Praca wykonana w ramach projektu TEAM kierowanego przez Prof. Pawła Moskala. Projekt finansowany przez Fundację na rzecz Nauki Polskiej

AKADEMIA GÓRNICZO-HUTNICZA IM. STANISŁAWA STASZICA W KRAKOWIE

WYDZIAŁ ELEKTROTECHNIKI, AUTOMATYKI, INFORMATYKI I INŻYNIERII BIOMEDYCZNEJ

KATEDRA AUTOMATYKI I ROBOTYKI

Praca dyplomowa magisterska

System akwizycji danych dla modularnego skanera PET oparty na układach FPGA

Data acquisition system for modular PET scanner based on FPGA devices

Autor: *Maciej Bakalarek*
Kierunek studiów: *Automatyka i Robotyka*
Opiekun pracy: *Dr Grzegorz Korcyl*

Kraków, 2020

Uprzedzony o odpowiedzialności karnej na podstawie art. 115 ust. 1 i 2 ustawy z dnia 4 lutego 1994 r. o prawie autorskim i prawach pokrewnych (t.j. Dz.U. z 2006 r. Nr 90, poz. 631 z późn. zm.): „Kto przywłaszcza sobie autorstwo albo wprowadza w błąd co do autorstwa całości lub części cudzego utworu albo artystycznego wykonania, podlega grzywnie, karze ograniczenia wolności albo pozbawienia wolności do lat 3. Tej samej karze podlega, kto rozpowszechnia bez podania nazwiska lub pseudonimu twórcy cudzy utwór w wersji oryginalnej albo w postaci opracowania, artystycznego wykonania albo publicznie zniekształca taki utwór, artystyczne wykonanie, fonogram, wideogram lub nadanie.", a także uprzedzony o odpowiedzialności dyscyplinarnej na podstawie art. 211 ust. 1 ustawy z dnia 27 lipca 2005 r. Prawo o szkolnictwie wyższym (t.j. Dz. U. z 2012 r. poz. 572, z późn. zm.): „Za naruszenie przepisów obowiązujących w uczelni oraz za czyny uchybiające godności studenta student ponosi odpowiedzialność dyscyplinarną przed komisją dyscyplinarną albo przed sądem koleżeńskim samorządu studenckiego, zwanym dalej «sądem koleżeńskim».", oświadczam, że niniejszą pracę dyplomową wykonałem(-am) osobiście i samodzielnie i że nie korzystałem(-am) ze źródeł innych niż wymienione w pracy.

Składam serdeczne podziękowania dla dr Grzegorza Korcyla, który zawsze gotów był służyć pomocą i dobrą radą. Składam również podziękowania dla prof. dr hab. Pawła Moskala za daną mi szansę uczestniczenia w projekcie J-PET oraz Rodzicom za wsparcie duchowe i polonistyczne.

Spis treści

1. Akronimy

TK – tomografia komputerowa (ang. computed tomography - CT)

RM – rezonans magnetyczny (ang. magnetic resonance imaging - MRI)

PET – pozytonowa tomografia emisyjna (ang. positron emission tomography)

TOF – czas przelotu (ang. time of flight)

LOR – linia odpowiedzi (ang. line of response)

ROR – region odpowiedzi (ang. region of response)

FTAB – płyta z elekroniką czołową

ASIC – wyspecjalizowany układ scalony (ang. aplication specific integrated circuit)

DSP – procesor sygnałowy (ang. digital signal processor)

FPGA – bezprośrednio programowalna macierz bramek logicznych (ang. field programmable gate array)

DAQ – system akwizycji danych (ang. data acquisition system)

PHA – analizator sygnałów z fotopowielaczy (ang. pulse height analizer)

J-PET – Jagiellonian Positron Emission Tomograph

CM-PET – Cyfrowo Modularny PET

TDC – układ dyskryminujący oraz przetwarzający informacje o czasie na cyfrową (ang. time-to-digital converter).

2. Wprowadzenie

Gwałtowny rozwój techniki w drugiej połowie XX wieku umożliwił powstanie nowych metod obrazowania medycznego takich jak TK (**T**omografia **K**omputerowa), PET (**P**ozytonowa **T**omografia **E**misyjna) czy RM (**R**ezonans **M**agnetyczny). Metody te w nieinwazyjny sposób zapewniają wgląd we wnętrze żywego organizmu, czy to na poziomie anatomicznym (TK, RM), czy metabolicznym (PET).

Tomografia PET będąca przedmiotem tej pracy, powstawała na przełomie lat 60 i 70 ubiegłego wieku. Wtedy to w USA powstały pierwsze cylindryczne urządzenia PET, które swym działaniem przypominały współczesne tomografy. Ich użycie, poprzez badanie miejscowego zużycia glukozy oraz chwilowego przepływu krwi dało początek nauce znanej dziś jako mapowanie mózgu [1]. Dziś, w zależności od użytych radiofarmaceutyków obrazowanie metodą PET wykorzystywane jest w onkologii, neurologii, czy obrazowaniu układu mięśniowo-szkieletowego.

Pierwsze tomografy składały się z kilkunastu detektorów o długości nieprzekraczającej 10 cm [1]. Współcześnie buduje się kilkuwarstwowe skanery liczące nawet kilkaset detektorów. Trwają również badania nad tomografami *total-body* PET (np. Explorer [2], czy Total-Body TOF J-PET), które to miałyby mieć możliwość obrazowania całego pacjenta jednocześnie. Naturalnie, zwiększając powierzchnie detektorów należy się spodziewać proporcjonalnego wzrostu ilości generowanych danych oraz potrzebnej mocy obliczeniowej do ich przetworzenia [3].

2.1. Cele pracy

Celem poniższej pracy jest przedstawienie projektu systemu akwizycji danych oraz ich wstępnego przetwarzania w skanerze PET zbudowanym w ramach projektu J-PET. Na opisywany w pracy system składają się mechanizmy selekcji i wstępnej obróbki danych tomograficznych oraz wizualizacji rezultatów na żywo. Implementacji systemu dokonano w układach reprogramowalnych FPGA oraz heterogenicznej platformie MPSoC firmy Xilinx.

2.2. Zawartość pracy

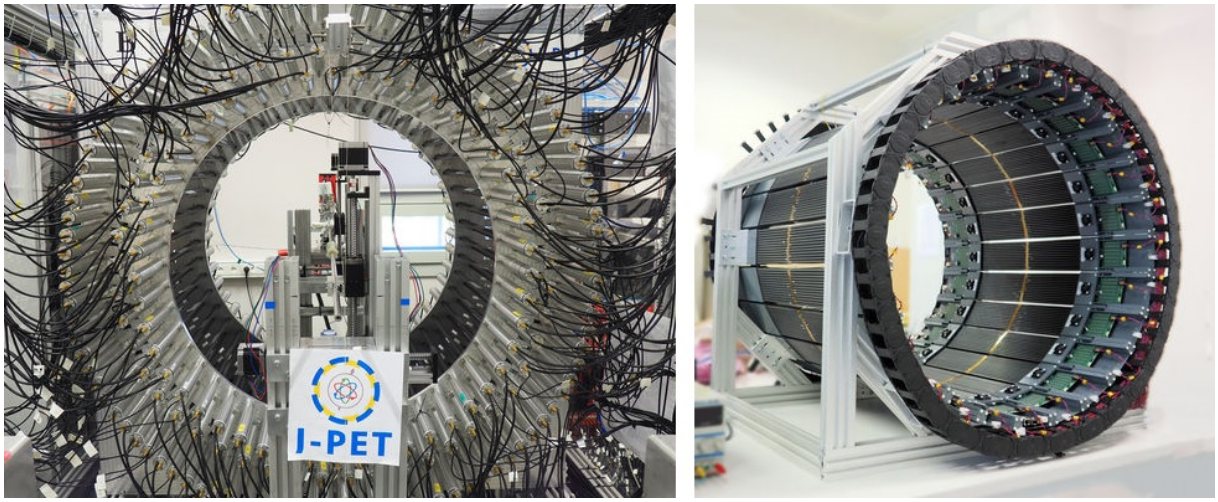
Pierwsze rozdziały pracy poświęcono na wstęp teoretyczny do pozytonowej tomografii emisyjnej oraz do projektu J-PET. W rozdziale 1 umieszczono rozwinięcie najczęściej używanych w pracy akronimów. Rozdział 2 został przeznaczony na wprowadzenie do pracy oraz opis projektu J-PET. W kolejnym

rozdziale przedstawiono fizyczne podstawy działania pozytonowej tomografii emisyjnej. Porównano tam także budowę typowego tomografu PET z skanernem J-PET oraz CM-PET. Na przykładzie algorytmu *filtered back-projection* przedstawiono podstawową ideę rekonstrukcji obrazu tomograficznego. Rozdział 4 poświęcono układom FPGA, ponieważ elektronika cyfrowa skanera CM-PET jest niemalże w całości o nie oparta. W odniesieniu do procesorów i układów wyspecjalizowanych przedstawiono podstawowe cechy układów rekonfigurowalnych. Opisano tam także budowę oraz sposób ich programowania. W rozdziale 5 szczegółowo przedstawiono architekturę skanera CM-PET. Osobne podrozdziały poświęcono na opisanie układów kontrolera, koncentratorów oraz płyt czołowych (w szczególności układów TDC). Rozdział 6 poświęcono zaimplementowanej w ramach pracy architekturze systemu. Na zaprezentowane w tam urządzenie wstępnego przetwarzania danych składają się mechanizmy selekcji zdarzeń z surowych danych tomograficznych, parowania ich w koincydencję, estymacji miejsca uderzenia kwantu gamma w powierzchnię detektora oraz łączenia koincydencji w linie odpowiedzi. Przedstawiona architektura systemu cechuje się dużym zrównolegleniem obliczeń. Dodatkowo w tym rozdziale opisano próby przygotowania wizualizacji rezultatów przetwarzania na żywo. Opracowano dwie metody ilustracji wyników analizy - poprzez serwis sieciowy oraz przy użyciu układu graficznego znajdującego się w platformie MPSoC. W rozdziale 7 przedstawiono przykładowy wynik pomiarów. W załączniku A zamieszczono schemat zaimplementowanego systemu. Ostatni załącznik poświęcono na uzupełnienia.

2.3. Projekt J-PET

Celem projektu J-PET jest skonstruowanie nowoczesnego skanera TOF-PET (ang. **T**ime **o**f **F**light **P**ositron **E**mission **T**omograph) wykorzystującego organiczne, plastikowe scyncylatory (rys. 2.1). Projekt ten jest realizowany na Uniwersytecie Jagiellońskim w Krakowie od około 2012 roku pod przewodnictwem Prof. Pawła Moskala. Tym co wyróżna J-PET jest nowatorskie zastosowanie organicznych scyncylatorów oraz wyjatkowy układ detektorów. Typowe, komercyjne tomografy (GE Healthcare, Philips i Siemens) używają nieorganicznych kryształów jako elementów czułych na promieniowanie jonizujące (odpowiednio BGO, LSO oraz LYSO) [4]. Na korzyść organicznych polimerowych scyncylatorów przemawia niższa cena oraz łatwość ich produkcji. Dodatkowo, nietypowy sposób umieszczenia fotopowielaczy oraz wykorzystanie dyskryminacji czasu jako podstawowej medoty konwersji A/D umożliwia zaprojektowanie dłuższego tomografu bez drastycznego zwiększenia kosztów jego produkcji [5].

Równolegle do tomografu J-PET od roku 2016 powstaje skaner o nazwie Cyfrowo Modularny PET. W skanerze CM-PET również wykorzystywane są plastikowe paski scyncylacyjne. Tym co odróżnia oba tomografy jest elektronika czołowa. Podczas gdy skaner J-PET oparty jest na tradycyjnych fotopowielaczach, w tomografie CM-PET wykorzystuje się krzemowe fotopowielacze oraz układy FPGA wykonujące konwersję TDC. Użycie krzemowych fotopowielaczy pozwoliło na zaprojektowanie kompaktowego oraz modularnego tomografu. Opisywany w niniejszej pracy system został zaimplementowany w prototypie skanera CM-PET.

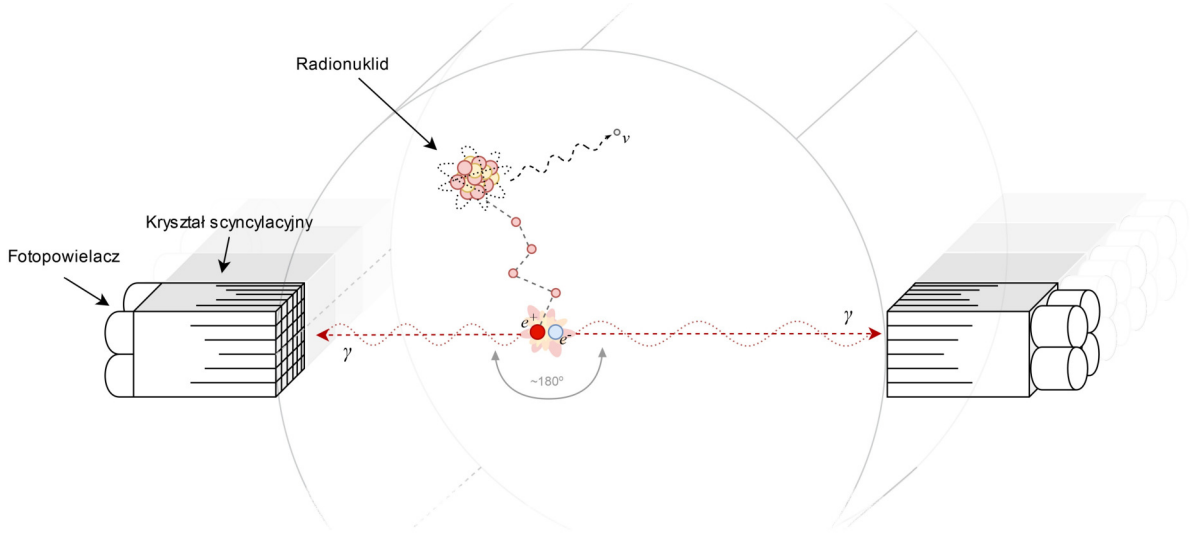


Rys. 2.1. Po lewej: trzywarstwowy skaner J-PET, po prawej: prototyp modularnego skanera CM-PET

3. Pozytonowo Emisyjna Tomografia

W tym rozdziale pokrótce przedstawiono fizyczne podstawy działania skanera PET. Zaprezentowano również podstawową ideę rekonstrukcji obrazu tomograficznego na przykładzie algorytmu analitycznego *filtered back-projection*.

3.1. Podstawy fizyczne PET



Rys. 3.1. Ilustracja anihilacji pozytonu oraz elektronu. W wyniku rozpadu β^+ radionuklid emituje pozyton (e^+) oraz neutrino (ν). Pozyton anihiluje z napotkanym elektronem. W rezultacie wyemitowane kwanty gamma (γ) wpadają w detektory.

Celem pozytonowej tomografii emisyjnej jest estymacja przestrzennego rozkładu wybranej substancji w ciele pacjenta. W tym celu pacjentowi podaje się farmaceutyk z domieszką izotopów promieniotwórczych. Najpopularniejszym radiofarmaceutykiem jest fludeoksyglukoza (^{18}FDG). W wyniku promieniotwórczego rozpadu izotopów powstają pozytony (e^+). Te, po przemieszczeniu się ok. 1 do 2 mm w ciele napotykają elektrony (e^-) i z nimi anihilują. W wyniku anihilacji ich masa zmienia się w energię. Najczęściej energia ta przyjmuje postać pary kwantów gamma (2γ) o energii 511 keV każdy i poruszających się w przeciwnych kierunkach (rys. 3.1) [6]. Kiedy kwant gamma wpada w detektor, może

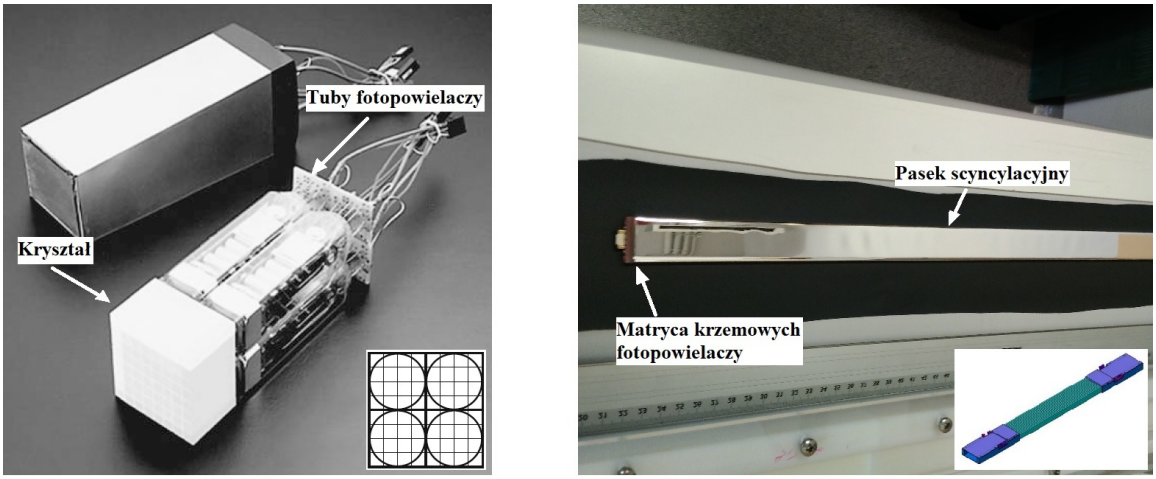
on oddać część lub całość swojej energii elektronom znajdujących się w absorberze. Przekazanie energii w zależności od materiału dektora (w szczególności jego liczby atomowej) oraz od energii samego kwantu odbywa się poprzez zjawisko fotoelektryczne, efekt Comptona lub kreakcję par [7]. Wybite z atomów elektrony na wskutek jonizacji wywołują w scyncylatorze błyski światła, które to są wzmacniane oraz przekształcane na sygnały napięciowe lub prądowe za pomocą fotopowielaczy [8]. Sygnały z fotopowielaczy zbierane są przez dedykowaną elektronikę czołową (często zwaną z ang. **P**ulse **H**eight **A**nalizer). Działanie PHA opiera się na dyskryminacji i digitalizacji tych sygnałów, które mieszczą się w oknie energetycznym [6] . W rezultacie otrzymuje się zbiór przybliżonych czasów w których to kwanty gamma docierały do detektorów (dalej zwanych zdarzeniami). Zebrane w ten sposób zdarzenia są parowane w tzw. linie odpowiedzi (ang. **L**ine **o**f **R**esponse - LOR). Dodatkowo, w skanerach TOF-PET możliwe jest również określenie tzw. regionów odpowiedzi (ang. **R**egion **o**f **R**esponse - ROR) na podstawie różnicy czasów dotarcia fotonów gamma do detektorów. Zbiór LOR (ROR) jest podstawowym elementem, na podstawie którego możliwa jest rekonstrukcja obrazu tomograficznego.

3.2. Skaner J-PET, a tradycyjne tomografy

W tradycyjnych tomografach jako detektorów używa się niewielkich, kilkucentymetrowych kryształów scyncylacyjnych ponacinanych w kwadratową siatkę (np. 5x5 mm). Do największej ściany kryształu przyłożone są zazwyczaj cztery fotopowielacze [6]. W takim układzie pojedynczy fotopowielacz obejmuje kilkanaście pól siatki. Przykładowy układ detektora tomografu PET opartego na kryształach przedstawiono na rys. 3.2A. Na podstawie amplitudy lub wielkości zdeponowanego ładunku na poszczególnych fotopowielaczach możliwe jest określenie „oczka" siatki w które wpadł foton gamma. Rozwiązanie to, choć proste, ogranicza rozdzielczość przestrzenną tomografu do rozmiarów pola siatki. Dodatkowo zwiększając długość tomografu należy liczyć się z proporcjonalnym zwiększeniem zapotrzebowania na elektronikę czołową [6].

W przeciwieństwie do tradycyjnych tomografów w skanerze J-PET wykorzystano scyncylatory plastikowe (rys. 3.2B). Choć są one znacznie tańsze oraz prostsze w obróbce od nieorganicznych kryształów, to jak dotąd nie były rozważane w tomografii. Powodem tego jest ich znikoma czułość na promieniowanie jonizujące oraz małe prawdopodobieństwo na zajście efektu fotoelektrycznego. Jednak poprzez zwiększenie rozdzielczości czasowej i zmianę sposobu ułożenia detektorów (rys. 3.2B) możliwa jest kompensacja tych wad [5]. W tomografie J-PET wykorzystano scyncylatory w formie długich na około 30 cm plastikowych pasków, a do najbardziej odległych boków detektorów zamocowano fotopowielacze. Dzięki temu finalna możliwa do uzyskania rozdzielczość skanera wynosi ok. 80 ps [10] [11].

W opracowanym prototypie skanera CM-PET tuby fotopowielaczy zastąpiono krzemowymi diodami lawinowymi z efektem Geigera. Dodatkowo układy dokonujące digitalizacji oraz dyskryminacji zaimplementowano w układach FPGA. W tym celu wykorzystano bufory wejściowe syngnałów różnicowych



Rys. 3.2. Porównanie detektorów. Obraz (A) przedstawia tradycyjny detektor zbudowany z nieogranicznego kryształu oraz tub fotopowielaczy (w prawym dolnym rogu zamieszczono schemat przymocowania tub do kryształu), źródło [6]. Na obrazie (B) zaprezentowano jeden z pasków używanych w CM-PET, źródło [9].

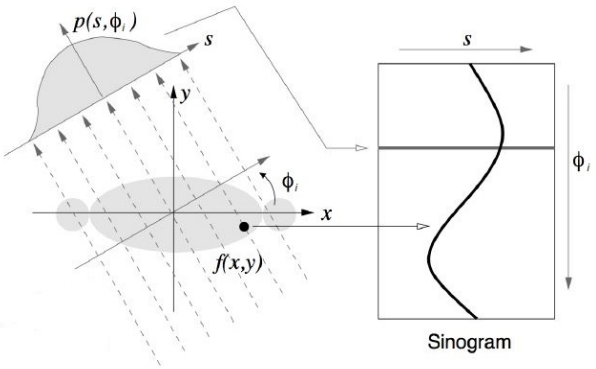
jako komparatory oraz logikę *carry-chain* jako elementy opóźniające sygnał. Dokładniejszy opis działania TDC (ang. **T**ime-to-**D**igital **C**onverter) zaimplementowanego w układzie rekonfigurowalnym znajduje się w rozdziale 5. Wykorzystanie krzemowych fotopowielaczy oraz układów FPGA rodzi nadzieje na jeszcze lepszą rozdzielczonść skanera oraz niższe koszty wykonania tomografu [12].

3.3. Rekonstrukcja obrazu

W poniższej sekcji przedstawiono ideę rekonstrukcji obrazu tomograficznego na podstawie algorytmu *filtered back-projection*. Jest to jeden z pierwszych oraz prostszych algorytmów analitycznych używanych w tomografii. Dla uproszczenia przyjęto dwuwymiarowy przypadek.

Mając zbiór zdarzeń uzyskanych podczas skanu tomograficznego możliwe jest ich połączenie w LOR. W tym celu paruje się wszystkie zdarzenia które wystąpiły w danym oknie czasowym, a miejsca ich rejestracji mieszczą się w osiowym polu widzenia skanera (ang. axial field-of-view). By uzyskać obraz tomograficzny zbiór LOR przekszałca się w sinogram, dokonuje się jego filtracji oraz tzw. projekcji wstecznej. Formalizując, rozkład radiofarmaceutyku w ciele pacjenta można oznaczyć poprzez $f(x, y)$. Suma wszystkich LOR leżących pod pewnym kątem ϕ_i jest projekcją dystrybucji f na płaszczynieźnie i-tego detektora [13]. Oznacza się ją jako $p(s, \phi_i)$. Funkcja $p(s, \phi)$ (dla $\phi \in (0, \pi)$) tworzy tzw. sinogram. Idea ta została przestawiona na rys. 3.3.

Celem rekonstrukcji jest estymacja oryginalnej dystrybucji f na podstawie projekcji p. Niestety ze względu na naturę procesu detekcji zebrane dane są stochastyczne, przez co problem znalezienia transformaty $p \to f$ jest źle zdefiniowany [13]. W celu polepszenia jakości obrazu oraz uniknięcia artefaktów



Rys. 3.3. Proces powstawania sinogramu. Suma LOR leżących pod kątem ϕ_i, będąca rzutem dystrybucji f na płaszczyznę detektora tworzy sinogram $p(s, \phi_i)$, źródło: [13].

należy wykorzystać metody regularyzacji takie jak np. całkowitej wariacji (ang. total variation method) [14].

Na podstawowy algorytm FBP składają się dwa główne kroki - filtracja sinogramu w dziedzinie częstotliwości:

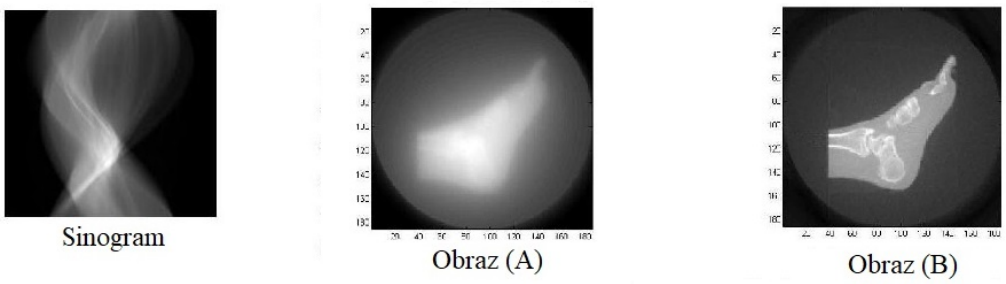
$$p^F(s,\phi) = \mathcal{F}^{-1}(W(v_s)|v_s|\mathcal{F}(p(s,\phi))) \tag{3.1}$$

oraz projekcja wsteczna (ang. back-projection):

$$\tilde{f}(x,y) = \int_0^{\pi} p^F(s,\phi)d\phi \tag{3.2}$$

gdzie $s = xcos\phi + ysin\phi$, a $\tilde{f}$ jest estymatą f

W pierwszym kroku (wz. 3.1) na otrzymanym sinogramie dokonuje się filtracji górnoprzepustowej - najczęściej filtrem rampowym $|v_s|$. Często wyrażenie pod odwrotną transformatą Fouriera jest mnożone przez okno np. Hamminga lub Hanna oznaczone w wzorze 3.1 jako $W(v_s)$. Operacja ta ma na celu uniknięcie wzmocnienia wysokich częstotliwości przez filtr rampowy [15]. W rezultacie otrzymuje się przefiltrowany (w dziedzinie częstotliwości) sinogram p^F, który to następnie można użyć do wyznaczenia estymaty funkcji f poprzez projekcję wsteczną (wz. 3.2).



Rys. 3.4. Przykładowy obraz uzyskany w wyniku przetwarzania algorytmem projekcji wstecznej (A) oraz *filtered back-projection* (B). Źródło: [16]

4. FPGA

W tym rozdziale przedstawiono podstawowe informacje o układach FPGA. Porównano je z układami wyspecjalizowanymi oraz z procesorami, opisano ich budowę oraz przedstawiono sposób w jaki się je programuje.

4.1. Układy reprogramowalne - wprowadzenie

Niezależnie od przeznaczenia urządzenia, dobór odpowiedniej platformy obliczeniowej pełni kluczową rolę w procesie jego projektowania. Podstawowym aspektem podczas wyboru sprzętu jest cena oraz czas który należy poświęcić na jego oprogramowanie (ang. time-to-market). Duże znaczenie mają także elastyczność oraz wydajność platformy. Współcześnie można wyróżnić cztery grupy urządzeń. Są to procesory (np. ogólnego przeznaczenia, sygnałowe itp); układy graficzne; układy wyspecjalizowane (ASIC, ang. **A**pplication **S**pecific **I**ntegrated **C**ircuit) oraz układy reprogramowalne takie jak CPLD, czy FPGA (ang. **C**omplex **P**rogrammable **L**ogic **D**evice, **F**ield **P**rogrammable **G**ate **A**rray). Procesory ogólnego przeznaczenia, układy DSP oraz karty graficzne z pewnością spełniają kryteria niskiego kosztu produkcji oraz elastyczności. Współcześnie wysoki poziom abstrakcji znacząco ułatwia programowanie tych układów, co skraca czas potrzebny na wykonanie urządzenia. Niestety, takie układy często nie są tak wydajne zarówno pod względem czasowym jak i energetycznym jak wyspecjalizowane układy scalone. Układy ASIC posiadają natomiast szereg wad, które sprawiają że ich produkcja uzasadniona jest tylko w konkretnych przypadkach. Należy tu wymienić długi czas projektowania układu, brak elastyczności oraz wrażliwość na błędy projektanta. Nie ma bowiem możliwości aktualizacji urządzenia po jego wyprodukowaniu. Wady te powodują, że układy wyspecjalizowane są opłacalne tylko podczas produkcji masowej (wielomilionowe sztuki) [17].

Urządzenia rekonfigurowalne łączą zalety procesorów oraz układów ASIC, nie dziedzicząc przy tym ich wad. Układy FPGA to półprzewodnikowe urządzenia zbudowane w oparciu o macierz prostych, rekonfigurowalnych elementów logicznych (ang. Configurable Logic Block - CLB) połączonych reprogramowalną siecią. Struktura ta sprawia, że zmiana fizycznych połączeń wewnątrz układu wymaga jedynie modyfikacji jego oprogramowania. Skutkiem tego układy FPGA są znacząco prostsze, tańsze i szybsze w prototypowaniu niż układy wyspecjalizowane. Dodatkowo cykl życia produktu opartego na układach reprogramowalnych nie kończy się wraz z opuszczeniem fabryki – istnieje bowiem możliwość zdalnego aktualizowania tych układów.

Uniwersalność układów FPGA sprawiła, że są one dziś coraz chętniej wykorzystywane w niemal każdej gałęzi przemysłu (telekomunikacja, medycyna, przemysł samochodowy, obliczenia o wysokiej wydajności, astronautyka) [17] oraz nauki (projekty takie jak PANDA [18][19], HADES [20] czy J-PET).

4.2. Budowa układów FPGA

Podstawowym elementem budującym układy FPGA są konfigurowalne bloki logiczne. W zależności od producenta oraz typu układu może ich być od 20 (np. Cypress PSoC5LP) do nawet 10 mln (układ Intel Stratix 10 GX10M). Bloki te najczęściej są zbudowane z kilku wejściowej tabeli LUT, realizującej funkcje logiczne, multipleksera oraz przerzutników typu D. Komponenty te umożliwiają realizację funkcji kombinatorycznych i sekwencyjnych. Dodatkowo często istnieje możliwość konfiguracji LUT w formie pamięci RAM lub rejestru przesuwnego. Takie rozwiązanie nosi nazwę rozproszonej pamięci (ang. distributed RAM). Bloki CLB połączone są za pomocą programowalnej sieci tworzącej przełączalne skrzyżowania (ang. interconnect). Dodatkowo, każdy blok logiczny jest połączony łańcuchem *carry-chain* z innymi blokami sąsiadującymi z nim w kolumnie. Połączenie to ma zapewnić mechanizm niezbędny do wykonywania szybkich operacji arytmetycznych.

Inne często spotykane bloki budujące macierz układów FPGA to bloki RAM (BRAM); moduły obsługi zegara – odpowiedzialne za prawidłowe rozprowadzenie sygnału zegara po układzie, redukcję zjawiska *jitter* i *drift*; bloki wejść i wyjść – np. transceivery; liczne układy DSP, które w najnowszych układach prócz zadań typowej mnożarki dostarczają wsparcie do zadań takich jak FFT [17].

4.3. Platformy SoC

Dziś coraz chętniej wykorzystuje się heterogeniczne platformy obliczeniowe, tzw. układy SoC (ang. **S**ystem **o**n **C**hip). Łączą one w jednym układzie scalonym różne elementy obliczeniowe. Najczęściej spotykanym, szczególnie w rozwiązaniach IoT (ang. **I**nternet-**o**f-**T**hings, tzw. internet rzeczy) jest zestawienie mikroprocesora oraz radia (np. WiFi lub Bluetooth). Innym, używanym gdy istnieje potrzeba akceleracji pewnych algorytmów jest połączenie procesorów i układów rekonfigurowalnych, takich jak np. FPGA.

MPSoC to heterogeniczny układ produkowany przez firmę Xilinx. W jednym układzie scalonym znajduje się układ FPGA, cztery procesory ogólnego przeznaczenia ARM Cortex A53, dwa procesory czasu rzeczywistego Cortex R5 oraz układ graficzny ARM MALI400. Komunikacja między procesorami, układem konfigurowalnym i układem graficznym może się odbywać na kilka różnych sposobów. Tym, który jest najczęściej wykorzystywany jest magistrala AMBA AXI4. Jest to rodzina magistral przeznaczonych do komunikacji wewnątrz układowej. Cechuje się ona dużą rekonfigurowalnością oraz uniwersalnością [17]. Dodatkowo istnieje możliwość konfiguracji mostu *AXI Chip2Chip* opartego na

protokole Aurora, który zapewnia enkapsulację magistrali AXI pomiędzy oddzielnymi układami scalonymi. Metoda ta został szerzej opisana w rozdziale 5. Duża uniwersalność oraz rekonfigurowalność sprawiły, że platforma MPSoC pełni kluczową rolę w skanerze CM-PET.

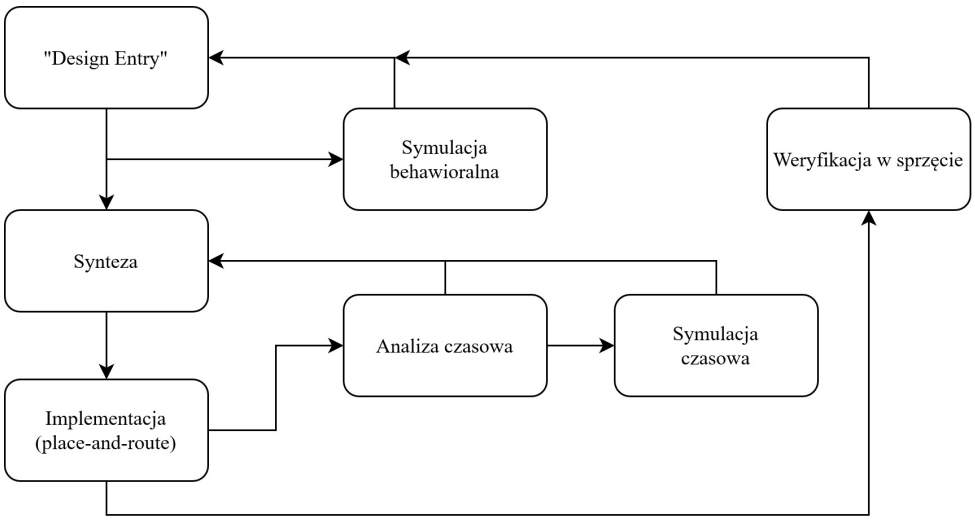
4.4. Metodologia pracy z FPGA

Programowanie układów FPGA znacząco różni się od programowania konwencjonalnych układów, takich jak procesory lub układy graficzne. Projektanci używają języków opisu sprzętu (ang. **H**ardware **D**escription **L**anguage) do zdefiniowania struktury (programowanie strukturalne) lub zachowania układu (programowanie behawioralne). Kod HDL następnie jest syntezowany za pomocą tzw. narzędzi EDA (ang. **E**letronic **D**esign **A**utomation), tak by użytkownik mógł później go wgrać do pamięci układu FPGA. W pierwszym kroku syntezy narzędzie EDA generuje strukturę zwaną *netlist*. Jest to lista wygenerowana na podstawie kodu, która definiuje połączenia poszczególnych elementów logicznych. W kolejnej fazie syntezy (*place-and-route*) struktura *netlist* jest mapowana na konkretne elementy logiczne znajdujące się w wybranym układzie. Następnie narzędzia EDA optymalizują rozmieszczenie elementów, tak by urządzenie spełniało ograniczenia czasowe (ang. timing constraints). Jeżeli wszystkie z wyżej wymienionych kroków zakończą się sukcesem, generowany jest plik konfiguracyjny, który użytkownik może wgrać do układu FPGA (np. za pomocą interfejsu J-TAG).

Najczęściej używanymi językami opisu sprzętu są VHDL (ang. **V**ery High Speed Integrated Circuit **HDL**) oraz Verilog. Na popularności zyskują także syntezatory wysokiego poziomu np. Vivado HLS, SDAccel, Vitis, czy Matlab HDL Coder. Rozwiązania te pozwalają na konwersję jezyków C, C++ lub Simulink do struktury opisującej sprzęt. Często również wspierają programowanie urządzeń SoC. Przykładem tego może być środowisko SDAccel, które daje programiście możliwość wybrania, jakie części kodu C/C++ mają być wykonywane na procesorze, a jakie będą akcelerowane w układzie FPGA.

Proces programowania układów FPGA można podzielić na kilka etapów (rys. 4.1). W pierwszym kroku programista powinien określić wymagania względem projektowanego systemu. Znając przeznaczenie urządzenia, ale także jego ograniczenia ze względu na cenę, wydajność itp. możliwe jest stworzenie zarysu architektury systemu. Upewniwszy się, że projekt ten nie zawiera krytycznych błędów, projektant może przystąpić do opisywania urządzenia w językach HDL. Zazwyczaj implementowany system dzieli się na mniejsze podsystemy - tzw. moduły. Takie podejście ułatwia weryfikację oraz umożliwia ponowne wykorzystanie kodu. Poszczególne moduły poddawane są testom (ang. unit-test). Narzędziem używanym podczas tych testów jest tzw. symulacja behawioralna, gdzie to po stymulacji wejść modułów obserwuje się stan ich wyjść, a następnie wyniki symulacji porównuje się z modelem wzorcowym. Mając pewność że poszczególne bloki budujące system działają poprawnie, można przystąpić do ich integracji. Oczywiście, projekt powstały w wyniku integracji także powinien zostać przetestowany w symulacji. Najczęściej używaną metodą weryfikacji układów FPGA (oraz ASIC) jest UVM (ang. **U**niversal **V**erification **M**ethodology). Jest to zbiór bibliotek ułatwiających stymulacje, obserwacje oraz weryfikacje symulowanego układu. W przypadku mniejszych projektów warto jest jednak napisać własne, mniej

skomplikowane środowisko weryfikacyjne (przykład takiego środowiska przedstawniono w rozdziale 6). Po upewnieniu się, że system zachowuje się tak jak powinien, projektant może rozpocząć jego syntezę. Poszczególne kroki syntezy zostały już opisane wyżej. Zaimplementowane urządzenie należy przeanalizować oraz sprawdzić, czy wszystkie ograniczenia czasowe zostały spełnione. Jeżeli nie - należy ręcznie nałożyć dodatkowe ograniczenia (o ile to możliwe) lub poprawić problematyczne moduły (np. poprzez dodanie dodatkowych rejestrów). Ostatnim krokiem jest weryfikacja działającego urządzenia. W zależności od poziomu skomplikowania układu może zaistnieć potrzeba zbudowania osobnego systemu do jego weryfikacji. Najczęściej jednak wystarcza użycie wbudowanego analizatora stanów logicznych (np. ChipScope, ILA).



Rys. 4.1. Tradycyjny schemat postępowania podczas projektowania układów FPGA. Na podstawie „*Xilinx ISE Design Flow*".

Przedstawiony powyżej schemat postępowania nie jest liniowy. Często zachodzi potrzeba powrotu do poprzednich kroków. Sprawia to, że projektowanie systemów opartych o urządzenia rekonfigurowalne jest znacznie bardziej czasochłonne niż tradycyjne programowanie.

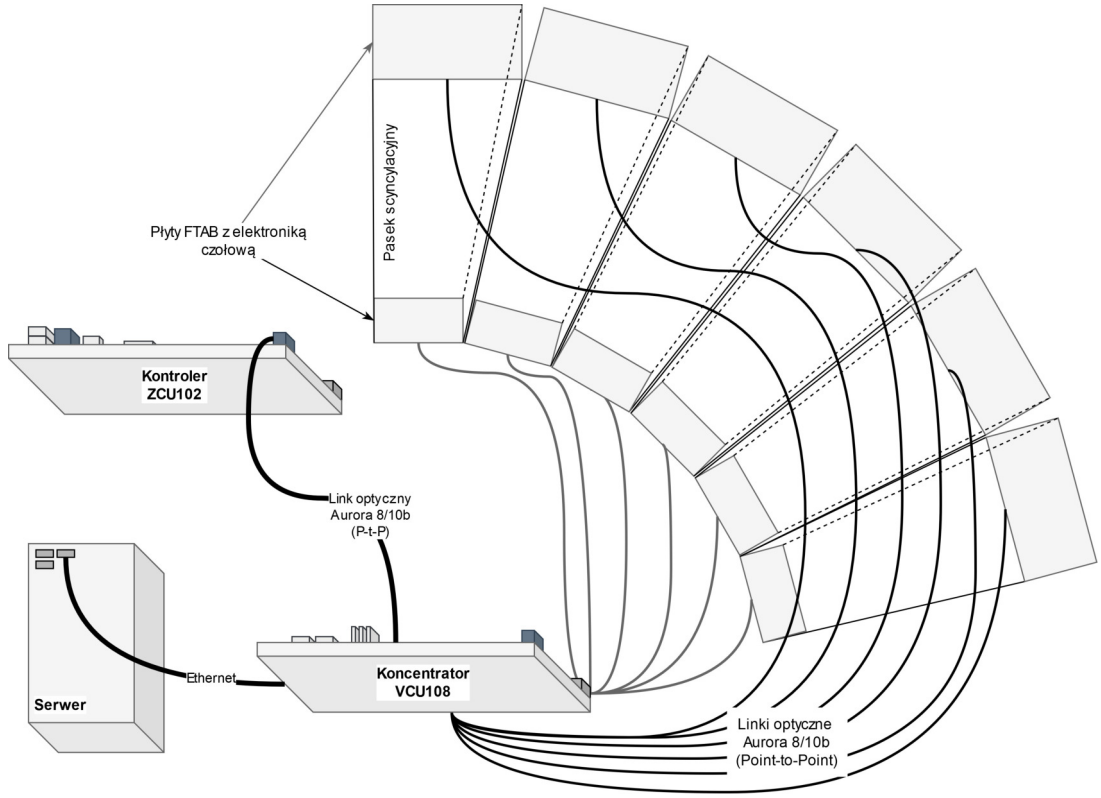
5. Tomograf CM-PET

W poniższym rozdziale opisano architekturę systemu skanera CM-PET. Przedstawiono zaimplementowany w tomografie hierarchiczny system akwizycji i agregacji danych. W kolejnych podrozdziałach przedstawiono ogólny zarys architektury skarnera CM-PET, układu kontrolera oraz koncentratorów i systemu akwizycji danych.

5.1. Architektura systemu CM-PET

Odczyt danych z całego skanera (ok. 5 tys. kanałów pomiarowych) wymaga zastosowania odpowiedniej architektury systemu. Podobnie jak w większych eksperymentach fizycznych (np. ATLAS [21], HADES) w skanerze CM-PET wykorzystano hierarchiczną strukturę systemu pomiarowego. System składa się z trzech warstw: płyt z elektroniką czołową, koncentratorów oraz kontrolera. Dane periodycznie spływają z płyt czołowych do koncentratorów, a następnie po obróbce są przesyłane do kontrolera.

Prototyp skanera CM-PET zbudowany jest z 24 modułów. Każdy moduł składa się z 13 pasków scyncylacyjnych, co finalnie daje liczbę 312 pasków w całym tomografie. Do dystalnych końców modułów przymocowane są płyty z elektroniką zawierającą matryce krzemowych fotopowielaczy oraz układy PHA. Matryce fotopowielaczy zbudowane są z diód lawinowych z efektem Geigera ułożonych tak, by na każdy pasek detektora przypadały cztery diody. Płytka z fotopowielaczami podłączona jest do płyty FTAB, która odczytuje, dyskryminuje i digitalizuje sygnały z fotopowielaczy. Na płycie FTAB znajduje się szereg filtrów analogowych odpowiedzialnych za dopasowanie sygnałów (ang. signal conditioning) oraz układ FPGA dokonujący konwersji TDC. Dane z 48 płyt czołowych spływają do czterech koncentratorów za pomocą światłowodowych łącz. Każdy koncentrator agreguje dane z 12 płyt, a więc sześciu modułów. Surowe dane z koncentratorów są przesyłane na serwer w celu ich przechowania do dalszej, programowej obróbki (np. poprzez użycie J-PET framework [9]). Jednocześnie na koncentratorze zaimplementowano pierwszy stopień przetwarzania danych, czyli mechanizm selekcji, parowania wydarzeń oraz obliczania pozycji ich wystąpień na poszczególnych paskach scyncylacyjnych. Przetworzone dane są przesyłane do płyty kontrolera, gdzie następuje ich dalsza obróbka, czyli parowanie w linie odpowiedzi. Opisywany system został przedstawiony na rys. 5.1.



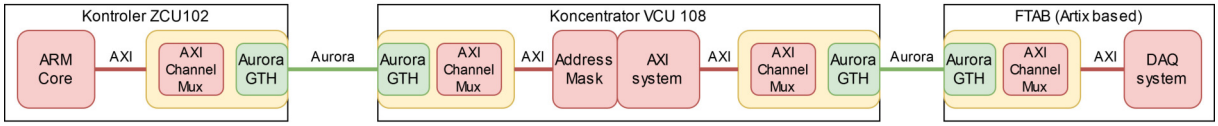
Rys. 5.1. Schemat systemu akwizycji danych skanera CM-PET. Dla zachowania przejrzystości na rysunku przedstawiono tylko sześć pasków oraz jeden koncentrator.

5.2. Kontroler

Kontroler systemu zbudowany jest w oparciu o układ MPSoC znajdujący się na płycie Xilinx ZCU102 Evaluation Board. Układ Zynq UltraScale+ MPSoC złożony jest z czterordzeniowego procesora ARM A-53, dwóch procesorów czasu rzeczywistego ARM R5, układu graficznego ARM MALI400 MP2, oraz średniej wielkości układu rekonfigurowalnego (600 tys. bloków CLB, 2500 mnożarek DSP oraz około 30 Mb pamięci wewnętrznej). Układ ten jest także wyposażony w MMU (ang. **M**emory **M**anagment **U**nit), co sprawia że system nadaje się zarówno do tworzenia aplikacji *bare-metal*, jak i do obsługi systemów operacyjnych. Kontroler jest połączony z czterema koncentratorami za pomocą łącz optycznych. W zależności od konfiguracji łącza te służą do przesyłania surowych lub wstępnie przetworzonych danych pomiarowych. Za pomocą systemu *slow-control* z kontrolera wysyłane są także instrukcje sterujące do koncentratorów oraz płyt czołowych. Dodatkowo jest tam generowany periodyczny sygnał wyzwalający odczyt danych z płyt czołowych.

W układzie FPGA systemu MPSoC zaimplementowano strukturę opartą o *IP-Core* AXI Chip2Chip. Moduł ten umożliwia zaimplementowanie mostka (ang. bridging) pomiędzy różnymi układami scalonymi wykorzystującymi magistralę AXI. Podobne moduły zostały zaimplementowane w układach koncentratorów oraz płyt czołowych. Moduł AXI Chip2Chip agreguje kanały przerwań, AXI Lite oraz AXI

w jeden strumień [22]. Strumień ten jest następnie enkapsulowany protokołem Aurora i przesyłany pomiędzy płytami za pomocą transceiver'ów GTH. Poniżej przedstawiono schemat komunikacji pomiędzy poszczególnymi elementami systemu.



Rys. 5.2. Schemat komunikacji pomiędzy układmi systemu. Na podstawie [22]

Z punktu widzenia procesora układy koncentratorów oraz płyt FTAB znajdują się w lokalnej przestrzeni adresowej. Dwupoziomowy dostęp do płyt czołowych z poziomu procesora jest możliwy dzięki odpowiedniemu maskowaniu adresów AXI w układach koncentratorów.

W celu zapewnienia środowiska do obsługi aplikacji do kontroli skanera, akwizycji danych oraz ich wizualizacji na procesorach układu MPSoC uruchomiono dedykowaną dystrybucję systemu Linux, tj. PetaLinux. Dzięki temu możliwe jest między innymi tworzenie prostych programów do obsługi skanera bez konieczności użycia - często uciążliwej - kompilacji skrośnej (ang. cross-compilation). Sposób przygotowania dystrybucji opisano w rozdziale 6.

5.3. Koncentrator

W skanerze CM-PET są cztery koncentratory, które agregują sygnały z 12 płyt FTAB każdy. Prócz zbierania danych układy koncentratorów służą także do rozprowadzania sygnałów kontrolno-wyzwalających z kontrolera do płyt czołowych. Koncentrator został zbudowany w oparciu o płytę Xilinx VCU108 Evaluation Board rozszerzoną o moduły powiększające liczbę łącz światłowodowych do 14. Układ Virtex UltraScale znajdujący się na tej płycie ma ponad 1 mln bloków CLB, 750 mnożarek DSP oraz około 60 Mb pamięci wewnętrznej. Podobnie jak w kontrolerze w układzie koncentratora zaimplementowano moduły AXI Chip2Chip. Ponadto wykorzystanie odpowiednio skonfigurowanej maski adresów AXI umożliwiło transparentną, dwupoziomową propagację transakcji AXI z kontrolera do płyt czołowych (rys. 5.2).

W podstawowej konfiguracji koncentrator przesyła zebrane dane na serwer za pomocą łącza Ethernet. Wielkość układu FPGA znajdującego się na płycie ewaluacyjnej VCU108 pozwoliła jednak na zaimplementowanie wstępnej obróbki danych (tj. selekcji oraz parowania zdarzeń). Wstępnie przetworzone dane są następnie przesyłane do kontrolera, gdzie następuje ich dalsza obróbka (tj. łączenie w LOR oraz wizualizacja). System przetwarzania będący przedmiotem tej pracy został opisany w rodziale 6.

5.4. Układ TDC na płytach FTAB

W skanerach PET kluczowym aspektem zapewniającym wysoką jakość obrazu jest precyzyjny pomiar czasu. Wysoka rozdzielczość czasowa skanera odgrywa jeszcze większą rolę w skanerach typu TOF

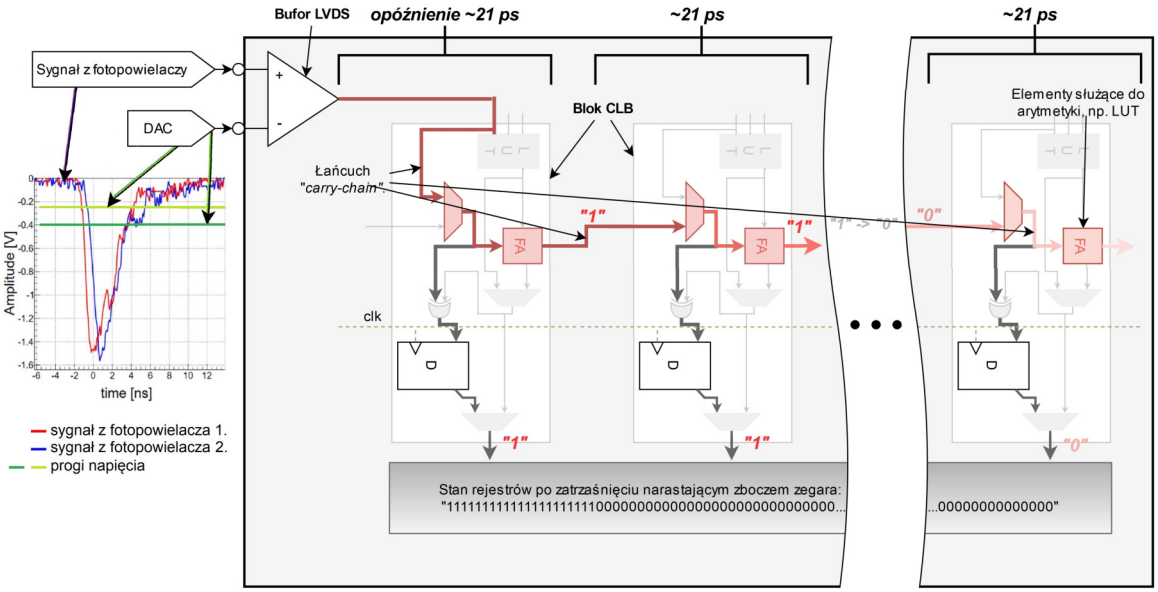
PET, gdzie informacja o czasie przelotu kwantów gamma jest bezpośrednio wykorzystywana w rekonstrukcji obrazu [23]. Tradycyjne układy elektroniczne w skanerach PET wykorzystują komparatory do dyskryminacji sygnałów oraz przetworniki TDC do konwersji informacji o czasie wystąpienia wydarzenia. Zazwyczaj oba te urządzenia zaimplementowane są w jednym układzie wyspecjalizowanym. W rozdziale 4 przytoczono główne wady rozwiązań wykorzystujących układy ASIC.

W skanerze CM-PET została zaproponowana metoda wykorzystująca układy reprogramowalne Xilinx Artix-7. Jak wykazano w „*A compact system for high precision time measurements...*" [19] rezygnacja z urządzeń ASIC na rzecz układów rekonfigurowalnych może być korzystna nie tylko ze względów ekonomicznych, ale także wydajnościowych - w odpowiednich warunkach możliwe jest bowiem uzyskanie rozdzielczości czasowej około 10 ps.

W celu implementacji urządzenia TDC w układzie FPGA użyto buforów wejściowych LVDS oraz logiki *carry-chain*. Bufory LVDS są używane, gdy istnieje potrzeba bardzo szybkiej i odpornej na zakłócenia transmisji danych. Służą do konwersji sygnałów różnicowych na sygnały unipolarne (ang. single-ended signals). W istocie, bufor LVDS jest bardzo szybkim komparatorem (zdolnym do przetwarzania nawet do 10 Gb/s), który może zostać użyty do porównywania różnych sygnałów. W elektronice czołowej skanera CM-PET na jedno z wejść bufora podaje się próg napięcia, a do drugiego doprowadza się sygnał pochodzący z fotopowielaczy. W ten sposób dokonuje się dyskryminacji sygnałów [12]. Zdyskryminowany sygnał jest następnie przesyłany do modułów dokonujących konwersji TDC. Do zaimplementowania przetworników TDC użyto logiki *carry-chain*. Są to specjalne łańcuchy asychronicznych połączeń pomiędzy blokami CLB służące do przenoszenia bitów *carry* podczas wykonywania operacji arytmetycznych. Każde z ogniw łańcucha *carry* wprowadza opóźnienie rzędu 10-30 ps, oraz jest sprzężone z rejestrem typu D. Rejestry te w momencie wystąpienia narastającego zbocza zegara zatrzaskują się i zapisują stan linii *carry-chain*. Przy odpowiednio skonfigurowanej logice możliwa jest więc konwersja TDC z rozdzielczością odpowiadającą opóźnieniem poszczególnych elementów łańcucha *carry* [24]. Ideowy schemat sposobu działania tego układu został przedstawiony na rys. 5.3.

W celu zwiększenia dokładności systemu należy dodatkowo zindentyfikować opóźnienia wprowadzone na ścieżkach układu FPGA (w szczególności różnice opóźnień pomiędzy elementami łańcucha *carry-chain*), oraz określić wpływ temperatury i napięcia zasilania na pracę układu. Po zebraniu wyżej wymienionych parametrów możliwa jest korekcja zmierzonych czasów, czy to w analizie *offline*, czy podczas przetwarzania w czasie rzeczywistym - np. w formie tablic mapujących.

Zaimplementowane w skanerze CM-PET układy TDC cechują się rozdzielczością czasową równą ok. 21 ps. Umożliwiają także ustawienie dwóch progów napięcia oraz niezależne wykrywanie narastających i opadających zbocz sygnałów. Na tej podstawie możliwe jest zgrubne określenie kształtu oryginalnego sygnału, a więc i ładunku który niesie [3][12].



Rys. 5.3. Uproszczony schemat przetwornika TDC. W odcieniach czerwieni zaznaczono ścieżki opóźniające sygnał, kolorem ciemnoszarym - ścieżki „przechwytujące" stan łańcucha *carry*. Na wykresie po lewej (źródło [25]) przedstawiono odpowiedź fotopowielaczy na zdarzenie (kolor czerwony i niebieski) oraz dwa progi napięcia (odcienie zieleni). Każdy fotopowielacz oraz odpowiadająca mu wartość progu dyskryminacji podłączone są do osobnych układów TDC.

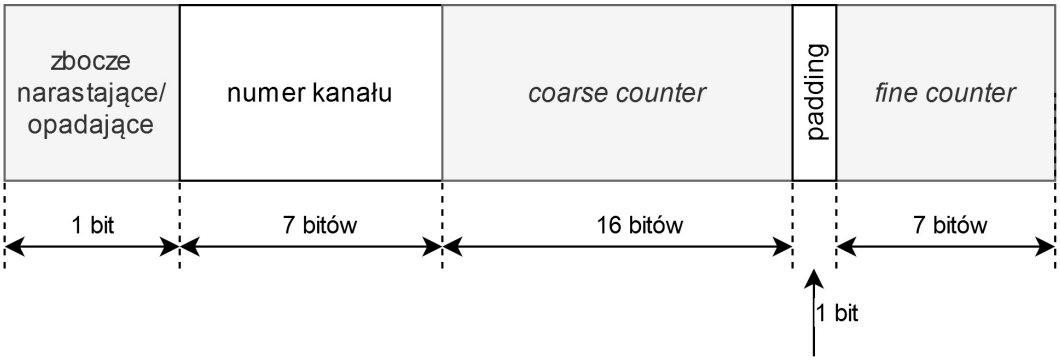
.

5.5. Odczyt danych z płyt FTAB

W tomografie CM-PET na każdy pasek przypadają cztery krzemowe fotopowielacze. Zestawiając to z dwoma progami dyskryminacji, wykrywaniem narastającego oraz opadającego zbocza sygnału, każde zdarzenie jest w stanie wygenerować maksymalnie do 16 próbek. Zebrane przez DAQ dane są wstępnie gromadzone w pamięci FIFO umieszczonej obok układu TDC. W momencie odebrania sygnału wyzwalającego z kontrolera dane te są zbiorczo wysyłane do koncentratorów. Każda informacja o wykrytym zdarzeniu jest przesyłana za pomocą 32-bitowego słowa. Jego struktura została pokazana na rysunku 5.4.

Pierwszy bit sygnalizuje zbocze narastające lub opadające sygnału, kolejne siedem bitów koduje kanał pomiarowy. Następne 24 bity oznaczają czas wystąpienia zdarzenia - 16 bitów to tzw. *coarse counter*, a siedem ostatnich to *fine counter*. *Coarse counter* to licznik taktowany głównym zegarem układu TDC o częstotliwości 400 MHz. Jego długość jest dobrana tak, by nie przepełnił się on więcej niż jeden raz w ciągu okna pomiarowego (trwającego 50 μs). Licznik *fine counter* to stan łańcucha *carry-chain* zatrzaśnięty ostatnim zboczem narastającym sygnału zegara TDC po wystąpieniu zdarzenia.

Za pomocą systemu *slow-control* rozprowadzany jest cykliczny sygnał wyzwalający odczyt tych danych. Jego częstotliwość wynosi 20 kHz. Sygnał ten pełni podwójną rolę - wyzwala odczyt danych z



Rys. 5.4. 32-bitowe słowo kodujące czas oraz miejsce wystąpienia zdarzenia

pamięci FIFO, oraz stanowi czas referencyjny. Podobnie jak sygnały z fotopowielaczy sygnał wyzwalający jest przesyłany do układu TDC w celu konwersji. Dysponując stanami liczników *coarse counter* oraz *fine counter* zarówno zdarzeń, jak i sygnału referencyjnego możliwe jest odtworzenie czasu wystąpienia zdarzenia.

Strumień spływających z płyty FTAB danych jest enkapsulowany nagłówkiem zawierającym numer samej płyty i słowa kontrolne. Dodatkowo same dane, poprzez sposób ich gromadzenia są posegregowane kolejno pod względem: klastra[1]; zbocza; numeru kanału. Ta ostatnia cecha jest szczególnie istotna w zaimplementowanym systemie przetwarzania.

[1] zbiór próbek które można przyporządkować pojedynczemu zdarzeniu.

6. Zaimplementowany system

Poniższy rozdział został podzielony na 5 sekcji. W pierwszej części umotywowano wykorzystanie układów FPGA do realizacji wstępnego przetwarzania danych. Zdefiniowano także wymagania względem zaprojektowanego systemu. W kolejnym podrozdziale przedstawiono zarys architektury zaimplementowanego układu wstępnego przetwarzania danych. Szczególny nacisk położono na opisanie wykorzystanych metod zrównoleglenia obliczeń. W załączniku A przedstawiono wysokopoziomowy schemat architektury systemu. Osobne podrozdziały przeznaczono także na opis środowiska weryfikacyjnego, systemu operacyjnego PetaLinux i sposobu wizualizacji rezultatów przetwarzania.

6.1. Motywacja

W tomografii PET ilość surowych danych generowanych przez skaner jest bardzo duża (np. rzędu setek Mb/min dla czterech modułów skanera CM-PET). Skutkiem tego przeprowadzono wiele badań mających na celu zredukowanie liczby danych poprzez ich wstępne przetwarzanie. Najpopularniejsze urządzenia są oparte o układy GPU, które wykorzystuje się do akceleracji obliczeń *offline* [26][27]. Pojawiły się także rozwiązania wykorzystujące układy FPGA w celu implementacji logiki parującej wydarzenia w koincydencje w czasie rzeczywistym (ang. coincidence finder) [28].

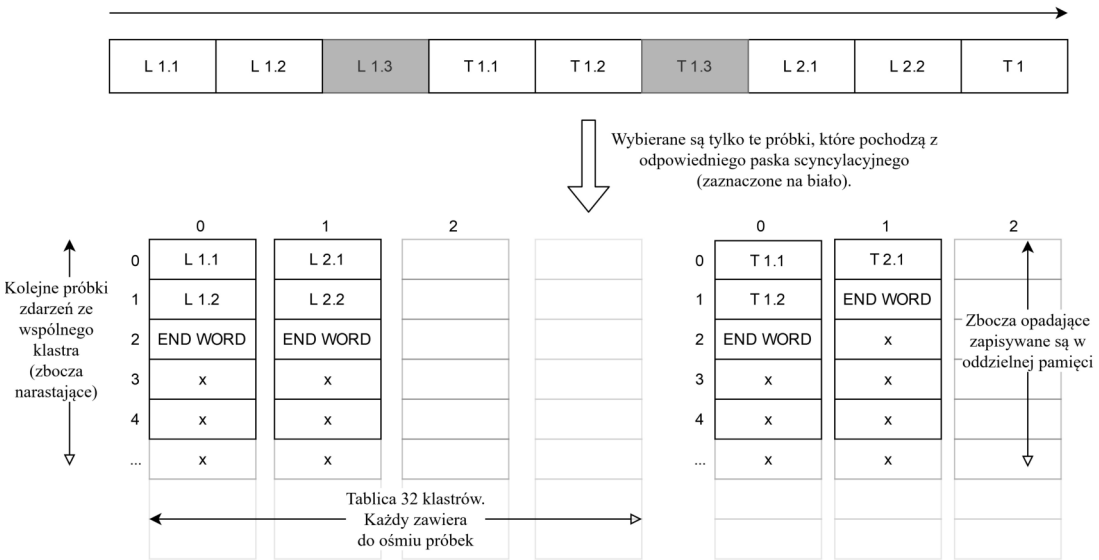
Cyfrowa elektronika wykorzystana w skanerze CM-PET jest niemalże w całości oparta o układy FPGA (co zostało opisane w rozdziale 5). W podstawowej konfiguracji skanera układy rekonfigurowalne wykorzystywane są tylko i wyłącznie do akwizycji danych oraz do kontroli urządzenia. Pojemność wykorzystywanych układów FPGA pozwoliła jednak na zaimplementowanie mechanizmów wstępnej obróbki danych w czasie rzeczywistym. Długoterminowym celem prowadzonych prac jest implementacja wybranego algorytmu rekonstrukcji tomograficznej w czasie rzeczywistym. W ramach tej pracy podjęto się zaimplementowania i opisania następujących mechanizmów wstępnego przetwarzania: selekcji i odzyskania wydarzeń z surowych danych tomograficznych, parowania koincydencji na poszczególnych paskach detektora, parowania koincydencji w linie odpowiedzi (LOR) pomiędzy modułami skanera.

Podstawowym wymaganiem postawionym przed systemem jest to, by czas potrzebny na wykonanie wyżej wymienionych obliczeń nie przekraczał 50 μs. Nałożony reżim czasu rzeczywistego ma przede wszystkim zagwarantować, by podczas przetwarzania próbki z różnych okien pomiarowych nie mieszały się. Dodatkowo założono, że układ powinien móc przetworzyć maksymalnie do 32 klastrów (każdy składający się z maks. 16 próbek) wykrytych na każdym pasku tomografu.

6.2. Architektura zaimplementowanego systemu

Zaprojektowany system przetwarzania danych został podzielony pomiędzy dwa układy - koncentratora i kontrolera. Na koncentratorze zaimplementowano wszystkie operacje, które wymagają jedynie kontekstu pojedynczych modułów. Są to metody selekcji, parowania zdarzeń w koincydencje oraz obliczania czasu i miejsca uderzenia kwantu gamma w pasek skanera. W układzie kontrolera zaimplementowano systemem parowania koincydencji w linie odpowiedzi. Schemat zaprojektowanej architektury został umieszczony w załączniku A (wyszczególniono na nim poszczególne sekcje systemu do których odwołano się w poniższym rozdziale).

W pierwszej fazie (zał. A, sek. 1) dane spływające z poszczególnych płyt czołowych przetwarzane są niezależnie. Potok danych przesyłany jest przez kolejne moduły celem usunięcia próbek, które zawierają błędy lub są niekompletne (najczęściej pojawiające się błędy zostały opisane w załączniku B). Dla każdej próbki na podstawie wartości liczników *coarse* i *fine* obliczany jest czas wykrycia zdarzenia względem początku okna pomiarowego. Także na tym etapie próbki rozdzielane są pomiędzy 13 równoległych torów przetwarzania (każdy tor jest przyporządkowany do jednego paska scyncylacyjnego). Strumień danych jest następnie mapowany do pamięci dwuwymiarowej (tablicy przechowującej do 32 klastrów, każdy po maksymalnie osiem próbek). Na rysunku 6.1 przedstawiono ideę mapowania dla przykładowych danych.



Rys. 6.1. Ilustracja procesu rozdzielania oraz mapowania danych do pamięci.

Dane spływające z płyt czołowych są posortowane kolejno względem klastrów, rodzaju wykrytego zbocza oraz numeru kanału TDC. Dzięki temu próbki zapisane w pamięci są uporządkowane rosnąco względem numerów kanałów. Ta konkretna cecha umożliwia szybkie (bo bez konieczności przeszukiwania całej przestrzeni pamięci) łączenie próbek kodujących zbocze narastające z tymi, w których zapisane jest odpowiadające im opadające zbocze sygnału. Próbki do których nie udało się przyporządkować odpowiedniego zbocza są odrzucane. Z połączonych próbek w ramach pojedynczego klastra obliczana jest

średnia arytmetyczna. Kłopotliwa w realizacji w układach FPGA, ale wymaganana podczas obliczania średniej operacja dzielenia została zastąpiona przez iloczyn (zgodnie z $\frac{d}{q} = d * \frac{1}{q}$). Tym samym czas potrzebny na obliczenie średniej został zredukowany z blisko 40 do maksymalnie dziewięciu cykli zegara (w załączniku B przedstawiono więcej szczegółów na ten temat).

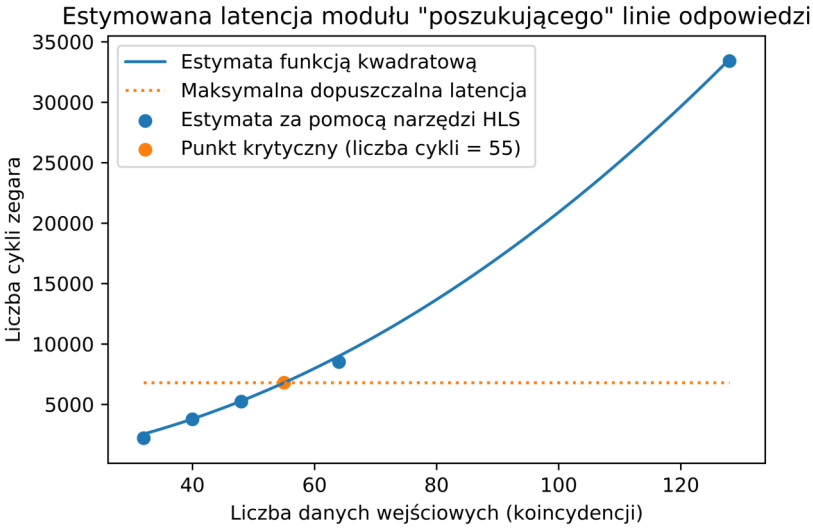
W kolejnym kroku dwie tablice przechowujące uśrednione dane z strony A oraz B modułu tomografu przeszukiwane są pod kątem koincydencji (zał. A, sek. 2). Działanie zaimplementowanego w HLS modułu „poszukującego" opiera się na dwóch zagnieżdżonych pętlach *for* iterujących po każdej komórce pamięci. Jeżeli różnica czasów wykrycia porównywanych zdarzeń zawiera się określonym przedziale, to są one parowane w koincydencję. Tak jak zostało to przedstawione na schemacie w załączniku A, w układzie koncentratora zaimplementowanych jest łączenie 78 równoległych torów służących do szukania koincydencji. Ścieżki te są ostatecznie agregowane do jednego strumienia poprzez szereg pamięci FIFO oraz tzw. lejków (ang. funnel).

Ostatnią operacją przeprowadzaną w układzie koncentratora jest estymacja miejsca uderzenia kwantu gamma w pasek scyncylatora. Informacja jest wyliczana na podstawie różnicy czasów wykrycia zdarzeń poprzez detektory, zgodnie z wzorem $z = \frac{v}{2}(t_a - t_b)$ (gdzie v to prędkość rozchodzenia się światła w scyncylatorze, ok. 12 $\frac{cm}{ns}$).

Przetworzone na koncentratorze dane są następnie przesyłane do układu kontrolera. W kontrolerze następuje ich ponowne rozdzielenie pomiędzy równoległe ścieżki przetwarzania (zał A, sek. 3). Tym razem podział ten odbywa się na podstawie czasu zarejestrowania zdarzenia. W każdym z torów przetwarzania, w sposób niezależny od pozostałych odbywa się parowanie koincydencji w linie odpowiedzi. Liczba przedziałów czasowych może zostać dowolnie ustalona, jednak domyślnie przyjęto 13 torów. Zbyt duża liczba przedziałów może skutkować niechcianym odseparowaniem próbek wspólnego pochodzenia, zbyt mała zaś niepotrzebnie wydłuży czas przetwarzania. Sam mechanizm poszukiwania LOR jest bardzo podobny do sposobu parowania zdarzeń w koincydencje. Ponownie badana jest różnica czasu pomiędzy zarejestrowaniem zdarzeń. Korzystając z symetrii tomografu można na podstawie różnicy numerów ID oszacować odległość pomiędzy dwoma dowolnymi modułami (a więc i potencjalny, maksymalny czas przelotu kwantów gamma). Znalezione linie odpowiedzi są zapisywane do pamięci FIFO, skąd są odczytywane przez procesor poprzez magistralę AXI.

Ścieżka przetwarzania zaimplementowana w układzie koncetratora trwa nie dłużej niż około 2800 cykli zegara. Przesłanie danych pomiędzy układami FPGA (z płyt FTAB do kontrolera) zajmuje blisko 400 cykli. Czas wymagany na przetworzenie danych w układzie kontrolera zależy od wybranej liczby przedziałów czasowych. Na rys. 6.2 przedstawiono estymowaną liczbę cykli zegara wymaganą by przetworzyć przykładowy zbiór koincydencji. Na potrzeby symulacji ustawiono tylko jeden tor przetwarzania (przedział czasowy).

Z wykresu przedstawionego na rys. 6.2 wynika, iż warunek przetwarzania w czasie rzeczywistym zostanie spełniony jeżeli w ciągu jednego okna pomiarowego nie zostanie wykrytych więcej niż 55 koincydencji (w jednym przedziale czasowym). W praktyce zaobserwowano zaś, że dla czterech modułów



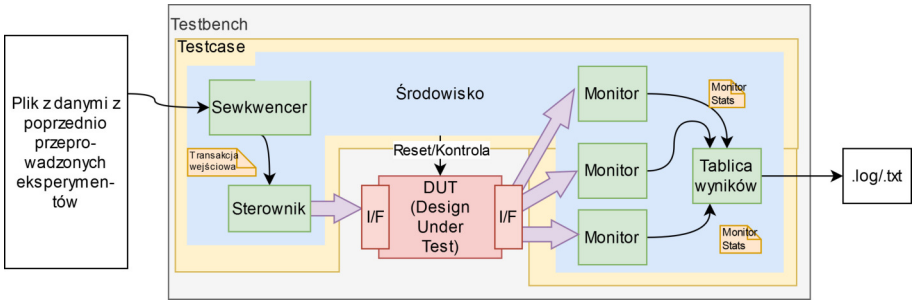
Rys. 6.2. Wymagana liczba cykli zegara do przeszukania określonego zbioru koincydencji.

skanera bardzo rzadko zdażają się sytuacje, gdy wykrytych koincydencji jest wiecej niż kilka. Jednak by zapobiec możliwości przekroczenia limitu czasu zaprojektowano odpowiedni układ resetujący.

6.3. Środowisko weryfikacyjne

W ciągu ostatnich kilku lat wielkość układów FPGA wzrosła kilkukrotnie (nawet do 10 mln elementów logicznych w największych układach) [29] [30]. Coraz pojemniejsze układy oraz rozwój narzędzi programistycznych spowodowały, że układy rekonfigurowalne są coraz częściej wybierane jako podstawowe platformy obliczeniowe, a budowane w oparciu o nie systemy są większe i bardziej skomplikowane niż kiedykolwiek były. Naturalnie wraz z wzrostem wielkości implementowanych systemów rośnie także liczba potencjalnych i ciężkich do wykrycia błędów. Zatem istotną rolę w procesie projektowania urządzenia odgrywa jego właściwe testowanie. Podstawowym środowiskiem używanym do weryfikacji układów scalonych w przemyśle jest Accellera UVM. Jest to zbiór metodologii, klas, funkcji, makr oraz bibliotek napisanych w języku SystemVerilog służących do automatyzacji procesu weryfikacji. Typowy scenariusz testu zaprogramowanego przy pomocy UVM składa się trzech faz: *Build*; *Run* oraz *Clean-up*. W pierwszym kroku budowane jest całe środowisko testowe (ang. testbench). Zazwyczaj składa się ono z sekwencera, sterownika, kilku monitorów, tablicy wyników (ang. scoreboard) oraz „fabryki obiektów” (ang. factory). W kolejnej fazie rozpoczyna się właściwe testowanie. Dane wygenerowane przez sekwencer są przesyłane do sterownika, który to za pomocą wirtualnego interfejsu stymuluje testowany system (tzw. DUT, ang. **D**esign **u**nder **T**est). Kluczowe sygnały i komponenty układu obserwowane są przez monitory (pasywne - „podsłuchujące” lub aktywne - „odpytujące”), a wynik symulacji przesyłany jest do tablicy *scoreboard*. W ostatniej fazie (*Clean-up*) rezultat symulacji zapisywany jest do pliku, a samo środowisko weryfikacyjne jest czyszczone (obiekty są usuwane, a pamięc jest zwalniana). Metodologia

UVM powstała głównie z myślą o testowaniu dużych systemów, gdzie co cykl zegara dochodzi do setek lub tysięcy transakcji. Na potrzeby prowadzonych prac postanowiono zaprogramować mniejsze i prostsze w obsłudze środowisko weryfikacyjne. Zostało ono napisane w oparciu o informacje pochodzące z artykułu „*SystemVerilog Virtual Classes, Methods...*" [31], książki „*SystemVerilog for Verification*" [32] oraz strony internetowej *testbench.in* [33]. Struktura zaprojektowanego środowiska weryfikacyjego została zaprezentowana na rys. 6.3.



Rys. 6.3. Schemat środowiska weryfikacyjnego, które zostało użyte podczas projektowania oraz weryfikacji układu wstępnego przetwarzania danych tomograficznych.

Przygotowane środowisko zostało zaprojektowane tak, by umożliwić jak najprostsze i najszybsze dołączanie kolejnych monitorów wraz z rozrastającym się modułem (więcej szczegółów na ten temat przedstawiono w załączniku B). Podczas symulacji systemu próbki którymi stymulowany jest moduł pochodzą z wcześniej przeprowadzonych eksperymentów (podczas eksperymentów surowe dane są przesyłane z koncentratora do serwera za pomocą łącza Ethernet). Na koniec rezultat symulacji porównywany jest z modelem wzorcowym napisanym w języku Python.

6.4. System PetaLinux

By w pełni wykorzystać dostępny na płycie ZCU102 heterogeniczny układ scalony przygotowano dedykowaną dystrybucję systemu PetaLinux. Do jej przygotowania wykorzystano narzędzia Xilinx PetaLinux Tools. Oprogramowanie to jest oparte na projekcie *open-source* „Yocto Project", który to zapewnia interfejs do środowisk automatycznej kompilacji skrośnej BitBake oraz OpenEmbedded [34]. Yocto dostarcza programiście odpowiednich narzędzi do przygotowania wyspecjalizowanych dystrybucji systemu Linux dla układów wbudowanych. Z dostępnych programów należy wymienić konfigurator jądra systemu oraz systemu plików (rootfs), kompilator skrośny, emulator wybranej architektury systemu QEMU oraz narzędzia wspomagające licencjonowanie programów [35].

W pracy magisterkiej „System sterująco/kontrolny dla tomografu J-PET” [36] opisującej system slow-control skanera CM-PET autor Michał Jazowski szczegółowo omówił kroki wymagane do przygotowania dystrybucji systemu PetaLinux. Zaimplementowany przez niego system operacyjny miał zapewnić podstawowe środowisko do uruchamiania aplikacji kontrolno-sterujących. Jednak na potrzeby omawianego w tej pracy projektu przygotowano nową dystrybucję systemu, rozszerzoną o obsługę układu graficznego ARM MALI400.

By prawidłowo skonfigurować system, do narzędzi Yocto należy zaimportować plik deskrypcji systemu (ang. Hardware Description File). Zawiera on informacje na temat konfiguracji procesora (listę aktywnych peryferii, modułów *IP Core* zaimplementowanych w logice, mapę pamięci itp.). Na podstawie pliku HDF konfigurator buduje drzewo urządzenia (ang. device-tree) oraz dołącza do systemu wymagane sterowniki. Następnie użytkownik może skonfigurować jądro systemu oraz system plików. W tym celu narzędzia PetaLinux dostarczają łatwe w użyciu środowisko GUI.

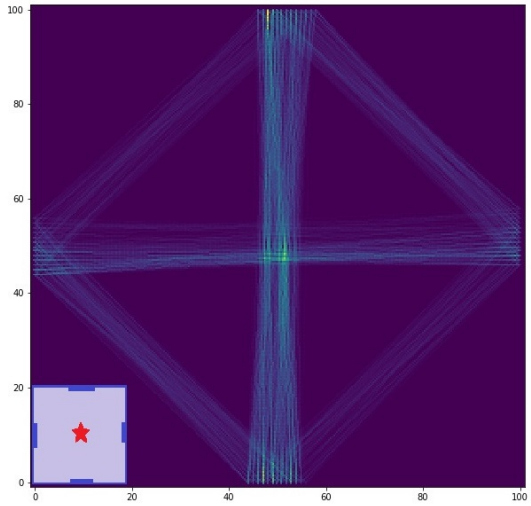
W przygotowanej dystrybucji systemu przede wszystkim dołączono oprogramowanie graficzne (libmali - obsługa układu graficznego, libdrm kms - sterowniki układu wyświetlającego, serwer X11, gstreamer, matchbox i v4lutils). Dołączono także środowisko Python, kompilator GCC i inne programy wspierające tworzenie aplikacji celem wyeliminowania konieczności przeprowadzania kompilacji skrośnej.

6.5. Wizualizacja danych

W tym podrozdziale opisano próby podjęte celem realizacji wizualizacji rezultatów przetwarzania, czyli linii odpowiedzi. W ramach pracy przygotowano dwie metody na przedstawienie danych. Pierwszą, opartą na serwisie sieciowym, oraz drugą, zrealizowaną w całości w układzie MPSoC przy użyciu wbudowanego układu graficznego.

Na potrzeby zgrubnego sprawdzenia poprawności generowanych LOR przygotowano w języku Python skrypt do ich wizualizacji. Skrypt ten pobiera wcześniej zapisane do pliku *.txt* dane i na ich podstawie generuje histogram 2D (rys. 6.4). Algorytm generowania histogramu opiera się na równaniu prostej - na podstawie dwóch punktów reprezentujacych sparowane detektory obliczane są parametry linii, która to następnie jest nanoszona na siatkę histogramu.

W celu przygotowania wizualizacji na żywo zaprogramowano prostą aplikację sieciową opartą na module Flask. Moduł ten to tzw. *micro-framework* napisany w języku Python. Zapewnia on minimalną wymaganą funkcjonalność do uruchomienia aplikacji webowej, dzięki czemu jest bardzo lekki. Sprawia to, że Flask idealnie nadaje się do implementacji serwerów WWW w systemach wbudowanych. Pierwszy utworzony serwis wykorzystuje bibliotekę Plotly oraz wcześniej wspomniany algorytm napisany przy użyciu języka Python. W tym rozwiązaniu procesor ARM jest odpowiedzialny za pobranie danych z układu FPGA, obliczenie i narysowanie histogramu 2D oraz wypełnienie obiektu biblioteki Plotly, który to zapewnia wizualizację. Obiekt Plotly jest następnie cyklicznie pobierany (i odświeżany) przez



Rys. 6.4. Przykładowy histogram 2D przedstawiający linie odpowiedzi pomiędzy czterema modułami. W lewym dolnym rogu rysunku przedstawiono miejsce umieszczenia źródła. Histogram został przygotowany na wczesnym etapie prac.

przeglądarkę użytkownika. Rozwiązanie to okazało się być mało efektywne - aplikacja działa z maksymalną częstotliwością odświeżania równą około $\frac{1}{2}$ Hz. Nie jest ona także stabilna (gdy danych jest zbyt dużo serwer często się zawiesza). Jest to spowodowane dużą złożonością obliczeniową wykorzystanego algorytmu oraz względnie małą wydajnością procesora.

W celu zwiększania wydajności systemu część zadań oddelegowano z procesora układu (tj. serwera) do przeglądarki (tj. klienta). W języku JavaScript w oparciu o bibliotekę Plotly napisano przeglądarkową aplikację do obliczania i wizualizacji histogramu. W niniejszej konfiguracji procesor ARM jest odpowiedzialny jedynie za odczytanie danych z układu FPGA oraz udostępnienie ich w zdekodowanej formie poprzez serwis sieciowy. Obliczanie i generowanie histogramu ma miejsce bezpośrednio w przeglądarce użytkownika. Rozwiązanie to znacząco odciążyło procesor kontrolera. W wyniku tej zmiany częstotliwość odświeżania histogramu wzrosła do około 1-2 Hz, a sama aplikacja stała się bardziej stabilna. Wymienione wyżej aplikacje były testowane w przeglądarce Google Chrome, a liczba wizualizowanych linii odpowiedzi nie przekraczała 50.

Rozwiązania oparte na języku Python, serwisach oraz przeglądarkach internetowych z pewnością są łatwe w implementacji. Niestety w przypadku omawianego projektu nie spełniają one podstawowego warunku - wysokiej wydajności. W celu podniesienia częstotliwości, a także liczby wizualizowanych linii odpowiedzi postanowiono napisać aplikację wykorzystującą układ MALI400 MP2.

ARM MALI400 to układ graficzny zaprojektowany w 2008 roku i udostępniany przez firmę ARM w formie *hard IP Core*. Ze względu na niewielkie rozmiary (ang. footprint) oraz niski pobór mocy najczęściej spotyka się go w urządzeniach mobilnych oraz systemach wbudowanych (np. w nawigacji samochodowej) [37]. Znajdujący się w urządzeniu Xilinx MPSoC układ MALI400 MP2:

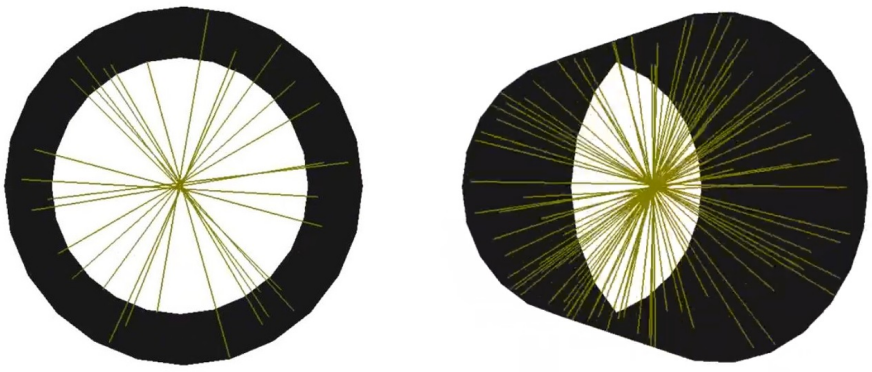
– wyposażony jest w dwa procesory pikseli przeprowadzające operacje rasteryzacji,

– maksymalnie do 256 kB pamięci wewnętrznej,
– jest taktowany zegarem do 600 MHz,
– jest kompatybilny z specyfikacją OpenGL 1.1/2.0 ES [38].

Należy jednak zaznaczyć, że układ MALI400 nie jest bezpośrednio odpowiedzialny za generowanie sygnału wideo. Operację tą wykonuje oddzielny układ wyświelający (ang. display processor) Mali DP500.

W oparciu o API OPEN GL 2.0 ES przygotowano w języku C aplikacje do wizualizacji linii odpowiedzi. Wygenerowany przy pomocy aplikacji uproszczony model tomografu CM-PET pokazany jest na rys. 6.5. Do każdego paska scyncylacyjnego przyporządkowano dwa skrajne wierzchołki (ang. vertex). Punkty te są używane nie tylko podczas generacji modelu skanera, ale służą także do mapowania poszczególnych linii odpowiedzi. Celem zachowania prostoty aplikacji (a więc i wysokiej wydajności) LOR przedstawione są za pomocą jednokolorowych odcinków będących podstawowymi kształami (ang. primitives). Model skanera został zaimplementowany przy użyciu grafiki 3D, a prezentowany jest poprzez rzut ortagonalny. Macierz przekształcenia aktualizowana jest na polecenie użytkownika (przez komendy pochodzące z klawiatury). Dzięki temu możliwy jest obrót oraz przemieszczenie modelu. Inne cechy implementacyjne, o których warto wspomnieć dotyczą zarządaniem pamięcią. Sam model skanera (współrzędne wierzchołków oraz ich indeksy) jest zapisany w pamieci wewnętrznej układu graficznego. Rozwiązanie to skraca czas potrzebny na wygenerowanie obrazu.

Aplikacja została przetestowana przy użyciu czterech modułów (rys. 7.3). W zależności od liczby generowanych linii odpowiedzi obraz jest odświeżany z częstotliwością od 15 do około 30 Hz.



Rys. 6.5. Model tomografu CM-PET. Istnieją plany zaimplementowania modelu w lepszej jakości przy użyciu MS HoloLens 2. Umożliwiłoby to wizualizację danych w tzw. rzeczywistości mieszanej.

7. Rezultaty

Poniższy rozdział poświęcono na przedstawienie rezultatów prowadzonych prac. W ostatnim podrozdziale podsumowano niniejszą pracę oraz wskazano potencjalny dalszy kierunek rozwoju.

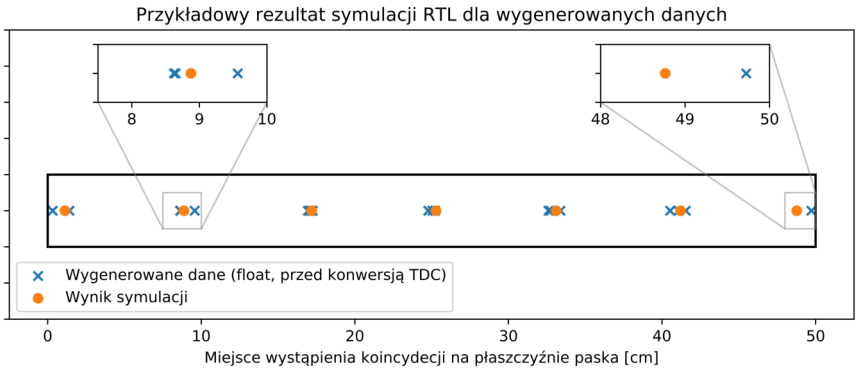
7.1. Testy symulacyjne

Podczas prowadzenia prac nad kolejnymi stopniami przetwarzania każdy z implementowanych podsystemów testowano z osobna. Wraz z rozrastającym się kodem RTL obudowywano go coraz to szerszym i bardziej skomplikowanym środowiskiem testowym. Za pomocą symulacji behawioralnej sprawdzano zachowanie systemu. Dane wejściowe do symulacji pochodziły z wcześniej przeprowadzonych eksperymentów, a rezultat symulacji był porównywany z modelem wzorcowym (przygotowanym w języku Python).

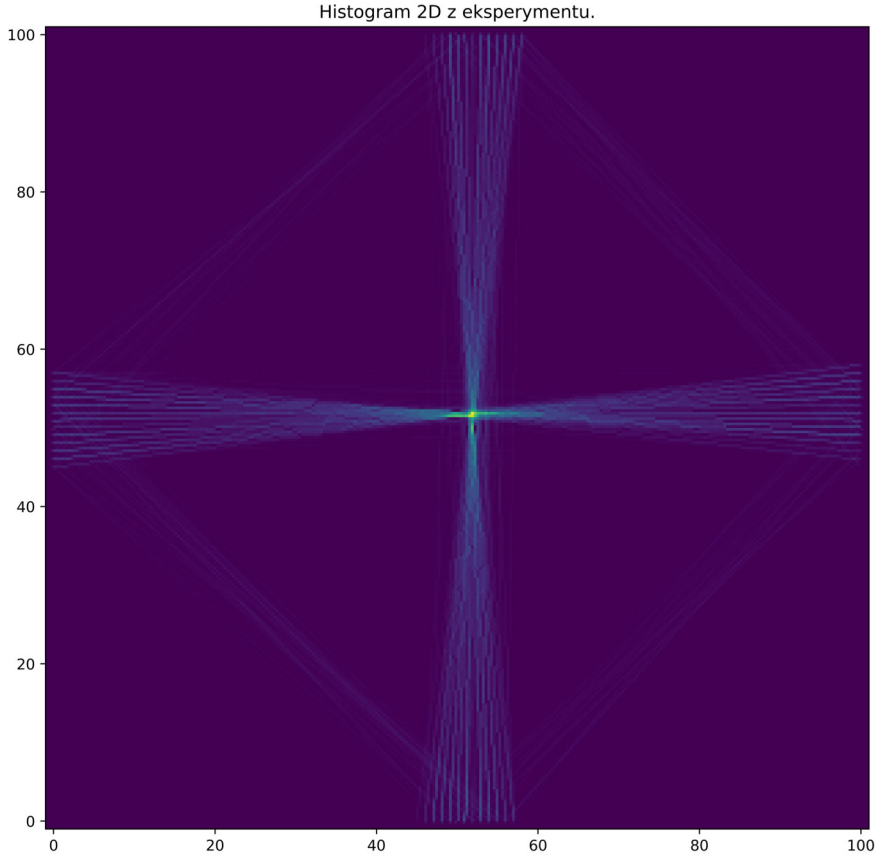
Prototyp skanera CM-PET jest jeszcze we wczesnej rozwoju. Z tego powodu nie ma możliwości przetestowania wszystkich mechanizmów wstępnej obróbki tylko i wyłącznie przy pomocy danych wygenerowanych podczas eksperymentu. Na rys. 7.1 przedstawiono przykładowy wynik symulacji RTL układu zaimplementowanego w koncentratorze (zał. B, sek. 1 i 2). Dane którymi stymulowano system zostały wygenerowane za pomocą skrytpu w języku Python. Dla każdego z siedmiu punktów (klastrów) w odpowiedni sposób wylosowano czas wystąpienia oraz liczbę zdarzeń. Wylosowane próbki (zaznaczone niebieskimi „x”) następnie poddano kwantyzacji i zapisano je w 32-bitowym słowie TDC. Przygotowane dane wprowadzono do układu tak jakby bezpośrednio pochodziły z płyt czołowych.

7.2. Testy na sprzęcie

Celem potwierdzenia poprawności działania zaimplementowanego systemu przeprowadzono serię testów. Eksperymenty przeprowadzono z użyciem czterech modułów tomografu. Punktowe źródło promieniotwórcze o aktywności 1 MBq umieszczono w centralnej części tomografu. Progi napięciowe w układach TDC ustawiono na 110 mV, a akceptowalną różnicę czasów (badaną podczas parowania koincydencji oraz LOR) ustalono na względnie dużą, bo ok. 5 razy większą niż wynikałoby to z obliczeń. Powodem przyjęcia tak dużych okien akceptacji jest wciąż udoskonalana metoda konwersji TDC oraz obliczania czasu zdarzenia na podstawie liczników *coarse* oraz *fine* (np. odpowiednie tablice mapujące, o których wspomniano w rozdziale 3 nie są jeszcze zaimplementowane).



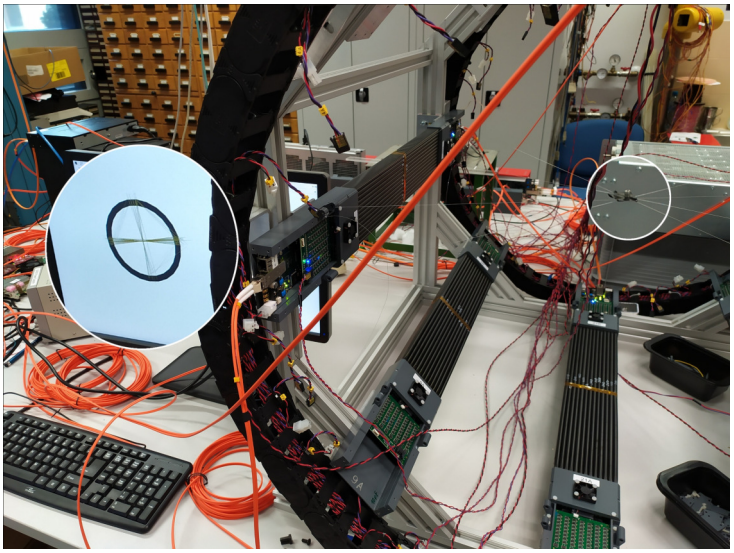
Rys. 7.1. Przykładowy wynik symulacji RTL. Testowi poddano moduł dokonujący selekcji zadarzeń, uśredniania czasów w klastrach, parowania zdarzeń w koincydencje oraz estymaty miejsca uderzenia kwantu gamma w detektor.



Rys. 7.2. Na histogramie przedstawiono ok. 2 tys linii odpowiedzi uzyskanych w trakcie eksperymentu. Eksperyment wykonano z użyciem czterech modułów tomografu. Źródło promieniotówrcze umieszczono w centralnym punkcie skanera.

Dodatkowo, w ramach tego samego eksperymentu przeprowadzono test aplikacji wizualizującej linie odpowiedzi na żywo. Na poniższym zdjęciu (rys. 7.3) przedstawiono „urywek” z działania aplikacji.

Liczba rysowanych linii odpowiedzi w jednej klatce została ograniczona do 100. Animacja wyświelała się z częstotliwością około 20 klatek na sekundę oraz sprawnie reagowała na przemieszanie źródła.



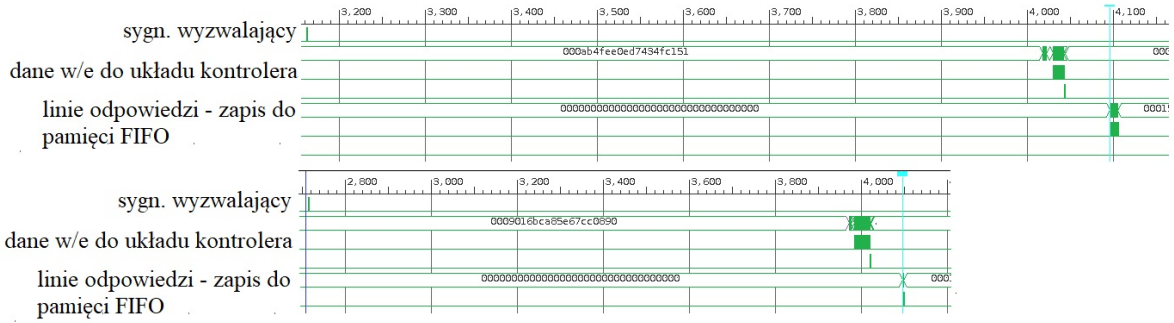
Rys. 7.3. Test aplikacji wizualizującej LOR na żywo. Obok skanera umieszczono monitor, do którego podłączono układ ZCU102 za pomocą łącza DisplayPort. Liczba wyświetlanych linii w każdej klatce została ograniczona do 100.

7.3. Przepustowość systemu

Szacowana maksymalna przepustowość zaprojektowanego systemu przetwarzania to ok. 14 MSps (tj. linii odpowiedzi). Estymaty tej dokonano na podstawie raportu wygenerowanego przez narzędzia HLS oraz ograniczeń nałożonych na urządzenie podczas jego projektowania (np. wielkości pamięci, buforów itp.). W systemie zaprojektowano specjalne układy resetujące, które w razie przekroczenia limitu czasu rzeczywistego przerywają prowadzone obliczenia i przygotowują system na nowe dane.

W bieżącej wersji skanera CM-PET sygnał wyzwalający generowany jest z częstotliwością 20 kHz. Okno pomiarowe o długości 50μs (10 tys cykli zegara o częstotliwości 100 MHz) jest wystarczające by przetworzyć spływające z tomografu dane. Należy się jednak spodziewać coraz to większej liczby danych generowanych przez skaner. Przykładowo w przyszłości sygnał wyzwalający ma być wysyłany z częstotliwością 50 kHz. Nieustannie trwają także prace nad udoskonaleniem metody detekcji i prezycji układu TDC, co również wpłynie na liczbę spływających danych.

Ponieważ architektura zaimplementowanego w układzie koncentratora systemu jest stała, to sposobów na przyśpieszenie obliczeń należy szukać w kontrolerze (modyfikując liczbę torów przetwarzania). Należy także rozważyć kwestię zmiany częstotliwości zegara taktującego moduły przetwarzania. Póki co jednak nie zaobserwowano sytuacji, gdy danych było więcej niż kilkanaście próbek. Na poniższych rysunkach zaprezentowano przykładowe przebiegi uzyskane podczas eksperymentu.



Rys. 7.4. Przykładowe zrzuty ekranu z narzędzia ILA. Pierwszy z sygnałów (sygnał wyzwalający) stanowi limit przetwarzania. Kolejne sygnały to: dane wpływające z koncentratorów do kontrolera, rezultat (linie LOR) przetwarzania. Skala na wykresach podana jest w cyklach zegara (o taktowaniu 100 MHz).

Z wykresów wynika, że przeciętny czas przetwarzania danych nie przekracza 4 tys cykli, a więc mieści się w limicie czasu rzeczywistego.

7.4. Podsumowanie

W niniejszej pracy przedstawiono projekt systemu wstępnego przetwarzania danych tomograficznych. System ten przygotowano w oparciu o układy FPGA i platformę MPSoC, a następnie zintegrowano z prototypem tomografu CM-PET. Na zaprezentowane w pracy urządzenie składają się mechanizmy selekcji zdarzeń z surowych danych tomograficznych, parowania ich w koincydencję, estymacji miejsca uderzenia kwantu gamma w powierzchnię detektora oraz łączenia koincydencji w linie odpowiedzi. Przedstawiona architektura systemu cechuje się dużym zrównolegleniem obliczeń, dzięki czemu system pracuje w reżimie czasu rzeczywistego. Dodatkowo w ramach pracy podjęto próby przygotowania wizualizacji rezultatów na żywo. Opracowano dwie metody ilustracji wyników analizy - poprzez serwis sieciowy oraz przy użyciu układu graficznego znajdującego się w platformie MPSoC. W rozdziale 7 przedstawiono przykładowy wynik eksperymentu. Rezultaty przetwarzania zostały porównane z analizą programową oraz z symulacją RTL - są one zbieżne.

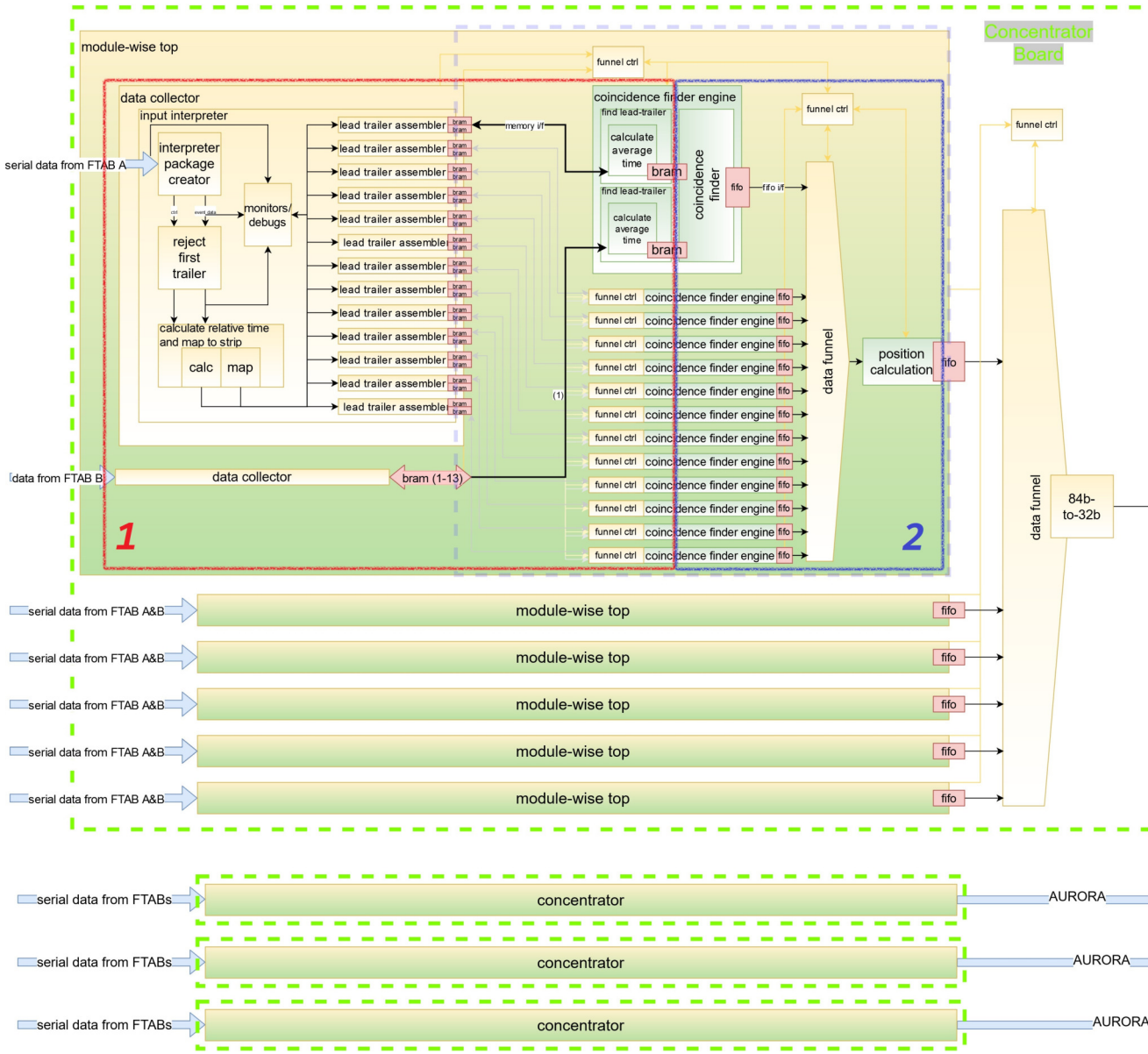
Współcześnie w układach rekonfigurowalnych implementuje się coraz to większe i bardziej skomplikowane systemy. By sprostać wymaganiom projektu wykorzystano różne narzędzia EDA (np. syntezatory HDL oraz HLS). Podczas pracy pomocne okazało się także rozbudowane środowisko weryfikacyjne, które to zapewniło szczegółowy wgląd we wnętrze układu. Dzięki temu możliwe było zaprogramowanie sprawnie działającego systemu złożonego z wielu równoległych torów przetwarzania. Dostarczone przez Xilinx narzędzia PetaLinux Tools znacząco ułatwiły integrację przygotowanego systemu z układem mikroprocesorowym dostepnym na platformie MPSoC.

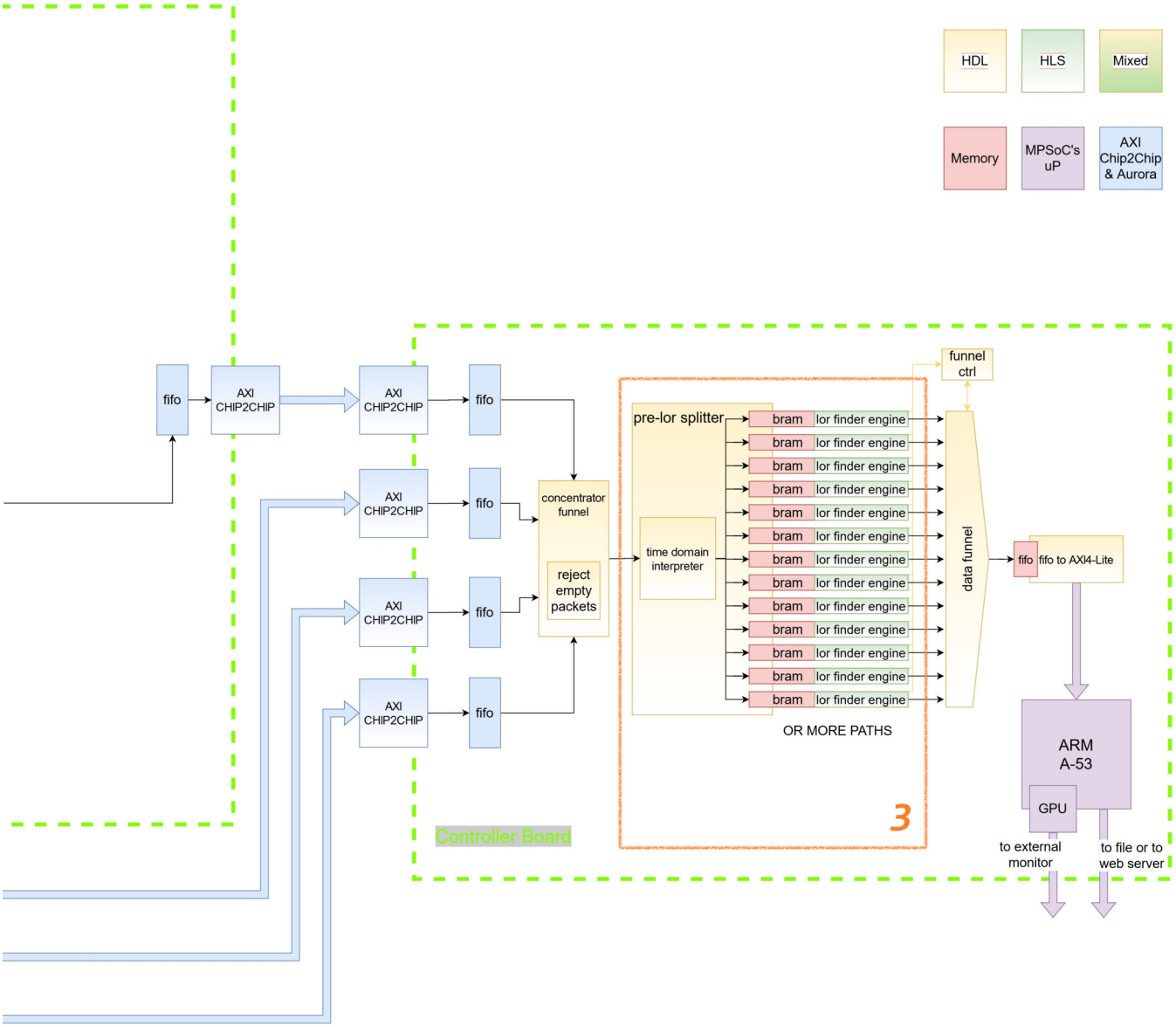
Prace nad przedstawionym systemem będą kontynuowane do momentu ukończnia projektu skanera CM-PET. Wraz z rozwojem układów DAQ do systemu będą dodawane kolejne funkcje. W piewszej kolejności należy się skupić nad zaimplementowaniem odpowiednich kanałów komunikacyjnych do kontroli przygotowanych modułów. Możliwość dynamicznej zmiany parametrów toru przetwarzania w znaczący sposób ułatwiłaby proces jego kalibracji.

Długoterminowym celem prowadzonych prac jest implementacja algorytmu rekonstrukcji obrazu tomograficznego w układach FPGA. Zaimplementowany w ramach pracy system przetwarzania stanowi podstawowy krok wielu algorytmów rekonstrukcji. Ze względu na użycie układów rekonfigurowalnych naturalnym wydaje się być implementacja analitycznego algorytmu rekonstrukcji (np. takiego jak *filtered back-projection*).

Również ciekawym kierunkiem rozwoju przygotowanego systemu jest implementacja wizualizacji w goglach Microsoft HoloLens2. Po zapewnieniu odpowiedniego kanału komunikacyjnego pomiędzy układem MPSoC a goglami, możliwa byłaby wizualizacja zrekonstruowanego obrazu w rzeczywistości mieszanej.

A. Załącznik A - schemat architektury zaimplemntowanego systemu




HDL
HLS
Mixed
Memory
MPSoC's uP
AXI Chip2Chip & Aurora
fifo
AXI CHIP2CHIP
AXI CHIP2CHIP
fifo
AXI CHIP2CHIP
fifo
AXI CHIP2CHIP
fifo
AXI CHIP2CHIP
fifo
concentrator funnel
reject empty packets
pre-lor splitter
time domain interpreter
bram
lor finder engine
OR MORE PATHS
funnel ctrl
data funnel
fifo
fifo to AXI4-Lite
ARM A-53
GPU
to external monitor
to file or to web server
Controller Board
3

B. Załącznik B

B.1. Zakłócenia w danych

Spływające dane z przetworników TDC nie są wolne od zakłóceń. W idealnych warunkach jedno wydarzenie byłoby scharakteryzowane przez 16 próbek (osiem próbek opisujących każde ze zbocz sygnału, w tym po cztery z każdego z fotopowielaczy, oraz po dwie pochodzące z dyskryminacji dwoma progami napięciowymi). Jednak spływające z płyt czołowych dane są często niekompletne lub uszkodzone. Najczęściej spotykane zakłócenia to:

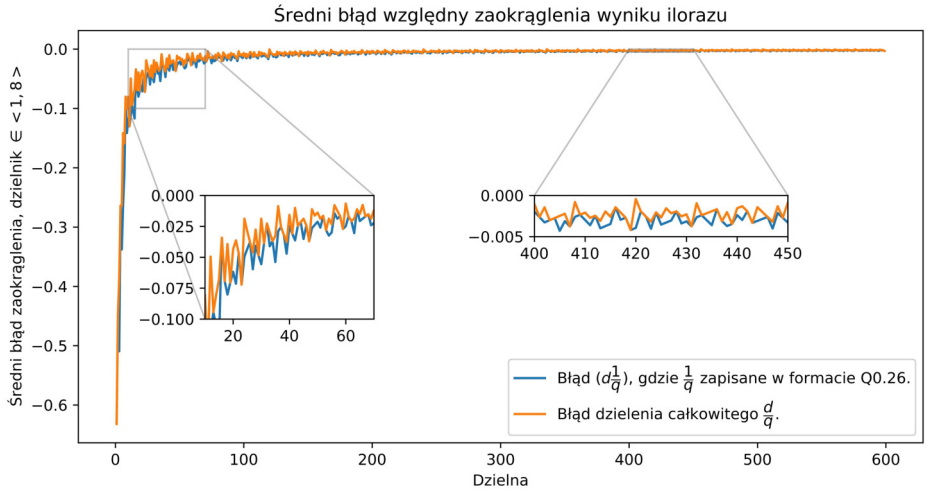
- brak czasu referencyjnego - uniemożliwia to odtworzenie czasu wystąpienia wszystkich wydarzeń.
- *cross-talking* - powoduje, że sygnał który wykryto na n-tym kanale TDC, pojawia się także na sąsiednich kanałach pomiarowych $n \pm i$.
- brak próbek kodujących zbocza narastające/opadajace sygnałów - zdarzają się wydarzenia, kiedy to np. jedynie zbocze opadające jest wykryte przez układ TDC.
- liczba próbek opisujących jedno wydarzenie przekracza 16.
- próbki spływające do koncentratorów nie są odpowiednio posortowane - jeden z mechanizmów parowania próbek w wydarzenia wymaga, by strumień próbek był posortowany rosnąco pod względem numerów kanałów TDC. Zdarzają się jednak takie dane, gdzie np. dwie próbki są zamienione kolejnością. Sprawia to, że zaimplementowana maszyna stanów przedwcześnie kończy działanie, odrzucając przy tym pozostałe dane z okna pomiarowego.

By weliminować powyższej opisane zakłócenia należało zaprojektować skomplikowaną logikę kontrolującą ścieżkę przetwarzania.

B.2. Obliczanie średniej arytmetycznej

Obliczanie średniej arytmetycznej, a w szczególności przeprowadzanie operacji dzielenia jest wyjątkowo trudne do zaimplementowania układach FPGA. Jednym z najpopularniejeszych algorytmów dzielenia liczb binarnych jest tzw. metoda restytucyjna (ang. restoring). Jest to algorytm iteracyjny, który w najgorszym przypadku wymaga 2N (lub po drobnej modyfikacji N) taktów zegara, gdzie N to długość

dzielnej w bitach. Metoda ta polega na przesuwaniu w lewo bitów słowa złożonego z dzielnej i reszty częściowej, odejmowaniu dzielnika i badaniu znaku różnicy. Gdy znak jest ujemny, oznacza to, że należy przywrócić poprzednią wartość reszty częściowej. W przeciwnym wypadku bieżącą wartość reszty należy zastąpić wynikiem odejmowania [39]. Zaimplementowany w HLS moduł obliczający średnią arytmetyczną na przeprowadzenie operacji dzielenia w tradycyjny sposób wymagał do 32 cykli zegara. Jednak ze względu na ograniczoną liczbę wartości które może przyjmować dzielnik (tj. q∈[2, 3,..., 8]) postanowiono wykorzystać podstawową własność dzielenia ($\frac{d}{q} = d\frac{1}{q}$), a wartości mnożnika zapisać w formacie stałoprzecinkowym *Q0.26*. Dzięki temu przy stosunkowo niewielkiej utracie dokładności czas wymagany na przeprowadzenie operacji dzielenia spadł z 32 cykli zegara do zaledwie dwóch. Porównanie dokładności obu metod zostało pokazane na rys. B.1.



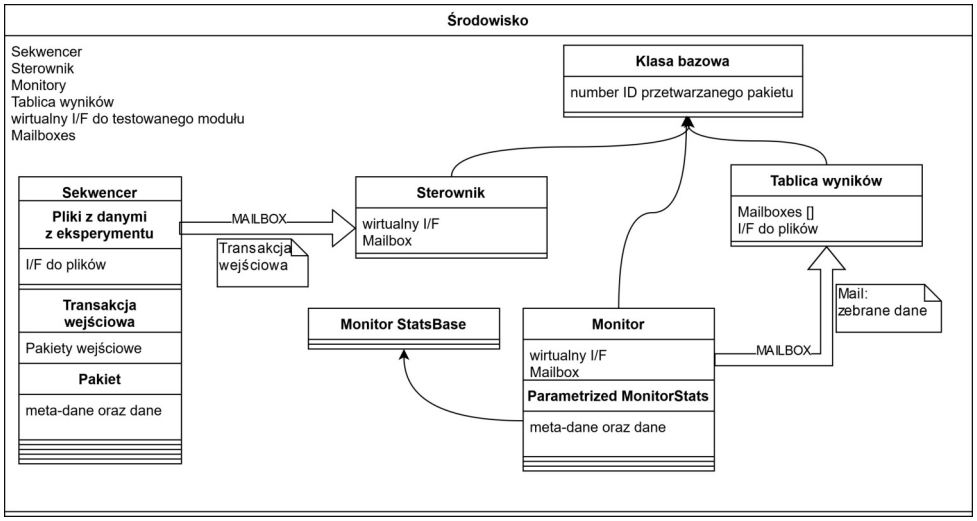
Rys. B.1. Porównanie błędów zaokrąglenia ilorazu całkowitoliczbowego uzyskanego metodą tradycyjną oraz poprzez użycie arytmetyki zmiennoprzecikowej $Q0.26$.

B.3. Środowisko weryfikacyjne - uzupełnienie

Projektowane za pomocą języka SystemVerilog środowiska weryfikacyjne cechują się wyraźnym podziałem pomiędzy częścią statyczną (zawierającą testowany model, tzw. DUT), a dynamiczną (składającej się z sterowników, sekwencerów, monitorów itd.). Wymiana informacji pomiędzy obiema częściami zrealizowana jest zazwyczaj przy pomocą tzw. wirtualnych interfejsów.

W typowym systemie zaimplementowanym w układzie FPGA komunikacja pomiędzy jego modułami oparta jest na całej gamie różnych magistral. Przykładowo w zaimplementowanym potoku przetwarzania będącym przedmiotem tej pracy można wyróżnić co najmniej trzy typy protokołów, które różnią się zarówno sygnałami kontrolnymi, jak i szerokością linii danych. Odpowiednio zaprojektowane środowisko testowe powinno w łatwy sposób (w implementacji, instancjonowaniu oraz dekodowaniu)

zapewnić połączenie do tych magistral. W tym celu zastosowano mechanizmy dziedziczenia oraz polimorfizmu, które ujednolicają sposób w jaki traktuje się przesyłane dane przez różne magistrale. Dzięki temu do zbierania danych z różnych punktów pomiarowych można użyć tej samej klasy monitora oraz skrzynki mailowej (ang. mailbox). Drzewo zależności klas zostało przedstawione na rys. B.2.



Rys. B.2. Drzewo zależności klas wykorzystanych w przygotowanym środowisku weryfikacyjnym.

Bibliografia

[1] Henry N. Wagner. „A brief history of positron emission tomography (PET)". W: *Seminars in Nuclear Medicine* 28.3 (1998). The Coming Age of Pet (Part 1), s. 213 –220. ISSN: 0001-2998. DOI: *https://doi.org/10.1016/S0001-2998(98)80027-5*.

[2] Ramsey D. Badawi i in. „First Human Imaging Studies with the EXPLORER Total-Body PET Scanner*". W: *Journal of Nuclear Medicine* 60.3 (2019), 299–303. DOI: *10.2967/jnumed.119.226498*.

[3] G. Korcyl i in. „Evaluation of Single-Chip, Real-Time Tomographic Data Processing on FPGA SoC Devices". W: *IEEE Transactions on Medical Imaging* 37.11 (2018), s. 2526–2535.

[4] P. Moskal i in. „A novel method for the line-of-response and time-of-flight reconstruction in TOF-PET detectors based on a library of synchronized model signals". W: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 775 (2015), s. 54 –62. ISSN: 0168-9002. DOI: *https://doi.org/10.1016/j.nima.2014.12.005*.

[5] Pawel Moskal i in. „TOF-PET detector concept based on organic scintillators". W: *Nuclear Medicine Review* 15 (maj 2013).

[6] Ph.D. Gopal B. Saha. *Basics of PET imaging, Physics, Chemistry and Regulations. Second Edition*. Springer, 2010.

[7] Simon Cherry, Magnus Dahlbom i Michael Phelps. *PET: Physics, Instrumentation, and Scanners*. Sty. 2006. ISBN: 978-0-387-32302-2. DOI: *10.1007/0-387-34946-4*.

[8] A. Kubica-Misztal P. Moskal. „J-PET: nowy Pozytonowy Emisyjny Tomograf zbudowany z plastikowych detektorów". W: *Inżynier i Fizyk Medycyny* 4 (mar. 2015).

[9] *„Wiki" zespołu J-PET*. URL: *http://koza.if.uj.edu.pl/petwiki/* (term. wiz. 2020-08-15).

[10] L. Raczyński i in. „Compressive sensing of signals generated in plastic scintillators in a novel J-PET instrument". W: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 786 (2015), s. 105 –112. ISSN: 0168-9002. DOI: *https://doi.org/10.1016/j.nima.2015.03.032*.

[11] L. Raczyński i in. „Statistical analysis of time resolution of the J-PET scanner". W: (2016), s. 1–4.

[12] M. Palka i in. „A novel method based solely on FPGA units enabling measurement of time and charge of analog signals in Positron Emission Tomography". W: *Bio-Algorithms and Med-Systems* 10 (list. 2013). DOI: *10.1515/bams-2013-0104*.

[13] Adam M. Alessio i Paul E. Kinahan. „PET Image Reconstruction". W: 2005.

[14] L. Raczyński i in. „Introduction of Total Variation Regularization into Filtered Backprojection Algorithm". W: *Acta Physica Polonica B* 48 (paź. 2017). DOI: *10.5506/APhysPolB.48.1611*.

[15] R.E. Henkin. *Nuclear Medicine*. Nuclear Medicine t. 2. Mosby Elsevier, 2006. ISBN: 9789997639967.

[16] *Filtered back-projection example picture*. URL: *https://sites.google.com/site/fanyangspace111/* (term. wiz. 2020-09-09).

[17] Louise H. Crockett i in. *The Zynq Book: Embedded Processing with the Arm Cortex-A9 on the Xilinx Zynq-7000 All Programmable Soc*. Glasgow, GBR: Strathclyde Academic Media, 2014. ISBN: 099297870X.

[18] Paweł Strzępek. „Development and evaluation of a signalanalysis and a readout system of straw tubedetectors for the PANDA spectrometer". Prac. dokt. Jagiellonian University, 2017.

[19] M Traxler i in. „A compact system for high precision time measurements (< 14 ps RMS) and integrated data acquisition for a large number of channels". W: *Journal of Instrumentation* 6 (grud. 2011), s. C12004. DOI: *10.1088/1748-0221/6/12/C12004*.

[20] M. Liu i in. „FPGA-Based Particle Recognition in the HADES Experiment". W: *IEEE Design Computers* 28.4 (2011), s. 48–57.

[21] K. Kordas i in. „The ATLAS Data Acquisition and Trigger: concept, design and status". W: *Nuclear Physics B - Proceedings Supplements* 172 (2007), s. 178 –182. ISSN: 0920-5632. DOI: *https://doi.org/10.1016/j.nuclphysbps.2007.08.004*.

[22] M Zieliński i in. *czipipi*. Spraw. tech. 14/2012. Instytut Fizyki, Uniwersytet Jagielloński, 2012.

[23] W. W. Moses. „Time of flight in PET revisited". W: *IEEE Transactions on Nuclear Science* 50.5 (2003), s. 1325–1330.

[24] „Odczyt danych z tomografu półprzewodnikowego". J-PET, internal document. List. 2016.

[25] M Zieliński i in. *Synchronizacja czasowa/baza danych sygnałów*. Spraw. tech. 14/2012. Instytut Fizyki, Uniwersytet Jagielloński, 2012.

[26] Nicolas GAC i in. „High Speed 3D Tomography on CPU, GPU, and FPGA". W: *EURASIP Journal on Embedded Systems* (2008). DOI: *10.1155/2008/930250*.

[27] Qi J. Zhou J. „Fast and efficient fully 3D PET image reconstruction using sparse system matrix factorization with GPU acceleration." W: *Phys Med Biol.* (paź. 2017). DOI: *doi:10.1088/0031-9155/56/20/015*.

[28] Michael Haselman i in. „FPGA-Based Front-End Electronics for Positron Emission Tomography". W: FPGA '09 (2009), 93–102. DOI: *10.1145/1508128.1508143*.

[29] *Intel Stratix 10 Product Table*. URL: *https://www.intel.pl/content/dam/www/programmable/us/en/pdfs/literature/pt/stratix-10-product-table.pdf* (term. wiz. 2020-08-23).

[30] Bouthaina Dammak i in. „Hardware resource utilization optimization in FPGA-based Heterogeneous MPSoC architectures". W: *Microprocessors and Microsystems* 39.8 (2015), s. 1108 –1118. ISSN: 0141-9331. DOI: *https://doi.org/10.1016/j.micpro.2015.05.006*.

[31] Heath Chambers Clifford E. Cummings. „SystemVerilog Virtual Classes, Methods, Interfaces and Their Use in Verification and UVM". W: (2018).

[32] Greg Tumbush Chriss Spear. *SystemVerilog for Verification*. Springer US, 2012. ISBN: 987-1-4614-0714-0.

[33] *Strona zawierająca informacje o testowaniu układów FPGA*. URL: *https://www.testbench.in* (term. wiz. 2020-08-23).

[34] *OpenEmbedded*. URL: *https://www.openembedded.org/wiki/Main_Page* (term. wiz. 2020-08-23).

[35] *Yocto Project*. URL: *https://www.yoctoproject.org/* (term. wiz. 2020-08-23).

[36] Michał Jazowski. „System sterująco/kontrolny dla tomografu J-PET". Prac. mag. Uniwersytet Jagielloński, 2019.

[37] *ARM MALI400*. URL: *https://developer.arm.com/ip-products/graphics-and-multimedia/mali-gpus/mali-400-gpu* (term. wiz. 2020-08-23).

[38] *Khronos, OpenGL 2.0 ES*. URL: *https://www.khronos.org/registry/OpenGL-Refpages/es2.0/* (term. wiz. 2020-08-23).

[39] Zbigniew Hajduk. *Wprowadzenie do języka Verilog*. Wydawnictwo BTC, 2009. ISBN: 978-83-60233-50-4.