Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií

Dokumentácia k tímovému projektu

Automatické testovanie v prostredí Internetu vecí
(Dokument k inžinierskemu dielu)

Vedúci projektu: doc. Ing. Tibor Krajčovič, PhD.

Product owner: Ing. Lukáš Ondriga

Členovia tímu: Bc. Tomáš Bujna
Bc. Marián Ján Franko
Bc. Rastislav Kováč
Bc. Igor Labát
Bc. Miroslav Sabo
Bc. Filip Starý
Bc. Stanislav Širka

Akademický rok: 2018/2019

Obsah

1. Úvod

Tento dokument je technická dokumentácia k projektu, obsahujúci časti odkonzultované s Product Ownerom a obsahuje niektoré časti podľa jeho požiadaviek.
Globálne ciele pre ZS:

Cieľom zimného semestra je testovanie analógového a digitálneho signálu odosielaného z BeagleBonu Black na zariadenie COMONEO od spoločnosti Kistler. Cez COMONEO sa vytvorí test, ktorý odosiela konfiguráciu signálov na BeagleBone Black. Na základe konfigurácii BeagleBone Black vygeneruje signály. Tento test taktiež nastaví COMONEO tak, aby očakával konfigurované signály na vstupe a v prípade ak na vstupe sú očakávané signály, test sa považuje za úspešný.

2. Analýza

Pri riešení projektu bolo potrebné naštudovať si viaceré technológie s ktorými sa stretneme počas návrhu a implementácie zadania. Táto kapitola je venovaná práve týmto technológiám. Vysvetľuje, ktoré technológie sme použili, aké predpoklady museli byť splnené, aby sme s týmito technologiami mohli pracovať a taktiež vysvetľuje ako samotné technológie fungujú a načo sa používajú. V nasledujúcich kapitolách sú podorbne analyzované webové servery, vývojvá doska BeagleBone Black, Real Time Unit ako aj samotný prototyp zariadenia, ktoré nám bolo poskytnuté.

2.1 Analýza pre výber webového servera:

Cieľom bolo vybratie webového servera, ktorý by umožňoval komunikáciu medzi ARM a RTU a tiež ARM a RobotFrameworkom.

Akceptačné kritéria product ownera:
Porovnanie 3 web. serverov s kladmi a zápormi.

Lighttpd

Výhody:

- Extremne rýchly pre statický kontent web. serveru
- Schopný zvládnuť tisíce požiadaviek za sekundu
- Minimálna záťaž pamäte/CPU pri behu programu
- Neblokuje I/O operácie lebo beží ako single proces
- Pridávanie funkcionalít cez moduly

Nevýhody:

- Problémy so stabilitou
- Nekompatibilný s niektorými Apache modulmi
- Slabá podpora
- PHP

- Nemá zabudovanú podporu pre WSGI

Nginx

Výhody:

- Dokumentácia a podpora na vysokej úrovni
- Minimálna záťaž pamäte/CPU pri behu programu
- Schopný zvládnuť milióny požiadaviek za sekundu
- Pridávanie funkcionalít cez moduly
- Neblokuje I/O operácie lebo beží ako single proces

Nevýhody:

- Nevhodný pre malé projekty a scenáre s malým počtom požiadaviek
- Oproti Apache menej dostupných modulov pre rozšírenie funkcionalít

Flask

Výhody:

- Dokumentácia a podpora na vysokej úrovni
- Vhodný pre malé projekty a začiatočníkov
- Minimalistický Python framework
- Jednoduchá konfigurácia
- Flexibilný
- Veľa dostupných knižníc
- Nezávisí od ORM, preto je ľahká integrácia s databázami

Nevýhody:

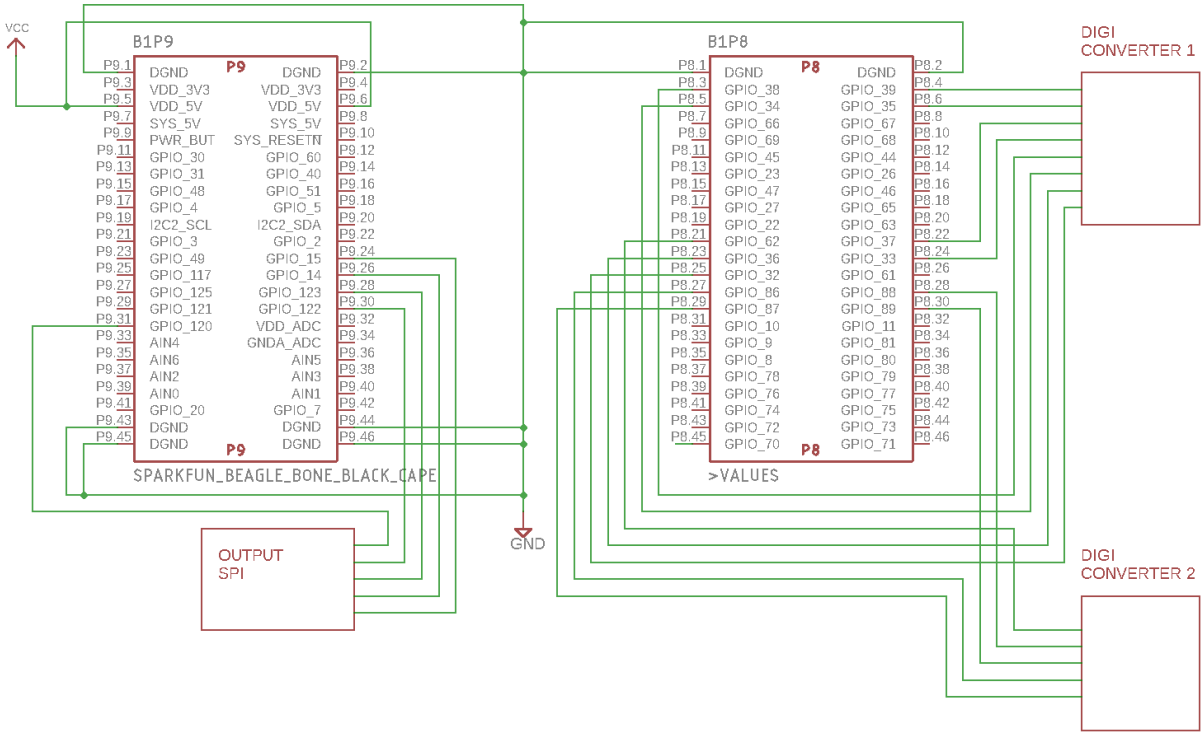
- Ťažšie async. programovanie
- Limitovanie v niektorých funkciách
- Väčšie projekty si vyžadujú dôkladné preštudovanie frameworku

Zhodnotenie:

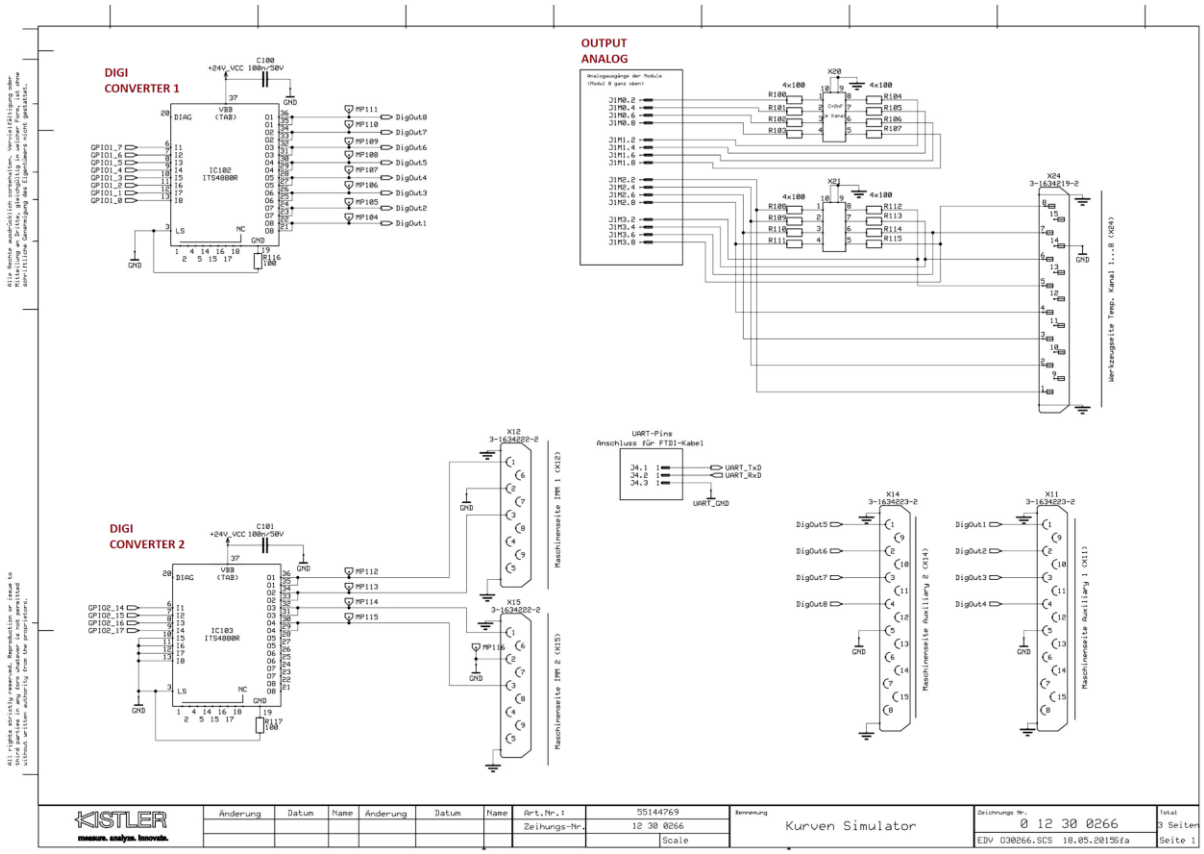
Vybrali sme si web. server Flask najmä kvôli flexibilite a vhodnosti pre malé projekty. Taktiež pri vyberaní zavážila aj skutočnosť, že product owner už na začiatku prvého stretnutia nám odporučil Flask.

2.2 Refaktoring dosky:

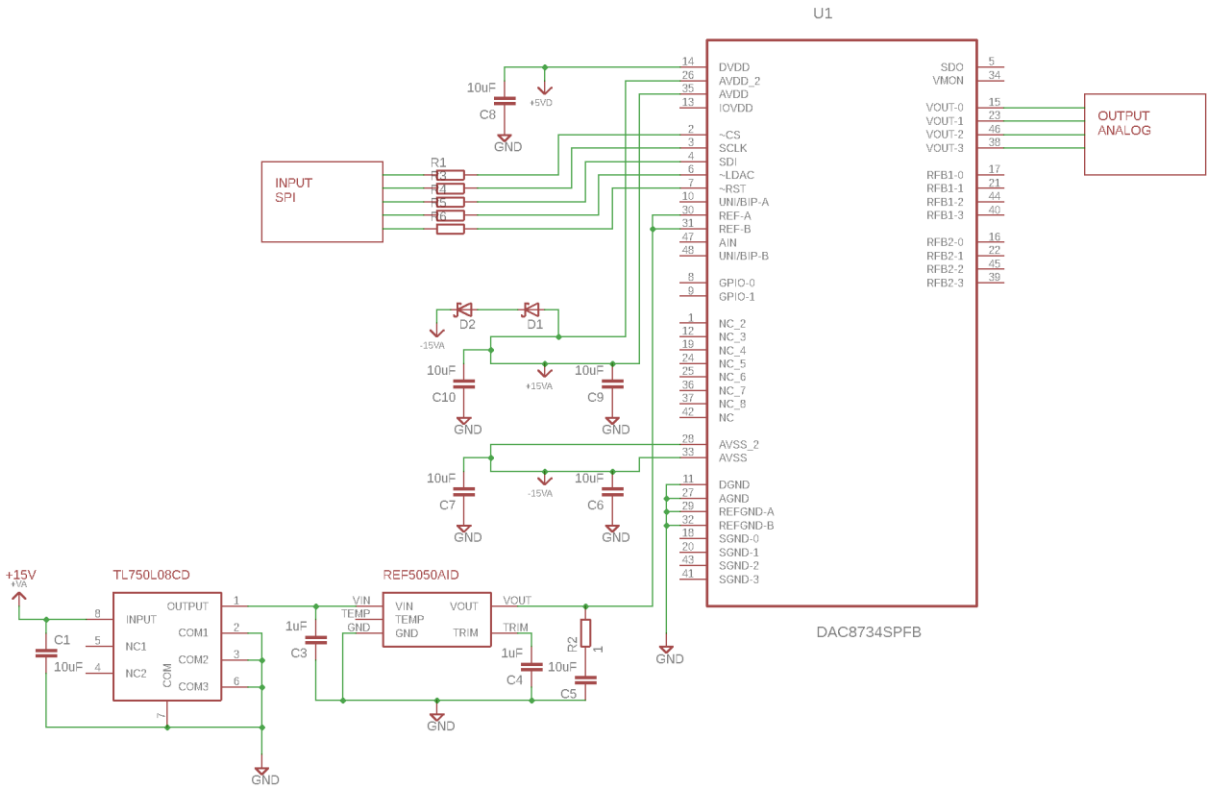
a) pripojenie dosky BeagleBone Black na konektory



b) zapojenie konektorov

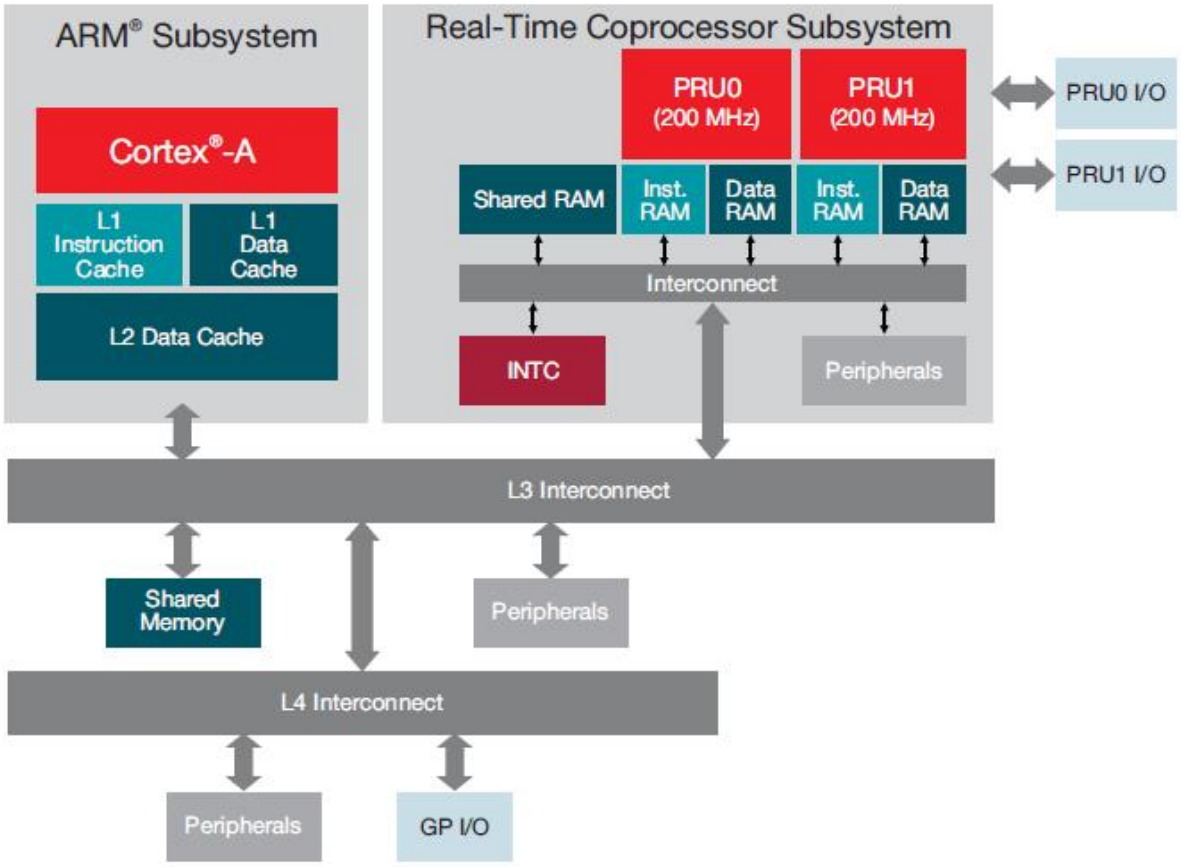


c) refaktoring DAC dosky



2.3 BeagleBone Black Real Time Unit

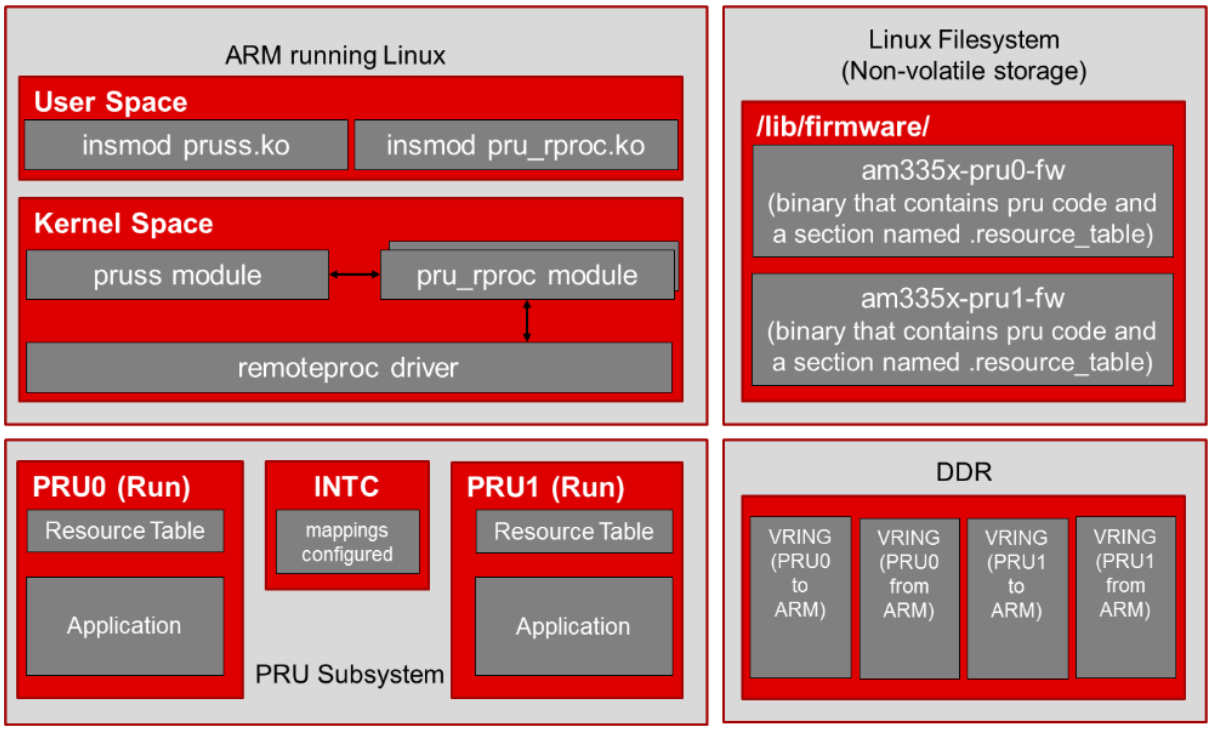
BeagleBone Black je lacný počítač s veľkosťou kreditnej karty, ktorý má dva vstavané mikrokontroléry nazývané PRU. PRU poskytuje schopnosť spracovania v reálnom čase, ktoré chýbajú v systéme Linux.
BeagleBone používa Sitara AM3358, je to procesorový čip ARM bežiaci na frekvencii 1 GHz. Na vykonávanie operácií v reálnom čase, procesor ARM BeagleBone nebude fungovať správne, pretože Linux nie je operačný systém v reálnom čase. Čip Sitara však obsahuje dva 32-bitové mikrokontroléry PRU (programmable real time unit). Použitím PRU je možné dosiahnuť rýchle, deterministické riadenie I / O pinov a zariadení v reálnom čase.



Obr.č.x: Architektúra CPU/PRU

Na používanie PRU je potrebné nainštalovať špeciálny linux. Tento linux musí byť schopný poskytnúť nasledovné služby:

1. Vložiť firmvér do PRU jadra
2. Riadiť vykonávanie programu na PRU (štart, stop, atď)
3. Správu prostriedkov (pamäť, mapovanie prerušení atď)
4. Umožniť odosielanie/prijimanie správ



Na obrázku vyššie sú znázornené štyri bloky:

1. ARM na ktorom beží Linux
2. Linux súborový systém
3. PRU podsystém
4. DDR pamäť

Všetky tieto služby sú poskytované prostredníctvom pru_rproc a rpmsg_pru ovládačov.

V jadre sa nachádza Remoteproc ovládač. Remoteproc je framework, ktorý umožňuje ARM procesoru načítanie firmvéru do jadier PRU, spúšťanie jadier PRU, zastavenie jadier PRU a konfigurovanie prostriedkov (resources), ktoré môžu PRU počas behu potrebovať.
Sysfs rozhranie sa nachádza v používateľskom priestore, pomocou neho vieme spustiť alebo zastaviť PRU jadrá a načítať firmvér.
Binárne firmvér súbory sa nachádzajú v /linux/filesystem/ adresáry.

Postup načítávania firmvéru a spustenia programu:

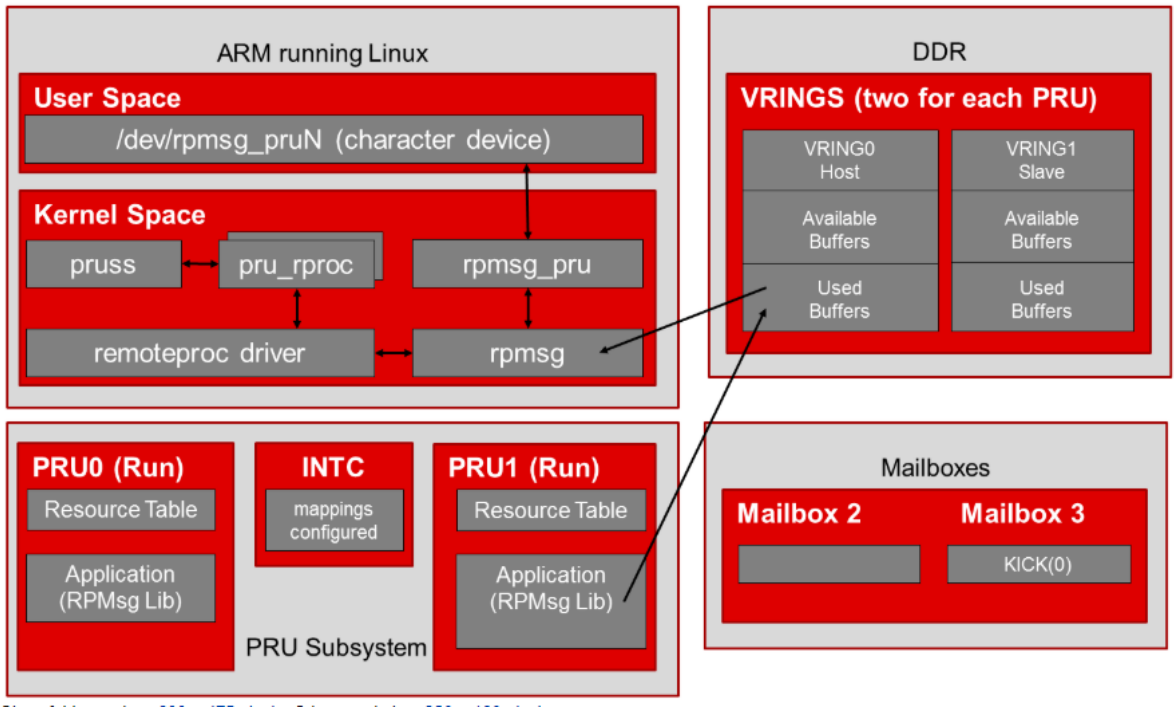
1. Pru_rproc modul musí predtým, ako čo načíta firmvér do PRU zistiť, či sa firmvérové binárne súbory nachádzajú v /lib/firmware/ .Modul pru_rproc tiež analyzuje binárne súbory firmvéru a hľadá sekciu s názvom .resource_table. Sekcia .resource_table firmvéru špecifikuje systémové prostriedky, ktoré PRU budú potrebovať počas vykonávania programu.
2. Modul pru_rproc konfiguruje všetky zdroje, ktoré firmvér potrebuje.V tomto prípade to zahŕňa vytvorenie vrstiev v pamäti DDR na komunikáciu, ako aj nastavenie mapovania prerušenia v module INTC PRU subsystému.
3. Modul pru_rproc potom načíta binárne súbory do inštrukčnej pamäti PRU a taktiež skopíruje resource table do dátovej pamäti PRU.

4. Keď je všetko nakonfigurované a program sa nachádza v pamäti, modul pru_rproc spustí vykonanie programu na PRU.

Príklad načítania firmvéru do PRU a spustenia programu:

1. echo 'am335x-pru0-fw' > /sys/class/remoteproc/remoteproc1/firmware
2. echo 'start' > /sys/class/remoteproc/remoteproc1/state

RPMsg je mechanizmus posielania správ, ktorý požaduje prostriedky prostredníctvom remoteproc a beží nad virtio frameworkom. Zdieľané vyrovnávacie pamäte (buffer) sú požadované prostredníctvom resource_table a poskytované remoteproc modulom počas načítania firmvéru do PRU. Zdieľané vyrovnávacie pamäte sa nachádzajú vnútri vring dátovej štruktúry v pamäti DDR. Každé PRU jadro má k dispozícii dve vring, jedna sa používa pre správy prenesené na ARM a druhá sa používa pre správy prijaté z ARM. Systémové mailboxy sa používajú na oznamovanie jadrom (ARM alebo PRU), keď nové správy čakajú v zdieľaných vyrovnávacích pamätiach.
K dispozícii sú dve softvérové implementácie RPMsg. Na strane ARM Linuxu je komunikácia RPMsg prijatá v priestore jadra. Je poskytnutý modul rozhrania (rpmsg_pru), v používateľskom priestore, takže používatelia môžu zapisovať / čítať do / zo znakového zariadenia na odosielanie / prijímanie správ do / z PRU. Na strane PRU je k dispozícii knižnica RPMsg v PRU softvérovom balíku podpory (Software support package), ktorej cieľom je umožniť prepojenie, kde používateľský kód môže jednoducho volať funkcie pru_rpmsg_receive a pru_rpmsg_send, aby komunikoval s jadrom ARM.



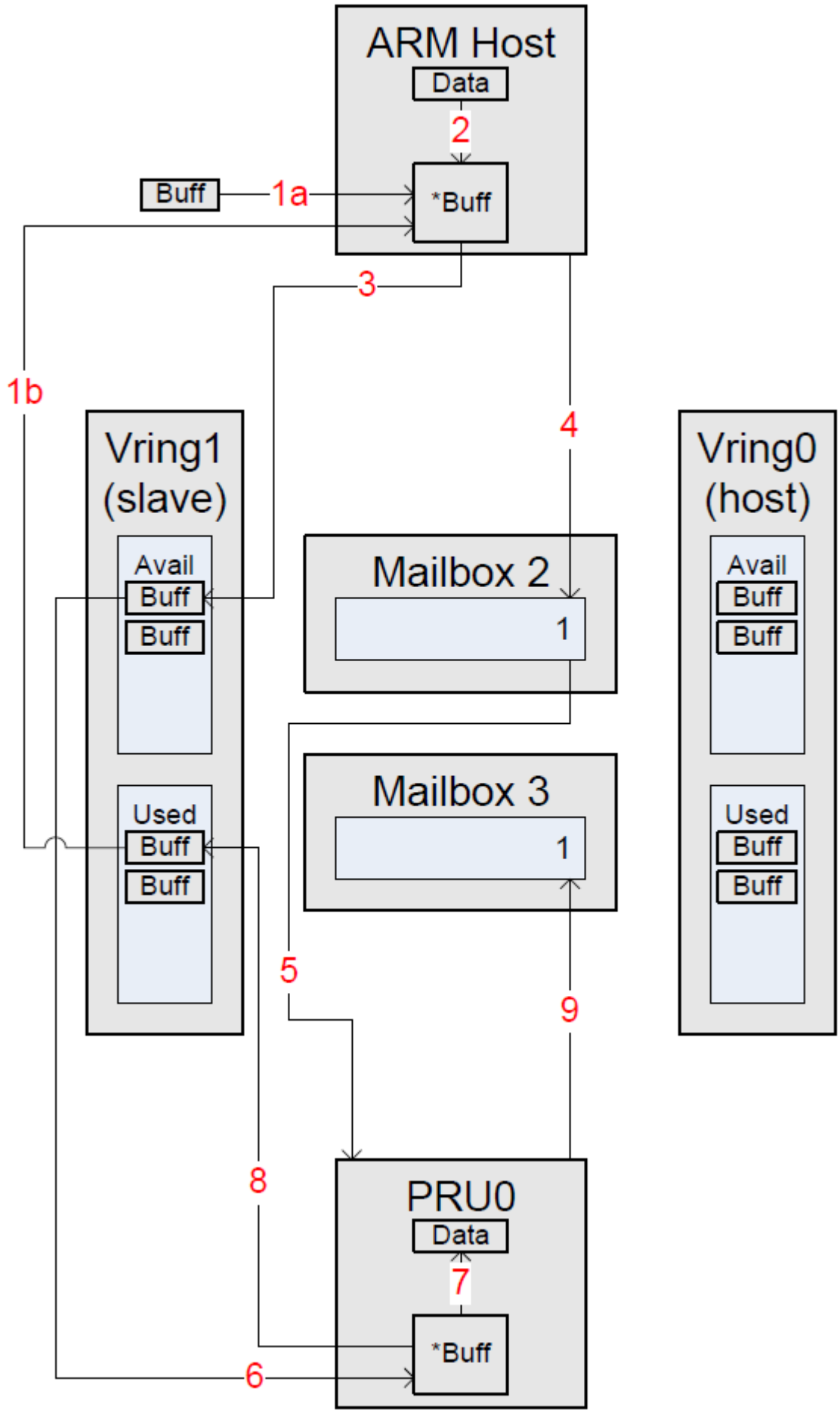
ARM - PRU správy

Na obrázku nižšie je znázornený proces posielania správ z ARM na PRU.

ARM:

1a. Alokuj vyrovnávaciu pamäť
alebo
1b. Získaj použitú vyrovnávaciu pamäť z slave Vring
2. Skopiruj dáta do vyrovnávacej pamäte
3. Pridaj naplnenú vyrovnávaciu pamäť do zoznamu dostupných v slave Vring
4. Naštartuj slave Vring zapísaním jeho indexu (1) a odoslaním správy do Mailbox 2

PRU:
5. V Mailbox 2 sa nájde správa s indexom Vring-u (1), čo znamená že dáta sú pripravené na príjem.
6. Ziskaj vyrovnávaciu pamäť z slave Vring
7. Skopiruj dáta z vyrovnávacej pamäte z kroku 2.
8. Pridaj vyrovnávaciu pamäť do zoznamu použitých v slave Vring.
9. Naštartuj slave Vring zapísaním jeho indexu (1) do správy v Mailbox 3.


ARM Host
Data
2
Buff
1a
*Buff
3
1b
4
Vring1
(slave)
Avail
Buff
Buff
Used
Buff
Buff
Mailbox 2
1
Mailbox 3
1
Vring0
(host)
Avail
Buff
Buff
Used
Buff
Buff
5
9
8
PRU0
Data
7
*Buff
6

PRU - ARM správy

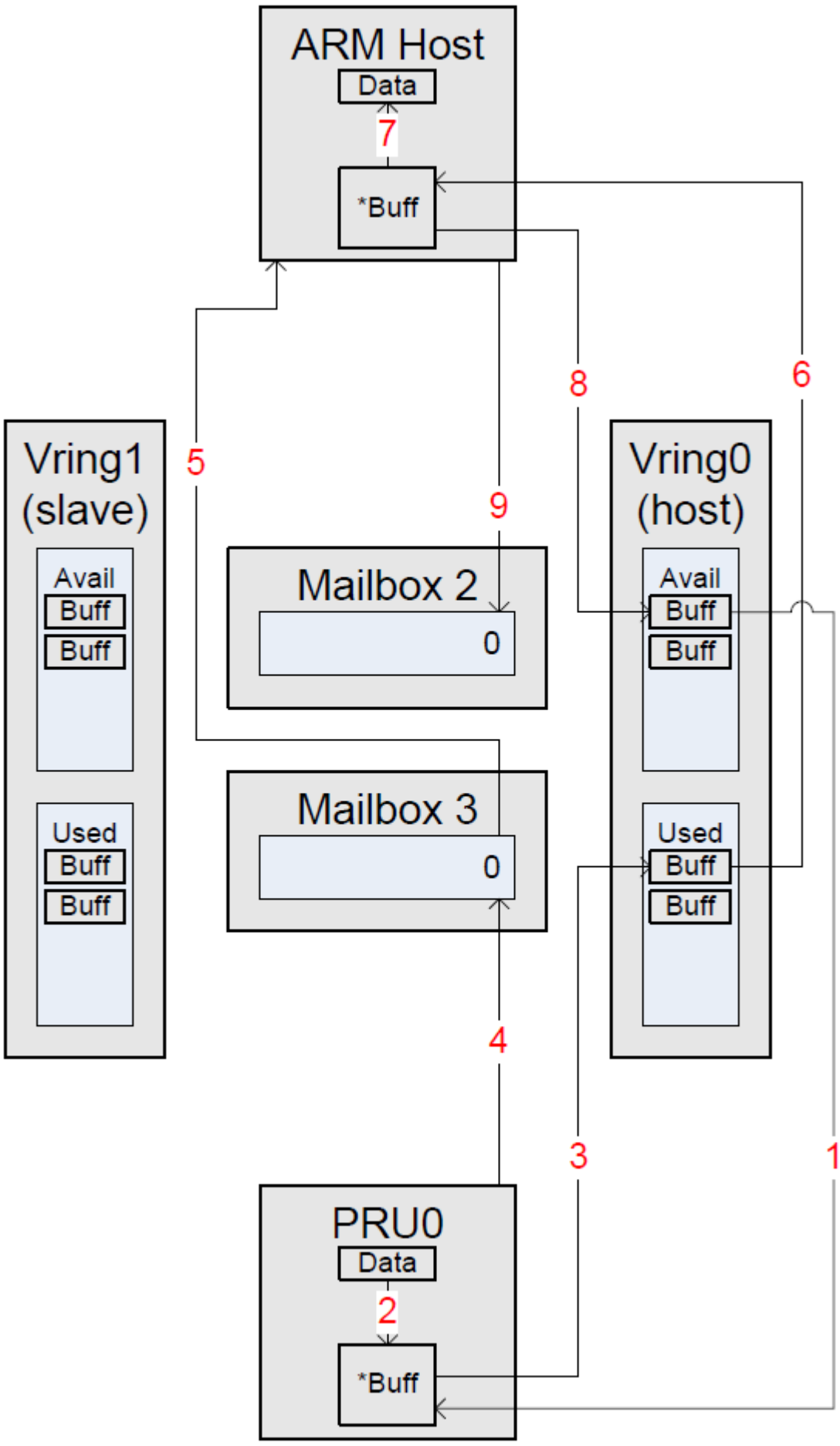
Na obrázku nižšie je znázornený proces posielania správ z PRU do ARM.

PRU:

1. Získaj voľnú vyrovnávaciu pamäť z host Vring
2. Skopiruj dáta na prenos do vyrovnávacej pamäti
3. Pridaj naplnenú vyrovnávaciu pamäť do zoznamu použitých v host Vring
4. Naštartuj host Vring zapísaním jeho indexu (0) do správy v Mailbox 3

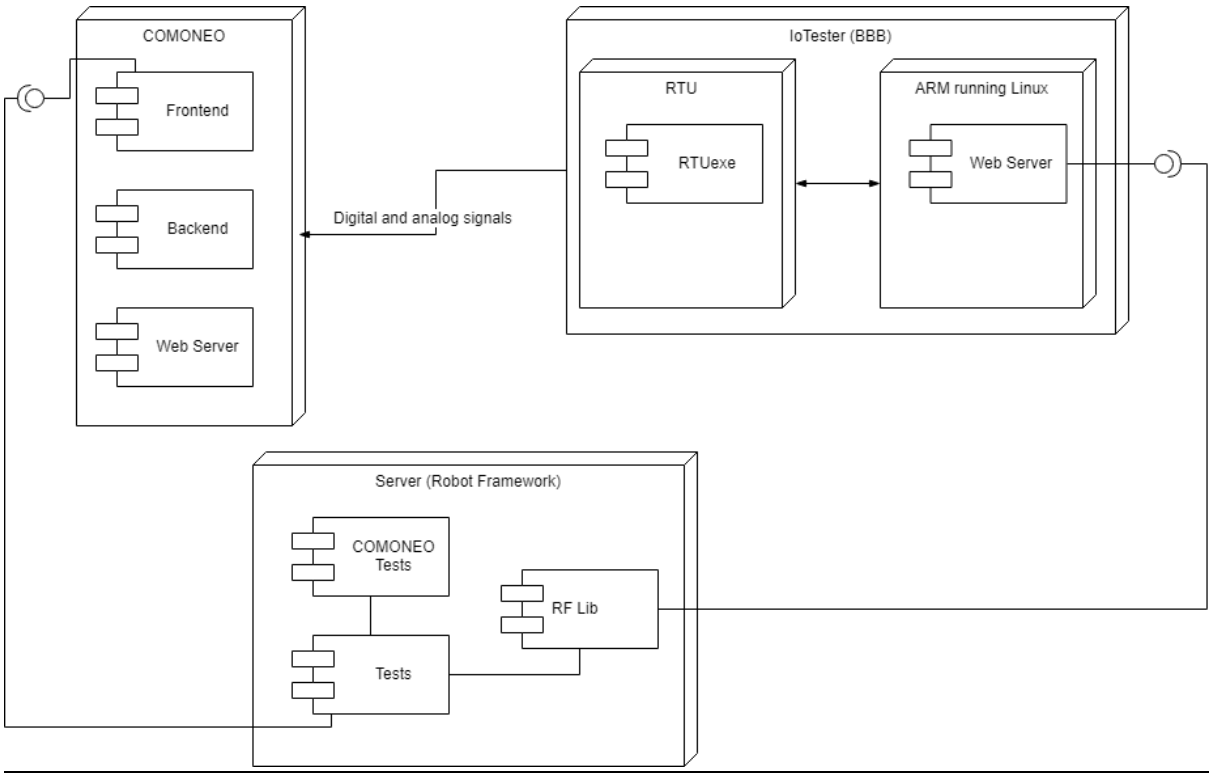
ARM:

1. Prerušenie indikuje, že Mailbox 3 má správu od Vring (0). To oznamuje ARM procesoru že má dostupné dáta na príjem
2. Získaj použitú vyrovnávaciu pamäť z host Vring
3. Skopiruj dáta na príjem z vyrovnávacej pamäti z kroku 2.
4. Pridaj prázdnu vyrovnávaciu pamäť do zoznamu dostupných v host Vring
5. Naštartuj host Vring zapísaním jeho indexu (0) do správy v Mailbox 2.


ARM Host
Data
7
*Buff
8
6
Vring1
(slave)
5
9
Vring0
(host)
Avail
Buff
Buff
Mailbox 2
0
Avail
Buff
Buff
Mailbox 3
0
Used
Buff
Buff
Used
Buff
Buff
4
3
1
PRU0
Data
2
*Buff

3. Návrh a implementácia

3.1 Architektúra systému



Architektúra systému pozostáva z trocha dôležitých častí:

1. COMONEO
2. Server (Robot Framework)
3. IoTester (BBB)

COMONEO je zariadenie od spoločnosti Kistler, ktoré umožňuje viacero funkcionalít. Z pohľadu nášho systému, toto zariadenie očakáva digitálne alebo analógové signáli na svojom vstupe od zariadenia IoTester (BBB - Beaglebone Black). Taktiež na tomto zariadený beží rozhranie pomocou ktorého nastavujeme testy (očakávané signály) na vstupe COMONEO za pomoci Robot Frameworku. Testy na tomto rozhraní sa automaticky nastavujú na základe zadefinovaných testov na serveri, kde je Robot Framework. Súčasťou týchto testov na serveri sú aj konfigurácie signálov, ktoré sa odosielajú na IoTester, ktorý vygeneruje signály na základe získanej konfigurácii. Vygenerované signály z IoTestera sa musia zhodovať s očakávanými vstupnými signálmi na COMONEO, aby test bol úspešný.

3.2 REST API:

Z pohľadu výsledného produktu ide o programové rozhranie, ktoré zabezpečuje komunikáciu centrálnej testovacej jednotky Robot Framework so vzdialeným zariadením BeagleBone, ktoré pomocou príslušného modulu spoločnosti Kistler, komunikuje s monitorovacím zariadením ComoNeo, taktiež od spoločnosti Kistler.

Na základe aktuálneho priebehu je žiaduce, aby REST API umožňovalo za pomoci POST REQUESTOV odosielanie jednoduchého digitálneho signálu(0/1). Preto môžeme hovoriť o prototype. V budúcnosti bude potreba toto rozhranie rozšíriť aj o zasielanie zložitejších signálov, čo sa podarilo načrtnúť už v doterajšom priebehu. Aktuálne REST API umožňuje zasielanie aj zložitejších signálov a to za pomoci dát zasielaných vo formáte JSON, čo môžeme považovať za výhodu a do budúcna.

Vo všeobecnosti ide o program napísaný v jazyku Python, ktorý pomocou knižnice Flask a knižnice JSON, umožňuje prijímanie a zapisovanie dát. Pre funkčnosť tohto rozhrania boli následne vytvorené akceptačné testy využitím Robot frameworku, kde Robot odošle za pomoci rozhrania testovacie dáta, pričom z hľadiska testov je požiadavka aby rozhranie zaslané dáta v nezmenenom formáte vrátilo. Ak rozhranie tieto dáta vráti ako odpoveď, vieme že rozhranie je aktívne.

4. Testovanie

4.1 Robot Framework testing:

Pre spustenie testov je potrebné nainštalovať a nastaviť nasledovné časti:

1. Inštalácia Pyhton (https://www.python.org/downloads)
2. Pridanie cesty do systémovej premennej Path (..\PythonXX;..\PythonXX\Scripts)
3. Inštalácia robotframework pomocou cmd (pip install robotframework)
4. Inštalácia SeleniumLibrary pomocou cmd (pip install robotframework-seleniumlibrary)
5. Stiahnuttie chromedriver (http://chromedriver.chromium.org/downloads)
6. Vloženie chromedriver do priečinka (..\PythonXX\Scripts)
7. Inštalácia PyCharm (https://www.jetbrains.com/pycharm/download)
8. Inštalácia pluginu IntelliBot (v nastaveniach PyCharm)
9. Inštalácia requests pre python kvôli využitiu REST API (pip install requests)

Zatiaľ sme pracovali na dvoch hlavných funkcionalitách:

1. Test, ktorý testuje digitálny input na webovej aplikácii zariadenia ComoNeo. Tento test si otvori webovú aplikáciu, dostane sa k digitálnym input-om a prečíta hodnotu, aké je tam zobrazená. Hodnota predstavuje, či bol digitálny vstup 1/0.
2. Aby sme vedeli poslať digitálny input, bolo potrebné implementovať špeciálnu knižnicu, ktorá vedela pomocou nami implementovaného REST API odoslať samotný digitálny input. Ten sa opäť testuje vo webovej aplikácii.

Samotné testovanie a výsledné testy sú verziované a ukladané na gitlabe spoločnosti Kistler, nakoľko sa pri testoch využívajú citlivé informácie.