20mA. Jinými slovy pracuje na základě přepínání frekvence a princip je založen na standardu Bell 202. Logické hodnoty jsou následující. Logické 1(H) odpovídá signál o frekvenci 1200 Hz a logické 0(L) odpovídá signál o frekvenci 2200 Hz. Superponovaný signál na analogovém signálu dosahuje hodnot amplitudy přibližně 0,5 mA, ale důležité je, že střední hodnota signálu je vždy nulová a tím pádem nijak neovlivňuje hodnotu proudové smyčky.



Obrázek 1: Ukázka logických úrovní na analogovém signálu 4-20mA

Na Obrázek 1 můžeme vidět klasický analogový signál, na němž je superponován signál nesoucí logické hodnoty 0(L) a 1(H). Základní přenosová rychlost protokolu je 1200 bit/s, což je ve srovnání s jinými protokoly docela malá hodnota, ale i tak dostačující.

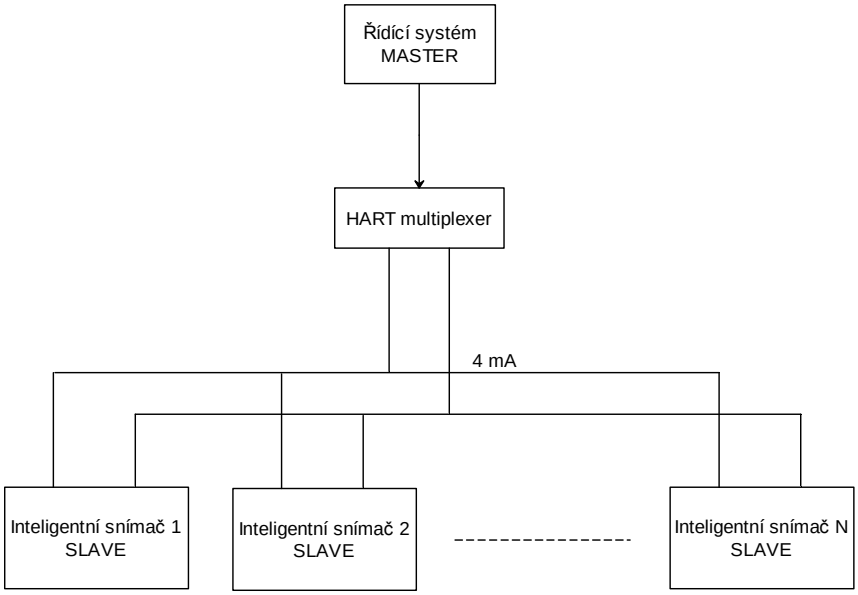
Samotný číslicový přenos je organizován systémem Master-Slave. V tomto systému může Slave zařízení komunikovat pouze tehdy, je-li vyzváno Master zařízením. V celém zapojení se mohou vyskytovat dvě zařízení Master. Jeden z nich je obecně řídící systém nebo počítač (tzv.Primary Master) a lze použít ještě druhotné zařízení pro komunikaci, jako jsou například ruční terminály, nebo laptopy (tzv.Secondary Master). Ruční terminály se často používají pro seřizování snímačů a kontrolu komunikace systému. Všechna tato zařízení obsahují Hart modem.

1.1.2 Možnosti připojení HART zařízení

Pro komunikaci můžeme použít jak sběrnicovou, tak hvězdicovou topologii.

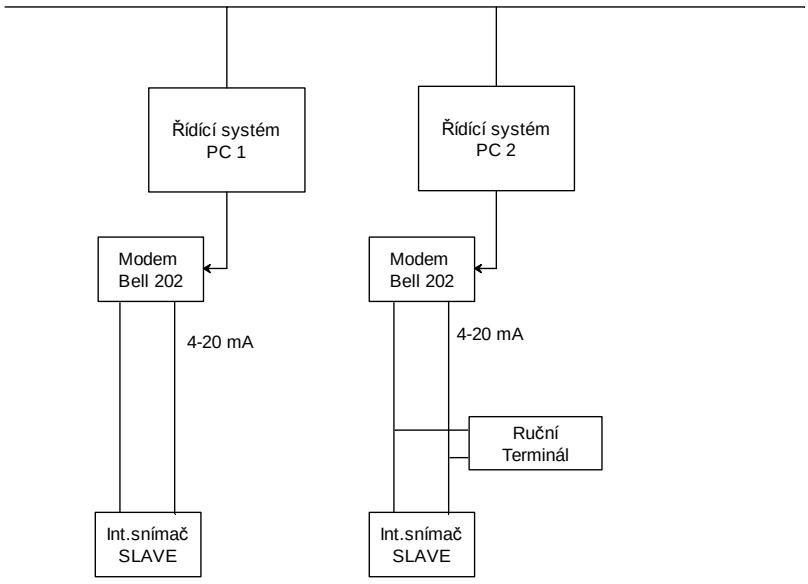
Sběrnicová topologie se vyznačuje tím, že všechna Slave zařízení jsou připojena paralelně na jedno vedení (proudovou smyčku) a každé zařízení má svou specifickou adresu, pomocí které jej může Master vyzvat ke komunikaci. Sběrnicové zapojení má však tu nevýhodu, že zařízení mohou komunikovat pouze číslicově a

hodnota proudové smyčky je jen udržována na minimální konstantní hodnotě(4mA) a jednotlivá zařízení jsou postupně vyzívána ke komunikaci. Toto uspořádání je známo také jako „Multidrop“.



Obrázek2: Zapojení zařízení v topologii sběrnice (Multidrop)

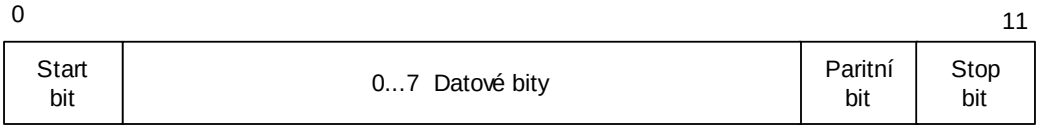
Při hvězdicové topologii komunikuje Master zařízení se všemi Slave zařízením po samostatném vedení (proudové smyčce), což dovoluje mírné zvýšení rychlosti komunikace, použijeme-li tzv.Burst režim. Tento režim se používá výhradně pro komunikaci pouze s jedním Slave zařízením.



Obrázek 3: Zapojení zařízení v topologii hvězda (příklad)

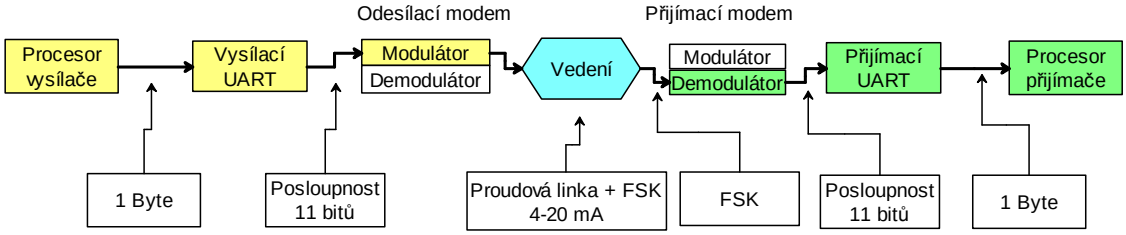
1.1.3 Struktura rámce protokolu HART

Protokol HART odesílá a přijímá data ve formě rámců. Každý rámec se skládá ze sedmi částí. Než popíši strukturu rámce protokolu, je dobré vědět, z čeho se každá část rámce skládá. Každá část je složená z jednoho nebo několika bytů. Ke každému bytu jsou jako u většiny ostatních sériových protokolů přidávány k datovým bitům ještě start bit, parity bit a stop bit tak, jak je ukázáno na Obrázek 4.



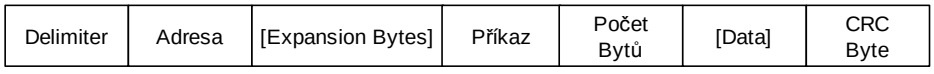
Obrázek 4: Struktura jednoho bytu v protokolu HART

Na Obrázek 5 je blokově zobrazen průběh odeslání jednoho bytu od vysílacího procesoru, tedy od vysílače až po přijímací procesor. Chce-li vysílací procesor odeslat datový byte pomocí HART protokolu, odešle byte do zásobníku pro odeslání rozhraní UART. V tomto rozhraní se „přibalí“ k datovému bytu další tři bity, jak je ukázáno na Obrázek 4, což jsou start bit, lichý paritní bit a stop bit. Rozhraní UART odešle tuto zprávu do modulátoru, kde se na proudovou linku superponuje číslicový signál a tak se přivede až k demodulátoru, který jej převede zpět na 11 bitovou zprávu a předá dále do přijímacího registru UART, odkud si již přijímací procesor vyzvedne jeden byte odesílaný vysílacím procesorem.



Obrázek 5: Zobrazení průběhu odeslání jednoho bytu pomocí protokolu HART

Teď se vrátím zpět k rámci protokolu. Jak jsem již řekl, protokol se skládá ze sedmi částí, jak ukazuje Obrázek 6. K tomuto rámci se ovšem ještě přidává jedna část a to Preambule, která slouží obzvláště k synchronizaci.



Obrázek 6: Formát rámce protokolu HART

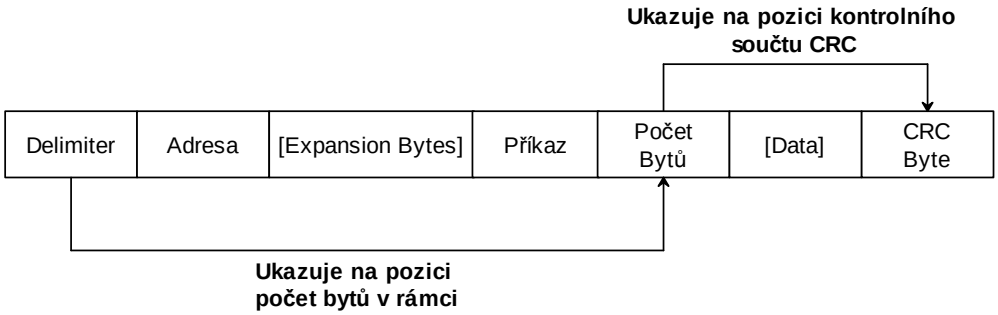
1.1.3.1 Popis jednotlivých částí rámce protokolu HART

Preambule

Obsahuje 3 nebo více hodnot 0xff a slouží pro synchronizaci přijímacího modemu.

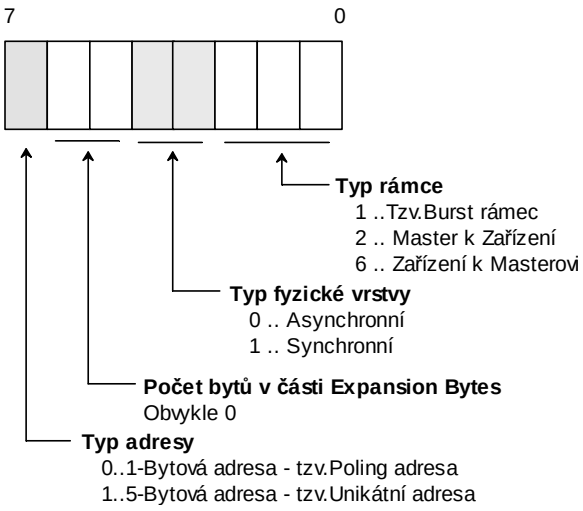
Delimiter

Delimiter indikuje začátek rámce a ukazuje na pozici, kde se v rámci nachází zpráva o počtu datových bytů „Počet bytů“. To zjistíme tak, že v delimiteru dostaneme informaci o počtu adresových bytů (krátká nebo dlouhá adresa) a tím pádem víme, který byte bude obsahovat údaj o počtu datových bytů. Z této informace jsme schopni říct, kde bude rámec končit, jelikož za datové byty se přidává do rámce už jen jeden byte kontrolního součtu CRC (viz Obrázek 7).



Obrázek 7: Zobrazení závislosti pozic jednotlivých částí rámce protokolu HART

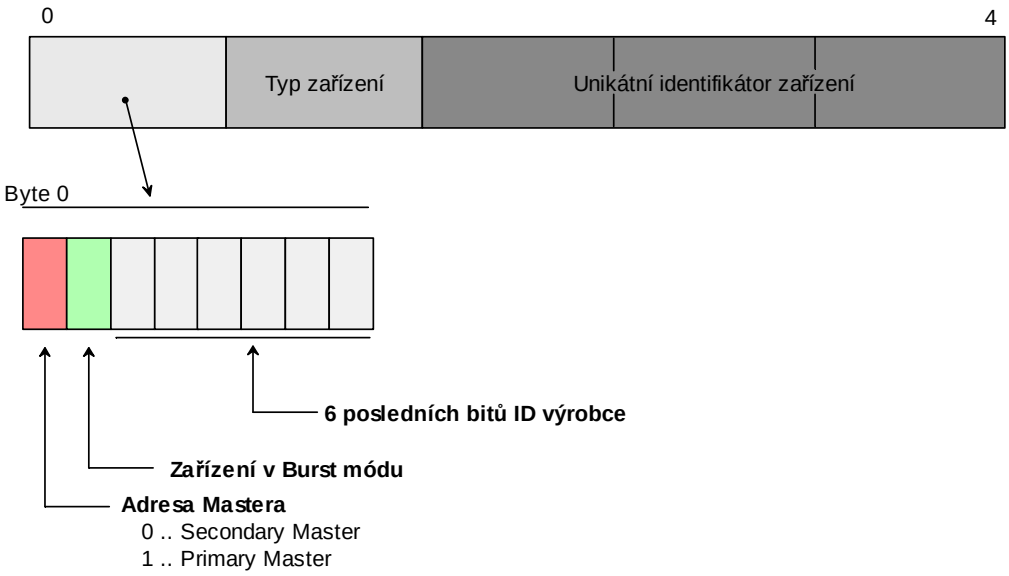
Jak ukazuje Obrázek 8, nejvyšší bit delimiteru nám říká, jestli bude následovat krátká jednobytová adresa, nebo zda budeme přijímat unikátní pětibytovou adresu. Podle dalších dvou bitů delimiteru zjistíme, zda bude rámec obsahovat nějaké byty v poli Expansion bytes. Toto pole pro implementovanou revizi protokolu HART je vždy nulové.Bity 3 a 4 říkají, o jaký typ fyzické vrstvy se jedná. Pro asynchronní FSK volíme 0. Zbývající 3 bity<3;0> delimiteru označují, o jakou komunikaci se jedná. Jsou tři možnosti a to, že komunikujeme v Burst-módu(režim, při kterém je zapojen Master přímo k Slave zařízení), v režimu, kdy odesílá data Master k Slave zařízení a opačně. První režim není moc rozšířen, nejčastěji se používají režimy Master k Slave zařízení a Slave zařízení k Masterovi(hodnoty 2 nebo 6).



Obrázek 8: Význam jednotlivých bitů v delimiteru

Adresa

Protokol podporuje dlouhou 5-ti bytovou tzv. unikátní, nebo krátkou 1bytovou tzv. Polling adresu. Význam jednotlivých bitů dlouhé a krátké adresy je na Obrázek 9 a Obrázek 10.

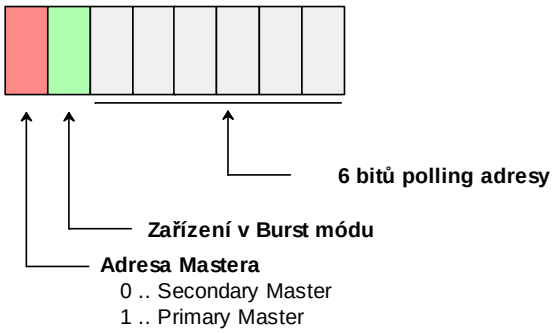


Obrázek 9: Význam jednotlivých bitů dlouhé adresy

Hodnota „Typ zařízení“ je dána výrobcem, kde každý výrobce má své ID, ze kterého použije posledních 6 bitů a vkládá je do prvního bytu dlouhé adresy, jak je ukázáno na Obrázek 9. Jak je popsáno výše, k proudové smyčce můžou být připojeny až 2 zařízení v režimu Master. Se kterým Master zařízením komunikujeme, určíme z prvního bitu v prvním bytu dlouhé adresy.

Unikátní identifikátor zařízení je obdoba sériovému číslu. Každé zařízení se stejným kódem „Typ zařízení“ musí mít jiný unikátní identifikátor.

V HART protokolu je možné, aby Master odeslal příkaz ke všem zařízením najednou. Pro tuto možnost využije tzv. Broadcast adresu, která má místo unikátní identifikace 38 nulových bitů. Tato adresa se smí používat pouze pro servisní účely, pro příkazy, kde není žádná odpověď nebo kde víme, že odpoví maximálně jedno zařízení.



Obrázek 10: Význam jednotlivých bitů polling adresy(krátké adresy)

Na Obrázek 10 je zobrazena krátká, 1-bytová adresa. Tato adresa se nesmí používat, je-li zařízení ve sběrnicovém zapojení, tzv.Multi-drroped zapojení. Jak je vidět, identifikaci zařízení tvoří pouze 6ti bitová adresa.

Expansion bytes

Pole je 0-3 byty dlouhé a jeho délka je dána v delimiteru. Obvykle je jeho délka nulová, tedy z rámce se vynechává. Jsou-li ale obsaženy neznámé byty, zařízení nesmí odpovídat.

Příkaz

Jeden byte obsahující číslo příkazu, který se má vykonávat. Protokol HART má několik typů příkazů. Příkazy 0-22 jsou univerzální příkazy a měly by být implementovány ve všech HART zařízeních. Příkazy 32-121 jsou obecné. Tyto příkazy nemusí být implementovány a příkazy 128-253 jsou uživatelské příkazy. Stručný seznam příkazů je v tabulce 1.

Význam jednotlivých příkazů protokolu HART			
číslo přík.	**Význam příkazu**	**číslo přík.**	**Význam příkazu**
0	Čtení unikátního identifikátoru	**33**	Čtení proměnných ze zařízení
1	Čtení primární proměnné	**34**	Zapsání tlumení Prim.proměnné
2	Hodnota proud.smyčky a procenta rozsahu	**35**	Zapsání rozsahu primární proměnné
3	Čtení dynamických proměnných a procent rozsahu	**36**	Nastavení primární proměnné jako horní limit
4	Rezervováno	**37**	Nastavení primární proměnné jako dolní limit
5	Rezervováno	**38**	Vymazání čítače změn
6	Zapsání krátké(polling) adresy	**39**	EEPROM kontrola(nedoporučováno)
7	Čtení konfigurace proud.smyčky	**40**	Povolení/zakázání proudové smyčky
8	Čtení klasifikace dynam.proměnných	**41**	Vykonání testu zařízení
9	Čtení proměnných se statusem	**42**	Vykonání resetu zařízení
10	Není	**43**	Nastavení primární proměnné jako nula
11	Čtení unik.identifikátoru spojeného s tagem	**44**	Zapsání jednotek prim.proměnné
12	Čtení zprávy	**45**	Nastavení proudu smyčky na spodní hodnotu
13	Čtení tagu, deskriptoru a datumu	**46**	Nastavení proudu smyčky na zadanou hodnotu
14	Čtení Primární prom.- informace snímače	**47**	Zapsání přenosové funkce pro prim.proměnnou
15	Čtení informací o zařízení	**48**	Čtení dodatečného statusu zařízení
16	Čtení konečného výrobního čísla		
17	Zapsání zprávy		
18	Zapsání tagu, deskriptoru a datumu		
19	Zapsání konečného výrobního čísla		
20	Čtení dlouhého tagu		
21	Čtení unik.ID spojeného s dlouhým tagem		
22	Zapsání dlouhého tagu		

Tabulka 1: Význam příkazů protokolu HART

Počet bytů

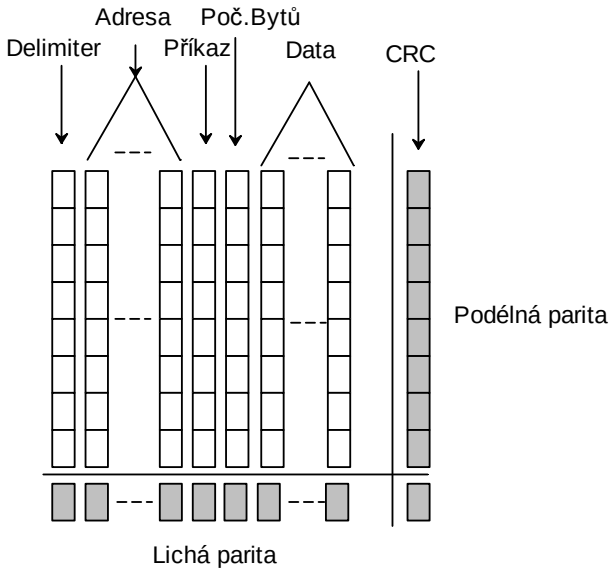
Pole je také jeden byte dlouhé a obsahuje počet bytů obsažených mezi poli „Počet bytů“ a „Kontrolní součet CRC“. Jsou dovoleny hodnoty 0-255.

Data

Obsahuje data pro přenos mezi zařízeními. Pole je volitelné. Do tohoto pole patří také Status, který se skládá ze dvou bytů a je odesílán pouze při komunikaci Slave zařízení k Master zařízení. První byte indikuje případnou chybu komunikace a druhý stav zařízení.

Kontrolní součet CRC

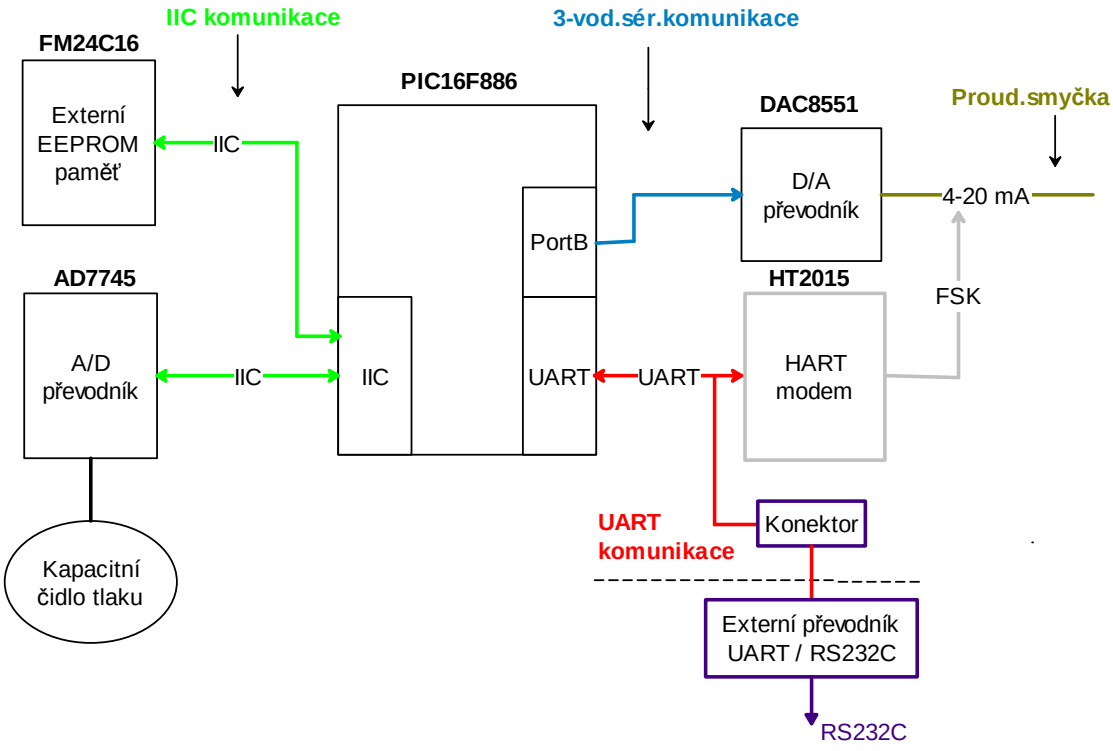
Kontrolní součet obsahuje výsledek funkce X-OR všech bytů od delimiteru až po poslední datový byte. Tuto kontrolu nazýváme „podélná parita“. Kromě této kontroly parity obsahuje HART protokol ještě jednu kontrolu liché parity a to při příjmu / vysílání jednotlivých bytů. Ukázka kontroly obou parit je na Obrázek 11.



Obrázek 11: Paritní kontrola komunikace

1.2 Základní popis snímače tlaku

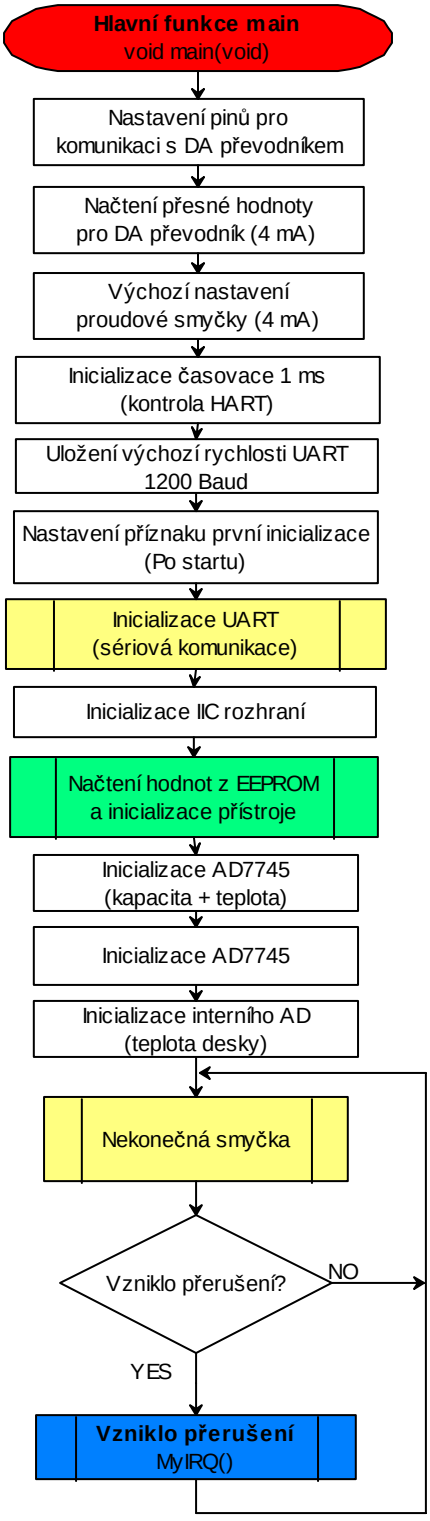
Funkci celého snímače řídí mikroprocesor PIC16F886. K mikroprocesoru je připojena externí EEPROM, ve které jsou uloženy konstanty pro měření, různá nastavení mikroprocesoru i ostatních obvodů, apod. Dále je připojen AD převodník typu AD7745, který převádí měřenou kapacitu na digitální hodnotu, ze které se posléze počítá samotný tlak. K A/D převodníku je tedy připojeno čidlo, které mění svou kapacitu se změnou tlaku. EEPROM i A/D převodník komunikují s mikroprocesorem pomocí sériové komunikace IIC. Základní blokové schéma na Obrázek 12 ukazuje, jak je celý snímač navržen.



Obrázek 12: Základní blokové schéma kapacitního snímače tlaku

Obvykle se u protokolu HART využívá pro přenos informace proudové smyčky, jak je ukázáno na Obrázek 1. V případě, že se snažíme ušetřit místo na desce plošného spoje, je možné vynechat části z obrázku, které jsou vyznačeny světle šedou barvou (HART modem, který moduluje datový signál na proudovou smyčku) a náhradou za konektor, ke kterému lze připojit externí převodních logických úrovní UART / RS232C a snímač tak můžeme připojit k PC a komunikovat pomocí protokolu HART s jinou fyzickou vrstvou (případ použitého kapacitního snímače tlaku).

2. Programové řešení snímače s protokolem HART



Obrázek 13: Inicializace mikroprocesoru

O celou funkčnost protokolu HART v mikroprocesoru PIC16F886 se v tomto případě starají dva stavové automaty a funkce na kontrolu platnosti přijatých dat. Jeden z automatů je určen pro odesílání a druhý pro příjem. Na základě přerušení při příjmu či vysílání se volá konkrétní automat, který podle statusu ví, v které části rámce protokolu HART se nachází a podle toho postupuje dál.

2.1 Inicializace mikroprocesoru

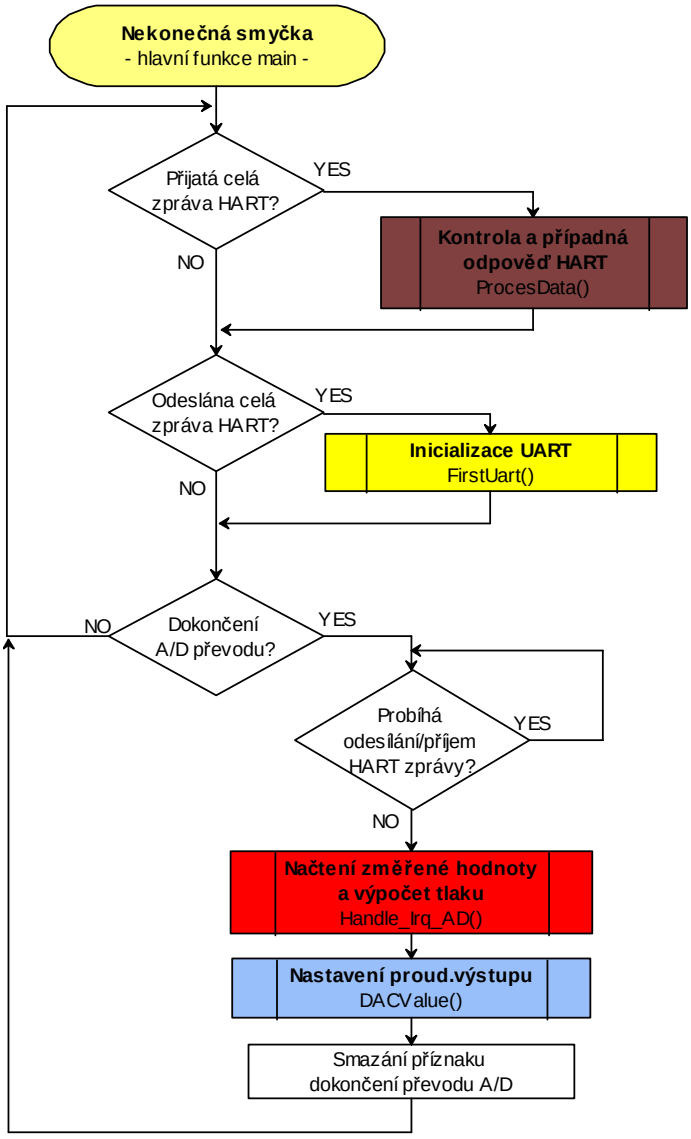
Ze všeho nejdřív je zapotřebí inicializovat samotný mikroprocesor a jeho registry. Po zapnutí napájení či resetu mikroprocesoru se provede inicializace, jak ukazuje Obrázek 13

Po samotném startu, je zapotřebí nejprve nastavit piny mikroprocesoru tak, aby byla umožněna komunikace s D/A převodníkem, poté načíst přesnou kalibrovanou hodnotu odesílanou do D/A převodníku, aby hodnota proudové smyčky byla co nejdříve po startu 4 mA. Dále inicializuji časovač pro hlídání, zda mezi byty při komunikaci HART nedochází k velkému zpoždění a následně je nastavena rychlost komunikace UART na 1200 b/s, příznak prvního spuštění po zapnutí napájení nebo resetu zařízení a je spuštěna inicializace UART rozhraní viz Obrázek 15.

Poté proběhne nastavení IIC rozhraní pro komunikaci mezi mikroprocesorem a EEPROM či A/D převodníkem pro měření hodnoty kapacity a teploty. Díky tomu lze jako další krok načíst hodnoty limitů, rozsahů a nastavení A/D převodníku AD7745 z EEPROM. Jakmile jsou získány tyto data, proběhne výchozí nastavení tohoto A/D převodníku.

2.2 Hlavní funkce

Po úspěšné inicializaci program začne cyklicky kontrolovat příznaky, které mohli vzniknout v hlavní smyčce např. při přerušení a případně je ihned zpracovává. Vývojový diagram na Obrázek 14 ukazuje, co všechno je hlídáno v cyklickém zpracování hlavní funkce.

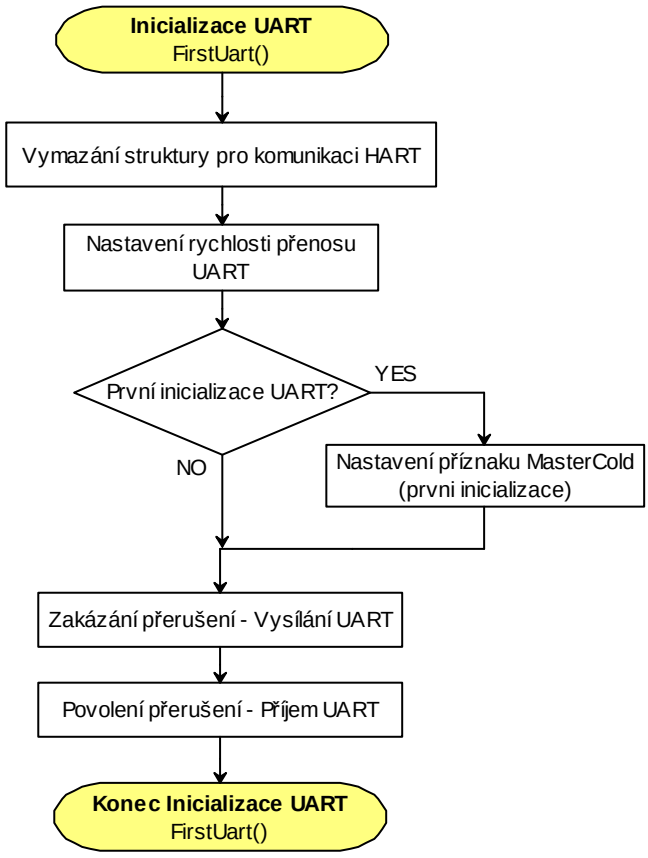


Obrázek 14: Nekonečná smyčka v hlavní funkci

Pokud zařízení přijalo kompletní zprávu HART, nastaví se při posledním z řady přerušení ve stavovém automatu příznak přijetí celé zprávy a v hlavní smyčce je splněna podmínka pro kontrolu a případnou odpověď na HART zprávu. Tato funkce je detailně zobrazena na vývojovém diagramu v příloze na Obrázek 36 a Obrázek 37.

Další příznak, který je v hlavní smyčce hlídán, je příznak odeslání celé zprávy HART, kdy stavový automat pro odesílání zpráv HART odešle kompletní zprávu a nastaví příznak, který povolí inicializaci UART rozhraní, která je zobrazena na Obrázek 15, kde se nejprve vymaže celá struktura pro příjem/odesílání dat pomocí protokolu HART, aby do ní poté mohly být uloženy přijaté hodnoty a nezůstaly v ní například hodnoty z minulé komunikace.

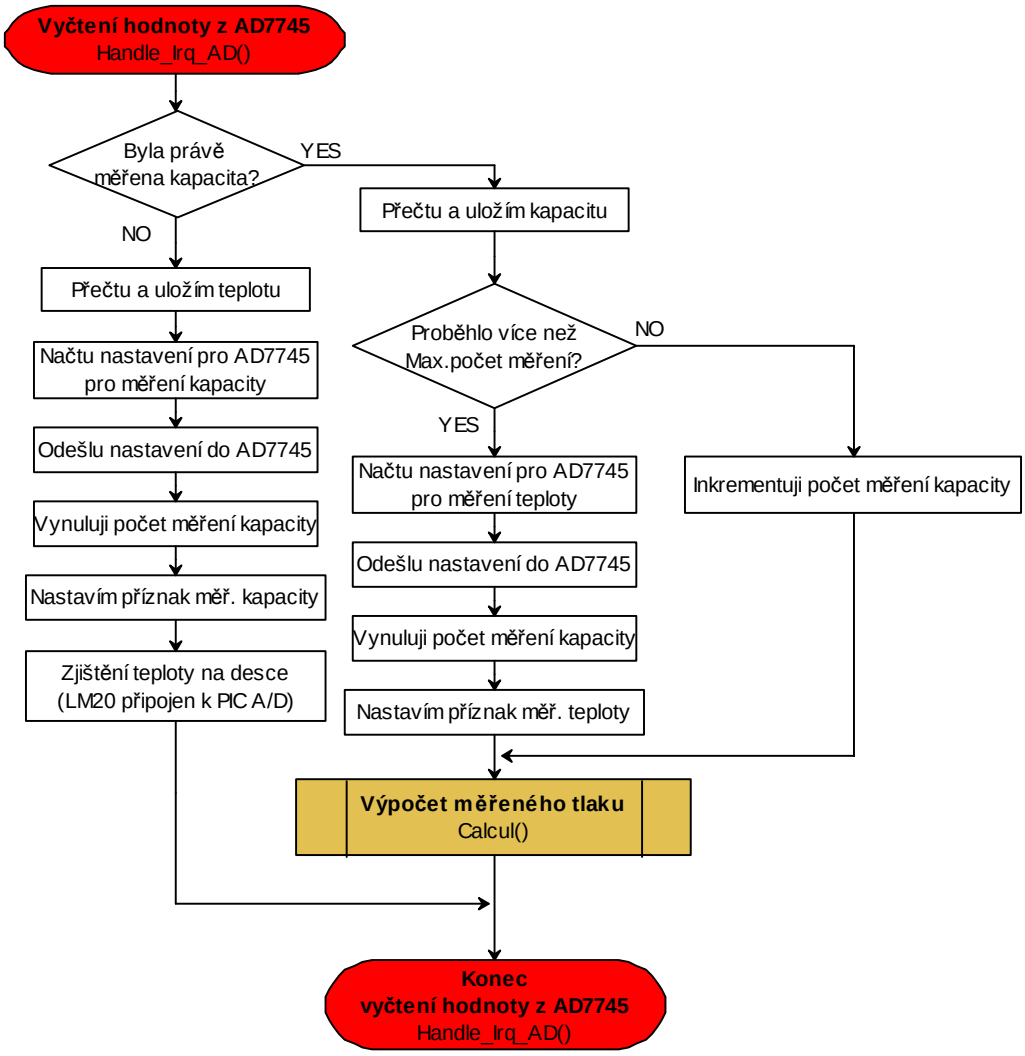
Pro případ, že by uživatel chtěl změnit komunikační rychlost za běhu programu, nastavuji při každé inicializaci znovu rychlost komunikace, jinak se používá výchozí rychlost 1200 b/s. Jelikož můžu komunikovat až se dvěmi Master zařízeními, nastavuji si pro PrimaryMaster i SecondaryMaster svůj status, zda již komunikovali. Po dokončení inicializace povolím přerušení pro příjem dat přes rozhraní UART a tím pádem povolím komunikaci HART.



Obrázek 15: Inicializace rozhraní UART

Po dokončené inicializaci se vykonávání programu vrací zpět do smyčky v hlavní funkci main(), kde setrvá až dokud mikroprocesor nezjistí přerušení, které vyvolá komunikaci buď přes protokol HART, nebo přes sériové rozhraní IIC pro vyčítání měřené veličiny z AD7745.

Poslední příznak, který je v hlavní smyčce hlídán, je dokončení A/D převodu převodníku AD7745, který vyvolá v mikroprocesoru přerušení (viz Obrázek 17) a je nastaven tento příznak dokončení převodu. Pokud právě neprobíhá komunikace HART, je ihned vyčítána hodnota kapacity či teploty z AD převodníku a v návaznosti na novou hodnotu je přepočítán i měřený tlak. Popis komunikace s A/D převodníkem a výpočet hodnoty tlaku bude popsán dále, princip je zobrazen z důvodu návaznosti Obrázek 16 na Obrázek 14.

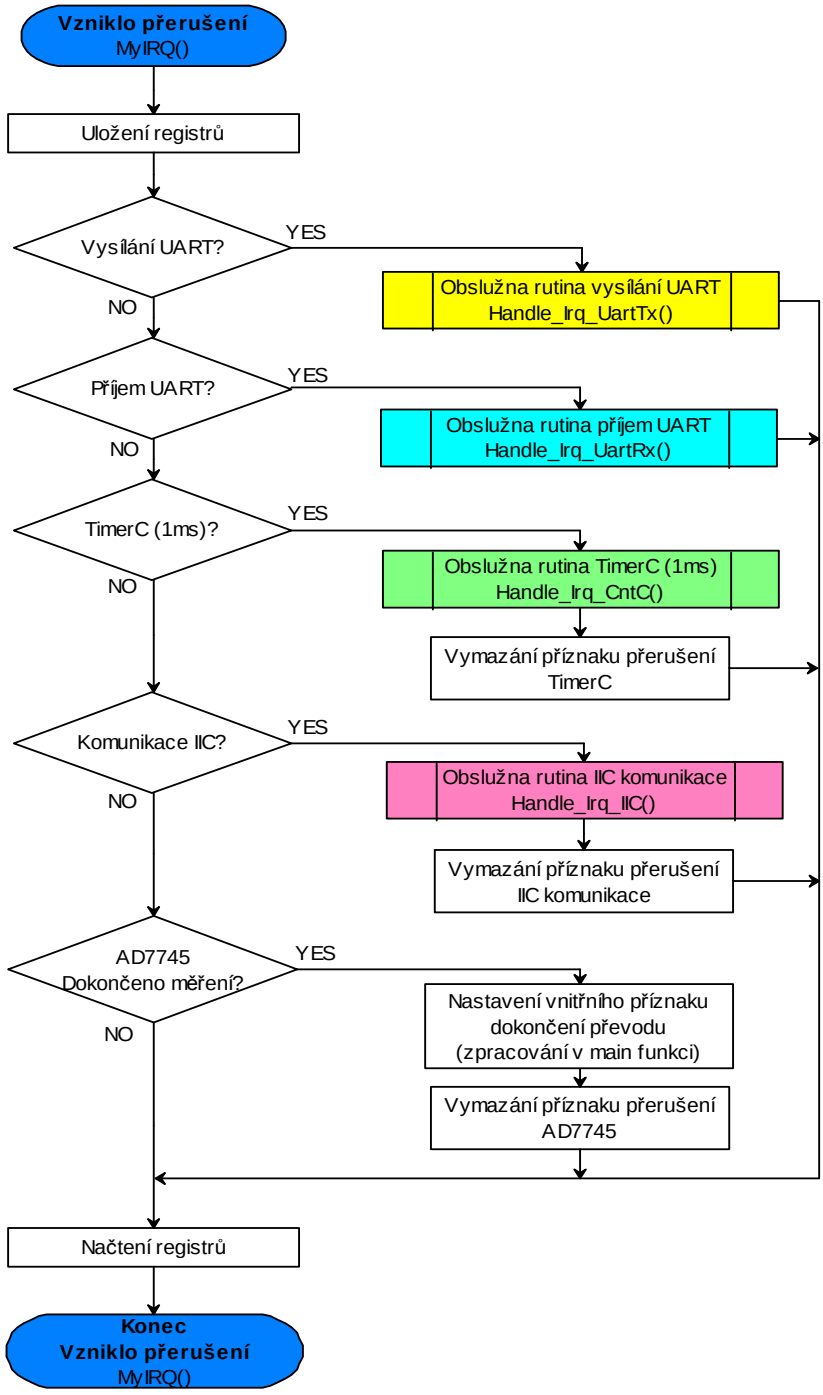


Obrázek 16: Obsluha A/D převodníku při dokončeném převodu

Na Obrázek 16 je zobrazen princip nastavování A/D převodníku. Z vývojového diagramu je vidět, že se jednou změří teplota pomocí AD7745, současně s teplotou desky pomocí integrovaného obvodu LM20 připojeného k A/D převodníku mikroprocesoru PIC16F886 a pak se přenastaví obvod AD7745 na měření kapacity, která probíhá obecně n-krát za sebou. Po dokončení převodu kapacity se také vždy vypočítá měřený tlak.

2.3 Vznik přerušení a jeho obsluha

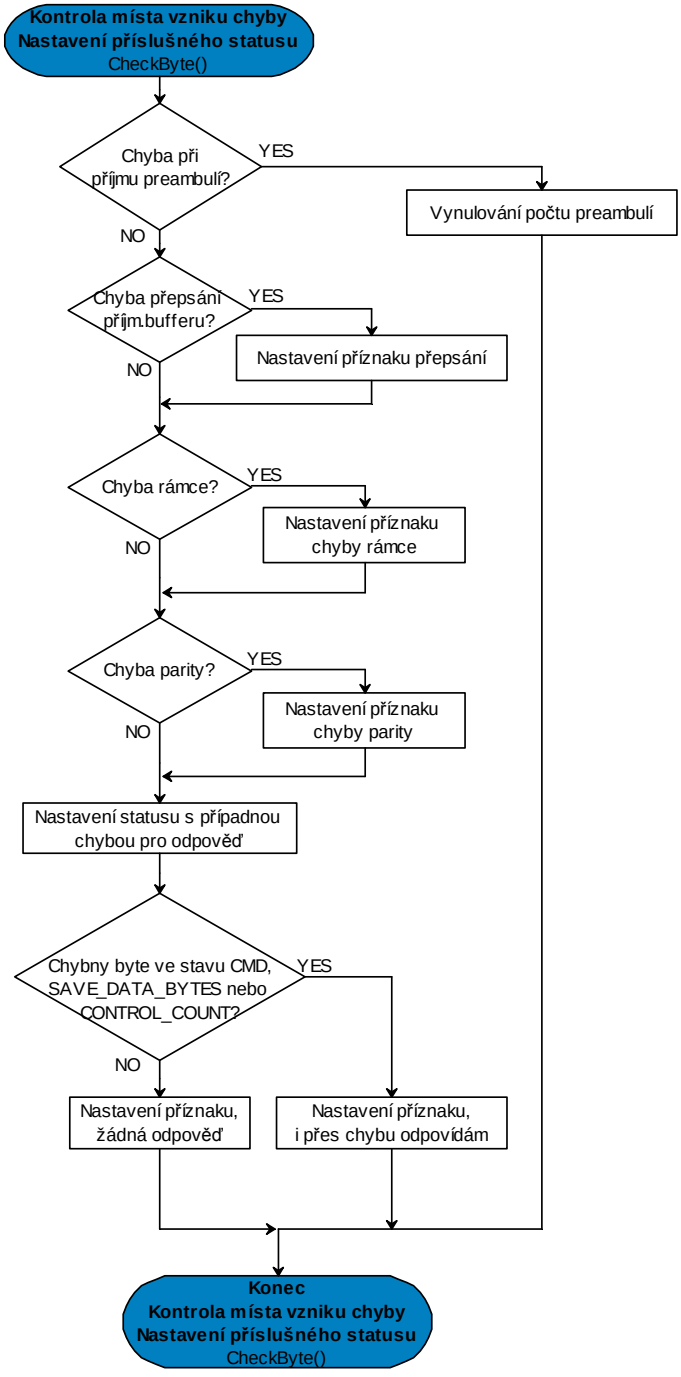
Po inicializaci samotného procesoru se dostáváme do stavu vyčkávání na přerušení, které svým způsobem řídí chod celého tlakového snímače. Vznikne-li jakékoliv přerušení v mikroprocesoru, bude volána funkce MyIRQ(), jejíž princip můžeme vidět na Obrázek 17.



Obrázek 17: Zjištění typu přerušení a zavolání obslužné rutiny

Pro samotnou činnost protokolu HART jsou v této aplikaci zapotřebí 3 typy přerušení mikroprocesoru a to jsou přerušení od časovače TimerC, od příjmu dat UART a vysílání dat UART rozhraním (pro HART modem). Přerušení pro IIC rozhraní je určeno pro komunikaci mikroprocesoru a externí EEPROM nebo A/D převodníku AD7745, který měří samotnou hodnotu kapacity čidla. Kapacita je převedena na číslo, které bude dále zpracováváno a odpovídajícím způsobem vyhodnoceno jako informace o hodnotě tlaku.

Jako poslední přerušení, které je obsluhováno, jde o přerušení právě od převodníku AD pro měření kapacity. Toto přerušení „oznamuje“ mikroprocesoru, že převodník AD7745 dokončil převod kapacity na jeho vstupu na číslo a to je připraveno v registru pro odeslání.



Obrázek 18: Kontrola jednotlivých přijímaných bytů

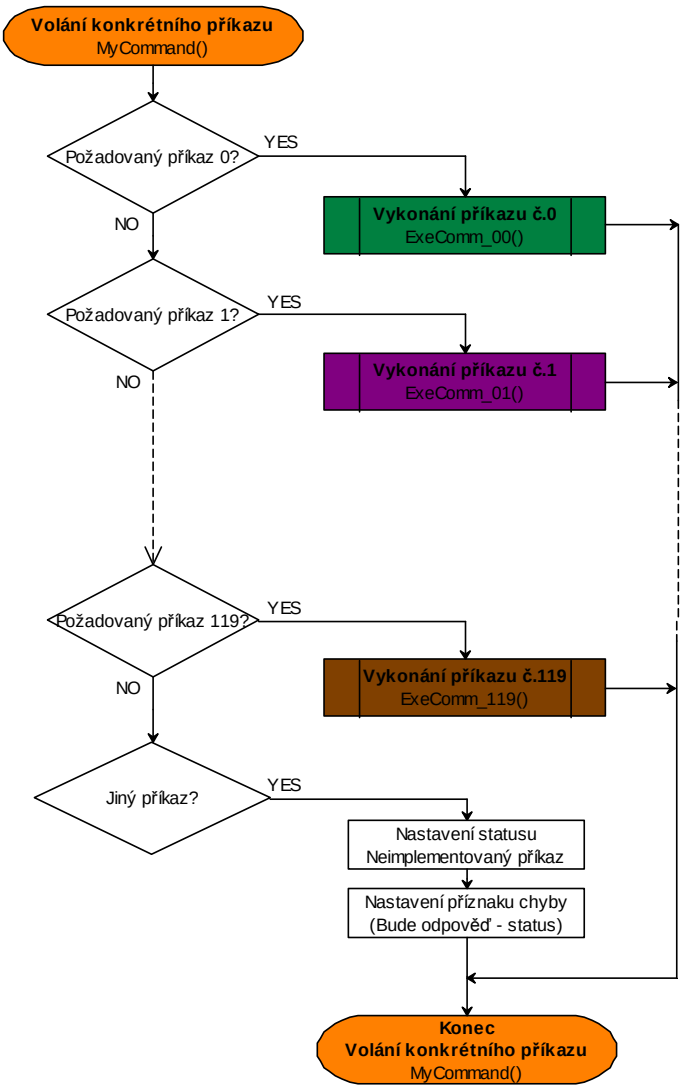
2.3.1 Přerušení pro příjem dat - UART (příjem dat HART)

V obslužné rutině pro příjem dat přes UART rozhraní, je první stavový automat pro uložení přijímaných hodnot přes HART protokol. Za normálních okolností se pomocí HART modemu připojeného k UART rozhraní mikroprocesoru převedu data posílaná po proudové smyčce na číslicovou hodnotu a tu posílám ve formě datového bytu do přijímacího registru UART, přesně jak je zobrazeno na Obrázek 5. V tomto případě ale deska plošného spoje snímače neobsahuje HART modem a data jsou posílána přímo do přijímacího registru UART (blokové zobrazení viz Obrázek 12). Po příjmu datového bytu tímto rozhraním se vyvolá přerušení a je volán stavový automat pro příjem dat UART. Přesný vývojový diagram takového stavového automatu pro příjem dat můžeme vidět v příloze na Obrázek 34 a Obrázek 35.

Automat se stará o příjem celého rámce, jak je na Obrázek 6.

Přijímací automat kontroluje každý přijatý byte pomocí funkce na správnou lichou paritu, zda nedošlo k přepsání zásobníku UART či chybu v rámci jednoho bytu. Tato funkce je ukázána na Obrázek 18.

Funkce automatu je založena na tom principu, že po přijetí prvního bytu se inicializuje automat a očekává přijetí synchronizačních bytů - preambulí. Po přijetí dostatečného množství preambulí, se automat přepne do dalšího stavu, kdy očekává příjem delimiteru. Po jeho obdržení otestuje, zda je delimiter platný, jestli ano, uloží ho pro další zpracování, vypočte z něj část kontrolního součtu tzv. podélné parity a nastaví další stav automatu. Tímto způsobem probíhá příjem celého rámce protokolu HART. Po kompletním přijetí rámce můžeme vidět na vývojovém digramu v druhé části přijímacího automatu, že se nastaví vnitřní příznak pro přijetí rámce HART a na to reaguje mikroprocesor v cyklickém zpracování hlavní funkce (Obrázek 14), která následně volá funkci ProcesData(), která má na starosti zpracovat přijaté data a případně na ně odpovědět. Tato funkce a její činnost je zobrazena na Obrázek 36 a Obrázek 37.



Obrázek 19: Principielní blokové zobrazení funkce MyCommand() pro volání konkrétního příkazu

Funkce bere jednotlivé části přijaté protokolem HART a testuje je, jestli jsou v pořádku. Takto projde všechny přijaté byty, jako je ukázáno v diagramu a jestli je vše v pořádku, může se provést příkaz, který byl obdržen v těle rámce HART protokolu. To má na starost další funkce, konkrétně MyCommand(), které se jako parametr předá číslo příkazu, který byl přijat pro vykonání v rámci přijaté zprávy a jestli je tento příkaz implementován v zařízení, funkce volá tento konkrétní příkaz pro vykonání. Pokud zařízení tento příkaz implementovaný nemá,

nastaví se příznak pro neznámý příkaz a odpoví se na tento rámec tak, že v datové části budou jen dva byty statusu a v něm bude nastaven tento příznak (viz Obrázek 19).

Ukázka několika vývojových diagramů jednoduchých HART příkazů je zobrazena v příloze na Obrázek 38, Obrázek 39 a Obrázek 40. Zbytek příkazů je přiloženo pouze ve formě zdrojového kódu.

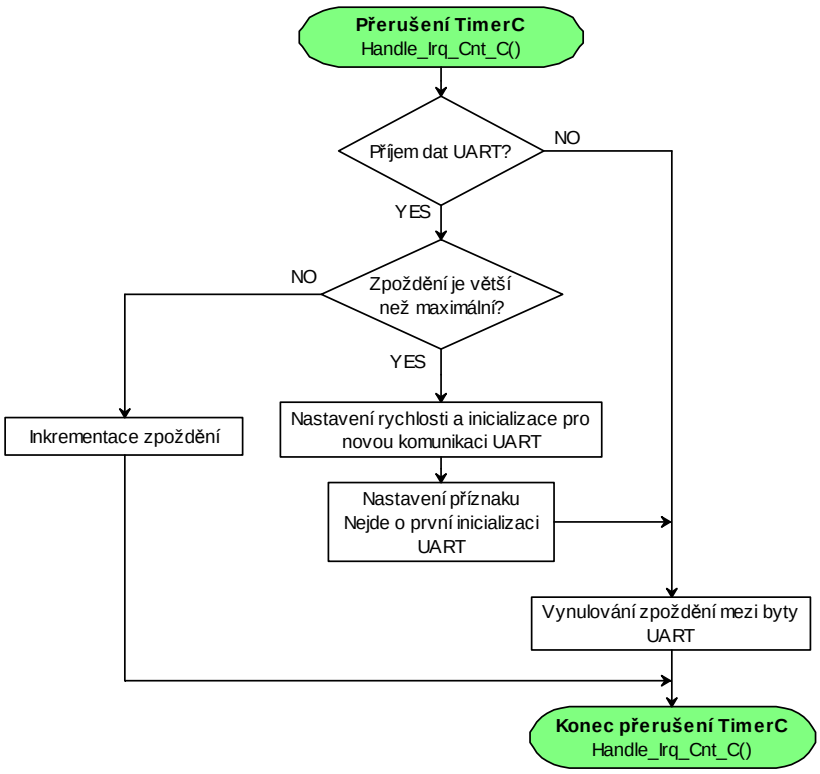
Po dokončení příkazu se vrátíme zpět až do funkce ProcesData(), která dále zjistí, zda při provádění příkazu nevznikla chyba. V případě chyby záleží také na typu chyby. Je-li chyba závažná, nebude se na příkaz vůbec odpovídat, zahodí se všechny data, znovu se inicializují automaty, vymaže se struktura pro příjem i odesílání dat a čeká se na novou komunikaci. Nenastane-li chyba, funkce inicializuje druhý stavový automat pro odesílání a povolí přerušení pro vysílání dat. Tím se dostaneme do druhého stavového automatu pro vysílání.

2.3.2 Přerušení pro vysílání dat - UART (vysílání dat HART)

V obslužné rutině přerušení od vysílacího modulu rozhraní UART je umístěn stavový automat pro vysílání dat protokolu HART prostřednictvím rozhraní UART. Princip tohoto stavového automatu je stejný jako princip stavového automatu pro příjem dat prostřednictvím rozhraní UART (viz Přerušení pro příjem dat – UART), tedy po zapsání prvního bytu do zásobníku pro odeslání dat rozhraním UART se volá obsluha přerušení (stavový automat), který zná aktuální pozici v odesílaném rámci protokolu HART a tak vždy provede odeslání následujícího bytu a případně změní svůj stav pro odesílání další časti rámce, nebo změní pouze některou z pomocných stavových proměnných, která udává, který byte byl právě odeslán z konkrétní části v rámci protokolu (např. nacházíme-li se v části rámce „Adresa“ a má se odesílat dlouhá 5 bytová adresa, stavová proměnná stavového automatu pro odesílání dat bude rovna „DELIMOK“ a pomocná stavová proměnná mi bude říkat, který byte z adresy již byl odeslán a který se má teprve odesílat. Pro bližší představu je vývojový diagram stavového automatu pro odesílání dat UART přiložen v příloze na Obrázek 41 a Obrázek 42. Po odeslání všech částí rámce protokolu HART, stavový automat nastaví vnitřní příznak pro dokončení odesílání rámce HART a v hlavní funkci se pak provede inicializace modulu UART, vymaže se celá struktura pro případnou příští komunikaci HART, zároveň se povolí přijímání dat prostřednictvím UART rozhraní a zakáže se vysílání (vysílání se povolí až po přijetí kompletního rámce HART – při odpovědi na HART příkaz).

2.3.3 Přerušení časovače TimerC (1ms)

Časovač TimerC je nastaven pro přerušení vznikající v intervalu 1 ms. Po vzniku příznaku přerušení se zavolá funkce MyIRQ(), která rozpozná, že se jedná o přerušení od časovače TimerC a zavolá funkci Handle_Irq_Cnt_C(), která se stará o kontrolu zpoždění mezi byty při příjmu dat prostřednictvím protokolu HART. Vývojový diagram obsluhy přerušení od tohoto časovače je na Obrázek 20.

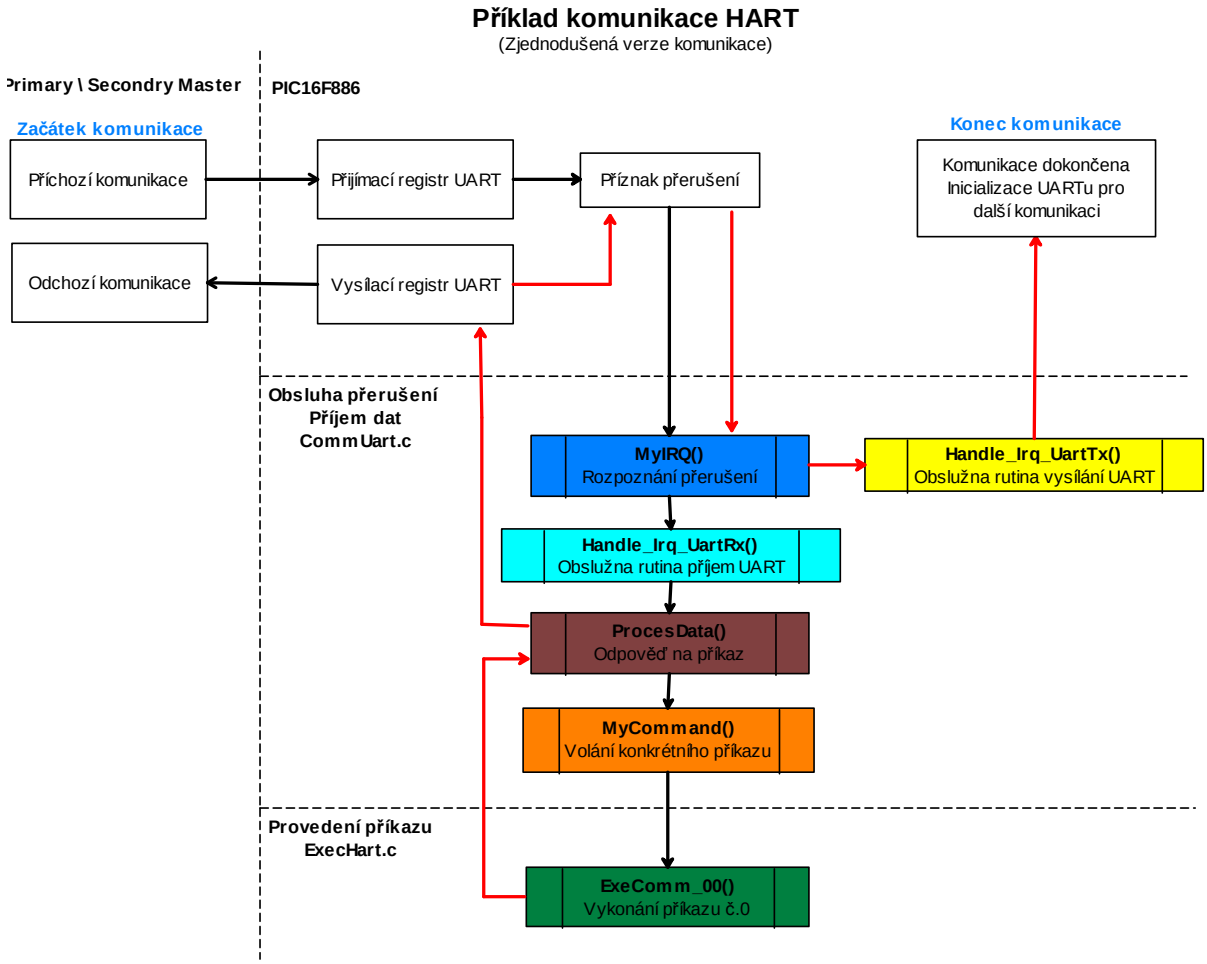


Obrázek 20: Obsluha přerušení časovače TimerC (1ms)

Po zavolání obslužné funkce časovače TimerC se nejprve testuje, zda se právě přijímají data pomocí modulu UART (probíhá příjem rámce HART protokolu), poté se otestuje, zda zpoždění, není větší než povolená doba zpoždění (zpoždění se nuluje pokaždé, když vznikne přerušení od příjmu dat UART (v obslužné rutině). Pokud není zpoždění vetší, pouze se zvětší hodnota zpoždění a ukončí se obsluha přerušení. V opačném případě, je-li hodnota zpoždění mezi byty větší, ukončuji celé přijímání a inicializuji UART modul pro další příjem dat.

2.4 Zjednodušený příklad HART komunikace

Na Obrázek 21 je ukázán princip celé implementované komunikace HART do mikroprocesoru PIC16F886. Tento příklad je jen pro představu, jakým způsobem se přijme komunikace a jak se na ni odpoví. Příklad, který je ukázán na obrázku by sloužil pro příjem a vysílání pouze jediného bytu a jelikož zpráva HART protokolu není o velikosti 1B, schéma není zobrazení skutečné komunikace a slouží jen pro představu. Ve skutečnosti se přerušení pro příjem / vysílání opakuje tolikrát, kolik je přijatých / vysílaných bytu a po dokončení přijímání se zpracovávají přijaté data a vykonává příkaz. Popis jednotlivých části programu viz 2.3 Vznik přerušení a jeho obsluha.



Obrázek 21: Zjednodušený příklad HART komunikace

3. Komunikace s externí EEPROM

Pro samotnou činnost snímače je zapotřebí mít někde v paměti (nejlépe nezávislé na napájení) uloženy jak nejrůznější konstanty pro výpočty, tak i základní konfiguraci snímače a A/D převodníku. Tyto údaje jsou v případě popisovaného tlakového snímače uloženy v externí EEPROM s označením 24LC02. Jedná se o paměť o velikosti 256 Bytů, která komunikuje s mikroprocesorem PIC16F886 sériovou komunikační linkou IIC.

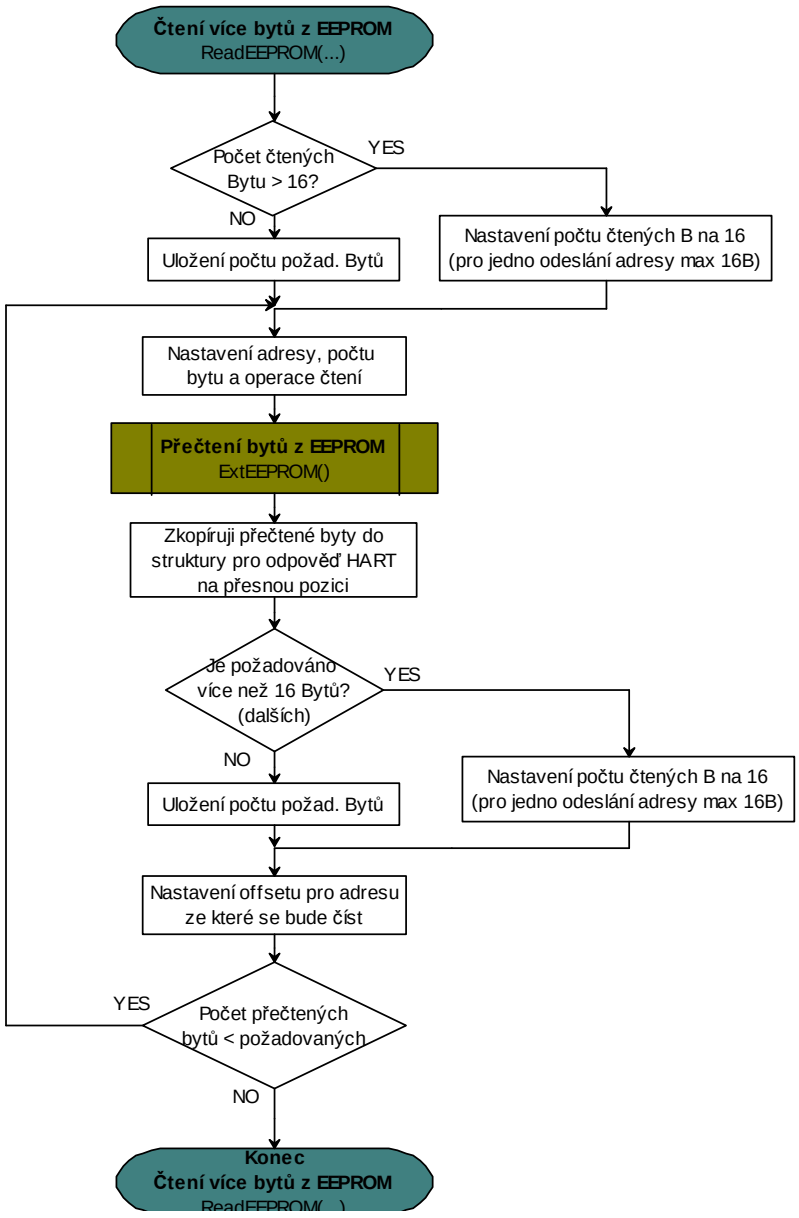
U navrhovaného snímače tlaku je využita jak vnitřní EEPROM mikroprocesoru, tak i externí EEPROM. Jedná se o modulový koncept, kdy na čidle jsou obvody A/D převodníku spolu s EEPROM, v které jsou uloženy data týkající pouze čidla (především nastavení A/D převodníku a polynomy pro kompenzaci tlaku).

Dále jsou ukázány jednotlivé principy, jak se provádí samotné vyčítání či zápis hodnot do/z této externí paměti.

3.1 Čtení více bytů z externí EEPROM

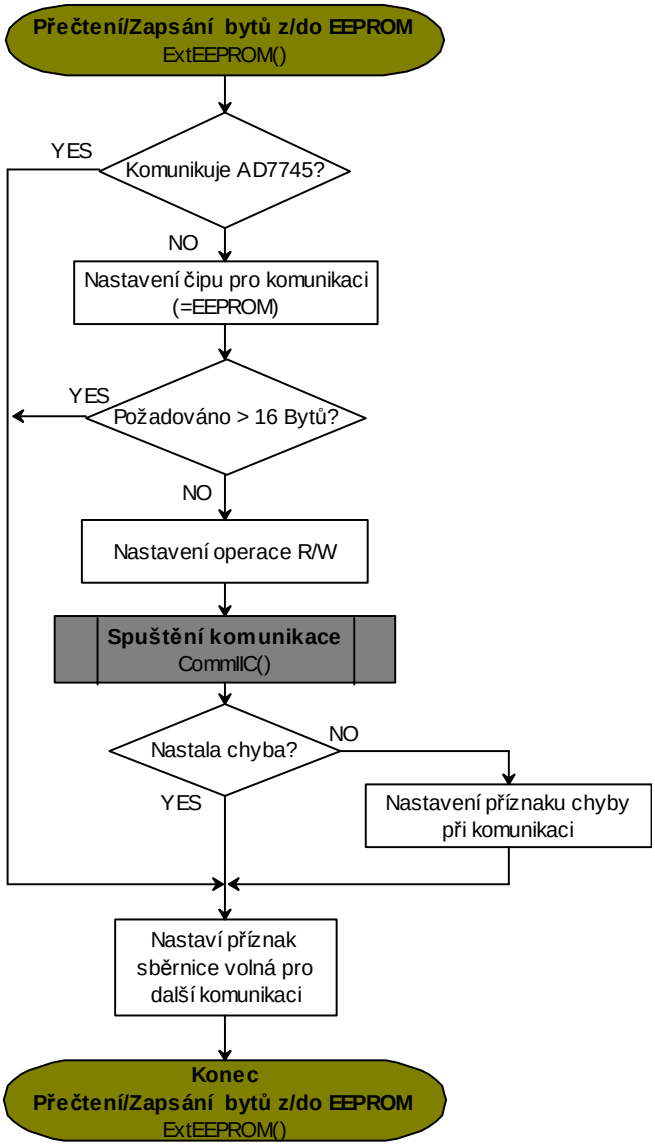
Pro vyčítání více bytů najednou je výhodné použít pro tuto sériovou komunikaci IIC možnosti odeslání pouze jedné adresy do paměti a poté pro další načtení stačí odeslat EEPROM potvrzovací bit ACK a paměť nám odešle další byte z následující adresy. Takto je schopna zmiňovaná EEPROM odeslat za sebou až 16 B, což je pro rychlost komunikace a i samotného měření tlaku velmi výhodné, protože jinak bychom museli pro každý vyčítaný byte z paměti odesílat adresu, ze které chceme číst data a tím by se doba nutná pro načtení určitého počtu bytů velmi prodloužila.

Pro tento případ jsem napsal funkci, která této vlastnosti sériové komunikace využívá a rozděluje velké množství čtených bytů na bloky maximálně po 16 B, které následně přečte. Vývojový diagram pro tuto funkci je zobrazen na Obrázek 22. Na začátku celé funkce můžeme vidět, že je omezen počet čtených bytů na maximálně 16 pro jeden cyklus. Poté nastavím adresu v EEPROM, ze které chci číst data a volám funkci ExtEEPROM(..) pro komunikaci mezi externí EEPROM a samotným mikroprocesorem (popsáno dále). Po návratu z této funkce zkopíruji načtené data na pozici, kterou jsem si předem zvolil (ukazatelem) a testuji, jestli je počet bytů, které jsou požadovány větší, než počet aktuálně načtených bytů. Pokud ano, nastavím počet požadovaných bytů, zvětším adresu, ze které chci číst o 16 (počet již načtených bytů) a vracím se před funkci pro čtení dat. Tímto způsobem pokračuji, dokud nenačtu všechny požadované byty z paměti.

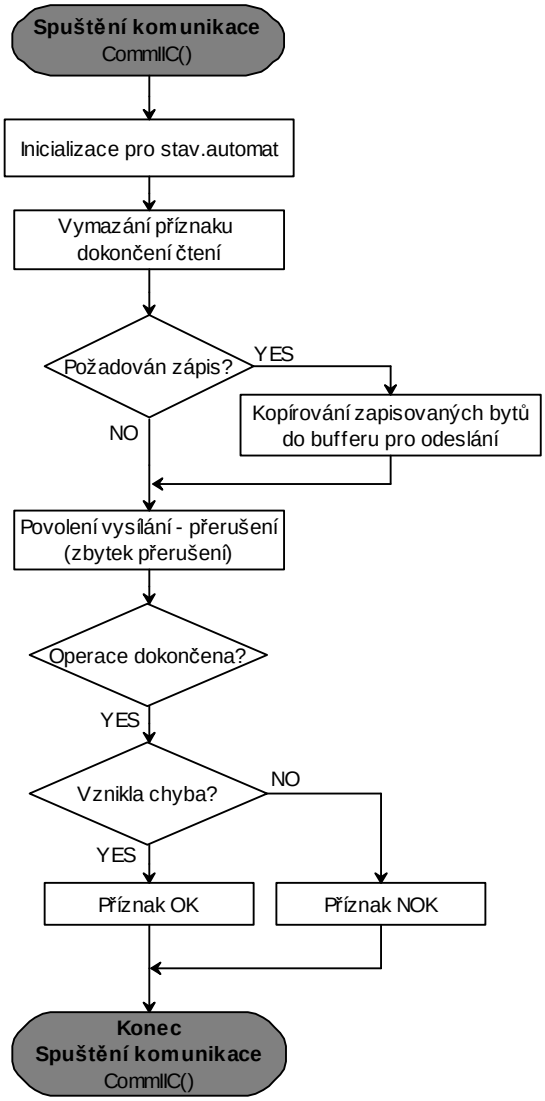


Obrázek 22: Funkce pro načtení více bytů z externí EEPROM

Princip funkce pro vyčítání hodnot z EEPROM je ukázán dále na Obrázek 24. Ihned po zavolání funkce ExtEEPROM(..) zjišťuji, zda již po sériovém rozhraní nekomunikuje A/D převodník AD7745 pro měření kapacity (teploty). Je-li sériové rozhraní volné, otestuji, zda není požadováno pro čtení více než 16 B a jestliže je všechno v pořádku, nastavím příznak R/W, nastavím příznak pro komunikaci s EEPROM a zahájím čtení spuštěním další funkce CommIIC(). Po návratu z této funkce testuji, zda při načítání dat nevznikla chyba, případně nastavím příznak a uvolním sběrnici pro další komunikaci.



Obrázek 24: Vývojový diagram funkce ExtEEPROM(...)



Obrázek 23: Vývojový diagram funkce CommIIC()

Samotná sériová komunikace IIC rozhraním je velmi obdobná principu samotného přijímání a odesílání dat pomocí protokolu HART. Pro řízení odesílání i přijímaní dat po IIC sběrnici je použit opět stavový automat.

Na vývojovém diagramu na Obrázek 23, můžeme vidět princip funkce CommIIC(), která se stará právě o počáteční nastavení tohoto automatu a poté čeká na dokončení čtení z EEPROM.

Na dokončení požadované operace se musí čekat jednak z důvodu, že sériové komunikace využívá A/D převodník (přepsání přijatých či odesílaných dat uložených v zásobníku), ale hlavně z důvodu, aby program nepracoval dále s náhodnými údaji, jako by byla předchozí komunikace dokončena.

Po dokončení čtení se provede kontrola, zda v průběhu komunikace po sériové lince nenastala chyba a v případě, že je vše v pořádku, nastaví se příznak pro operaci OK, v opačném případě nastavuji příznak NOK a opouštím funkci CommIIC(). Podrobnější popis obsluhy přerušení pro komunikaci IIC je uveden dále.

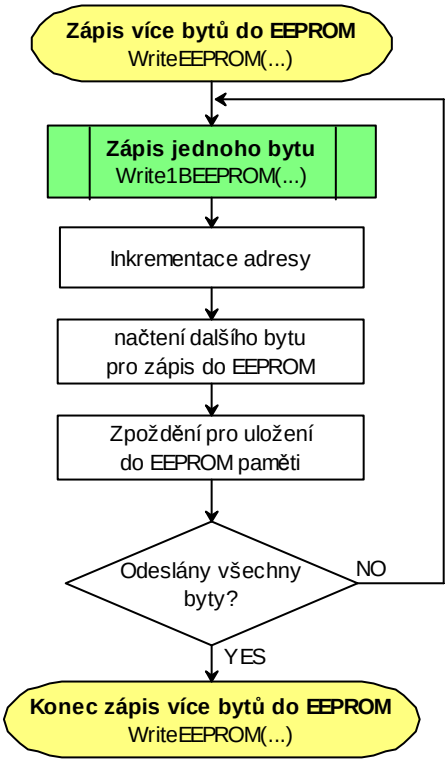
3.2 Zápis více bytů do externí EEPROM

Pro zápis do EEPROM je vytvořena funkce WriteEEPROM(), jejíž vývojový diagram je na Obrázek 25. Při zápisu podporuje EEPROM, stejně jako pro čtení, zápis až 16 B najednou, bez odesílání více adres, kam je požadováno datové byty zapsat. Jak je řečeno v předchozí kapitole, to podstatně zmenší dobu potřebnou pro komunikaci pomocí IIC sběrnice.

Při použití tzv. zápisu stránky nelze ale zapisovat menší počet bytů, než je právě zmiňovaná stránka, tedy 16 Bytů. Z toho důvodu je navrhnuta funkce pro zápis více bytů do EEPROM odlišně, než je funkce pro čtení více bytů z této paměti. Vývojový diagram funkce je ukázán na Obrázek 25.

Využívá se zde funkce Write1BEEPROM (..), jak již název napovídá, slouží pro zápis jednoho bytu do EEPROM.

Funkce pro zápis bytů dostává jako jeden z parametru ukazatel na pozici v paměti mikroprocesoru, ovšem ne s místem, kde se mají data ukládat, jako je tomu u příjmu dat, ale odkud se mají data nakopírovat do zásobníku pro odesílání. Překopírování dat do zásobníku se provádí po jednom bytu, tedy v každém cyklu se nakopíruje nový datový byte pro odeslání. Při zápisu je nastaven místo příznaku pro čtení dat z EEPROM příznak zápisu a o samotné vyslání bytu po sériové komunikaci se stará opět stavový automat implementovaný v obslužné rutině pro přerušení od IIC rozhraní. Tedy zavolá se funkce ExtEEPROM(), kde je kontrola, zda nekomunikuje A/D převodník a volá se funkce CommIIC(), která nastavuje stavový automat pro zápis dat do EEPROM a poté čeká na dokončení samotného zápisu dat signalizovaného stavovým automatem.



Obrázek 25: Zápis více bytů do EEPROM paměti WriteEEPROM()

3.3 Přerušení pro komunikaci IIC

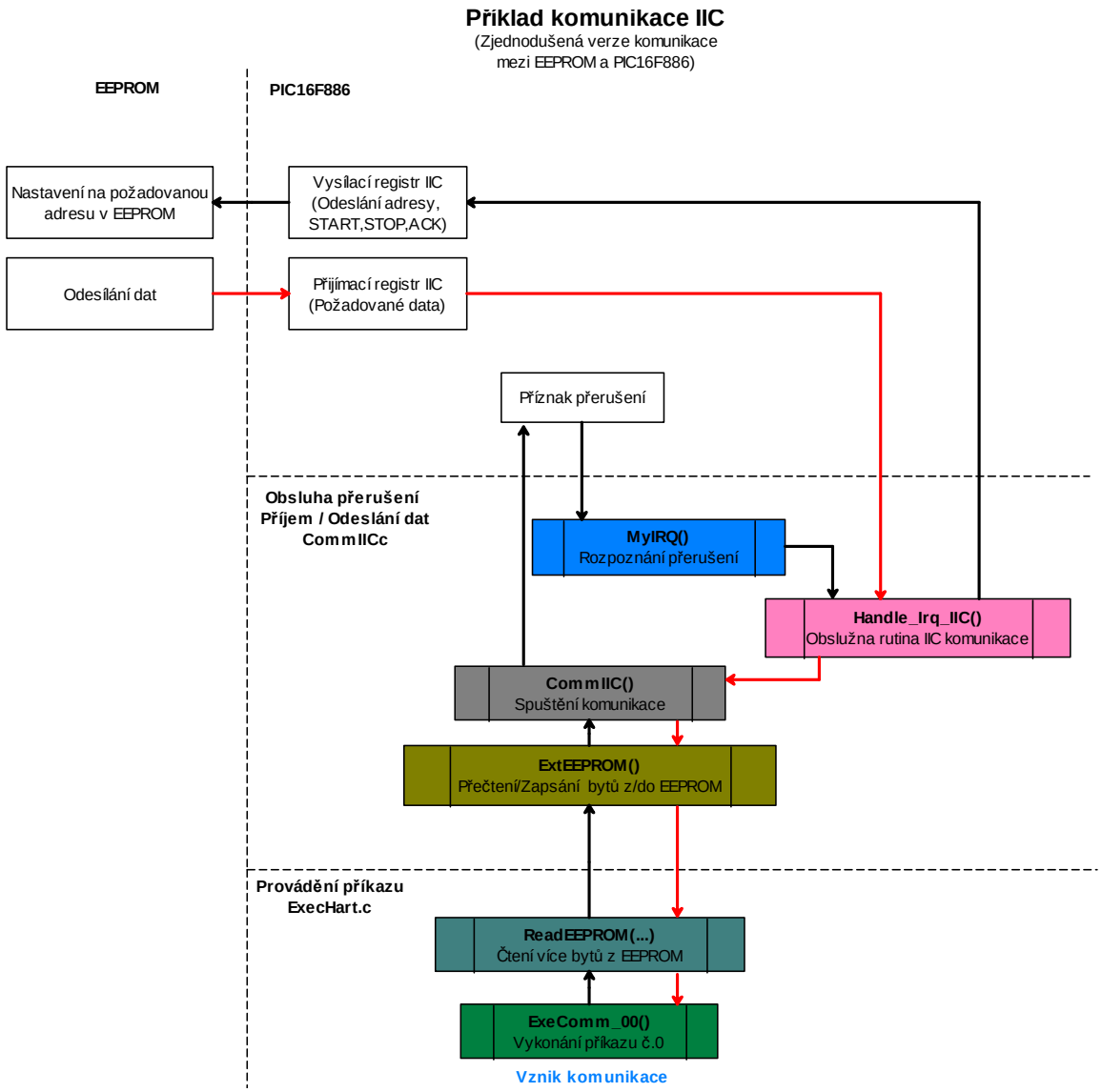
Přerušení pro komunikaci vzniká při zápisu/čtení dat z/do EEPROM, nebo A/D převodníkem AD7745. Přerušení nastane při zpracování funkce CommIIC(), která je zobrazena na Obrázek 23, v okamžiku, kdy se povolí vysílání dat (nastaví se synchronizační bit) a tím se program dostane do obslužné rutiny přerušení pro sériovou komunikaci, kde se nachází stavový automat, který již řídí celý průběh sériové komunikace. Vývojový diagram takového automatu je zobrazen v příloze na Obrázek 43 a Obrázek 44. Tento automat postupně zpracovává jednu část rámce sériové komunikace IIC za druhou a podle toho, o jakou operaci se má jednat (čtení/zápis), postupuje dále. Dále řeší odesílání ACK příznaku při čtení dat ze zařízení připojeného na sériovou sběrnici a lze pomocí něj přijímat/odesílat libovolné množství bytů ze/na sběrnici. Stavový automat je tedy napsán univerzálně, aby pomocí něj bylo možné komunikovat s jakýmkoliv zařízením podporujícím IIC sériovou komunikaci.

3.4 Zjednodušený příklad sériové komunikace IIC

Pro představu funkce celé komunikace IIC zde uvádím zjednodušenou verzi sériové komunikace tak, jak by měla probíhat mezi mikroprocesorem a externí EEPROM. Blokové zobrazení principu komunikace je ukázáno na Obrázek 26.

V tomto případě je funkce při provádění příkazu HART protokolu potřeba načítat z EEPROM určitá data a je tedy volána funkce ReadEEPROM(..). Zpracování pak prochází funkcemi ExtEEPROM() a CommIIC(), kde povolím synchronizaci a tím se dostávám do obslužné rutiny pro přerušení komunikace IIC. Zde se mi již stavový automat stará o celý průběh přijímání dat a přerušení nastane tolikrát, až se stavový automat dostane do stavu FINAL, kdy je nastaven příznak dokončeného přijímání dat a může se tak pokračovat ve zpracovávání příkazu HART, vracím se tak zpět až do původního místa volání funkce ReadEEPROM() pro načtení dat z EEPROM.

V příkladu je tedy „nepřesnost“, že obsluha přerušení komunikace IIC neproběhne jen jednou, ale stavový automat si ji řídí přesně podle diagramu zobrazeného na Obrázek 43 a Obrázek 44, a až při konečném stavu automatu se pak vracíme zpět do funkcí, odkud bylo čtení dat z EEPROM požadováno.



Obrázek 26: Blokové zobrazení principu sériové komunikace IIC

4. Komunikace s vnitřní EEPROM mikroprocesoru

Jak je zmínění v minulé kapitole pro komunikaci s externí EEPROM, snímač je navržen modulově, tedy čidlo s A/D převodníkem a EEPROM lze od desky s mikroprocesorem odpojit a připojit k jiné desce s tím, že čidlo má uloženo v dané EEPROM svou konkrétní konfiguraci. Pro uložení dat, která jsou potřebná pro správnou funkci jak mikroprocesoru, tak celého snímače například při komunikaci protokolem HART se využívá vnitřní EEPROM mikroprocesoru PIC16F886 o velikosti 256 Bytů.

Dále v textu bude ukázán princip, jakým způsobem se komunikuje s vnitřní EEPROM.

4.1 Čtení dat z vnitřní EEPROM

Číst data z vnitřní EEPROM mikroprocesoru PIC16F886 je v programu dvěma způsoby. První možností je vyčítání jednotlivých bytů. Pro tuto možnost je implementována funkce pro čtení jednoho Bytu z vnitřní EEPROM. Popis této funkce je v následující podkapitole.

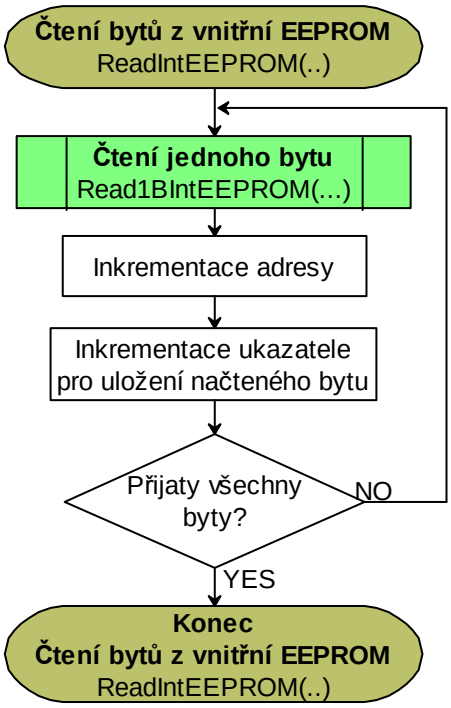
Jiný přístup pro čtení dat z vnitřní EEPROM bude potřeba při načítání více bytů z této paměti. Z tohoto důvodu je navržena funkce pro realizující čtení více bytů z vnitřní EEPROM. Problém čtení více bytu je ukázán v jedné z následujících kapitol.

4.1.1 Čtení jednoho Bytu z vnitřní EEPROM

Pro čtení dat z vnitřní EEPROM je využito funkce Read1BIntEEPROM(), která umožňuje čtení jednoho bytu. Mikroprocesor PIC16F886 je schopen přečíst a vystavit požadovaný byte z vnitřní EEPROM v jednom cyklu mikroprocesoru. Tedy zapíše se adresa do EEADR registru mikroprocesoru, ze které je požadavek na přečtení bytu, poté je smazán kontrolní bit EEPGD z registru EECON1 a nakonec se povolí čtení nastavením bitu RD (registr EECON1). Funkce pro čtení 1 Bytu z vnitřní EEPROM načtený byte přímo vrací. Ihned v dalším kroku programu jsou data z paměti připravena v registru EEDAT, tedy obsah tohoto registru se vrací jako návratová hodnota funkce pro načtení 1 Bytu.

4.1.2 Čtení více Bytů z vnitřní EEPROM

Čtení více bytů z vnitřní paměti mikroprocesoru umožňuje funkce ReadIntEEPROM(), které se předávají jako parametry počáteční adresa, ze které se mají data načítat, počet požadovaných bytů, které se mají přečíst z paměti a jako poslední parametr je uveden ukazatel, na které místo v paměti mikroprocesoru se mají načtená data uložit.



Obrázek 27: Funkce pro načtení více bytů z vnitřní paměti EEPROM mikroprocesoru ReadIntEEPROM(..)

Princip funkce je naznačen na Obrázek 27. Po zavolání této funkce z programu, se ihned volá funkce pro načtení jednoho bytu z vnitřní EEPROM (popsáno v minulé kapitole). Jako další postup se inkrementuje adresa, ze které se mají číst další data, zvýší se také ukazatel na místo, kam se má načtený byte z paměti uložit a opět se volá funkce pro načtení jednoho bytu, pouze s inkrementovanou adresou a uložením načtených dat na následující paměťové místo v mikroprocesoru. Takto se cyklicky funkce provádí do doby, než je načtený požadovaný počet datových bytů.

4.2 Zápis dat do vnitřní EEPROM

Stejně jako pro čtení dat z vnitřní EEPROM mikroprocesoru PIC16F886, jsou i pro zápis dvě implementované funkce, kterými je možné data do paměti zapsat. Následující dvě podkapitoly se zabývají zápisem dat do vnitřní EEPROM.

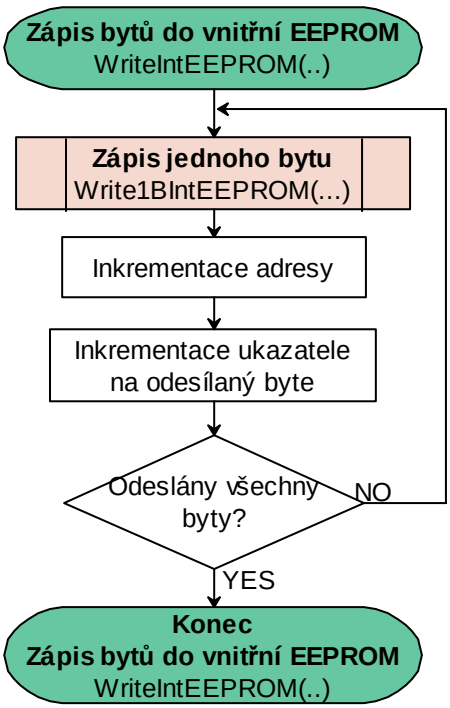
4.2.1 Zápis jednoho bytu do vnitřní EEPROM

Funkce Write1BIntEEPROM(..) pro zápis jednoho bytu do vnitřní EEPROM mikroprocesoru PIC16F886, je velmi podobná dříve popisované funkci pro čtení jednoho bytu z této paměti. Nejprve se tedy zapíše do registru EEADR adresa předaná jako parametr funkce. Na tuto adresu se bude následně zapisovat. Ještě před spuštěním zápisu je však zapotřebí načíst do registru EEDAT byte, který se má zapsat do paměti. Tento datový byte se také předává funkci jako parametr. Po načtení datového bytu do registru EEDAT, se nastaví příznakový bit pro zápis dat do EEPROM, tedy WREN. Na rozdíl od čtení dat, je v zapisovací funkci zapotřebí

zakázat veškerá přerušení z důvodu bezchybného zápisu, je tedy smazáno globální povolení přerušení GIE. Pro další postup je definováno několik kroků, bez kterých by zápis do paměti nebyl platný. Nejprve je do registru EECON2 načtena hodnota 0x55 a ihned poté je do registru načtena hodnota 0xAA. V návaznosti na tuto posloupnost se povolí příznakový bit pro spuštění zápisu do vnitřní EEPROM. Nyní je již možné povolit globální přerušení GIE. Ve funkci se pak ještě hlídá dokončení zápisu do paměti, který je signalizován nastavením bitu EEIF. Jakmile je zápis dokončen, je provedeno smazání příznaku pro zápis WREN a také příznaku dokončení zápisu EEIF.

4.2.2 Zápis více bytů do vnitřní EEPROM

Funkce pro zápis více bytů do vnitřní EEPROM mikroprocesoru je navrhnut téměř shodně s funkcí pro čtení více bytů z vnitřní EEPROM. Funkce umožňující zápis více bytů do paměti je WriteIntEEPROM(..). Stejně jako zmiňovaná funkce pro čtení, jsou předávány tři parametry. Prvním z parametrů je adresa, od které je požadováno zapisovat data do EEPROM. Druhým z parametrů je počet datových bytů, které se mají zapsat a posledním parametrem je ukazatel na byte, který označuje první byte v paměti mikroprocesoru, od kterého se mají data za sebou posílat pro uložení do EEPROM.



Obrázek 28: Funkce WriteIntEEPROM(..) pro zápis více bytů do EEPROM paměti mikroprocesoru

Vývojový diagram popisované funkce je zobrazen na Obrázek 28. Jak již bylo uvedeno, funkce je téměř shodná s funkcí pro čtení více bytů z vnitřní EEPROM mikroprocesoru, ale namísto volání funkce pro čtení jednoho bytu do paměti se volá funkce pro zápis jednoho bytu do této paměti.

5. Měření kapacity a teploty A/D převodníkem AD7745

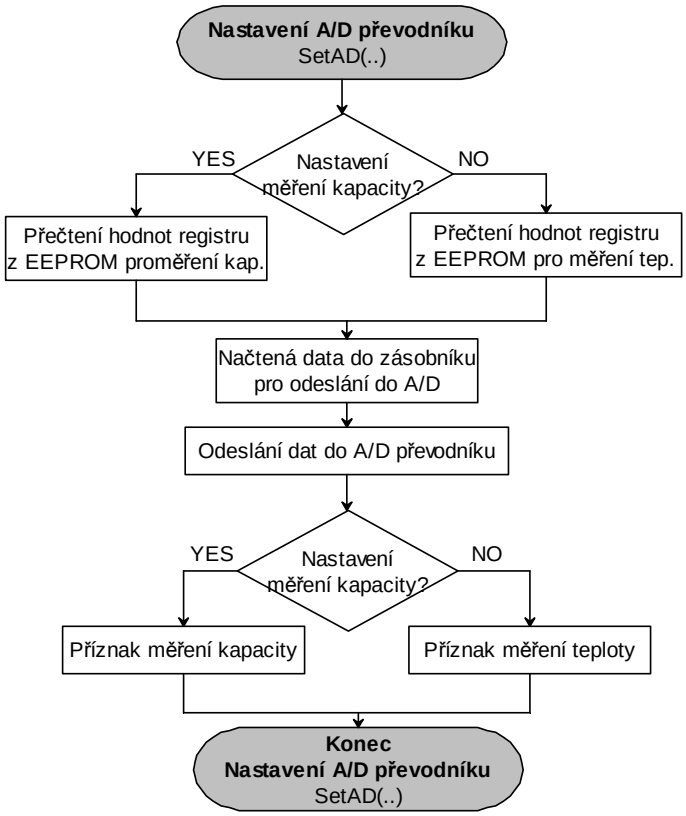
Celý kapacitní snímač tlaku je založen na kapacitním čidle připojeném k A/D převodníku, který převádí kapacitu čidla na 24 bitové číslo. Čidlo má membránu, která se se změnou tlaku deformuje a mění se tak kapacita čidla. A/D převodník je připojen k mikroprocesoru, který získává data z A/D převodníku po sériové komunikační lince IIC. Mikroprocesor si tak může vyčíst 24bitovou hodnotu kapacity čidla, nebo teplotu měřenou v tomto případě přímo A/D převodníkem AD7745.

5.1 Nastavení A/D převodníku AD7745

Pro nastavení A/D převodníku slouží vnitřní registry, do kterých můžeme hodnoty nahrávat, nebo je číst pomocí zmiňované sériové komunikace IIC. Převodník má několik hlavních registrů, které se nastavují během vykonávání programu.

Prvním registrem je příznakový registr „Status“ (adresa 0x00). V tomto registru jsou uloženy informace zejména o dokončení některého z převodů (kapacita/teplota). Za tímto registrem je na adrese 0x01 – 0x03 hodnota převodu kapacity (24bitová). Následuje další skupina tří registru (adresa 0x04 – 0x06), ve kterých je uložena hodnota převodu teploty. Obě hodnoty jsou seřazeny od nejvyššího bytu, po nejnižší. Například pro kapacitní převod je na adrese 0x01 uloženo nejvýznamnějších 8 bitů a na adrese 0x03 zase nejméně významných 8bitů. Dále následují registry pro nastavení převodů. Prvním z nich je registr „Cap Setup“ na adrese 0x07, do kterého zapisujeme údaje pro nastavení A/D převodu kapacitního kanálu. Stejně tak můžeme i převod kapacitního kanálu zakázat. Obdobně jako pro nastavení převodu kapacity, slouží další registr pro nastavení převodu teploty „VT Setup“. Registr pro nastavení teploty je na adrese 0x08. Obsahuje taktéž bit pro zakázání převodu teploty, nebo pro typ snímače teploty (umožňuje připojit externí snímač teploty, v tomto případě se využívá vnitřního snímače teploty). V dalším registru „EXC Setup“ se nastavuje buzení kapacitního kanálu a jeho úrovně (jsou voleny úrovně Low = 0 a High = V_{DD} a napětí na kapacitním čidle je $\pm V_{DD} / 2$. Následující registr „Configuration“, na adrese 0x0A, slouží, jak již název napovídá, k nastavení A/D převodníku. V registru lze nastavit doby (odpovídající přesnosti), s jakou je požadováno měření jak kapacitního, tak teplotního kanálu. Pro měření teploty je převod nastaven na 122.1ms, tedy nejpřesnější možný výsledek, stejně jako pro měření kapacity, který trvá 109.6ms. Posledními třemi bity tohoto registru si uživatel vybírá, v jakém režimu bude chtít měřit, jestli ve spojitém režimu, nebo spouštět převod „ručně“ vždy po dokončení předchozího převodu apod. V programu je navržen princip spojitého měření, kdy si vždy po dokončení převodu A/D převodník zažádá o obsluhu pomocí signálu RDY, který je přiveden k mikroprocesoru na pin RB0. Po změně logického stavu tohoto

pinu se vyvolá v mikroprocesoru přerušení, a pokud nekomunikuje mikroprocesor pomocí HART protokolu, vyčte se právě převedená hodnota z A/D převodníku. Pokud zrovna komunikuje, vyčtení nové hodnoty se provede, jakmile se komunikace dokončí. Pokud se číslo z A/D převodníku nevyčte, nic se nestane. Převodník vynuluje signál RDY, nastaví novou hodnotu převodu a opět nastaví signál RDY (signál RDY je invertovaný – aktivní v logické nule). Poslední dva registry A/D, které se budou nastavovat, jsou registry „CAP DAC A“ a „CAP DAC B“. Registry slouží pro nastavení rozsahu A/D převodníku tak, aby byl jeho rozsah plně využit. Hodnoty do těchto registrů se získávají kalibrací snímače.



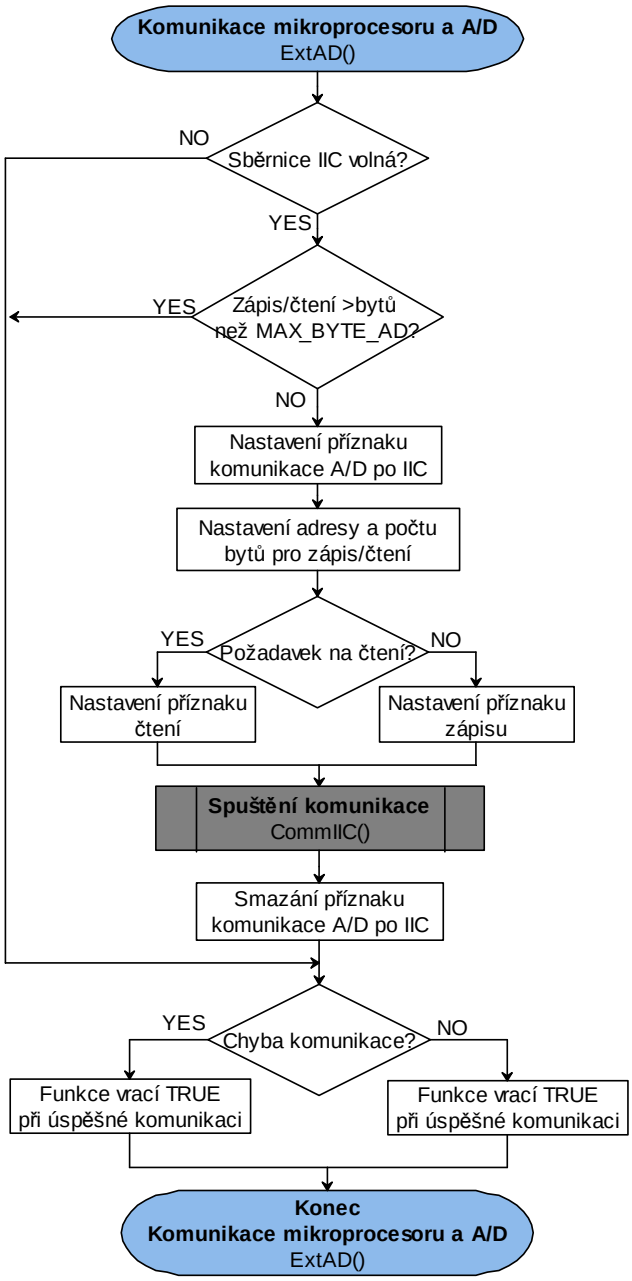
Obrázek 29: Funkce pro nastavení registrů A/D převodníku AD7745

Hodnoty všech nastavovaných registrů A/D převodníku jsou uloženy v externí EEPROM. Po zapnutí snímače se nejprve inicializuje sériová komunikační linka a nastaví se pin RB0 pro detekci změny logického stavu a vyvolání přerušení pomocí signálu RDY. Teprve až je dokončena veškerá inicializace, může se provést nastavení registrů A/D převodníků. Pro nastavení těchto registrů slouží funkce SetAD(..), které se předává jako parametr jeden bit, který slouží pro zjištění, zda se bude nastavovat měření teploty (0), či měření kapacity (1). Tato funkce je ukázána na vývojovém diagramu na Obrázek 29.

Jak již bylo zmiňováno dříve, odesílání/čtení dat do/z A/D převodníku probíhá po sériové komunikační lince IIC. Na sběrnici IIC komunikují tedy dvě zařízení EEPROM a A/D převodník. Způsob komunikace mezi A/D převodníkem a mikroprocesorem bude popsán v následující kapitole.

5.2 Komunikace mezi PIC16F886 a A/D převodníkem AD7745

Pro zajištění komunikace mezi mikroprocesorem a externím A/D převodníkem AD7745 bylo potřeba vytvořit funkci, která přistupuje k funkci obsluhující komunikaci sběrnice IIC. Tato funkce je velmi podobná funkci pro obsluhu komunikace mezi mikroprocesorem a externí EEPROM ExtEEPROM(). Před voláním této funkce je zapotřebí ale nejprve načíst do struktury „IICComm“ pro řízení sériové komunikace, adresu, počet datových bytů pro přenos a příznakový bit pro požadavek čtení nebo zápisu. Tyto data se nejprve načtou do prvního zásobníku a až v případě, že je sběrnice volná, se překopírují do zásobníku pro odeslání. Mezi funkcemi ExtEEPROM() a ExtAD() jsou dva rozdíly. První rozdíl je, že se v případě volné sběrnice, nastaví příznak probíhající komunikace mezi mikroprocesorem a A/D převodníkem v příznakovém bytu „ChipSel“ ve struktuře „IICComm“, namísto příznaku komunikace EEPROM. Druhým rozdílem je testování počtu datových bytů pro zápis/čtení do/z A/D převodníku. Maximální počet datových bytů je omezen konstantou “ MAX_BYTE_AD“, která je přednastavena na 6 bytů, z důvodu, že se při komunikaci s A/D převodníkem AD7745 odesílá/přijímá maximálně 6 datových bytů a to právě při konfiguraci jeho registrů. Po nastavení všech příznaků potřebných pro komunikaci IIC, se volá funkce CommIIC(), která spustí samotnou komunikaci mezi nastaveným

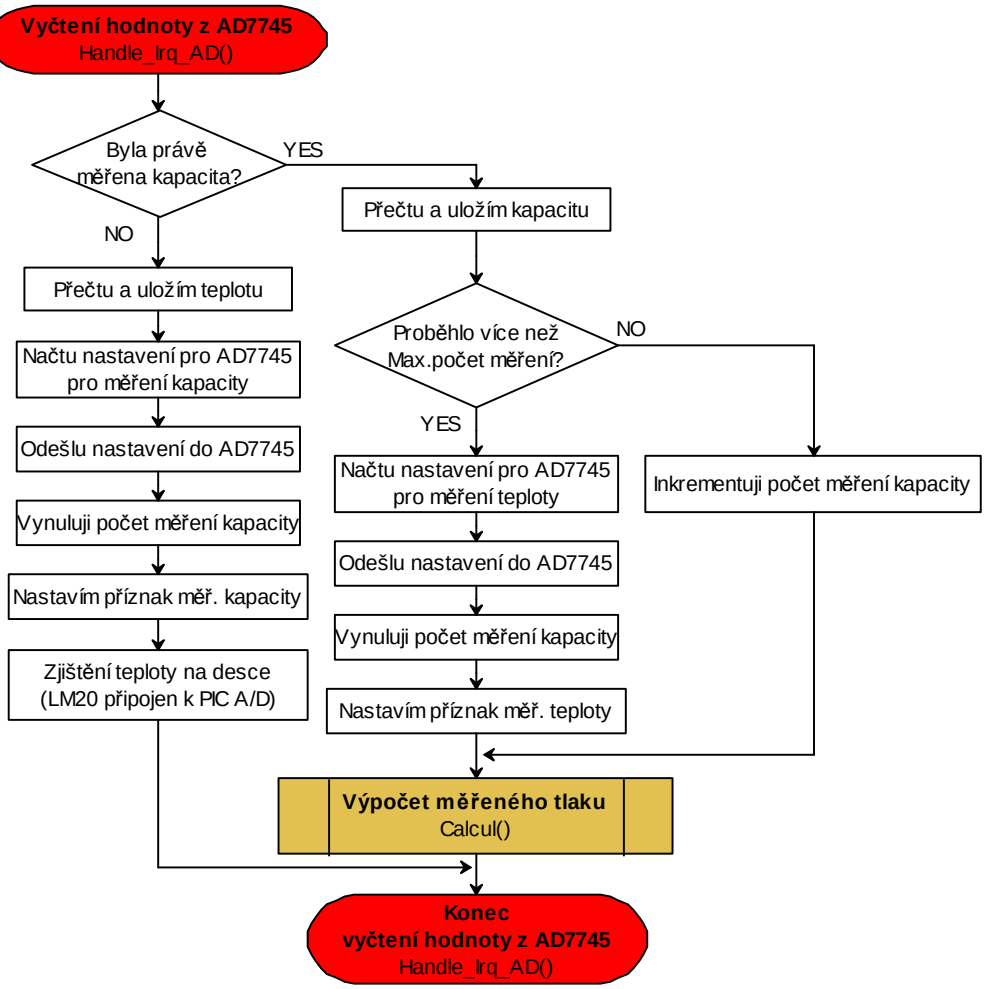


Obrázek 30: Komunikace mikroprocesoru a A/D převodníku AD7745

zařízením (A/D převodníkem) a mikroprocesorem. Funkce CommIIC() je zobrazena na Obrázek 23, vývojový diagram funkce ExtAD() pro zajištění a přístup k sériové komunikační lince IIC je zobrazen na Obrázek 30.

5.3 Řešení obsluhy A/D převodníku AD7745

Obsluha A/D převodníku je spuštěna vždy, když si o ni sám A/D převodník AD7745 zažádá pomocí signálu RDY, jak je popisováno dříve. Tento signál je přiveden do mikroprocesoru, kde se hlídá jeho stav a při jeho změně se vyvolá přerušení. Pokud mikroprocesor PIC16F886 právě nekomunikuje prostřednictvím protokolu HART, je schopen okamžitě zareagovat na přerušení a ihned volá funkci Handle_Irq_AD() pro obsluhu A/D převodníku.



Obrázek 31: Vyčtení hodnot z A/D převodníku AD7745

Po zapnutí snímače je automaticky vždy první změřena teplota a poté je snímač nastaven pro změření kapacity. To z důvodu teplotní kompenzace čidla, kdy se teplota dosazuje do polynomů pro získání lineární charakteristiky proudové smyčky. Jak si

lze všimnout z Obrázek 31, kde je zobrazen vývojový diagram pro obsluhu A/D převodníku, pokud je příznak pro měření teploty, přečte se z A/D převodníku 24bitová informace o teplotě a převodník se okamžitě přestaví na měření kapacity. Při příštím volání funkce Handle_Irq_AD() se přečte kapacita čidla ve formě 24bitové informace a provede se zjištění, kolikrát po sobě byla již kapacita měřena.
V programu je konstanta „MAX_CAP_MEAS“, která říká, kolikrát se bude opakovat měření kapacity. Pokud je počet opakování menší než daná konstanta, provede se pouze inkrementace počtu měření kapacity a následně se volá funkce Calcul(), která má na starosti veškeré výpočty snímače (upřesněno dále). Je-li naopak počet opakování větší než tato konstanta, provede se přestavení A/D převodníku pro měření teploty. Po změření teploty se A/D převodník opět ihned přestavuje pro měření kapacity. Tímto způsobem je provedeno vyčítání kapacity a teploty mikroprocesorem z A/D převodníku AD7745.

Při obsluze externího A/D převodníku pomocí funkce Handle_Irq_AD(), při měření teploty, je po dokončení konfigurace tohoto převodníku měřena ještě další teplota. Tato teplota se měří přímo na plošném spoji v blízkosti mikroprocesoru, pomocí integrovaného obvodu LM20. Tento obvod je připojen na pin RB2, který je při inicializaci mikroprocesoru nastaven pro A/D převod. Vnitřní A/D převodník mikroprocesoru PIC16F886 je 10bitový, převedená hodnota se ukládá do registrů ADRESH a ADRESL (zarovnání 10bitové hodnoty je dle nastavení konfiguračního registru ADCON1 mikroprocesoru). Pro inicializaci pinu RB2 mikroprocesoru slouží funkce PIC_AD_Init(), která nastaví tento pin pro A/D převod, nastaví referenční napětí pro převod a zarovnání výsledku ve zmiňovaných registrech ADRESH a ADRESL. Dále nastaví rychlost převodu odpovídající frekvenci externího oscilátoru s frekvencí 8MHz.

Při měření teploty pomocí obvodu LM20 se již využívá funkce Temp2Read(), která spustí A/D převod, počká na jeho dokončení a poté načte výslednou 10bitovou hodnotu do vnitřní proměnné. Tato hodnota odpovídá napětí měřenému na obvodu LM20, které je závislé na teplotě. Dále funkce tuto načtenou hodnotu převádí na teplotu ve °C pomocí následujícího výpočtu.

$$Teplota\ [°C] = \frac{\left(\left(\frac{AD\_Hodnota}{Rozsah\ A/D} \times Ref.napětí\right) - 1{,}8528\right)}{-0{,}01179}$$

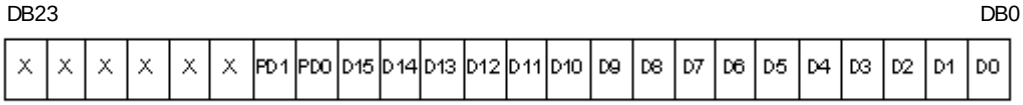
Kde AD_Hodnota je 10bitová hodnota načtená z A/D převodníku, rozsah A/D převodníku je $2^{10} = 1024$, a referenční napětí je 3V.

6. Nastavení hodnoty proudové smyčky snímače

Nastavení hodnoty proudové smyčky se provádí tak, že k mikroprocesoru PIC16F886 je připojen D/A převodník DAC8551, který převede číslo poslané pomocí 3vodičové sériového komunikačního rozhraní na napěťovou úroveň. K D/A převodníku jsou následně připojeny obvody pro převod napěťové úrovně na proud protékající proudovou smyčkou.

6.1 Komunikace s DAC8551

Pro odeslání čísla do D/A převodníku se využívá 3vodičové sériové komunikační sběrnice. Sběrnice má jeden datový vodič D_{IN}, jeden synchronizační vodič SYNC a hodiny SCLK. Na datovou sběrnici mikroprocesor sériově vysílá 24bitovou hodnotu. Význam jednotlivých bitů je uveden na Obrázek 32.



Obrázek 32: Význam bitů při sériovém přenosu mezi mikroprocesorem PIC16F886 a D/A převodníkem DAC8551

V nejvýznamnějším bytu posílaném sériovým komunikačním rozhraním jsou pouze dva bity (PDO a PD1), které jsou sledovány D/A převodníkem. Jejich hodnota nastavuje operační mód převodníku. Standardně jsou oba bity nastaveny na hodnotu logická nula (Normální mód). Nižší 2byty (DB15-DB0) již přenášejí hodnotu čísla pro převod na napětí, které se objeví na výstupu tohoto D/A převodníku. Výstupní napětí se dá určit pomocí vztahu:

$$V_{OUT} = \frac{D_{IN}}{10^{16}} \times V_{REF}$$

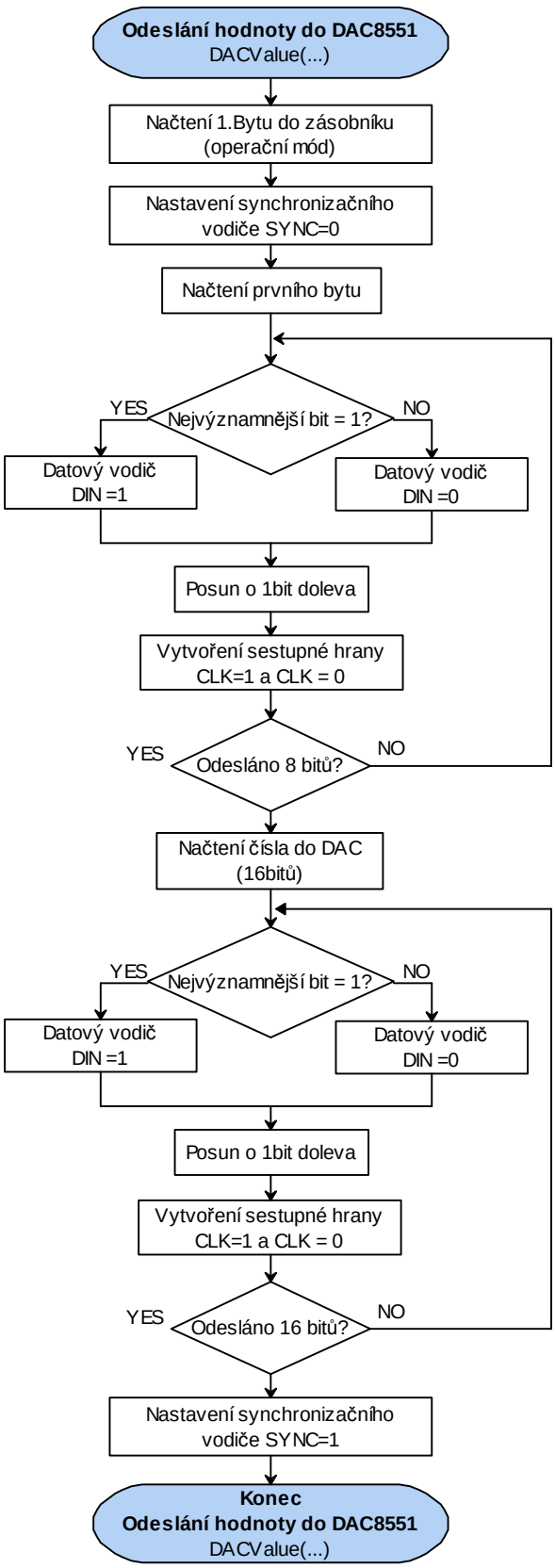
V uvedeném vztahu je V_{OUT}[V] výstupní napětí D/A převodníku, D_{IN} je hodnota čísla přivedeného na vstup po sériovém rozhraní v podobě 24bitů, 10^{16} reprezentuje rozsah D/A převodníku a konečně V_{REF}[V] je referenční napětí pro D/A převodník.

Pro odeslání hodnoty D_{IN} do D/A převodníku DAC8551 složí funkce DACValue(…), jejímž parametrem je právě 16bitová hodnota, kterou chceme odeslat do převodníku. Funkce automaticky přidá před odesílanou hodnotu mód převodníku, jak je zmíněno výše, dva nulové bity a dalších šest bitů, na kterých nezáleží (funkce přidá dalších 6 nulových bitů). Po uložení hodnoty do zásobníku se nejprve vynuluje synchronizační bit (opačná úroveň) a postupně se začnou hodnoty nastavovat bit po bitu na datový vodič D_{IN} sériového rozhraní. Vždy po nastavení nové hodnoty na datovém vodiči se provede nastavení vodiče SCLK=1 a následně SCLK=0 pro

vytvoření sestupné hrany, na kterou D/A převodník reaguje. Tímto způsobem se vyšle všech 24bitů a po dokončení odesílání se nastaví zpět na hodnotu logická jedna synchronizační vodič SYNC.

Vývojový diagram funkce provádějící odeslání čísla do D/A převodníku DAC8551 je zobrazen na Obrázek 33. Funkce je rozdělena na dvě obdobné části, kdy v první části se odesílá nejvýznamnějších 8bitů, v kterém je obsažen pouze režim D/A převodníku. Po dokončení odesílání těchto 8bitů funkce načte parametr, který reprezentuje 16bitové číslo, které se má odeslat po sériové komunikační lince. Následuje stejné odeslání 16bitů stejným způsobem jako v první části funkce.

Funkce pro odeslání hodnoty z mikroprocesoru do D/A převodníku je volána vždy po aktualizaci hodnoty měřeného tlaku. Funkce se tedy volá ihned, jakmile A/D převodník dokončí převod kapacity na číslo a vyčte se číslo z A/D převodníku pomocí funkce Handle_Irq_AD(), přesně tak, jak je zobrazeno na vývojovém diagramu hlavní smyčky programu na Obrázek 14.



Obrázek 33: Funkce pro odeslání čísla do DAC8551

7. Výpočty prováděné mikroprocesorem

Pro veškeré výpočty spojené s funkcí kapacitního snímače tlaku je vytvořena funkce Calcul(). Tato funkce se volá vždy po získání nové hodnoty kapacity z A/D převodníku ve funkci Handle_Irq_AD(). Pro výpočty hodnot polynomů se využívá funkce CompPolynom().

7.1 Výpočet hodnoty polynomu

Pro dosazení hodnoty obecně proměnné X do polynomu n-tého řádu se využívá funkce CompPolynom(..), které se jako parametr předává identifikace polynomu pro dosazení a jako druhý parametr se předává ukazatel na hodnotu X typu double pro dosazení do polynomu. Tato funkce vrací přímo výsledek vyčísleného polynomu při dosazené hodnotě X předané jako parametr. Všechny koeficienty polynomu jsou uloženy v externí EEPROM, včetně řádu polynomu. Funkce si nejprve zjistí řád tohoto polynomu a poté postupně vypočítá hodnoty jednotlivých členů polynomu, kdy je postupně sumuje do pomocné proměnné. Jakmile vypočte i poslední člen a sečte jej s dosavadní sumou všech členů, vrací se tato hodnota jako výsledek dosazení polynomu. Podrobnější princip funkce je zobrazen na Obrázek 45 v příloze.

7.2 Postup při výpočtech v mikroprocesoru PIC16F886

Ve funkci Calcul() se nejprve vypočítává teplota T_1, která se převádí z 24bitového čísla na hodnotu ve °C pomocí následujícího vztahu:

$$T_1[°C] = \frac{Hodnota\_AD}{2048} - 4096$$

Kde Hodnota_AD je 24bitová hodnota teploty vyčtená z AD7745. Následují výpočty pro teplotní závislost 24bitové hodnoty kapacity (označena jako „dp“) vyčtené taktéž z externího A/D převodníku. Nejprve se určí hodnota polynomu dpot(T_1), která udává teplotní závislost veličiny dp pro dolní mez tlaku a následně se vypočte teplotní závislost této veličiny pro horní mez tlaku vyjádřením polynomu dpfst(T_1). Obě hodnoty se získají dosazením teploty T_1 do polynomu dpot a dpfst, které jsou (0. až 4.)řádu. Jakmile jsou vypočteny hodnoty těchto polynomů, může se vypočítat na jejich základě a hodnotě kapacity z A/D převodníku normovaná teplotně kompenzovaná veličina „dpt“, která je v rozsahu < 0 ; 1 >, dle následujícího vztahu:

$$dpt = \frac{((dp - dpot(T_1))}{((dpfst(T_1) - dpot(T_1))}$$

Závislost kapacitního čidla na tlaku není ideálně lineární. Pro získání lineární závislosti se zjistí závislost normované teplotně kompenzované veličiny na tlaku,

následně se například proloží tato závislost polynomem 4. řádu a získané koeficienty se nahrají do externí paměti EEPROM. Tím při výpočtech dostaneme lineární závislost hodnoty dpt na měřeném tlaku.

Po vypočtení hodnoty normované teplotně kompenzované veličiny „dpt“ se tato hodnota dosadí do polynomu „lin“ pro získání hodnoty digitálního tlaku v jednotkách bar. Po získání digitálního tlaku lze určit digitální proud v rozmezí 4…20mA. Pro výpočet digitálního proudu „I_{DIG}“ musíme znát uživatelem nastavitelnou dolní a horní mez tlakového rozsahu snímače označovanou jako PVLR a PVUR. Obě tyto hodnoty musí být v intervalu <SLL;SUL> kde SLL je dolní mez tlakového rozsahu čidla a SUL je horní mez tlakového rozsahu čidla. Rozdíl hodnot (PVUR-PVLR) tvoří tlakový rozsah snímače SPAN a ten musí být větší, než je minimální rozsah snímače MS. Hodnota PVLR může být větší, než hodnota PVUR a tím je dán reverzní stav. Výpočet digitálního proudu pro normální stav (PVUR > PVLR) je:

$$I_{DIG} = 4 + 16 * \left(\frac{PVUR - lin}{PVUR - PVLR}\right)$$

Pro reverzní režim snímače (PVUR < PVLR), se digitální proud určí jako:

$$I_{DIG} = 4 + 16 * \left(\frac{lin - PVLR}{PVUR - PVLR}\right)$$

Získáme-li výpočtem hodnotu digitálního proud I_{DIG}, je potřeba zjistit číslo, které je odesláno do D/A převodníku, čímž se nastaví odpovídající proud proudové smyčky. Jelikož je D/A převodník závislý na teplotě, je zapotřebí při výpočtu tuto závislost vzít v úvahu. Proto se při výpočtu čísla do D/A převodníku označeného „N“ vyskytují další dva polynomy „J“ a „K“. Oba polynomy mohou být (0. až 3.)řádu. Výpočet čísla do D/A převodníku je dle vztahu:

$$N = J(T_2) + K(T_2) * (I_{DIG} - 4)$$

Kde polynom J vyjadřuje teplotní závislost ofsetu D/A převodníku a polynom K teplotní závislost směrnice D/A převodníku. Teplota T_2 označuje teplotu změřenou na desce plošného spoje pomocí interního A/D převodníku mikroprocesoru PIC16F886 a k němu připojeného obvodu LM20.

8. Závěr

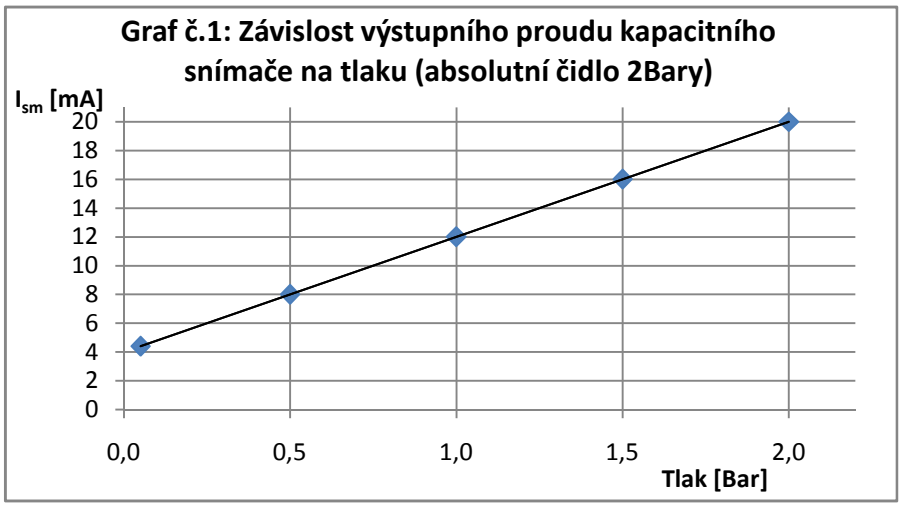
Cílem celé práce bylo navrhnout a implementovat programové vybavení snímače s protokolem HART pomocí programovacího jazyka C do mikroprocesoru PIC16F886, který řídí činnost celého kapacitního snímače tlaku. Mimo protokolu HART je navržena část, která převádí měřenou kapacitu čidla A/D převodníkem na informaci o právě měřeném tlaku a poté ji zobrazuje na proudové smyčce 4-20 mA pomocí D/A převodníku DAC8551. V případě dotazu lze i pomocí HART protokolu odeslat odpověď s hodnotou měřeného tlaku a teploty. Bohužel implementovaný protokol HART je svou podstatou relativně obsáhlý a programovat jej do mikroprocesoru s programovou pamětí 8 kB pomocí programovacího jazyka C je obtížné a v průběhu návrhu vzniklo několik problémů s pamětí, ať již programovou či datovou, z toho důvodu museli být některé části programu při testování vynechány a program se tak testoval po částech.

Rychlost měření kapacitního snímače je do značné míry ovlivněna tím, zda snímač komunikuje pomocí protokolu HART. Jelikož se využívá přerušení pro veškerou komunikaci, musí být upřednostněno v programu přerušení od komunikace HART před vyčtením nové hodnoty z A/D převodníku, která se vyčte, až jakmile je aktuální komunikace HART dokončena. Pokud by snímač byl příliš pomalý a nebylo prioritou zcela přesné měření, je možnost rychlost snímače zvýšit tím, že se změní konfigurace A/D převodníku AD7745 (lze i technologickým příkazem HART), pro rychlejší převod měřené kapacity.

Po základním nastavení snímače a jeho kalibraci byla získána následující charakteristika snímače (Graf č. 1) zobrazující závislost hodnoty proudové smyčky na tlaku. Při měření závislosti bylo použito tzv. absolutní čidlo tlaku s rozsahem 2 Bary. Absolutní čidlo tlaku měří od absolutní nuly, tedy od vakua, na rozdíl od čidla relativního, které měří tlak vzhledem k tlaku atmosférickému.

p_s[Bar]	I_m[mA]	p_m[Bar]
0,0500	4,4010	0,0503
0,5000	8,0010	0,5002
1,0000	12,0010	1,0007
1,5000	16,0020	1,5002
2,0000	20,0010	2,0002

Tabulka 2: Závislost výstupního proudu snímače na tlaku a hodnota měřená pomocí HART rozhraní



Princip celého návrhu snímače by měly objasnit uvedené vývojové diagramy, které shrnují principy a funkce jak implementovaného protokolu HART do mikroprocesoru PIC16F886, tak i princip měření a komunikace mikroprocesoru s periferiemi. Jak je popsáno výše, většina komunikace je založena na systému přerušení, které vznikají například při příjmu/odesílání dat prostřednictvím UART rozhraní, či komunikaci mezi externí EEPROM a mikroprocesorem.

Zdrojové kódy v této práci nejsou vůbec uvedeny, jelikož jsou relativně obsáhlé a byly by zde špatně čitelné. Proto jsou zde uvedeny pouze vývojové diagramy používaných funkcí a principů obecně. Kompletní projekt napsaný v prostředí programu [1] je přiložen na CD ve složce „Zdrojove_kody“.

9. Seznam použitých zdrojů:

[1] BURKHARD, Mann. C pro mikrokontroléry. 1. vyd. Praha : BEN, 2003. 280 s. ISBN 80-7300-077-6.

[2] ŠALOUN, P. Programovací jazyk C pro zelenáče. 2. přeprac. vyd. Praha: Neocortex, 2003. 208 s. ISBN 80-86330-08-7

[3] FOJTÍK, David. Operační systémy a programování. 1. vyd. Ostrava : VŠB - Technická univerzita Ostrava, 2007. 305 s. Dostupný z WWW: http://www.elearn.vsb.cz/archivcd/FS/OSP/Operacni_systemy_a_programovani.pdf. ISBN 978-80-248-1510-7.

[4] HART Communication Foundation. FSK Physical Layer Specification. Rev.8.1. 1999

[5] HART Communication Foundation. Data Link Layer Specification. Rev.8.0. 2001

[6] HART Communication Foundation. Command Summary Specification. Rev.8.0. 2001

[7] HART Communication Foundation. Universal Command Specification. Rev.6.0, 2001

[8] HART Communication Foundation. Common Practice Command Specification. Rev.8.0. 2001

[9] HART Communication Foundation. Common Tables Specification. Rev.20.0. 2009

[10] HART Communication Foundation. Command Specific Response Code Definitions. Rev.5.0. 2001

[11] MICROCHIP. MPLAB IDE User's Guide. Microchip Technology. 2006

[12] MICROCHIP. PIC16F882/883/884/886/887 Data Sheet. Microchip Technology. 2007

[13] B KNUDSEN DATA. CC5X – C Compiler for the PICmicro Device, User's Manual, Ver.3.4. 2006

[14] ANALOG DEVICES. AD7745/AD7746 24-Bit Capacitance-to-Digital Converter with Temperature Sensor, Data Sheet, Ver.0. 2005

[14] MICROCHIP. 24AA02/24LC02B – 2K I^2C Serial EEPROM, Data Sheet, Rev.J. 2009

[15] TEXAS INSTRUMETS. DAC8551 – 16-bit, Ultra-low Glitch, Voltage Output digital-to-analog converter, Data Sheet, Ver.2, 2006

10. Seznam použitých zkratek a symbolů:

HART	Highway Addressable Remote Transducer
EEPROM	Electrically Erasable Programmable Read-Only Memory
UART	Universal asynchronous receiver/transmitter
IIC (I^2C)	Inter-Integrated Circuit
A/D	Analog / Digital
DAC	Digital / Analog Converter
CRC	Cyclic redundancy check
FSK	Frequency-shift keying
XOR	Exclusive disjunction (Exclusive or)
ACK	Acknowledgment code

11. Použitý software:

[1] MPLAB IDE v.8.36

[2] B Knudsen Data CC5X v.3.3 A

[3] Diagram Designer v.1.2

[4] HHD Free Serial Port Monitor v3.31

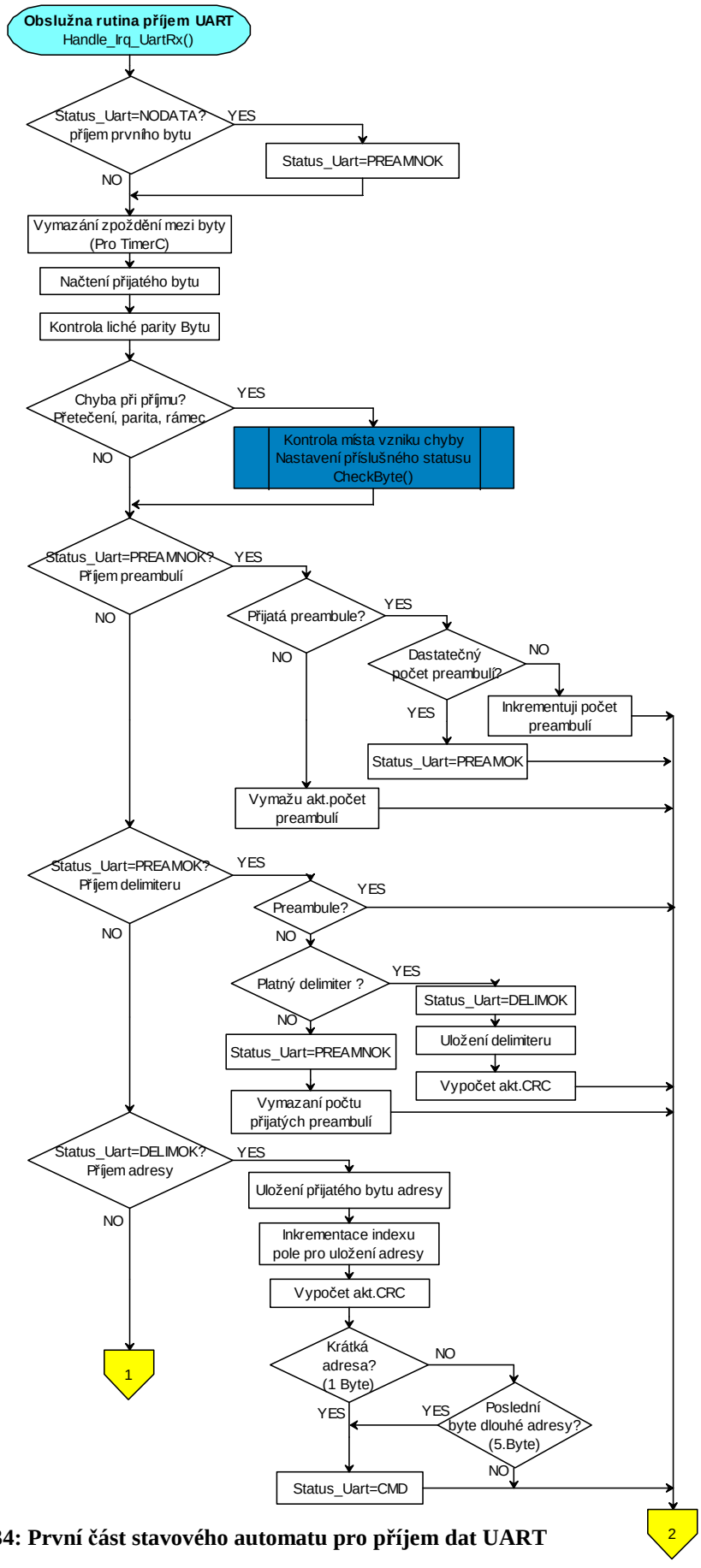
[5] Config v2.1

12. Seznam adresářů na přiloženém CD

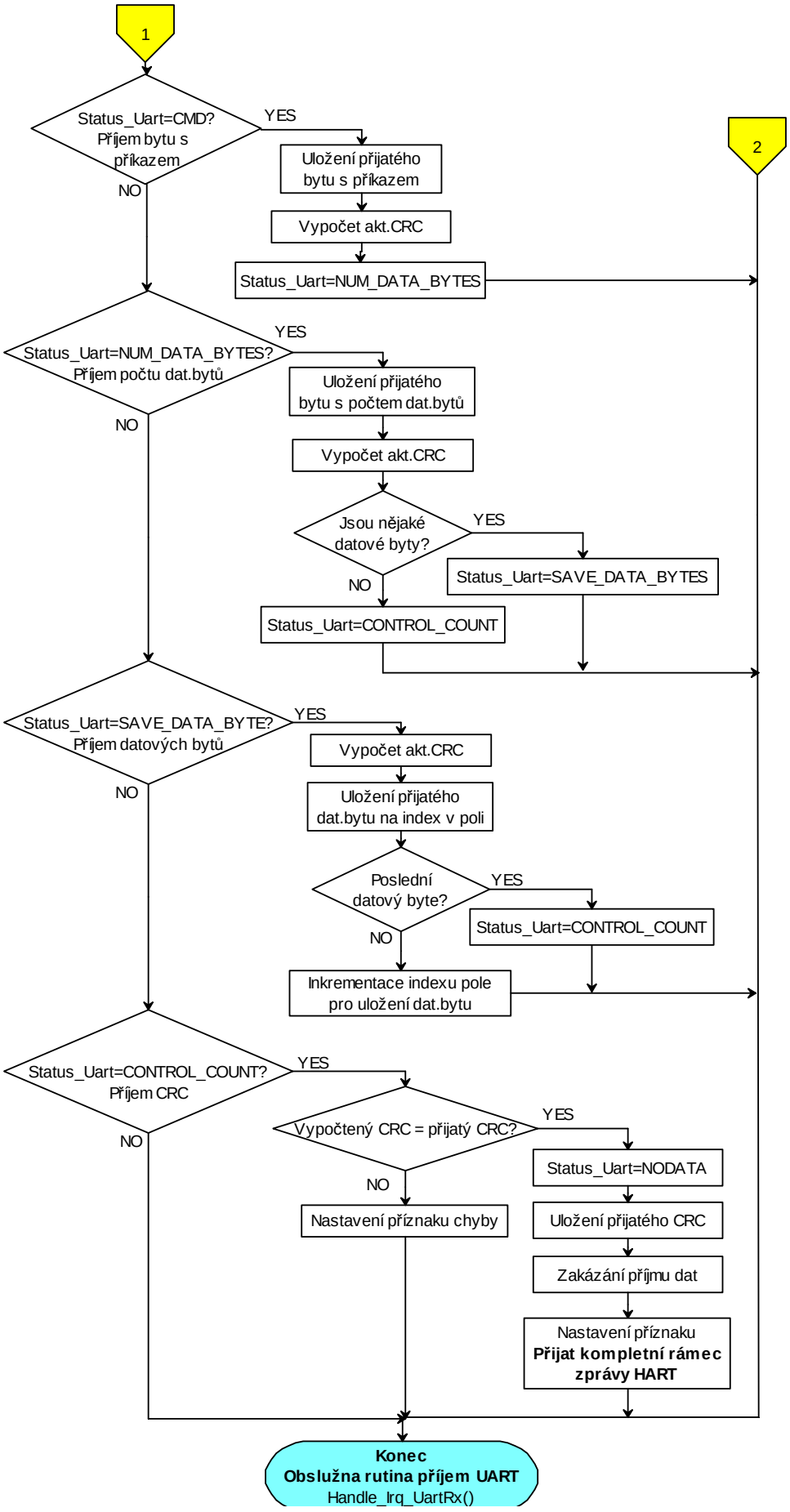
\Vyvojove_diagramy	Vývojové diagramy v programu [3]
\Zdrojove_kody	Zdrojové kódy implementace protokolu HART projekt napsaný v programu [1]
\Cozik_Ondrej.pdf	Bakalářská práce (tento dokument)

13. Přílohy

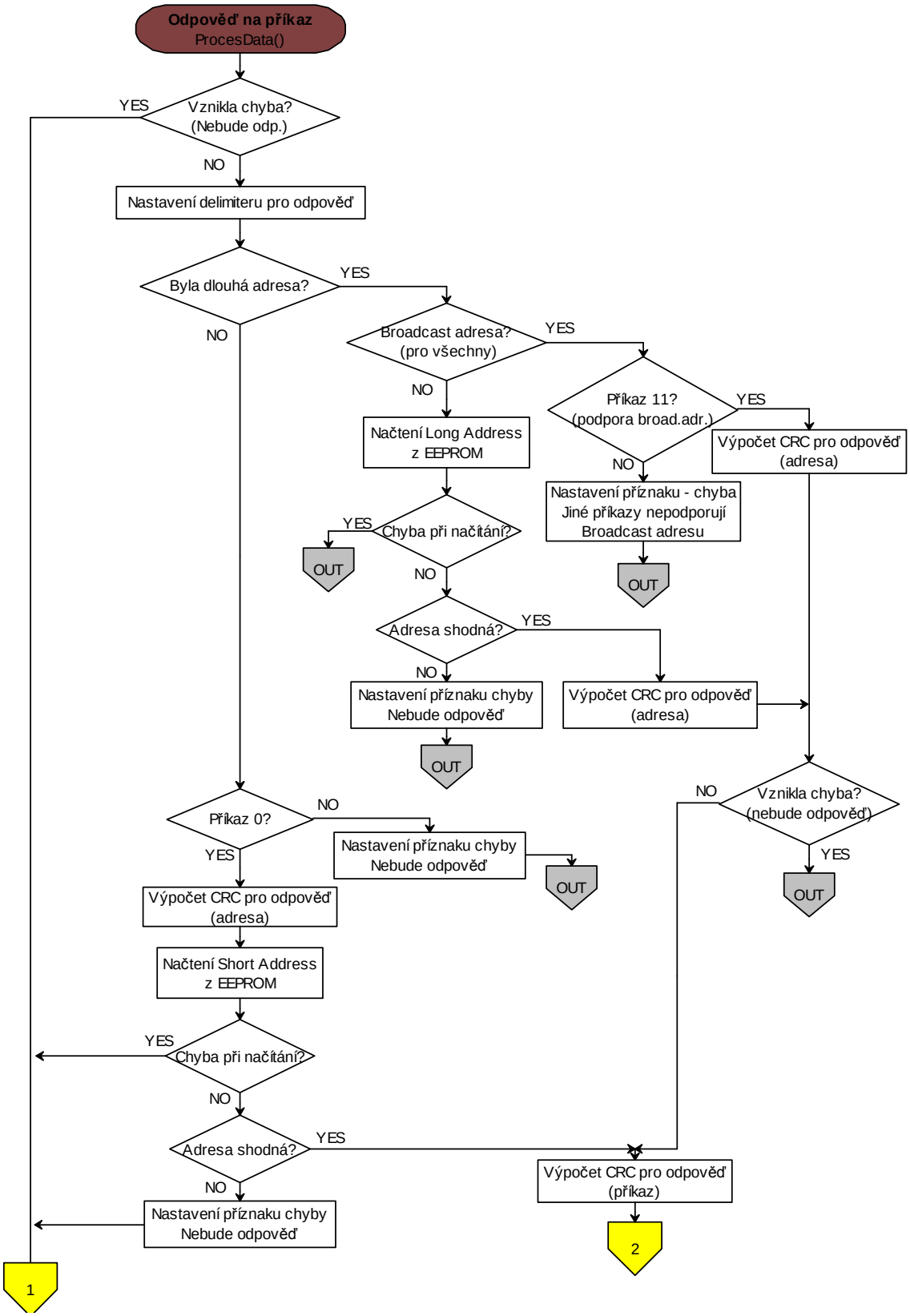
13.1 Vývojové diagramy



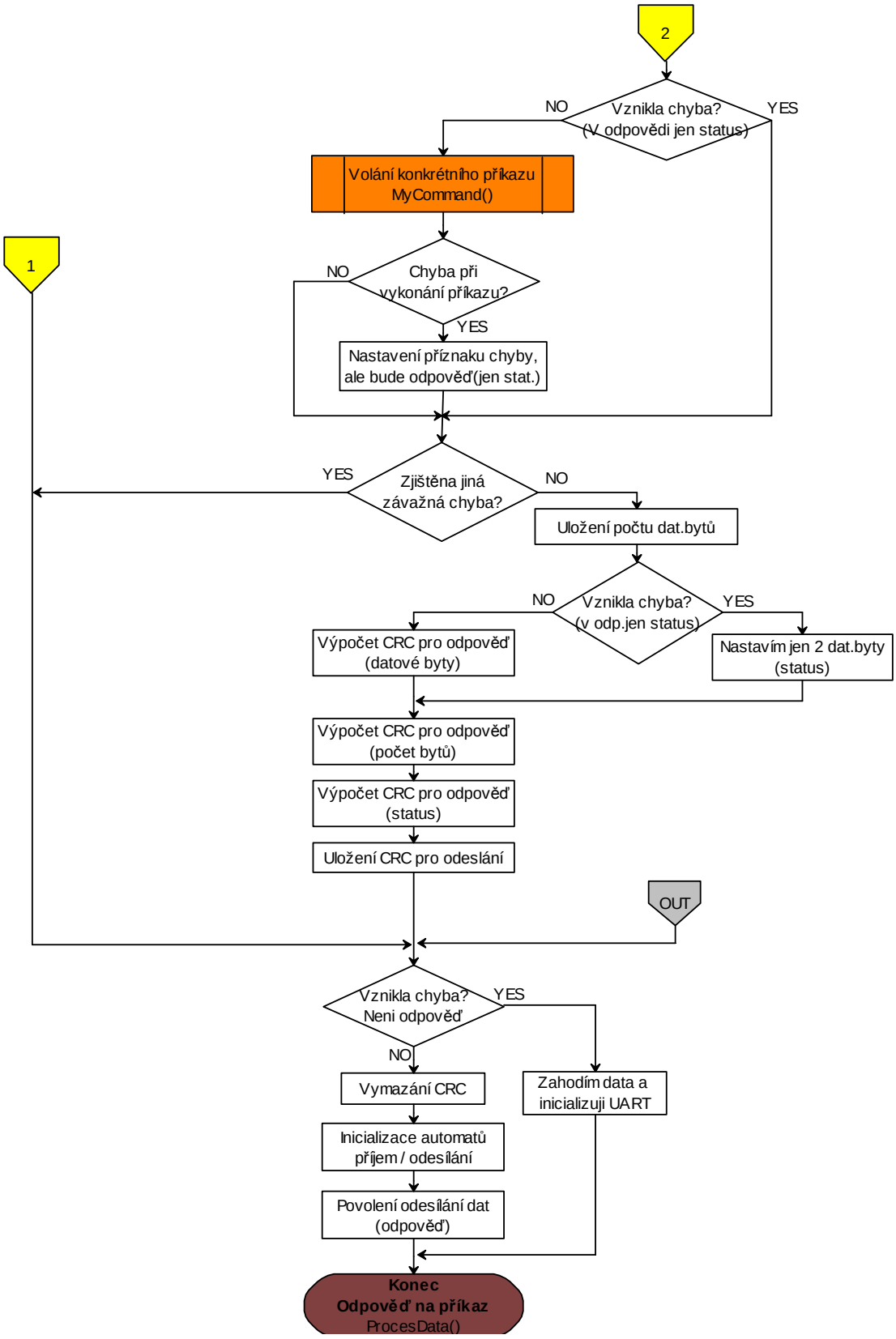
Obrázek 34: První část stavového automatu pro příjem dat UART



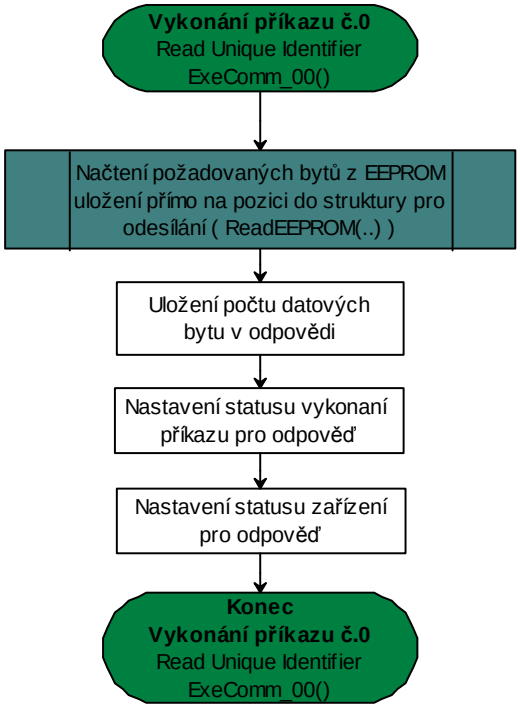
Obrázek 35: Druhá část stavového automatu pro příjem dat UART



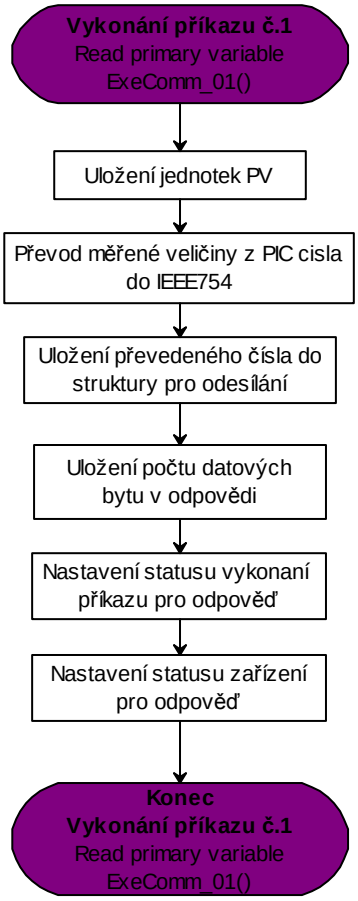
Obrázek 36: První část funkce ProcesData() pro odpověď na přijatá data



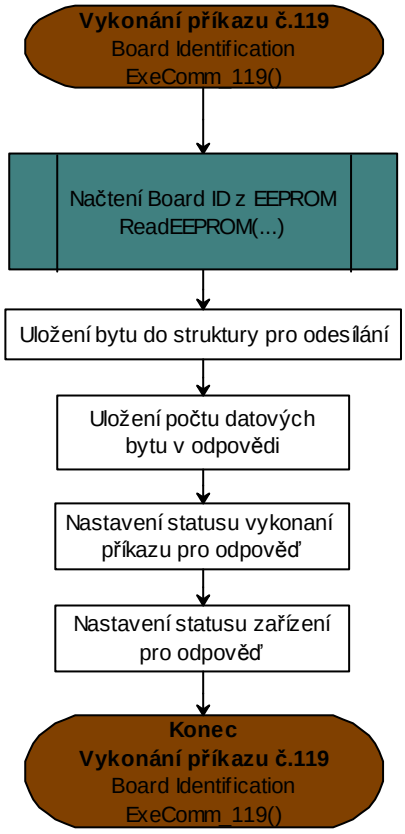
Obrázek 37: Druhá část funkce ProcesData() pro odpověď na přijatá data



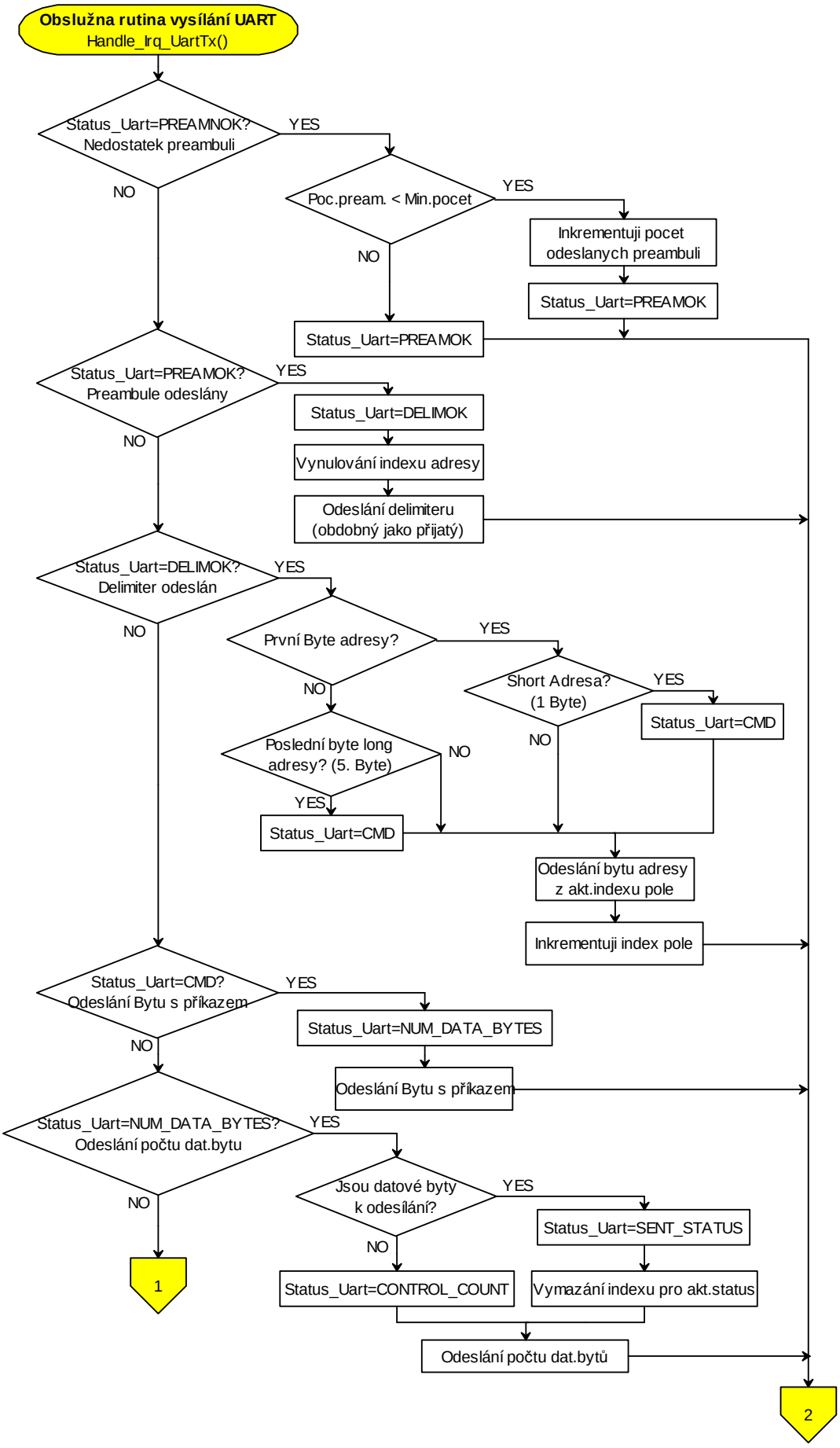
Obrázek 38: Diagram pro HART příkaz 0



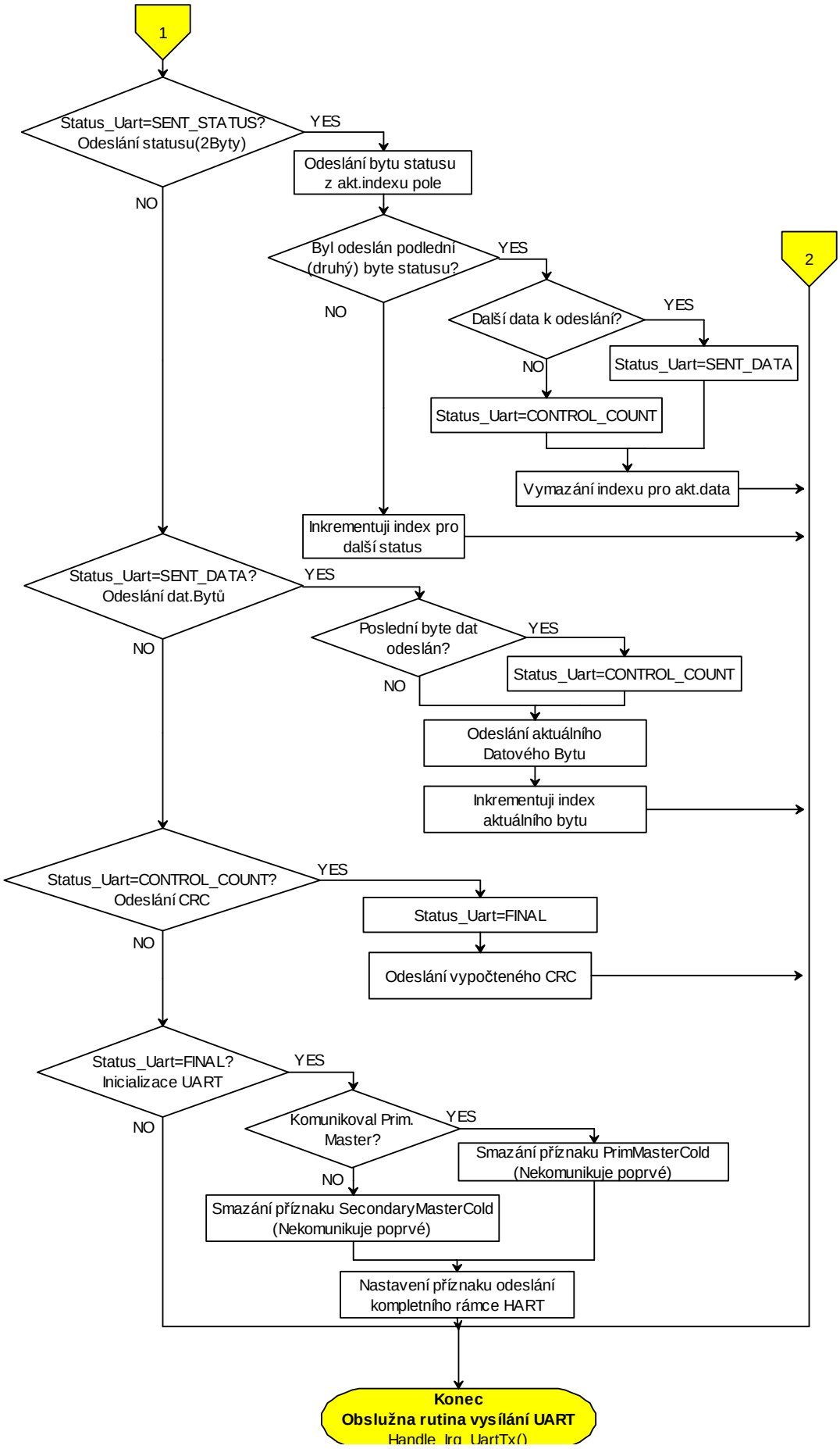
Obrázek 39: Diagram pro HART příkaz 1



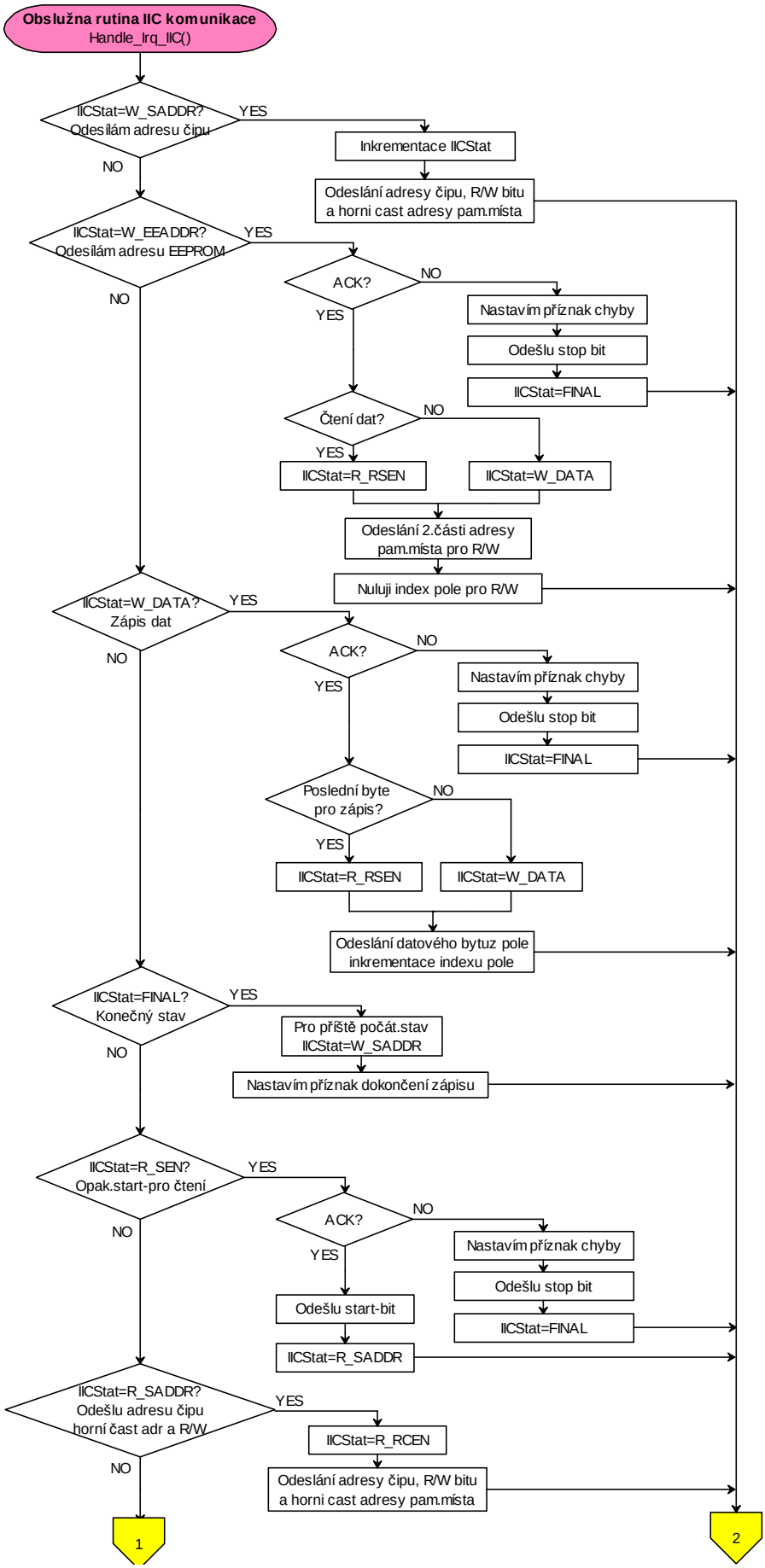
Obrázek 40: Diagram pro HART příkaz 119



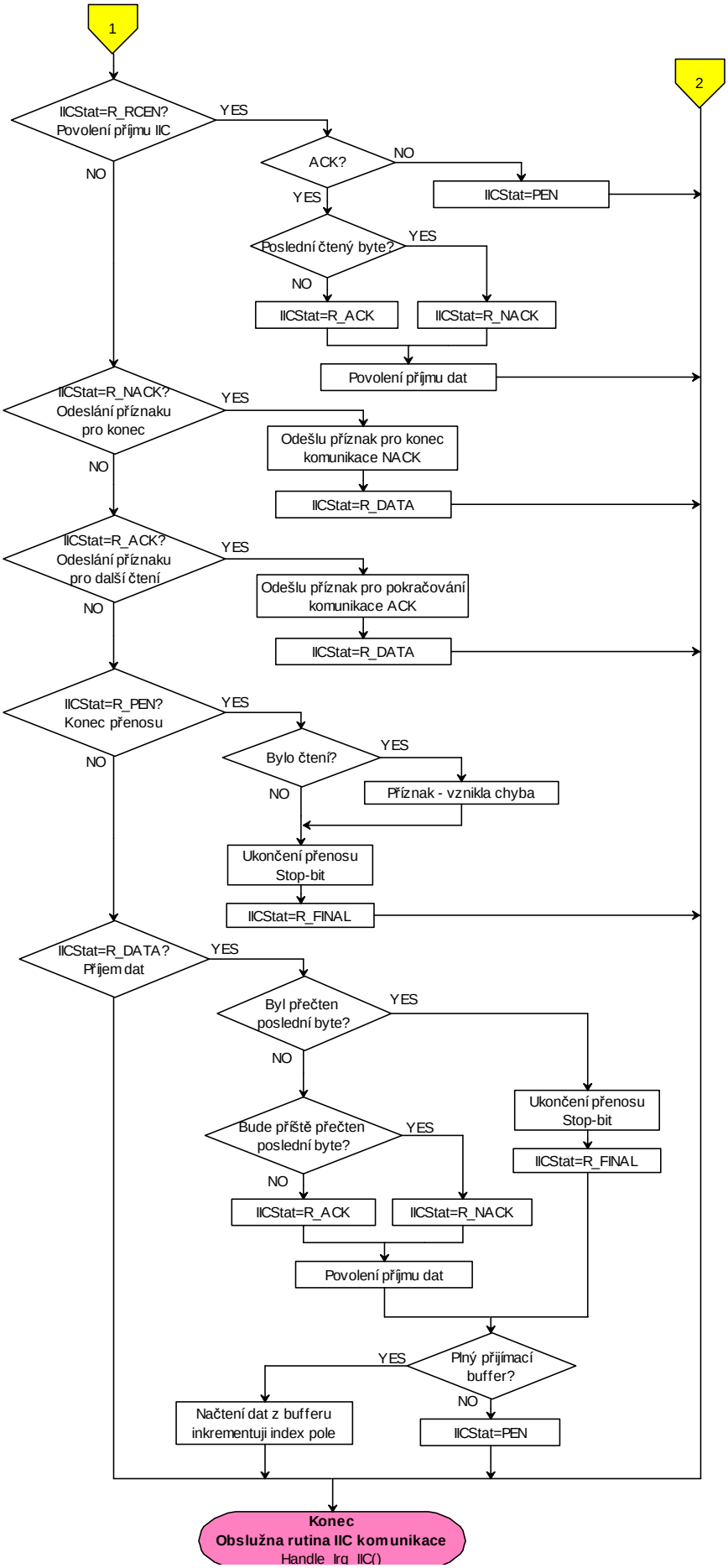
Obrázek 41 První část stavového automatu pro odesílání dat UART



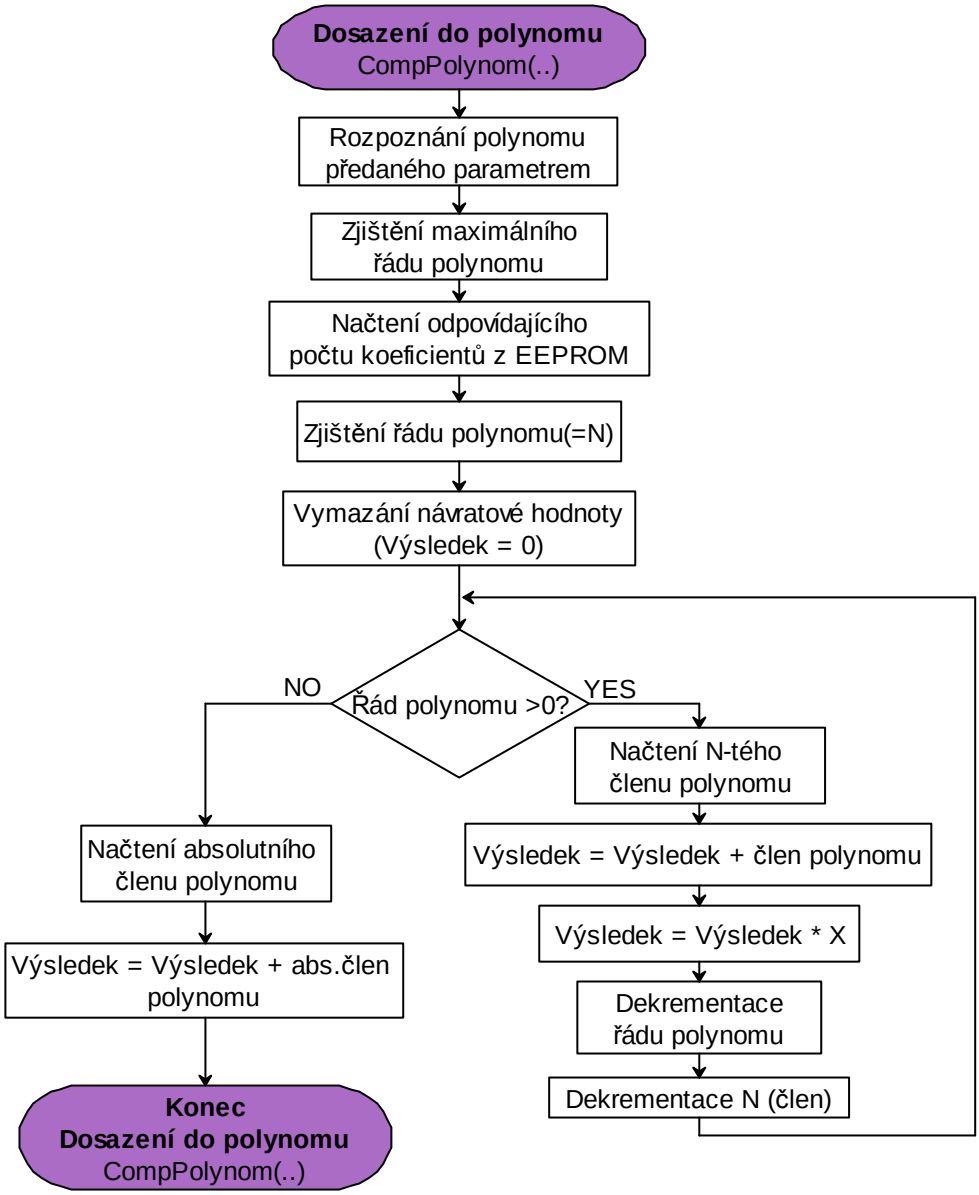
Obrázek 42: Druhá část stavového automatu pro odesílání dat UART



Obrázek 43: První část stavového automatu pro odesílání/příjem dat – IIC



Obrázek 44: Druhá část stavového automatu pro odesílání/příjem dat – IIC



Obrázek 45: Funkce CompPolynom() pro výpočet hodnoty polynomu