ZÁPADOČESKÁ UNIVERZITA V PLZNI
FAKULTA ELEKTROTECHNICKÁ

KATEDRA APLIKOVANÉ ELEKTRONIKY A TELEKOMUNIKACÍ

BAKALÁŘSKÁ PRÁCE

NÁVRH A ZHOTOVENÍ SOFTWARU NÍZKÉ VRSTVY PRO EMBEDDED ŘÍDÍCÍ SYSTÉM ZÁŽEHOVÉHO MOTORU

Martin Dosedla, DiS. 2018

Abstrakt

Tato práce dokumentuje návrh a realizaci nízkoúrovňového softwaru určeného pro řídící systém benzinového zážehového motoru. Zabývá se vývojem, testováním a kalibrací následujících algoritmů, funkcí a procedur:

- Rozpoznání polohy klikového a vačkového hřídele, výpočet rychlosti otáčení motoru.
- A/D převodů vstupních analogových signálů a jejich zabezpečení proti rušení.
- Ovladačů „Coil on plug“ zapalovacích modulů.
- Ovladačů vstřikovacích ventilů.
- Ovladačů krokového motoru řízení polohy škrtících klapek.
- Ovladačů komunikace s palubními přístroji a diagnostickými zařízeními.

Klíčová slova

Software nízké vrstvy, řídící systém benzinového motoru, řízení v reálném čase, STM32F4.

Abstract

This thesis documents the design and realisation of hardware control software for gasoline engine control system. It deals with the development, testing and calibration of the following algorithms, functions and procedures:

- Detection of the crankshaft and camshaft position and computation of the engine speed.
- A/D conversions of the analog input signals and suppression of interferences.
- „Coil on plug“ ignition modules control drivers.
- Injection valves control drivers.
- Throttle valves stepper motor control drivers.
- Drivers for serial communication with body instruments and diagnostic tools.

Keywords

Hadrware control software, gasoline engine control system, real – time control systems, coil on plug, STM32F4.

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně, s použitím odborné literatury a pramenů uvedených v seznamu, který je součástí této bakalářské práce.

Dále prohlašuji, že veškerý software, použitý při řešení této bakalářské práce, je legální.

Podpis:

V Plzni 7. 6. 2018 Martin Dosedla, DiS.

Poděkování

Děkuji vedoucímu práce, Ing. Petrovi Weissarovi, Ph.D. za cenné rady a připomínky, které pomohly k vyřešení mnohých problémů při tvorbě této práce.

Obsah

1 Seznam symbolů a zkratek....8
2 Úvod....9
2.1 Požadované vlastnosti navrhovaného systému řízení benzinového motoru....12
2.2 Požadavky na navrhovaný software nízké vrstvy....12
3 Výběr vhodného mikrokontroléru....14
3.1 Původní návrh....14
3.2 Nový návrh....15
4 Návrh modulu pro detekci pracovní polohy a fáze motoru....16
4.1 Požadované výkony softwarového modulu....16
4.2 Princip snímání polohy klikového a vačkového hřídele....16
4.3 Blokové schéma systémů pro detekci polohy a fáze válců motoru....17
4.4 Realizace detektoru pracovní polohy a fáze válců....19
4.5 Realizace výpočtu rychlosti otáčení klikového hřídele....24
4.6 Realizace řízení citlivosti vstupních zesilovačů....24
5 Návrh ovladačů zapalovacích modulů....25
5.1 Požadované výkony ovladačů zapalovacích modulů....25
5.2 Princip činnosti řízení zapalovacích modulů....25
5.3 Realizace ovladačů zapalovacích modulů....26
6 Návrh ovladačů vstřikovacích ventilů....34
6.1 Požadované výkony ovladačů vstřikovacích ventilů....34
6.2 Princip činnosti ovladačů vstřikovacích ventilů....34
6.3 Realizace ovladačů vstřikovacích ventilů....37
7 Návrh zpracování analogových vstupních signálů....37
7.1 Požadavky na zpracování analogových signálů....37
7.2 Realizace zpracování analogových signálů....38
8 Návrh regulátoru polohy sekundárních škrtících klapek....39
8.1 Požadavky na regulátor polohy sekundárních škrtících klapek....39
8.2 Realizace regulátoru polohy sekundárních škrtících klapek....39
9 Návrh modulu pro komunikaci s palubním přístrojem....42
9.1 Požadavky na modul pro komunikaci s palubním přístrojem....42
9.2 Realizace komunikace s palubním přístrojem....42

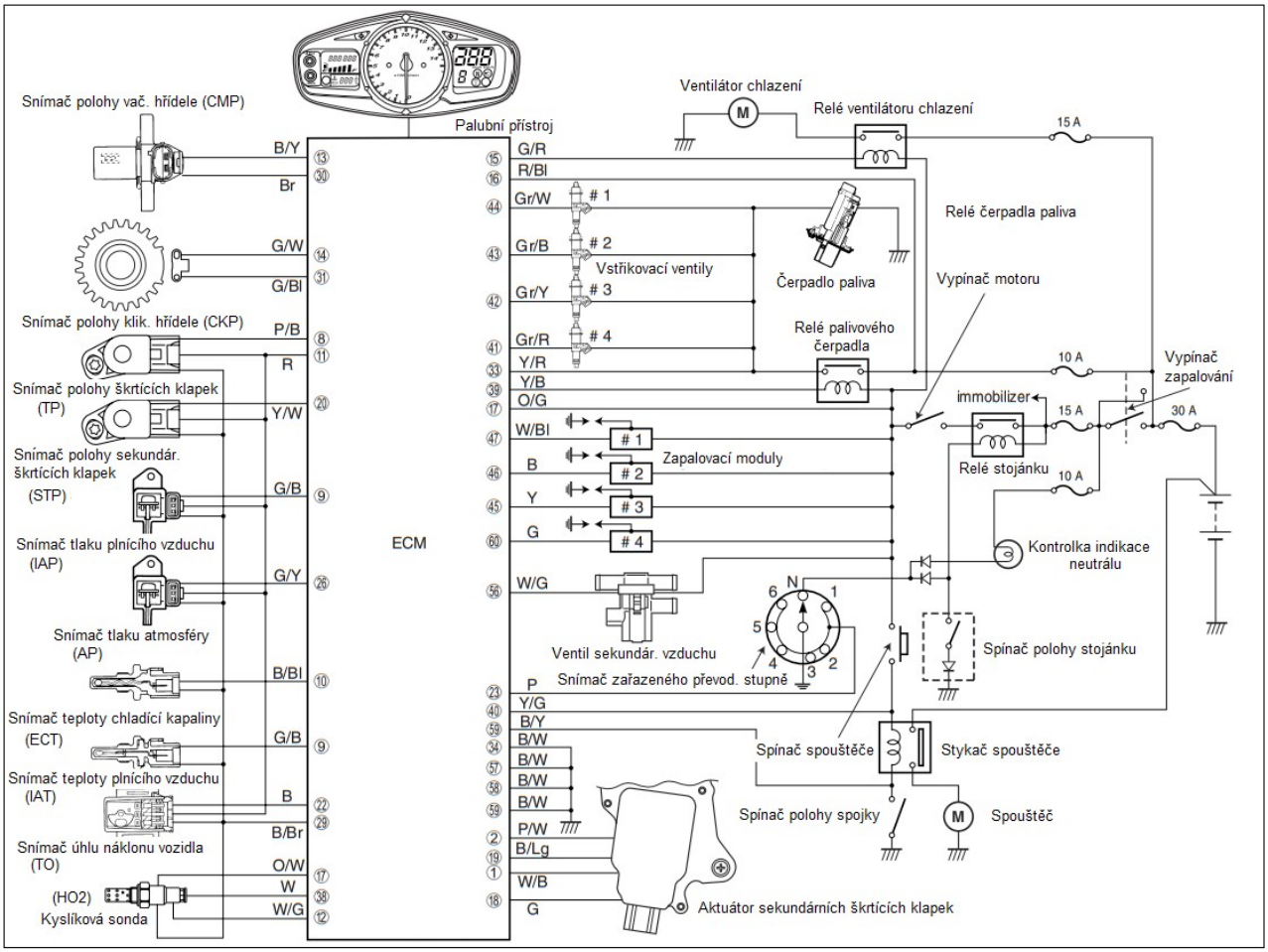
10 Návrh modulu pro komunikaci s diagnostickým přístrojem....44
10.1 Požadavky na modul komunikace s diagnostickým přístrojem....44
10.2 Realizace modulu komunikace s diagnostickým přístrojem....44
11 Testování navrženého řešení softwaru nízké vrstvy....45
11.1 Test detekce pracovní polohy a fáze motoru....45
11.2 Test ovladačů zapalovacích modulů....49
11.3 Test ovladačů vstřikovacích ventilů....51
11.4 Test modulu úpravy analogových signálů....52
11.5 Test regulátoru polohy sekundárních škrtících klapek....53
11.6 Test komunikace s palubním přístrojem....53
11.7 Test komunikace s diagnostickým přístrojem....54
11.8 Test nároků softwaru nízké vrstvy na výpočetní čas....54
12 Závěr....55
13 Seznam použité literatury....55

1 Seznam symbolů a zkratek

CKP	„Crankshaft position“ - poloha klikového hřídele
CMP	„Camshaft position“ - poloha vačkového hřídele
DMA	„Direct memory access“ - přímý přenos dat mezi periferiemi a pamětí MCU
DSP	„Digital signal processing“ - Číslicové zpracování signálu
ECM	„Engine Control Module“ - řídící jednotka motoru
ECVI	„Early closed valve injection“ - Strategie řízení vstřikování paliva, viz. Kap. 6.2.
HWD	„Hardware driver“ - Zkratka použitá v souborech a signálech softwaru nízké vrstvy
ICSP	„In circuit serial programming“ - sériové programování mikrokontroléru již zapájeného v cílové aplikaci
MCU	„Microcontroller unit“ - Mikrokontrolér
OC/IC	„Output Capture/Input Compare“ - hardwarové periferie mikrokontroléru pro měření/generování diskrétních signálů
VR	„Variable reluctance“ sensor - reluktanční snímač

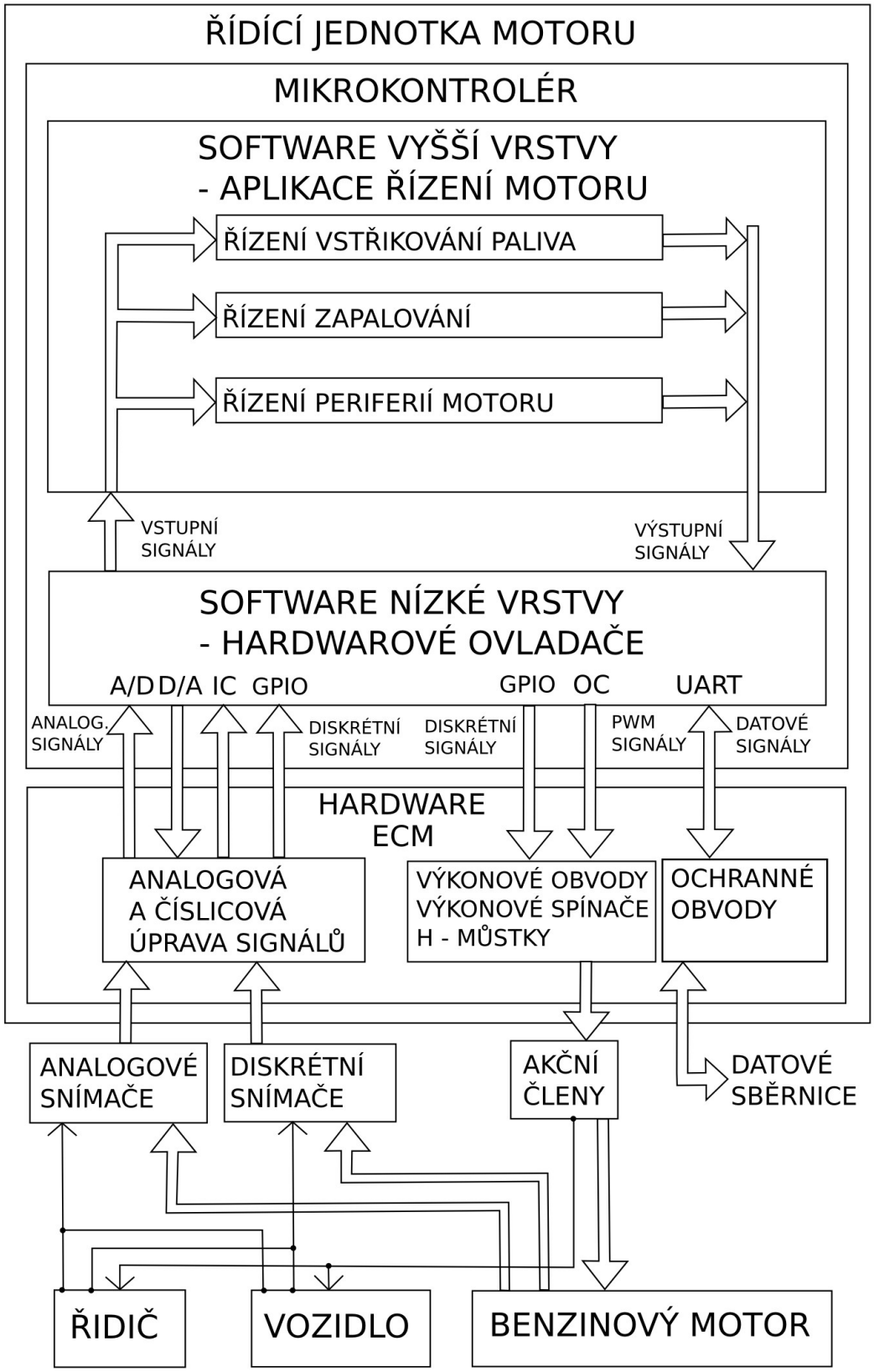
2 Úvod

Tento dokument popisuje vývoj, realizaci, testování a kalibraci nízkoúrovňového softwaru pro projekt řídícího systému benzinového spalovacího motoru. Řízení moderního, vysokootáčkového benzinového motoru je velmi náročné na přesnost v reálném čase. Příkladem je detekování polohy motoru a řízení zapalování, které musí udržet přesnost 0,5° otočení klikové hřídele při rychlosti otáčení 15000 1/min. Zvláštní důraz v návrhu je kladen na maximální využití hardwarových periferií čipu mikrokontroléru tak, aby bylo výrazně ušetřeno zatížení samotného procesoru a výpočetní čas, a byla maximalizována přesnost řízení v reálném čase. Práce se zabývá rovněž výběrem vhodného mikrokontroléru, který nejlépe vyhoví požadavkům aplikace řídícího systému benzinového motoru. Tato práce je částí rozsáhlejšího projektu autora. Cílem tohoto projektu je vývoj, zhotovení, testování a kalibrování kompletního řídícího systému pro až čtyřválcový, zážehový spalovací motor. Projekt zahrnuje vývoj hardwarové části systému – kompletní řídící jednotky. Tato obsahuje vstupní analogové a číslicové obvody zpracování signálu, filtry a tvarovače, mikrokontrolér, výstupní obvody zpracování analogového a číslicového signálu a obvody výkonové elektroniky (výkonové spínače a H – můstky). Součástí projektu je i vývoj kompletního embedded softwaru pro mikrokontrolér. Integrace tohoto projektu do motorového vozidla je naznačena na obr. 1. Struktura celého projektu je znázorněna blokovým schématem na obr. 2.



Obr. 1: Integrace řídícího systému motoru do elektrického vybavení vozidla. Zdroj: [2]

Nízkoúrovňovým softwarem rozumíme část softwaru, která tvoří rozhraní mezi softwarem vyšší vrstvy (výkonnou aplikací řízení motoru) a hardwarem řídící jednotky motoru, senzory a akčními členy. Software vyšší vrstvy je zhotoven takovým způsobem, kdy očekává vstupní signály jako čísla ve fyzikálních jednotkách měřených veličin. Výstupní signály opět poskytuje jako čísla ve fyzikálních jednotkách požadovaných veličin. Úkolem softwaru nízké vrstvy je zajistit měření vstupních signálů (A/D převody, měření diskrétních signálů, sběr dat ze sítí) a poskytnout je vyšší vrstvě, již připravené ve fyzikálních jednotkách. Na výstupní straně software nízké vrstvy přebírá od vyšší vrstvy požadované signály pro řízení akčních členů opět ve fyzikálních jednotkách požadovaných veličin. Úkolem softwaru nízké vrstvy je tyto veličiny převést na signály vhodné pro řízení akčních členů a požadované akční zásahy za pomoci hardwaru vykonat.



Obr. 2: Blokové schéma systému řízení benzinového motoru.

2.1 Požadované vlastnosti navrhovaného systému řízení benzinového motoru

Druh vstřikování paliva:	Nepřímé, vícebodové (MPI), sekvenční
Počet válců:	≤ 4
Počet vstřikovacích ventilů:	≤ 4
Počet zapalovacích jednotek:	≤ 4
Použitelný rozsah otáček motoru:	50 – 15000 1/min
Rozsah regulace úhlu zapalování:	0 – 120°

2.2 Požadavky na navrhovaný software nízké vrstvy

- Detekovat polohu a pracovní fázi klikového hřídele a každého válce. Měřit aktuální rychlost otáčení klikového hřídele.
- Zajistit periodický A/D převod analogových vstupních signálů. Pro přenos dat použít DMA. Pro odstranění impulzního rušení, indukovaného v kabelových vedeních, společných s výkonovými vodiči, použít mediánové filtry.
- Zajistit A/D převod vybraných analogových signálů v čase synchronním s polohou klikového hřídele.
- Zajistit čtení a předání stavů diskrétních vstupních signálů do softwaru vyšší vrstvy.
- Implementovat ovladače vstřikovacích ventilů. Ovladače musí synchronizovat činnost vstřikovacích ventilů s pracovní polohou každého válce podle strategie vstřikování ECVI. Požadovanou strategii vstřikování a dobu otevření vstřikovacích ventilů určuje software vyšší vrstvy.
- Implementovat ovladače zapalovacích modulů. Ovládání zapalovacích modulů musí být synchronizováno s pracovní polohou každého válce, s nejvyšší odchylkou do 0,5° otočení klikového hřídele. Požadovaný úhel zapalování každého válce a dobu nabíjení každého zapalovacího modulu určuje software vyšší vrstvy.
- Implementovat ovladače výkonových spínačů periferií motoru (ventilátorů chlazení, ventilů sekundárního vzduchu, palivových čerpadel, vyhřívání kyslíkových sond).
- Implementovat PWM výstup pro ovládání otáčkoměru.
- Implementovat ovladače pohonu krokového motoru sekundárních škrtících klapek

a zpětnovazební regulátor jejich polohy. Požadovanou polohu sekundárních škrtících klapek určuje software vyšší vrstvy.

- Implementovat datovou sběrnici UART/RS232 pro komunikaci s palubní deskou vozidla. Pro přenos dat použít DMA.
- Implementovat datovou sběrnici UART/RS232 pro komunikaci s diagnostickým zařízením. Pro přenos dat použít DMA.
- Implementovat několik (dle možností hardwaru) záložních analogových i diskrétních vstupů a výstupů pro případné další použití.
- Všechny spojité signály na rozhraní se softwarem vyšší vrstvy poskytovat/požadovat ve fyzikálních jednotkách.
- Při návrhu softwaru nízké vrstvy maximálně využít dostupné hardwarové periferie na čipu mikrokontroléru – minimalizovat tak zatížení procesoru a výpočetní čas. Zejména ovládání vstřikovacích ventilů a zapalovacích modulů implementovat s pomocí OC kanálů.

3 Výběr vhodného mikrokontroléru

3.1 Původní návrh

Původní návrh a počátek tohoto projektu byl realizován na mikrokontroléru od firmy Microchip, s nímž měl autor již zkušenosti. Zejména rodina dsPIC33FJ/EP se velmi osvědčila pro aplikace náročné na reálný čas a číslicové zpracování signálu. Nabízí:

- Rychlé 16-bitové jádro.
- Efektivní sadu strojových instrukcí, včetně bitových operací.
- Velmi rychlou odezvu přerušení.
- DSP jádro.
- Dostatečné vybavení hardwarovými periferiemi, zejména časovači s OC/IC moduly.
- Rychlé a pružně konfigurovatelné A/D převodníky.
- Kvalitní vývojové prostředí přímo od výrobce, zdarma.
- Překladač jazyka C přímo od výrobce, v základní a dostačující verzi zdarma.
- ICSP programátory a debuggery od výrobce.
- Zdarma softwarová podpora výrobce – hotové knihovny. Zejména pro efektivní číslicové zpracování dat, napsané v assembleru a využívající DSP jádro.
- Plná podpora počítačových platforem založených na Linuxu.

Při snaze použít tuto platformu pro projekt řídícího systému benzinového motoru se však objevily podstatné nedostatky. Software vyšší vrstvy, zahrnující model chování motoru a výkonné řídící a regulační systémy, bude obtížné provozovat na 16-bitovém mikrokontroléru. Hlubší analýza systémových požadavků ukázala, že se projekt bez 32-bitové platformy prakticky neobejde. Použití čipu z rodiny dsPIC33 tedy nebylo možné.

Následně bylo zvažováno použití 32-bitového mikrokontroléru firmy Microchip, z rodiny PIC32MX/MZ. Tyto mikrokontroléry jsou však primárně stavěny pro uživatelské grafické aplikace, a mají nedostatečné vybavení hardwarovými periferiemi. Chybí zejména dostatečný počet časovačů vybavených OC/IC kanály. Přesnost, rozlišení a rychlost A/D převodníků není vyhovující. Bylo nutné se tedy ohlédnout po nabídce jiných výrobců.

Před zvážením nabídek jiných mikrokontrolérů byly shrnuty hlavní systémové požadavky:

- 32-bitové jádro o frekvenci alespoň 100 MHz.
- Paměť programu Flash o kapacitě alespoň 256 kB a životnosti alespoň 10 000 zápisů.
- Automatické, hardwarové ukládání kontextu při vstupu do přerušení, vzhledem k požadavku na rychlou odezvu přerušení.
- 12-bitové A/D převodníky, konfigurovatelné pro automatické sekvenční skenování kanálů a přenos dat pomocí DMA.
- Alespoň 10 nezávislých hardwarových časovačů s kanály OC/IC/PWM, z toho alespoň dva 32-bitové.
- Dostupnost v pouzdrech LQFP/TQFP s počtem pinů okolo 100.
- Alespoň 2 D/A převodníky.
- Vývojové prostředí a překladač jazyka C dostupný zdarma, pokud možno podporovaný pro počítačové platformy založené na Linuxu.
- Srozumitelná dokumentace pro pokud možno bezproblémovou práci s mikrokontrolérem.

3.2 Nový návrh

Dobře použitelný se ukázal mikrokontrolér rodiny STM32F4 od výrobce ST Microelectronics. Jeho parametry a periferní vybavení jsou perfektní pro aplikaci v daném projektu. Avšak drobné nevýhody byly shledány:

- Mikrokontrolér je poměrně drahý. Vzhledem k vysokým systémovým požadavkům na rychlost a přesnost řízení však cena v tomto projektu není klíčovým kritériem.
- V porovnání s firmou Microchip výrobce neposkytuje natolik kvalitní softwarovou podporu. Chybí některé knihovny pro digitální zpracování dat (např. PID regulace), ovladače pro rozhraní SD karet (SDIO)...

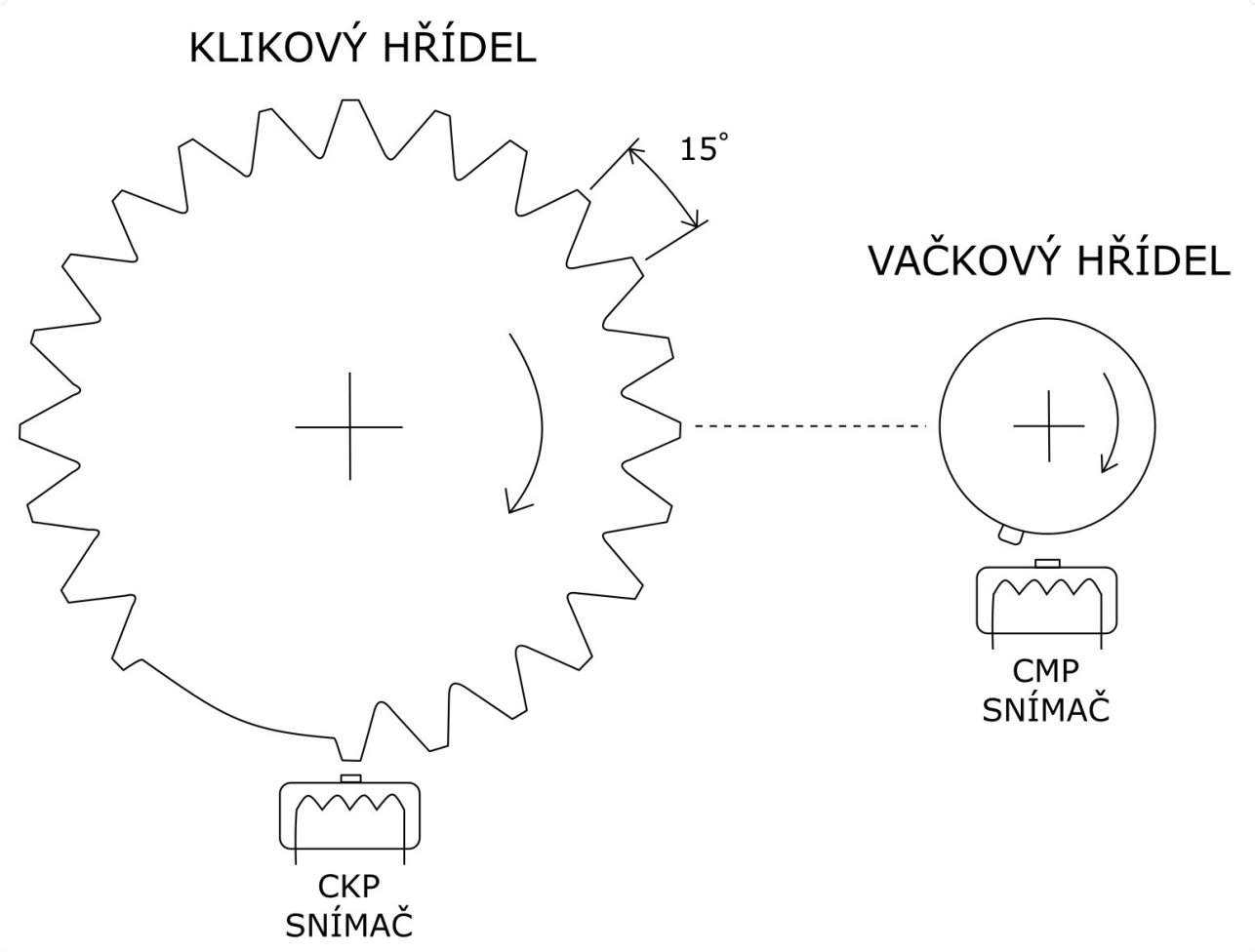
Výše uvedené nedostatky však nejsou zásadní a projekt tedy započal s použitím mikrokontroléru rodiny STM32F4.

4 Návrh modulu pro detekci pracovní polohy a fáze motoru

4.1 Požadované výkony softwarového modulu

S použitím signálů snímačů poloh klikového a vačkového hřídele detekovat pracovní polohu a pracovní fázi motoru pro další použití ovladači vstřikovacích ventilů a zapalovacích modulů. Vytvořit synchronizaci softwaru s aktuální polohou motoru. Vypočítat aktuální rychlost otáčení klikového hřídele. Ověřovat věrohodnost signálů snímačů poloh a výstupních signálů. Za použití D/A převodníků řídit citlivost zesilovačů vstupních signálů tak, aby signál snímačů polohy klikového hřídele a vačkového hřídele nebyl rušen ani ztrácen. Systém musí být schopen činnosti po výpadku signálu snímače polohy vačkového hřídele za provozu.

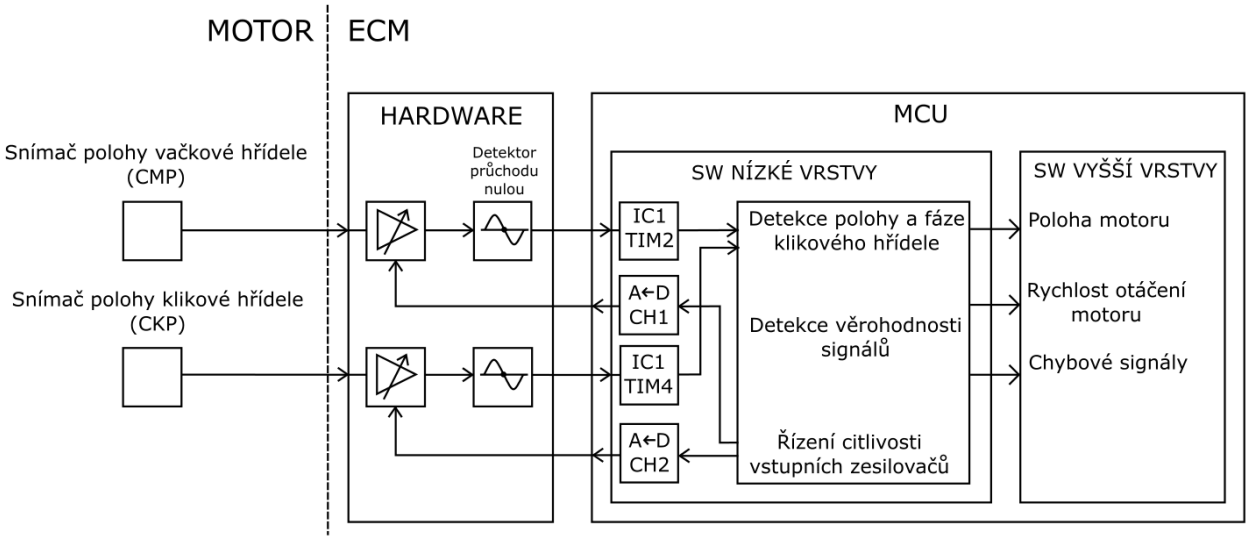
4.2 Princip snímání polohy klikového a vačkového hřídele



Obr. 3: Mechanické uspořádání impulzních kol a snímačů CKP a CMP.

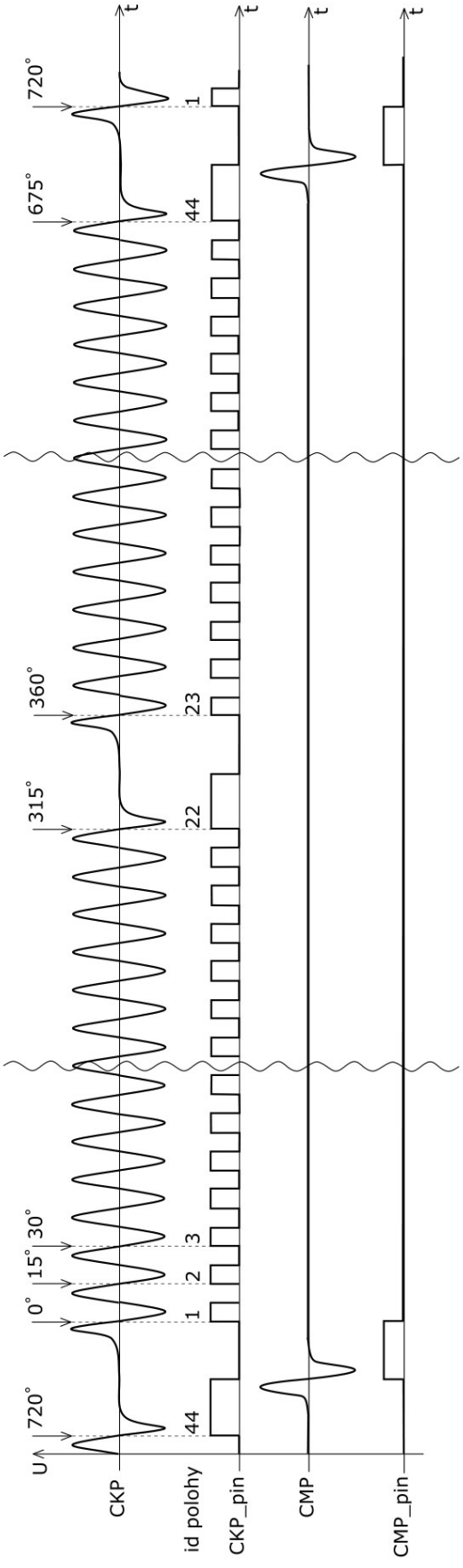
K obr. 3: Na klikovém hřídeli je namontováno impulzní kolo, vyrobené z nemagnetované, magneticky tvrdé oceli. Na svém obvodu má obrobeno 22 zubů s odstupem 15°. Dva zuby chybí – díky této mezeře může detekční systém rozeznat polohu hřídele. V blízkosti zubů je umístěn indukční snímač typu VR. Otáčí – li se klikový hřídel, VR snímač generuje sinusové napětí. Každý průchod tohoto napětí nulou (směrem od kladné hodnoty do záporné) značí okamžik průchodu zubu impulzního kola osou jádra snímače.

4.3 Blokové schéma systémů pro detekci polohy a fáze válců motoru



Obr. 4: Blokové schéma uspořádání pro detekci polohy motoru.

Sinusový signál, generovaný VR snímači, je v hardwaru ECM zesílen zesilovačem s proměnným napěťovým ziskem. Poté je detektorem průchodu nulou generován pravoúhlý signál (*CKP_pin* a *CMP_pin* na obr. 5), kdy náběžná hrana tohoto signálu odpovídá průchodu nulou signálu snímače. Amplituda sinusového napětí, generovaného snímačem, je závislá na rychlosti otáčení impulzního kola. Pohybuje se od desítek mV při nejnižších rychlostech otáčení, po asi 80 V při nejvyšší rychlosti otáčení. Z tohoto důvodu je nutné regulovat zisk vstupního zesilovače (viz. kapitola 4.6), aby bylo možné v celém rozsahu rychlostí otáčení bezpečně oddělit signál snímače od rušivých signálů a korektně detekovat průchod nulou. Regulace zisku zesilovačů je možná pomocí napěťového vstupu. Napětí na tyto vstupy je přiváděno z D/A převodníků mikrokontroléru a jeho velikost je řízena v softwaru nízké vrstvy podle rychlosti otáčení snímaného hřídele.

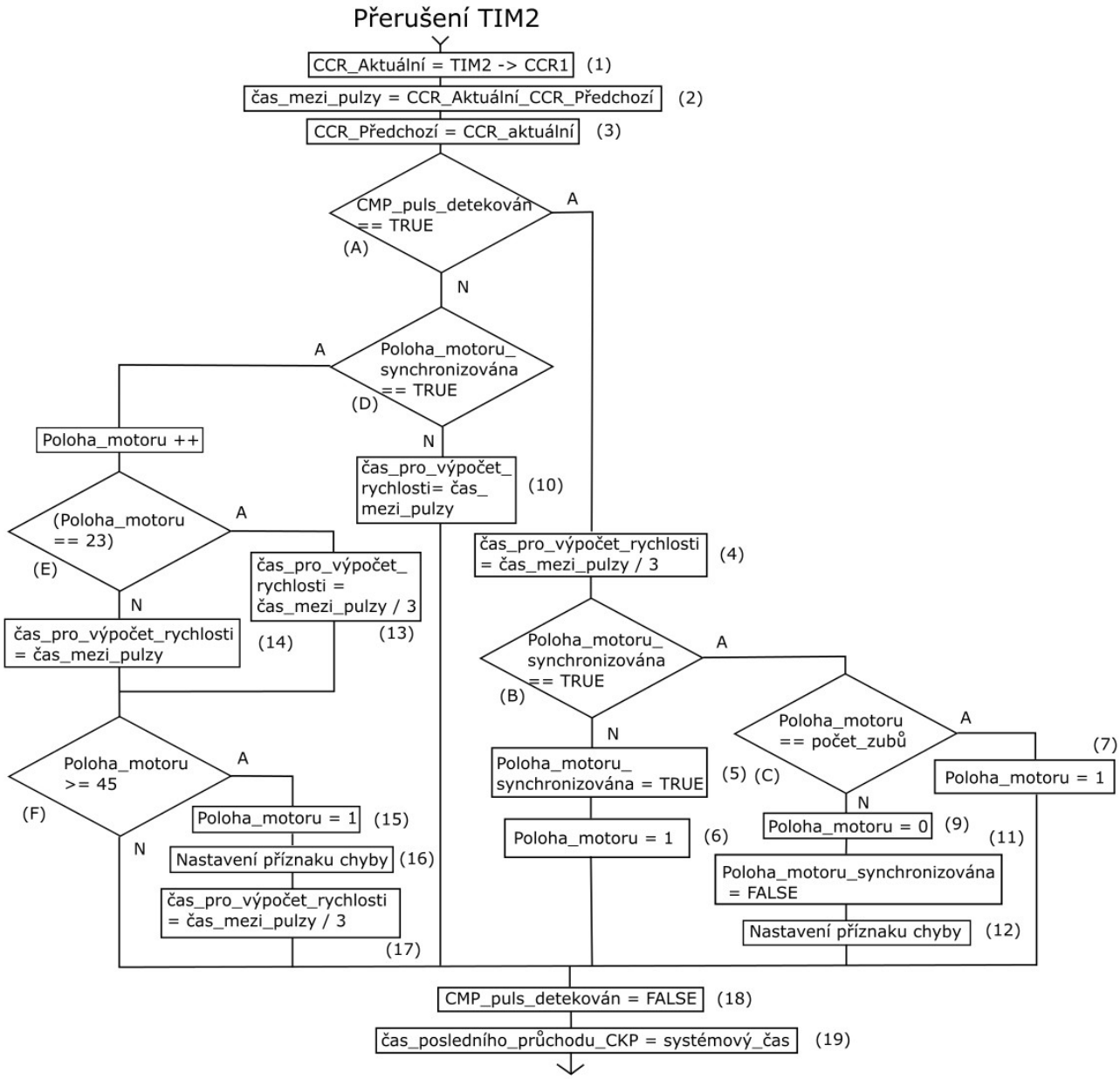


Obr. 5: Průběhy signálů snímačů CKP a CMP během chodu motoru. Údaj „id_polohy" je výstupem detektoru polohy motoru.

4.4 Realizace detektoru pracovní polohy a fáze válců

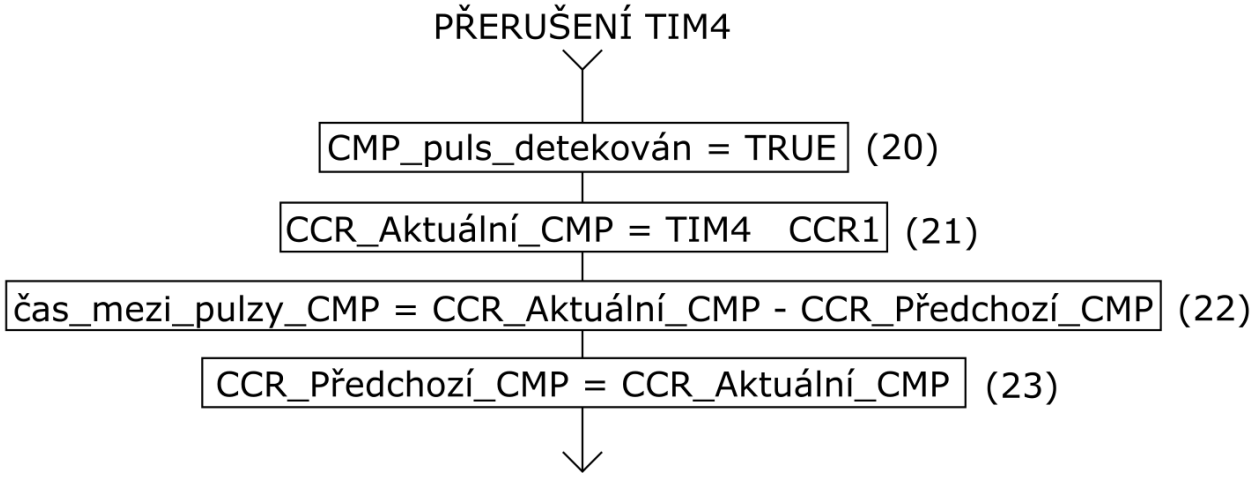
Pro realizaci detektoru pracovní polohy a fáze válců motoru lze s výhodou využít hardwarových časovačů mikrokontroléru s kanály IC. Pravoúhlý signál snímače CKP je přiveden z detektoru průchodu nulou na vstup kanálu IC1 časovače TIM2. Z důvodu velkého rozsahu měřených časů a požadavku na vysokou přesnost měření je použit 32-bitový časovač, který poskytne dostatečné rozlišení měření v celém měřícím rozsahu. Kanál IC1 časovače TIM2 je nastaven tak, aby při náběžné hraně vstupního signálu vyvolal přerušení, ve kterém jsou následně vykonány algoritmy detekce polohy motoru a další funkce, které je nezbytné synchronizovat s polohou klikového hřídele. Při zjištění náběžné hrany vstupního signálu IC hardware automaticky uloží hodnotu čítače TIM2 do registru CCR1. Zdroj hodinového signálu pro časovač TIM2 je nastaven z hodinové sběrnice mikrokontroléru APB1 o frekvenci 84 MHz, bez použití děliče. Pro maximalizaci rozlišení čítače zde můžeme použít nejvyšší dostupnou frekvenci čítání. 32-bitový čítač při této frekvenci přeteče za přibližně 51 sekund. Nejnižší rychlost otáčení klikového hřídele, kterou systém musí ještě zpracovat, je 50 1/min, což odpovídá nejdelší periodě mezi impulsy snímače CKP o délce 150 ms. Nechtěné přetečení tedy není možné. Pravoúhlý signál snímače CMP je z detektoru průchodu nulou přiveden vstup kanálu IC1 časovače TIM4. Měření délky periody signálu snímače CMP není náročné na přesnost, slouží pouze jako kontrolní signál pro diagnostiku věrohodnosti údaje o rychlosti otáčení klikové hřídele. Proto je možné použít 16-bitový čítač. Jeho hodinový signál je opět získán ze sběrnice APB1, o frekvenci 84 MHz. Zde je však nutné frekvenci čítání snížit, aby nedošlo při nízkých rychlostech otáčení vačkového hřídele k jeho přetečení. Nejnižší rychlost otáčení vačkového hřídele, kterou systém musí ještě zpracovat, je 25 1/min, což odpovídá nejdelší periodě mezi impulsy snímače CKP o velikosti 2,4 s. Chceme – li, aby čítač přestal přetékat právě při této rychlosti otáčení klikové hřídele, musíme hodinový kmitočet časovače nastavit na frekvenci přibližně 27,276 kHz. Nejbližší odpovídající hodnota děliče hodinových pulsů je potom 3076. Všechna nastavení periferií mikrokontroléru jsou provedena ve zdrojovém souboru *HWD_Init.c*.

Časovač TIM2 čítá nepřetržitě frekvencí 84 MHz. Jakmile osou snímače CKP projde zub impulzního kola, náběžná hrana signálu na vstupu kanálu IC1 vyvolá uložení hodnoty časovače do registru CCR1 a zároveň aktivuje požadavek na přerušení. Priorita přerušení od časovače TIM2 je nastavena na nejvyšší úroveň ze všech přerušení v systému – toto přerušení tedy bude obslouženo okamžitě. V tomto přerušení je vykonána procedura, naznačená na vývojovém diagramu obr. 6:



Obr. 6: Vývojový diagram detekce polohy a fáze motoru. Zdrojový soubor: HWD_CKP_Isr.c

Časovač TIM4 čítá rovněž nepřetržitě, frekvencí 27,276 kHz. Jakmile osou snímače CMP projde zub vačkového hřídele, náběžná hrana signálu na vstupu kanálu IC1 vyvolá uložení hodnoty časovače do registru CCR1 a zároveň aktivuje požadavek na přerušení. Priorita přerušení od časovače TIM4 je nastavena na nejnižší úroveň ze všech přerušení v systému – mezi průchodem zubu vačkového hřídele a průchodem dalšího zubu klikového hřídele, po kterém dochází k vyhodnocení polohy vačkového hřídele, je vždy dostatečná doba pro vykonání přerušení. V tomto přerušení je vykonána procedura, naznačená na vývojovém diagramu obr. 7:



Obr. 7: Vývojový diagram procedury přerušení TIM4. Zdrojový soubor: HWD_CMP_Isr.c

Vycházejme ze situace, kdy motor stojí a jeho poloha je neznámá. Začne – li se motor otáčet (působením spouštěče), je při průchodu každého zubu impulzního kola klikové hřídele osou snímače CKP, spuštěna procedura z obr. 6. V této proceduře je vždy uložena aktuální hodnota registru CCR1 a následně od ní odečtena hodnota předchozí (příkaz (1) a (2)). Tím je získána hodnota času mezi průchody posledních dvou zubů impulzního kola, tedy doba, kterou trvá otočení klikového hřídele o 15° (nebo 45°, pokud právě prošla okolo snímače zubová mezera impulzního kola). V tuto chvíli však ještě nelze určit pracovní polohu motoru – neznáme polohu vačkového hřídele. Lze však s jistotou říci, že motor se otáčí, lze rovněž i velmi nepřesně zjistit rychlost otáčení motoru. Tento signál je však důležitý pro software vyšší vrstvy, který na jeho základě např. aktivuje palivové čerpadlo. Dokud není známa poloha vačkového hřídele, podmínka (A) je nesplněna, poloha motoru ještě není synchronizována (podmínka (D) rovněž není splněna), dochází pouze k přiřazení změřené doby mezi průchody posledních dvou zubů (příkaz (10)), pro další

zpracování ve výpočtu rychlosti otáčení motoru. Projde – li osou snímače CMP zub vačkového hřídele, dojde ke spuštění procedury přerušení TIM4 (obr. 7). V této proceduře je nastaven příznak průchodu zubu vačkového hřídele (příkaz (20)) a je vypočítána doba mezi posledními dvěma průchody (byl – li nějaký předchozí). Následující procedura přerušení TIM2, podle výše zmíněného příznaku zjistí, že byl právě zaznamenán průchod zubu vačkového hřídele (podmínka (A) je splněna). Zub vačkového hřídele je mechanickým uspořádáním motoru umístěn takovým způsobem, že prochází osou snímače CMP vždy, když osou snímače CKP prochází zubová mezera impulzního kola klikového hřídele. Proto v je v tomto okamžiku měřená doba mezi pulzy CKP dělena třemi (příkaz (4)), aby z této doby mohla být následně vypočítána rychlost otáčení motoru. Poloha motoru ještě není synchronizována, podmínka (B) není splněna. Nyní je však poloha motoru již známa, je tedy nastaven příznak synchronizace (příkaz (5)). Z mechanického uspořádání motoru je nyní jasné, v jaké poloze se motor nachází – označme ji číslem 1 (příkaz (6)). Po proběhnutí procedury detekce polohy motoru, je vždy nulován příznak průchodu zubu vačkového hřídele (Příkaz (17)), aby bylo možné v následujícím cyklu opět rozpoznat případný průchod zubu.

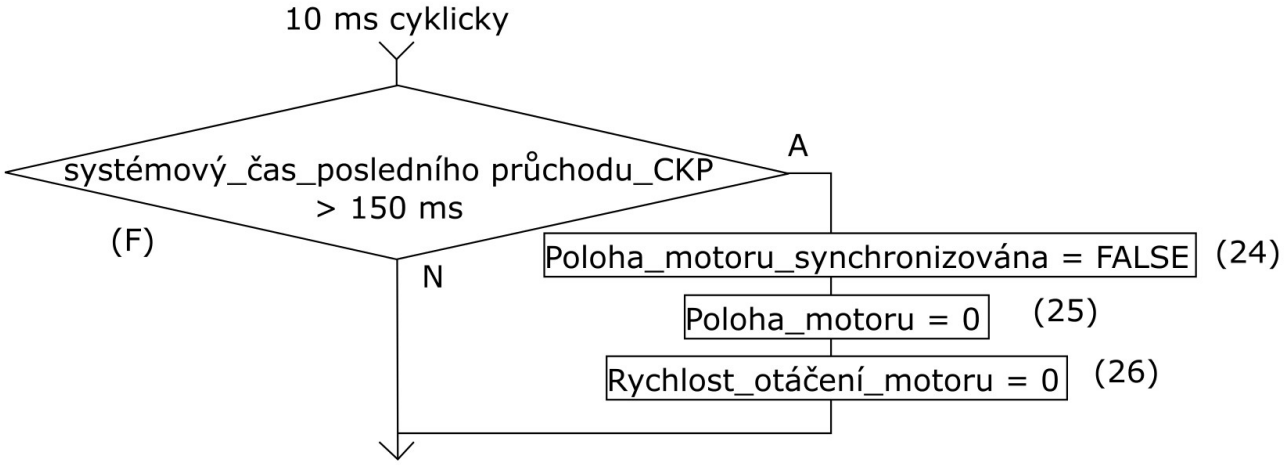
Po průchodu dalšího zubu osou snímače CKP, není rozpoznán příznak průchodu zubu vačkového hřídele (podmínka (A) nesplněna), a poloha motoru je již známa a synchronizována s detektorem polohy motoru (podmínka (D) splněna). Dochází k inkrementaci signálu o poloze motoru (příkaz (8)). Není – li poloha motoru rovna číslu 23 (podmínka (E) nesplněna), je úhlová vzdálenost posledních dvou zubů impulzního kola klikového hřídele 15°. Čas mezi posledními dvěma zuby může být použit jako čas pro výpočet rychlosti otáčení klikového hřídele (příkaz (14)). Je – li poloha motoru rovna číslu 23, je z mechanického uspořádání impulzního kola klikového hřídele jasné, že mezi posledními dvěma zuby se nachází zubová mezera. Podmínka (E) je splněna a naměřený čas je dělen třemi pro správný výpočet rychlosti otáčení klikového hřídele.

Děj, popsaný v předchozím odstavci, se opakuje do dalšího průchodu zubu vačkového hřídele. Po tomto bude podmínka (A) splněna. Poloha motoru je již známa a synchronizována, podmínka (B) je také splněna. Dle mechanického uspořádání motoru nyní musí být poloha motoru rovna číslu 44 (celkovému počtu zubů impulzního kola klikového hřídele). Není – li tomu tak, muselo dojít k rušení signálů snímačů nebo jejich ztrátě. Toto je detekováno podmínkou (C). Je – li v tomto okamžiku poloha motoru rovna číslu 44, můžeme usoudit, že signály jsou věrohodné. Poloha motoru je tedy nastavena na číslo 1 a celý cyklus chodu motoru se opakuje. Nebude – li podmínka (C) splněna, signály snímačů byly rušeny a poloha motoru nelze považovat za věrohodnou. V tomto případě jsou signály o poloze motoru a synchronizaci s detektorem polohy vynulovány. Tím je

vynuceno nové zasynchronizování polohy motoru, ale také nejméně jeden pracovní cyklus dlouhý výpadek řízení motoru. Proto je v tomto případě nastaven chybový příznak pro další diagnostické zpracování.

Vypadne – li během chodu motoru signál snímače CMP, bude podmínka (A) neustále nesplněna. Toto způsobí inkrementaci polohy motoru na číslo vyšší, než 44. Dosáhne – li signál polohy motoru čísla 45, je jasné, že signál snímače CKP nebyl zachycen. Tento stav je zachycen podmínkou (F). Je – li splněna, dojde k nastavení signálu polohy motoru na číslo 1 a k dělení naměřeného času pro výpočet rychlosti otáčení motoru třemi. Celý pracovní cyklus motoru se tak může opakovat a plná funkčnost systému zůstává zachována. Dojde – li k rozpoznání neplatných impulzů snímače CKP (např. vlivem rušení), dojde také ke splnění podmínky (F) a pokračování chodu systému s neplatným údajem o poloze motoru. Tento stav je však rozpoznán ihned po dalším impulzu signálu CMP podmínkou (C), po níž dojde k opětovnému synchronizování polohy motoru.

Dojde – li k zastavení chodu motoru nebo ke ztrátě signálu snímače CKP, nedojde k žádnému vykonání procedury přerušení TIM2. Nemůže tedy dojít k vynulování signálu o poloze motoru a příznaku synchronizace polohy. Proto musí být cyklicky spouštěna procedura (obr. 8), která tato vynulování zajistí. Na konci procedury přerušení TIM2 je uložen čas, ve kterém procedura proběhla (příkaz (19)). V cyklicky spouštěné proceduře je zjišťováno, jak dlouho procedura přerušení TIM2 neběžela. Je – li tato doba větší, než 150 ms (čas odpovídající rychlosti otáčení motoru nižší, než 50 1/min), dochází k vynulování signálu o poloze motoru a příznaku synchronizace polohy.



Obr. 8: Vývojový diagram detekce polohy a fáze motoru. Zdrojový soubor: HWD_EngSync.c

4.5 Realizace výpočtu rychlosti otáčení klikového hřídele

Procedura detekce polohy motoru, běžící v přerušení TIM2 (obr. 6), poskytuje po každém svém průběhu dobu, kterou trvalo otočení klikového hřídele o 15°. Údaj o rychlosti otáčení klikového hřídele je potřebný pro ovladače zapalovacích modulů, které běží rovněž v přerušení TIM2. Proto je nutné, aby rychlost otáčení klikového hřídele byla vypočtena okamžitě. Procedura výpočtu rychlosti otáčení je tedy ve zdrojovém kódu umístěna ihned za procedurou detekce polohy motoru. Časový údaj, poskytovaný procedurou detekce polohy motoru, je kvantován v počtu inkrementů časovače TIM2, které proběhly za dobu otočení klikového hřídele o 15°. Tento počet je nutné přepočítat na údaj ve fyzikálních jednotkách (počet otáček klikového hřídele za minutu), podle vztahu

$$n = \frac{f\alpha}{6\,N} \quad [1/\text{min; Hz, °, -}]. \tag{1}$$

Kde n je počet otáček klikového hřídele, f frekvence čítání čítače TIM2, α úhel otočení klikového hřídele, za který bylo načítáno N inkrementací čítače TIM2. V případě, kdy f = 84 MHz a α = 15°, můžeme vztah zjednodušit:

$$n = \frac{21 \cdot 10^7}{N} \quad [1/\text{min; -}] \tag{2}$$

Vztah (2) je již možno implementovat v celočíselné aritmetice.

4.6 Realizace řízení citlivosti vstupních zesilovačů

Řízení citlivosti vstupních zesilovačů signálů snímačů CKP a CMP je realizováno pomocí dvou funkcí rychlosti otáčení motoru. Průběhy obou funkcí jsou dány dvěma jednorozměrnými tabulkami, výstupní hodnoty jsou určeny lineární interpolací a použity jako vstupní data pro dvojici D/A převodníků, které napěťovými výstupy řídí zisk obou vstupních zesilovačů. Převodní charakteristiky obou funkcí musí být kalibrovány během testů na konkrétním motoru. Jsou závislé na typu snímačů, intenzitě rušivých signálů a provozním rozsahu rychlostí otáčení motoru. Konfigurace D/A převodníků je provedena ve zdrojovém souboru *HWD_Init.c*, vlastní řízení zisku vstupních zesilovačů probíhá ve zdrojovém souboru *HWD_VRSensorsAmpCtrl.c*.

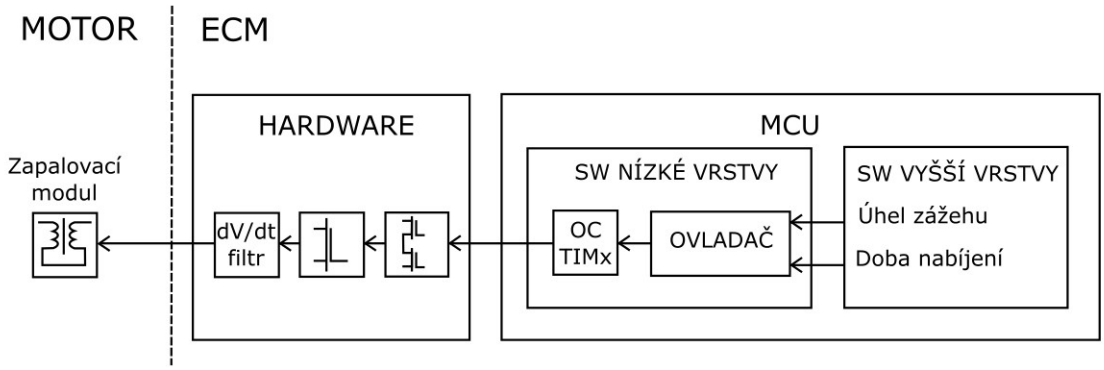
5 Návrh ovladačů zapalovacích modulů

5.1 Požadované výkony ovladačů zapalovacích modulů

Software vyšší vrstvy vypočítá a poskytne požadovaný úhel zážehu a požadovanou dobu nabíjení zapalovacího modulu pro každý válec. Úhel zážehu je poskytován jako úhel natočení klikového hřídele před dosažením horní úvrati pístu daného válce, v okamžiku, ve kterém má dojít k počátku zážehu. Úkolem ovladače zapalovacích modulů je tento zážeh vykonat s přesností ± 0,1°, při dodržení požadované doby nabíjení zapalovacího modulu.

5.2 Princip činnosti řízení zapalovacích modulů

Řídící systém je navržen pro nejmodernější typ zapalovacích modulů, tzv „Coil on plug“. Tyto moduly mají zapalovací transformátory umístěné přímo v hlavě válců motoru. Odpadá nutnost použití vysokonapěťových zapalovacích kabelů, čímž je podstatně zvýšena spolehlivost systému a snižují se nároky na montážní prostor. Zapalovací moduly „Coil on plug“ jsou schopny pracovat při vysokých frekvencích zapalování, mají nízké energetické ztráty a umožňují individuální řízení zapalování každého válce. Z důvodu nízké indukčnosti a nízkého sériového odporu však vyžadují přesné řízení doby nabíjení. Doba nabíjení je závislá na aktuálním napájecím napětí modulů a na požadované energii pro zapalovací jiskru.

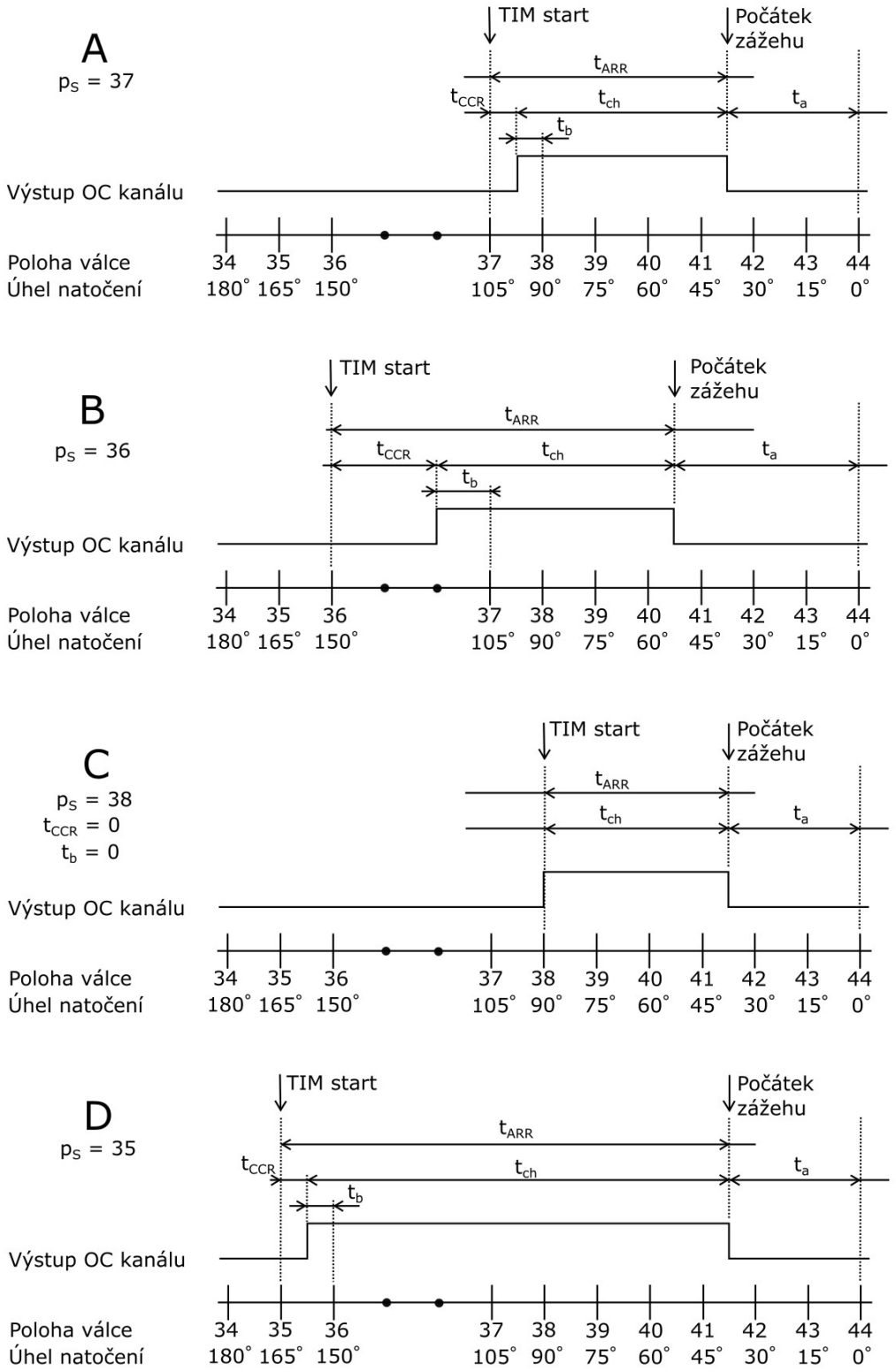


Obr. 9: Blokové schéma uspořádání pro řízení zapalovacích modulů.

Pro zjednodušení ovladačů zapalovacích modulů byly zavedeny signály polohy pístů každého válce, vypočítané z globální (hlavní) polohy motoru, poskytnuté detektorem polohy motoru. Pro čtyřválcový motor s pořadím zapalování 1-2-4-3 a odstupem zapalování 180° jsou signály o poloze každého válce přiřazeny podle Tab. 1. Polohy motoru v úhlech natočení klikového hřídele 330°, 345°, 690°, 705° chybí – nejsou známy, jelikož se nachází v mezeře mezi zuby impulzního kola klikové hřídele a není je možné detekovat. S tímto musí počítat i ovladače zapalovacích modulů a vstřikovacích ventilů.

5.3 Realizace ovladačů zapalovacích modulů

Za účelem generování impulzů pro ovládání zapalovacích modulů lze elegantně využít OC kanály hardwarových časovačů mikrokontroléru. Jsou nakonfigurovány na „One-pulse“ mód, viz. [1], kap. 19.3.10. V tomto módu časovač po svém spuštění generuje vždy jeden impulz o programovatelné délce, s programovatelným zpožděním. Po spuštění časovače, je výstup OC kanálu v úrovni L. Jakmile hodnota časovače dosáhne hodnoty naprogramované v registru CCR, dojde k nastavení výstupu OC kanálu do úrovně H. Dosáhne – li časovač hodnoty naprogramované v registru ARR, výstup se opět překlopí do úrovně L, časovač se vynuluje a zastaví. Tím je připraven na další spuštění a generování dalšího impulzu. Časové diagramy pro návrh ovladačů zapalovacích modulů, s využitím OC kanálů, jsou naznačeny na obr. 10.



Obr. 10: K návrhu ovladačů zapalovacích modulů.

Ze softwaru vyšší vrstvy ovladač obdrží požadovanou dobu nabíjení zapalovacího modulu t_{ch}. Dobu t_a musí ovladač zapalovacích modulů vypočítat z požadovaného zapalovacího úhlu, dodaného softwarem vyšší vrstvy, a rychlosti otáčení motoru, dodané modulem detekce polohy motoru, podle vztahu

$$t_a = \frac{\beta \cdot t_r}{3600} + t_s \quad [\text{s; °, s, s}], \tag{3}$$

kde β je zapalovací úhel, t_r je perioda otáčení klikového hřídele, t_s je synchronizační zpoždění, viz. strana 31.

Ovladač zapalovacích modulů musí vypočítat doby t_{ccr} a t_{arr}, a zjistit nejbližší polohu klikového hřídele, ve které je možné časovač spustit (p_s). Časy t_{ccr} a t_{arr} budou před spuštěním časovače vloženy do registrů CCR a ARR. Časovač bude poté spuštěn, jakmile klikový hřídel dosáhne polohy spuštění.

Nejdříve je nutné zjistit dobu t_b, podle vztahu

$$t_b = (t_{ch} + t_a) \, mod \, t_t \quad [\text{s; s, s, s}], \tag{4}$$

kde t_t je doba otočení klikového hřídele o 15°, tj.:

$$t_t = \frac{t_r}{24} \quad [\text{s; s}] \tag{5}$$

Je – li t_b nenulová (obr. 10, případ A), lze dobu t_{ccr} vypočítat vypočítat podle vztahu:

$$t_{ccr} = t_t - t_b \quad [\text{s; s, s}] \tag{6}$$

Dobu t_{arr} lze nyní vypočítat:

$$t_{arr} = t_{ccr} + t_{ch} + 1 \quad [\text{s; s, s}] \tag{7}$$

Přičtení jedničky ve vztahu (6) je nutné kvůli správné funkci OC kanálu, viz. [1] kap. 19.3.10.

Poloha spuštění časovače p_s je v tomto případě dána vztahem:

$$p_s = 44 - (t_{ch} + t_a) \cdot t_t + 1 \quad [-;\ s,\ s,\ s] \tag{8}$$

Situace obr. 10, případ B, nastává, vychází – li okamžik aktivace zapalovacího modulu do zubové mezery impulzního kola klikového hřídele. Tento stav je rozpoznán, jsou – li splněny podmínky:

$$(t_a + t_{ch}) > 7 \cdot t_t \quad \text{a} \quad (t_a + t_{ch}) \leq 10 \cdot t_t \tag{9}$$

Potom:

$$t_{ccr} = 10 \cdot t_t - t_{ch} - t_a \tag{10}$$

Pro t_{arr} platí vztah (7), $p_s = 36$.

Je – li t_b nulová (obr. 10, případ C), je nulová i doba t_{ccr}. Pro korektní funkci OC kanálu však registr CCR nemůže být nulový, proto je nastaveno $t_{ccr} = 1$ a $t_{arr} = t_{ch}$. Tímto nastavením však dojde ke zkrácení doby nabíjení t_{ch} o 1 μs. Dobu nabíjení však není možné zvýšit zpět inkrementací registru ARR. Tím by došlo ke zkreslení požadované hodnoty zapalovacího úhlu. V tomto případě tedy ponecháme dobu nabíjení o 1 μs kratší. Tato nepřesnost je vzhledem k nabíjecím dobám řádů milisekund zanedbatelná. Můžeme psát:

$$t_{ccr} = 1 \tag{11}$$

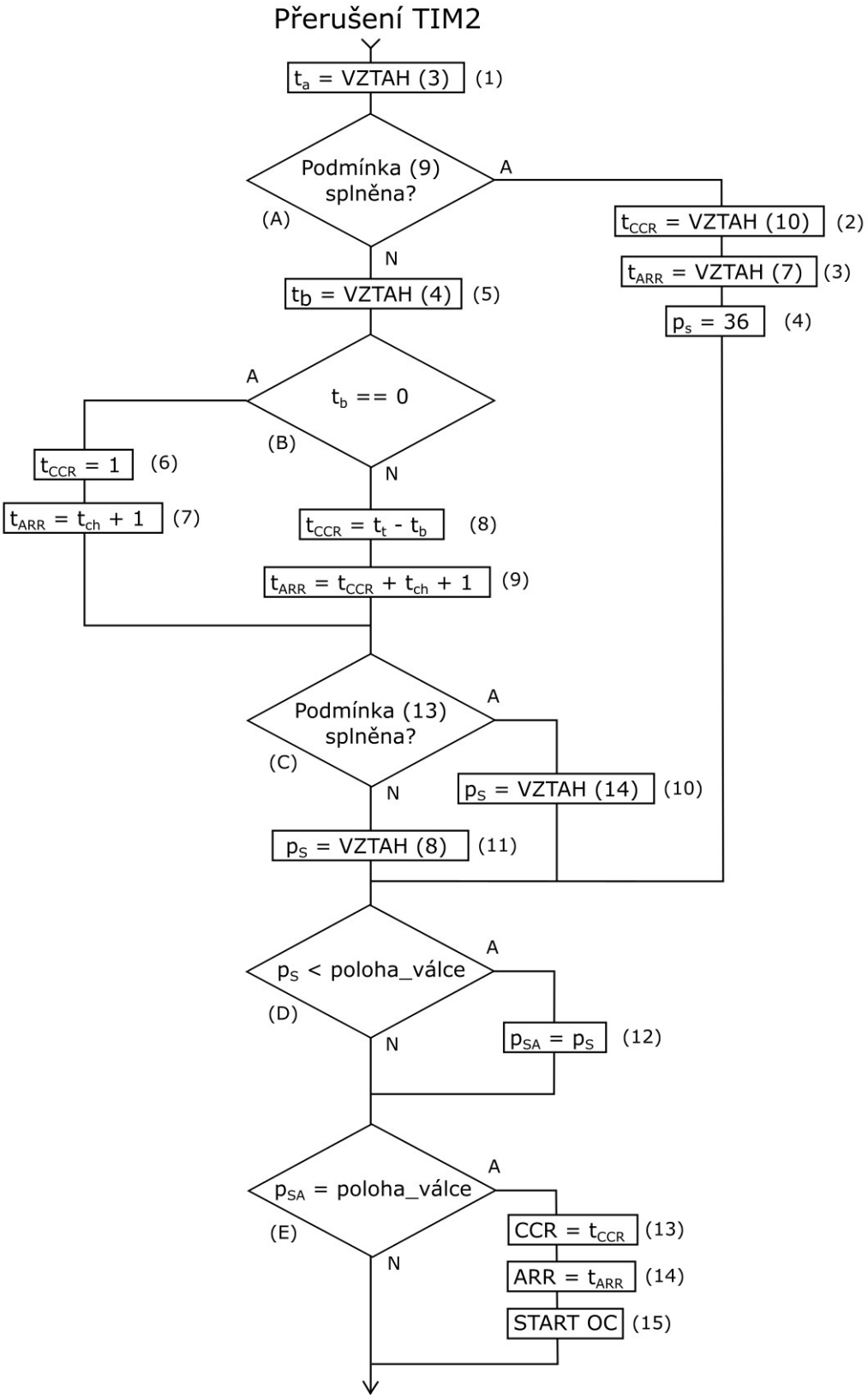
$$t_{arr} = t_{ch} + 1 \tag{12}$$

Je – li splněna podmínka:

$$(t_a + t_{ch}) > 10 \cdot t_t \quad , \tag{13}$$

nastává situace obr. 10, případ D. Pro tento případ platí vztahy (4), (5), (6) a (7), liší se pouze výpočet p_s:

$$p_s = 46 - (t_{ch} + t_a) \cdot t_t + 1 \tag{14}$$



Obr. 11: Vývojový diagram procedury ovladačů zapalovacích modulů.

Výše uvedené výpočty jsou algoritmizovány na obr. 11. Procedura z obr. 11 běží v přerušení TIM2 (CKP), výpočet probíhá pro každý válec. Ve zdrojovém kódu jsou procedury umístěny za procedurou výpočtu rychlosti otáčení motoru. Tato musí být v době běhu ovladačů zapalovacích modulů již známa. Po dokončení procedur ovladačů zapalovacích modulů probíhá vlastní spouštění časovačů. Spuštění musí být synchronní s polohou klikového hřídele. Jelikož před spuštěním časovačů běží v přerušení TIM2 ještě tři procedury, jejichž dobu běhu nelze přesně určit, musí být provedena synchronizace. Tato je zajištěna vyčkáním do doby, než časovač TIM2 načítá o 10 µs více, než je hodnota CCR (čas zachycení impulsu CKP). Za dobu 10 µs předchozí procedury s jistotou proběhnou. Ovladače zapalovacích modulů je možné nastavit pomocí doby synchronizace t_s (vztah (11)), aby počítaly s tím, že spuštění jejich časovačů proběhne o dobu t_s později. Spouštěním ovladačů i časovačů zapalovacích modulů přímo v přerušení CKP je dosaženo nejvyšší přesnosti řízení zapalovacího úhlu, zejména při změnách rychlosti otáčení klikového hřídele. Zároveň je zaručena přesná synchronizace řízení zapalování s polohou klikového hřídele. Při spouštění časovačů ovladačů zapalovacích modulů je zapotřebí dbát zvýšené opatrnosti. Nastane – li situace, kdy ovladač požaduje start časovače v poloze motoru o 1 vyšší, než je aktuální, ke startu nedojde – mělo by k němu dojít v následující poloze klikového hřídele. Pokud však v následující poloze klikového hřídele ovladač rozhodne, že požaduje start časovače v poloze předchozí, ke startu rovněž nedojde – předchozí poloha již tento pracovní cyklus proběhla. Výsledkem je vynechání zážehu jeden pracovní cyklus dotyčného válce. Toto vynechání je nepřípustná chyba. Je tedy nutné ovladači zamezit snížení hodnoty p_s „na poslední chvíli“. Toto je zajištěno podmínkami (D), (E) a příkazem (12) na obr. 11, kde signál p_{sa} je akceptovaná poloha spuštění časovače.

Tab. 1: Přiřazení signálů o poloze každého válce pro ovladače zapalovacích modulů:

Poloha motoru		Poloha válce 1		Poloha válce 2		Poloha válce 3		Poloha válce 4	
id	[°]	id	[°]	id	[°]	id	[°]	id	[°]
1	0	37	105	25	285	3	645	15	465
2	15	38	90	26	270	4	630	16	450
3	30	39	75	27	255	5	615	17	435
4	45	40	60	28	240	6	600	18	420
5	60	41	45	29	225	7	585	19	405
6	75	42	30	30	210	8	570	20	390
7	90	43	15	31	195	9	555	21	375
8	105	44	0	32	180	10	540	22	360
9	120	1	705	33	165	11	525	23	345
10	135	2	690	34	150	12	510	24	330
11	150	3	675	35	135	13	495	25	315
12	165	4	660	36	120	14	480	26	300
13	180	5	645	37	105	15	465	27	285
14	195	6	630	38	90	16	450	28	270
15	210	7	615	39	75	17	435	29	255
16	225	8	600	40	60	18	420	30	240
17	240	9	585	41	45	19	405	31	225
18	255	10	570	42	30	20	390	32	210
19	270	11	555	43	15	21	375	33	195
20	285	12	540	44	0	22	360	34	180
21	300	13	525	1	705	23	345	35	165
22	315	14	510	2	690	24	330	36	150
-	330	-	495	-	675	-	315	-	135
-	345	-	480	-	660	-	300	-	120
23	360	15	465	3	645	25	285	37	105
24	375	16	450	4	630	26	270	38	90
25	390	17	435	5	615	27	255	39	75
26	405	18	420	6	600	28	240	40	60
27	420	19	405	7	585	29	225	41	45
28	435	20	390	8	570	30	210	42	30
29	450	21	375	9	555	31	195	43	15
30	465	22	360	10	540	32	180	44	0
31	480	23	345	11	525	33	165	1	705
32	495	24	330	12	510	34	150	2	690
33	510	25	315	13	495	35	135	3	675
34	525	26	300	14	480	36	120	4	660
35	540	27	285	15	465	37	105	5	645
36	555	28	270	16	450	38	90	6	630
37	570	29	255	17	435	39	75	7	615
38	585	30	240	18	420	40	60	8	600
39	600	31	225	19	405	41	45	9	585
40	615	32	210	20	390	42	30	10	570
41	630	33	195	21	375	43	15	11	555
42	645	34	180	22	360	44	0	12	540
43	660	35	165	23	345	1	705	13	525
44	675	36	150	24	330	2	690	14	510
-	690	-	135	-	315		675	-	495
-	705	-	120	-	300		660	-	480

6 Návrh ovladačů vstřikovacích ventilů

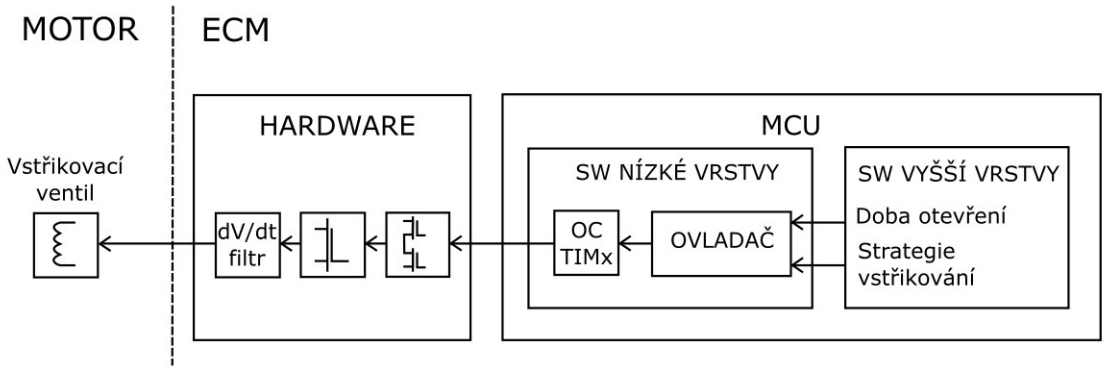
6.1 Požadované výkony ovladačů vstřikovacích ventilů

Software vyšší vrstvy vypočítá a poskytne požadovanou délku otevření vstřikovacího ventilu pro každý válec. Poskytne rovněž požadovanou strategii vstřikování. Úkolem ovladačů vstřikovacích ventilů je vstřikovací ventil po tuto dobu otevřít, v okamžiku odpovídajícím požadované strategii vstřikování.

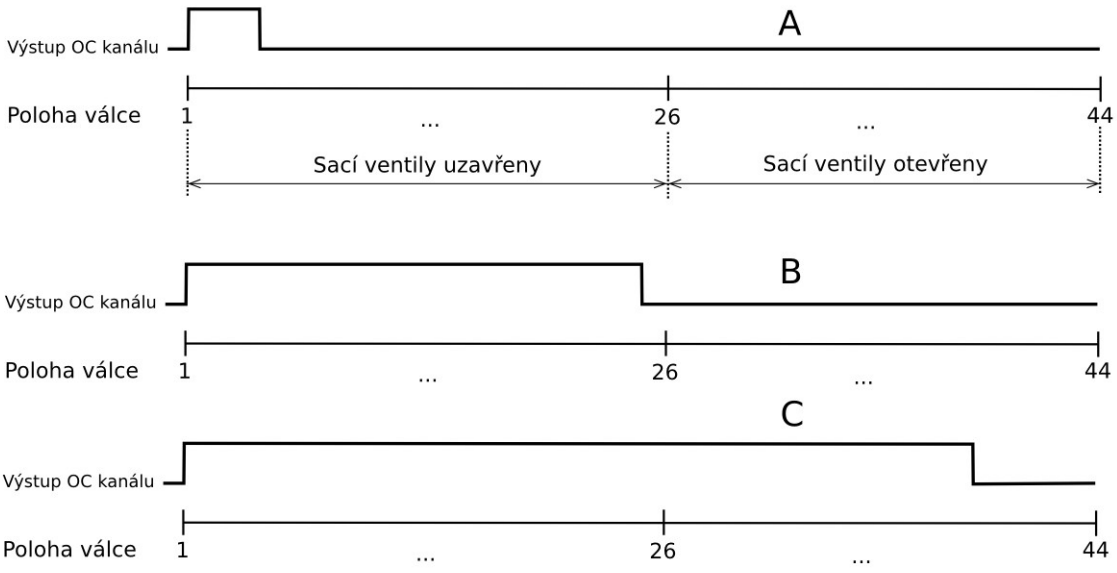
6.2 Princip činnosti ovladačů vstřikovacích ventilů

V současné době je v softwaru nízké vrstvy implementována nejpoužívanější strategie vstřikování ECVI. V této strategii dochází k počátku vstřikování paliva vždy právě po uzavření vstřikovacích ventilů. Strategie poskytuje dostatek času pro dobré odpaření a smísení dávky paliva se vzduchem při nízkých výkonech motoru. Při nízkých výkonech rovněž není na škodu ohřátí palivové směsi v sacím kanálu a její objemové zvětšení, způsobené dlouhou dobou od vstřiku dávky paliva do okamžiku otevření sacích ventilů. Při vyšších výkonech (vyšších dávkách paliva) přesahuje doba vstřikování přes otevřené sací ventily. Dochází tak k ochlazení přiváděného vzduchu o měrné skupenské teplo odparu dávky paliva. Toto ochlazení přispívá ke snížení měrného objemu palivové směsi a tím ke zvýšení výkonu motoru.

Pro zjednodušení ovladačů vstřikovacích ventilů byly, podobně jako u ovladačů zapalovacích modulů, zavedeny signály polohy pístů každého válce, vypočítané z globální (hlavní) polohy motoru, poskytnuté detektorem polohy motoru. Pro čtyřválcový motor s pořadím zapalování 1-2-4-3 a odstupem zapalování 180° jsou signály o poloze každého válce přiřazeny podle Tab. 2.



Obr. 12: Blokové schéma uspořádání pro řízení vstřikovacích ventilů.



Obr. 13: Princip činnosti strategie vstřikování paliva ECVI.

K obr. 13:

Situace *A* zobrazuje ovládání vstřikovacího ventilu při nízkých výkonech motoru, situace *B* při středních výkonech motoru, situace *C* při výkonech vysokých.

Tab. 2: Přiřazení signálů o poloze každého válce pro ovladače vstřikovacích ventilů:

Poloha motoru		Poloha válce 1		Poloha válce 2		Poloha válce 3		Poloha válce 4	
id	[°]	id	[°]	id	[°]	id	[°]	id	[°]
1	0	1	0	33	540	11	180	23	360
2	15	2	15	34	555	12	195	24	375
3	30	3	30	35	570	13	210	25	390
4	45	4	45	36	585	14	225	26	405
5	60	5	60	37	600	15	240	27	420
6	75	6	75	38	615	16	255	28	435
7	90	7	90	39	630	17	270	29	450
8	105	8	105	40	645	18	285	30	465
9	120	9	120	41	660	19	300	31	480
10	135	10	135	42	675	20	315	32	495
11	150	11	150	43	690	21	330	33	510
12	165	12	165	44	705	22	345	34	525
13	180	13	180	1	0	23	360	35	540
14	195	14	195	2	15	24	375	36	555
15	210	15	210	3	30	25	390	37	570
16	225	16	225	4	45	26	405	38	585
17	240	17	240	5	60	27	420	39	600
18	255	18	255	6	75	28	435	40	615
19	270	19	270	7	90	29	450	41	630
20	285	20	285	8	105	30	465	42	645
21	300	21	300	9	120	31	480	43	660
22	315	22	315	10	135	32	495	44	675
-	330	-	330	-	150	-	510	-	690
-	345	-	345	-	165	-	525	-	705
23	360	23	360	11	180	33	540	1	0
24	375	24	375	12	195	34	555	2	15
25	390	25	390	13	210	35	570	3	30
26	405	26	405	14	225	36	585	4	45
27	420	27	420	15	240	37	600	5	60
28	435	28	435	16	255	38	615	6	75
29	450	29	450	17	270	39	630	7	90
30	465	30	465	18	285	40	645	8	105
31	480	31	480	19	300	41	660	9	120
32	495	32	495	20	315	42	675	10	135
33	510	33	510	21	330	43	690	11	150
34	525	34	525	22	345	44	705	12	165
35	540	35	540	23	360	1	0	13	180
36	555	36	555	24	375	2	15	14	195
37	570	37	570	25	390	3	30	15	210
38	585	38	585	26	405	4	45	16	225
39	600	39	600	27	420	5	60	17	240
40	615	40	615	28	435	6	75	18	255
41	630	41	630	29	450	7	90	19	270
42	645	42	645	30	465	8	105	20	285
43	660	43	660	31	480	9	120	21	300
44	675	44	675	32	495	10	135	22	315
-	690	-	690	-	510	-	150	-	330
-	705	-	705	-	525	-	165	-	345

6.3 Realizace ovladačů vstřikovacích ventilů

Podobně, jako u ovladačů zapalovacích modulů, lze s výhodou využít OC kanálů hardwarových časovačů mikrokontroléru, nakonfigurovaných v „One-pulse“ módu. Perioda jejich časové základny je nastavena na 1 μs. Jsou – li určeny pracovní polohy všech válců pro ovladače vstřikovacích ventilů (Tab. 2), není realizace ovladačů pro strategii vstřikování ECVI složitá. Časovače ovladačů jsou spouštěny vždy v polohách válců $p_s = 1$. Do CCR registrů je vždy vkládána hodnota 1 (nejrychlejší aktivace výstupů do úrovně H), do registrů ARR je vždy vkládána doba otevření vstřikovacích ventilů + 2. Přičtení další jedničky je nutné pro kompenzaci zpoždění otevření vstřikovacího ventilu o 1 μs (CCR = 1), aby zůstala zachována požadovaná doba otevření vstřikovacího ventilu. K uzavření vstřikovacího ventilu tedy dojde u 1 μs později. Toto zpoždění však nemá na funkci motoru žádný vliv. Konfigurace časovačů a jejich OC kanálů pro ovladače vstřikovacích ventilů je provedena ve zdrojovém souboru *HWD_Init.c*. Vlastní spouštění časovačů probíhá synchronně s polohou klikového hřídele v proceduře přerušení TIM2, zdrojový soubor *HWD_CKP_Isr.c*.

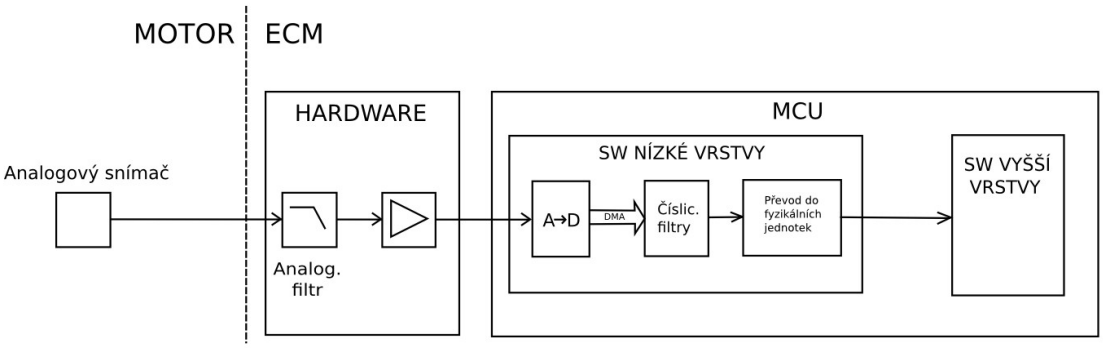
7 Návrh zpracování analogových vstupních signálů

7.1 Požadavky na zpracování analogových signálů

Systém řízení benzinového motoru je vybaven celkem deseti analogovými vstupy. Devět vstupů bude vzorkováno periodicky, zbývající jeden musí být možno vzorkovat synchronně s polohou klikového hřídele.

Prostředí motorového vozidla je významně ovlivněno elektromagnetickými rušivými signály. Kabelová vedení analogových snímačů jsou protažena ve společných kabelových svazcích s vodiči výkonovými. Z těchto důvodů je nutné vstupní analogové signály zabezpečit proti rušení.

7.2 Realizace zpracování analogových signálů



Obr. 14.: Blokové schéma realizace zpracování analogových signálů.

Pro A/D převod periodicky vzorkovaných signálů bude použit A/D převodník ADC1. Tento bude nakonfigurován v módech „Scan“ a „Continuous“ s použitím DMA. Takto nakonfigurovaný A/D převodník automaticky vzorkuje a převádí předem naprogramovanou sekvenci vstupních kanálů a data předává do paměti s využitím DMA. Po převodu všech vstupních kanálů A/D převodník sekvenci opět automaticky opakuje. DMA kanál převedené vzorky ukládá do pole o velikosti 126 vzorků v paměti RAM, nazvaného *HWD_ADCDrv_ADC1BUFDMA*. Po převodu čtrnácti sekvencí ($14 \cdot 9 = 126$) je pole v paměti zaplněno. DMA kanál je nastaven v „Circular“ módu, začne tedy pole zaplňovat daty opět od začátku. Stará data jsou přepisována. Kompletní nastavení ADC1 a DMA je provedeno ve zdrojovém souboru *HWD_Init.c*.

Periodicky, každých 10 ms, je spouštěna procedura softwaru nízké vrstvy pro zpracování dat, která jsou naplněna v bufferu *HWD_ADCDrv_ADC1BUFDMA*. Z těchto dat je zde vytvořeno trojrozměrné datové pole, ve kterém jsou připravena data pro vykonání mediánové filtrace. Toto pole, nazvané *HWD_ADCDrv_ADC1BUFMED*, o rozměrech 9 x 10 x 5 vzorků, obsahuje data pro devět kanálů, a pro každý kanál 10 pěticí vzorků pro vykonání mediánové filtrace. Následně je vykonána mediánová filtrace, jejímž výstupem je deset filtrovaných vzorků pro každý kanál. Z těchto vzorků jsou poté vypočítány průměrné hodnoty pro každý signál. Konečné výsledky jsou uloženy do pole *HWD_ADCDrv_ADC1RES*. Procedura je implementována zdrojovým souborem *HVD_ADCDrv.c*.

Pro desátý analogový vstup je použit A/D převodník ADC2. Je nastaven tak, aby jej bylo možné

spustit softwarově – nastavením bitu *SWSTART* v registru *ADC2_CR2*. Takto je možné A/D převod spustit z přerušení snímače CKP v požadované poloze klikového hřídele. Po dokončení A/D převodu je výsledek je uložen v bufferu A/D převodníku *ADC_DR*.

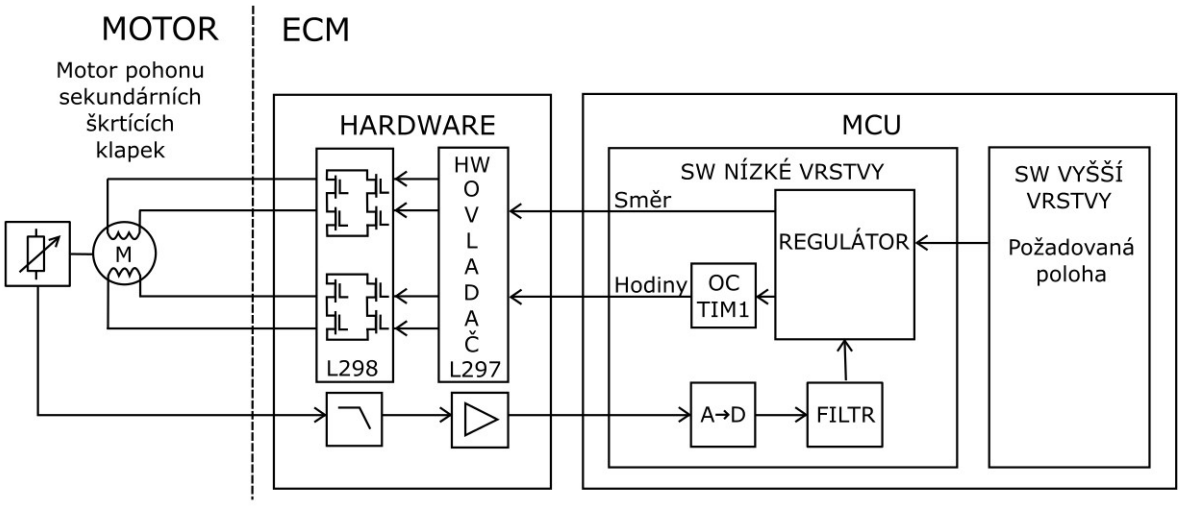
Konečné výsledky A/D převodů jsou dále přepočteny do fyzikálních jednotek příslušných měřených veličin. Signály nelineárních snímačů (NTC termistorů) jsou linearizovány a převedeny na fyzikální jednotky pomocí jednorozměrných tabulek v paměti flash, udávajících ve dvanácti bodech přenosovou charakteristiku příslušného snímače. Mezi body tabulky je provedena lineární interpolace. Funkce pro lineární interpolace je implementována ve zdrojovém souboru *HWD_GlobalFunctions.c*, je využívána různými procedurami a funkcemi napříč softwarem nízké vrstvy.

8 Návrh regulátoru polohy sekundárních škrtících klapek

8.1 Požadavky na regulátor polohy sekundárních škrtících klapek

Software vyšší vrstvy poskytne požadovanou polohu sekundárních škrtících klapek. Software nízké vrstvy musí za použití signálu snímače polohy sekundárních škrtících klapek a krokového motoru pohonu sekundárních škrtících klapek požadovanou polohu nastavit. Regulace musí probíhat bez kmitání a musí být odolná proti šumu měření.

8.2 Realizace regulátoru polohy sekundárních škrtících klapek

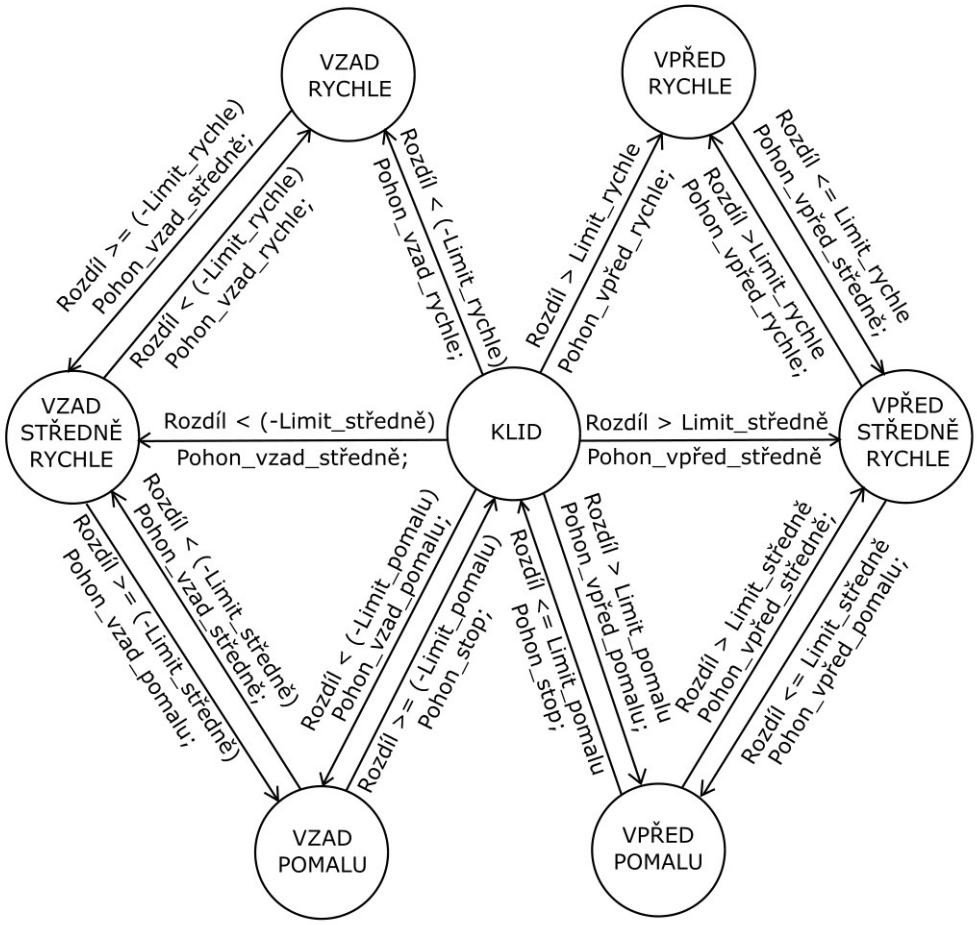


Obr. 15: Blokové schéma uspořádání regulátoru polohy sekundárních škrtících klapek.

Signál snímače polohy sekundárních škrtících klapek je vzorkován v 10 ms periodě A/D převodníkem a následně filtrován v modulu zpracování analogových signálů (viz. Kap. 7). Následně je použit regulátorem polohy sekundárních škrtících klapek. Regulátor polohy sekundárních škrtících klapek je stavový program (obr. 16), který je spouštěn cyklicky s periodou 10 ms. Na základě rozdílu požadované a skutečné polohy sekundárních škrtících klapek vykonává akční zásahy řízením krokového motoru pohonu škrtících klapek. Výstup regulátoru má 7 stavů:

- Klidový
- Pohon vpřed pomalý (frekvence krokování 200 Hz)
- Pohon vpřed středně rychlý (frekvence krokování 400 Hz)
- Pohon vpřed rychlý (frekvence krokování 800 Hz)
- Pohon vzad pomalý (frekvence krokování 200 Hz)
- Pohon vzad středně rychlý (frekvence krokování 400 Hz)
- Pohon vzad rychlý (frekvence krokování 800 Hz)

Hardware ECM je vybaven dvojitým H – můstkem pro pohon krokového motoru typu L298 a ovladačem krokového motoru typu L297. Tato konfigurace výrazně sníží požadavky na software řízení krokového motoru. Ovladači L297 jsou z regulátoru pohonu v mikrokontroléru předány pouze dva signály – diskrétní signál o požadovaném směru chodu motoru, a obdélníkový signál určující rychlost chodu motoru (s každou náběžnou hranou tohoto signálu vykoná ovladač L297 jeden krok pohonu). Aby bylo možné libovolně kalibrovat rychlost pohonu klapek, je obdélníkový signál generován v mikrokontroléru za použití OC kanálu TIM1 v režimu PWM. Stavový regulátor předává OC kanálu pouze požadovanou periodu kroku pohonu. Periodu kroku pohonu regulátor určí v závislosti na rozdílu požadované a aktuální polohy sekundárních škrtících klapek. Je – li rozdíl vysoký, volí nejvyšší rychlost pohonu. Blíží – li se skutečná poloha klapek k požadované poloze, regulátor zpomalí nejdříve na střední rychlost. Konečné dosažení požadované polohy je provedeno nejnižší rychlostí. Tímto způsobem je zabráněno kmitání regulátoru. Je – li zapotřebí pohon zastavit, regulátor zastaví běh čítače TIM1. Požadovaný směr chodu krokového motoru regulátor předá ovladači L297 jedním pinem GPIO portu mikrokontroléru. Procedura regulátoru polohy sekundárních škrtících klapek je implementována ve zdrojovém souboru *HWD_STVControl.c*.



Obr. 16: Uspořádání stavového programu regulátoru polohy sekundárních škrtících klapek.

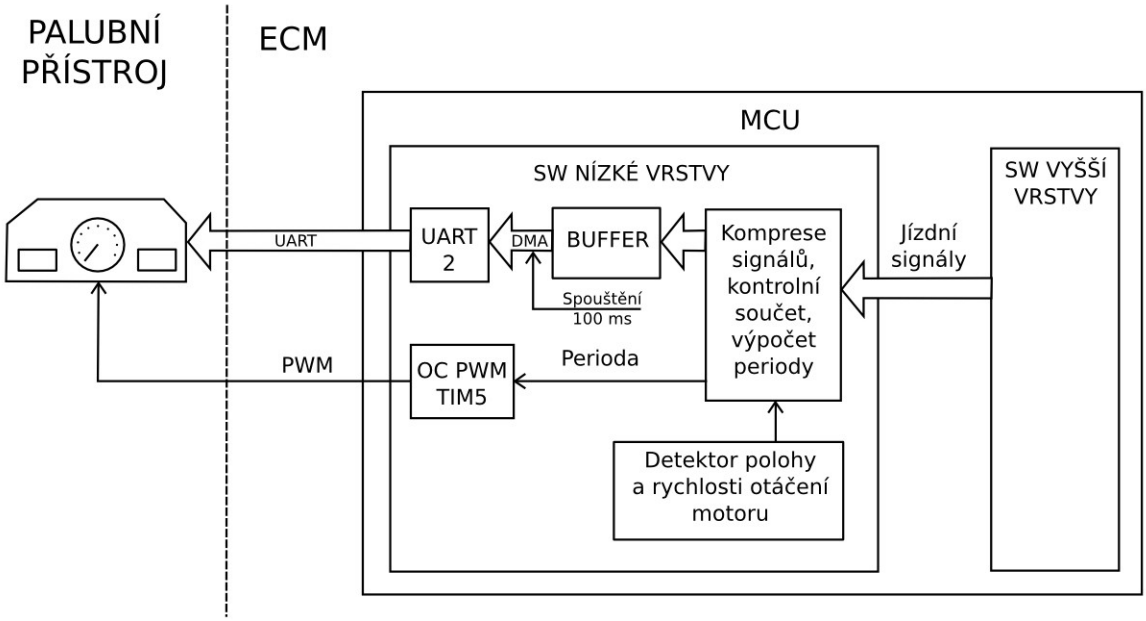
9 Návrh modulu pro komunikaci s palubním přístrojem

9.1 Požadavky na modul pro komunikaci s palubním přístrojem

Software vyšší vrstvy poskytne jízdní data pro palubní počítač. Tato data jsou tvořena signály o teplotě chladící kapaliny motoru, zařazeném rychlostním stupni v převodovce, přítomnosti závady na systému řízení motoru a rychlosti otáčení motoru.

Data software nízké vrstvy zkomprimuje, zabezpečí kontrolním součtem a odešle palubnímu přístroji po sériové lince UART o rychlosti 7800 baud/s. Data budou odesílána v paketech o osmi bytech, poslední byte je kontrolní součet. Datové pakety budou odesílány cyklicky s periodou 100 ms. Pro přenos dat z paměti do kontroléru UART bude použit kanál DMA. Tímto způsobem jsou mezi ECM a palubním přístrojem přeneseny všechny potřebné signály, mimo signálu o rychlosti otáčení motoru. Pro přenos tohoto signálu je perioda obnovení 100 ms příliš pomalá a zobrazení dynamických změn rychlosti otáčení motoru na otáčkoměru palubního přístroje by neodpovídalo realitě. Proto je tento signál přenášen zvláštní linkou za použití modulace PWM.

9.2 Realizace komunikace s palubním přístrojem



Obr. 17: Blokové schéma realizace komunikace s palubním přístrojem.

Pro odesílání sériové komunikace palubnímu přístroji je použit modul UART2. Komunikace je pouze jednostranná, modul UART2 je aktivní pouze v TX módu. Cyklicky, s periodou 100 ms, je spouštěna procedura softwaru nízké vrstvy, která shromáždí jízdní data, komprimuje je do jednobytových slov a vypočítá kontrolní součet. Připravená data včetně kontrolního součtu uloží do pole *HWD_DashBoardOut_TXBUF*. Poté aktivuje přenos dat pomocí DMA. Nyní již může procedura skončit, samotné odeslání dat je provedeno samočinně součinností DMA a modulu UART2.

Pro generování PWM modulovaného signálu je použit časovač TIM5 a jeho kanál OC1 v režimu PWM. Pro zachování dynamické přesnosti PWM signálu je perioda PWM v registru TIM5->ARR aktualizována cyklicky s periodou 10 ms.

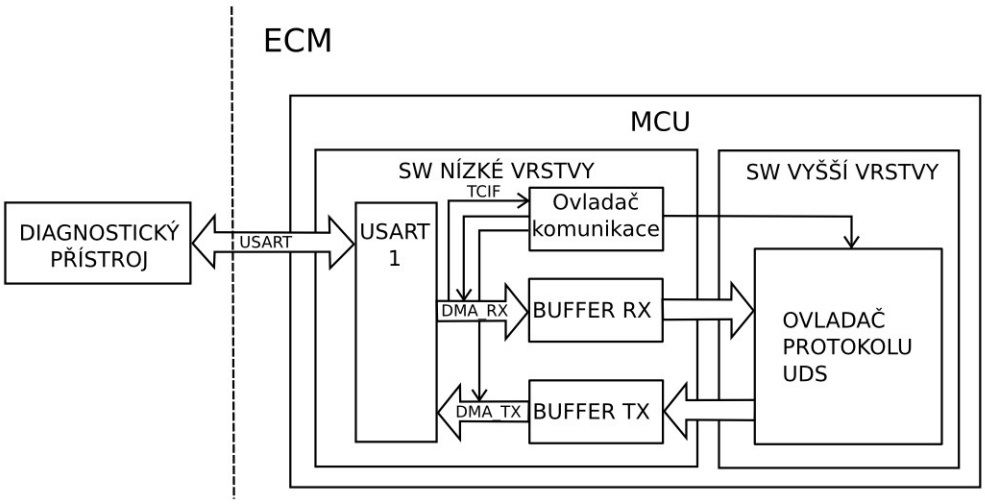
Konfigurace DMA, TIM5 a UART2 je implementována ve zdrojovém souboru *HWD_Init.c*, procedura zpracování sériových dat ve zdrojovém souboru *HWD_DashBoardOut.c*, procedura zpracování PWM signálu o rychlosti otáčení motoru ve zdrojovém souboru *HWD_TachometerOut.c*.

10 Návrh modulu pro komunikaci s diagnostickým přístrojem

10.1 Požadavky na modul komunikace s diagnostickým přístrojem

Komunikace systému řízení benzinového motoru s diagnostickým zařízením probíhá po sériové sběrnici UART obousměrně, po paketech o délce 8 bytů. DMA kanály jsou použity pro oba směry komunikace.

10.2 Realizace modulu komunikace s diagnostickým přístrojem



Obr. 18: Blokové schéma realizace komunikace s diagnostickým přístrojem.

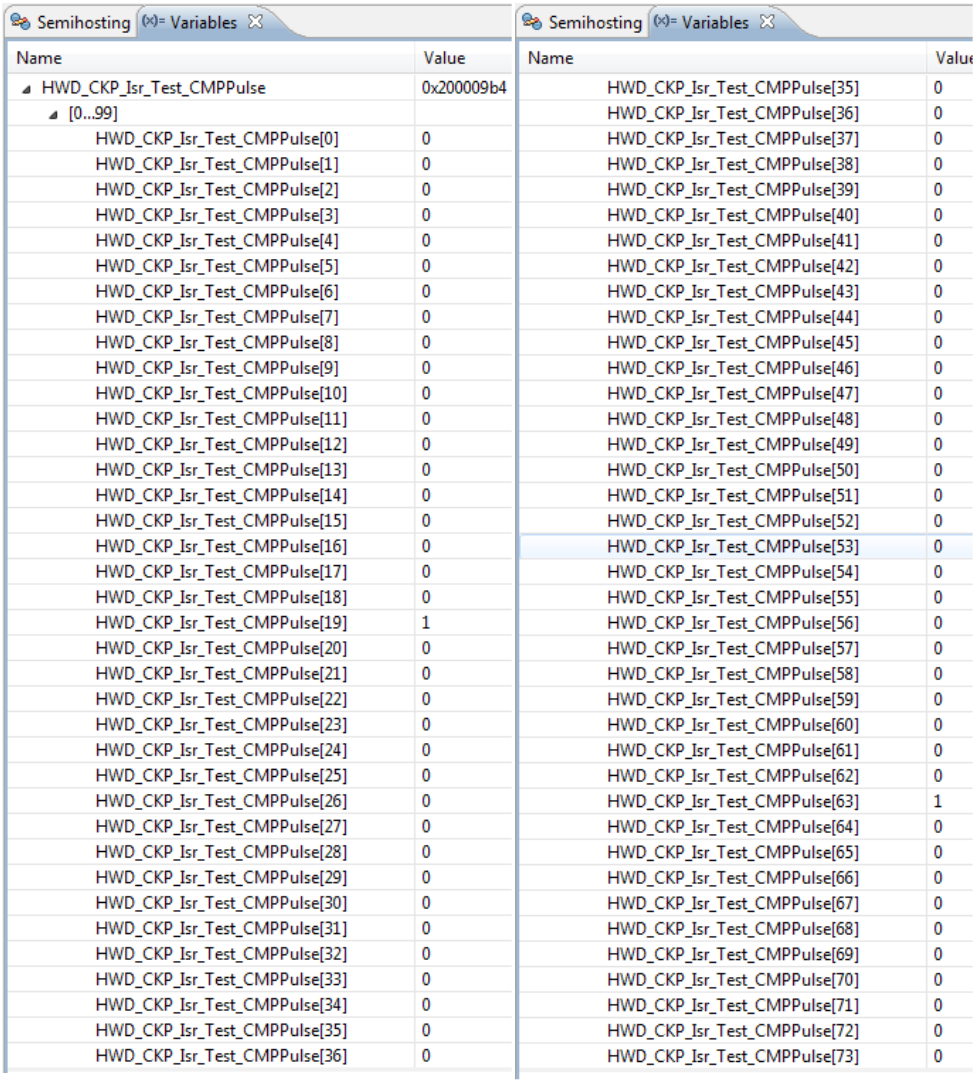
Komunikace je realizována prostřednictvím modulu USART1. DMA kanál pro příjem dat ukládá každý přijatý byte do pole *HWD_USART1Drv_RXBUF* o velikosti 8 bytů. DMA kanál po naplnění pole přijatými daty nastaví příznak dokončení přenosu dat TCIF (viz. [1] kap. 10.5.2). Tento příznak sleduje 10 ms cyklická procedura softwaru nízké vrstvy *HWD_USART1Drv.* Zjistí – li nastavený příznak TCIF DMA RX kanálu, zavolá proceduru softwaru vyšší vrstvy *Diag_ComDrv*. V této proceduře je implementován ovladač protokolu UDS. Procedura softwaru vyšší vrstvy přečte data, uložená v poli *HWD_USART1Drv_RXBUF,* provede jejich zpracování a 8 bytů dlouhou odpověď diagnostickému zařízení připraví do bufferu *HWD_USART1Drv_TXBUF.* Poté procedura softwaru vyšší vrstvy skončí a vrátí procesor zpět do softwaru nižší vrstvy, kde dojde k nastavení DMA RX kanálu pro příjem nového paketu a k odeslání obsahu pole *HWD_USART1Drv_TXBUF* spuštěním datového přenosu kanálu DMA TX. K samotnému odeslání paketu již dojde samočinně.

11 Testování navrženého řešení softwaru nízké vrstvy

11.1 Test detekce pracovní polohy a fáze motoru

Pro testovací a kalibrační účely byl zhotoven simulátor signálů snímačů CKP a CMP (viz. [3]). Tento je připojen ke vstupům CKP a CMP řídící jednotky motoru a simuluje signály snímačů tak, aby byly shodné se signály, které jsou snímači generovány za chodu motoru.

Za účelem otestování detektoru polohy motoru byla v softwaru nízké vrstvy deklarována globální pole pro ukládání výsledných signálů. Velikost polí je 256 vzorků a zápis dat probíhá způsobem „ring“. Výsledné signály jsou do polí uloženy vždy po proběhnutí procedury detekce polohy motoru. Obsah polí je možné kdykoliv zobrazit pomocí debuggeru. Na obr. 19 – 21 je zachycen proces synchronizace polohy motoru po jeho rozběhnutí a sledování jeho polohy za chodu. Dokud není zaznamenán impulz CMP (signál *HWD_CKP_Isr_Test_CMPPulse* ≠ 1), poloha motoru není známa (signál *HWD_CKP_Isr_Test_EngPos* = 0), a nemůže být synchronizována (signál *HWD_CKP_Isr_Test_EngPosSync* = 0). Po zjištění impulsu CMP (signál *HWD_CKP_Isr_Test_CMPPulse* = 1), je poloha motoru detekována, synchronizována a inkrementována celý pracovní cyklus motoru. Chod detektoru po výpadku signálu CMP za provozu byl vyzkoušen jeho odpojením. Vstřikovací ventily i zapalovací moduly pokračovaly v činnosti, je tedy zřejmé, že detektor polohy motoru pracuje správně i po výpadku signálu CMP.



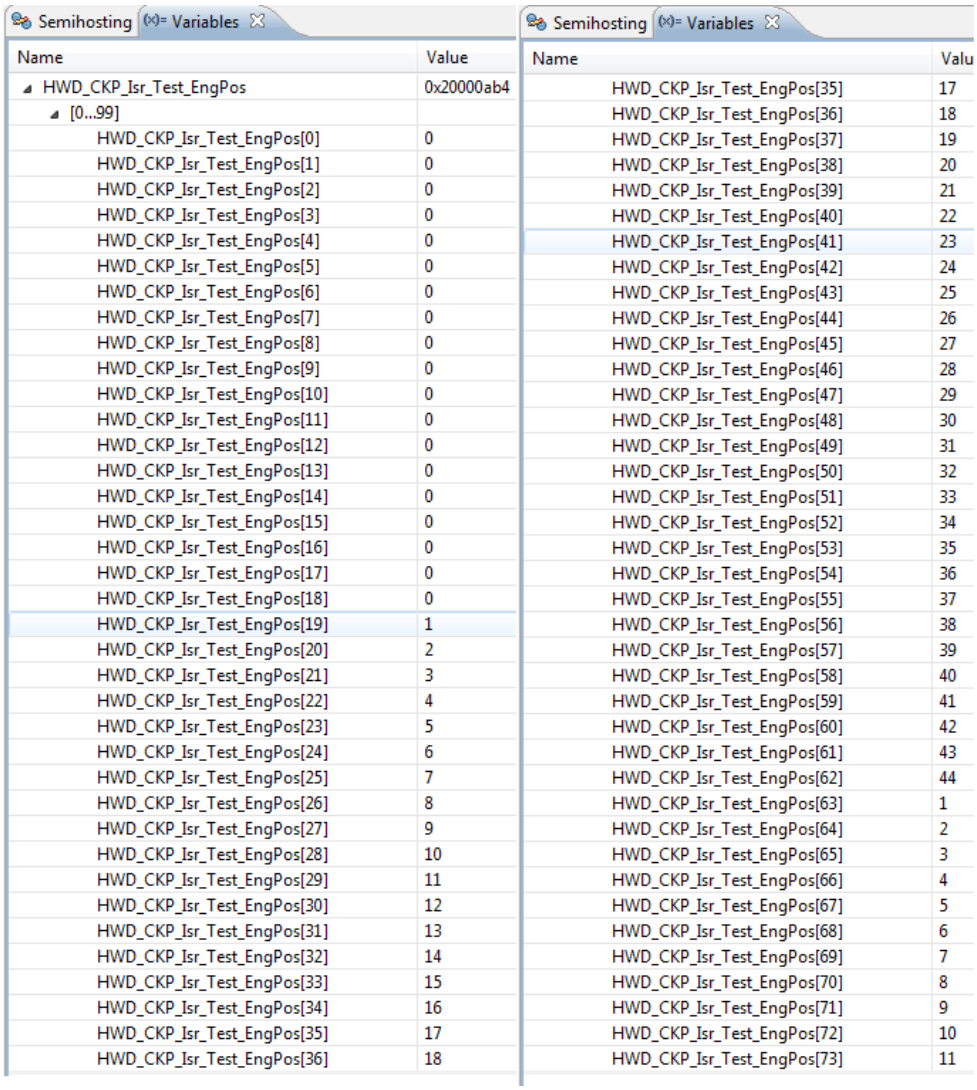
Semihosting | Variables

Name	Value
HWD_CKP_Isr_Test_CMPPulse	0x200009b4
[0...99]	
HWD_CKP_Isr_Test_CMPPulse[0]	0
HWD_CKP_Isr_Test_CMPPulse[1]	0
HWD_CKP_Isr_Test_CMPPulse[2]	0
HWD_CKP_Isr_Test_CMPPulse[3]	0
HWD_CKP_Isr_Test_CMPPulse[4]	0
HWD_CKP_Isr_Test_CMPPulse[5]	0
HWD_CKP_Isr_Test_CMPPulse[6]	0
HWD_CKP_Isr_Test_CMPPulse[7]	0
HWD_CKP_Isr_Test_CMPPulse[8]	0
HWD_CKP_Isr_Test_CMPPulse[9]	0
HWD_CKP_Isr_Test_CMPPulse[10]	0
HWD_CKP_Isr_Test_CMPPulse[11]	0
HWD_CKP_Isr_Test_CMPPulse[12]	0
HWD_CKP_Isr_Test_CMPPulse[13]	0
HWD_CKP_Isr_Test_CMPPulse[14]	0
HWD_CKP_Isr_Test_CMPPulse[15]	0
HWD_CKP_Isr_Test_CMPPulse[16]	0
HWD_CKP_Isr_Test_CMPPulse[17]	0
HWD_CKP_Isr_Test_CMPPulse[18]	0
HWD_CKP_Isr_Test_CMPPulse[19]	1
HWD_CKP_Isr_Test_CMPPulse[20]	0
HWD_CKP_Isr_Test_CMPPulse[21]	0
HWD_CKP_Isr_Test_CMPPulse[22]	0
HWD_CKP_Isr_Test_CMPPulse[23]	0
HWD_CKP_Isr_Test_CMPPulse[24]	0
HWD_CKP_Isr_Test_CMPPulse[25]	0
HWD_CKP_Isr_Test_CMPPulse[26]	0
HWD_CKP_Isr_Test_CMPPulse[27]	0
HWD_CKP_Isr_Test_CMPPulse[28]	0
HWD_CKP_Isr_Test_CMPPulse[29]	0
HWD_CKP_Isr_Test_CMPPulse[30]	0
HWD_CKP_Isr_Test_CMPPulse[31]	0
HWD_CKP_Isr_Test_CMPPulse[32]	0
HWD_CKP_Isr_Test_CMPPulse[33]	0
HWD_CKP_Isr_Test_CMPPulse[34]	0
HWD_CKP_Isr_Test_CMPPulse[35]	0
HWD_CKP_Isr_Test_CMPPulse[36]	0

Semihosting | Variables

Name	Value
HWD_CKP_Isr_Test_CMPPulse[35]	0
HWD_CKP_Isr_Test_CMPPulse[36]	0
HWD_CKP_Isr_Test_CMPPulse[37]	0
HWD_CKP_Isr_Test_CMPPulse[38]	0
HWD_CKP_Isr_Test_CMPPulse[39]	0
HWD_CKP_Isr_Test_CMPPulse[40]	0
HWD_CKP_Isr_Test_CMPPulse[41]	0
HWD_CKP_Isr_Test_CMPPulse[42]	0
HWD_CKP_Isr_Test_CMPPulse[43]	0
HWD_CKP_Isr_Test_CMPPulse[44]	0
HWD_CKP_Isr_Test_CMPPulse[45]	0
HWD_CKP_Isr_Test_CMPPulse[46]	0
HWD_CKP_Isr_Test_CMPPulse[47]	0
HWD_CKP_Isr_Test_CMPPulse[48]	0
HWD_CKP_Isr_Test_CMPPulse[49]	0
HWD_CKP_Isr_Test_CMPPulse[50]	0
HWD_CKP_Isr_Test_CMPPulse[51]	0
HWD_CKP_Isr_Test_CMPPulse[52]	0
HWD_CKP_Isr_Test_CMPPulse[53]	0
HWD_CKP_Isr_Test_CMPPulse[54]	0
HWD_CKP_Isr_Test_CMPPulse[55]	0
HWD_CKP_Isr_Test_CMPPulse[56]	0
HWD_CKP_Isr_Test_CMPPulse[57]	0
HWD_CKP_Isr_Test_CMPPulse[58]	0
HWD_CKP_Isr_Test_CMPPulse[59]	0
HWD_CKP_Isr_Test_CMPPulse[60]	0
HWD_CKP_Isr_Test_CMPPulse[61]	0
HWD_CKP_Isr_Test_CMPPulse[62]	0
HWD_CKP_Isr_Test_CMPPulse[63]	1
HWD_CKP_Isr_Test_CMPPulse[64]	0
HWD_CKP_Isr_Test_CMPPulse[65]	0
HWD_CKP_Isr_Test_CMPPulse[66]	0
HWD_CKP_Isr_Test_CMPPulse[67]	0
HWD_CKP_Isr_Test_CMPPulse[68]	0
HWD_CKP_Isr_Test_CMPPulse[69]	0
HWD_CKP_Isr_Test_CMPPulse[70]	0
HWD_CKP_Isr_Test_CMPPulse[71]	0
HWD_CKP_Isr_Test_CMPPulse[72]	0
HWD_CKP_Isr_Test_CMPPulse[73]	0

Obr. 19: Test detektoru polohy motoru – záznam příznaku impulzu CMP.



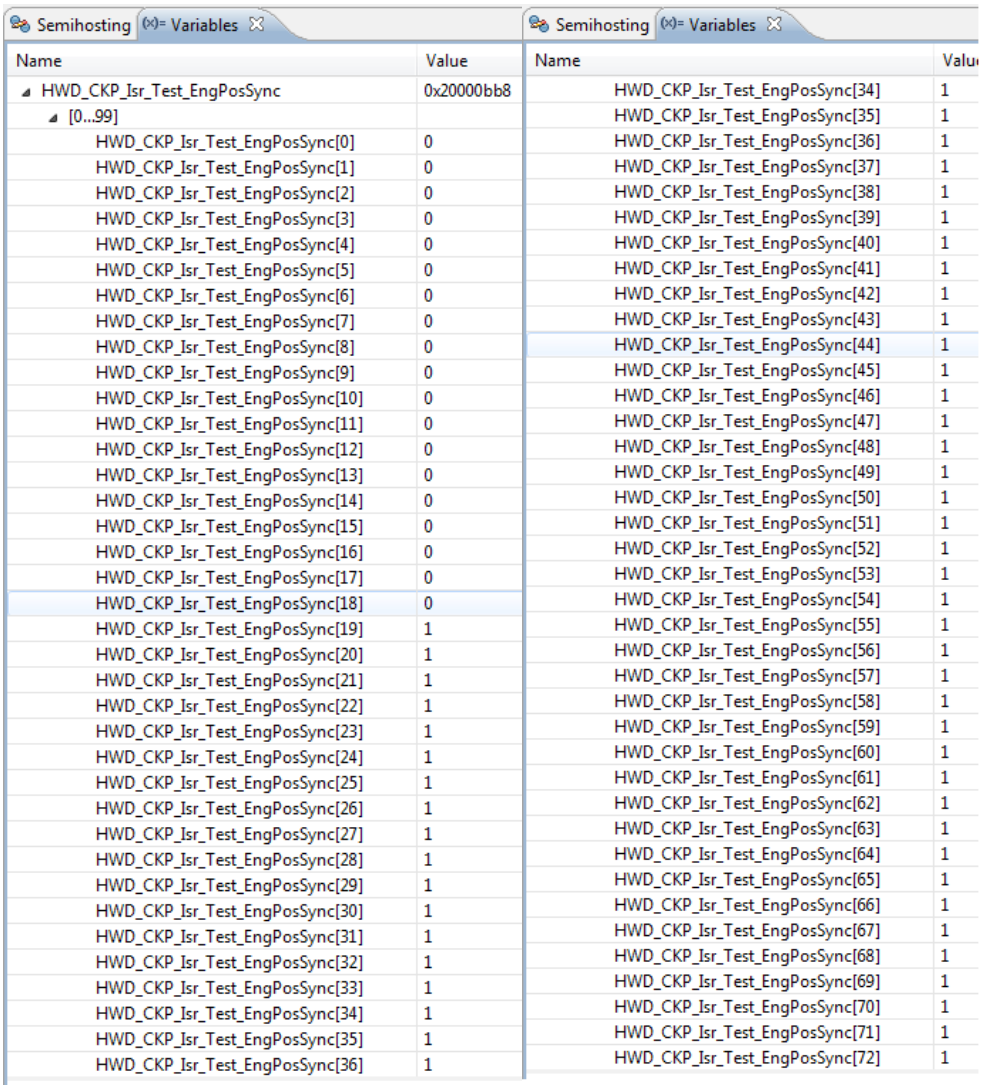
Semihosting (x)= Variables

Name	Value
HWD_CKP_Isr_Test_EngPos	0x20000ab4
[0...99]	
HWD_CKP_Isr_Test_EngPos[0]	0
HWD_CKP_Isr_Test_EngPos[1]	0
HWD_CKP_Isr_Test_EngPos[2]	0
HWD_CKP_Isr_Test_EngPos[3]	0
HWD_CKP_Isr_Test_EngPos[4]	0
HWD_CKP_Isr_Test_EngPos[5]	0
HWD_CKP_Isr_Test_EngPos[6]	0
HWD_CKP_Isr_Test_EngPos[7]	0
HWD_CKP_Isr_Test_EngPos[8]	0
HWD_CKP_Isr_Test_EngPos[9]	0
HWD_CKP_Isr_Test_EngPos[10]	0
HWD_CKP_Isr_Test_EngPos[11]	0
HWD_CKP_Isr_Test_EngPos[12]	0
HWD_CKP_Isr_Test_EngPos[13]	0
HWD_CKP_Isr_Test_EngPos[14]	0
HWD_CKP_Isr_Test_EngPos[15]	0
HWD_CKP_Isr_Test_EngPos[16]	0
HWD_CKP_Isr_Test_EngPos[17]	0
HWD_CKP_Isr_Test_EngPos[18]	0
HWD_CKP_Isr_Test_EngPos[19]	1
HWD_CKP_Isr_Test_EngPos[20]	2
HWD_CKP_Isr_Test_EngPos[21]	3
HWD_CKP_Isr_Test_EngPos[22]	4
HWD_CKP_Isr_Test_EngPos[23]	5
HWD_CKP_Isr_Test_EngPos[24]	6
HWD_CKP_Isr_Test_EngPos[25]	7
HWD_CKP_Isr_Test_EngPos[26]	8
HWD_CKP_Isr_Test_EngPos[27]	9
HWD_CKP_Isr_Test_EngPos[28]	10
HWD_CKP_Isr_Test_EngPos[29]	11
HWD_CKP_Isr_Test_EngPos[30]	12
HWD_CKP_Isr_Test_EngPos[31]	13
HWD_CKP_Isr_Test_EngPos[32]	14
HWD_CKP_Isr_Test_EngPos[33]	15
HWD_CKP_Isr_Test_EngPos[34]	16
HWD_CKP_Isr_Test_EngPos[35]	17
HWD_CKP_Isr_Test_EngPos[36]	18

Semihosting (x)= Variables

Name	Valu
HWD_CKP_Isr_Test_EngPos[35]	17
HWD_CKP_Isr_Test_EngPos[36]	18
HWD_CKP_Isr_Test_EngPos[37]	19
HWD_CKP_Isr_Test_EngPos[38]	20
HWD_CKP_Isr_Test_EngPos[39]	21
HWD_CKP_Isr_Test_EngPos[40]	22
HWD_CKP_Isr_Test_EngPos[41]	23
HWD_CKP_Isr_Test_EngPos[42]	24
HWD_CKP_Isr_Test_EngPos[43]	25
HWD_CKP_Isr_Test_EngPos[44]	26
HWD_CKP_Isr_Test_EngPos[45]	27
HWD_CKP_Isr_Test_EngPos[46]	28
HWD_CKP_Isr_Test_EngPos[47]	29
HWD_CKP_Isr_Test_EngPos[48]	30
HWD_CKP_Isr_Test_EngPos[49]	31
HWD_CKP_Isr_Test_EngPos[50]	32
HWD_CKP_Isr_Test_EngPos[51]	33
HWD_CKP_Isr_Test_EngPos[52]	34
HWD_CKP_Isr_Test_EngPos[53]	35
HWD_CKP_Isr_Test_EngPos[54]	36
HWD_CKP_Isr_Test_EngPos[55]	37
HWD_CKP_Isr_Test_EngPos[56]	38
HWD_CKP_Isr_Test_EngPos[57]	39
HWD_CKP_Isr_Test_EngPos[58]	40
HWD_CKP_Isr_Test_EngPos[59]	41
HWD_CKP_Isr_Test_EngPos[60]	42
HWD_CKP_Isr_Test_EngPos[61]	43
HWD_CKP_Isr_Test_EngPos[62]	44
HWD_CKP_Isr_Test_EngPos[63]	1
HWD_CKP_Isr_Test_EngPos[64]	2
HWD_CKP_Isr_Test_EngPos[65]	3
HWD_CKP_Isr_Test_EngPos[66]	4
HWD_CKP_Isr_Test_EngPos[67]	5
HWD_CKP_Isr_Test_EngPos[68]	6
HWD_CKP_Isr_Test_EngPos[69]	7
HWD_CKP_Isr_Test_EngPos[70]	8
HWD_CKP_Isr_Test_EngPos[71]	9
HWD_CKP_Isr_Test_EngPos[72]	10
HWD_CKP_Isr_Test_EngPos[73]	11

Obr. 20: Test detektoru polohy motoru – záznam aktuální polohy motoru.



Semihosting | Variables

Name	Value
HWD_CKP_Isr_Test_EngPosSync	0x20000bb8
[0...99]	
HWD_CKP_Isr_Test_EngPosSync[0]	0
HWD_CKP_Isr_Test_EngPosSync[1]	0
HWD_CKP_Isr_Test_EngPosSync[2]	0
HWD_CKP_Isr_Test_EngPosSync[3]	0
HWD_CKP_Isr_Test_EngPosSync[4]	0
HWD_CKP_Isr_Test_EngPosSync[5]	0
HWD_CKP_Isr_Test_EngPosSync[6]	0
HWD_CKP_Isr_Test_EngPosSync[7]	0
HWD_CKP_Isr_Test_EngPosSync[8]	0
HWD_CKP_Isr_Test_EngPosSync[9]	0
HWD_CKP_Isr_Test_EngPosSync[10]	0
HWD_CKP_Isr_Test_EngPosSync[11]	0
HWD_CKP_Isr_Test_EngPosSync[12]	0
HWD_CKP_Isr_Test_EngPosSync[13]	0
HWD_CKP_Isr_Test_EngPosSync[14]	0
HWD_CKP_Isr_Test_EngPosSync[15]	0
HWD_CKP_Isr_Test_EngPosSync[16]	0
HWD_CKP_Isr_Test_EngPosSync[17]	0
HWD_CKP_Isr_Test_EngPosSync[18]	0
HWD_CKP_Isr_Test_EngPosSync[19]	1
HWD_CKP_Isr_Test_EngPosSync[20]	1
HWD_CKP_Isr_Test_EngPosSync[21]	1
HWD_CKP_Isr_Test_EngPosSync[22]	1
HWD_CKP_Isr_Test_EngPosSync[23]	1
HWD_CKP_Isr_Test_EngPosSync[24]	1
HWD_CKP_Isr_Test_EngPosSync[25]	1
HWD_CKP_Isr_Test_EngPosSync[26]	1
HWD_CKP_Isr_Test_EngPosSync[27]	1
HWD_CKP_Isr_Test_EngPosSync[28]	1
HWD_CKP_Isr_Test_EngPosSync[29]	1
HWD_CKP_Isr_Test_EngPosSync[30]	1
HWD_CKP_Isr_Test_EngPosSync[31]	1
HWD_CKP_Isr_Test_EngPosSync[32]	1
HWD_CKP_Isr_Test_EngPosSync[33]	1
HWD_CKP_Isr_Test_EngPosSync[34]	1
HWD_CKP_Isr_Test_EngPosSync[35]	1
HWD_CKP_Isr_Test_EngPosSync[36]	1

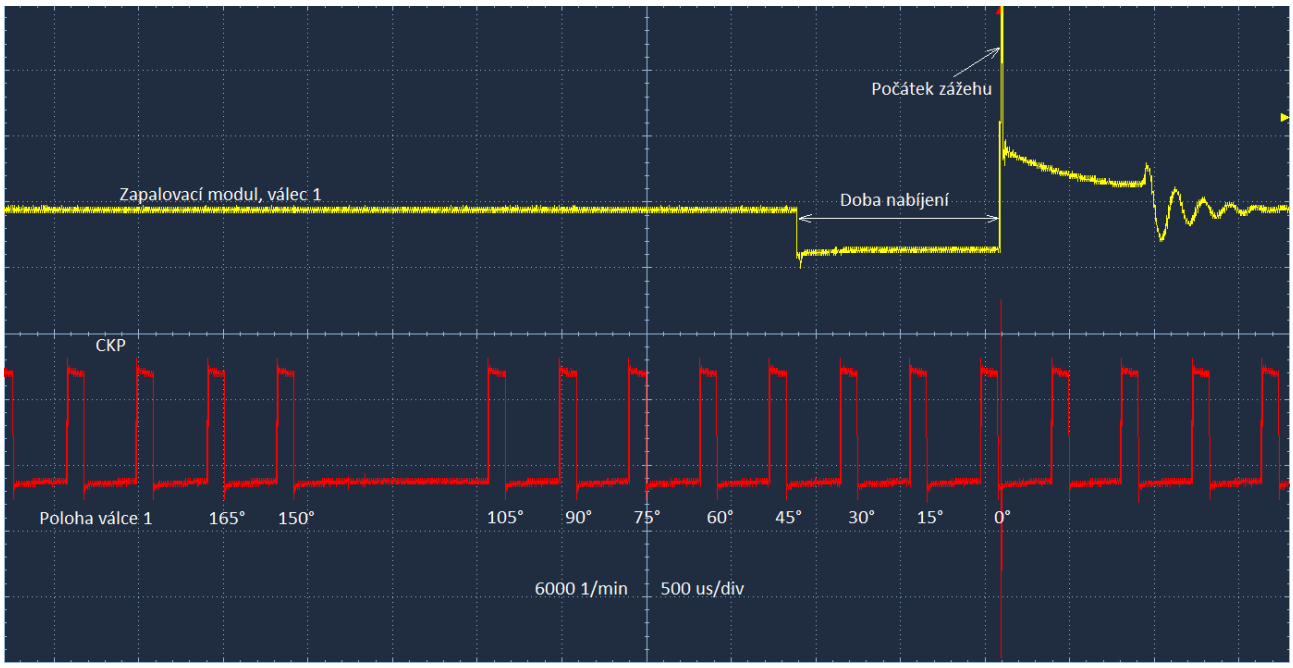
Semihosting | Variables

Name	Valu
HWD_CKP_Isr_Test_EngPosSync[34]	1
HWD_CKP_Isr_Test_EngPosSync[35]	1
HWD_CKP_Isr_Test_EngPosSync[36]	1
HWD_CKP_Isr_Test_EngPosSync[37]	1
HWD_CKP_Isr_Test_EngPosSync[38]	1
HWD_CKP_Isr_Test_EngPosSync[39]	1
HWD_CKP_Isr_Test_EngPosSync[40]	1
HWD_CKP_Isr_Test_EngPosSync[41]	1
HWD_CKP_Isr_Test_EngPosSync[42]	1
HWD_CKP_Isr_Test_EngPosSync[43]	1
HWD_CKP_Isr_Test_EngPosSync[44]	1
HWD_CKP_Isr_Test_EngPosSync[45]	1
HWD_CKP_Isr_Test_EngPosSync[46]	1
HWD_CKP_Isr_Test_EngPosSync[47]	1
HWD_CKP_Isr_Test_EngPosSync[48]	1
HWD_CKP_Isr_Test_EngPosSync[49]	1
HWD_CKP_Isr_Test_EngPosSync[50]	1
HWD_CKP_Isr_Test_EngPosSync[51]	1
HWD_CKP_Isr_Test_EngPosSync[52]	1
HWD_CKP_Isr_Test_EngPosSync[53]	1
HWD_CKP_Isr_Test_EngPosSync[54]	1
HWD_CKP_Isr_Test_EngPosSync[55]	1
HWD_CKP_Isr_Test_EngPosSync[56]	1
HWD_CKP_Isr_Test_EngPosSync[57]	1
HWD_CKP_Isr_Test_EngPosSync[58]	1
HWD_CKP_Isr_Test_EngPosSync[59]	1
HWD_CKP_Isr_Test_EngPosSync[60]	1
HWD_CKP_Isr_Test_EngPosSync[61]	1
HWD_CKP_Isr_Test_EngPosSync[62]	1
HWD_CKP_Isr_Test_EngPosSync[63]	1
HWD_CKP_Isr_Test_EngPosSync[64]	1
HWD_CKP_Isr_Test_EngPosSync[65]	1
HWD_CKP_Isr_Test_EngPosSync[66]	1
HWD_CKP_Isr_Test_EngPosSync[67]	1
HWD_CKP_Isr_Test_EngPosSync[68]	1
HWD_CKP_Isr_Test_EngPosSync[69]	1
HWD_CKP_Isr_Test_EngPosSync[70]	1
HWD_CKP_Isr_Test_EngPosSync[71]	1
HWD_CKP_Isr_Test_EngPosSync[72]	1

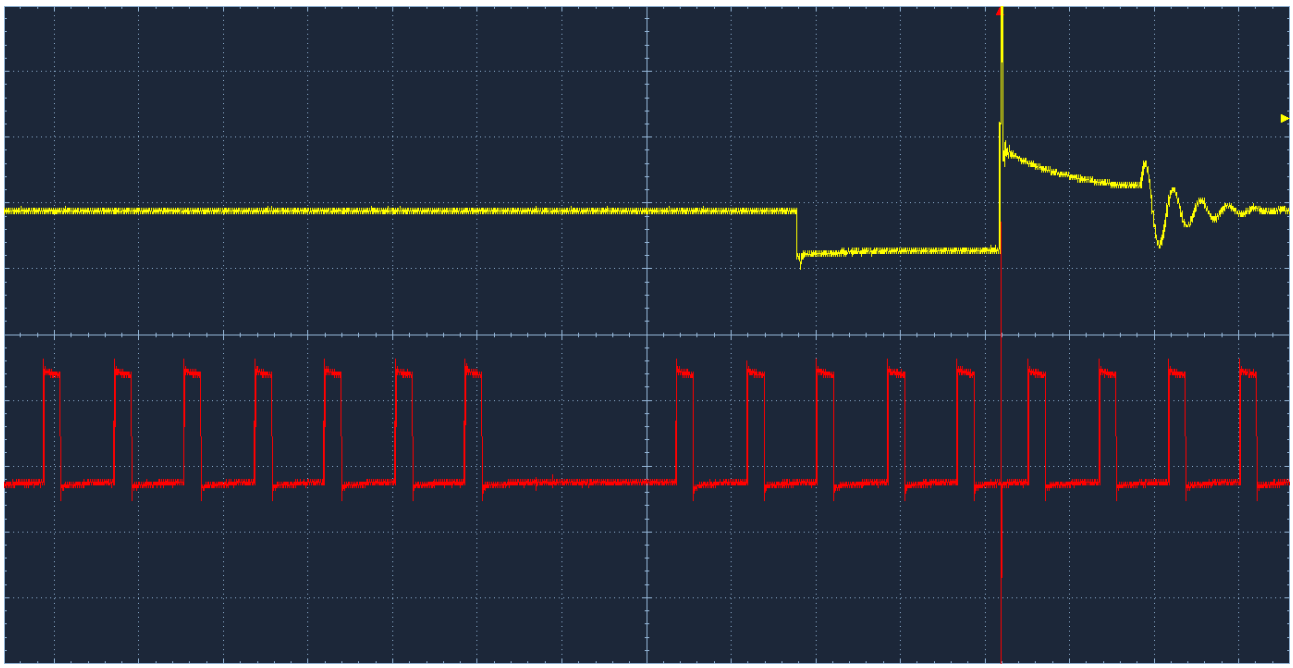
Obr. 21: Test detektoru polohy motoru – záznam příznaku synchronizace.

11.2 Test ovladačů zapalovacích modulů

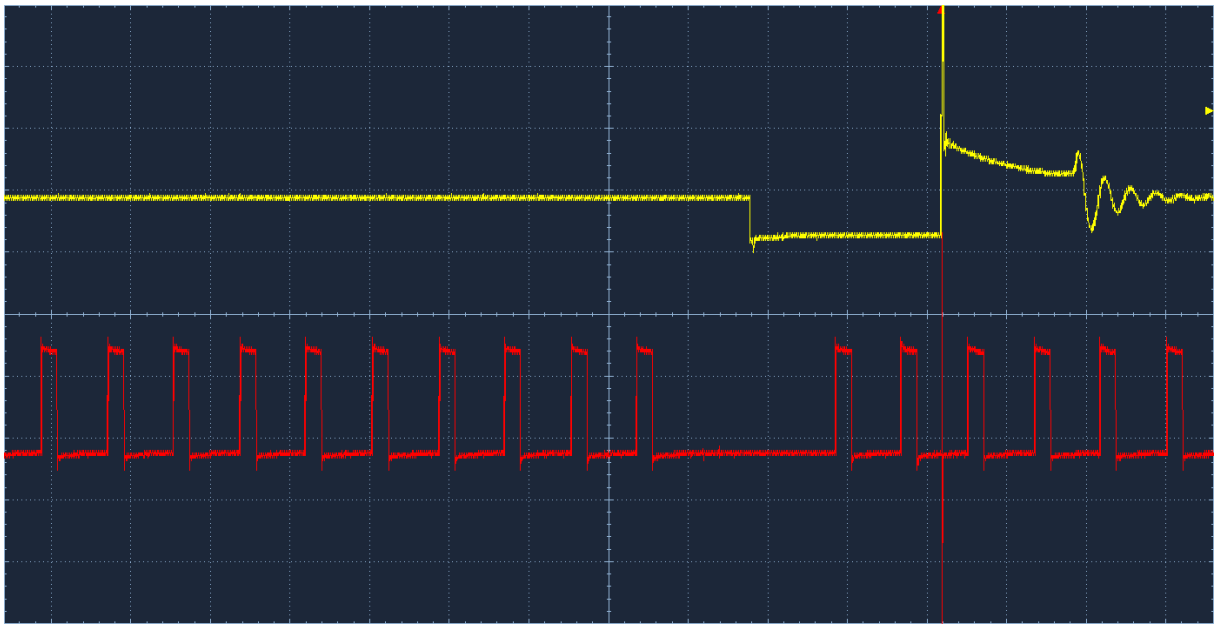
Ovladače zapalovacích modulů byly testovány použitím osciloskopu – sledováním napětí výkonového přívodu k zapalovacímu modulu (žlutý kanál), a polohy klikového hřídele (červený kanál).



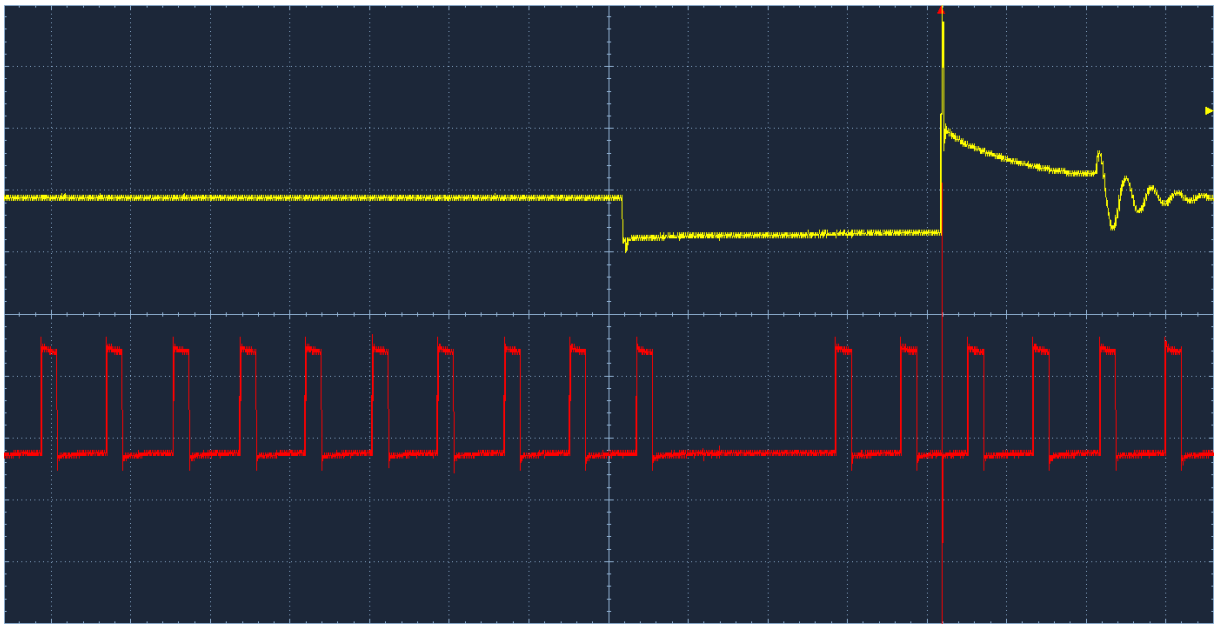
Obr. 22: Test ovladače zapalovacích modulů, úhel zážehu 0°, doba nabíjení 1,2 ms.



Obr. 23: Test ovladače zapalovacích modulů, úhel zážehu 40°, doba nabíjení 1,2 ms.



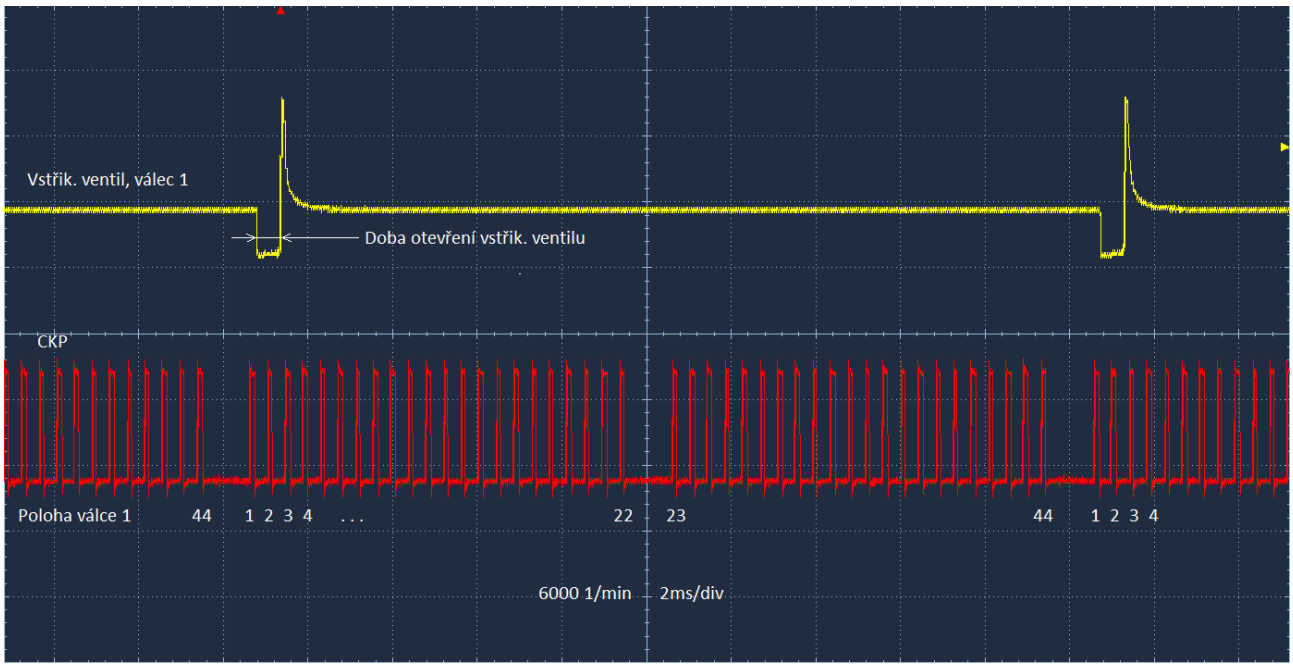
Obr. 24: Test ovladače zapalovacích modulů, úhel zážehu 85°, doba nabíjení 1,2 ms.



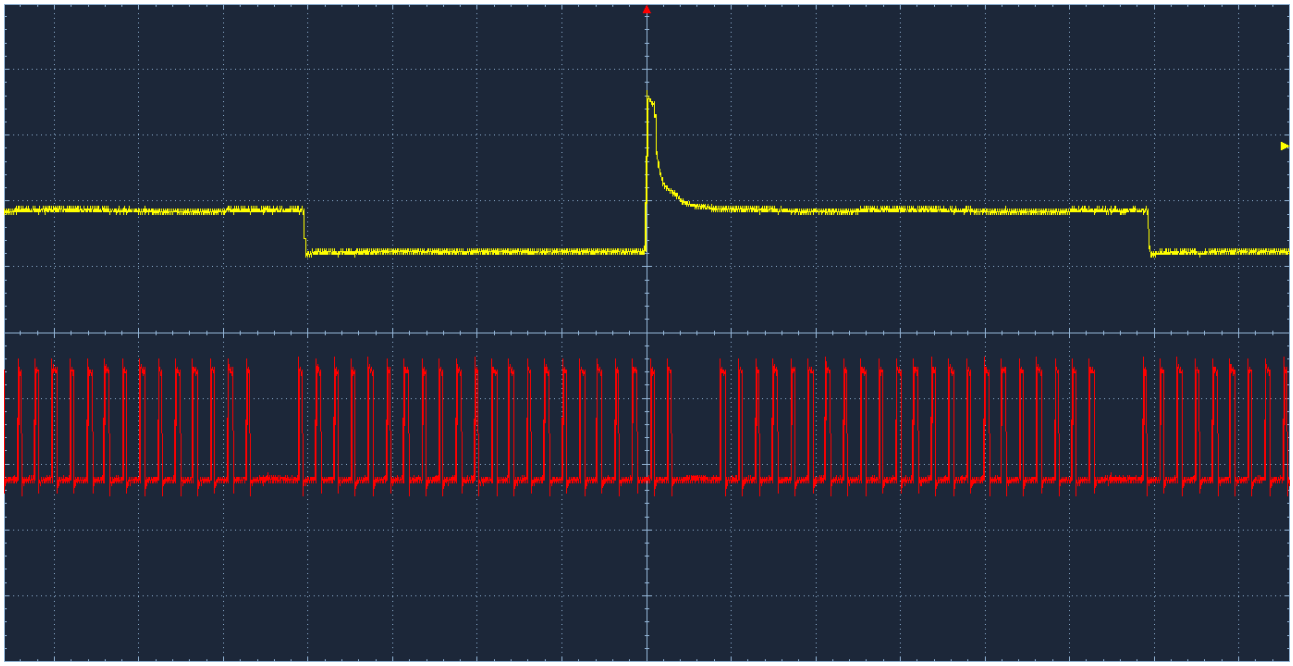
Obr. 25: Test ovladače zapalovacích modulů, úhel zážehu 85°, doba nabíjení 4 ms.

11.3 Test ovladačů vstřikovacích ventilů

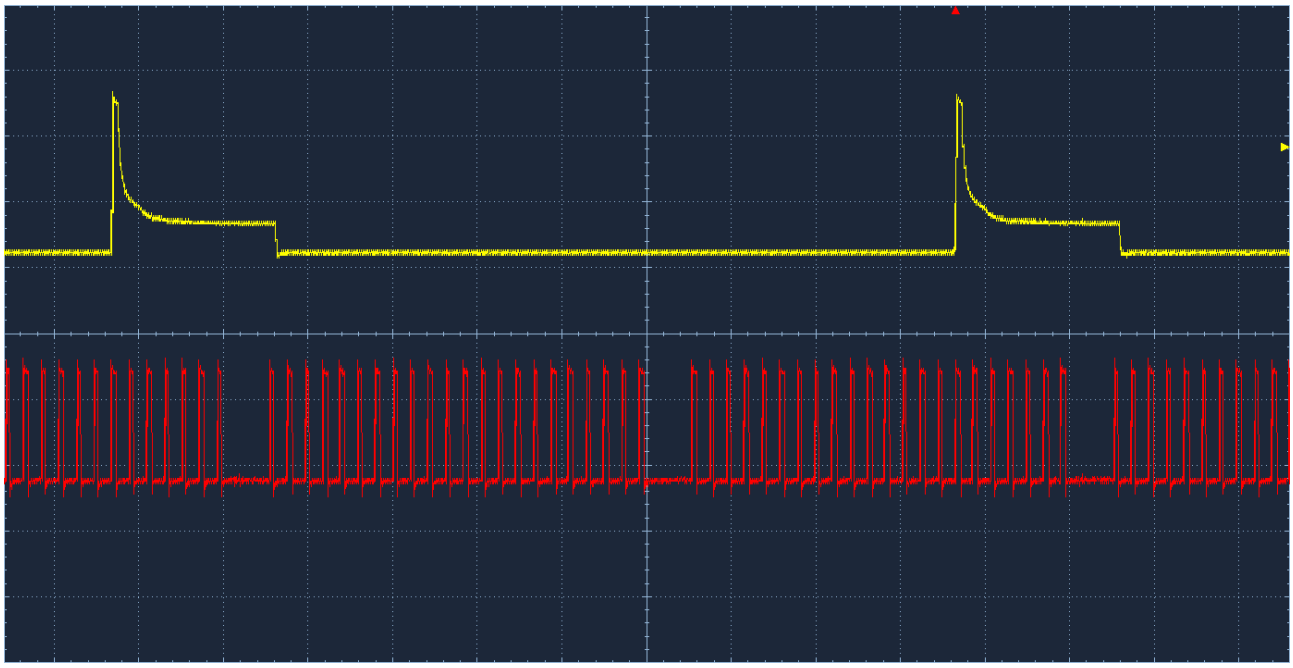
Ovladače zapalovacích modulů byly testovány použitím osciloskopu – sledováním napětí výkonového přívodu ke vstřikovacímu ventilu (žlutý kanál), a polohy klikového hřídele (červený kanál).



Obr. 26: Test ovladačů vstřikovacích ventilů, doba otevření 0,5 ms.



Obr. 27: Test ovladačů vstřikovacích ventilů, doba otevření 8 ms.



Obr. 28: Test ovladačů vstřikovacích ventilů, doba otevření 16 ms.

11.4 Test modulu úpravy analogových signálů

Test byl proveden přiložením známých napětí na analogové vstupy ECM a následným přečtením pole výsledků převodů a filtrace pomocí debuggeru.

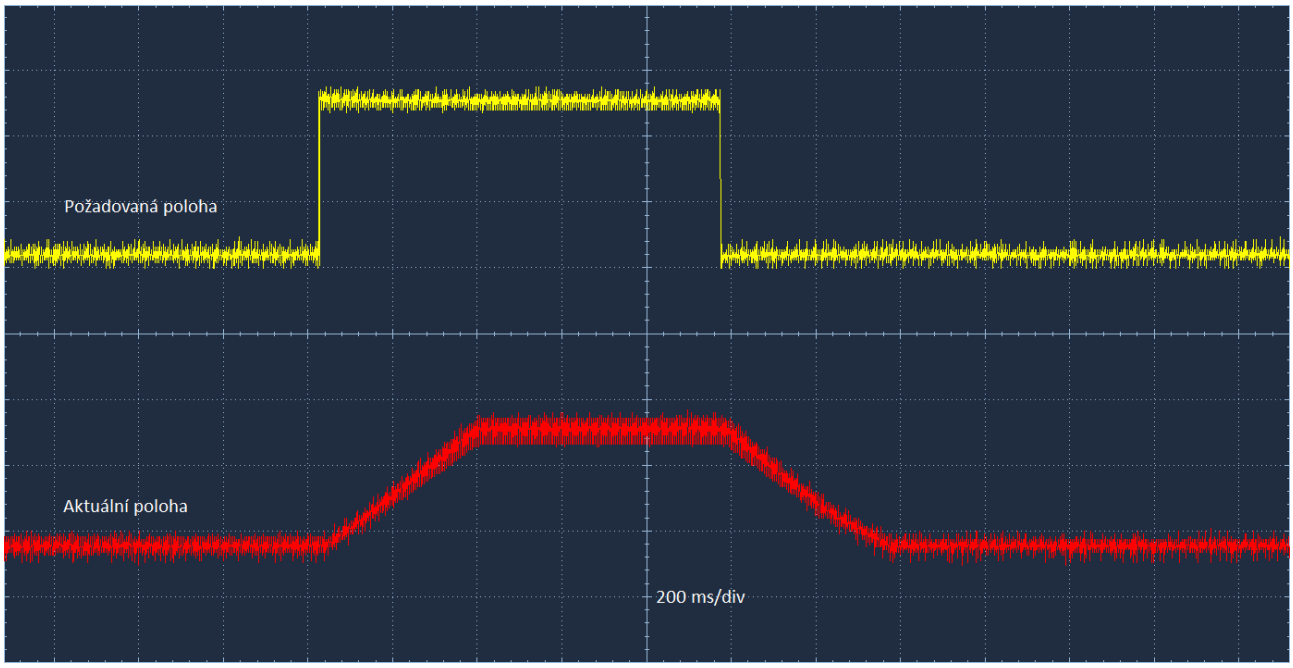
Semihosting | (x)= Variables

Name	Value
HWD_ADCDrv_ADC2BUF	529
◢ HWD_ADCDrv_ADC1RES	0x20001208
HWD_ADCDrv_ADC1RES[0]	1190
HWD_ADCDrv_ADC1RES[1]	1186
HWD_ADCDrv_ADC1RES[2]	3931
HWD_ADCDrv_ADC1RES[3]	3006
HWD_ADCDrv_ADC1RES[4]	615
HWD_ADCDrv_ADC1RES[5]	4095
HWD_ADCDrv_ADC1RES[6]	422
HWD_ADCDrv_ADC1RES[7]	1836
HWD_ADCDrv_ADC1RES[8]	0

Obr. 29: Test modulu zpracování analogových signálů.

11.5 Test regulátoru polohy sekundárních škrtících klapek

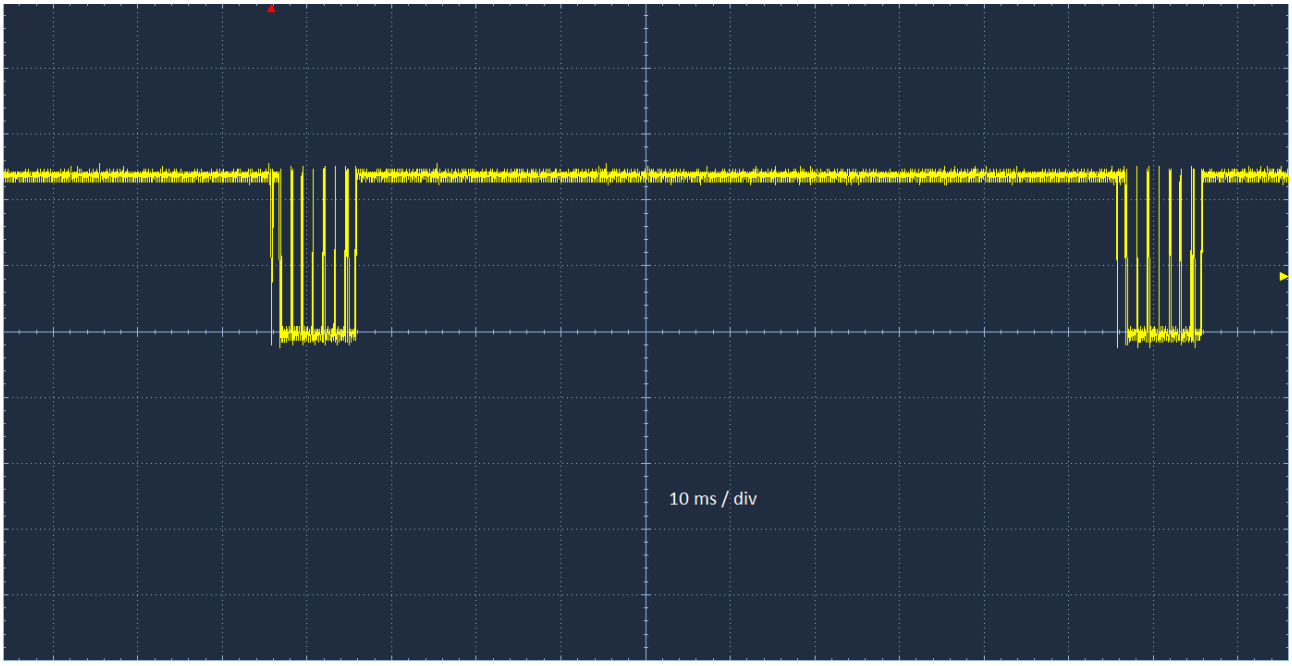
Základní test byl proveden změřením odezvy na jednotkové skoky 0° – 90° – 0° úhlu otevření.



Obr. 30: Test regulátoru polohy sekundárních škrtících klapek.

11.6 Test komunikace s palubním přístrojem

Základní test byl proveden měřením signálu na sériové datové lince osciloskopem.

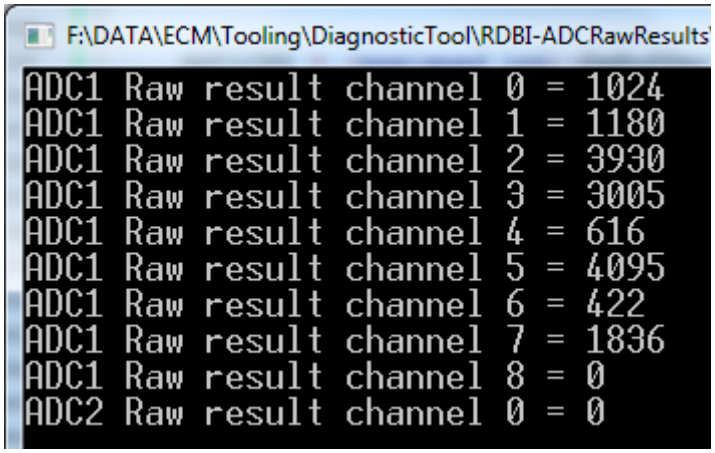


Obr. 31: Test komunikace s palubním přístrojem.

Na obr. 31 jsou patrné dva 8-bajtové pakety, vzdálené od sebe 100 ms. Další testy proběhly již přímo na vozidle.

11.7 Test komunikace s diagnostickým přístrojem

Diagnostický přístroj byl realizován jako konzolový program pro počítač PC, využívající sběrnici USB a převodník USB->UART. V budoucnosti autor plánuje zhotovit diagnostický program s grafickým uživatelským rozhraním.



Obr. 32: Test komunikace ECM s diagnostickým zařízením – čtení výsledků A/D převodů.

11.8 Test nároků softwaru nízké vrstvy na výpočetní čas

Během návrhu byly, z důvodu zajištění nejvyšší přesnosti řízení, implementovány ovladače zapalovacích modulů do procedury přerušení. Nyní je nutné ověřit, že tato procedura přerušení příliš nezatíží mikrokontrolér a nezablokuje jej pro vykonávání ostatních operací. Pro měření výpočetního času procedury byl na její počátek přidán příkaz k nastavení volného GPIO pinu, na konec procedury příkaz k vynulování pinu. Doba výpočtu poté byla měřena osciloskopem, měření ukázalo nejdelší dobu běhu přibližně 15 µs. Při zatížení ECM na hranici projektovaného rozsahu otáček motoru, tato procedura poběží každých přibližně 166 µs. Zatížení procesoru tedy vychází na 9 %. S tímto zatížením je třeba počítat při návrhu softwaru vyšší vrstvy, není však limitující.

12 Závěr

Použití výkonného mikrokontroléru, vybaveného pokročilými periferiemi, umožnilo zvolit odlišný přístup k návrhu softwaru nízké vrstvy pro řídící systém benzinového motoru. Nejnáročnější procesy na reálný čas, zejména měření času pro detekci polohy klikového hřídele, ovládání zapalovacích a vstřikovacích jednotek, bylo implementováno přímo na hardwaru mikrokontroléru. Tato implementace významně přispěla k přesnosti řízení v reálném čase a ušetřila výpočetní výkon jádra mikrokontroléru. Konvenční řídící systémy používají k řízení synchronních, událostmi řízených procesů, přerušení. Pro čtyřválcový motor je potom nutno obsluhovat minimálně 10 procedur přerušení (řízení zapalovacích a vstřikovacích jednotek, detekce signálů snímačů CKP a CMP). Jelikož požadavky na vykonání přerušení mohou nastávat paralelně, je nutno tato přerušení vhodně prioritizovat. Vykonány poté mohou být nejprve procesy, které jsou na reálný čas nejnáročnější, poté dochází k vykonání ostatních. Řešení, navržené v této práci, umožňuje tyto procesy vykonávat paralelně. Jsou zde použita pouze dvě přerušení – pro obsluhu snímačů CKP a CMP. Tato přerušení však nemohou nastat zároveň.

Výkonné a pružné A/D převodníky v součinnosti s DMA umožnily zajistit dostatek měřených vzorků pro provedení kvalitní filtrace. Tímto bylo dosaženo vysoké přesnosti měření a odolnosti vůči rušení, které je obecným problémem implementací řídících systémů spalovacích motorů.

13 Seznam použité literatury

[1] STMicroelectronics: RM0090, Reference manual for STM32F4 microcontroller, 2016.

[2] Suzuki Motor Corporation: Suzuki GSR 600 Service manual, Part No. 99500-36160-01E, 2005.

[3] DOSEDLA, Martin: Simulátor signálů snímačů polohy klikového a vačkového hřídele pro testovací a kalibrační účely, semestrální projekt, 2018.

[4] BANISH, Greg: *Designing and tuning high performance fuel injection systems.* CarTech, North Branch, USA, 2009. ISBN 978-1-932494-90-7.

[5] BANISH, Greg: *Engine management advanced tuning.* CarTech, North Branch, USA, 2007. ISBN 978-1-932494-42-6.