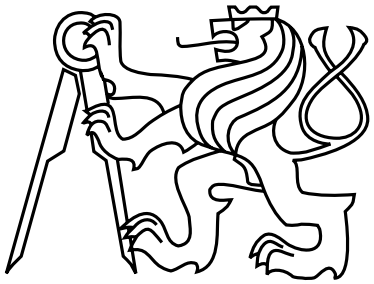
České vysoké učení technické v Praze
Fakulta elektrotechnická
Katedra počítačů



Bakalářská práce

Kalibrační a ovládací software sítě částicových pixelových detektorů umístěných uvnitř experimentu ATLAS na LHC v CERN

Jakub Begera

Vedoucí práce: Ing. Štěpán Polanský

Studijní program: Otevřená informatika, Bakalářský

Obor: Softwarové systémy

27. května 2016

České vysoké učení technické v Praze
Fakulta elektrotechnická

katedra počítačů

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Student: **Jakub Begera**

Studijní program: Otevřená informatika
Obor: Softwarové systémy

Název tématu: **Kalibrační a ovládací software sítě částicových pixelových detektorů umístěných uvnitř experimentu ATLAS na LHC v CERN**

Pokyny pro vypracování:

Cílem této bakalářské práce je návrh a implementace software pro energetickou kalibraci částicových pixelových detektorů pracující v Time-Over-Treshold (TOT) režimu a pro řízení sítě těchto detektorů (Atlas TPX), instalované uvnitř experimentu ATLAS na LHC v CERN.
Kalibrační část:
Vstupními daty jsou matice hodnot získané měřením dvou až deseti energií rentgenového záření. Pro každý pixel je třeba vytvořit pomocí analytických a statistických metod uvedených v [2] parametry popisující kalibrační funkci, udávající vztah mezi TOT a energií.
Řídící část:
Software bude umožňovat ovládat síť hybridních částicových pixelových detektorů a bude umožňovat následující:
Nastavovat parametry akvizice snímků (akviziční čas, počet snímků, mód)
Nastavovat HW parametry detektoru (bias, TPX clock, DACs)
Vyčítání a ukládání dat

Seznam odborné literatury:

[1] X. Llopart, R. Ballabriga, M. Campbell, L. Tlustos, W. Wong, Erratum to “Timepix, a 65 k programmable pixel readout chip for arrival time, energy and/or photon counting measurements” [Nucl. Instr. and Meth. A. 581 (2007) 485–494], Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment, Volume 585, Issues 1–2, 21 January 2008, Pages 106-108, ISSN 0168-9002, http://dx.doi.org/10.1016/j.nima.2007.11.003.
[2] Jan Jakubek, Precise energy calibration of pixel detector working in time-over-threshold mode, Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment, Volume 633, Supplement 1, May 2011, Pages S262-S266, ISSN 0168-9002, http://dx.doi.org/10.1016/j.nima.2010.06.183.

Vedoucí: Ing. Štěpán Polanský

Platnost zadání: do konce letního semestru 2016/2017

prof. Ing. Filip Železný, Ph.D.
vedoucí katedry

prof. Ing. Pavel Ripka, CSc.
děkan

V Praze dne 11. 1. 2016

Poděkování

Na tomto místě bych rád poděkoval svojí rodině a přátelům, kteří mě během studia podporovali. Také bych rád poděkoval kolegům z ÚTEF ČVUT v Praze, za možnost podílet se na projektu ATLAS TPX, zejména pak vedoucímu této bakalářské práce Ing. Štěpánu Polanskému, za předání mnoha zkušenosti, rad a také času, který mi věnoval. Dále bych chtěl poděkoval Ing. Janu Jakůbkovi, Ph.D. za uvedení do problematiky energetické kalibrace a za pomoc při implementaci kalibračního softwaru.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze dne 27. 5. 2016 .....

Abstract

During the year 2014, a network of 16 Timepix based hybrid particle pixel detectors were installed at various positions within the ATLAS experiment at LHC at CERN. The goal of this bachelor thesis is to develop software for controlling and energy calibration of this network. The control software provides control of these detectors independently, such as settings of measurement parameters, data acquisition control, data readout, data processing, etc. This software provides `JSON REST API` server for remote control and for transfer of status informations of this network to CERN's systems. Calibration software provides to user to pass through the calibration process from calibration spectra assembly, to spectra analysis and creation of the individual calibration points, to make the calibration functions for the individual pixels of the detector.

Abstrakt

V roce 2014 byla na různé pozice experimentu ATLAS na LHC v CERN nainstalována síť 16 hybridních částicových pixelových detektorů typu Timepix. Cílem této bakalářské práce je vyvinout software pro řízení a energetickou kalibraci této sítě. Řídící software umožňuje nezávislé ovládání detektorů, zejména pak nastavení měřících parametrů, řízení akvizice dat, vyčítání a zpracování dat apod. Tento software poskytuje `JSON REST API` server pro své vzdálené řízení a přenáší informace o stavu celé sítě systémům CERNu. Kalibrační software umožňuje uživateli průchod celým procesem zpracování kalibračních dat, od sestavení spekter z naměřených dat, přes jejich analýzu a vytvoření jednotlivých kalibračních bodů, až po výpočet parametrů kalibrační funkce pro jednotlivé pixely detektoru.

Obsah

1 Úvod **1**
- 1.1 Motivace . . . 1
- 1.2 Struktura dokumentu . . . 1

2 Polovodičové pixelové detektory ionizujícího záření rodiny Medipix **3**
- 2.1 Princip detekce . . . 4
- 2.2 Detektory rodiny Medipix . . . 5
- 2.3 Provozní módy detektoru Timepix . . . 6
- 2.4 FITPix . . . 7
- 2.5 Pixelman . . . 8

3 Energetická kalibrace **9**
- 3.1 Motivace . . . 9
- 3.2 Přehled kalibračních metod . . . 10
  - 3.2.1 Kalibrace detektorů za použití rentgenového záření . . . 10
  - 3.2.2 Kalibrace detektorů pomocí LED diod . . . 11
- 3.3 Software pro kalibraci detektorů za použití rentgenového záření . . . 13
  - 3.3.1 Vstupní data kalibrace . . . 13
  - 3.3.2 Analýza spekter . . . 15
  - 3.3.3 Vytvoření kalibrační funkce . . . 16
  - 3.3.4 Dodatečné úpravy . . . 18

4 ATLAS TPX **21**
- 4.1 ATLAS MPX . . . 22
  - 4.1.1 Hardwarová a softwarová architektura sítě ATLAS MPX . . . 23
- 4.2 Hardwarová architektura sítě ATLAS TPX . . . 24
- 4.3 Softwarová architektura sítě ATLAS TPX . . . 26
- 4.4 Řídící software a jeho implementace . . . 28
  - 4.4.1 Řízení detektorů . . . 29
    - 4.4.1.1 Komunikační protokol . . . 29
    - 4.4.1.2 Popis příkazů komunikačního protokolu . . . 29
    - 4.4.1.3 Synchronní a asynchronní příkazy komunikačního protokolu . . . 34
    - 4.4.1.4 Implementace modulu pro řízení detektorové sítě na straně serveru . . . 35
    - 4.4.1.5 Emulátor detektoru . . . 38

4.4.2 REST API server 39
4.4.2.1 Konfigurace a spuštění serveru 40
4.4.2.2 Metody poskytované serverem 41
4.4.2.3 Implementace serveru 41
4.4.3 Zpracování a ukládání dat 42
4.4.3.1 Datový server 44

5 Závěr **47**

A Seznam použitých zkratek **53**

B Konfigurační soubor ATLAS TPX serveru **57**

C Obsah přiloženého CD **61**

Seznam obrázků

2.1 Princip detekce ionizujícího záření (převzato z [12]), kde červená šipka je procházející částice, žlutě jsou znázorněny elektrony, modře díry. Elektronika každého pixelu zpracovává napěťový pulz. 4
2.2 Struktura hybridního polovodičového pixelového detektoru rodiny Medipix (převzato z [12]) 5
2.3 Zpracování signálu z pohledu módu pixelu (převzato z [12]) 6
2.4 FITPix (vlevo zařízení FITPix, vpravo DPS zařízení FITPix) 7

3.1 Kalibrační funkce (převzato z [6]) udává závislost mezi energií a TOT. Vzniká proložením naměřených kalibračních bodů pomocí funkce 3.1. Tato funkce vznikla složením hyperboly (popisující nelineární oblast nižších energií) a přímky (pro oblast s vyšší energií). 9
3.2 Spektrum TOT hodnot jednoho pixelu s proložením Gaussovou funkcí, sečtenou s Gaussovou chybovou funkcí (tzv. error funkce). Zdrojem rentgenové fluorescence byla měď. 10
3.3 Snímek z detektoru, ozařovaném modulem s LED diodami 12
3.4 Screenshot kalibračního softwaru - volby zpracování v rámci kalibračního procesu 13
3.5 Screenshot kalibračního softwaru, zobrazující složené spektrum pro gama emisi americia (pravý červený pík) a fluoresceni india (levý zelený pík), proložené funkcí 3.2. Levou část spekta tvoří fotony vzniklé Comptonovým rozptylem. . 14
3.6 Screenshot kalibračního softwaru - Measurement manager 14
3.7 Screenshot kalibračního softwaru - maska špatných pixelů 16
3.8 Screenshot kalibračního softwaru - kalibrační funkce 17
3.9 Screenshot kalibračního softwaru - vizualizace výsledných parametrů kalibrace v řezu přes řádky detektoru 19

4.1 ATLAS TPX - přehledem rozmístění detektorů v experimentu ATLAS 21
4.2 ATLAS TPX detektor - vrstvy a rozmístění konvertorů 22
4.3 Snímek z ATLAS MPX detektoru s výřezem zachycených částic (převzato z [2]) 23
4.4 Fotografie znázorňující Medipix2 detektor s neutronovými konvertory (**a**) a struktura neutronových konvertorů (**b**)) [19] 23
4.5 ATLAS MPX - řídící aplikace (převzato z [17]) 24
4.6 ATLAS TPX - diagram hw komponent, kde TPX01 až TPX15 jsou detektory umístěné v prostorách UX15 a zbytek sítě (vyčítací elektronika, propojená pomocí switche se servery TPX a DCS) umístěné v prostorách USA15 25

4.7 ATLAS TPX - fotografie hw komponent 26
4.8 ATLAS TPX - diagram softwarových komponent 27
4.9 Cluster analýza - 6 základních typů clusterů (převzato z [17]) 28
4.10 Příklad použití komunikačního protokolu 35
4.11 Diagram tříd modulu pro ovládání detektorů 36
4.12 Emulátor detektoru - sekvenční diagram připojení klienta a pořízení snímku . 39
4.13 BPMN inicializace REST API serveru 42
4.14 Diagram tříd komponenty pro ukládání dat ATLAS TPX serveru 44

C.1 Obsah přiloženého CD 62

Seznam tabulek

4.1 Komunikační protokol - struktura paketů z pohledu serveru 29
4.2 Komunikační protokol - přehled příkazů 30

Seznam zdrojových kódů

B.1 Konfigurační soubor ATLAS TPX serveru 59

Kapitola 1

Úvod

Tato bakalářská práce se zabývá návrhem a implementací software pro ovládání a energetickou kalibraci sítě hybridních částicových pixelových detektorů umístěných uvnitř experimentu ATLAS na urychlovači LHC v CERN. Tento projekt je nazývá ATLAS TPX (viz kapitola 4).

Jelikož proces kalibrace je zcela nezávislý na následném řízení těchto detektorů, je software členěn na dvě nezávislé části, a to na energetickou kalibraci a řízení sítě ATLAS TPX.

Kalibrační software umožňuje průchod procesem zpracování kalibračních dat, od sestavení spekter z naměřených snímků, přes jejich analýzu a vytvoření jednotlivých kalibračních bodů, až po sestavení kalibrační funkce pro jednotlivé pixely detektoru.

Řídící software sítě ATLAS TPX slouží pro nezávislé ovládání funkce těchto detektorů. Umožňuje nastavování různých parametrů detektorů, řízení akvizice snímků, vyčítání naměřených dat a jejich ukládání, ev. jejich odesílání datovému serveru pro jejich hlubší analýzu a uložení do perzistentní centrální CERNské databáze (tzv. `EOS`). Zároveň tento řídící software poskytuje `JSON REST API` pro své vzdálené řízení a také pro poskytování stavových informací o teto sítí CERNu.

1.1 Motivace

Ionizující záření je spjato s naším světem už od počátku jeho existence. Jeho studium začalo koncem 19. století a pomáhá nám pochopit podstatu hmoty, její interakce s prostředím a další vlastnosti. Tyto poznatky našly své uplatnění v mnoha oborech, jako například ve zdravotnictví, defektoskopii, energetice a v mnoha dalších. Spolu s rostoucími znalostmi o ionizujícím záření a s technologickým pokrokem se rozvíjela i detekční technika, která za poslední století prodělala veliký posun. Od prvních bublinových komor, až po polovodičové pixelové detektory, kterými se tato práce zabývá.

1.2 Struktura dokumentu

Kapitola 2 - Polovodičové pixelové detektory ionizujícího záření rodiny Medipix: V této kapitole jsou představeny hybridní částicové pixelové detektory rodiny Medipix, je-

jich rozdělení, principy detekce, provozní módy a další vlastnosti detektorů, relevantní k této práci.

Kapitola 3 - Energetická kalibrace: Tato kapitola pojednává o metodách energetické kalibrace částicových pixelových detektorů rodiny Medipix, pracujících v `Time-Over-Treshold` módu a také je zde zmíněna implementace jedné z těchto metod pro účely kalibrace detektorové sítě ATLAS TPX.

Kapitola 4 - ATLAS TPX: V rámci této kapitoly je popsán návrh softwarové a hardwarové architektury detektorové sítě ATLAS TPX a také implementace řídícího softwaru ATLAS TPX serveru.

Kapitola 2

Polovodičové pixelové detektory ionizujícího záření rodiny Medipix

Ionizující záření je lidskými smysly nedetekovatelné. Tento fakt dal vzniku detekční technice a metodám toto záření měřit. Tato kapitola je věnována pokročilé instrumentaci pro detekci ionizujícího záření - hybridním polovodičovým pixelovým detektorům.

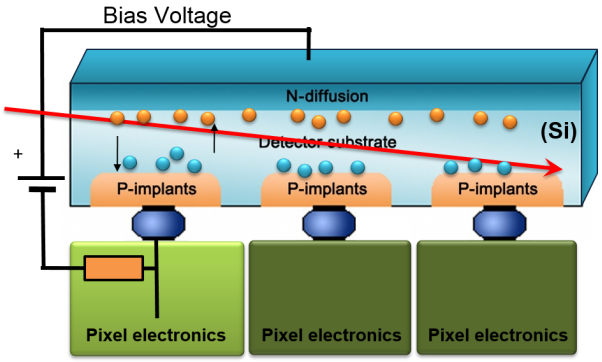
Existuje celá řada částicových pixelových detektorů (AGH_Fermilab, Pilatus, Philips Chromaix apod.) [1], tato práce se však zabývá pouze detektory z rodiny Medipix, které jsou vyvíjeny v rámci stejnojmenné kolaborace Medipix[1] v CERN. Tato kolaborace sdružuje několik desítek vědeckých institucí a univerzit po celém světě, mezi které roku 1999 patří od i ÚTEF ČVUT v Praze.

[1] <http://medipix.web.cern.ch/>

2.1 Princip detekce

Činnost hybridních pixelových detektorů je založena na známém principu detekce ionizujícího záření v polovodiči.

Na obrázku 2.1 je znázorněn princip této detekce. V horní části se nachází polovodičový senzor, pro který je jako materiál nejčastěji použit křemík, ale výjimkou není ani `GaAs`, či `CdTe`. Pod tímto senzorem se nachází vyčítací elektronika, která tvoří jednotlivé pixely. Jako náhradní schéma tohoto obvodu si lze představit diodu zapojenou v závěrném směru, skrze kterou protéká jen minimální proud.

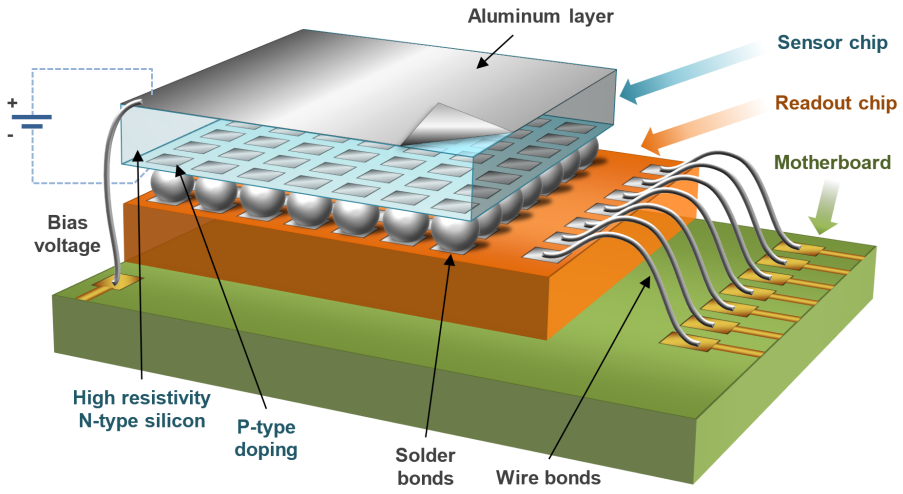


Obrázek 2.1: Princip detekce ionizujícího záření (převzato z [12]), kde červená šipka je procházející částice, žlutě jsou znázorněny elektrony, modře díry. Elektronika každého pixelu zpracovává napěťový pulz.

Při vniku ionizující částice do detektoru dojde k předání části její energie detekčnímu materiálu. Vznikají elektron-děrové páry a díky lavinovému efektu dochází k otevření PN přechodu. Tím dojde ke vzniku proudového pulsu, jež je pomocí měřícího odporu převeden na napětí a dále měřící elektronikou zpracováván.

Na obrázku 2.2 je znázorněna struktura Medipix detektoru. Nahoře se nachází, polovodičový senzor (`Sensor chip`), který je spojen s integrovaným `ASIC`[2] vyčítacím čipem (tzv. `Readout chip`) pomocí technologie zvané `bump-bonding` (na obr. 2.2 jako `Solder bonds`). Odtud také pochází název "hybridní"- jedná se o spojení senzoru a `ASIC` čipu. Každý pixel senzoru tvoří jeden PN přechod. Vyčítací čip je spojen s další nezbytnou elektronikou (na obr 2.2 znázorněnou jako `Motherboard`) pomocí tzv. `wire-bonds`. Z této elektroniky je vyvedeno napětí na polovodičový senzor - `bias`, zajišťující vyprázdnění detekčního objemu polovodičového senzoru.

[2] z angl. Application Specific Integrated Circuit



Obrázek 2.2: Struktura hybridního polovodičového pixelového detektoru rodiny Medipix (převzato z [12])

2.2 Detektory rodiny Medipix

Pro realizaci sítě ATLAS TPX byly použity pouze detektory typu Timepix. Pro srovnání je níže uveden stručný popis ostatních detektorů rodiny Medipix. Do této rodiny patří především: Medipix1, Medipix2 [10], Timepix [11], Medipix3, nově Timepix3 [13] a další jsou ve vývoji, například Timepix2 a Dosepix.

Medipix1 Prvním detektorem z této rodiny je Medipix1, který byl uveden v roce 1997. Také je známy pod názvem PCC (z angl. Photon Counting Chip). Jedná se o prototyp digitálního `CMOS`[3] zobrazovacího čipu, který nachází uplatnění ve vysokoenergetických fyzikálních experimentech [8]. Je schopný operovat jen v `Medipix módu` (viz 2.3). Tento detektor má matici 64×64 pixelů, každý s hranou o délce 170 μm a celková aktivní plocha je $1,2\ cm^2$. Detektor obsahuje 15-bitový čítač, čímž umožňuje v rámci jedné akvizice zaregistrovat až 32767 událostí.

Medipix2 Jedná se o přímého následníka detektoru Medipix1. Díky větší integraci `CMOS` technologie bylo možné přidat novou funkcionalitu a zmenšit velikost pixelů. Detektor obsahuje matici 256×256 pixelů, délka hrany jednoho pixelu se zmenšila na 55 μm a celková aktivní plocha vzrostla na 2 cm^2.

Timepix Tento detektor je na bázi detektoru Medipix2, prodělal však výraznou obměnu digitální části. Byla přidána synchronizační logika, která přinesla dva nové módy - TOT (měření energie) a TOA (měření doby příletu částice), přičemž každý pixel v jeden okamžik umožňuje měřit jen v jednom módu (více o módech v 2.3).

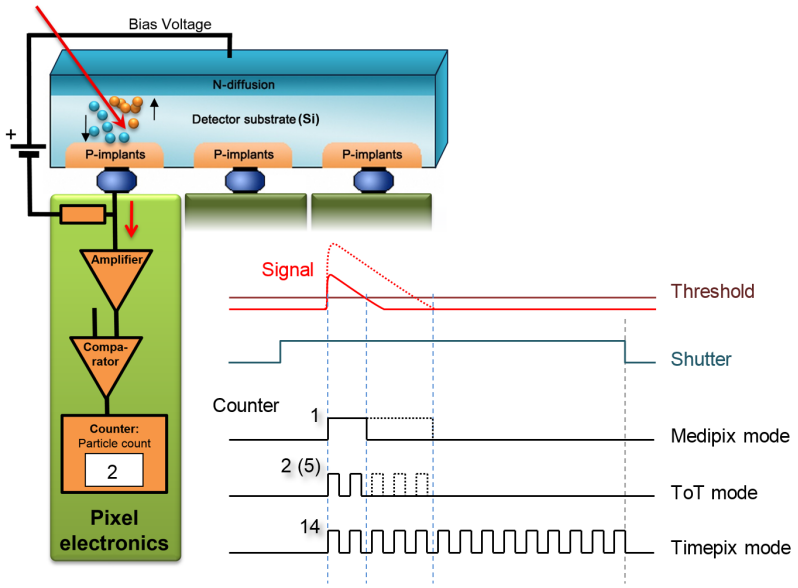
[3] z angl. Complementary Metal–Oxide–Semiconductor

Medipix3 U Medipix3, byla výrazným způsobem přepracována vyčítací elektronika za cílem snížení zkreslení, způsobeném sdílením náboje mezi sousedními pixely (tento efekt je také znám pod pojmem `Charge Sharing` efekt [5]).

Timepix3 Tento detektor vychází z detektoru Timepix. Oproti kterému má tento detektor vylepšenou vyčítací logiku a o jeden čítač na pixel více. To mu mimo jiné umožňuje měřit v TOT a TOA módu současně. Navíc ještě přináší `Data-driven` vyčítací mód (obdobně, jako Medipix3), který na rozdíl od `Frame-based` vyčítání minimalizuje mrtvou dobu detektoru.

2.3 Provozní módy detektoru Timepix

V této podkapitole jsou popsány módy detektoru Timepix [11]. V levé části obrázku 2.3 je znázorněn pixel detektoru s blokovým schématem vyčítací elektroniky. Interagující částice (na obr. znázorněna červenou šipkou) v detekčním materiálu vyvolá proudový pulz. Ten je měřícím odporem převeden na napětí, které je dále zesilovačem (`Amplifier`) zesíleno. Toto napětí je dále porovnáno komparátorem (`Comparator`) s komparačním napětím (tzv. tresholdem[4]). Výsledek komparace je zpracován dle módu detektoru. Pro úplnost je třeba dodat, že `Shutter` na obr. 2.3 slouží pro spouštění, resp. ukončování akvizice.



Obrázek 2.3: Zpracování signálu z pohledu módu pixelu (převzato z [12])

Medipix mód Tento mód počítá počet částic, které během doby akvizice dopadly na aktivní plochu detektoru. Na obrázku 2.3 znázorněn, jako `Medipix mode`.

[4]Treshold je úroveň komparačního napětí, které je porovnáváno s aktuálním měřícím napětím na každém pixelu. Je-li tato úroveň překročena, dojde k detekováni události.

TOT (Time-Over-Treshold) Tento mód udává, jak dlouhou dobu (v počtu hodinových cyklů měřící frekvence) bylo zesílené napětí na detektoru vyšší, než komparační (treshold). Počet těchto cyklů je ekvivalentní deponované energii částice. Tento vztah je nelineární a pro získání energie z TOT je třeba detektor kalibrovat - o tom pojednává kapitola 3.

TOA (Time-Of-Arrival) Čítač pro daný pixel je spuštěn po překročení tresholdu a zůstává v běhu až do konce akvizice. Tím je určena doba příletu částice. Tento mód je také známý pod označením `Timepix mode` a nalézá své uplatnění především při měření koincidencí (rekonstrukce trajektorie částice, interagující s více detektory pomocí času dopadu a souřadnic zasažených pixelů).

2.4 FITPix

FITPix[5] [9] je vyčítací rozhraní, pracující téměř se všemi detektory rodiny Medipix, vyvíjené v ÚTEF ČVUT v Praze od roku 2010 - viz obr. 2.4. Toto rozhraní se skládá z FPGA[6] obvodu, USB 2.0 rozhraní, DAC převodníků (převodník digitálního signálu na analogový), ADC převodníků (převodník analogového signálu na digitální) a z obvodů generujících napětí pro polovodičový senzor (tzv. bias). Toto zařízení umožňuje plnohodnotné ovládání připojeného detekčního čipu, včetně nastavování měřící frekvence, tresholdu, řízení shutteru (sloužícího pro ovládání akvizice, viz 2.3) apod. Také přináší možnost ovládat shutter pomocí hardwarového trigger signálu pro měření s více detektory současně, resp. pro jejich synchronizaci.

Obrázek 2.4: FITPix (vlevo zařízení FITPix, vpravo DPS zařízení FITPix)

Tato architektura s FPGA byla použita především z důvodu dosažení vyšších datových toků a také vyšší radiační odolnosti, které by za použití konvenčních mikroprocesorů nemohlo být dosaženo. Další výhodou je menší počet aktivních prvků, což se projeví krom spotřeby i na nižších tepelných ztrátách. Tento parametr je velice důležitý, především pro nasazení ve vakuu.

[5] z angl. Fast Interface for Timepix Pixel Detectors
[6] z angl. Field Programmable Gate Array

2.5 Pixelman

Pixelman [15] je softwarový balík, vyvíjený v ÚTEF ČVUT v Praze, sloužící pro řízení detektorů z rodiny Medipix pomocí vyčítacího rozhraní FITPix 2.4, Muros a dalších. Tento software umožňuje akvizici dat, jejich vizualizaci a následnou analýzu.

Jedná se o vysoce modulární systém, který mimo jiné umožňuje rozšíření své funkcionality o pluginy, které mají přístup k funkcím, poskytovaným jádrem Pixelmanu. Každý plugin může zaregistrovat své funkce, takže i ostatní pluginy mohou využívat jeho funkcionalitu. Jsou podporovány pluginy, vyvinuté v jazycích Java, C/C++ a Python.

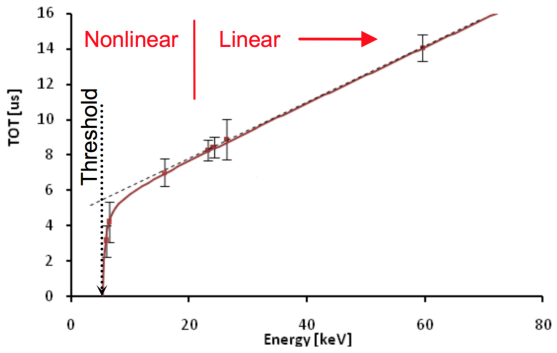
Kapitola 3

Energetická kalibrace

Tato kapitola pojednává o metodách energetické kalibraci hybridních částicových pixelových detektorů, pracujících v Time-Over-Treshold módu a o implementaci jedné z nich pro účely kalibrace detektorů sítě ATLAS TPX.

3.1 Motivace

Každý detektor ionizujícího záření je třeba pře použitím zkalibrovat pomocí známých zrojů záření. Vzniká tak přepočet vnitřních elektrických veličin detektoru na energii. Pro účely fyzikálních měření je zvykem užívat jako jednotku energie eV, resp. keV. V případě detektoru Timepix se z důvodu nelineární odezvy (obr. 3.1) pixelové elektroniky jedná o netriviální úlohu. Cílem kalibrace je nalézt parametry analyticky popisující kalibrační křivku, na příklad pomocí funkce 3.1.



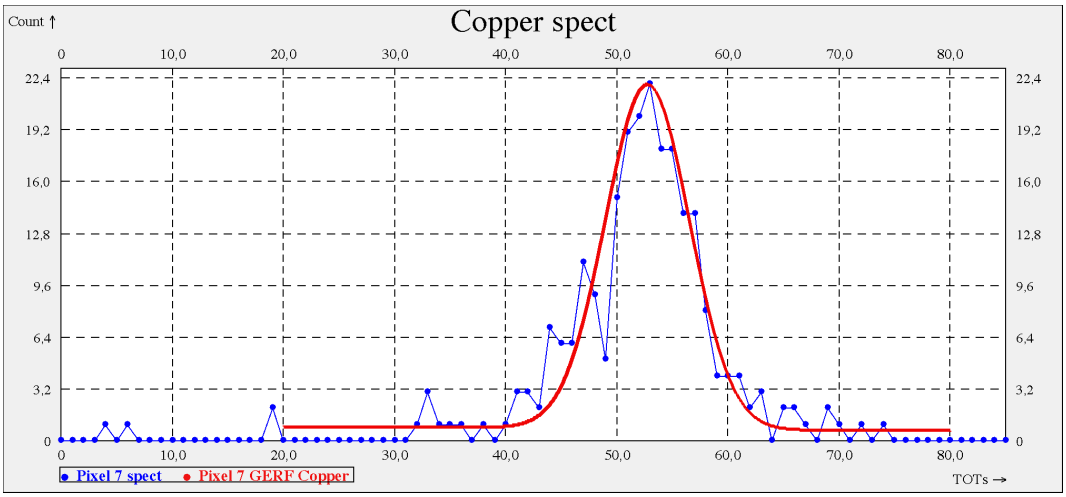
Obrázek 3.1: Kalibrační funkce (převzato z [6]) udává závislost mezi energií a TOT. Vzniká proložením naměřených kalibračních bodů pomocí funkce 3.1. Tato funkce vznikla složením hyperboly (popisující nelineární oblast nižších energií) a přímky (pro oblast s vyšší energií).

$$f_{calib}(x) = ax + b - \frac{c}{x - t} \tag{3.1}$$

3.2 Přehled kalibračních metod

3.2.1 Kalibrace detektorů za použití rentgenového záření

Tato kalibrační metoda [6] spočívá v měření rentgenové fluorescence (viz [5]), což je děj, ke kterému dochází, když je materiál[1] (terč) ozařován rentgenovým zářením, které vyráží excitované elektrony z jeho atomů. Je-li vyražen elektron na nižší energetické úrovní, tak elektron z vyšší energetické úrovně deexcituje a obsadí jeho místo. Přebytečnou energii ztratí ve formě vyzářeného fotonu (tzv. charakteristické záření). Spojitou část rentgenového spektra je potřeba odstínit. Výběrem vhodných terčů lze získat několik diskrétních energií záření, tzv. kalibrační body. Z důvodů statistických vlastností záření je třeba pro každý kalibrační bod pořídit velké množství snímků, ze kterých jsou následně vyfiltrovány jen tzv. `Single-Hit` události, při kterých interagující částice zasáhla jen jeden pixel. Tyto události jsou filtrovány, z důvodu dosažení vyšší kvality kalibrace, pomocí potlačení zkreslení způsobeného `Charge Sharing` efektem[2].



Obrázek 3.2: Spektrum TOT hodnot jednoho pixelu s proložením Gaussovou funkcí, sečtenou s Gaussovou chybovou funkcí (tzv. error funkce). Zdrojem rentgenové fluorescence byla měď.

Na obrázku 3.2 je příklad spektra pro jeden pixel detektoru a fluorescenčního záření z mědi. Na vodorovné ose tohoto spektra se nachází jednotlivé TOT hodnoty a na svislé pak jejich četnost ve všech snímcích. Z obrázku je patrné, že nejčetnější hodnotou TOT je zhruba hodnota 53, která odpovídá energii fluorescenčního záření mědi, což je $5,9\ keV$. Požadovanou hodnotu TOT lze získat proložením spektra funkcí 3.2. Ta vznikla z Gaussovy funkce, ke které byla z důvodu levé nesymetrie, způsobené `Charge Sahring` efektem, přičtena

[1] Pro kalibraci se používají kovy, na příklad Am, In, Cu, Fe apod.

[2] Když dopadne nabitá částice na polovodičový senzor, vzniknou elektron-děrové páry, které jsou staženy nejen zasaženým pixelem, ale vetšinou i několika sousedními. To je dáno jednou společnou elektrodou pro všechny pixely senzoru (viz obr. 2.2).

Gaussova chybová funkce.

$$f_{GERF}(x) = \underbrace{Ae^{-\frac{(x-\mu)^2}{2\sigma^2}}}_{\text{Gaussova funkce}} + \underbrace{\frac{avg_{right} - avg_{left}}{\sigma\sqrt{2\pi}} \int_{-\infty}^{t} e^{-\frac{(t-\mu)^2}{2\sigma^2}} + avg_{left}}_{\text{Gaussova chybová funkce}} \tag{3.2}$$

Parametry funkce 3.2 jsou následující:

- **A** je amplituda.
- μ je střední hodnota hledané energie.
- σ udává rozptyl střední hodnoty energie μ, kterou je možné ji vypočítat ze vzorce $\sigma = \frac{2\sqrt{2ln_2}}{FWHM}$, kde $FWHM$[3] udává šířku gausiánu v polovině jeho výšky.
- $\mathbf{avg_{right}}$ (resp. $\mathbf{avg_{left}}$) je průměrná hodnota spektra na pravém (resp. levém) úpatí gausiánu.

Z těchto kalibračních bodů je možné sestavit kalibrační funkci (viz vzorec 3.1), udávající závislost mezi energií a TOT.

3.2.2 Kalibrace detektorů pomocí LED diod

Princip kalibrace pomocí SMD LED diod spočívá v působení přesného množství světelného záření na polovodičový senzor detektoru. V současné době je tato metoda ve fázi vývoje a doposud nebyla publikována. Metoda je použitelná pouze pro detektory, na jejichž senzoru není napařena tenká vrstva hliníku, která světelné záření nepropouští.

Jako zdroj světla byl v ÚTEF vyvinut modul s maticí 8×8 SMD LED diod, který je možné ovládat pomocí `RS232` sériové linky. K tomuto modulu byl rovněž vytvořen plugin do softwarového balíku Pixelman 2.5, který automatizuje proces nabírání dat této metody. Přes sériovou linku je schopen řídit modul s LED diodami a zároveň pomocí jádra Pixelmanu ovládá akvizici dat detektoru.

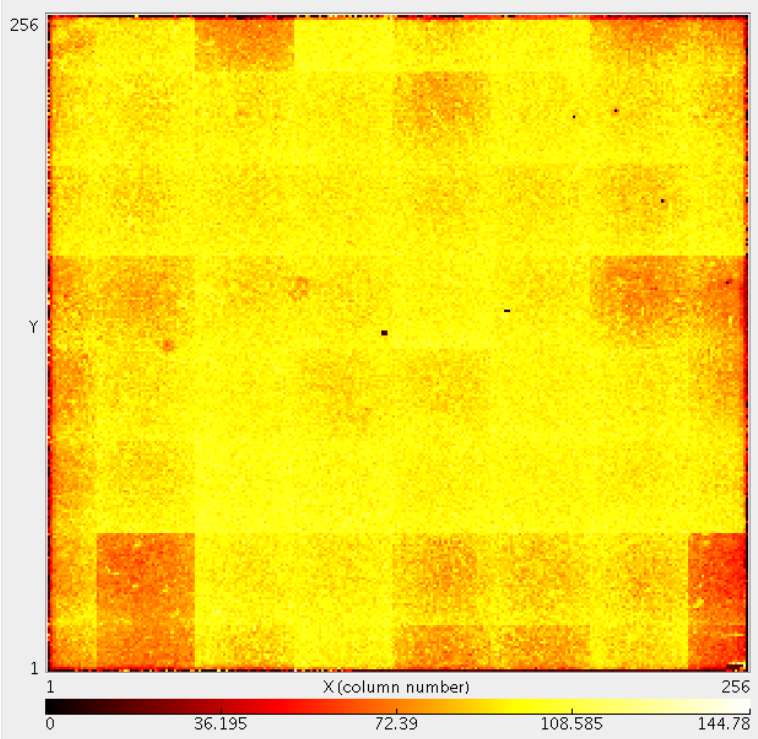
Kroky algoritmu kalibrace jsou následující:

1. Inicializace:
   - nastavení spouštění akvizice detektoru na externí hardwarový trigger (který bude ovládán modulem s LED diodami)
   - nastavení energie světelného záření (počet zabliknutí diody, délka periody jednoho bliknutí a doba aktivace diody v jedné periodě)
   - délku akvizice snímku (vypočtené dle periody blikání a počet opakování)
2. Měřící smyčka (opakuje se pro všechny LED diody). V rámci jednoho průchodu se provede následující:

[3] z angl. Full Width at Half Maximum

- Zamaskování všech pixelů detektoru, krom těch pixelů, které jsou pod aktivní diodou.
- Spuštění akvizice.
- Vyčtení a uložení snímku z detektoru.

3. Následně se hodnoty všech snímků sečtou do jednoho snímku - viz obr. 3.3.



Obrázek 3.3: Snímek z detektoru, ozařovaném modulem s LED diodami

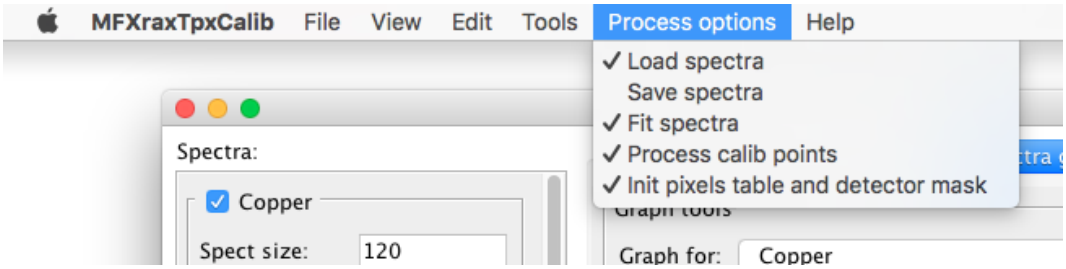
Z obrázku 3.3 jsou patrné hrany jednotlivých dílčích měření, které vznikly nehomogenitou elektronických vlastností jednotlivých diod. Tento jev může být odstraněn normalizací jejich světelné intenzity pomocí vynásobení času aktivace každé diody normalizační maticí.

Tímto způsobem jsou pro každý pixel detektoru získají jednotlivé kalibrační body, které jsou následně proloží kalibrační funkcí (viz vzorec 3.1), jak již bylo popsáno v kapitole 3.2.1.

3.3 Software pro kalibraci detektorů za použití rentgenového záření

Tato podkapitola pojednává o softwaru pro energetickou kalibraci pixelových detektorů za použití kalibrační metody s rentgenovým zářením (viz 3.2.1). Tento software původně vznikal pro účely kalibrace sítě ATLAS TPX, avšak později byl rozšířen a nyní je kompatibilní se všemi detektory, pracujícími v TOT módu. Software vyl vyvinut v programovacím jazyce `Java` a grafické rozhraní bylo vytvořeno za pomoci knihovny `Swing`. V rámci této práce rovněž vznikla vlastní knihovna pro vizualizaci 2D grafů.

Jak již bylo zmíněno v kapitole 3.2.1, proces kalibrace se skládá z několika kroků, a to z vytvoření spekter pro jednotlivé pixely a zdroje záření, následném nalezení kalibračních bodů z těchto spekter a vytvoření kalibrační funkce pro každý pixel detektoru. Na obrázku 3.4 je znázorněna možnost nastavení provedení jednotlivých kroků kalibračního procesu nad zvolenými měřeními (viz obr. 3.5 - levý postranní panel), v rámci jednoho průchodu po stisknutí tlačítka `Start/Abort` (viz obr. 3.5 vlevo dole).



Obrázek 3.4: Screenshot kalibračního softwaru - volby zpracování v rámci kalibračního procesu

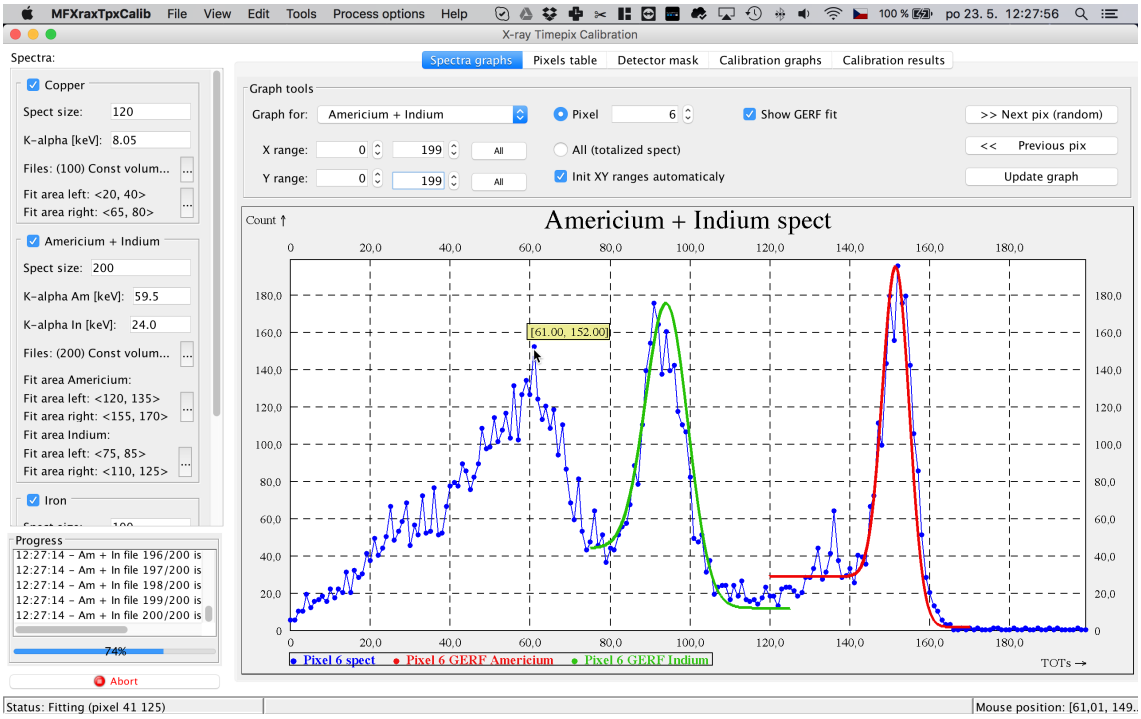
3.3.1 Vstupní data kalibrace

Vstupní data se skládají z několika sad měření pro různé zdroje mono-energetického ionizujícího záření (např. fluorescenčního), jejichž energie jsou předem známy. Z naměřených hodnot jsou vytvořena spektra pro každý pixel detektoru (příkladem takového spektra může být obrázek 3.5.

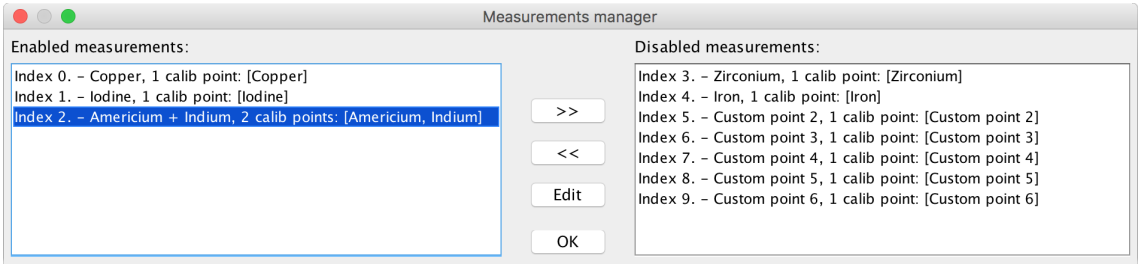
K výběru vstupních dat slouží postranní panel hlavního okna kalibračního softwaru, kde je seznam jednotlivých měření. K manipulaci s položkami v tomto seznamu slouží `Measurement manager` (dostupný ze záložky `File` v hlavní liště - viz obr. 3.6). V rámci tohoto nástroje je možné přidávat, odebírat, či jinak upravovat jednotlivá měření a jejich parametry (například velikost spektra apod.). Každé měření může obsahovat až tři kalibrační body, jejichž parametry je možné také nastavit skrze `Measurement manager`.

Vstupní data jsou podporována ve třech formátech:

Multiframe Formát slouží pro hromadné ukládání surových snímků z detektoru, kde jsou jednotlivé snímky zapsány jako seznam zasažených pixelů a jejich hodnot (pro více o



Obrázek 3.5: Screenshot kalibračního softwaru, zobrazující složené spektrum pro gama emisi americia (pravý červený pík) a fluoresceni india (levý zelený pík), proložené funkcí 3.2. Levou část spekta tvoří fotony vzniklé Comptonovým rozptylem.



Obrázek 3.6: Screenshot kalibračního softwaru - Measurement manager

tomto formátu viz 4.4.3). Každý snímek je třeba načíst a vyfiltrovat pouze `Single-Hit` události (viz 3.2.1), ze kterých jsou následně vytvořena spektra pro všechny pixely.

Cluster log Tento formát obsahuje snímky zapsané jako množinu clusterů, resp. shluků vzájemně sousedících pixelů s nenulovou hodnotou. Z těchto clusterů jsou vybrány jen clustery o velikosti jednoho pixelu (`Single-Hit` události), z kterých jsou následně vytvořena spektra.

Spektra Tento formát obsahuje již zpracovaná spektra. Pro spektrum o velikosti n hodnot jsou vstupní data tvořena n soubory, kde každý soubor obsahuje matici výskytů n-té

hodnoty spektra ve všech pixelech detektoru. Tento formát je nejvýhodnější, z důvodu malého objemu dat a zvýšení rychlosti zpracovávání dat (odpadá filtrování `Single-Hit` událostí).

3.3.2 Analýza spekter

Pro nalezení závislosti TOT na energii zdroje ionizujícího záření v naměřených spektrech dochází pomocí proložení spekter funkcí 3.2 (viz 3.2.1).

Tento algoritmus je v softwaru implementován za použití metody nejmenších čtverců. Úkolem tohoto algoritmu je nalézt takové parametry A, μ, σ, avg_{left} a avg_{right} funkce 3.2 tak, aby hodnota funkce 3.3 byla minimální, resp. aby suma čtverců vzdáleností jednotlivých hodnot spektra a jejich příslušných funkčních hodnot funkce 3.2 (tzv. reziduí) byla minimální.

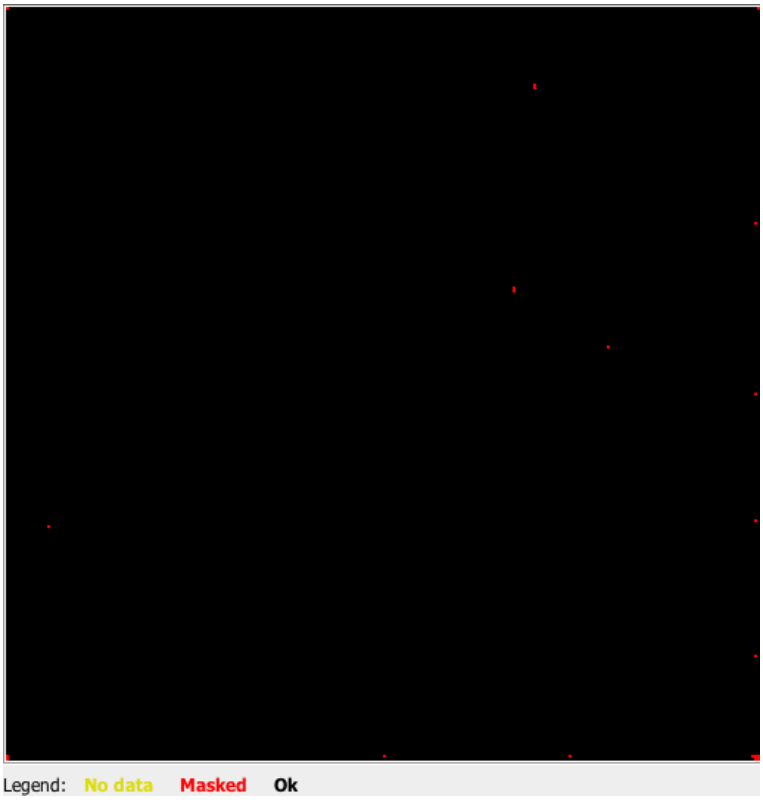
$$E = \frac{1}{N}\sum_{i=0}^{N}(f_{GERF}(i) - count_i)^2 \tag{3.3}$$

Jedná se o iterativní metodu, kde hodnoty parametrů A, μ, σ, avg_{left} a avg_{right} jsou v rámci jednotlivých iterací postupně vylepšovány, resp. hodnota funkce 3.3 je minimalizována. Jelikož všechny tyto parametry jsou na sobě nezávislé, optimum každého z parametrů může být získáno pomocí Gradientní metody v rámci jedné iterace.

Algoritmus této metody je následující: Nejprve se k danému parametru přičte konstanta základního kroku a sleduje se změna sumy reziduí. Když se tato suma zvětší, jedná se o znamení špatného směru kroku, který je třeba vrátit a konstantu základního kroku násobit -1. Dále se ve smyčce přičítá postupně se zvětšující násobek základního kroku a sleduje se změna sumy reziduí. Začne-li se tato suma zvětšovat, vrátí se aktuální krok zpátky a algoritmus končí. Toto se v rámci jedné iterace provede pro všechny zkoumané parametry.

Počet těchto iterací je možné v softwaru nastavit v hlavní liště (`Edit > Set number of iterations`). Jelikož tato metoda konverguje relativně rychle, optimální (z hlediska doby a kvality kalibrace) počet iterací je 3-5.

Z nalezených optimálních parametrů A, σ, avg_{left} a avg_{right}, resp. z odchylky od jejich mediánu, je možné pro každý pixel určit jeho kvalitu, na jejímž základě vzniká maska špatných pixelů detektoru (viz obr. 3.7). Maska může být generována na základě jednoho z parametrů, či více parametrů a může být uložena v textové podobě (matice hodnot, kde 0 znamená nezamaskovaný a 1 zamaskovaný pixel).



Obrázek 3.7: Screenshot kalibračního softwaru - maska špatných pixelů

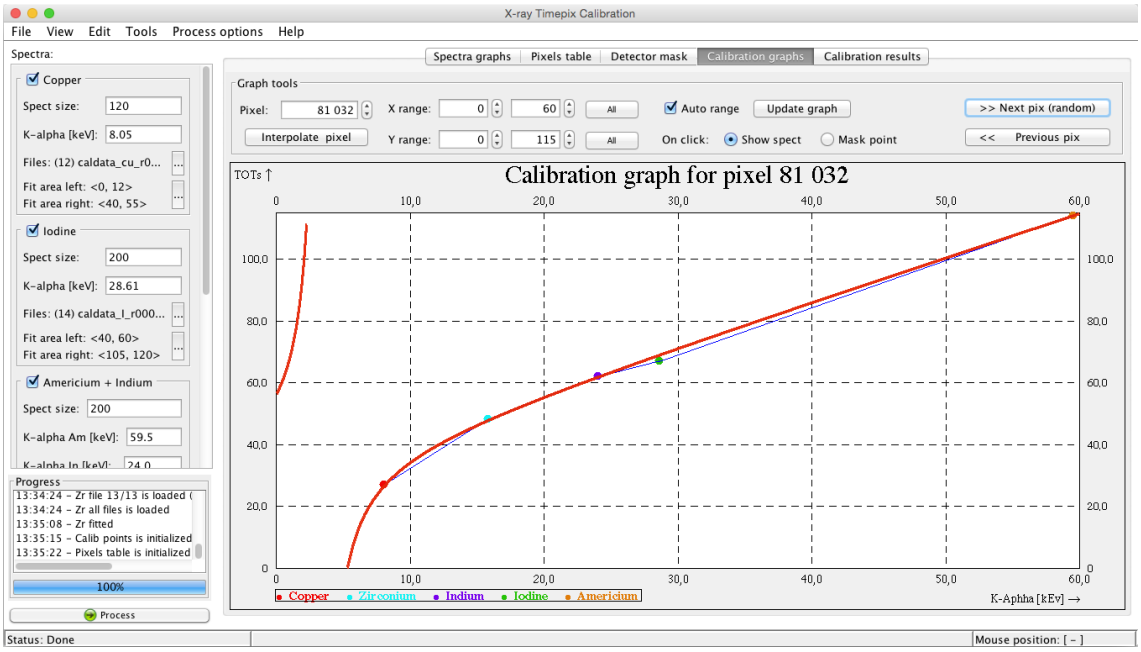
3.3.3 Vytvoření kalibrační funkce

Po získání jednotlivých kalibračních bodů je možno přikročit k vytvoření kalibrační funkce pro každý pixel detektoru (viz obr. 3.8). Tato úloha spočívá v proložení kalibračních bodů jednotlivých pixelů kalibrační funkcí 3.1, resp. nalezení takových parametrů a ,b ,c a t kalibrační funkce 3.1 tak, aby součet reziduí byl minimální (viz 3.3.2).

Jednotlivé parametry kalibrační funkce 3.1 jsou vzájemně závislé, proto pro nalezení těchto parametrů nelze použít stejný algoritmus, jako při analýze spekter (3.3.2). Tato metoda byla navržena tak, aby byla schopná kalibrační funkci vytvořit ze dvou bodů. Počet kalibračních bodů je variabilní položkou, proto jsou pro nalezení parametrů kalibrační funkce 3.1 použity různé algoritmy.

Algoritmus pro dva body Při kalibraci pomocí dvou kalibračních bodů je uživatel vyzván k zadání parametrů c a t kalibrační funkce. Zbylé parametry jsou vypočteny pomocí soustavy rovnic 3.4 a dvou známých kalibračních bodů.

$$\begin{aligned} ax_1 + b - \frac{c}{x_1 - t} &= y_1 \\ ax_2 + b - \frac{c}{x_2 - t} &= y_2 \end{aligned} \tag{3.4}$$



Obrázek 3.8: Screenshot kalibračního softwaru - kalibrační funkce

Při parametrizaci proměnných c a t vzniká velká nepřesnost, především v nelineární oblasti nižších energiích, proto je tato metoda pro dva body nejméně přesná.

Algoritmus pro tři body Tří-bodová kalibrace je oproti dvou-bodové kalibraci značně přesnější. Protože kalibrační funkce má čtyři parametry, je třeba nějaký z nich parametrizovat. Uživatel je na začátku kalibračního procesu vyzván k zadání parametru t a zbylé parametry budou vypočteny ze soustavy rovnic 3.5.

$$\begin{aligned} ax_1 + b - \frac{c}{x_1 - t} &= y_1 \\ ax_2 + b - \frac{c}{x_2 - t} &= y_2 \\ ax_3 + b - \frac{c}{x_3 - t} &= y_3 \end{aligned} \tag{3.5}$$

Algoritmus pro čtyři body Algoritmus pro čtyři body je komplikovanější. Nejprve je parametrizován parametr t (bez zásahu uživatele). Poté jsou parametry a, b a c vypočteny pomocí třech bodů soustavy rovnic 3.5. Tyto tři body jsou vybrány následovně:

1. **bod** je vybrán jako bod s nejnižší energií.
2. **bod** je vybrán z některého z prostředních bodů (ve výchozím nastavení 2. bod s nejnižší energií - možné změnit v `Edit > Adjust middle trial point`).
3. **bod** je vybrán jako bod s nejvyšší energií.

Poté pomocí metody bisekce a metody nejmenších čtverců je nalezena taková hodnota parametru t, kdy jeho reziduum je minimální, v ideálním případě nulové.

Algoritmus pro pět a více bodů První část algoritmu pro pět a více bodů se od algoritmu pro čtyři body nikterak neliší - pomocí třech bodů (volba prostředního je opět na uživateli) jsou vypočteny parametry a, b a c a poté pomocí metody bisekce a čtvrtého bodu je určen parametr t. V ideálním případě prochází funkce 3.1 všemi čtyřmi body, nebo alespoň jejich rezidua jsou minimální.

Pátý bod (popř. další body) je zohledněn pomocí metody bisekce a metody nejmenších čtverců, přičemž v rámci jedné iterace tohoto algoritmu se mění všechny parametry (jelikož jsou vzájemně závislé). Postupně je ke každému parametru přičtena konstanta základního kroku a je sledována se změna sumy reziduí. Z těchto změn je vytvořen vektor směru změny všech parametrů. Pomocí metody bisekce je přičítán násobek tohoto vektoru (v jednotlivých krocích) k parametrům funkce, přičemž je sledována změna sumy reziduí. Daná iterace končí, když se suma reziduí přestane zmenšovat.

Počet těchto iterací uživatel může opět nastavit v hlavní liště (`Edit > Set number of iterations`).

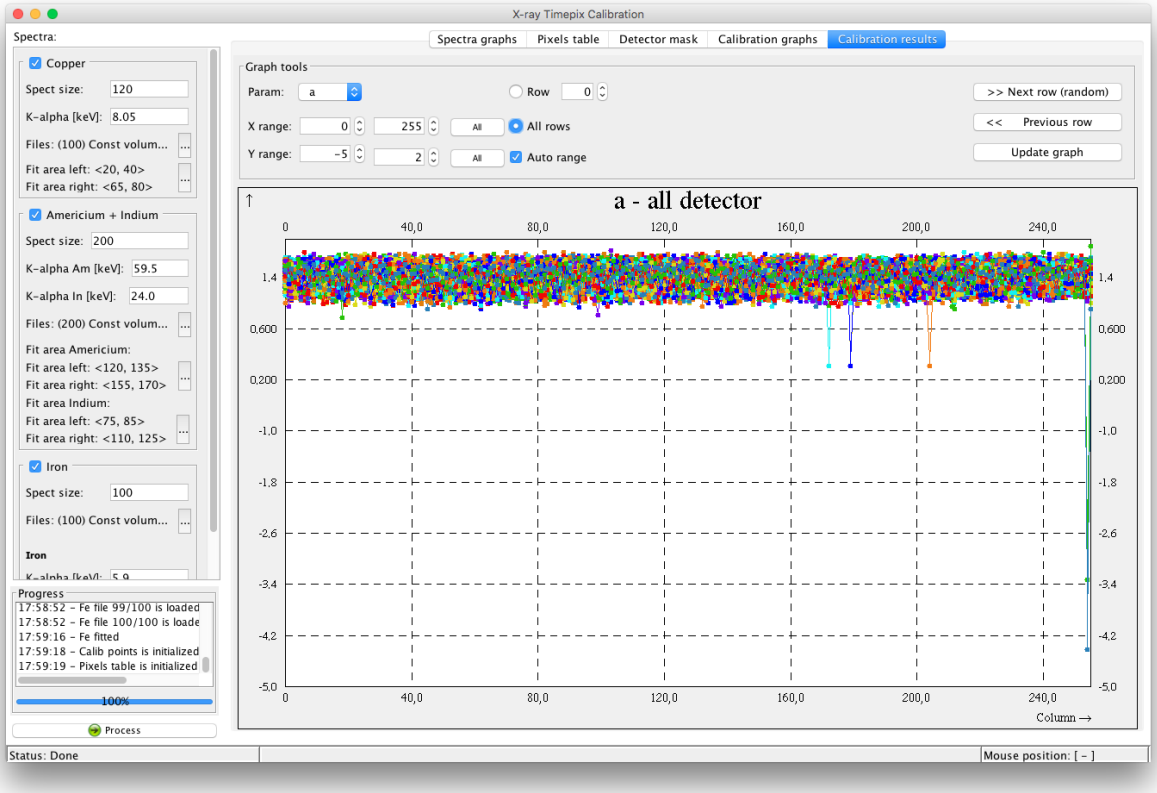
Po dokončení procesu kalibrace software nabízí v záložce `Calibration graphs` možnost zobrazení kalibrační funkce pro jednotlivé pixely detektoru. Zde je možné jednotlivé pixely interpolovat (více v 3.3.4), nebo kliknutím na daný kalibrační bod zamaskovat, nebo zobrazit jeho spektrum (dle preferencí uživatele). Software rovněž umožňuje hromadné maskování kalibračních bodů ve zvoleném intervalu (`Tools > Batch calib points masking` v hlavní liště).

Výstupní data tvoří a ,b ,c a t parametry kalibrační funkce 3.1, které je možné uložit (pomocí `File > Save calib files` v hlavní liště). Tato data je možné uložit v samostatných souborech pro každý parametr, kde hodnoty daného parametru pro jednotlivé pixely jsou zapsány v matici, nebo je možné tato data uložit do jednoho souboru, kde jednotlivé parametry jsou zapsány ve sloupcích.

3.3.4 Dodatečné úpravy

Tento kalibrační software rovněž nabízí nástroje pro vizualizaci kvality kalibrace. K tomuto účelu slouží záložka `Calibration results`, kde v rámci bočního pohledu na detektor je možné si v grafu zobrazit všechny hodnoty vybraného parametru kalibrační funkce 3.1 - viz obr. 3.9.

Díky této vizualizaci výsledků kalibrace je možné snadno odhalit případnou deviaci některého z parametrů pro určitý pixel detektoru. Kalibrační hodnoty je pro takto poškozený pixel možné upravit, například pomocí interpolace hodnot čtyř sousedních pixelů (spojených hranou s poškozeným pixelem). Pixely je možné interpolovat samostatně, nebo automaticky pomocí nástroje zvaném `Batch interpolating` (v záložce `Tools` hlavní lišty), který uživateli umožňuje zvolit rozsah hodnot daného parametru, ve kterém budou všechny pixely interpolovány.

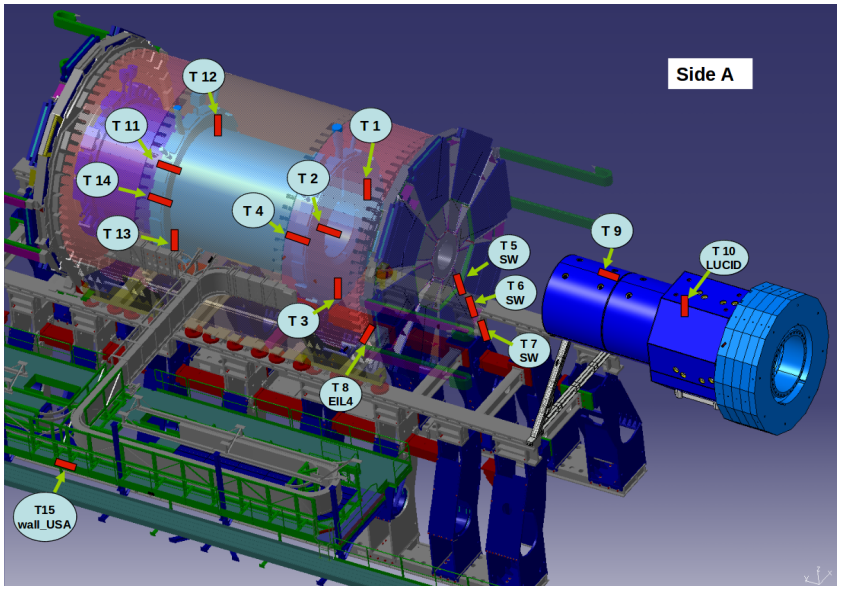


Obrázek 3.9: Screenshot kalibračního softwaru - vizualizace výsledných parametrů kalibrace v řezu přes řádky detektoru

Kapitola 4

ATLAS TPX

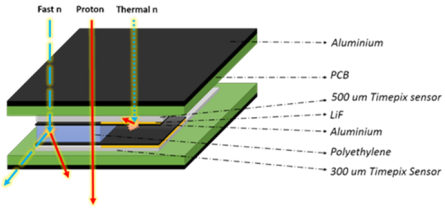
ATLAS TPX, síť 16[1] hybridních částicových pixelových detektorů typu Timepix 2.2, instalovaných na různé pozice experimentu ATLAS na LHC[2] v CERN (viz obr. 4.1) během LS2[3] (leden 2013 až březen 2015) je následníkem svého předchůdce - sítě ATLAS MPX (viz 4.1). Cílem modernizace této sítě bylo využití nových technologií, především pak nového detekčního čipu Timepix. Ten na rozdíl od svého předchůdce Medipix2 2.2 umožňuje rozšíření měřené informace o časovou oblast (viz 2.2). To nově umožňuje provozovat detektory v módech TOA[4] a TOT[5].



Obrázek 4.1: ATLAS TPX - přehledem rozmístění detektorů v experimentu ATLAS

[1] V průběhu LS3[3] (plánováno 2017 - 2018) je plánováno rozšíření teto sítě o další detektory
[2] z angl. Large Hadron Collider
[3] z angl. long shutdown - dlouhodobá technologická přestávka LHC
[4] z angl. Time of Arrival - čas příletu částice v hodinových cyklech detektoru od začátku akvizice
[5] z angl. Time Over Treshold - počet hodinových cyklů, kdy komparační napětí je větší, než referenční (ekvivalent energie deponované částice, viz kapitola 3)

Další změnou nové detektorové sítě ATLAS TPX oproti svému předchůdci je, že každý detektor obsahuje dva detekční čipy (senzory) s tloušťkami 300 μm a 500 μm, umístěné předními stranami k sobě - viz 4.2a. To přináší možnost měřit koincidence. Pokud částice projde oběma vrstvami detektoru a zároveň v každé zanechá jisté měřitelné množství své energie, je detekována oběma vrstvami a je možné zpětně zrekonstruovat její trajektorii. Tyto koincidence se nejsnáze detekují, pokud oba Timepix čipy pracují v módu TOA. Jelikož rychlost částice se blíží rychlosti světla, je vysoce pravděpodobné, že zasažené pixely budou mít stejnou hodnotu.



(a) Vrstvy detektoru se zobrazením principu detekce rychlých a termálních neutronů

(b) Rozmístění konvertorů (LiF je Lithium fluoride, Al je hliník, PE je Polyethylen a free je místo bez konvertorů)

Obrázek 4.2: ATLAS TPX detektor - vrstvy a rozmístění konvertorů

Mezi vrstvami detektoru jsou umístěny konvertory pro detekci termálních a rychlých neutronů. Rozmístění těchto konvertorů je na obrázku 4.2b.

Hlavním úkolem sítě ATLAS TPX je online monitorování spektrálních charakteristik různorodého radiačního prostředí ATLAS experimentu, založené na prostorovém uspořádání sítě a (vzhledem k aktuálním módům detektoru) na informaci o deponované energii interagujících částic, či na času jejich interakce.

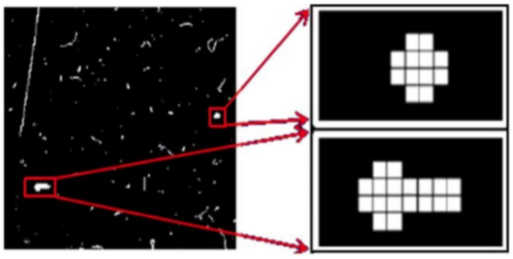
Detektory, instalované blízko interakčnímu bodu, jsou rovněž použity jako monitory integrované luminozity, což je veličina, která udává počet realizovaných srážek, resp. s intenzitou svazku urychlovače. Podle [18] je to veličina, která v případě srážení dvou proti sobě letících svazků ukazuje, jaký je součin počtů částic v jednotlivých svazcích prolétajících jednotkovou plochou v srážkové oblasti, vynásobený počtem obletů svazků za jednotku času (nejčastěji se vyjadřuje v jednotkách vyjadřujících počet částic na centimetr čtvereční za sekundu).

4.1 ATLAS MPX

ATLAS MPX[19][2] je předchůdcem detektorové sítě ATLAS TPX, který je v současné době plně nahrazen. Detektorová síť ATLAS MPX se skládala z 16 Medipix2 detektorů, které byly instalovány na různé pozice ATLAS detektoru. Hlavním cílem této sítě bylo měření vlastností radiačního pole uvnitř experimentu Atlas, jeho složení, spektroskopických charakteristik a částečně také přispěla k měření neutronů.

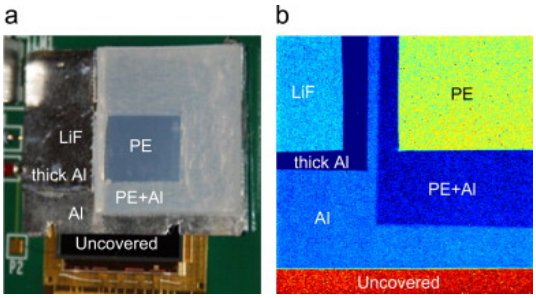
Všechny detektory operovaly v tzv. `Medipix módu`, který se vyznačuje tím, že v rámci jedné akvizice počítá počet částic, které interagovaly s pixelovou maticí detektoru a jejichž deponovaná energie byla vyšší než prahová. Na obrázku 4.3 je znázorněn snímek z jednoho

detektoru s detailem zachycených částic. Vpravo nahoře je částice typu `heavy blob` (těžká nabitá částice, jejíž trajektorie byla kolmá s povrchem detektoru), vpravo dole je pak zachycena částice typu `heavy track` (také těžká nabitá částice, která ale přiletěla pod větším úhlem a proto zanechala delší stopu). Více o klasifikaci částic v podkapitole 4.3.



Obrázek 4.3: Snímek z ATLAS MPX detektoru s výřezem zachycených částic (převzato z [2])

Každý detektor sítě ATLAS MPX byl osazen 300 μm tlustým křemíkovým senzorem, který byl pokryt konvertory pro lepší neutronovou detekční účinnost (obr. 4.4).

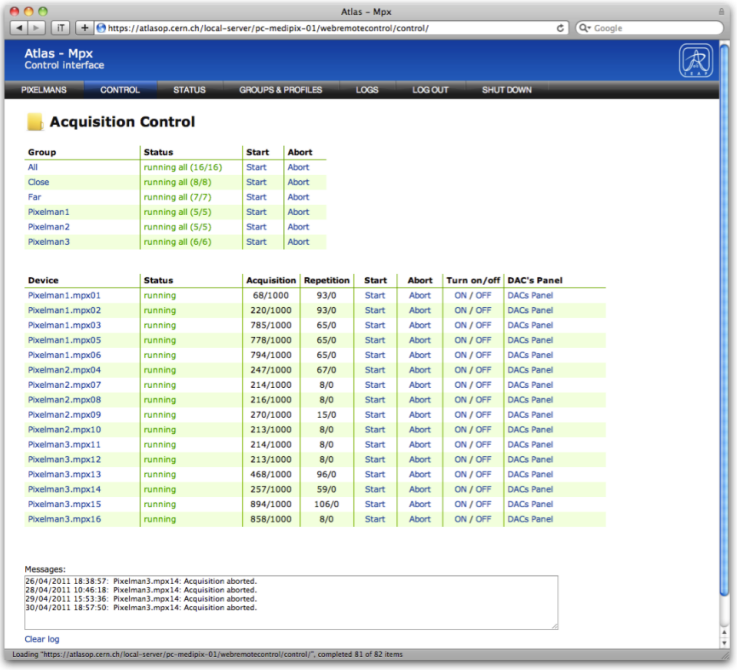


Obrázek 4.4: Fotografie znázorňující Medipix2 detektor s neutronovými konvertory (**a**) a struktura neutronových konvertorů (**b**)) [19]

4.1.1 Hardwarová a softwarová architektura sítě ATLAS MPX

Tato síť se skládala z 16 `Medipix2` 2.2 detektorů, které byly pomocí USB vyčítacího rozhraní `FITPix` 2.4 připojeny ke třem počítačům (z důvodu distribuce toku dat a výkonu). Na každém počítači se o komunikaci s detektory staral software `Pixelman` 2.5, který řídil akvizici dat, nastavování parametrů detektorů apod.

Pro vzdálené ovládání byl vyvinut plugin pro Pixelman, který umožňoval jeho rozšíření o TCP/IP ovládací vrstvu. Pomocí jednoduchého textového protokolu bylo tedy možné řídit každý ze třech uzlů. Pro tyto účely byla vyvinuta centrální řídící aplikace [16], pomocí které bylo možné řídit akvizici všech detektorů a nastavovat jejich parametry. Tato aplikace poskytovala webové rozhraní (obr. 4.5), které bylo možné ovládat odkudkoliv z internetu díky tou dobou méně striktním nárokům ze strany CERNu na síťovou bezpečnost.



Obrázek 4.5: ATLAS MPX - řídící aplikace (převzato z [17])

4.2 Hardwarová architektura sítě ATLAS TPX

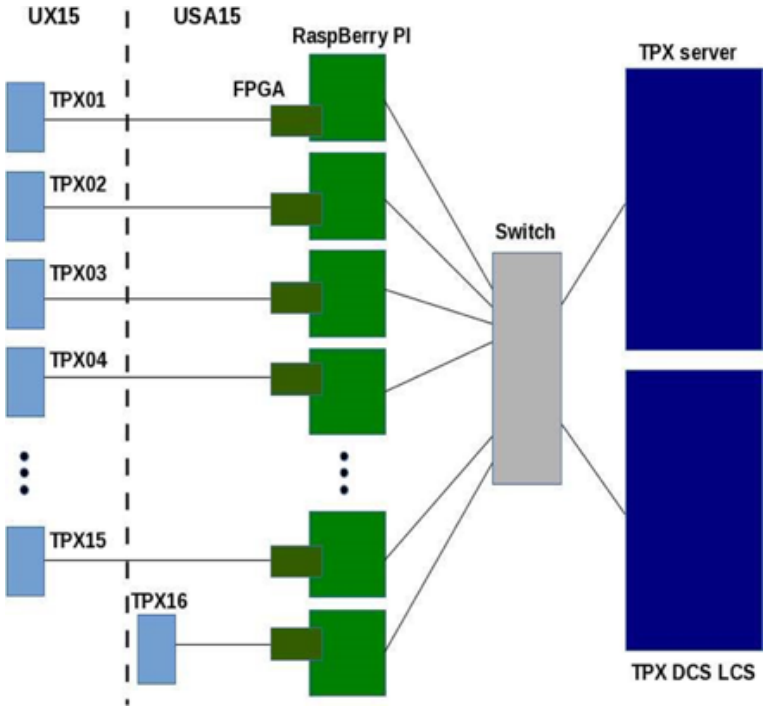
Při návrhu hardwarové architekty sítě ATLAS TPX musela být zohledněna zvýšená intenzita radiačního a elektromagnetického pole v prostorách ATLAS detektoru. Snahou proto bylo, umístit co nejvíce hardwarových komponent z dosahu těchto polí. Z pohledu hardwarové instalace této detektorové sítě se prostory ATLAS experimentu dělí na dvě části - `UX15` a `USA15` (viz obr. 4.6). V `UX15` se nachází vlastní experiment. V tomto prostoru jsou umístěny pouze detektory (na obr. 4.6 `TPX01` až `TPX15`) a zbytek sítě je instalován v `USA15`, kterou od zbytku experimentu odděluje masivní stínění. Tady se nachází vyčítací elektronika a další nezbytný hardware.

Na obrázku 4.7 je fotografie jednotlivých komponent sítě. Každý detektor je složen z dvojice detekčních čipů `Timepix`, které jsou pomocí `LVDS` zesilovačů a cca 100 m dlouhých ethernetových kabelů propojeny se zařízením `AtlasPix` (obr. 4.7 dole). Toto zařízení vzniklo modifikací vyčítacího rozhraní `FITPix` 2.4. Toto zařízení obsahuje `FPGA`[6], minipočítač `Raspberry Pi` a další podpůrnou elektroniku.

`FPGA` zajišťuje komunikaci s `Timepix` detektory, v rámci které dochází k nastavování řídících registrů `Timepix` čipů, ovládání akvizice, vyčítání dat, řízení triggeru[7] apod.

Minipočítač `Raspberry Pi` plní dvě úlohy. První je komunikace s `FPGA` pomocí `SPI`[8] rozhraní, deserializace (získání dat ze struktury komunikačního protokolu) a derandomizace

[6] z angl. Field Programmable Gate Array (programovatelné hradlové pole)
[7] řídící signál, který spouští resp. zastavuje (dle konfigurace) akvizici detektoru
[8] z angl. Serial Peripheral Interface (sériové periferní rozhraní)

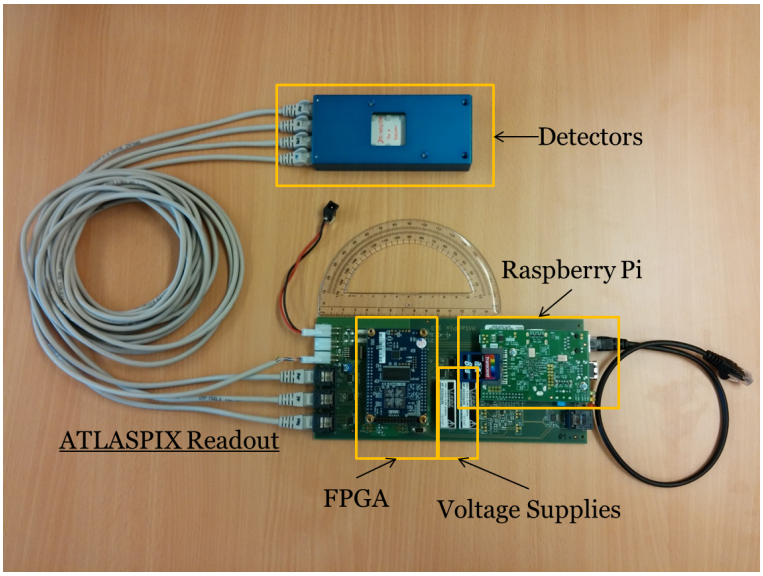


Obrázek 4.6: ATLAS TPX - diagram hw komponent, kde TPX01 až TPX15 jsou detektory umístěné v prostorách UX15 a zbytek sítě (vyčítací elektronika, propojená pomocí switche se servery TPX a DCS) umístěné v prostorách USA15

(není zaručena časová posloupnost) surových dat z `FPGA`. Druhou úlohou tohoto zařízení je poskytování API[9] vyšším řídícím vrstvám sítě pomocí specifikovaného komunikačního protokolu a klasického ethernetového rozhraní.

Všechna zařízení jsou pomocí ethernetového switche propojeny s `TPX serverem`, centrálním bodem této sítě, který jí pomocí řídícího softwaru 4.4 a komunikačního protokolu 4.4.1 ovládá. Zároveň je k síti připojen `TPX DCS`[10] `server`, pomocí kterého jsou různé stavové informace `ATLAS TPX` sítě předávány CERNu, resp. řízení ATLAS experimentu. Tyto stavové informace jsou převážně hardwarového charakteru (např. napětí, časování apod.), také jsou předávána data o počtu pořízených snímků, jejich okupanci apod.

[9] z angl. Application Programming Interface (aplikační programovací rozhraní)
[10] z angl. Data Control System



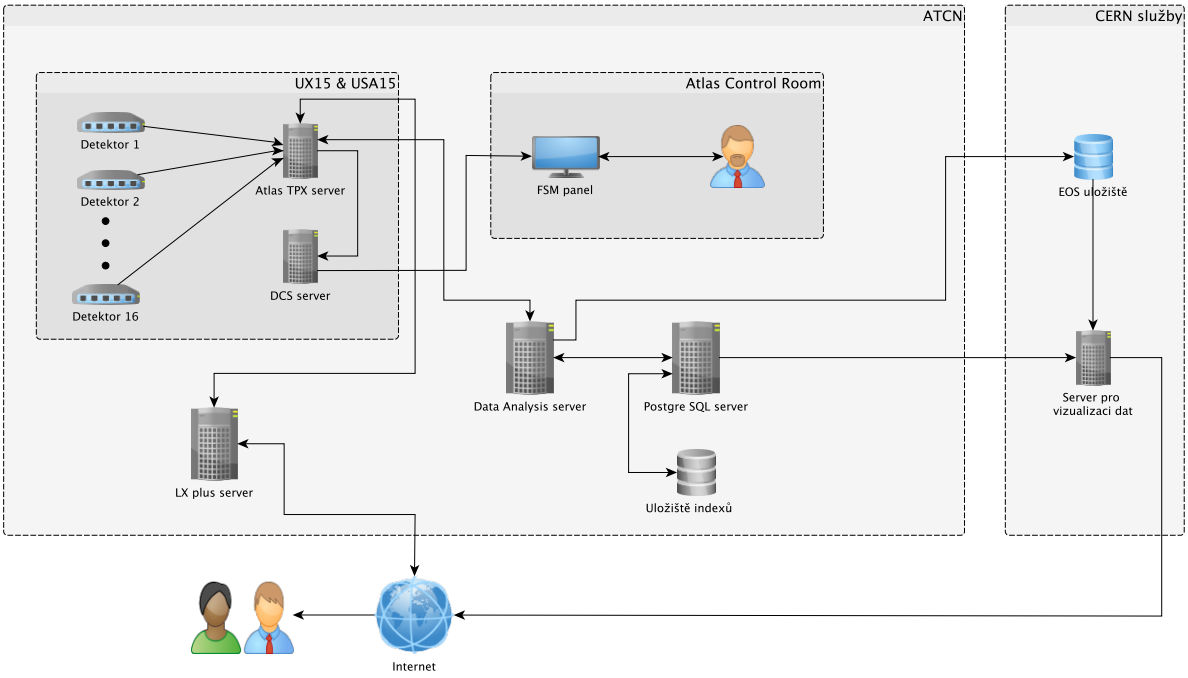
Obrázek 4.7: ATLAS TPX - fotografie hw komponent

4.3 Softwarová architektura sítě ATLAS TPX

Na obrázku 4.8 je znázorněn diagram návrhu softwarové architektury sítě ATLAS TPX z pohledu jejího řízení, vizualizace dat a předávání stavových informací CERNu. Diagram je členěn do dvou základních částí - ATCN[11] (technická síť ATLAS experimentu, která je oddělena od zbytku ATLAS sítě a obsahuje systémy pro vyčítání dat a pro řízení, včetně TDAQ[12] a DSC [14]) a CERN služby, které poskytují perzistentní úložiště dat a web server pro jejich vizualizaci.

Popis architektury z pohledu řízení: Na obrázku 4.8 se nachází ATLAS TPX server, který je umístěn v serverové místnosti (USA15) ATLAS experimentu, umístěné cca 100 m pod zemským povrchem. Tento server pomocí komunikačního protokolu (specifikovaném v 4.4.1) řídí činnost detektorů (nastavování parametrů, ovládání akvizice apod.). Zároveň pomocí JSON REST API poskytuje rozhraní pro své řízení a předávání stavových informací (více v 4.4.2). Díky tomuto rozhraní je možné činnost serveru řídit z ATCN sítě. Pro potřeby vzdáleného ovládání mimo síť ATCN slouží `LX plus server`, který zajišťuje spojení vytvořením SSH tunelu.
Předávání stavových informací zajišťuje `DCS server` pomocí API `ATLAS TPX serveru`. Hlavním úkolem systému DCS je získávání stavových informací ze všech experimentů a detektorů homogenním způsobem a také interakce s LHC (předávání dat luminozitě, stavu svazku urychlovače, radiační pozadí apod.). Tato data jsou dále předávána do místnosti ATLAS Control Room, která se nachází na povrchu. Tam jsou tato data operátorům prezentována pomocí `FSM` panelu, což je aplikace vizualizující stromovou strukturu všech systému a detektorů ATLAS experimentu. Každý list této stromové

[11] z angl. ATLAS Technical Control Network
[12] z angl. Trigger and Data Aquisition (trigger a akvizice dat)



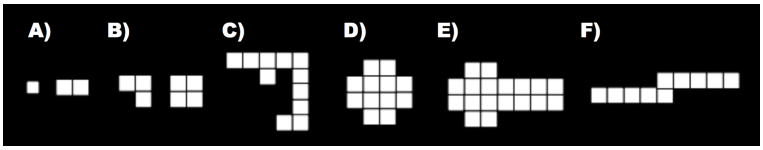
Obrázek 4.8: ATLAS TPX - diagram softwarových komponent

struktury (detektor, senzor atd.) má několik proměnných, z nichž každá má předem definované intervaly s příslušnými stavy (`OK`, `WARNING`, `ERROR`, `FATAL` atd.). Výhodou této struktury je, že pokud kterýkoliv list změní svůj stav, tak se tato informace propaguje přes všechny nadřazené uzly, tudíž odhalení případné chyby je pro operátory mnohem snazší.

Popis architektury z pohledu analýzy a vizualizace dat: Když kterýkoliv detektor dokončí akvizici snímku, tak vygeneruje a pošle asynchronní událost `ATLAS TPX serveru` s informací, že data jsou připravena k vyčtení. Následně server vyčte snímek z detektoru (i s jeho metadaty), zpracuje a připojí k němu informace o nastavení detektoru. Poté jsou data přenesena do `Data analysis serveru`, což je možné dvěma[13] způsoby:

1. `ATLAS TPX server` uloží získaná data v textové podobě do lokálního (či síťového) datového úložiště, odkud jsou přenesena do `Data analysis serveru` pomocí automatického kopírovacího skriptu.
2. Druhou možností je přenesení dat pomocí JSON REST API protokolu, který je `Data analysis serverem` implementován. Tento druhý způsob minimalizuje prodlevu mezi dobou pořízení snímku a následném zpracováním `Data analysis serverem` a dostupností jeho vizualizace pomocí web serveru. Zároveň přináší úsporu objemu přenesených dat.

[13] V současné době je používán první způsob. Data analysis server a všechny s ním související systémy (web server, databáze s indexem a vlastní úložiště dat) jsou prozatím umístěny v ÚTEF ČVUT v Praze.



Obrázek 4.9: Cluster analýza - 6 základních typů clusterů (převzato z [17])

Hlavní úlohou `Data analysis serveru` je provedení tzv. Cluster analýzy [4]. Jde o proces, při kterém jsou z každého snímku získány shluky sousedních pixelů (tzv. clusterů), které mají nenulovou hodnotu. Z těchto clusterů, resp. z jejich tvaru a celkové deponované energie částice (pokud zasažené pixely operovaly v TOT módu) je možné zjistit typ částice, která danou událost způsobila. Na obrázku 4.9 můžete vidět 6 základních typů clusterů, kde

(a) je tzv. `DOT`, způsobený fotony, či elektrony o energii do 10 keV

(b) je tzv. `SMALL BLOB`, způsobený fotony, či elektrony s energií vetšinou nad 10 keV

(c) je tzv. `CURLY TRACK`, způsobený elektrony do 10 MeV

(d) je tzv. `HEAVY BLOBS`, způsobený těžce nabitými částicemi (např. alfa)

(e) je tzv. `HEAVY TRACK`, způsobený těžce nabitými částicemi (např. protony)

(f) je tzv. `STAIGHT TRACK`, způsobený těžce nabitými částicemi (muony apod.)

Po dokončení analýzy dat, jsou data uložena do `ROOT`[14] souborů. Každou hodinu je pro každý detektor vytvořen nový `ROOT` soubor. `ROOT` je framework, který je vyvíjen v CERN a je určen pro ukládání velkého objemu dat a jejich následnou analýzu. Vygenerované soubory jsou ukládány do `EOS úložiště`, což je služba pro perzistentní ukládání `ROOT` souborů provozovaná CERN.
Jelikož každý `ROOT` soubor má velikost řádově v jednotkách GB, jakékoliv operace nad nimi (například vyhledávání) jsou velice časově náročné. Z tohoto důvodu vznikla `PostgreSQL` databáze s indexem na jednotlivé clustery, obsažené v `ROOT` souborech. Pro vizualizaci dat slouží web server, který pomocí databáze s indexem a `ROOT` souborů poskytuje online výsledky cluster analýzy a další informace.

4.4 Řídící software a jeho implementace

Tato kapitola je věnována návrhu a implementaci řídícího software sítě ATLAS TPX, který je nasazen na ATLAS TPX serveru (viz obr. 4.8). Úkolem tohoto software je zajištění komunikace s detektory, zejména pak řízení akvizice, nastavování parametrů a vyčítání dat - tato komunikace bude popsána v kapitole 4.4.1. Další úlohou tohoto software je poskytování rozhraní pro řízení své činnosti a pro předávání stavových informací o ATLAS TPX síti. Toto rozhraní je implementováno pomocí `JSON REST API serveru` a detailně bude popsáno v kapitole 4.4.2.

[14] <https://root.cern.ch/>

4.4.1 Řízení detektorů

Řízení detektorů je zajištěno pomocí komunikačního protokolu 4.4.1.1 mezi minipočítačem `Raspberry Pi` a `ATLAS TPX` serverem. Tento protokol využívá jak synchronní tak asynchronní příkazy 4.4.1.3. Pro účely vývoje a testování byl vyvinut emulátor detektoru 4.4.1.5.

4.4.1.1 Komunikační protokol

Tabulka 4.1 znázorňuje strukturu komunikačního rámce (tzv. paketu) tohoto komunikačního protokolu pomocí posloupnosti bytů z pohledu ATLAS TPX serveru, resp. z pohledu řídícího software. V horní části se nachází struktura odchozího paketu, kde:

0x55 [15] je tzv. `START BYTE`, který značí začátek paketu,

CMD je typ příkazu (tzv. `COMMAND TYPE`) - viz tabulka přehledu příkazů 4.2,

SIZE 1..4 je pole vždy o velikosti čtyřech bytů značících velikost (resp. počet bytů) položky `DATA`, zakódovaných v `BIG ENDIAN`,

DATA 1 .. DATA n je pole vlastních přenesených dat o velikosti n,

0xAA `STOP BYTE`, který značí konec paketu.

Struktura odchozího paketu je velice podobná, až na byte `ERR`, který oproti příchozímu paketu obsahuje. Tento byte může nabývat hodnoty 0x00 (když zpracování požadavku serveru detektorem proběhlo bez chyby), nebo 0x01 (jinak).

0x55	CMD	SIZE 1	SIZE 2	SIZE 3	SIZE 4	DATA 1 .. DATA n	0xAA	} odchozí paket
0x55	CMD	ERR	SIZE 1	SIZE 2	SIZE 3	SIZE 4	DATA 1 .. DATA n	0xAA } příchozí paket

Tabulka 4.1: Komunikační protokol - struktura paketů z pohledu serveru

4.4.1.2 Popis příkazů komunikačního protokolu

Následuje stručný popis příkazů komunikačního protokolu z pohledu obsahu vlastních přenesených dat odchozího a příchozího paketu (viz tabulka 4.1)

0x01 - Ping : Příkaz pro ověření spojení s detekorem. Na základě rozdílu času odeslání a přijetí paketu je vypočtena prodleva spojení v ns (tzv. ping)

Odchozí data: nic

Příchozí data: nic

[15] značení v hexadecimální soustavě

Hodnota příkazu	Název příkazu
0x01	Ping
0x02	Get status of the detector
0x03	Reset of the device
0x04	Set Bias and Timepix Clock
0x05	Get Bias and Timepix Clock
0x06	Set Pixel Configuration
0x07	Get Pixel Configuration
0x08	Set DAC
0x09	Get DAC
0x0A	Perform Digital Test
0x0B	Perform Acquisition
0x0C	Readout Measured Data
0x0D	Direct FITPix Command
0x0E	Stop acquisition
0xFD	Asynchronous Event from device
0xFE	Reboot device
0xFF	Shut down

Tabulka 4.2: Komunikační protokol - přehled příkazů

0x02 - Get status of the detector : Příkaz pro zjištění stavu detektoru.

Odchozí data: nic

Příchozí data (2 B): GST MST

- GST - General Status (obecný status)
  - 0x00 - Ok
  - 0x01 - Chyba detekčního čipu
  - 0x02 - Obecná chyba
- MST - Measurement Status (měřící status)
  - 0x00 - Nečinný
  - 0x01 - Probíhá akvizice
  - 0x02 - Čekání na trigger
  - 0x03 - Data připravena k vyčtení
  - 0x04 - Chyba akvizice

0x03 - Reset of the device : Tento příkaz slouží pro vyresetování FPGA a dalších řídících struktur detektoru.

Odchozí data: nic

Příchozí data: nic

0x04 - Set Bias and Timepix clock : Příkaz nastavující napětí (Bias) na obou Timepix čipech a jejich měřící frekvenci (Timepix clock)

Odchozí data (24 B**):** BIAS1(8 B) BIAS2(8 B) CLK(8 B)

- BIAS1, BIAS2 - hodnota napětí pro Timepix čipy (zakódovaná jako 64-bitové double-precision číslo dle standartu `IEEE 754`)
- CLK - Měřící frekvence obou Timepix čipů (zakódovaná jako 64-bitové double-precision číslo dle standartu `IEEE 754`)

Příchozí data: nic

0x05 - Get Bias and Timepix clock : Příkaz pro vyčtení úrovně napětí z obou Timepix čipů a jejich měřící frekvence

Odchozí data: nic

Příchozí data (24 B**):** BIAS1(8 B) BIAS2(8 B) CLK(8 B)

- BIAS1, BIAS2 - hodnota napětí pro Timepix čipy (zakódovaná jako 64-bitové double-precision číslo dle standartu `IEEE 754`)
- CLK - Měřící frekvence obou Timepix čipů (zakódovaná jako 64-bitové double-precision číslo dle standartu `IEEE 754`)

0x06 - Set Pixel Configuration : Příkaz pro nastavení konfigurace pro každý pixel detektoru

Odchozí data ($(1 + (65536\ nebo\ 2 * 65536))\ B$**):** TYPE(1 B) PIXCFG((65536 *nebo* 2∗ 65536) B)

- TYPE - Výběr čipu, pro který je konfigurace určena
  - 0x00 - Konfigurace určena oboum čipům (délka PIXCFG je $2 * 65536\ B$)
  - 0x01 - Konfigurace určena prvnímu čipu (délka PIXCFG je 65536 B)
  - 0x02 - Konfigurace určena druhému čipu (délka PIXCFG je 65536 B)
- PIXCFG - Pole konfiguračních bytů (každý byte je pro jeden pixel) pro jednotlivé pixely. Délka tohoto pole je závislá na parametru TYPE. Struktura bytu pro konfiguraci pixelu je následovná:
  MASK_BITE(1 b) TEST_BITE(1 b) THL(4 b) MODE(2 b)
  - MASK_BITE - Pixel je aktivní při hodnotě 1, zamaskovaný při 0.
  - TEST_BITE - Slouží pro zapnutí testovacího módu pixelu, pomocí externího generátoru pulzů.
  - THL - Tyto čtyři bity udávají číslo od 0 do 15, které je použito pro posun vůči globální hodnotě THL
  - MODE - Mód pixelu (0 - Medipix, 1 - Time-Over-Treshold, 2 One-Hit, 3 - Time-of-Arrival)

Příchozí data: nic

0x07 - Get Pixel Configuration : Příkaz pro vyčtení konfigurace z každého pixelů detektoru.

Odchozí data (1 B**):** TYPE - Výběr čipu, pro který je konfigurace určena (viz předchozí příkaz)

Příchozí data ($(65536\ nebo\ 2 * 65536)\ B$**):** PIXCFG (viz předchozí příkaz)

0x08 - Set DAC : Příkaz pro nastavení hodnot DAC převodníku detektoru.

Odchozí data (3 *B*): DAC_IDX DAC_VAL DAC_VAL

- DAC_IDX - DAC index
- DAC_VAL - DAC hodnota

Příchozí data: nic

0x09 - Get DAC : Příkaz pro vyčtení hodnoty DAC převodníku detektoru.

Odchozí data (1 *B*): DAC_IDX

- DAC_IDX - DAC index

Příchozí data (2 *B*): DAC_VAL DAC_VAL

- DAC_VAL - DAC hodnota

0x0A - Perform Digital Test : Příkaz pro vykonání testu digitální části detektoru.

Odchozí data: nic

Příchozí data (3 *B*): BPC BPC BPC

- BPC - Počet chybných pixelů detektoru

0x0B - Perform Acquisition : Příkaz pro zahájení akvizice snímku/snímků.

Odchozí data (13 *B*): ACQTM(8 *B*) ACQCNT(4 *B*) TGRMOD

- ACQTIM - Akviziční čas v sekundách (zakódovaný jako 64-bitové double-precision číslo dle standartu `IEEE 754`).
- ACQCNT - Počet snímku (pokud je 0, detektor bude opakovat akvizici s těmito parametry až do jejího manuálního zastavení - viz příkaz `0x0E - Stop acquisition`)
- TRIGMOD - Mód triggeru
  - 0x00 - bez triggeru
  - 0x01 - akvizice zahájena na signál triggeru
  - 0x02 - akvizice ukončena na signál triggeru

Příchozí data: nic

0x0C - Read Measured Data : Tento příkaz slouží pro vyčtení naměřených dat z detektoru.

Odchozí data (6 *B*): DET TYPE FRAME_ID FRAME_ID FRAME_ID FRAME_ID

- DET - selektor Timepix čipu
  - 0x00 - oba čipy
  - 0x01 - jen první čip
  - 0x02 - jen druhý čip
- TYPE - typ vyčítaných dat
  - 0x00 - deserializovaný a derandomizovaný snímek

  - 0x01 - surová data z FPGA
  - 0x02_ID - metadata k danému snímku (akviziční čas, napětí apod.)
- FRAME_ID - ID naměřeného snímku

Příchozí data: Velikost a typ příchozích dat se liší dle použitých parametrů DET a TYPE

- Pro TYPE 0x00 přijde pole bytů hodnot čítačů jednotlivých pixelů (hodnota každého pixelu je reprezentována pomocí dvou bytů)
- Pro TYPE 0x01 přijdou blíže nespecifikovaná surová data z FPGA
- Pro TYPE 0x02 přijdou metadata, vázající se k danému snímku s následující strukturou:
  - UNIX TIME časové razítko začátku akvizice $[ms]$ (5 B)
  - Bias1 - měřící napětí prvního čipu $[V]$ (8 B, double-precision dle `IEEE 757`)
  - Bias2 - měřící napětí druhého čipu $[V]$ (8 B, double-precision dle `IEEE 757`)
  - Měřící frekvence $[Hz]$ (8 B, double-precision dle `IEEE 757`)
  - Doba akvizice $[s]$ (8 B, double-precision dle `IEEE 757`)

0x0D - Direct FPGA Command : Příkaz pro přímé poslání dat do FPGA a získání odpovědi.

Odchozí data: Dle použitého příkazu

Příchozí data: Dle použitého příkazu

0x0E - Stop acquisition : Příkaz pro zastavení aktuálně probíhající akvizice.

Odchozí data (1 B): TYPE

- TYPE: Typ zastavení
  - 0x00 - Zastavení akvizice po dokončení aktuálně pořizovaného snímku
  - 0x01 - Bezprostřední zastavení akvizice

Příchozí data: nic

0xFD - Asynchronous Event From Device : Tento typ příkazu je asynchronní a detektor ho posílá samovolně dle nastalé události - například dokončení akvizice.

Příchozí data (5 B): EVID VAL VAL VAL VAL

- EVID - Typ nastalé události (např. 0x00 pro událost dokončení akvizice)
- VAL - Doplňující data (např. ID snímku pro událost dokončení akvizice)

0xFE - Reboot of the Device : Příkaz pro restartování detektoru.

Odchozí data: nic

Příchozí data: nic

0xFF - Shutdown of the Device : Příkaz pro vypnutí detektoru.

Odchozí data: nic

Příchozí data: nic

4.4.1.3 Synchronní a asynchronní příkazy komunikačního protokolu

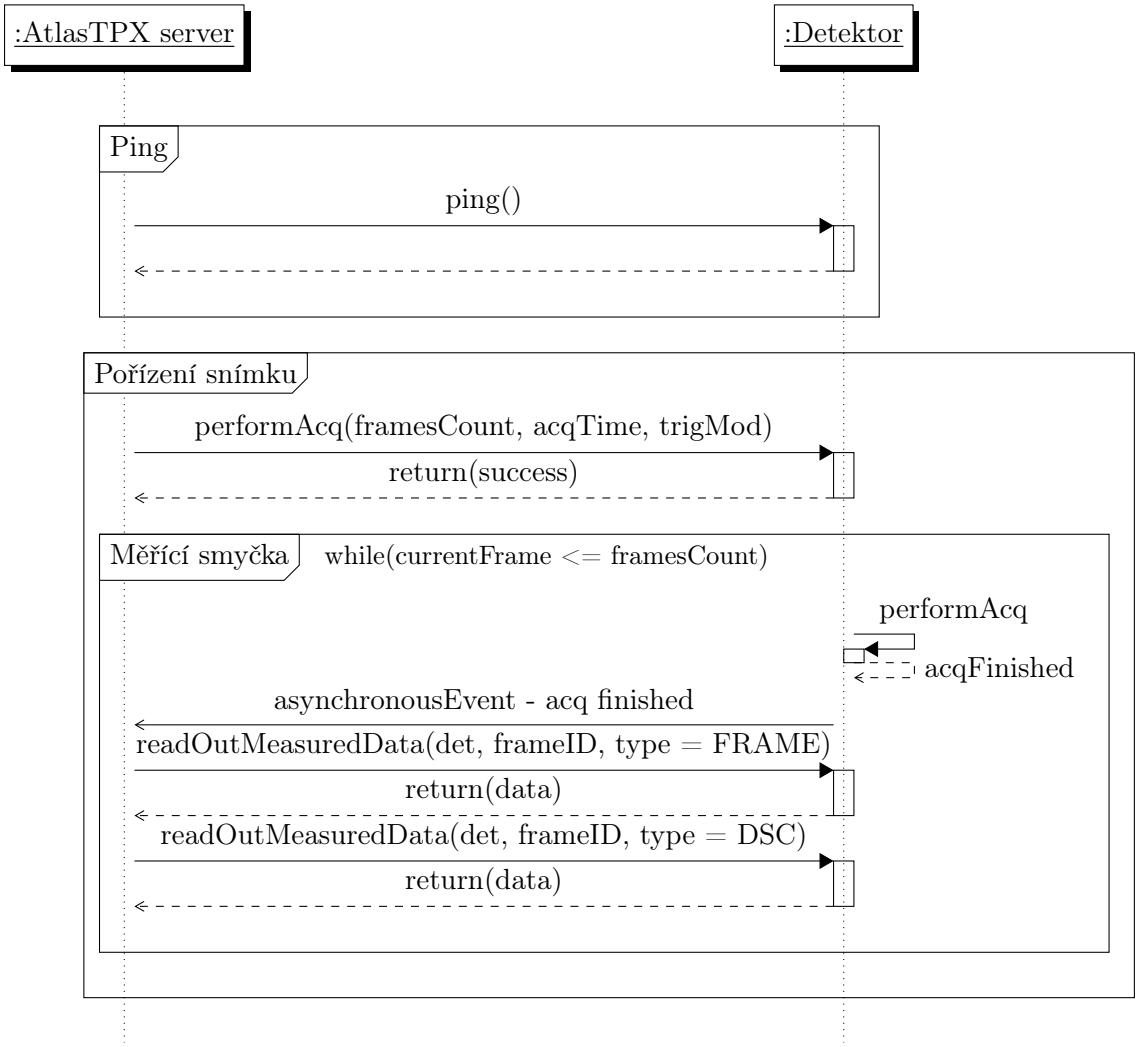
Téměř všechny příkazy komunikačního protokolu jsou synchronní, tzn. server vyšle odchozí paket (s typem příkazu a jeho daty), který detektor zpracuje a bezprostředně vygeneruje a pošle odchozí paket (s příslušnými daty, typem příkazu a informací, zda-li při jeho zpracování došlo k chybě). Příkladem této synchronní komunikace je příkaz ping na obrázku 4.10 nahoře.

Vedle synchronních příkazů existuje i jeden asynchronní - `0xFD - Asynchronous Event From Device`. V současné verzi komunikačního protokolu je tento příkaz využit jen pro oznámení serveru, že detektor dokončil akvizici a má data připravena k vyčtení.

Na obrázku 4.10 je znázorněn příklad kombinace synchronní a asynchronní komunikace pro pořízení snímku. Nejprve server vyšle synchronní příkaz s žádostí o provedení akvizice (s parametry: doba akvizice, počet snímků a mód triggeru), na který hned dostane odpověď. Následuje vyčítací smyčka - detektor udělá akvizici snímku (pokud již všechny neudělal) a vyšle serveru asynchronní příkaz s kódem právě dokončené akvizice a s ID[16] snímku. Tuto asynchronní zprávu server zachytí a provede vyčítací sekvenci (pomocí příkazu `0x0C - Read Measured Data`), jejíž kroky se mohou lišit dle konfigurace. Ve výchozím nastavení tato sekvence vypadá následovně:

1. Vyčtení vlastního snímku (hodnot jednotlivých pixelů).
2. Vyčtení metadat (tzv. `DSC`). Tato data obsahují dodatečné informace ke snímku, jako například přesnou dobu akvizice, čas začátku akvizice, měřící napětí a frekvenci Timepix čipů.

[16] unikátní číslo snímku od spuštění detektoru



Obrázek 4.10: Příklad použití komunikačního protokolu

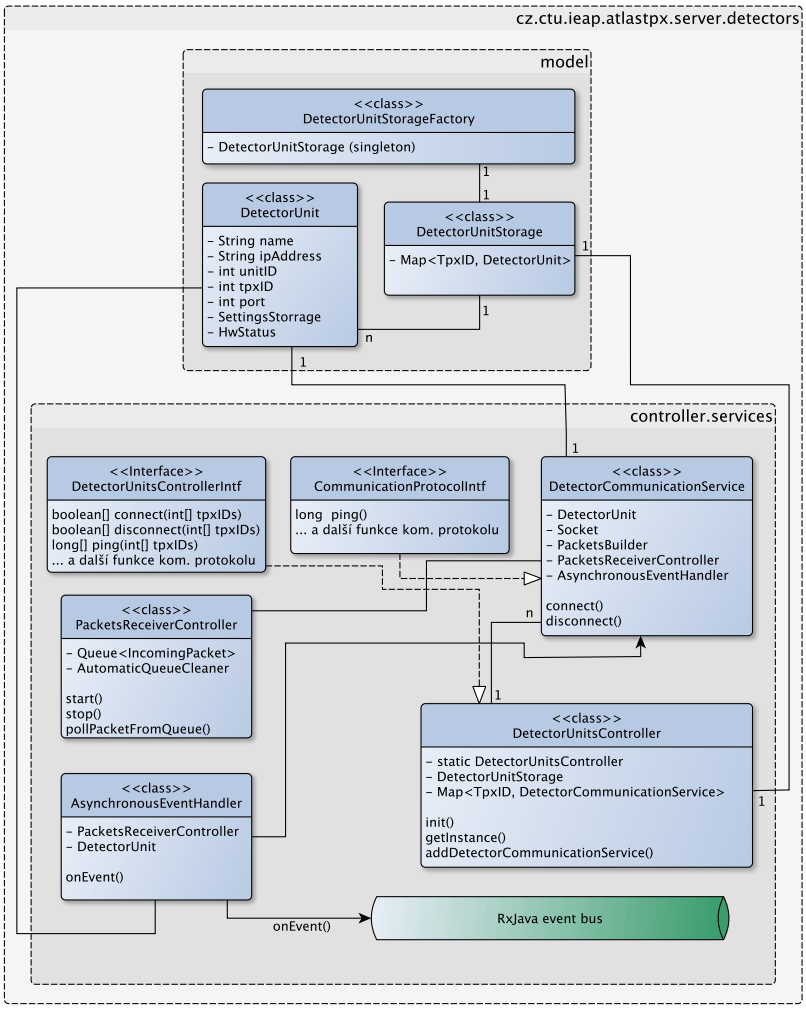
4.4.1.4 Implementace modulu pro řízení detektorové sítě na straně serveru

V této podkapitole bude popsána implementace řízení detektorů v řídícím softwaru sítě ATLAS TPX. Software byl implementován v jazyce `JAVA` za pomoci build nástroje `Maven`[17] a knihoven `Dropwizard`[18], `Retrofit`[19] a `RxJava`[20].

Jak již bylo zmíněno výše, detektory jsou připojeny k ATLAS TPX serveru přes ethernetové rozhraní a pomocí TPC/IP protokolu. Vlastní komunikace je realizována pomocí komunikačního protokolu 4.4.1.1. Z pohledu navazování spojení byla použita architektura klient-server tak, že detektor plní roli serveru a ATLAS TPX server zase klienta. Tato ar-

[17] <https://maven.apache.org/>
[18] <http://www.dropwizard.io>
[19] <http://square.github.io/retrofit/>
[20] <https://github.com/ReactiveX/RxJava>

chitektura byla použita s ohledem na robustnost a stabilitu sítě a z důvodu zachování úlohy navazování spojení v kompetenci ATLAS TPX serveru. Pokud by například došlo k přerušení napájení nebo jiné chybě jednoho z detektorů, ATLAS TPX server by nebyl schopen pokusit se o znovu navázání spojení.



Obrázek 4.11: Diagram tříd modulu pro ovládání detektorů

Na obrázku 4.11 je zobrazen diagram tříd modulu pro práci s detektory řídícího softwaru. Tento diagram obsahuje pouze nejdůležitější třídy, zbytek je k nalezení na přiloženém CD.

Při návrhu tohoto modulu bylo vycházeno z návrhového vzoru Model-View-Controller [3], resp. z jeho modifikace Model-Controller, protože tento modul žádnou prezentační vrstvu nemá.

Hlavní třídou balíčku `model` je třída `DetectorUnit`, která uchovává všechny informace o jednom detektoru, jako například název, tpxID, unitID, ip adresu, port, `SettingsStorage` (úložiště nastavení detektoru, obsahující bias, clock, parametry akvizice, DAC hodnoty atd.)

a `HwStatus` (tento objekt nese informace o posledních naměřeních údajích získaných z detektoru, jako třeba aktuální hodnoty bias, ping, obecný a měřící status apod.).

Pro uchovávání všech instancí třídy `DetectorUnit` slouží singleton [3] třída `DetectorUnitStorage`, která tyto instance uchovává v kolekci HashMap, jejímž klíčem je TpxID (Integer). Instanci tohoto singletonu je možné získat pomocí třídy `DetectorUnitStorageFactory`, resp. pomocí její statické metody `getStorage()`. Před prvním použití je třeba nejprve toto úložiště inicializovat pomocí statické metody `DetectorUnitStorageFactory.initStorage(String initFilePath)`, které se coby parametr předá cesta v souborovém systému k "*.csv"souboru se seznamem detektorů. Každý řádek tohoto souboru reprezentuje jeden detektor a je v následujícím formátu:

název_detektoru;ip_adresa;unit_id;tpx_id;port

V balíčku `controller.services` se nachází vlastní logika tohoto modulu. Třída `DetectorCommunicationService` se stará o vlastní komunikaci s detektorem (jedna instance rovná se jeden detektor). Tato třída obsahuje instanci třídy `DetectorUnit` (předané v konstruktoru), která mimo jiné obsahuje IP adresu a port příslušného detektoru. Tyto parametry se používají v metodě `connect()`, ve které se vytvoří spojení s detektorem (`socket`, `BufferedOutputStream`, `BufferedInputStream` apod.). Pro odpojení detektoru slouží metoda `dicconnect()`. Mimo metod pro síťovou komunikaci třída `DetectorCommunicationService` obsahuje i všechny metody implementované z interface `CommunicationProtocolIntf` - všechny synchronní metody komunikačního protokolu.

Pro příjem příchozích paketů z detektoru vznikla třída `PacketsReceiverController`. Ten ve vlastním vlákně vyčítá proud bytů přicházejících z detektoru a následně provádí jejich parsování, jehož výsledkem je instance objektu `IncomingPacket`. Ten obsahuje typ příkazu, příznak chyby a vlastní data. Takto vzniklý paket je zařazen do fronty příchozích paketů detektoru (implementované jako `ConcurrentLinkedQueue<IncomingPacket>`). Nad touto frontou pracují jednak všechny metody se synchronními příkazy komunikačního protokolu (které sledují, zda-li se v ní v daném časovém intervalu objeví odpověď), ale také instance objektu `AsynchronousEventHandlerController`. Ten ve vlastním vlákně tuto frontu sleduje a objeví-li se paket s typem příkazu `0xFD (Asynchronous Event From Device)`, tak ho z fronty odebere, jeho vlastní data rozparsuje a dále zpracuje. Pokud se například jedná o událost dokončené akvizice, tak `ReadOutService` snímek z detektoru vyčte a pomocí `RxJava event bus` vyšle asynchronní zprávu napříč celou aplikací s vyčteným snímkem. Zde byl použit návrhový vzor Producer - Consumer, kde `AsynchronousEventHandlerController` představuje Producer a na kterémkoliv jiném místě aplikace se Consumer (možno i více Consumerů) může zaregistrovat ke sledování událostí v event bus. Příkladem takového consumera může být `FrameSaverController`, který se postará o uložení získaného snímku (viz kapitola 4.4.3).

Aby bylo možné pohodlně ovládat více detektorů současně, vznikla třída `DetectorUnitsController`. Ten implementuje interface `DetectorUnitsControllerIntf`, jehož metody pokrývají veškerou funkcionalitu detektoru, jako třeba metody k navázání a ukončení spojení, ale také všechny metody komunikačního protokolu. Všechny tyto metody mají jeden společní parametr - `int[] tpxIDs` (pole TpxID detektorů). Tento parametr určuje, nad kterými detektory se má daná metoda hromadně vykonat. Jejich výstupem je pole, jehož typ se liší dle příkazu (například pro metodu connect je to pole boolean proměnných - značící úspěch připojení, pro ping zase double čísel, udávajících odezvu spojení v ns). Tato třída

je typu singleton a její instanci lze získat pomocí metody `getInstance()`. Po spuštění aplikace je však třeba tento singleton nainicializovat pomocí metody `init()`, která pomocí třídy `DetectorUnitStorageFactory` vytvoří datovou strukturu s `DetectorCommunicationService` jednotlivých detektorů.

4.4.1.5 Emulátor detektoru

Pro účely vývoje a testování řídícího software pro `ATLAS TPX server` byl vyvinut plnohodnotný emulátor detektoru. Emulátor je rovněž napsán v jazyce `JAVA` a jeho zdrojové kódy a spustitelný `jar` soubor jsou na přiloženém CD.

Emulátor se spouští se dvěma povinnými parametry, kde:

1. parametr je port, na kterém server emulátoru naslouchá (celé číslo) a
2. parametr je pravděpodobnost ve formě desetinného čísla, s jakou při zpracovávání požadavku bude simulována chyba, kde 0 znamená bez chyb a 1 samé chyby.

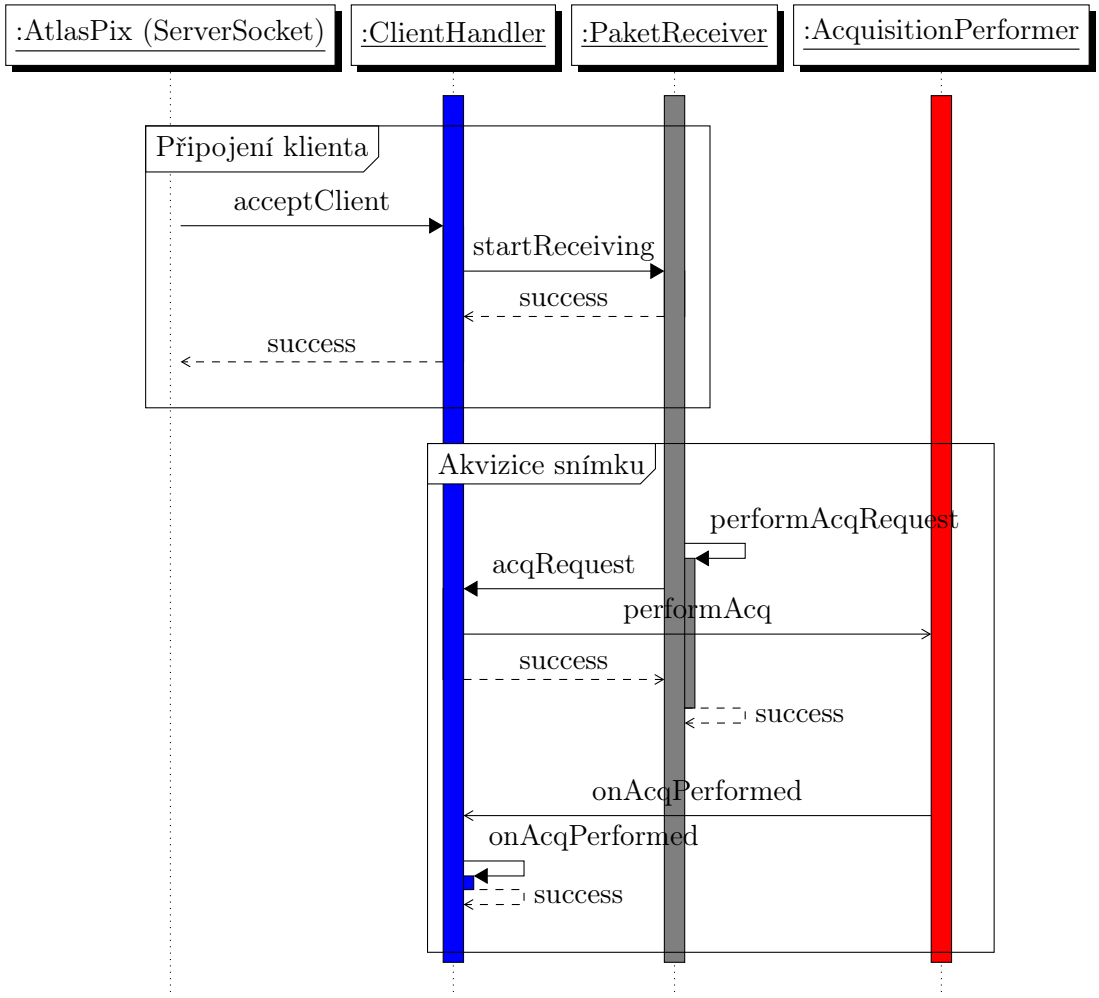
Na obrázku 4.12 je zobrazen sekvenční diagram dvou případů užití tohoto emulátoru.

První případ užití znázorňuje připojení klienta (např. `ATLAS TPX serveru`). Ve třídě `AtlasPix`, resp. v její metodě `startServer()` se nachází nekonečná smyčka, ve které je pomocí instance třídy `ServerSocket` navázáno spojení s novými klienty. Když je spojení s klientem navázáno, je vytvořen klientský socket, pomocí kterého dojde k vytvoření instance třídy `ClientHandler`. Ta se v samostatném novém vláknu stará o obsluhu nově připojeného klienta. Zároveň `ClientHandler` vytvoří instanci třídy `PaketReceiver`, která čte proud bytů přijatých od klienta a parsuje jej do objektů typu `IncomingPacket`.

Druhý případ užití znázorňuje proces zpracování žádosti o snímek a jeho vygenerování. Poté, co `PaketReceiver` přijme nový paket, tak ho vloží do fronty paketů. Tuto frontu z druhé strany čte a zpracovává `ClientHandler`. Všechny příchozí pakety od klienta mohou obsahovat jen synchronní příkazy komunikačního protokolu, takže může být okamžitě nasimulován příslušný stav emulátoru a vygenerována odpověď klientovi. V případě akvizice snímku `PaketReceiver` odchytí nový paket, obsahující žádost o provedení akvizice, který předá do fronty příchozích paketů. Když se `ClientHandler` dostane k jeho zpracování, tak vytvoří objekt typu `AcquisitionRequest` a předá ho instanci třídy `AcquisitionPerformer`. Ten žádost o akvizici zařadí do příslušné fronty, kde čeká, až se samostatné akviziční vlákno dostane k jejímu zpracování. Při zpracovávání žádosti o akvizici je vygenerována náhodná matice hodnot pixelů (o velikosti 256×512) a také příslušná metadata, jako například akviziční čas, bias obou čipů (s přičtenou náhodnou veličinou, simulující fluktuaci napětí na křemíkovém povrchu detekčních čipů) apod. Po vygenerování těchto hodnot se akviziční vlákno uspí na dobu akvizice, aby se chování emulátoru co nejvíce přiblížilo skutečnému detektoru.

Následně je připojenému klientovi poslána asynchronní zpráva o dokončené akvizici. Tato zpráva obsahuje příznak, že naměřená data jsou připravena k vyčtení a také ID vygenerovaného snímku.

Poté klient pošle příkaz pro vyčtení naměřených dat (resp. dva příkazy - jeden pro vlastní snímek a druhý pro metadata), což bylo popsáno v kapitole 4.4.1.3 - viz obr. 4.10.



Obrázek 4.12: Emulátor detektoru - sekvenční diagram připojení klienta a pořízení snímku

4.4.2 REST API server

Řídící software ke svému vzdálenému ovládání poskytuje rozhraní přes `JSON REST`[21] `API`. Pro toto rozhraní byla použita knihovna Dropwizard[7], která vznikla složením několika dalších knihoven, zejména pak:

Jetty je open-source software, v současné době vyvíjen Eclipse Foundation. Tato knihovna obsahuje Java HTTP webový serverů, který na rozdíl od běžných webových serveru (které vetšinou přes HTTP protokol poskytují soubory koncovým uživatelům) byl navržen pro strojově orientovanou komunikaci.

Jersey je open-source knihovna (dle [7] vycházející z `JAX-RS`[22]), která byla použita pro zajištění REST API. Tato knihovna umožňuje elegantně pomocí anotací mapovat HTTP

[21] z angl. Representational State Transfer
[22] <http://jcp.org/en/jsr/detail?id=311>

dotazy na jednoduché Java objety.

Jackson je výkonný nástroj pro práci s `JSON`[23] objekty v jazyce Java. Tato knihovna umožňuje ergonomicky mapovat data v JSON do Java objektů pomocí anotací.

V následujících podkapitolách je popsána implementace knihovny Dropwizard do `ATLAS TPX serveru` a její provázanost na modul pro řízení detektorů (4.4.1.4).

4.4.2.1 Konfigurace a spuštění serveru

Server je možné spustit z příkazové řádky pomocí příkazu `"java -jar atlasTpxServer.jar server config.yml"`, kde `atlasTpxServer.jar` je spustitelný binární soubor serveru, `server` je povinný parametr (znamenající spuštění programu v módu server) a `config.yml`, což je také povinný parametr, udávající cestu v souborovém systému ke konfiguračnímu souboru serveru.

V příloze B naleznete strukturu tohoto konfiguračního souboru, ve kterém je možné nastavit následující:

Umístění tabulky s detektory (detectorsConfigPath - viz B řádek 8)
Tento parametr udává cestu v souborovém systému ke konfiguračnímu "*.csv"souboru, obsahujícímu tabulku všech detektorů a jejich parametrů, popsanou v 4.4.1.4.

Nastavení webového serveru (server - viz B řádek 11)
V rámci tohoto objektu je možné nastavit použitý typ protokolu spojení (`HTTP`/`HTTPS`), port, počty vláken a další.

Logování (logging - viz B řádek 30)
Pomocí tohoto parametru je možné nastavit úroveň a cíl vytvářených logů. V přiloženém konfiguračním souboru bylo použito dvojího logování a to na standardní výstup a do souboru (oba úrovně `INFO`). Při logování do souboru je možné navíc nastavit automatickou archivaci.

Automatické vyčítání snímků (readOutDataAutomatically - viz B řádek 54)
Tento parametr typu `boolean`. Když přijde asynchronní událost z detektoru o datech připravených k vyčtení, dojde k automatickému vyčtení dat (při hodnotě parametru `true`), nebo je třeba data vyčíst manuálně (při hodnotě `false`).

Výstupní adresář (outputDir - viz B řádek 57)
Parametr udávající cestu v souborovém systému pro ukládání dat z detektorů.

Formát výstupních dat (outputFramesType - viz B řádek 68)
Tento parametr obsahuje pole výstupních formátů dat, ve kterých získaná data z detektorů budou ukládána. V této verzi programu je podporovaný jediný formát - `MULTIFRAME`.

Konfigurace data serveru (v B od řádku 77)
Na tomto místě je možné nastavit parametry (url a port) serveru, sloužícího pro příjem a zpracování naměřených dat a zda-li má být použit. Více o tomto serveru bude zmíněno v 4.4.3.1.

[23] JSON je v dnešní době standardem pro zápis a výměnu strukturovaných, člověkem i strojem čitelných dat.

4.4.2.2 Metody poskytované serverem

Tento JSON REST API server poskytuje metody pro získání stavových informací `ATALS TPX serveru` a také pro jeho řízení, zejména pak pro ovládání detektorů. Seznam všech těchto metod s jejich popisem je k nalezení v dokumentaci na přiloženém CD.

URL schéma jednotlivých metod je následující - `protokol://host:port/vx/skupina/metoda`, kde

protokol je použitý protokol (`HTTP/HTTPS`),

host je doména, či IP adresa serveru,

port - port serveru,

vx je verze API, na které komunikace bude probíhat (současná verze je "v1"),

skupina a metoda udávají cestu a název metody (u některých metod stačí zadat pouze skupinu).

Server byl implementován za použití následujících standardů:

- Všechny zde popsané metody jsou založeny na HTTP metodě POST.
- Všechna předávaná data jsou ve formátu `JSON`.
- Všechny časy a data jsou reprezentovány v milisekundách od 1.1.1970 (tzv. UNIX Time) a časové intervaly v sekundách, není-li specifikováno jinak.
- Server může vracet zprávy s následujícími HTTP kódy: 200 (OK), 400 (Bad Request), 404 (Not Found) a 500 (Internal Server Error).

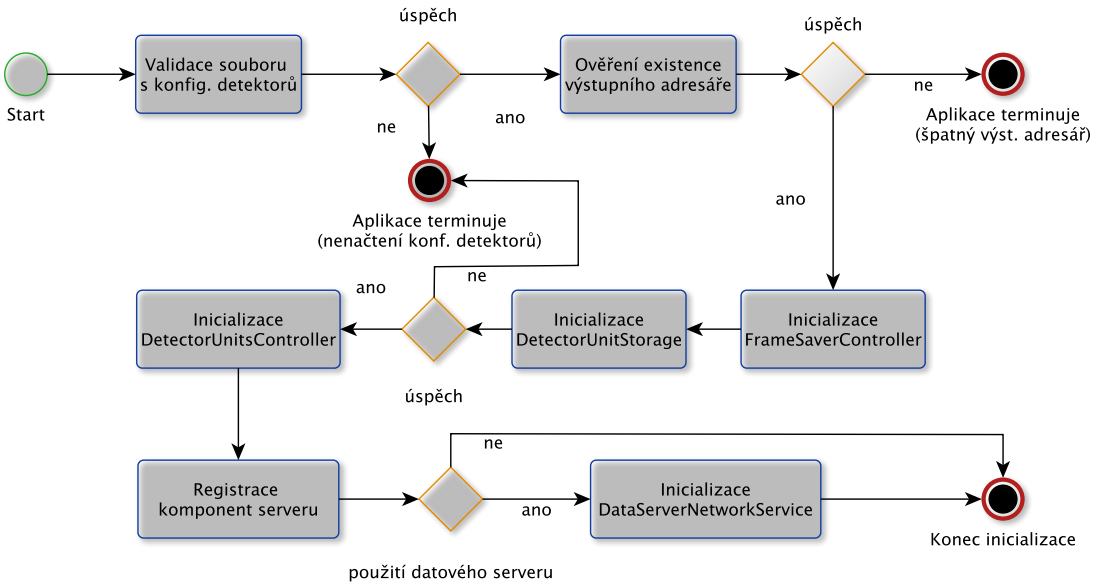
4.4.2.3 Implementace serveru

Server byl implementován za použití knihovny `Dropwizard` a veškeré s ním související třídy jsou v balíčku `rest_server`.

V tomto balíčku se nachází třída `RestServerApplication`, která obsahuje statickou metodu `main()`, která slouží ke spuštění celé aplikace. Zároveň tato třída je potomkem třídy `io.dropwizard.Application` a přepisuje její metodu `run()`, ve které dochází ke konfiguraci serveru, spuštění jeho podpůrných služeb a k registraci komponent REST API serveru (v URL výše uvedených jako skupina).

Na obrázku 4.13 je znázorněn BPMN[24] diagram procesu inicializace REST API serveru (tělo metody `run()` zmíněné výše). V rámci tohoto procesu nejprve dojde k validaci existence konfiguračního "*.csv"souboru, obsahujícím tabulku s detektory a jeho parametry (viz 4.4.2.1), pokud tato validace selže, program vypíše chybovou hlášku a ukončí se. Následuje ověření platnosti výstupního adresáře (pokud selže, program terminuje). Poté dojde k inicializaci `FrameSaverController`(viz kapitola o zpracování a ukládání dat 4.4.3) a `DetectorUnitStorage` (úložiště všech `DetectorUnit`, již zmíněné v 4.4.1.4). Pokud se toto

[24] z angl. Business Process Model and Notation



Obrázek 4.13: BPMN inicializace REST API serveru

úložiště nepodaří vytvořit (z důvodu chyby parsování tabulky detektorů) aplikace opět terminuje. Dalším krokem je inicializace nástroje pro řízení detektorů (`DetectorUnitsController`) a registrace jednotlivých komponent REST API serveru. Na závěr procesu inicializace dojde k vytvoření spojení s datovým serverem (viz 4.4.3.1) pro odesílání naměřených dat, je-li v konfiguračním souboru serveru povolen.

4.4.3 Zpracování a ukládání dat

Zpracovávání a ukládání dat, resp. získaných snímků z detektoru, bylo navrženo s ohledem na modularitu a možnost rozšíření o podporu nových výstupních formátů. V současné verzi je podporován jeden formát, a to tzv. `Multiframe`. Tento formát spočívá v ukládání snímků do tří souboru a to:

Frame File v tomto souboru je ukládán vlastní snímek - hodnoty jednotlivých pixelů. Struktura tohoto souboru je taková, že při ukládání snímku jsou po řádcích zapsány indexy a hodnoty jednotlivých pixelů (oddělených znakem tabulátoru a mezerou), které mají nenulovou hodnotu (z důvodu úspory místa). Jednotlivé snímky jsou odděleny novým řádkem se znakem `'#'`. Tento soubor má příponu `"*.txt"`.

Description File Tento soubor nese metadata o snímku, jako například akviziční čas, časové razítko začátku akvizice, ale také hardwarové parametry detektoru v době akvizice (napětí detekčního čipu, měřící frekvenci, hodnoty DAC apod.). Každý snímek v tomto souboru začíná svým ID (začínajícím od nuly, unikátní v rámci souboru), následovaným výčtem parametrů, oddělených prázdnými řádky. Každý parametr je uložen na tři řádky, kde

1. řádek obsahuje název parametru, jeho jednotku, popř. další informace.
2. řádek udává datový typ a velikost vlastní hodnoty parametru - třetího řádku.
3. řádek obsahuje hodnotu parametru.

Přípona tohoto souboru je "*.txt.dsc".

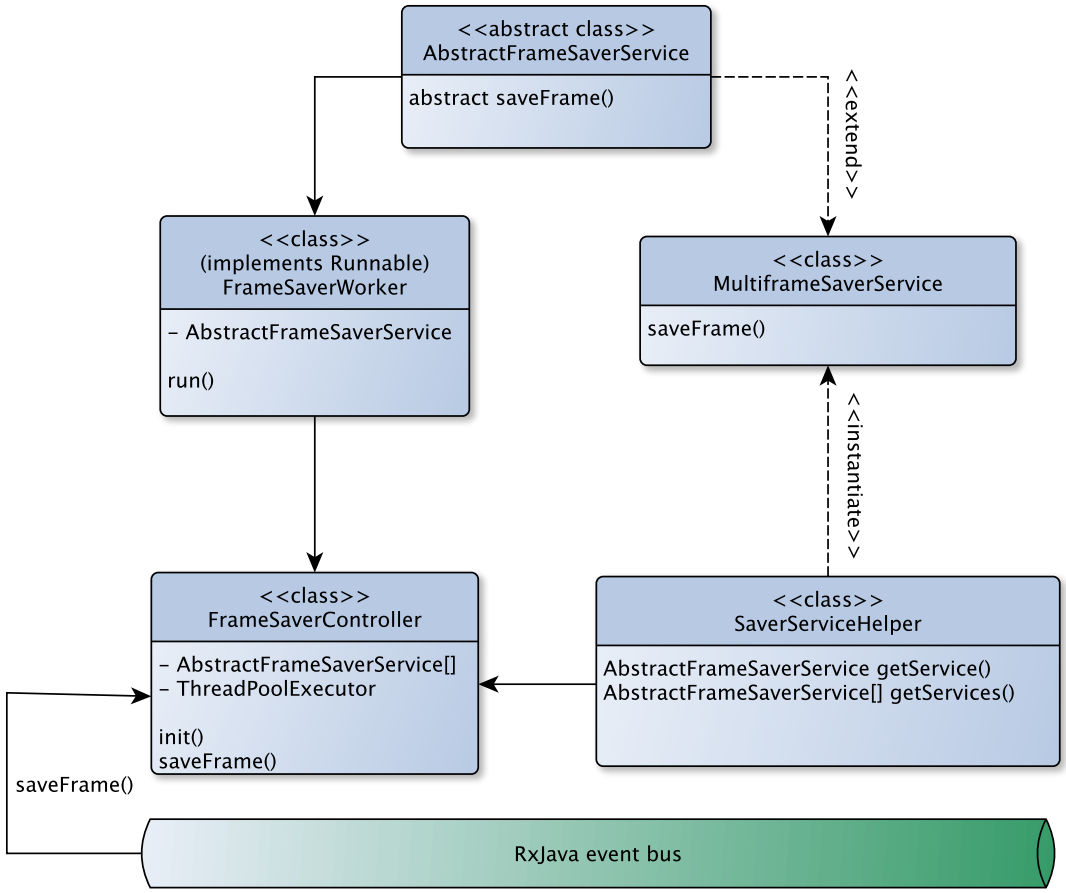
Index File Třetím souborem je tzv. `Index File`. Tento soubor nese binární informace, které spojují `Data File` a `Description File`. Pro každý snímek je do tohoto souboru zapsána bytová adresa začátku snímku v obou výše zmíněných souborech (pro každý 8 B). Z důvodu možného rozšíření do budoucna a zachování zpětné kompatibility, těchto 16 B je následováno dalšími 8 B (s nulovou hodnotou). Přípona tohoto souboru je "*.txt.idx".

Když ATLAS TPX server odchytí asynchronní událost od nějakého z detektorů o tom, že daný detektor dokončil akvizici a že má data připravena k vyčtení, postará se o jejich vyčtení a vygeneruje objet typu `Frame`. Tento objekt obsahuje všechna data, vázající se k dané akvizici, jako například hodnoty jednotlivých pixelů, ale i některá metadata. Po vygenerování objektu `Frame` je napříč celou aplikací vyslána asynchronní událost o této skutečnosti, obsahující právě vygenerovaný objekt typu `Frame`, pomocí `RxJava event bus`.

Tato asynchronní událost je odchycena a zpracována pomocí pomocí singleton třídy `FrameSaverController` - viz obr. 4.14. Ta je po spuštění aplikace inicializována (viz 4.4.2.3). Tomuto procesu jsou předány všechny výstupní typy dat, uvedené v konfiguračním souboru 4.4.2.1. Pomocí třídy `SaverServiceHelper` dojde k vytvoření instancí tříd všech příslušných služeb pro ukládání dat (dědících od třídy `AbstractFrameSaverService`).

Jelikož každý z 16 detektorů může generovat několik snímků za sekundu, bylo třeba vyvinout robustní řešení, které se bude přizpůsobovat dané zátěži. Za těmito účely byl implementován `ThreadPoolExecutor`, který sleduje a vykonává frontu úkolů (instancí třídy `FrameSaverWorker`, implementujících interface `Runnable`). Dle velikosti této fronty, resp. podle počtu snímků čekajících na uložení, je počet pracujících vláken dynamicky měněn (ve stávající konfiguraci od 3 do 50).

Když je zachycena asynchronní události s vyčteným snímkem, dojde k vytvoření instance třídy `FrameSaverWorker` pro každou ukládací službu (dědící od `AbstractFrameSaverService`, např. `MultiframeSaverService`) a následnému zařazení této instance do fronty, kterou obsluhuje již výše zmíněný `ThreadPoolExecutor`.



Obrázek 4.14: Diagram tříd komponenty pro ukládání dat ATLAS TPX serveru

4.4.3.1 Datový server

Současně s možností ukládání dat `ATLAS TPX serverem` na lokální, či síťové úložiště je možné i jejich přímé odeslání datovému serveru (viz 4.3), pomocí jeho API, což je předmětem této podkapitoly.

Přehled všech metod s jejich technickými specifikacemi je k nalezení v dokumentaci na přiloženém CD, zde je uveden jen jejich stručný popis.

Kvůli optimalizaci přenášeného obejmu dat byl navržen úsporný komunikační protokol, který odstraňuje redundanci přenášených dat (oproti `Multiframe` formátu, popsaném výše). Toho je dosaženo především tím, že s každým snímkem jsou přenášena jen data, přímo související a unikátní k dané akvizici. To jsou zejména vlastní hodnoty jednotlivých pixelů, časové razítko začátku akvizice, napětí v době akvizice na polovodičovém povrchu detekčních čipů apod. Mezi údaje, které se v průběhu akvizice nemění (a tudíž je zbytečné je s každým snímkem přenášet) patří konfigurace pixelů, měřící frekvence, hodnoty DAC apod.

Metoda Status Metoda status slouží pro získání základních informací datového serveru,

jako třeba volné místo v jeho úložišti, ale také informace o přenesených snímcích (celkový počet, zpracováno, atd.).

Metoda pro odeslání snímků v rámci této metody je přenesen vlastní snímek a příslušná metadata s ním související, uvedená výše. Zde je dobré zmínit, že oproti formátu `Multiframe` jsou vyloučeny pixely ne s nulovou, nýbrž s nejčetnější hodnotou. Tím je docíleno minimalizace počtu přenesených pixelů. Tento mechanismus je účinný především na přeexponované snímky. Jako jeden z parametrů se rovněž odesílá i ID konfigurace, získané z datového serveru - viz další bod.

Metoda pro odeslání konfigurace Touto metodou je přenášena statická konfigurace jednotlivých detektorů. Tuto metodu je třeba provést nad všemi detektory před zahájením odesílání snímků a potom až při změně některého z parametrů konfigurace. Výstupem této metody je seznam ID, které byly jednotlivým konfiguracím přiděleny datovým serverem a které je třeba datovému serveru posílat společně se snímky.

Spojení s tímto serverem zajišťuje modul `data_sender`, resp. jeho třída `DataSenderController`. Jedná se opět o singleton, který při své inicializaci vytvoří dvě fronty - frontu snímků a frontu konfigurací (obě implementované, jako `ConcurrentLinkedQueue`) a také vytvoří mapu (mapující ID interních konfigurací detektorů na ID, získané z datového serveru). Zároveň dojde ke spuštění samostatného vlákna, které tyto fronty sleduje a jejich položky odebírá a posílá datovému serveru.

Odesílání dat datovému serveru je realizováno pomocí knihovny `Retrofit`[25], která poskytuje nástroje pro implementaci jednoduchého `HTTP` klienta.

[25] <http://square.github.io/retrofit/>

Kapitola 5

Závěr

Hlavním cílem této bakalářské práce byl vývoj software pro řízení sítě hybridních částicových pixelových detektorů (označované jako ATLAS TPX), operujících uvnitř ATLAS experimentu na LHC v CERN - viz kapitola 3. V rámci této práce byl navržen a implementován software, který pomocí multi-vláknového přístupu umožňuje současně ovládat všechny detektory této sítě, zejména pak řízení akvizice dat, vyčítání pořízených snímků, nastavování měřících parametrů a také vyčítání a uchovávání stavových informací detektorů. Řídící software rovněž poskytuje JSON REST API server pro vzdálené ovládání, díky kterému bude mimo jiné možné řízení detektorové sítě implementovat do CERNského systému pro řízení a kontrolu detektorů (tzv. DCS). Další funkcionalitou tohoto software je zpracování a ukládání pořízených dat z detektorů, ty mohou být ukládány v Multiframe formátu do předem definovaného úložiště nebo odesílány datovému serveru pomocí jeho API, který data dále zpracuje a uloží do CERNského systému pro skladování dat (EOS).

Dalším cílem této práce byl vývoj nástrojů pro energetickou kalibrací sítě ATLAS TPX, pomocí metody popsané v [6]. Byl vyvinut univerzální software 3.3, který uživateli umožňuje průchod procesem zpracování dat pro energetickou kalibraci detektoru typu Timepix, pracujícího v Time-Over-Treshold módu. V rámci kalibračního procesu software ze spekter jednotlivých pixelů vytvoří jednotlivé kalibrační body pro různé zdroje záření. Z těchto bodů pak sestaví kalibrační funkci (viz vzorec 3.1), udávající vztah mezi energií a TOT, pro každý pixel detektoru. Software rovněž nabízí nástroje pro ověření kvality vstupních dat, kvality detektoru, ověření kvality kalibrace a také pro různé dodatečné úpravy výstupu kalibračního procesu, popsané v 3.3.4.

V této práci byly rovněž shrnuty a popsány vlastnosti hybridních částicových pixelových detektorů rodiny Medipix - viz kapitola 2.

Literatura

[1] BALLABRIGA, R. et al. Review of hybrid pixel detector readout ASICs for spectroscopic X-ray imaging. *Journal of Instrumentation*. 2016, 11, 01, s. P01007. Dostupné z: <http://stacks.iop.org/1748-0221/11/i=01/a=P01007>.

[2] BOUCHAMI, J. et al. Estimate of the neutron fields in ATLAS based on ATLAS-MPX detectors data. *Journal of Instrumentation*. 2011, 6, 01, s. C01042. Dostupné z: <http://stacks.iop.org/1748-0221/6/i=01/a=C01042>.

[3] GAMMA, E. et al. *Design Patterns: Elements of Reusable Object-oriented Software*. Boston, MA, USA : Addison-Wesley Longman Publishing Co., Inc., 1995. ISBN 0-201-63361-2.

[4] HOLY, T. et al. Pattern recognition of tracks induced by individual quanta of ionizing radiation in Medipix2 silicon detector. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*. 2008, 591, 1, s. 287 – 290. ISSN 0168-9002. doi: http://dx.doi.org/10.1016/j.nima.2008.03.074. Dostupné z: <http://www.sciencedirect.com/science/article/pii/S0168900208004592>. Radiation Imaging Detectors 2007Proceedings of the 9th International Workshop on Radiation Imaging Detectors.

[5] JAKUBEK, J. Energy-sensitive X-ray radiography and charge sharing effect in pixelated detector. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*. 2009, 607, 1, s. 192 – 195. ISSN 0168-9002. doi: http://dx.doi.org/10.1016/j.nima.2009.03.148. Dostupné z: <http://www.sciencedirect.com/science/article/pii/S0168900209006408>. Radiation Imaging Detectors 2008Proceedings of the 10th International Workshop on Radiation Imaging Detectors.

[6] JAKUBEK, J. Precise energy calibration of pixel detector working in time-over-threshold mode. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*. 2011, 633, Supplement 1, s. S262 – S266. ISSN 0168-9002. doi: http://dx.doi.org/10.1016/j.nima.2010.06.183. Dostupné z: <http://www.sciencedirect.com/science/article/pii/S0168900210013732>. 11th International Workshop on Radiation Imaging Detectors (IWORID).

[7] Kolektiv autorů Dropwizard. *Dropwizard online dokumentace - Getting Started* [on-

line]. 2016. [cit. 16. 5. 2016]. Dostupné z: <http://www.dropwizard.io/0.9.2/docs/getting-started.html>.

[8] Kolektiv autorů Medipix kolaborace. *Web Medipix* [online]. 2016. [cit. 20. 5. 2016]. Dostupné z: <https://medipix.web.cern.ch/medipix/pages/medipix1.php>.

[9] KRAUS, V. et al. FITPix — fast interface for Timepix pixel detectors. *Journal of Instrumentation*. 2011, 6, 01, s. C01079. Dostupné z: <http://stacks.iop.org/1748-0221/6/i=01/a=C01079>.

[10] LLOPART, X. et al. Medipix2: A 64-k pixel readout chip with 55- mu;m square elements working in single photon counting mode. *IEEE Transactions on Nuclear Science*. Oct 2002, 49, 5, s. 2279–2283. ISSN 0018-9499. doi: 10.1109/TNS.2002.803788.

[11] LLOPART, X. et al. Timepix, a 65k programmable pixel readout chip for arrival time, energy and/or photon counting measurements. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*. 2007, 581, 1-2, s. 485 – 494. ISSN 0168-9002. doi: http://dx.doi.org/10.1016/j.nima.2007.08.079. Dostupné z: <http://www.sciencedirect.com/science/article/pii/S0168900207017020>. {VCI} 2007Proceedings of the 11th International Vienna Conference on Instrumentation.

[12] PLATKEVIč, M. *Signal Processing and Data Read-Out from Position Sensitive Pixel Detectors*. PhD thesis, Czech Technical University in Prague, Czech Republic, 2014.

[13] POIKELA, T. et al. Timepix3: a 65K channel hybrid pixel readout chip with simultaneous ToA/ToT and sparse readout. *Journal of Instrumentation*. 2014, 9, 05, s. C05013. Dostupné z: <http://stacks.iop.org/1748-0221/9/i=05/a=C05013>.

[14] S. Ballestrero, A. Bogdanchikov , F. Brasolin, A. C. Contescu, S. Dubrov, M. Hafeez, A. Korol , C. J.Lee, D. A. Scannicchio, M. Twomey, A. Voronkov, A. Zaytsev. *ATLAS TDAQ application gateway upgrade during LS1* [online]. 2014. [cit. 7. 5. 2016]. Dostupné z: <https://cds.cern.ch/record/1664275/files/ATL-DAQ-SLIDE-2014-054.pdf>.

[15] TURECEK, D. et al. Pixelman: a multi-platform data acquisition and processing software package for Medipix2, Timepix and Medipix3 detectors. *Journal of Instrumentation*. 2011, 6, 01, s. C01046. Dostupné z: <http://stacks.iop.org/1748-0221/6/i=01/a=C01046>.

[16] TURECEK, D. et al. Remote control of ATLAS-MPX Network and Data Visualization. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*. 2011, 633, Supplement 1, s. S45 – S47. ISSN 0168-9002. doi: http://dx.doi.org/10.1016/j.nima.2010.06.117. Dostupné z: <http://www.sciencedirect.com/science/article/pii/S0168900210013070>. 11th International Workshop on Radiation Imaging Detectors (IWORID).

[17] TURECEK, D. Software for Radiation Detectors Medipix. Master's thesis, Czech Technical University in Prague, Czech Republic, 2011.

[18] Vladimír Wagner. *Článek: Jak se daří urychlovači LHC* [online]. 2009. [cit. 4. 5. 2016]. Dostupné z: <http://hp.ujf.cas.cz/~wagner/popclan/lhc/lhc_rok2010.htm>.

[19] VYKYDAL, Z. et al. The Medipix2-based network for measurement of spectral characteristics and composition of radiation in {ATLAS} detector. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*. 2009, 607, 1, s. 35 – 37. ISSN 0168-9002. doi: http://dx.doi.org/10.1016/j.nima.2009.03.104. Dostupné z: <http://www.sciencedirect.com/science/article/pii/S0168900209005956>. Radiation Imaging Detectors 2008Proceedings of the 10th International Workshop on Radiation Imaging Detectors.

Příloha A

Seznam použitých zkratek

ADC Analogově digitální převodník

Al Aluminium

API Application Programming Interface

ASIC Application-specific Integrated Circuit

ATLAS A Toroidal LHC Apparatus

B byte

b bite

BPMN Business Process Model and Notation

CdTe Cadmium telluride

CERN Evropská organizace pro jaderný výzkum (Originální název: *Conseil Européen pour la Recherche Nucléaire*), se sídlem v Ženevě, ve Švýcarsku.

CMOS Complementary Metal Oxide Semiconductor

CSM Charge Summing Mode

DAC Digitálně analogový převodník

DCS Detector Control Systems

DPS Deska plošného spoje

eV elektronvolt

FITPix Fast Interface for Timepix Pixel Detectors

FPGA Field Programmable Gate Array

FSM Finite State Machine

FWHM Full width at half maximum

GaAs Arsenid gallitý

HTTP Hypertext Transfer Protocol

HTTPS Hypertext Transfer Protocol Secure

HW Hardware

IP Internet Protocol

JSON JavaScript Object Notation

LED Light Emitting Diode

LHC Large Hadron Collider

LiF Lithium fluoride

LS Long Shutdown - dlouhodobá technologická přestávka LHC

LVDS Low-voltage differential signaling

MPX Medipix

PC Personal Computer

PCC Photon Counting Chip

PE Polyethylen

PN Přechod polovodiče typu P a polovodiče typu N

REST Representational State Transfer

RS232 Standart sériové linky

SMD Surface Mount Technology

SPI Serial Peripheral Interface

SPM Single Pixel Mode

SQL Structured Query Language

SSH Secure Shell

SW Software

TCP Transmission Control Protocol

TDAQ Trigger and Data Aquisition

TOA Time of Arrival - mód detektoru (viz 2.3)

TOT Time Over Treshold - mód detektoru (viz 2.3)

TPX Timepix

URL Uniform Resource Locator

USA15 Serverová místnost ATLAS experimentu

USB Universal Serial Bus

UX15 Označení prostor s ATLAS detektorem (tzv. cavern)

ÚTEF Ústav technické a experimentální fyziky

Příloha B

Konfigurační soubor ATLAS TPX serveru

```
################################################################################
# Basic configuration
################################################################################
# Path to config csv file with detecotrs config
# File structure: 1 detector struct per line
#  - detector_name[String];detector_ip[String];unit_id[int];tpx_id[int]
#     ;port[int]
detectorsConfigPath: data/detectors2.csv

# Controll REST API server configuration
server:
  applicationConnectors:
    - type: http
      port: 9179
      outputBufferSize: 32KiB
      idleTimeout: 30 seconds
      minBufferPoolSize: 64 bytes
      bufferPoolIncrement: 1KiB
      maxBufferPoolSize: 64KiB
      acceptorThreads: 1
      selectorThreads: 2
      acceptQueueSize: 1024
      reuseAddress: true
      soLingerTime: 600s
  adminConnectors:
    - type: http
      port: 9180

# Logging settings
logging:
```

```
  level: INFO
  appenders:
    - type: file
      currentLogFilename: log/atlastpx_server_app.log
      threshold: ALL
      archive: true
      archivedLogFilenamePattern: log/atlastpx_server_app-%d.log.gz
      archivedFileCount: 5
      timeZone: UTC
logging:
  level: INFO
  appenders:
    - type: console
      threshold: ALL
      timeZone: UTC
      target: stdout


################################################################################
# Data readout & saving configuration
################################################################################
# If true asynchronous event handler will takes care abouth automatical data
# (frame + DSC) readout from the detector.
readOutDataAutomatically: true

# Directory for saving output data
outputDir: ../frames

# Local data saving configuration
# Alowed types:
# - MULTIFRAME: Pixelman's multiframe format (3 files per hour).
#               Output files:
#                 1) Frames: Line [X, C], where X is pix. index and
#                         C pix value. Frames are separated by '#'
#                 2) Description file ( DSC)
#                 3) Index file (indexing file 1 and 2)
# Example - outputFramesType: [MULTIFRAME]
outputFramesType: [MULTIFRAME]


################################################################################
# Data server configuration
################################################################################
# If true all data (configs on change, performed frames) will be automatically
# send to the data server
# and also data server status will be awailable by info endpoint.
```

```
77 useDataServer: true
78
79 # URL of dataServer
80 dataServerBaseUrl: atlastpx.utef.cvut.cz
81
82 # port of dataServer
83 dataServerPort: 8042
```

Zdrojový kód B.1: Konfigurační soubor ATLAS TPX serveru

Příloha C

Obsah přiloženého CD

```
CD/
├─ atlas_tpx/
│  ├─ dokumentace/ - adresář obsahující popis API
│  ├─ exe/
│  │  ├─ emulator/
│  │  │  ├─ readme.txt - README soubor
│  │  │  └─ AtlasPixEmulator-0.1.jar - spustitelný jar soubor emulátoru
│  │  └─ atlas_tpx_server/
│  │     ├─ readme.txt - README soubor
│  │     ├─ AtlasTPX.server-0.1.jar - spustitelný jar soubor serveru
│  │     ├─ server-configuration.yml - soubor s konfigurací serveru
│  │     └─ detectors.csv - tabulka s detektory
│  └─ src/ - adresář se zdrojovými kódy
├─ kalibrace/
│  ├─ exe/ - spustitelné binární soubory
│  │  ├─ libs/ - jar knihovny
│  │  ├─ test_data/ - vstupní data kalibrace (spektra)
│  │  ├─ readme.txt - README soubor
│  │  └─ X-rayTimepixCalibration.jar - spustitelný jar soubor
│  └─ src/ - adresář se zdrojovými kódy
└─ text/
   ├─ LaTeX/ - adresář se zdrojovými soubory tohoto dokumentu
   ├─ thesis-begerjak-2016.pdf - tato práce ve formátu PDF
   ├─ abstract_cz.txt - abstrakt česky
   └─ abstract_en.txt - abstrakt anglicky
```

Obrázek C.1: Obsah přiloženého CD