FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

Zadání bakalářské práce

Název:	Systém pro emulaci akceleračního pedálu osobního automobilu
Student:	Jakub Mareček
Vedoucí:	Ing. Pavel Kubalík, Ph.D.
Studijní program:	Informatika
Obor / specializace:	Počítačové inženýrství
Katedra:	Katedra číslicového návrhu
Platnost zadání:	do konce letního semestru 2023/2024

Pokyny pro vypracování

1) Prozkoumejte existující řešení.
2) Analyzujte protokol SENT.
3) Analyzujte možnosti generování kódů emulujících funkci akceleračního pedálu a navrhněte řešení s využitím platformy ESP32.
4) Proveďte analýzu komunikace pedálu z koncernu VW.
5) Navrhněte zařízení pro replikaci signálů akceleračního pedálu tak, aby zařízení bylo schopno komunikovat s reálnou řídící jednotkou.
6) Pro komunikaci pedálu s řídicí jednotkou využijte protokol SENT.
7) Zařízení musí být ovladatelné přes CAN sběrnici i fyzickými ovladači.
8) Navržené řešení zrealizujte a řádně otestujte.

Elektronicky schválil/a prof. Ing. Hana Kubátová, CSc. dne 22. září 2022 v Praze.

Bakalárska práca

SYSTÉM PRE EMULÁCIU AKCELERAČNÉHO PEDÁLU OSOBNÉHO AUTOMOBILU

Jakub Mareček

Fakulta informačních technologií
Katedra číslicového návrhu
Vedúci: Ing. Pavel Kubalík, Ph.D.
16. februára 2023

Obsah

Zoznam obrázkov

Zoznam tabuliek

Chcel by som sa poďakovať môjmu vedúcemu práce Ing. Pavlovi Kubalíkovi, Ph.D., za pomoc pri vypracovaní tejto práce. Týmto ďakujem spoločnosti Digiteq Automotive s. r. o za zadanie témy a poskytnutiu prostriedkov na jej realizáciu. Ďakujem Ing. Jindřichovi Novákovi za podporu pri štúdiu a v práci. V neposlednej rade som vďačný mojej rodine, za pomoc a oporu počas štúdia na vysokej škole.

Vyhlásenie

Prehlasujem, že som predloženú prácu vypracoval samostatne a že som uviedol všetky použité informačné zdroje v súlade s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací. Beriem na vedomie, že sa na moju prácu vzťahujú práva a povinnosti vyplývajúce zo zákona č. 121/2000 Sb., autorského zákona, vo znení neskorších predpisov, menovite skutočnosť, že České vysoké učenie technické v Prahe má právo na uzavretie licenčnej zmluvy o použití tejto práce ako školského diela podľa § 60 odst. 1 citovaného zákona.

V Praze dne 16. februára 2023 .....

Abstrakt

Bakalárska práce sa zaoberá využitím protokolu SENT, možnosťami jeho generovania (emulovania) a posielania protokolárnych správ na základe užívateľského vstupu podľa normy SAE J2716_201604.

Teoretická (literárna) časť je zameraná na popis protokolu, následne je vykonaná rešerš existujúcich komerčných riešení a ich ohodnotenie na základe požadovaných vlastností. Uvažované riešenia sa týkajú problému generovania protokolu SENT a možnosti emulovania akceleračného pedálu.

Prvá časť praktickej časti je zameraná na časovú analýzu protokolu na koncovom zariadení akceleračného pedálu koncernu Volkswagen.

V druhej časti je popísaný vývoj vlastného riešenia založeného na platforme Arduino a ESP-32. Vyvinuté zariadenie generuje signály na dvoch fyzických výstupoch a poskytuje komunikačné rozhranie CAN a SPI. Výstup zbernice CAN je použitý pre vzdialené ovládanie. SPI slúži pre možnosť navýšenia počtu výstupov protokolu po pripojení ďalšieho hardvéru.

Výsledné riešenie je otestované krajnými hodnotami stanovenými normou.

Klíčová slova emulátor protokolu, vstavateľný systém, Arduino, Esp-32, SENT protokol, automotive, Volkswagen, embedded C, FreeRTOS

Abstract

The bachelor thesis is about means of use of the SENT protocol, possibilities of its generating (emulating) and sending protocolar messages based on user input by norm SAE J2716_201604.

Theoretical (literal) part is focused on description of the protocol, next research of existing commercial solutions is made and they are rated based on required attributes. Considered solutions are of mean of generating protocol SENT and possibility of emulating an accelerator pedal.

First part of practical part is about a time analysis of the protocol on the endpoint device of the acceleration pedal from Volkswagen concern.

In second part a development of own solution based on Arduino platform and ESP-32 is described. The developed device is generating signals on two physical outputs and provides an CAN and SPI interface. Output of CAN bus is used for remote access. SPI is used for possible increase of number of protocolar outputs after connecting additional hardware.

The finished solution is tested with limiting values declared by norm.

Keywords protocol emulator, embedded system, Arduino, Esp-32, SENT protocol, automotive, Volskwagen, embdedded C, FreeRTOS

Zoznam skratiek

CAN	Controller Area Network
CSV	Comma-separated Values
FD	Flexible Data-Rate
IDE	Integrated Development Environment
SAE	Society of Automotive Engineer
SENT	Single Edge Nibble Transmission
SPI	Serial Peripheral Interface

Úvod

V súčasnej dobe narastá komplexnosť a počet zariadení použitých v moderných automobiloch na meranie stavu vozidla a jeho okolia. Mechanické ovládanie a meranie veličín postupne nahradili analógové senzory. Neskôr využitie analógových meracích zariadení, pripojených k analógovým zobrazovacím prvkom, nahradila digitalizácia. To umožnilo lepšie spracovanie a presnejšie zobrazenie nameraných dát. Špecifický typ dát a meranej veličiny viedol na rozvoj rôznych komunikačných protokolov.

Akceleračný pedál prešiel vývojom od priameho pripojenia k motoru na meranie polohy pedálu a následného spracovania riadiacou jednotkou. Pri prenose informácie od senzoru bola analógová veličina nahradenou digitálnou vo forme SENT protokolu, ktorý umožnil prenášať väčšie množstvo dát. Rozličné použité senzory a typy protokolov zvyšujú náročnosť na testovanie jednotlivých automobilových komponent. Pre korektné testovanie potrebujú jednotlivé riadiace jednotky dostávať namerané dáta od senzorov. Testovanie jednotky nezahrnuje testovanie periférnych senzorov, preto sa ukazuje ako dostačujúce stimulovať jednotku externým zariadením miesto stimulovania senzoru pre potrebný vstup. Komplexné testovanie zároveň implikuje automatizáciu a vzdialené nastavenie podmienok testu. Zároveň takýto prístup dovoľuje zaniesť chybu do komunikácie, alebo navodiť situáciu, ktorá by bola v reálnom svete ťažko dosiahnuteľná.

Táto práca vznikla, pretože simulovanie akceleračného pedálu má priamočiare mechanické riešenie, ktoré je založené na princípe pohybovania fyzického dielu. To so sebou prináša komplexnosť a malú flexibilitu riadenia vstupu. Zároveň takéto riešenie prináša priestorovú a mechanickú náročnosť na potrebné stimulujúce vybavenie.

Cieľom je nahradiť potrebu stimulácie periférneho senzoru a umožniť tak iba testovanie prijímacej jednotky. Primárne použitie je testovanie vo vývojových centrách, preto je jednou z dôležitých výsledných vlastností flexibilita a škálovateľnosť. Zároveň je cieľom riešiť problém komerčných riešení, ktoré môžu byť závislé na použitom hardvéry, a preto ťažko prenositeľné, ako aj problém closed-source riešení, kedy zákazník nemá kontrolu nad spracovaním dát a úprava pre vývoj prototypu je časovo náročná. Riešením je vytvoriť open-source softvér, ktorý je ľahko prenositeľný použitím open-source platformy Arduino, takže nie je úplne viazaný na konkrétny použitý hardvér, a zároveň je možné ho jednoducho rozšíriť a upraviť pre konkrétnu aplikáciu.

Táto práca sa nezaoberá testovaním konkrétneho akceleračného pedálu koncernu Volkswagen, takže analýza SENT komunikácie je použitá len pre získanie informácií o komunikácií. Zároveň sú implementované iba základné módy komunikácie. Zvyšné údaje sú čerpané z normy definujúcej SENT protokol.

Obsahom práce je analýza SENT protokolu podľa normy a rešerš existujúcich riešení, ktoré umožňujú generovanie SENT protokolu ako takého a porovnanie ich vlastnosti. Nasleduje analýza konkrétnej podoby protokolu použitého v akceleračnom pedáli. Analýza protokolu určuje obmedzenia, ktoré musia byť brané v úvahu pri návrhu riadiaceho softvéru. V ďalšej časti sú porovnané dostupné hardvérové riešenia, ktoré sú kompatibilné s Arduino platformou, a zároveň vyho-

vujú stanoveným obmedzeniam analyzovaným protokolom. Ďalšia kapitola sa zaoberá návrhom riešenia a rozdelením na základné návrhové prvky. Návrh je následne implementovaný s využitím poznatkov z teoretickej analýzy.

Zrealizované riešenie je otestované použitím limitných hodnôt definovaných normou pre SENT komunikáciu, aby bola zaručená správnosť aj pre iné použitia.

Cieľ

Cieľom práce je vytvorenie zariadenia pre replikáciu komunikácie medzi akceleračným pedálom a riadiacou jednotkou. Oproti stávajúcim riešeniam bude zariadenie využívať open-source hardvér a softvér. Pri vývoji bude kladený dôraz na možnosť rozšírenia a možnosť jednoduchej úpravy.

Aby mohlo byť zariadenie použité samostatne, bude obsahovať fyzické ovládače pre nastavenie hodnoty. Pre možnosť použitia zariadenia vo väčšom celku, bude možné hodnotu nastaviť aj vzdialene pomocou CAN zbernice. Aktuálne generovaná hodnota bude zobrazená priamo na obrazovke zariadenia. Zariadenie bude rozšíriteľné na rôzny počet výstupov, za týmto účelom bude poskytovať rozhranie pre pripojenie dodatočného hardvéru potrebného na generovanie.

Postup riešenia bude rozdelený do jednotlivých čiastočných cieľov. Prvým bude analýza SENT protokolu podľa stanovenej normy SAE J2716. Nasledujúcim cieľom bude analýza protokolu z výstupu akceleračného pedálu. Ďalším analýza možných hardvérových prostriedkov, ktoré umožnia generovanie protokolárnych správ. Po analýze dostupných hardvérových prostriedkov bude navrhnutá realizácia, ktorá bude schopná splniť dané požiadavky.

Ďalším krokom bude spojenie použitých prvkov a ich testovanie. Pre účely testovania a analýzu budú výsledné generované signály protokolu SENT porovnané s dátami z reálneho akceleračného pedálu koncernu Volkswagen. Následne budú všetky použité komponenty prepojené na nepájavom kontaktnom poli.

Kapitola 1
Analýza

Akceleračný pedál komunikuje pomocou protokolu SENT, jeho analýza bola nutná pre bližšie skúmanie priebehu komunikácie medzi pedálom a riadiacou jednotkou a vývojom nového univerzálneho riešenia.

1.1 Komunikačný protokol SENT

SENT je skratka pre Single Edge Nibble Transmission. Parametre protokolu sú definované v norme j2716 vytvorená spoločnosťou SAE (Society of Automotive Engineers). Pri výskume bola ako referencia použitá revízia z roku 2016. [1]

Komunikácia sa uskutočňuje výhradne jednosmerne medzi senzorom (vysielačom) a kontrolérom (prijímačom). Napr. pri použití v akceleračnom pedáli medzi senzorom polohy (vysielač) a riadiacou jednotkou (prijímač). Vysielač posiela dáta konštantne, takže každá správa nasleduje po skončení predchádzajúcej. Dáta sú zakódované ako časové úseky medzi zostupnými hranami. Konkrétne parametre kódovania dát sa môžu líšiť v závislosti na aplikácií komunikácie. [1, str.5]

Logická hodnota na zbernici je určená napätím: [1, str.29]

- logická 0 $\leq$ 0.5 V (ďalej len log. 0),
- logická 1 $\geq$ 4.1 V (ďalej len log. 1),
- nábežná hrana 3.8–1.1 V,
- dobežná hrana 1.1–3.8 V.

Jednotka času sa označuje ako tick. Norma odporúča trvanie v rozsahu 3 μs až 90 μs. Môže byť ale použitý iný časový alebo napäťový interval pre účely daného merania. Odchýlka trvania je stanovená na ± 20 % od zadanej dĺžky. [1, str.10]

1.1.1 Priebeh komunikácie

Rámec jednej správy sa dá rozdeliť do niekoľkých kľúčových častí na pulzy.

Obr. 1.1 Jeden rámec správy SENT s voliteľným pauzovacím pulzom

Komunikácia sa začína synchronizačným pulzom, ktorý následne určuje parametre komunikácie. To umožňuje prijímacej strane korekciu hodinového cyklu pre dané spojenie. Zároveň je možné komunikovať s viacerými protistranami, ktoré nemajú vzájomne synchronizované hodinové cykly. Dĺžka synchronizačného pulzu (sync) je 56 tick, z ktorých je 5 tick zbernica v log. 0 a zbytok v log. 1. [1, str.12]

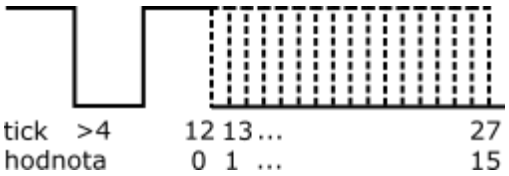
Výsledná hodnota jedného tick sa dá v zjednodušenej podobe odvodiť podľa vzťahu 1.1. Daná hodnota potom určuje reálnu dobu trvania jedného tick.

$$|tick| = \frac{sync}{56} \tag{1.1}$$

Po synchronizačnom pulze nasleduje status. Slúži na prenos bitov, ktoré môžu niesť dodatočné informácie ako číslo dielu alebo chybové kódy. Dva bity môžu byť použité na prenos dát v pomalom kanáli. [1, str.13]

Jednotlivé dáta sú posielané po 4 bitoch vo forme tzv. nibble. Jeden nibble je zložený zo 4 bitov, má predpísaný čas, počet tick, počas ktorého je zbernica v log. 0 a potom v log. 1. Norma stanovuje celkovú dĺžku takéhoto prefixu na 12 tick, z ktorých aspoň 5 tvorí log. 0. Zbytok nibble je doplnený o dáta. Počas vysielania dátovej časti sa logická hodnota na zbernici nezmení.

To znamená, že nibble kódujúci hodnotu 0 trvá 12+0 tick, hodnotu 1 trvá 12 + 1 tick, ..., hodnota 15 trvá 12 + 15 tick. [1, str.12]



Obr. 1.2 Variabilná dĺžka jedného nibble v závislosti na hodnote

Norma určuje algoritmus 1.2 na odhadnutie hodnoty jedného nibble ako výpočet pomeru (Rcal) medzi nameranou dĺžkou synchronizačného pulzu a nominálnou hodnotou s dĺžkou tick (Ticknom) a následný výpočet hodnoty dát N zaokrúhlenej na celé číslo. [1, str.18]

$$\begin{aligned} Rcal &= \frac{nameraná\ dĺžka\ synchronizačného\ pulzu}{56 * Ticknom} \\ Hodnota\ dát\ N &= Zaokrúhlenie\left(\frac{\frac{nameraná\ dĺžka\ pulzu\ N}{Rcal-12*Ticknom}}{Ticknom}\right) \end{aligned} \tag{1.2}$$

Za dátovou časťou nasleduje CRC. To je definované polynómom $x^4+x^3+x^2+1$ so počiatočným kľúčom 5. Aktuálna hodnota súčtu sa vzťahuje iba na dátovú časť rámca. [1, str.21]

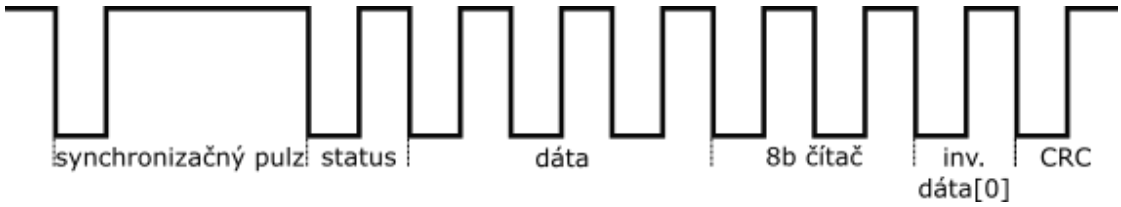
Pretože dátový rámec je závislý na časovom trvaní, rôzne dáta spôsobujú zmenu celkovej dĺžky trvania správy. Voliteľný pauzovací pulz môže byť pridaný na koniec správy, aby bola zaručená konštantná dĺžka správy. Môže mať dĺžku 12–768 tick. [1, str.17]

1.1.2 Formát prenášaných dát

V rámci jednej správy je možné preniesť max. $4 * 6 = 24$ bitov dát. Dátový rámec je možné rozdeliť na dva kanály, napr. 12 bitov a 12 bitov. Pre jednotlivé typy vysielačov norma odporúča určitý typ usporiadania jednotlivých dát v rámci jednej správy. Ďalej boli zhrnuté niektoré ďalšie formáty správ definované normou.

1.1.2.1 Zabezpečená správa

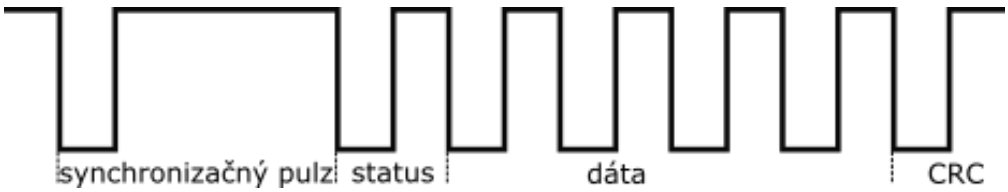
Správa môže byť odosielaná v tzv. zabezpečenom formáte. Odosiela sa iba 12 bitov dát. Za dátami nasleduje 8bitový čítač a posledný nibble obsahuje invertovanú hodnotu najviac významného dátového nibble. Správa je ukončená kontrolným súčtom. [1, str.110]



Obr. 1.3 Jeden rámec zabezpečenej správy správy

1.1.2.2 Vysoko rýchlostná 12bitová správa

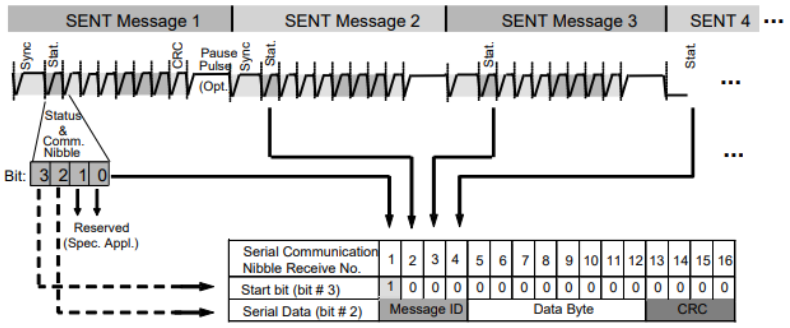
Pri využití rýchleho kanálu (Fast Channel High Speed Format) sa odosiela 12 bitov dát v 4 nibble. Z definície jeden nibble obsahuje 4 bity dát, preto je najviac významný bit rovný 0. Okrem menšieho počtu pulzov, správa dodržiava predpísanú štruktúru správy so status a CRC nibble. Pre tento typ správy je stanovená tolerancia odchýlky hodinového cyklu ± 10 % a nominálna hodnota tick 2.67 μs. [1, str. 109–110]



Obr. 1.4 Jeden rámec vysoko rýchlostnej 12bitovej správy

1.1.2.3 Pomalý kanál

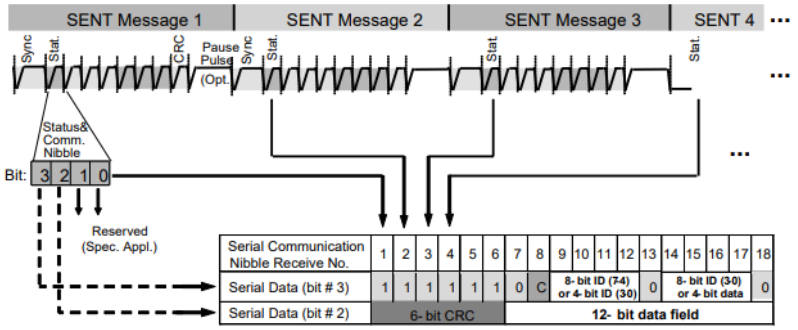
Správa s pomalým kanálom (Short Serial Message) je rozdelená na tzv. pomalý a rýchly kanál. Pomalý kanál slúži najmä pre prenos dát, ktoré sú monotónne (teplota, identifikačné číslo, ...). Dáta pomalého kanálu tvoria 3. a 4. bit status nibble. Začiatok správy je indikovaný hodnotou 1 v treťom bite a po zbytok správy je tretí bit rovný nule a dáta sa prenášajú v štvrtom bite. Takáto správa môže obsahovať 4bitový identifikátor, 8 bitov dát a 4 bity kontrolného súčtu.



Obr. 1.5 Časť správy v pomalom kanáli [1, obr. 5.2.4.1–1, str.13]

Identifikátor je unikátny pre každý produkt a definované od výrobcu. Tento spôsob umožňuje preniesť jednu pomalú správu a zároveň 16 rýchlych. [1, str.13]

Druhým spôsobom je použitie rozšíreného formátu (Enhanced Serial Message Format). Pomalý kanál obsahuje 12 alebo 16 bitov dát. Pomalá správa je uvedená 6bitovým kontrolným súčtom, počas ktorých je tretí status bit v hodnote 1. Nasledujúce dátové bity sú posielané s tretím status bitom v hodnote 0. Poslanie takejto správy trvá 18 (22) rýchlych správ. [1, str.14]

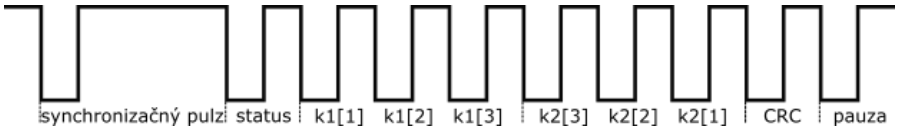


Obr. 1.6 Časť správy v pomalom kanáli v rozšírenom formáte [1, obr. 5.2.4.2–1, str.14]

1.1.2.4 Formát dát pre duálny akceleračný pozičný senzor

Dáta sú vysielané v dvoch 12bitových rýchlych kanáloch. Prvý kanál odosiela dáta od naviac významného po najmenej významný bit v troch nibble. Pri detekcií chyby je bit 0 v status nibble nastavený na 1, inak je nastavený na 0. Druhý kanál vysiela dáta od najmenej významného po najviac významný bit v troch nibble. Detekcia chyby je ako pri prvom kanály signalizovaná v bite 1 v status nibble. Dátový vektor je rozdelený na 4bitové skupiny do jednotlivých nibble, od najviac významného bitu po najmenej významný a potom je poradie pre druhý kanál obráterné.

Dodatočne norma stanovuje, že pri detekcií chyby jedného senzoru, by mali byť príslušné hodnoty nibble nastavené do hodnoty 0xf alebo 0x0. [1, str.38]



Obr. 1.7 Jeden rámec správy z duálneho akceleračného pozičného senzoru. K1, k2 označujú jednotlivé kanály, s naznačenou prioritou dát

1.2 Rešerš

Pred započatím vlastného výskumu bolo potreba preskúmať existujúce riešenia. Pri testovaní riadiacich jednotiek je potrebné stimulovať potrebné vstupy. Periférny senzor, akceleračný pedál, môže byť mechanicky stimulovaný. V takom prípade, je ale aj senzor súčasťou testovanej komponenty. Iným prístupom je generovanie potrebnej protokolárnej komunikácie.

Existujúce riešenia boli porovnávané z hľadiska vhodnosti do začlenenia do prototypového testovania a možností ich ďalšieho prispôsobenia a rozšírenia.

1.2.1 AB Dynamics

Firma AB Dynamics sa zaoberá výrobou systémov pre testovanie v automotive oblasti. Pre stimuláciu pedálov poskytuje riešenie založené na elektrických motoroch. Produkt CBAR umožňuje kombináciu s ďalšími produktami robotického ovládania, na riadenie viacerých prvkov vozidla. Zároveň je umožnený priamy zásah človeka do ovládania, pretože z vozidla nie sú odstránené žiadne súčiastky. [2]

Toto riešenie sa ukazuje ako veľmi jednoduché na implementáciu a rýchle na prispôsobenie širokej škále situácií. Nevýhodou je najmä potreba vyhradiť priestor a kotviace miesta pre ovládacie konštrukcie, tak aby bola dosiahnutá stabilita pohyblivých častí. Zároveň tento spôsob neumožňuje ovplyvnenie protokolárnej komunikácie ako takej.

Obr. 1.8 AB Dynamics CBAR600, *prevzaté z* [2]

1.2.2 Arbitrárne vlnové generátory spoločnosti Keysight

Spoločnosť Keysight ponúka niekoľko modelov generátorov vlnového signálu. Generátory môžu fungovať na princípe funkčného generovania, kedy je výstup modulovaný matematickou funkciou, alebo ako arbitrárne generátory. Arbitrárny režim umožňuje generovanie zložitejšieho priebehu funkcie. [3]

Použitý ovládací softvér BenchVue umožňuje programovanie výstupu podľa dát získaných z osciloskopu. To umožňuje použiť namerané hodnoty pre replikáciu protokolu podľa nameraných dát zo vzorového akceleračného pedálu. [4]

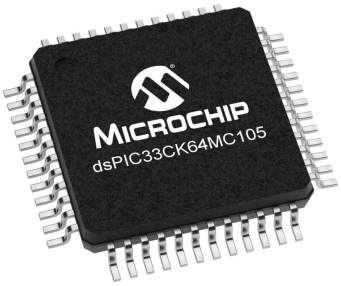
Výhoda riešenia od spoločnosti Keysight je možnosť prehratia nameraných hodnôt z osciloskopu, ktoré umožňuje vysokú presnosť medzi nameranými a požadovanými hodnotami SENT protokolu. Využitie generátora alebo obdobného princípu nie je vhodné, kvôli potrebe externého riadiaceho programu pre programovanie výstupu a vysokej náročnosti na generovanie nových hodnôt.

Obr. 1.9 Keysight M8196A, *prevzaté z* [5]

1.2.3 Rodina procesorov dsPIC33

Procesory od spoločnosti Microchip z rodiny dsPIC 33 obsahujú SENT modul. Modul operuje v módoch synchrónny a asynchrónny vysielač, prijímač. Synchrónny vysielač je riadený programom, kedy vyslanie ďalšej správy je riadené nastaveným príznakom. Asynchrónny mód posiela dáta nepretržito v slučke. Výstup signálov môže byť invertovaný. Popri štandardnej komunikácií poskytuje hardvér podporu výpočtu CRC. Ako prijímač umožňuje detekciu chýb v CRC, chýb rámca (spôsobených nesprávnym časovaním), v prípade chyby modul čaká na nový správny synchronizačný pulz. [6]

Výhodou priamej hardvérovej podpory protokolu v procesore je optimalizácia pre daný typ komunikácie. Hlavnou nevýhodou je priama závislosť na určitom type procesoru, kvôli ktorému je dané riešenie nevyhovujúce pre použitie v tejto práci.



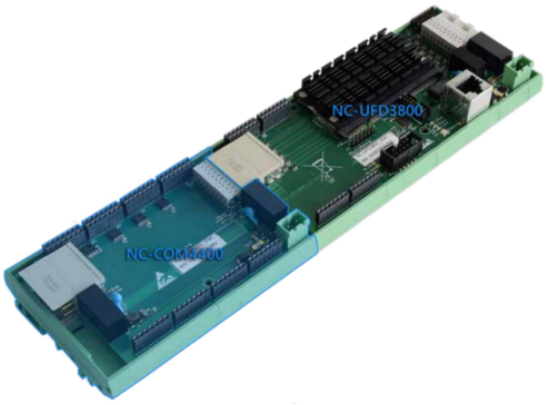
Obr. 1.10 Microchip dsPIC33CK64MC105, *prevzaté z* [7]

1.2.4 Micronova NovaCarts Universal FPGA Device

Riešenie je súčasťou väčšieho celku systému reálneho času (RT System), ktorý firma používa pre automotive testovanie, zároveň je možné zariadenie ovládať samostatne. Okrem protokolu SENT podporuje emulovanie protokolov PSI5, ISO SPI.

Modul je založený na čipe Zynq, ktorý obsahuje hradlové pole. Konštrukcia zariadenia je uspôsobená pre montáž na DIN lištu. Zariadenie je možné rozšíriť pripojením modulov NovaCarts Connection Module a získať tak celkový počet 18 vstupne výstupných slotov. [8]

Výhodou je možnosť rozšírenia vstupov a výstupov použitím prídavných modulov. Kombinácia bežného procesoru s hradlovým poľom umožňuje rýchlu obsluhu výstupov, zároveň ale prináša silnú závislosť na použitom hardvéri. Zároveň je toto riešenie nevyhovujúce pre nízku flexibilitu možnosti úpravy pre konkrétne potreby vývoja, nakoľko sa jedná o proprietárny produkt firmy Micronova.



Obr. 1.11 Micronova NovaCarts Universal FPGA Device, *prevzaté z* [8]

1.2.5 Mach Systems SAE J2716 (SENT) to RS-232/CAN bus gateway

Zariadenie poskytuje dva obojsmerné SENT kanály a je dostupné v troch variantoch prevodu, a to na RS-232, CAN, USB. Jednotlivé kanály môžu byť nezávisle nastavené ako prijímacie alebo odosielacie. So zariadením je dodávaný PC program SENT Gateway Analyser. Daný softvér slúži na konfiguráciu a záznam nameraných hodnôt. Pre ovládanie slúži RS-232, CAN alebo USB zbernica. Zároveň je možné injektovať chybu do výpočtu CRC. [9]

Zariadenie dokáže inteligentne filtrovať prijatú SENT komunikáciu, aby nebola zahltená výstupná zbernica. Odosielací kanál preposiela všetky správy alebo sú preposielané s periódou 100 ms, alebo preposielané pri zmene v rýchlom rámci s periódou 1 s. [10]

Nevýhodou je závislosť na tretej strane, tá môže spôsobiť problémy v ďalšom vývoji, napríklad pri využití proprietárneho hardvéru a jeho nedostatku. Využitie takéhoto spôsobu prináša nedostatky s použitím closed-source riešenia, ktoré sú nevhodné pre prototypový vývoj so špecifickými nárokmi.

Obr. 1.12 Mach Systems SAE J2716, *prevzaté z* [9]

1.3 Analýza komunikácie akceleračného pedálu

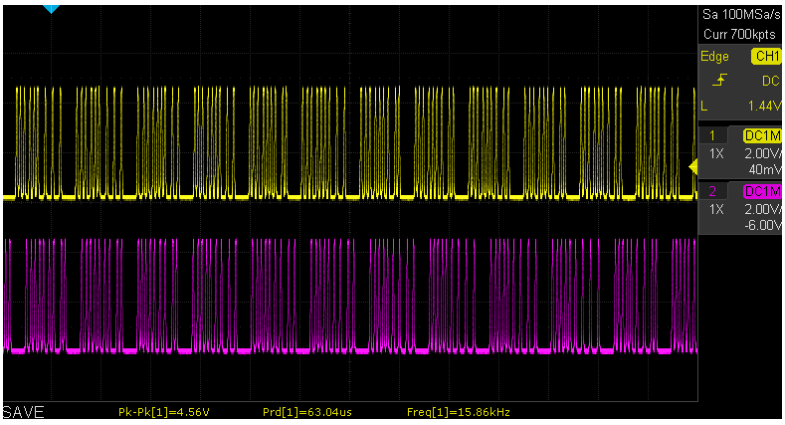
Pre referenciu generovaného protokolu bol zvolený akceleračný pedál z koncernu Volkswagen. Ten bol pripojený na externý zdroj napätia a následne bola komunikácia analyzovaná pomocou osciloskopu.

Priebeh komunikácie bol meraný dvoma spôsobmi. Prvým bola manuálna analýza pomocou osciloskopu, pre získanie základných vlastností komunikácie. Akceleračný pedál komunikuje na dvoch kanáloch SENT. Tieto kanály nie sú medzi sebou vzájomne synchronizované. Zároveň sú logické hodnoty na zbernici voči norme invertované.

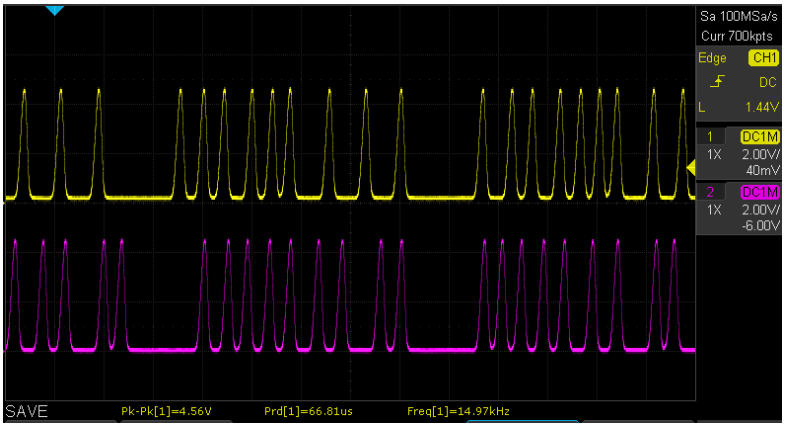
Každá správa je zložená zo synchronizačného pulzu nasledovaného 8 nibble dát. Dáta sa menia s pohybom pedálu.

Pretože správa má kvôli vlastnostiam protokolu premenlivú dĺžku, analýza osciloskopom sa dala vykonať len pri zastavení merania. Pre detailnejšiu analýzu dát, bol takýto záznam exportovaný do csv súboru a následne spracovaný skriptom, ktorý bol použitý aj pre testovanie výsledného riešenia. Jeho bližší popis je v kapitole Testovanie 4.2.

Následná časová analýza umožnila urýchlenie spracovania viacerých správ. Kvôli odchýlke, ktorá môže dosiahnuť až 20 %, boli namerané hodnoty spriemerované. V priemere mal každý nibble prefix s dĺžkou trvania 6 μs počas ktorej bola hodnota v log. 1. Synchronizačný pulz trval v priemere 162 μs, z čoho sa podľa vzťahu 1.1 dá odvodiť dĺžka jedného tick ako 3 μs.



Obr. 1.13 Priebeh komunikácie akceleračného pedálu



Obr. 1.14 Bližšia analýza komunikácie akceleračného pedálu

1.4 Dostupné hardvérové riešenia

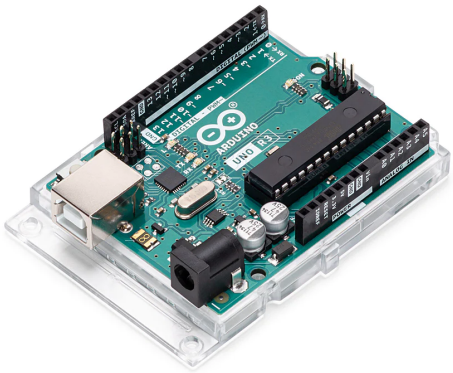
Pre zachovanie nezávislosti riešenia na hardvéri bola zvolená platforma Arduino. Jednotlivé modely sa od seba odlišujú v použitých procesoroch, preto bolo pri návrhu dôležité využitie spoločných vlastností v čo najväčšej miere. Zároveň táto platforma poskytuje množstvo knižníc, ktoré môžu urýchliť vývoj. Jedná sa najmä o knižnice pre komunikáciu s periférnymi zariadeniami.

1.4.1 Arduino Uno/Mega

Platforma Arduino zahrnuje vývojové dosky používajúce procesory od spoločnosti Atmel. Medzi najrozšírenejšie patria modely s procesormi ATmega328p (Uno, Nano), ATmega2560 (Mega). Oba spomínané procesory sú založené na 8bitovej platforme s frekvenciou 16 MHz. Procesor ATmega2560 obsahuje dva 8bitové a štyri 16bitové časovače (čítače) [11] oproti ATmega328p, ktorý poskytuje iba dva 8bitové a jeden 16bitový čítač (časovač). [12]

Ďalšie špecializované modely s rovnakými procesormi sa môžu odlišovať počtom a typom periférií, ktorý nie je rozhodujúci pre potreby tejto práce. Tieto špecifické modely môžu byť použité pri rozšírení stávajúceho riešenia v ďalšom vývoji podľa potreby danej aplikácie.

Výhodou použitia riešenia založeného na procesore ATmega328p je prenositeľnosť kódu, ktorý môže byť koncipovaný nezávisle na hardvérovej implementácií, pretože procesor ATmega2560 obsahuje rovnaké komponenty a ďalej rozširuje ich funkcionalitu.

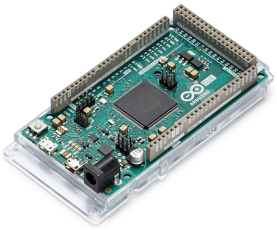


Obr. 1.15 Arduino Uno Rev3, *prevzaté z* [13]

1.4.2 Arduino Due

Model založený na procesore AT91SAM3X8E. Odlišuje sa 32bitovou architektúrou s frekvenciou 84 MHz. Oproti modelom Uno, Mega, ktoré umožňujú 5V výstupy, umožňuje Due iba 3,3V výstupy. Obsahuje rozšírené možnosti komunikácie s perifériami. Napr. integrovaný časovač (čítač) obsahuje deväť kanálov. [14]

Výhodou tejto dosky je vyššia frekvencia a väčší počet periférií procesoru, ktorý zdiela open-source platformu Arduino. Nevýhodou je menšie výstupné napätie a iná architektúra, ktorá zmenšuje prenositeľnosť kódu (napr. na 32bitový procesor), ktorá by spôsobila závislosť riešenia na jednom konkrétnom type procesoru.



Obr. 1.16 Arduino Due, *prevzaté z* [15]

1.4.3 ESP32

Vývojová doska od spoločnosti Espressif. Podporuje Arduino knižnice a je možný prístup cez Arduino IDE. [16]

Hlavným rozdielom od spomínaných možností je použitie procesoru s dvoma jadrami a frekvenciou 240 MHz. V čase písania tejto práce existujú tri typy použitých modulov: WROOM a WROVER a MINI. Modul WROVER poskytuje 8 MB PSRAM a modul Mini 2 MB PSRAM. [17]

ESP-IDF je oficiálny framewok (Espressif Internet Development Framework), ktorý umožňuje jednoduchší vývoj nových aplikácií vďaka podpore ďalších softvérových komponent. Výhodou IDF je podpora upraveného operačného systému FreeRTOS. [18]

ESP-IDF FreeRTOS je založený na operačnom systéme FreeRTOS. Podporuje fungovanie na dvoch uvedených jadrách, preto sa oproti klasickému FreeRTOS systému odlišuje vo fungovaní určitých funkcií. Zároveň tento prístup umožňuje rýchly a ľahký vývoj, pretože je možné použiť FreeRTOS riešenia priamo alebo len s minimálnymi zmenami potrebnými pre fungovanie na danej platforme. [19]

Výhodou je možnosť použitia stávajúcich algoritmov bez nutnosti zmien a využitie knižníc pre prístup k perifériám ako podpora pre CAN ovládač, I2C komunikácia. Ďalšou výhodou je možnosť škálovania, pretože existuje niekoľko mutácií dosky (Ethernet, Wi-Fi, Bluetooh), ktoré umožňujú rozšírenie periférií bez potreby zásahu do jadra kódu, obdobne ako pri doskách Arduino Uno alebo Arduino Mega.

Obr. 1.17 ESP 32-DevKitC, *prevzaté z* [20]

1.4.4 STM32

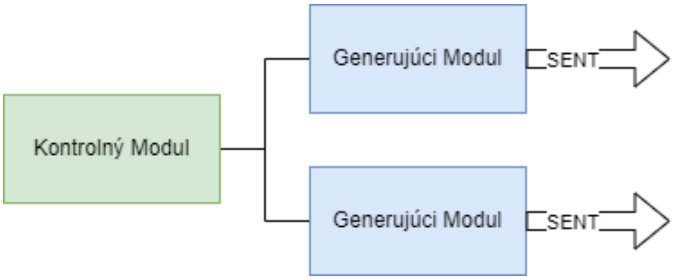
Model, ktorý oficiálne nie je zaradený do platformy Arduino, ale poskytuje možnosti využitia Arduino knižníc. Používa procesor ARM STM32 využívajúci 32bitovú architektúru s frekvenciou 72 MHz. Procesor obsahuje tri 16bitové časovače/čítače a výstupy na úrovni 3.3 V. [21]

Výhodou riešenia sú menšie rozmery dosky, vysoký výkon a možnosť pristupovať k procesoru cez Arduino IDE a následné použitie kompatibilných knižníc. Nevýhodou je malá prenositeľnosť kódu, kvôli použitiu ARM procesoru a následná závislosť na danej triede procesorov.

Obr. 1.18 STM32F103C8, *prevzaté z* [22]

Kapitola 2
Návrh riešenia

Problém emulovania protokolu sa skladá z časti generovania SENT správy s korektnými časovými parametrami. Správa musí dáta korektne reprezentovať vo forme jednotlivých nibble a obsahovať korektný kontrolný súčet. Pre zadávanie hodnôt správy je nutné obslúženie užívateľských vstupov, fyzických a vzdialených. Preto je návrh rozdelený na dve hlavné časti, generujúci modul a kontrolný modul.



Obr. 2.1 Blokové schéma návrhu riešenia

2.1 Generujúci modul

Problém generovania je popísaný od najnižšej vrstvy po najvyššiu. Generujúci modul bude musieť ovládať fyzické výstupy v korektný čas, a tak generovať správu SENT protokolu. Zároveň musí obsluhovať komunikáciu s Kontrolným modulom prostredníctvom zbernice.

Zariadenie bude generovať protokolárne pulzy v rozmedzí 0–4.1 V. Úbytok napätia na zbernici pri konkrétnej komunikácií nie je v rámci návrhu uvažovaný a môže byť riešený pridaním externého zosilňovača na zbernicu. Ďalšie nastavenie protokolárnej komunikácie musí byť potom upravené pomocou programových premenných, aby zachovávalo časové požiadavky komunikácie.

Samotné protokolárne generovanie sa dá rozdeliť na štyri hlavné časti:

- generovanie prefixovej hodnoty nibble synchronizačného pulzu,
- generovanie synchronizačného pulzu,
- generovanie prefixovej hodnoty nibble,
- generovanie dátových častí nibble.

Generovanie protokolárnych správ sa dá previesť na jednoduchú úlohu nastavovania digitálneho výstupu v čase. Modul bude zachovávať dostatočné rozlíšenie a presnosť pre danú aplikáciu. V aktuálnom použití 3 μs. Najdlhším intervalom je synchronizačný pulz. V aktuálnom použití $51*3 = 153\mu s$. Pre zachovanie rozšíriteľnosti je potreba uvažovať aj o iných dĺžkach pulzu. Maximálnou možnou dĺžkou tick podľa normy je 90 μs. Potom je dĺžka pulzu $51 * 90 = 4590\,\mu s$.

Pri generovaní bude potreba zachovať parametrizáciu programových premenných, aby bola zachovaná flexibilita pre iné možnosti použitia. Najdôležitejším parametrom je dĺžka tick a dĺžka prefixových hodnôt. Tieto hodnoty nebudú nastaviteľné počas behu, pretože sú špecifické pre jednotlivé spôsoby použitia zodpovedajúce konkrétnemu typu periférie ktorú má zariadenie emulovať.

Za účelom minimalizácie výpočtového času bude k vstupným dátam modul pristupovať ako ku všeobecným 8 nibble dát a synchronizačnému pulzu. Obsah nibble bude stanovený vyššími vrstvami, resp. Kontrolným modulom. Jeden generujúci modul bude obsluhovať niekoľko nezávislých protokolárnych výstupov.

Okrem protokolárnej komunikácie bude musieť generujúci modul obsahovať možnosť komunikácie po zbernici s Kontrolným modulom. Táto komunikácia by mala prebiehať asynchrónne od generovania protokolu SENT a generovanie správ by nemalo byť príjmom dát prerušené.

Obr. 2.2 Blokové schéma generujúceho modulu

2.2 Kontrolný modul

Architektúra modulu je popísaná od najvyššej vrstvy, ktorá obsluhuje aktuálnu komunikáciu s používateľom, po najnižšiu vrstvu, ktorá komunikuje s Generujúcim modulom.

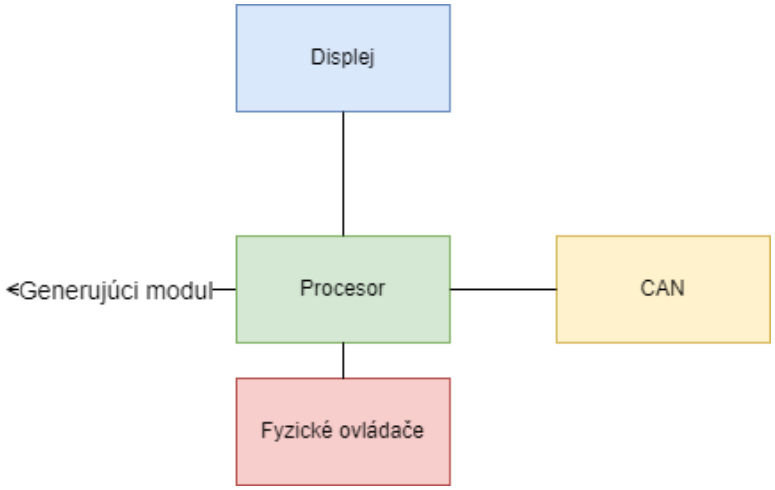
Pre poskytnutie používateľského výstupu bude kontrolný modul obsahovať displej na zobrazenie aktuálne generovanej hodnoty. Zároveň bude displej poskytovať zobrazenie typu vstupu, ktorý sa používa pre generovanie. Pre aktuálne potreby práce sa jedná o fyzické ovládače a zbernicu CAN.

Pre priame nastavenie vstupu budú slúžiť fyzické ovládače, ktoré budú riadiť parametre správy a výber kanálu. Pre účely práce je počet kanálov obmedzený na dva, ktoré sú obsluhované na jednom zadanom Generujúcom zariadení, pri ďalšom vývoji môže byť modul rozšírený o ďalšie funkcie adresovania modulov a ich jednotlivých kanálov. V rámci jedného kanálu bude nastavovať dve 12bitové hodnoty a jednu 4bitovú hodnotu statusu.

Pre vzdialený prístup bude kontrolný modul obsahovať obsluhu zbernice CAN, ktorá umožní rovnakú škálu možností nastavenia parametrov. Pre nastavenie bude slúžiť jedna správa, ktorá bude obsahovať všetky potrebné parametre pre daný kanál.

Kontrolný modul bude prevádzať užívateľské dáta do unifikovanej podoby. Tento formát bude slúžiť ako jednotné rozhranie na prenos hodnôt medzi jednotlivými perifériami (fyzické ovládače, CAN zbernica, displej) a zároveň umožní jednoduché rozšírenie v budúcnosti.

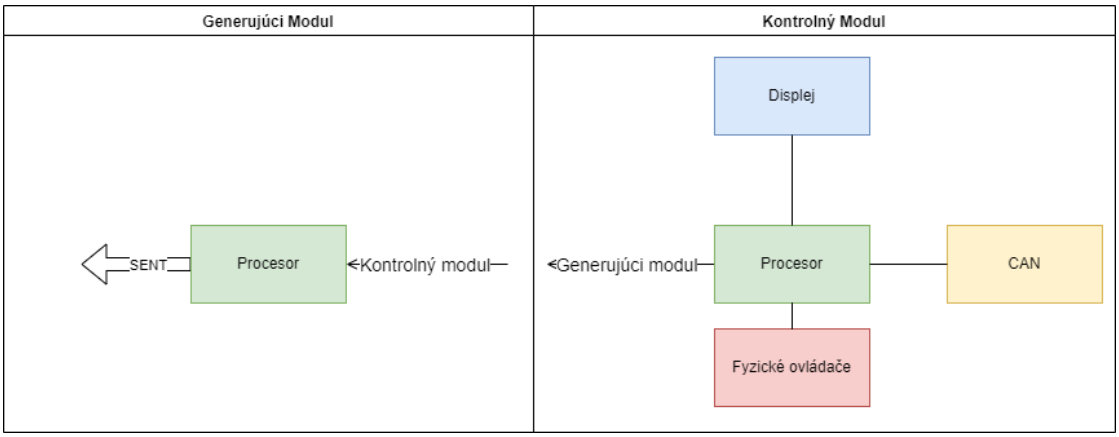
Spracované používateľské dáta bude kontrolný modul upravovať podľa pravidiel stanovených normou. Dáta budú usporiadané do korektného poradia nibble podľa významnosti bitov. Ďalej musí modul vykonať korektný výpočet kontrolného súčtu definovaného normou. Takto upravené dáta budú nasledovne prispôsobené do rámca stanoveného protokolom použitej zbernice a následne odoslané do Generujúceho modulu.



Obr. 2.3 Blokové schéma kontrolného modulu

2.3 Prepojenie modulov

Aby bolo možné nastaviť hodnoty jednotlivých nibble a zároveň umožniť rozšíriteľnosť celého zariadenia o ďalšie generujúce moduly, bude modul komunikovať s kontrolným modulom pomocou zbernice. Pre potreby práce bude dostačujúca jednosmerná komunikácia od kontrolného modulu. Komunikácia medzi modulmi nemá stanovené žiadne časové obmedzenie. To znamená, že nie je stanovená maximálna doba od prijatia užívateľského vstupu a zmeny generovanej správy protokolu SENT.



Obr. 2.4 Blokové schéma návrhu zapojenia modulov

Kapitola 3
Riešenie

V rámci realizácie a vývoja bola dodržaná štruktúru stanovená návrhom. Bližšie kroky vývoja sú popísané v nasledujúcich kapitolách.

3.1 Výber hardvérového a softvérového riešenia

Pre hardvérovú realizáciu boli zvolené dosky Arduino Uno pre Generujúci modul a ESP-32 pre Kontrolný modul.

Doska Arduino Uno obsahuje dostatočné hardvérové prostriedky na samotné generovanie protokolu SENT. Zároveň sú dostupné rôzne typy knižníc na komunikáciu po zbernici (I2C, SPI,...). Procesor ale nedosahuje potrebný výkon na obsluhu vyšších vrstiev na spracovanie používateľskej komunikácie. Výhodou je, že sa jedná o riešenie, ktoré umožňuje rýchlu a jednoduchú implementáciu algoritmov potrebných pre generovanie protokolu.

Vývojová doska ESP-32 bola zvolená z dôvodu podpory viacvláknového programovania. Pre jednoduchú implementáciu bol zvolený upravený operačný systém ESP-IDF FreeRTOS. Využitie dvoch paralelných procesov umožnilo jednoduchú implementáciu vyšších vrstiev. Viacvláknové riešenie obsahuje dostatočné prostriedky na implementáciu nižších vrstiev. Použitie jedného procesoru znižuje náklady, zároveň ale zabraňuje rozšíreniu riešenia o ďalšie výstupy.

Pretože je použitý operačný systém reálneho času, bolo by nutné použiť odlišné prístupy pre generovanie protokolárneho výstupu SENT v rámci jednej úlohy. Takéto riešenie sa ukázalo ako veľmi časovo a algoritmicky náročné, a preto nebolo vhodné pre účely tejto práce.

Pre účely programovanie daných dosiek bolo dostupné vývojové prostredie Arduino IDE použité pri vývoji a testovaní dielčich častí riešenia. Samotné moduly boli realizované v prostredí PlatformIO, ktoré umožňuje vytvárať projekty, ktoré sú kompatibilné s platformou Espressif a Arduino.

3.2 Generujúci modul

3.2.1 Meranie časového intervalu

Pre generovanie protokolárnej komunikácie bolo potrebné dodržať predpísané časové rozlíšenie pri ovládaní výstupov. Pri vývoji bolo otestovaných niekoľko možností generovania intervalu.

Knižnice Arduino poskytujú funkcie millis() a micros(), pre získanie časovej hodnoty od počiatku behu programu. Meraním bolo zistené, že konkrétne funkcia micros() dosahovala odchýlky až 4 μs, ktorá nie je pre dané použitie dostatočná.

Ďalšou z možností merania časového intervalu bolo použitie vnútorných časovačov. Aby bolo možné preniesť program na čo najväčší počet procesorov použitých v doskách Arduino, boli uvažované iba časovače 0, 1, 2. Časovače 0 a 2 sú 8 bitové a časovač 1 16 bitový. Z dostupných preddeličiek 1, 8, 64, 256, 1024 bola zvolená hodnota pres = 8. Frekvencia použitého časovača f = 16 MHz. Výsledná frekvencia bola podľa vzťahu 3.1 znížená na $f_{pres} = 2\ MHz$. Podľa vzťahu 3.2 je výsledná perióda $T_{pres} = 0.5\ \mu s$. Táto perióda poskytla dostatočné rozlíšenie potrebné pre správu s dĺžkou tick 3 μs. Použitie ďalšej hodnoty 64 by podľa vzťahov 3.1 a 3.2 malo za výsledok periódu 4 μs, ktorá je mimo 20% tolerancie protokolu.

$$f_{pres} = \frac{f}{pres} \tag{3.1}$$

$$Tpres = \frac{1}{f_{pres}} \tag{3.2}$$

Takto zvolené rozlíšenie znamenalo, že jedna hodnota v čítači reprezentuje 0.5 μs. Najkratší interval definovaný v protokole je dĺžka prefixu (5 tick), ktorá pri dĺžke tick 3 μs trvá 15 μs, to znamená 30 hodnôt v časovači. Najdlhším intervalom je druhá časť synchronizačného pulzu (51 tick), ktorá pri dĺžke tick 3 μs trvá 153 μs, to znamená 306 hodnôt v časovači. Pre splnenie tohto rozsahu bol zvolený 16bitový časovač 2. Zároveň sa tento časovač nachádza v procesoroch ATmega328p aj ATmega2560.

3.2.2 Nastavenie časovača

Časovač bol nastavený do módu porovnávania (compare). V takomto móde je iterovaná hodnota v rozsahu 0x0–0xffff. Pri zhode hodnoty časovača s porovnávacím registrom bolo vyvolané prerušenie. Na danom časovači sa nachádzajú dva porovnávacie registre. Jeden časovač zvládne obslúžiť dva výstupné protokolárne kanály.

Meraný časový interval bol reprezentovaný ako rozdiel aktuálnej hodnoty časovača a daného porovnávacieho registra. Hodnota bola podľa potreby inkrementovaná, tak aby nasledujúce prerušenie nastalo za požadovaný čas.

Kritickou časťou bol výpočet a nastavenie novej hodnoty porovnávacieho registra. Nová hodnota musela byť vždy vyššia, než aktuálna hodnota časovača. Ak nastalo oneskorenie medzi nastavovaním novej hodnoty ďalšieho intervalu a behom čítača, vznikol interval o nesprávnej dĺžke, ktorá trvala jednu periódu časovača (mínus hodnota porovnávacieho registra).

Testovaním bolo zistené, že nastavenie príznaku v obsluhe prerušenia a následné spracovanie v hlavnej loop() slučke je nedostačujúce a má veľkú latenciu. Preto obsluha prerušenia musí vykonať všetky potrebné úkony spojené s nastavením registrov a ovládaním výstupov.

3.2.3 Optimalizácia

Pri implementácií na procesore ATmega328p bolo potrebné softvérovú implementáciu optimalizovať, pre využitie minimálneho počtu inštrukcií a nízkeho prístupu do pamäte, aby obsluha prerušenia trvala najkratší možný čas. Preto bola navrhovaná architektúra kanálov implementovaná ako samostatný súbor globálnych premenných. Využitie štruktúr alebo objektov implikovalo použitie nepriameho adresovania, ktoré vytváralo dodatočné nároky na prístupy ku konkrétnym hodnotám. Kanál bol abstraktným návrhovým vzorom, ktorý bol uvedený prefixom názvu (CH1, CH2) premenných.

Knižnica Arduino.h poskytuje funkciu digitalWrite() pre kontrolu výstupov. Táto funkcia obsahuje ďalšie volania funkcií, ktoré spomaľujú kontrolu zápisu, a preto bola nahradená priamym zápisom do registra kontrolujúceho hodnotu výstupného pinu. Procesor podporuje funkciu prepínania výstupnej hodnoty kanálu pri zhode kontrolného registra a časovača. Tento prístup bol meraním overený ako najrýchlejší, ale zároveň najviac náchylný na propagáciu chyby. Pri

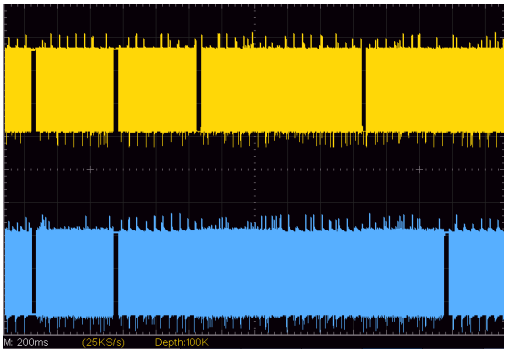
manuálnom zápise na výstup bola hodnota vždy deterministická podľa fázy správy. Pri prepínaní výstupu bol pri oneskorení obsluhy prerušenia prepnutý výstup do zlej hodnoty voči fáze správy, čo spôsobilo jej celé invertovanie až do ďalšej chyby alebo reštartu zariadenia.

3.2.4 Komunikácia s kontrolným modulom

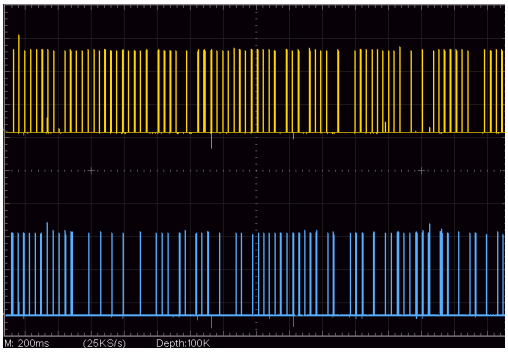
Prvotným riešením komunikácie bol protokol I2C. Tento protokol bol zvolený pretože ho Kontrolný modul využíva pre komunikáciu s displejom. Zároveň umožňuje pripojenie viacerých prijímacích zariadení pri zachovaní malého počtu vodičov. Pre softvérové riešenie bola použitá knižnica Wire.h.

Pri implementovaní prijímača I2C zároveň s vysielačom SENT boli ale zistené nedostatky pri použití spomínanej knižnice. Arduino obsahuje hardvérovú podporu protokolu a prijímanie správy vyvoláva prerušenie, následne sú dáta uložené do knižničného zásobníku, z ktorého je ich možné vyčítať po jednom bajte. Prerušenia I2C a SENT medzi sebou vyvolávali interferencie, ktoré mali za následok časové nepresnosti generovania SENT intervalov.

Knižnica bola upravená, aby bolo dáta možné kopírovať vo väčšom počte funkciou memcpy do užívateľského poľa daného kanálu. Bola implementovaná funkcia flush() pre vyprázdnenie prijímacieho zásobníku. Zároveň bol manuálne obnovený príznak globálneho prerušenia. Tento príznak bráni vyvolaniu obsluhy prerušenia pri vykonávaní inej obsluhy. Protokol I2C sa ukázal ako príliš výpočtovo náročný na obsluhu prijatých dát paralelne s vysielaním protokolárnych výstupov. Vhodné výsledky neboli dosiahnuté ani po úprave knižnice Wire.h. Merania z osciloskopu 3.1 a 3.2 zachytávajú komunikáciu pri použití upravenej knižnice. V nepretržitom prúde vysielaných správ viditeľné miesta, na ktorých nastala chyba nastavenia časového intervalu spôsobená konfliktom obsluhy prerušení pri použití neupravenej knižnice Wire.h. Samotné povolenie globálneho príznaku prerušenia spôsobilo nestabilnosť behu programu, ktorý po nedefinovanej dobe prestal vysielať SENT signály.



Obr. 3.1 Výpadky generovaných SENT správ pri spracovaní I2C komunikácie

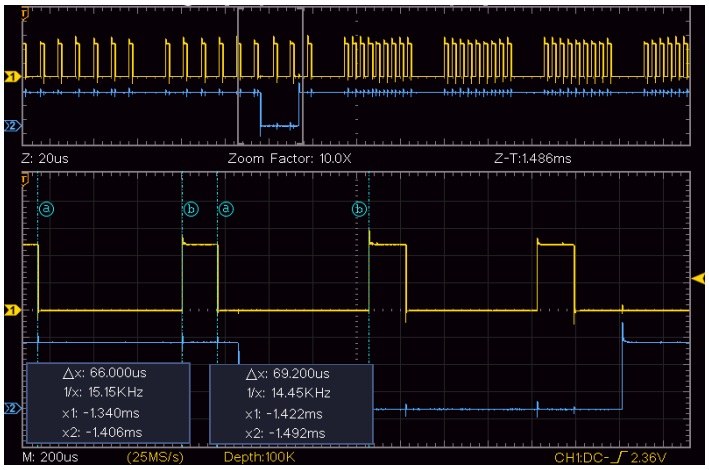


Obr. 3.2 Výpadky generovania jednotlivých nibble správy

Na základe predchádzajúcich meraní bol zvolený protokol SPI, ktorý má menšie výpočtové nároky na obsluhu prerušenia, poskytuje možnosť pripojenia viacerých zariadení a má hardvérovú podporu v procesore. Platforma Arduino poskytuje knižničné funkcie pre obsluhu komunikácie v knižnici SPI.h. Nevýhodou tohto riešenia bola potreba väčšieho množstva vodičov SPI (4) oproti I2C (2).

Pri prijatí jedného bajtu bolo vyvolané prerušenie, v ktorom sú dáta vyčítané z registra SPDR. Aby bolo zabránené interferencií medzi obsluhami SPI a SENT bol globálny príznak prerušenia obnovený na začiatku obsluhy prerušenia, obdobne ako pri úprave knižnice Wire.h pri optimalizácií komunikácie cez I2C. Táto úprava dovolila udržať presnosť správy SENT a zároveň príjem nových dát. Ak by nastala kritická časová závislosť, bude iba jedna správa vyslaná s nesprávnymi dátami a ďalšia chyba po dokončení SPI prenosu nebude propagovaná.

Meraním bolo zistené, že príjem správy SPI ovplyvňuje časovanie vysielania správy SENT. Zároveň, bola overená hypotéza, že sa následná chyba nepropaguje do ďalších nibble a je ovplyvnený len jeden rámec. Na obrázku 3.3 je zobrazená správa SENT na kanále 1 a SPI Slave Select signál na kanále 2. Zobrazená správa obsahovala všetky nibble v hodnote 15. Prvý meraný interval mal korektnú dĺžku 66 μs a počas obsluhy SPI komunikácie, indikovanej zmenou hodnoty Slave Select, je tento interval 68.8 μs. Ak by súčasné riešenie bolo vyhodnotené ako nedostačujúce, bolo by vhodné použiť výkonnejší hardvér.



Obr. 3.3 Protokolárna správa SENT CH1, SPI Slave Select CH2

Dáta boli z registra vyčítané po jednom bajte a následne uložené do poľa príslušného kanálu. Rámec začínal identifikačným číslom kanálu. Nasledovalo 8 bajtov kódujúcich hodnoty jednotlivých nibble. Pretože nibble môže niesť iba 4bitovú informáciu, boli vyššie 4 bity bajtu využité pre indexovanie. To znamená, že výpadok jedného bajtu by neovplyvnil zostávajúce informácie. Pri detekcií chybnej hodnoty na výstupe musí používateľ rámec znova odoslať.

#	dáta	hodnota
0	id kanála	0xf1–0xf2
1	Status	0x00–0x0f
2	signál 1 dáta[1]	0x10–0x1f
3	signál 1 dáta[2]	0x20–0x2f
4	signál 1 dáta[3]	0x30–0x3f
5	signál 2 dáta[3]	0x40–0x4f
6	signál 2 dáta[2]	0x50–0x5f
7	signál 2 dáta[1]	0x60–0x6f
8	CRC	0x70–0x7f

Tabuľka 3.1 Rámec SPI komunikácie

Dáta na výstupe boli aktualizované počas synchronizačného pulzu. V tomto pulze sa nemenila hodnota dát, a tak bola zabezpečená konzistencia správy bez ohľadu na čas prijatia SPI rámca. Vysielací zásobník SENT a prijímací zásobník SPI boli zamenené a staré pole SENT bolo využité pre ukladanie nového rámca SPI. Pokiaľ neboli polia zamenené a validované pre príjem nových dát, novo prichádzajúce dáta neboli uložené do pamäte. Tento prístup poskytol prijímací priestor o dĺžke jednej správy. Dáta v zásobníku čakajú na spracovanie maximálne dĺžku jednej SENT správy. Jedna SENT správa, s maximálnou hodnotou 15 v každom nibble, s dĺžkou tick 3 μs trvá 272 tick (synchronizačný pulz 56 tick, 8 dátových nibble 12 + 15 tick), to znamená 816 μs. Filtrovanie, ktoré by zabránilo zahlteniu, musí byť riešené SPI master zariadením.

3.3 Kontrolný modul

3.3.1 Realizácia užívateľského vstupu a výstupu

Vyššie vrstvy kontrolného modulu boli rozdelené medzi dve vlákna programu.

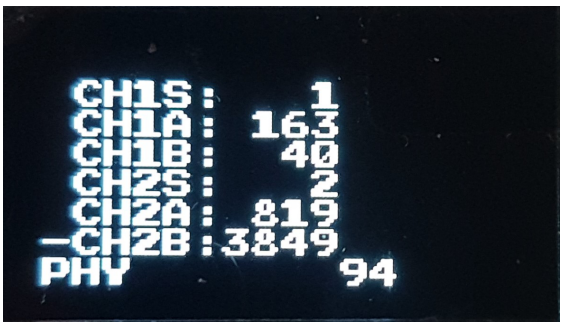
Prvé vlákno implementovalo najvyššiu vrstvu, komunikáciu s používateľom a CAN zbernicou.

Používateľ môže nastavovať jednotlivé parametre správy pre oba kanály pomocou rotačného enkodéra. Použitie potenciometra sa pre danú dosku ukázalo ako nevyhovujúce, pretože vstavaný analógovo-digitálny prevodník nedokázal odfiltrovať rušenie pri čítaní analógovej hodnoty, ktorá musela obsiahnuť šírku 12 bitov s rozlíšením na 1 bit. Použitie kĺzavého priemeru pri čítaní hodnoty bolo nedostačujúce a implementácia hardvérového filtrovania by zvýšila náročnosť modulárneho riešenia.

Zároveň použitá knižnica PCNT.h umožnila rýchlu a jednoduchú implementáciu spracovania hodnoty rotačného enkodéra. V knižnici je implementovaný softvérový filter digitálneho signálu, ktorý bol pre filtráciu komunikácie rotačného enkodéru dostatočný. Priame mapovanie hodnoty rotačného enkodéra na výstupnú hodnotu nibble nebolo používateľsky prívetivé. Hodnota enkodéra je čítaná v rozsahu -100–100, ktorými sú následne inkrementované alebo dekrementované zvolené nibble.

Pre výber jednotlivých častí správy a pre odoslanie boli zvolené dve fyzické tlačidlá. Jedno tlačidlo bolo súčasťou rotačného enkodéra, druhé bolo umiestnené samostatne. Tlačidlo na enkodéri potvrdzovalo hodnotu a posúvalo výber na ďalšiu položku, druhé tlačidlo slúžilo ako potvrdenie hodnôt, ktoré sú odoslané na spracovanie do druhého vlákna a následne odoslané do generujúceho modulu.

Na zobrazenie hodnôt bol zvolený OLED displej, zobrazujúci aktuálne nastavené dáta a zdroj hodnoty. Konkrétny displej komunikuje pomocou zbernice I2C. Za účelom obsluhy displeja vznikla samostatná knižnica, ktorá umožnila zobrazenie číselných a textových hodnôt. Na zobrazenie textu vznikol font implementujúci písmená a čísla so základnými znakmi. Pre ďalšie potreby môže byť množina znakov rozšírená.



Obr. 3.4 Menu na OLED displeji pri voľbe dát z rotačného enkodéra

Po obslúžení fyzického vstupu nasledovalo čakanie 100 milisekúnd na prichádzajúcu správu po zbernici CAN. Toto čakanie bolo zároveň použité ako súčasť prevencie zahltenia SPI komunikácie na periférnych zariadeniach. Pre aktuálny rozsah práce bola implementovaná správa pre zápis novej hodnoty. Správa CAN umožňuje nastavenie všetkých parametrov pre jeden kanál. Komunikácia prebieha s hodinovou frekvenciou 80 MHz a rýchlosťou prenosu 500 kbit/s. Pre účely práce nebola implementovaná komunikácia FD rámcami.

Formát správy kopíruje rámec SENT protokolu. Prvý bajt označuje ID kanálu, ktorý má byť nastavený. Ďalej správa obsahuje 7 bajtov hodnôt jednotlivých nibble. Poradie bajtov zodpovedá poradiu jednotlivých nibble v správe SENT. Následný 8. bajt kontrolného súčtu nie je zadávaný, pretože je vypočítaný automaticky.

3.3.2 Spracovanie dát

Dáta z fyzického vstupu a CAN zbernice boli prevedené do unifikovanej podoby. Pre aktuálne riešenie je použitý formát dvoch 12bitových hodnôt dát, 4bitovej hodnoty statusu a 4bitovej hodnoty kontrolného súčtu.

Ak nastala zmena dát oproti predchádzajúcemu nastaveniu CAN zbernice, alebo bolo stlačené potvrdzovacie tlačidlo, boli takéto dáta odoslané pomocou fronty na spracovanie do druhého vlákna. Prijaté dáta boli následne spracované podľa špecifikácie použitia. Pre rozsah tejto práce neboli dáta z užívateľskej vrstvy upravované. Pre použitie emulátoru na emuláciu konkrétneho senzoru je možné vytvorenie ďalších prevodných funkcií, napríklad zrkadlenie hodnôt na oboch kanáloch, prevod na zabezpečenú správu, a pod.

Základnou implementovanou funkciou pre spracovanie dát je výpočet kontrolného súčtu. Pre výpočet bolo možné použitie dvoch prístupov.

Prvým prístupom bolo postupné delenie dát (XOR) s generujúcim polynómom a výsledkom bola hodnota súčtu. Tento spôsob výpočtu mal väčšiu časovú zložitosť, ale mal menšie pamäťové nároky.

Druhým prístup bolo vytvorenie tabuľky hodnôt delenia jednotlivých možných hodnôt 0–15. Následne boli dáta delené po jednotlivých nibble a výsledok operácie udával index, ktorý určoval hodnotu v tabuľke s výsledkom. Tento výpočet vyžadoval menší počet operácií, ale bol pamäťovo náročnejší.

Pre výpočet bol zvolený prístup s tabuľkou, pretože bola uprednostnená časová náročnosť pred pamäťovou. Pri nedostatku pamäte pri použití iného hardvéru alebo modifikácie softvéru je možné funkciu vymeniť za prvý typ výpočtu.

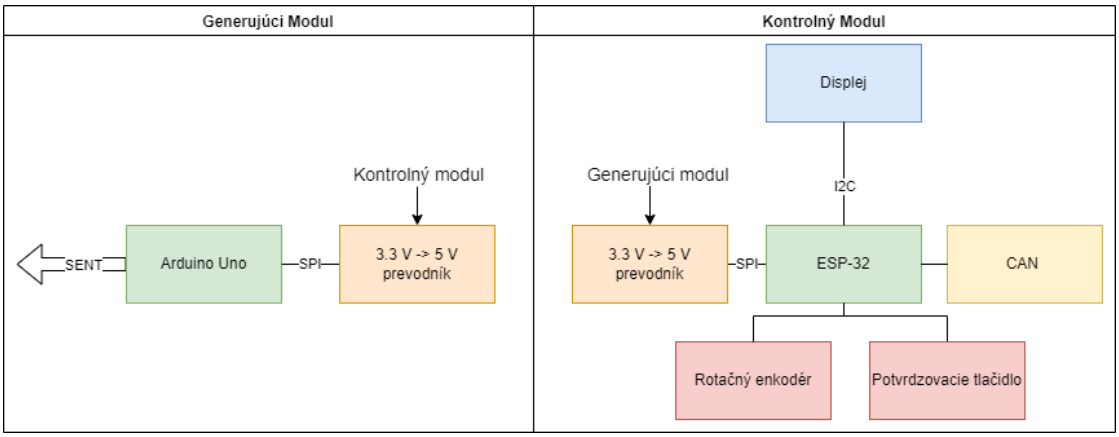
3.3.3 Komunikácia s generujúcim modulom

Spracované dáta vo formáte: číslo kanála, 8 nibble, boli posielané cez protokol SPI do periférneho zariadenia. Tento prenos slúžil na prepojenie implementovaných vrstiev medzi zariadeniami. Pre účely tejto práce je adresa zariadenia zadaná staticky bez možnosti užívateľskej zmeny. Pretože doska Arduino komunikuje s výstupným napätím 5 V a ESP32 komunikuje s napätím 3.3 V, bolo nutné použiť obojsmerný prevodník úrovní.

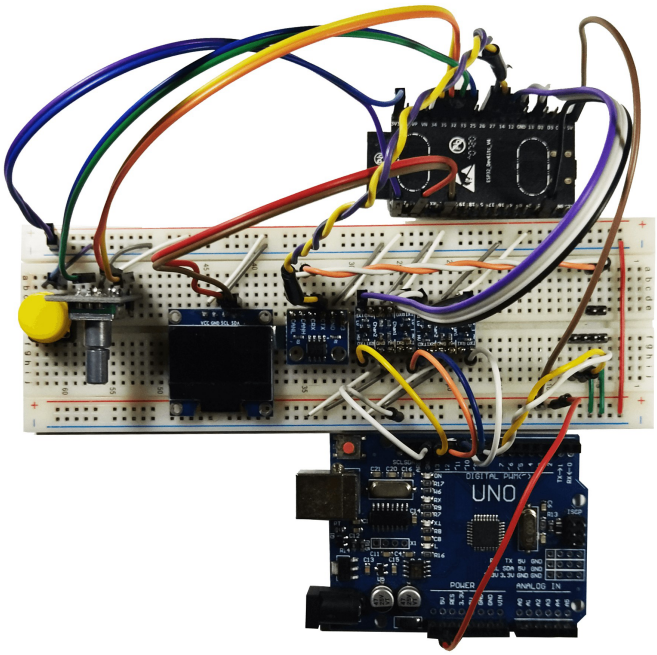
3.4 Realizácia hardvéru

Zariadenie bolo vytvorené pre prototypový vývoj, preto bola jeho realizácia uspôsobená pre možnosť ďalších úprav. Jednotlivé komponenty boli umiestnené na nepájavom kontaktnom poli, aby ich bolo možné pri ďalšom vývoji nahradiť podľa konkrétnej potreby. Použité vývojové dosky boli následne pripojené pomocou dupont prepojovacích vodičov.

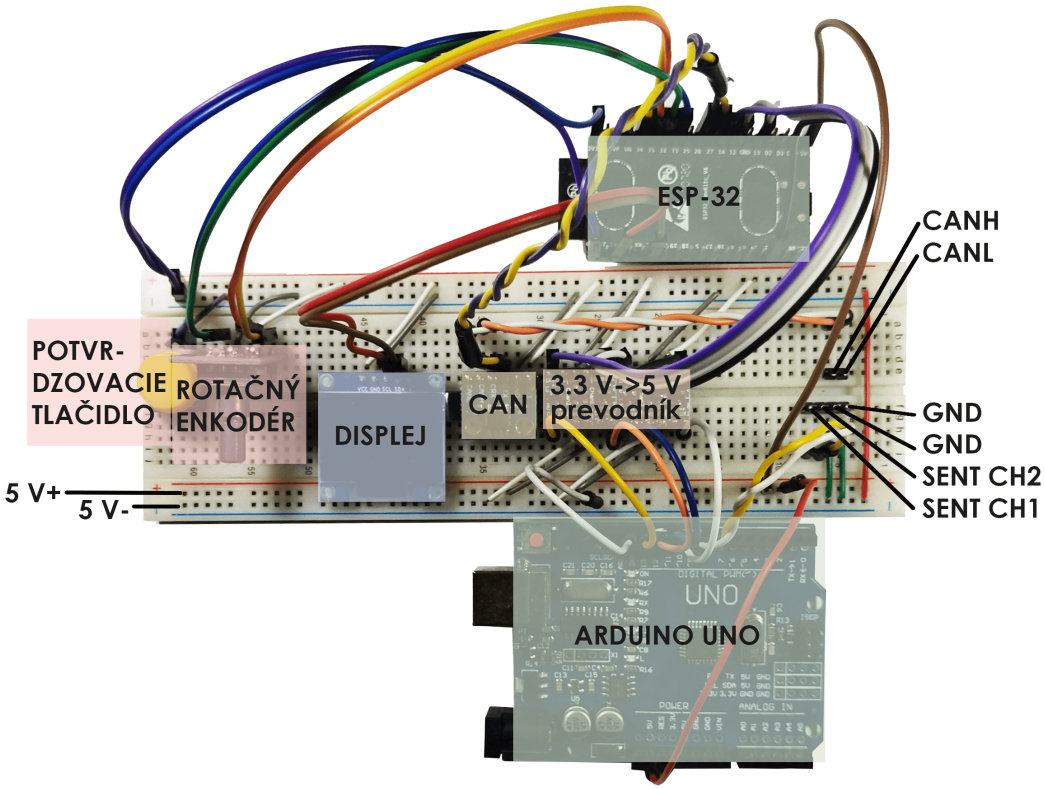
Pre zachovanie flexibility boli periférne zbernice SPI a CAN vyvedené na samostatné piny. Na dalšie piny boli vyvedené výstupy SENT protokolu.



Obr. 3.5 Blokové schéma výsledného zapojenia modulov



Obr. 3.6 Výsledné reálne zapojenia modulov na nepájavom kontaktnom poli



Obr. 3.7 Výsledné reálne zapojenia modulov

Kapitola 4
Testovanie

Jednotlivé časti riešenia boli oddelene testované pri ich vývoji. Na záver boli všetky časti testované ako jeden celok.

Pre urýchlenie testovania, bola pri testovaní komunikácie na zbernici medzi modulmi použitá ďalšia doska Arduino Mega, ktorá slúžila na testovanie protokolov I2C a následne SPI. Pre testovanie SPI boli implementované dva testovacie programy. SPIMaster, ktorý v stanovených intervaloch odosielal SPI rámec podľa definície 3.1. SPISlave, ktorý prijímal jednotlivé rámce a následne ich vypisoval na sériový port.

Za účelom testovania výstupu protokolu SENT boli zvolené dve metódy.

Prvým spôsobom bol integračný test, kedy bol analyzovaný priebeh napätia na osciloskope. Nevýhoda tohoto riešenia spočíva v skutočnosti, že jednotlivé SENT správy majú rozdielnu dĺžku a nie sú medzi sebou synchronizované. Tento prístup slúžil pre kontrolu chyby v jednotlivých rámcoch. Napríklad pri analýze odchýlky pri príjme SPI správy.

Druhá forma jednotkového testu bola softvérová (testovací skript), kedy boli exportované dáta z osciloskopu a tie boli následne spracované. Tento prístup bol vhodný pre analýzu statickej časti kódu a testovanie konštantnosti generovaných rámcov. Pre vybrané hodnoty boli porovnané používateľské vstupy s protokolárnymi výstupmi.

Pre vstupné dáta boli použité limitné hodnoty protokolu 0–15 v jednotlivých nibble. Zároveň boli testované jednotlivé úpravy generujúceho modulu pre rozličné hodnoty dĺžky tick.

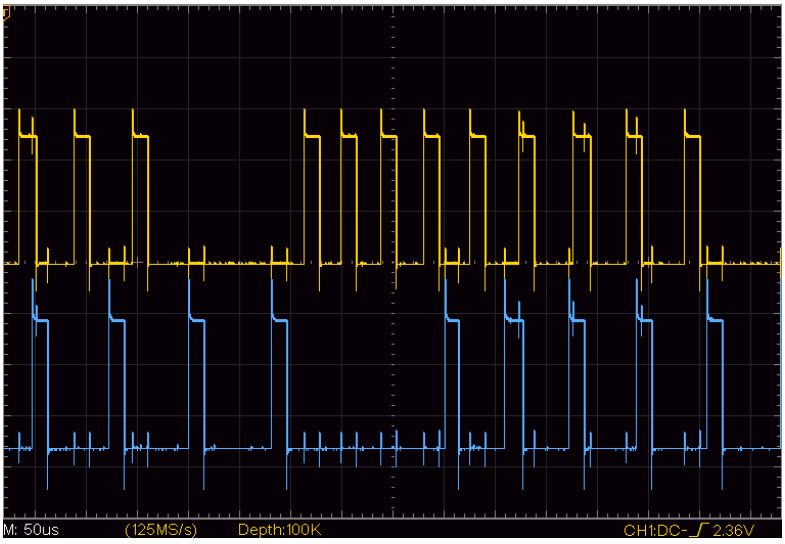
4.1 Integračný test

Pre testovanie boli použité postupne všetky možné hodnoty nibble. Následne boli dĺžky merané pomocou osciloskopu. Pre kontrolu celého rámca boli použité postupne dáta 0–15. Zároveň pri testovaní bola braná do úvahy komunikácia SPI, takže merania prebiehali bez komunikácie SPI, a potom s konštantne prebiehajúcou komunikáciou. Na obrázku 4.1 je zobrazený príklad správy použitej pre testovanie.

Výsledky merania s dĺžkou tick 3 μs ukázali odchýlky v maximálnom rozsahu 0.5 μs, ak neprebiehala žiadna SPI komunikácia. Tieto odchýlky sú prípustné v rámci tolerancie $\pm 20\,\%$.

Pri meraní s dĺžkou tick 3 μs a prebiehajúcej komunikácií boli namerané odchýlky, ktoré ovplyvňovali hodnotu kódovanú jedným nibble. Zároveň paralelná obsluha SPI komunikácie spôsobovala oneskorené nastavenie porovnávacej hodnoty v registri čítača, čo malo za následok veľkosť intervalu o veľkosti čítača.

Tieto odchýlky sa vyskytovali iba pri spracovaní niektorých SPI správ a chyba sa ďalej nešírila, takže zostávajúce správy boli validné.



Obr. 4.1 Meranie správy bez SPI komunikácie s dĺžkou tick 3 μs. Kanál 1 s hodnotami nibble 1–7, kanál 2 s hodnotami 8-15

Rovnaké testy boli vykonané pre maximálnu dĺžku tick 90 μs. Presnosť generovania nebola ovplyvnená zväčšením škálovania hodnôt jednotlivých nibble. Vďaka menšej frekvencií prerušení potrebných pre generovanie protokolu SENT nenastali chyby spojené s interferenciou obsluhy SPI komunikácie.

Integračným testovaním bolo zistené, že presnosť pre požadovanú dĺžku 3 μs je dostačujúca, zároveň bolo zistené, že riešenie generuje validné správy s maximálnou dĺžkou tick 90 μs.

4.2 Jednotkový test

Dáta z osciloskopu boli exportované do formátu csv (čas, napätie), ktorý analyzoval vytvorený skript. Vďaka vzorkovacej frekvencií meraní 1 GSa/s, sú chybné dáta vynechané bez straty presnosti merania. Následne sú dĺžky jednotlivých intervalov počítané pomocou stavového automatu.

Prechody medzi jednotlivými stavmi sú určené nameraným napätím, zároveň je v každom kroku zaznamenaný čas. Ako prechodové podmienky boli zvolené hodnoty 0.5 V a 4.1 V, definované protokolom SENT. K niektorým prechodovým hodnotám bola pripočítaná odchýlkou 0.05 V, kvôli oscilácií nameraných hodnôt okolo limitného napätia. Na obrázku 4.2 sú znázornené jednotlivé stavy automatu.

Výsledkom tejto analýzy je výpočet dĺžok nábežných a dobežných hrán signálu, dĺžky intervalu počas, ktorého je signál v log. 1 a log. 0, a ich priemerné hodnoty. Zároveň je počítaná dĺžka jedného tick a odhad hodnoty jednotlivých nibble. Ďalším z výstupov je dekódovanie dát na jednotlivé nibble podľa preddefinovanej dĺžky tick.

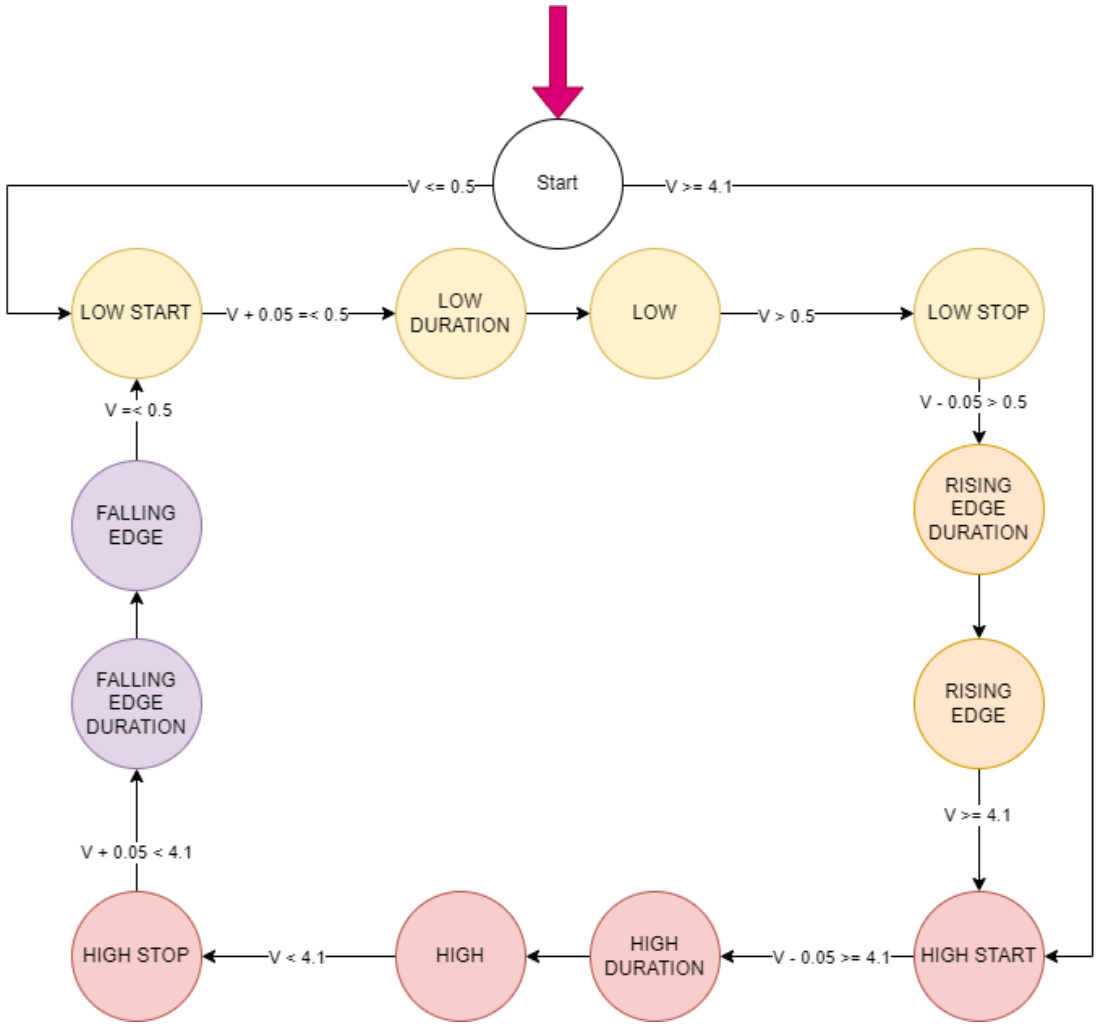
Dĺžka nábežnej hrany je počítaná ako rozdiel časov medzi posledným krokom v stave LOW STOP a posledným krokom v stave HIGH START. Dĺžka dobežnej hrany je rozdiel časov medzi poslednými krokmi HIGH STOP a LOW START. Rovnakým spôsobom sú počítané časy v log 0. medzi stavmi LOW START a LOW STOP resp. v log. 1 HIGH START a HIGH STOP. Po prechode zo stavu START do nového stavu nie je možné určiť korektný začiatok časového intervalu, a preto nie sú všetky prvé namerané intervaly zahrnuté do výsledkov meraní.

Nibble sú počítané ako časový úsek medzi dvoma stavmi RISING EDGE DURATION. Následne je použitý vzťah 1.1 pre výpočet dĺžky tick pre nibble dlhší 100 μs. Pre ďalšie operácie sa používa pevne zadaná dĺžka tick a zároveň nameraná dĺžka tick. Dôvodom dvoch prevodov

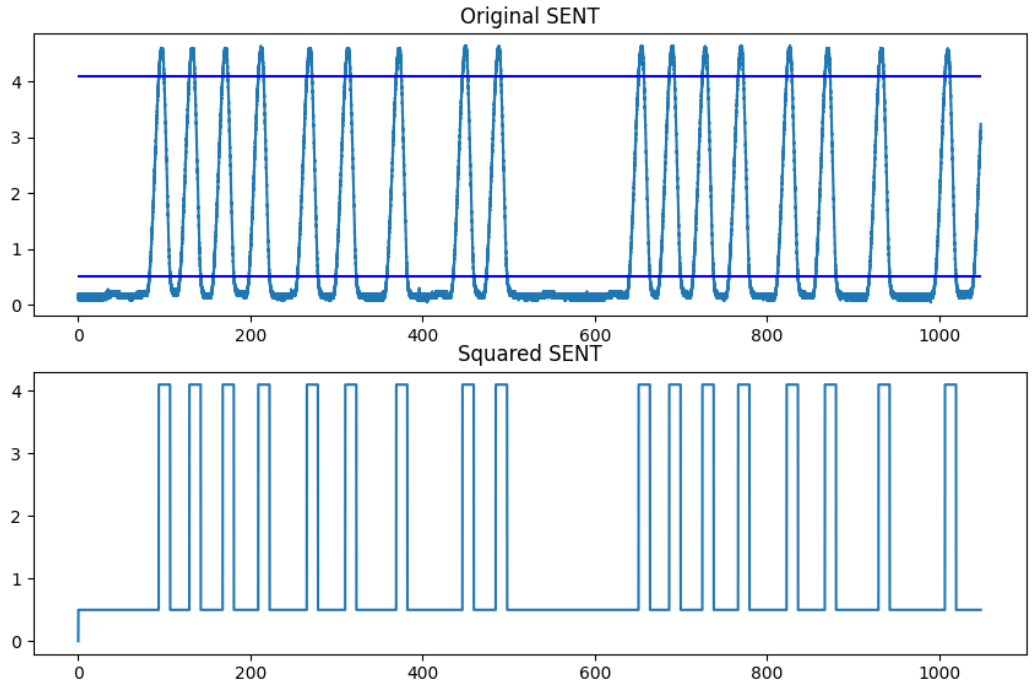
dĺžky nibble na počet tick je zmenšenie chyby zanesenej chybou merania a dôvod, že vstupná meraná správa nezačína vždy synchronizačným pulzom. Následne je z počtu tick odrátaný prefix o dĺžke 12 tick.

Tento skript slúži na orientačné zistenie hodnôt a odchýlok generujúceho zariadenia. Samotné porovnanie hodnôt voči referencií nie je možné automatizovať, pretože norma stanovuje nutnosť zaokrúhľovania časového trvania na konkrétne hodnoty.

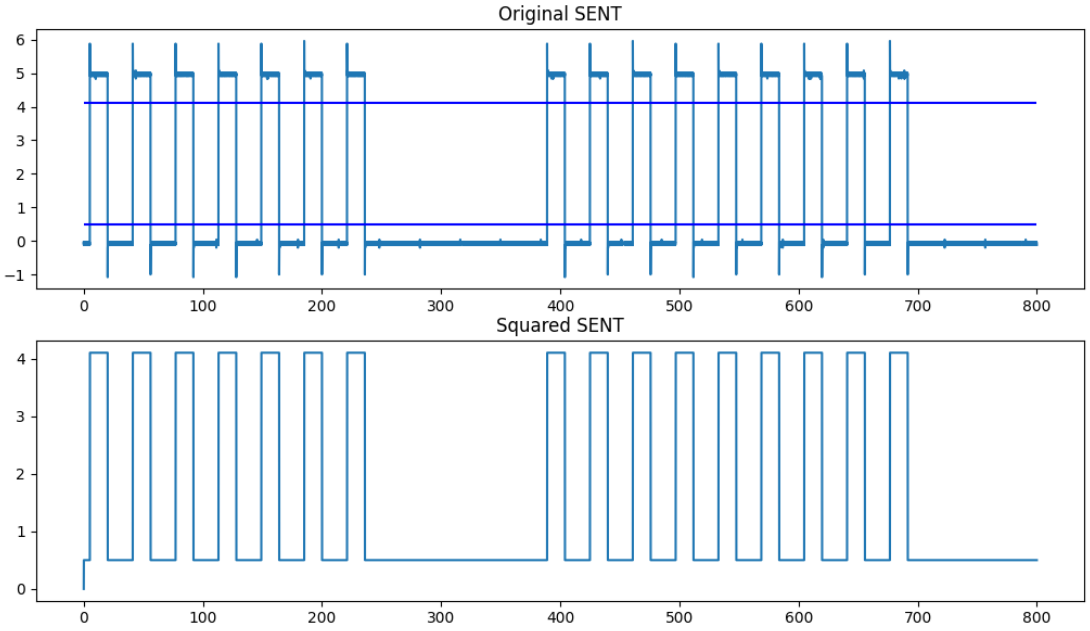
Skript zároveň vytvoril dva grafy, ktoré umožnil interaktívny náhľad hodnôt v čase. Na príklade 4.3 je výstup z merania akceleračného pedálu a výstup 4.4 je výsledkom spracovania výstupu vytvoreného riešenia. Prvým z dvojice výstupných grafov je graf zodpovedajúci reálnemu priebehu analógových hodnôt signálu s pomocnými čiarami indikujúcimi hodnoty 0.5 V a 4.1 V. Druhý graf reprezentuje digitalizovanú podobu protokolárnej komunikácie. Hodnota 0.5 V nastáva v stavoch LOW START až RISING EDGE a hodnota 4.1 V v stavoch HIGH START až FALLING EDGE.



Obr. 4.2 Stavový automat na spracovanie nameraných dát



Obr. 4.3 Spracovanie dát z akceleračného pedálu



Obr. 4.4 Spracovanie dát z výstupu riešenia

Kapitola 5
Záver

Práca sa zaoberala analýzou protokolu SENT a možnosťami jeho generovania. Cieľom bolo vytvoriť zariadenie, ktoré dokáže replikovať komunikáciu medzi akceleračným pedálom a riadiacou jednotkou pomocou tohoto protokolu. Súčasťou návrhu bola analýza existujúcich riešení a analýza dostupných prostriedkov pre riešenie problému emulácie komunikácie založeného na open-source platforme.

Výsledky analýzy poukázali na problémy existujúcich riešení, ktorých implementácia je závislá na použitom hardvéri. Vytvorené riešenie bolo preto založené na platforme Arduino a ESP a využíva moduly na ovládanie vstupu a výstupu, pričom softvérová implementácia je minimálne závislá na hardvérovom riešení.

Riešenie bolo navrhnuté modulárne, tak aby ho bolo v budúcnosti možné rozšíriť o implementáciu ďalších typov správ protokolu SENT a zároveň rozličných parametrov komunikácie, pre potreby vývoja prototypových aplikácií v automotive testovaní.

Výsledné zariadenie umožňuje generovať správy protokolu SENT na základe používateľského vstupu na dvoch výstupných kanáloch. Architektúra je navrhnutá na rozšírenie počtu výstupov. Dodatočný hardvér je možné pripojiť cez zbernicu SPI k existujúcemu riešeniu. Riešenie obsahuje rozhranie pre konfiguráciu obsahu správ pomocou fyzických ovládačov a zároveň vzdialené nastavenie pomocou zbernice CAN.

Vytvorené riešenie bolo testované počas jednotlivých častí vývoja a nakoniec overené ako funkčný celok. Výsledné hodnoty boli overené voči norme ustanovujúcej limitné hodnoty protokolu SENT a zároveň porovnané s nameranými dátami z akceleračného pedálu.

Dodatok A

Mach Systems Email

Zdravím Vás,
Upřesním, že převodník je dostupný ve 3 variantách:
SENT-RS232
SENT-CAN
SENT-USB
Inteligentní filtraci na přijímací straně je myšleno to, že lze nastavit SENT RX kanál takto:
přeposílání všech zpráv na CAN/RS-232/USB
přeposílání s periodou 100 ms
přeposílání s periodou 1 s + v případě změny něčeho ve Fast rámci
Cena převodníku je 6250 CZK + doprava 150 CZK + DPH. (Všechny modely stojí stejně.)
Pro akademické instituce nabízíme 10 % slevu.
V lednu budeme uveřejňovat nové SENT zařízení:
SAE J2716 - Ethernet interface
p/n: MACH-SENT-ETH
Specification:
4 bi-directional SENT channels
Short PWM Code (SPC) support
1 Ethernet port (10/100)
1 USB-C port
1 CAN(/FD) channel
4 Multi-purpose I/Os (analogue input/output, digital input)
MicroSD card slot (stand-alone SENT data logger)
Can also be used as a USB-CAN(/FD) interface
Open communication protocol over ETH, USB, CAN for easy integration
USB-powered or 9 – 30 V DC via terminal block
Aluminium enclosure (80 x 82 x 32 mm)
Use-cases:
SENT¡¿Ethernet
SENT¡¿CAN(/FD)
SENT¡¿USB
SENT¡¿Analogue
Stand-alone SENT data logger
USB-CAN(/FD) interface (can be used simultaneously with other device's functions)
S pozdravem
Miroslav Macháček

Ing. Miroslav Macháček, BEng
jednatel společnosti
e-mail: info@machsystems.cz
mobil: +420 607 568 342
MACH SYSTEMS s.r.o.
www.machsystems.cz : vývoj elektroniky, embedded systémů a IT aplikací/software, průmyslová automatizace, prodej a vývoj SW a HW nástrojů pro CAN/CAN FD (CANopen, SAE J1939), LIN, FlexRay, SENT, Automotive Ethernet

Bibliografia

1. SAE J2716_201604. *SENT Single Edge Nibble Transmission for Automotive Applications*. SAE International, 2016-04. Tech. spr. Dostupné z DOI: 10.4271/J2716_201604.
2. *Pedal Robots* [online]. AB Dynamics [cit. 2023-02-12]. Dostupné z : https://www.abdynamics.com/en/products/track-testing/driving-robots/pedal-robots.
3. TEO, Tit Bin. *Introduction to the Signal Generator* [online]. Keysight Technologies, 2018 [cit. 2023-02-12]. Dostupné z : https://blogs.keysight.com/blogs/tech/rfmw.entry.html/2018/10/30/introduction_to_the-nGk5.html.
4. CHANG, Choon-Hin. *Create Arbitrary Waveform for Use in a Function Generator* [online]. Keysight Technologies, 2021 [cit. 2023-02-12]. Dostupné z : https://blogs.keysight.com/blogs/tech/bench.entry.html/2021/03/18/create_arbitrarywav-wnvm.html.
5. *M8196A 92 GSa/s Arbitrary Waveform Generator* [online]. Keysight Technologies [cit. 2023-02-12]. Dostupné z : https://www.keysight.com/us/en/product/M8196A/92-gsa-s-arbitrary-waveform-generators.html.
6. *Single-Edge Nibble Transmission (SENT) Module* [online]. Microchip Technology Inc., 2014 [cit. 2023-02-12]. Dostupné z : https://ww1.microchip.com/downloads/en/DeviceDoc/70005145b.pdf.
7. *dsPIC33CK64MC105* [online]. Microchip Technology Inc. [cit. 2023-02-12]. Dostupné z : https://www.microchip.com/en-us/product/dsPIC33CK64MC105.
8. *NovaCarts Universal FPGA Device and Connection Module* [datasheet]. MicroNova s .r. o., 2022.
9. *SAE J2716 (SENT) – CAN/RS-232 převodník* [online]. MACH SYSTEMS s. r. o. [cit. 2023-02-12]. Dostupné z : https://www.machsystems.cz/produkty/sbernicove-systemy/brany-prevodniky-sbernic/sae-j2716-sent-can-rs-232-prevodnik.
10. ING. MIROSLAV MACHÁČEK, BEng. *Dotaz na SENT prevodník* [elektronická pošta]. 2022. [cit. 2023-02-12]. Odoslané z: info@machsystems.cz. Osobná komunikácia.
11. *ATmega640/V-1280/V-1281/V-2560/V-2561/V* [online]. Microchip Technology Inc., 2014 [cit. 2023-02-12]. Dostupné z : https://content.arduino.cc/assets/ATmega640-1280-1281-2560-2561-Datasheet-DS40002211A.pdf.
12. *ATmega48A/PA/88A/PA/168A/PA/328/P* [online]. Microchip Technology Inc., 2014 [cit. 2023-02-12]. Dostupné z : https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf.
13. *Arduino Uno Rev3* [online]. Arduino S. r. l. [cit. 2023-02-12]. Dostupné z : https://store.arduino.cc/products/arduino-uno-rev3?queryID=undefined.

14. *SAM3X / SAM3A Series* [online]. Atmel Corporation, 2015 [cit. 2023-02-12]. Dostupné z : http://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-11057-32-bit-Cortex-M3-Microcontroller-SAM3X-SAM3A_Datasheet.pdf.
15. *Arduino Due* [online]. Arduino S. r. l. [cit. 2023-02-12]. Dostupné z : https://store.arduino.cc/products/arduino-due?queryID=2409fca4889dd449a0c9aea62b884add.
16. *Getting Started* [online]. Espressif Systems [cit. 2023-02-12]. Dostupné z : https://docs.espressif.com/projects/arduino-esp32/en/latest/getting_started.html.
17. *ESP32 Wi-Fi & Bluetooth Modules I Espressif* [online]. Espressif Systems [cit. 2023-02-12]. Dostupné z : https://www.espressif.com/en/products/modules/esp32.
18. *Official IoT Development Framework* [online]. Espressif Systems [cit. 2023-02-12]. Dostupné z : https://www.espressif.com/en/products/sdks/esp-idf.
19. *ESP-IDF FreeRTOS (SMP)* [online]. Espressif Systems [cit. 2023-02-12]. Dostupné z : https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-guides/freertos-smp.html.
20. *Development Boards — Espressif Systems* [online]. Espressif Systems [cit. 2023-02-12]. Dostupné z : https://www.espressif.com/en/products/devkits.
21. *Mainstream Performance line, Arm Cortex-M3 MCU with 64 Kbytes of Flash memory, 72 MHz CPU, motor control, USB and CAN* [online]. STMicroelectronics [cit. 2023-02-12]. Dostupné z : https://www.st.com/en/microcontrollers-microprocessors/stm32f103c8.html.
22. *STM32F103C8T6 ARM STM32 kontroler* [online]. GM electronic, spol. s. r. o. [cit. 2023-02-12]. Dostupné z : https://www.gme.cz/v/1508961/stm32f103c8t6-arm-stm32-kontroler.

Obsah priloženého média

- `readme.txt` ..... stručný popis obsahu média
- `src`
  - `impl` ..... zdrojové kódy implementácie
  - `testing` ..... zdrojové kódy testovania
- `text` ..... text práce
  - `thesis` ..... zdrojová forma práce vo formáte LaTeX
  - `thesis.pdf` ..... text práce vo formáte PDF