SISTEM BAZAT PE AMD ZYNQ 7000 PENTRU VALIDAREA UNUI SISTEM DE PROCESARE DE TIP STREAMING CU INTERFATA AXI STREAM

Candidat: Stamurean Patrick Ioan

Coordonator științific: Profesor dr. ing. Oana Americai-Boncalo

Sesiunea: Iunie 2024

Cuprins

1 INTRODUCERE

1.1 Obiective

Obiectivul principal al acestei teze este de a proiecta și valida un sistem de procesare în flux bazat pe platforma AMD ZYNQ 7000, capabil să gestioneze eficient fluxurile de date în timp real printr-o interfață AXI Stream. Proiectul va integra funcționalități cheie în FPGA, inclusiv managementul stării (hold, soft reset), watchdog și programarea AXI Lite. În plus, software-ul încorporat pe procesorul dual-core ARM va calcula modelul C pentru blocurile de construcție a procesării în flux, în timp ce modulele DMA vor facilita intrarea și ieșirea datelor. Această lucrare va utiliza software-ul Vivado + Vitis pentru dezvoltare și va fi implementată pe hardware-ul ARM ZYNQ, urmărind un design de referință specificat

Obiectivele specifice includ:

1.Integrarea și configurarea componentelor hardware și software pe platforma AMD ZYNQ 7000.

2.Implementarea funcționalităților de bază și avansate în FPGA și ARM, inclusiv suport pentru întreruperi.

1.2 Context

Field-Programmable Gate Arrays (FPGA) sunt circuite integrate care pot fi configurate de utilizator după fabricare, permițând personalizarea hardware-ului pentru a satisface nevoile specifice ale aplicației. FPGA-urile sunt utilizate într-o varietate de aplicații, inclusiv procesarea semnalului, sistemele de control și calculul încorporat, datorită flexibilității, capabilităților de procesare paralelă și reconfigurabilității. Spre deosebire de microprocesoarele tradiționale, care execută instrucțiuni secvențial, FPGA-urile pot efectua mai multe operații simultan, ceea ce duce la îmbunătățiri semnificative ale performanței în aplicațiile adecvate.

Familia Zynq-7000 de la AMD combină flexibilitatea unui FPGA cu puterea de procesare a unui procesor dual-core ARM Cortex-A9. Această arhitectură face parte din familia Zynq-7000 All Programable SoC (System on Chip), care integrează atât logica programabilă, cât și sistemele de procesare pe un singur cip. Această combinație oferă o platformă versatilă pentru dezvoltarea sistemelor complexe, de înaltă performanță, care pot fi adaptate pentru sarcini specifice, făcându-l ideal pentru aplicații în sistemele auto, industriale și de comunicații.

Proiectarea și validarea unui sistem de procesare în flux bazat pe platforma AMD ZYNQ 7000 reprezintă o provocare tehnologică semnificativă datorită complexității și cerințelor de performanță impuse de aplicațiile moderne. Sistemele de procesare în flux sunt esențiale pentru gestionarea eficientă a fluxurilor de date în timp real, utilizate în diverse domenii, cum ar fi telecomunicațiile, procesarea video și analiza datelor, oferind o soluție flexibilă și de înaltă performanță pentru implementarea aplicațiilor de procesare a datelor în timp real. Utilizarea unei interfețe AXI Stream permite transferul rapid de date între diferitele componente ale sistemului, asigurând un randament ridicat și o latență scăzută.

1.3 Prezentare generală a FPGA-urilor și arhitecturii Zynq-7000

Vivado Design Suite și **Vitis Unified Software Platform** sunt instrumentele principale furnizate de AMD pentru dezvoltarea FPGA și SoC. Vivado este o suită cuprinzătoare pentru design hardware, care oferă instrumente avansate de sinteză și analiză, integrator IP (proprietate intelectuală) și capabilități de simulare. Le permite designerilor să dezvolte și să optimizeze design-uri FPGA cu eficiență și precizie ridicate.

Vitis, pe de altă parte, este un mediu de dezvoltare software conceput pentru procesoarele încorporate AMD, permițând dezvoltatorilor să programeze procesoarele ARM și logica programabilă într-o manieră unificată. Acceptă o gamă largă de modele de programare, de la C/C++ de nivel înalt și OpenCL până la limbaje de descriere hardware de nivel scăzut. Integrarea Vivado și Vitis permite un flux de lucru fără întreruperi în care accelerarea hardware poate fi gestionată eficient împreună cu dezvoltarea de software, asigurând performanța optimă a sistemului final.

1.4 Context istoric și tendințe actuale

Historical Context. **Field-Programmable Gate Arrays (FPGA)** au fost introduse în anii 1980 de către Xilinx ca o alternativă flexibilă la circuitele integrate specifice aplicației (ASIC), care sunt costisitoare și necesită timp pentru proiectare și fabricare. Primul FPGA a constat dintr-o matrice de blocuri logice configurabile (CLB) conectate prin interconexiuni programabile, permițând utilizatorilor să configureze hardware-ul în funcție de nevoile lor specifice după fabricație.

În anii 1960, introducerea MOSFET-urilor (tranzistoare cu efect de câmp metal-oxid-semiconductor) a marcat o piatră de hotar semnificativă în dezvoltarea FPGA. Acești tranzistori sunt componente fundamentale ale FPGA-urilor, acționând ca comutatoare electronice care pot fi pornite sau oprite pe baza semnalelor de intrare. Această tehnologie a pus bazele dispozitivelor logice programabile care vor evolua în FPGA-uri moderne.

Tendințe curente. Astăzi, FPGA-urile sunt utilizate într-o gamă largă de aplicații datorită performanței și flexibilității lor ridicate. Ele sunt cruciale în industrii precum telecomunicațiile, autovehiculele, aerospațiale și medicale. Progresele în tehnologia FPGA au condus la un consum redus de energie și la creșterea performanței prin tehnici precum scalarea dinamică a tensiunii și clock gating. Instrumentele de sinteză la nivel înalt (HLS) permit dezvoltatorilor să programeze FPGA-uri folosind limbaje de nivel înalt precum C/C++, simplificând procesul de dezvoltare și accelerând timpul de lansare pe piață.

1.5 Plan de executie Milestone-uri

Săptămâna 1: Inițierea proiectului
Task 1.1: Efectuați o revizuire detaliată a cerințelor și specificațiilor proiectului.
Task 1.2: Configurați mediul de dezvoltare: Instalați Vivado+EDK și pregătiți configurarea hardware-ului ARM ZYNQ.
Task 1.3: Elaborarea planului inițial al proiectului, inclusiv etapele de referință și livrabile.

Săptămâna 2: Proiectare și planificare preliminară
Task 2.1: Aprofundare în documentația AMD ZYNQ 7000 și în nucleele IP relevante (SISTEMUL DE PROCESARE ZYNQ, BRAM, FIFO etc.).

Săptămâna 3: Dezvoltarea funcționalității de bază FPGA - Partea 1
Task 3.1: Începeți dezvoltarea FPGA cu funcționalități de bază: hold, soft reset și timestamp.
Task 3.2: Implementați funcționalitatea watchdog și interfața de bază de programare AXI Lite.
Task 3.3: Efectuați testarea inițială a funcționalităților FPGA implementate.

Săptămâna 4: Dezvoltarea funcționalității de bază FPGA - Partea 2
Task 4.1: Dezvoltați FIFO de intrare și FIFO de ieșire pentru fluxul de date.
Task 4.2: Efectuați testarea inițială a funcționalităților FPGA implementate.

Săptămâna 5: Configurare DMA și integrare software
Task 5.1: Configurați DMA pentru fluxul de intrare și de ieșire.
Task 5.2: Începeți integrarea funcționalităților FPGA cu software-ul încorporat.

Săptămâna 6: Integrarea sistemului și testarea inițială
Task 6.1: Integrați toate funcționalitățile FPGA dezvoltate.
Task 6.2: Efectuați testarea integrării la nivel de sistem.

Săptămâna 7 - 8 - 9 - 10: Dezvoltare software încorporat - Configurare inițială
Task 7/ 8/ 9/ 10.1: Mediul de dezvoltare a procesorului dual-core ARM.
Task 7/ 8/ 9/ 10.2: Dezvoltarea modelului C pentru blocurile de procesare în flux.
Task 7/ 8/ 9/ 10.3: Multithreading de bază și gestionarea întreruperilor în software.

Săptămâna 11: Dezvoltare și optimizare avansată de software
Task 11.1: Optimizați software-ul încorporat pentru performanță și fiabilitate.
Task 11.2: Îmbunătățiți software-ul cu funcții suplimentare bazate pe feedback-ul de testare a integrării.
Task 11.3: Implementați capabilități avansate de gestionare a întreruperilor și multithreading.

Săptămâna 12: Testare cuprinzătoare a sistemului
Task 12.1: Efectuați teste cuprinzătoare ale sistemului, inclusiv teste de stres, performanță și fiabilitate.
Task 12.2: Rafinați sistemul pe baza rezultatelor cuprinzătoare ale testării.
Task 12.3: Validați funcționalitatea finală a sistemului în raport cu cerințele inițiale.

Săptămâna 13 - 14 - 15: Finalizare și documentare
Task 13/ 14/ 15.1: Finalizați toate funcționalitățile sistemului și asigurați conformitatea cu specificațiile proiectului.
Task 13/ 14/ 15.2: Compilați documentația cuprinzătoare a proiectului, inclusiv detalii de proiectare, cod, proceduri de testare și rezultate.
Task 13/ 14/ 15.3: Pregătiți un manual de utilizare sau un ghid pentru sistemul dezvoltat.

Săptămâna 16: Închiderea și prezentarea proiectului
Task 16.1: Pregătiți prezentarea finală a proiectului, concentrându-se pe obiective, proces, provocări, soluții și realizări.
Task 16.2: Repetați prezentarea și anticipați întrebările.
Task 16.3: Trimiteți toate livrabilele proiectului, inclusiv codul, documentația și materialele de prezentare.
Task 16.4: Prezentați proiectul.

Figure 1: Milestones Gantt Chart [**1**]

2 METODOLOGII

2.1 Configurare hardware: Descrierea Zynq-7000 ZedBoard și a dispozitivelor periferice

2.1.1 Zynq-7000 ZedBoard Prezentare generală

ZedBoard este o placă de dezvoltare și evaluare bazată pe Xilinx Zynq-7000 All Programable System-on-Chip (SoC). Această platformă inovatoare integrează un procesor ARM Cortex-A9 dual-core cu logică programabilă (PL) bazată pe tehnologia FPGA din seria 7 de la Xilinx, oferind o soluție puternică și flexibilă pentru o gamă largă de aplicații.

2.1.2 Caracteristici și specificații cheie

Caracteristici și specificații

- **Procesor și FPGA:**
  - **Procesor:** Dual-core ARM Cortex-A9
  - **FPGA:** XC7Z020-1CSG484 (Zynq-7000 AP SoC)
- **Memorie:**
  - 512 MB DDR3 RAM
  - 256 MB QSPI Flash
- **Interfete:**
  - Programare USB-JTAG
  - 10/100/1G Ethernet
  - USB OTG 2.0
  - SD Card slot
  - USB-UART bridge
  - HDMI and VGA outputs
  - 128x32 Display OLED
  - Audio (line-in, line-out, headphone, mic)

- **Conectivitate:**
  - 5 conectori Pmod (unul pentru PS (Processing System), 4 pentru PL(Programable Logic))
  - 1 LPC FMC connector
  - 1 AMS header
- **Interfata cu utilizatorul:**
  - 2 butoane reset (unul pentru PS, unul pentru PL)
  - 7 butoane
  - 8 comutatoare DIP
  - 9 LED-uri de utilizator (1 pentru PS, 8 pentru PL)
  - LED DONE care indică starea configurației FPGA

Dispozitive periferice

1. **Conectori Pmod:** ZedBoard include cinci conectori Pmod care oferă acces la I/O logic programabil. Acești conectori pot fi utilizați pentru interfața cu diverse module periferice (Pmods) pentru extinderea funcționalității plăcii.
2. **Porturi HDMI si VGA:** Aceste porturi permit conectarea ecranelor, permițând capabilități de ieșire video pentru aplicații precum procesarea video.
3. **Port Ethernet:** Portul Gigabit Ethernet facilitează conexiunea la rețea, făcând ZedBoard potrivit pentru aplicații de rețea și acces la date de la distanță.
4. **SD Card Slot:** Acceptă pornirea de pe un card SD și oferă opțiuni de stocare suplimentare pentru aplicații mai mari sau înregistrarea datelor.
5. **Interfete Audio:** Mufele de intrare, ieșire, căști și microfon sunt disponibile pentru intrare și ieșire audio, utile în aplicațiile multimedia.
6. **Display OLED:** Ecran OLED mic care poate fi utilizat pentru a afișa informații de stare, ieșiri sau elemente de interfață cu utilizatorul în aplicațiile încorporate. Aplicații și cazuri de utilizare

Această combinație de caracteristici și opțiuni de conectivitate face din ZedBoard o platformă ideală pentru prototipare rapidă, scopuri educaționale și dezvoltare de dovadă a conceptului pentru o gamă largă de aplicații

2.2 Instrumente Software

Vivado Design Suit

Prezentare Generală. Vivado Design Suite este suita completă de instrumente de la Xilinx pentru dezvoltarea designurilor FPGA și SoC. Include instrumente pentru sinteză, simulare și analiză a designurilor HDL, integrarea IP (Proprietate Intelectuală) și depanare hardware.

Caracteristici Cheie

1. **Sinteză și implementare:** Transformă proiectele HDL (VHDL/Verilog) în liste de net la nivel de poartă.
2. **IP Integrator:** facilitează integrarea blocurilor IP verificate în prealabil.
3. **Simulator Vivado:** Oferă un mediu de simulare pentru verificarea proiectelor.
4. **Analiză sincronizare și putere:** analizează caracteristicile de sincronizare și putere ale proiectelor.
5. **Hardware Debugging:** Oferă instrumente pentru depanarea în timp real și validarea proiectelor pe hardware.

Instalare și configurare

Instalare pas cu pas:

1. **Descărcați Vivado:**
   - Accesați Centrul de descărcare Xilinx.
   - Selectați versiunea corespunzătoare a Vivado Design Suite.
   - Descărcați programul de instalare complet pentru sistemul dvs. de operare.
2. **Instalați Vivado:**
   - Rulați programul de instalare descărcat.
   - Urmați instrucțiunile de pe ecran pentru a instala Vivado.
   - Descărcați programul de instalare complet pentru sistemul dvs. de operare.
   - Acceptați acordul de licență și finalizați instalarea.
3. **Configurarea Vivado:**
   - Deschideți Vivado și creați un nou proiect

- Configurați setările proiectului.

Sfaturi de configurare:

- IP Integrator: utilizați instrumentul IP Integrator pentru a adăuga și configura blocuri IP.
- Constrângeri: Aplicați constrângeri de sincronizare și fizice folosind fișiere XDC (Xilinx Design Constraints).
- Simulare: configurați mediul de simulare prin adăugarea a testbench-uri și configurarea setărilor de simulare.

Vitis Unified Software Platform

Prezentare Generală. Vitis este platforma software unificată a Xilinx, concepută pentru a funcționa cu hardware-ul lor (inclusiv FPGA-uri și SoC-uri). Acesta permite dezvoltatorilor să programeze atât sistemul de procesare, cât și logica programabilă.

Caracteristici Cheie

1. **Dezvoltare unificată:** Combină software încorporat, biblioteci accelerate și dezvoltare de modele AI.
2. **Sinteză la nivel înalt (HLS):** Permite compilarea codului C/C++ și OpenCL în descrierile hardware.
3. **Aplicații versatile:** Suportă o gamă largă de aplicații, de la sisteme încorporate la acceleratoare pentru centre de date.
4. **Integrare cu Vivado:** se integrează perfect cu Vivado pentru co-proiectare hardware/software.

Instalare și configurare

Instalare pas cu pas:

1. **Descărcați Vitis:**
   - Vizitați Centrul de descărcare Xilinx.
   - Selectați Vitis Unified Software Platform.
   - Descărcați programul de instalare.
2. **Instalați Vitis:**

- Rulați programul de instalare Vitis.
- Urmați instrucțiunile de pe ecran pentru a instala Vitis.
- Alegeți componentele de instalare în funcție de cerințele dvs.
- Acceptați acordul de licență și finalizați instalarea.

3. **Configurarea Vitis:**
    - Deschideți Vitis și configurați spațiul de lucru.
    - Importați sau creați proiecte noi.

Sfaturi de configurare:

- Configurare platformă: definiți setările platformei, inclusiv hardware-ul țintă (de exemplu, Zynq-7000).
- Dezvoltare de aplicații: Dezvoltați și depanați aplicații software folosind mediul de dezvoltare integrat.
- Accelerare hardware: Utilizați instrumentul Vitis HLS pentru a compila cod de nivel înalt în descrieri hardware compatibile cu FPGA.

Integrarea Vivado și Vitis

Integrarea fluxului de lucru:

1. **Design hardware în Vivado:**
    - Proiectați componentele hardware folosind Vivado.
    - Exportați designul hardware, inclusiv fișierele de flux de biți și de descriere hardware.
2. **Dezvoltare software în Vitis:**
    - Importați designul hardware în Vitis.
    - Dezvoltați aplicații software care vizează componentele hardware.
    - Utilizați instrumentele Vitis pentru a compila, depana și implementa aplicații.

2.3 Protocoale Folosite

Specificațiile AMBA AXI (Advanced eXtensible Interface) includ mai multe protocoale concepute pentru transferul rapid de date în circuite integrate, folosite în principal în designul sistemelor pe chip (SoC). Aceste protocoale sunt dezvoltate de ARM și sunt o parte esențială a arhitecturii ARM Advanced Microcontroller Bus Architecture (AMBA).

AXI3 face parte din specificația AMBA 3 și introduce o metodă de comunicare bazată pe canale care îmbunătățește protocolul anterior AHB, oferind performanțe superioare, o frecvență mai mare de operare și caracteristici cum ar fi completarea tranzacțiilor în ordine neconsecutivă. Acesta este potrivit pentru transferul de date de mare viteză în designuri complexe SoC.

AXI4 extinde AXI3 prin îmbunătățirea configurării lățimii interfeței și furnizarea de tranzacții burst mai eficiente. Suportă maparea de memorie de înaltă performanță și este folosit pe scară largă în aplicații care necesită un debit mare de date.

AXI4-Lite este o versiune simplificată a AXI4, proiectată pentru tranzacții mai puțin complexe, unde un debit ridicat nu este necesar. Este utilizat în mod obișnuit pentru interfețele de registru de control în cadrul SoC-urilor, datorită complexității reduse a interfeței și suprapunerii reduse de hardware.

AXI4-Stream, acest protocol este special conceput pentru datele stream. Simplifică conectivitatea prin eliminarea necesității informațiilor de adresă și control asociate fiecărui transfer de date. Este optim pentru aplicații precum transportul de date video, audio și pachete care necesită fluxuri de date continue, de mare viteză.

2.4 Componente principale ale Sistemului

Sistemul bazat pe AMD ZYNQ 7000 integrează o suită diversificată de blocuri de proprietate intelectuală (IP-uri) pentru a facilita procesarea eficientă și flexibilă a datelor într-un mediu de streaming. Aceste componente IP, esențiale pentru funcționalitatea și optimizarea sistemului, includ interfețe de comunicație, controlere de memorie, convertori de protocol și module de temporizare, toate configurate să lucreze împreună într-o arhitectură coerentă și robustă destinată aplicațiilor de procesare în timp real. IP-urile folosite in acest proiect, sunt:

2.4.1 ZYNQ 7000 Processing System (PS)

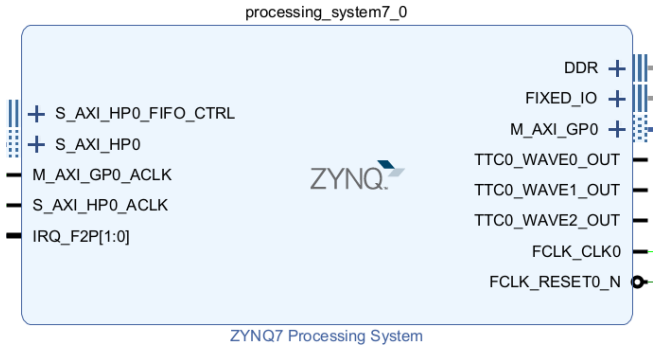

Figure 2: Zynq7000 Processing System

Familia Zynq®-7000 se bazează pe arhitectura Xilinx All Programmable system-on-chip (AP SoC). Aceste produse integrează un sistem de procesare (PS) bazat pe ARM® Cortex™-A9 MPCore™ cu două nuclee bogate în funcții și logică programabilă Xilinx (PL) într-un singur dispozitiv, construit pe un sistem de ultimă generație, Tehnologia de proces de performanță, de putere redusă (HPL), 28 nm și cu porți metalice de înaltă k (HKMG). Procesoarele ARM Cortex-A9 MPCore sunt inima PS, care include, de asemenea, memorie pe cip, interfețe de memorie externă și un set bogat de periferice I/O. Nucleul Processing System 7 este interfața software din jurul sistemului de procesare a platformei Zynq-7000. Familia Zynq-7000 constă dintr-un PS integrat în stil SoC și o unitate PL, oferind o soluție SoC extensibilă și flexibilă pe o singură matriță. Nucleul Processing System 7 acționează ca o conexiune logică între PS și PL, în timp ce vă ajută să integrați IP-uri personalizate și încorporate cu sistemul de procesare utilizând integratorul Vivado® IP.

Procesor System Reset IP Core de la Xilinx este folosit pentru a gestiona și controla operațiunile de resetare a unui întreg sistem într-un FPGA. O explicație detaliată a fiecărui pin din diagrama bloc furnizată:

1. **slowest_sync_clk:** Acest semnal de ceas de intrare este cel mai lent ceas din sistem care necesită resetare sincronizare. Acesta asigură că toate operațiunile de resetare sunt sincronizate pe diferite domenii de ceas.

2. **ext_reset_in:** Această intrare este un semnal de resetare extern, adesea conectat la un buton de resetare extern sau la alte surse externe. Inițiază o resetare atunci când este afirmată, ajutând în scenariile în care un eveniment extern ar trebui să declanșeze o resetare.

3. **aux_reset_in:** Această intrare auxiliară de resetare este utilizată pentru control suplimentar de resetare. Poate fi folosit pentru a iniția o resetare de la o sursă auxiliară, cum

ar fi un semnal software sau o altă componentă internă.

4. **mb_debug_sys_rst:** Această intrare este utilizată de obicei pentru resetarea sistemului de depanare MicroBlaze. Se asigură că procesorul MicroBlaze și logica aferentă acestuia sunt resetate corect în timpul depanării.

5. **dcm_locked:** Această intrare indică starea de blocare a Digital Clock Manager (DCM). Sistemul rămâne în resetare până când DCM este blocat, asigurându-se că toate semnalele de ceas sunt stabile înainte ca sistemul să înceapă să funcționeze.

6. **mb_reset:** Acest semnal de ieșire resetează procesorul MicroBlaze. Este afirmat ca răspuns la intrările de resetare și este sincronizat cu ceasul sistemului.

7. **bus_struct_reset[0:0]:** Acest semnal de ieșire resetează structurile magistralei din sistem. Acesta asigură că toată logica legată de magistrală este resetată și inițializată corect.

8. **peripheral_reset[0:0]:** Această ieșire resetează perifericele conectate la sistem. Acesta asigură că toate dispozitivele periferice pornesc dintr-o stare cunoscută după o resetare.

9. **interconnect_aresetn[0:0]:** Acest semnal de resetare activ-low este utilizat pentru interconexiunile AXI din cadrul sistemului. Acesta asigură resetarea și inițializarea corespunzătoare a interconexiunilor, care gestionează traficul de date între diferite componente ale sistemului.

10. **peripheral_aresetn[0:0]:** Acest semnal de resetare activ-low este pentru periferice, similar cu periferic_reset, dar special conceput pentru periferice care necesită resetare activ-low.

Miezul IP de resetare a sistemului procesorului este extrem de configurabil, permițându-i să se adapteze la diferite cerințe de sistem prin activarea sau dezactivarea diferitelor funcții de resetare. Acesta asigură un proces de resetare fiabil și sincronizat pentru toate componentele unui design FPGA, care este esențial pentru menținerea stabilității și a fiabilității sistemului.

2.4.2 Processor System Reset

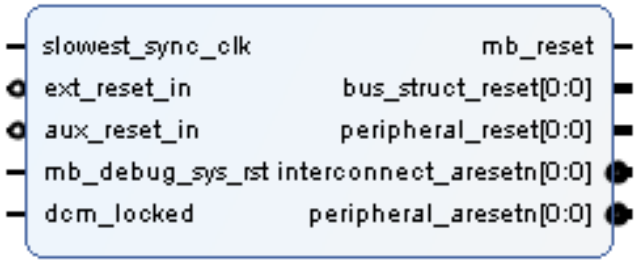

Figure 3: Processor System Reset

Processor System Reset este un IP soft care oferă un mecanism pentru a gestiona condițiile de resetare pentru un anumit sistem. Nucleul gestionează numeroase condiții de resetare la intrare și generează resetări corespunzătoare la ieșire. Acest nucleu generează resetările pe baza condițiilor de resetare externe sau interne.
Rezumatul caracteristicilor

- Atât intrările de resetare externe, cât și cele auxiliare pot fi selectate ca activ-HIGH sau activ-LOW.
- Generare resetare la pornire. Condiția Power On Reset face ca toate ieșirile de resetare să devină active în primele două clocks ale unei porniri și să rămână active timp de 16 clocks.
- Generarea semnalului de resetare activ parametrizat-scăzut pentru core și pentru interconectare.
- Secvențierea semnalelor de resetare care ies din resetare:
    1. Structurile de autobuz ies din resetare (Interconectare și pod).
    2. Perifericele ies din resetare 16 cicluri de clock mai târziu (UART, SPI, IIC).
    3. Procesorul MicroBlaze™ iese din resetare la 16 cicluri de clock după periferice.

Procesor System Reset IP Core de la Xilinx este folosit pentru a gestiona și controla operațiunile de resetare a unui întreg sistem într-un FPGA. O explicație detaliată a fiecărui pin din diagrama bloc furnizată:

1. **slowest_sync_clk:** acest semnal de ceas de intrare este cel mai lent ceas din sistem care necesită resetare sincronizare. Acesta asigură că toate operațiunile de resetare sunt sincronizate pe diferite domenii de ceas.
2. **ext_reset_in:** Această intrare este un semnal de resetare extern, adesea conectat la un buton de resetare extern sau la alte surse externe. Inițiază o resetare atunci când este afirmată, ajutând în scenariile în care un eveniment extern ar trebui să declanșeze o resetare.
3. **aux_reset_in:** Această intrare auxiliară de resetare este utilizată pentru control suplimentar de resetare. Poate fi folosit pentru a iniția o resetare de la o sursă auxiliară, cum ar fi un semnal software sau o altă componentă internă.
4. **mb_debug_sys_rst:** Această intrare este utilizată de obicei pentru resetarea sistemului de depanare MicroBlaze. Se asigură că procesorul MicroBlaze și logica aferentă acestuia sunt resetate corect în timpul depanării.

5. **dcm_locked**: Această intrare indică starea de blocare a Digital Clock Manager (DCM). Sistemul rămâne în resetare până când DCM este blocat, asigurându-se că toate semnalele de ceas sunt stabile înainte ca sistemul să înceapă să funcționeze.

6. **mb_reset:** Acest semnal de ieșire resetează procesorul MicroBlaze. Este afirmat ca răspuns la intrările de resetare și este sincronizat cu ceasul sistemului.

7. **bus_struct_reset[0:0]:** Acest semnal de ieșire resetează structurile magistralei din sistem. Acesta asigură că toată logica legată de magistrală este resetată și inițializată corect.

8. **peripheral_reset[0:0]:** Această ieșire resetează perifericele conectate la sistem. Acesta asigură că toate dispozitivele periferice pornesc dintr-o stare cunoscută după o resetare.

9. **interconnect_aresetn[0:0]:** Acest semnal de resetare activ-low este utilizat pentru interconexiunile AXI din cadrul sistemului. Acesta asigură resetarea și inițializarea corespunzătoare a interconexiunilor, care gestionează traficul de date între diferitele componente ale sistemului.

10. **peripheral_aresetn[0:0]:** Acest semnal de resetare activ-low este pentru periferice, similar cu periferic_reset, dar special conceput pentru periferice care necesită resetare activ-low.

Miezul IP de resetare a sistemului procesorului este extrem de configurabil, permițându-i să se adapteze la diferite cerințe de sistem prin activarea sau dezactivarea diferitelor funcții de resetare. Acesta asigură un proces de resetare fiabil și sincronizat pentru toate componentele unui design FPGA, care este esențial pentru menținerea stabilității și fiabilității sistemului.

2.4.3 AXI Interconnect

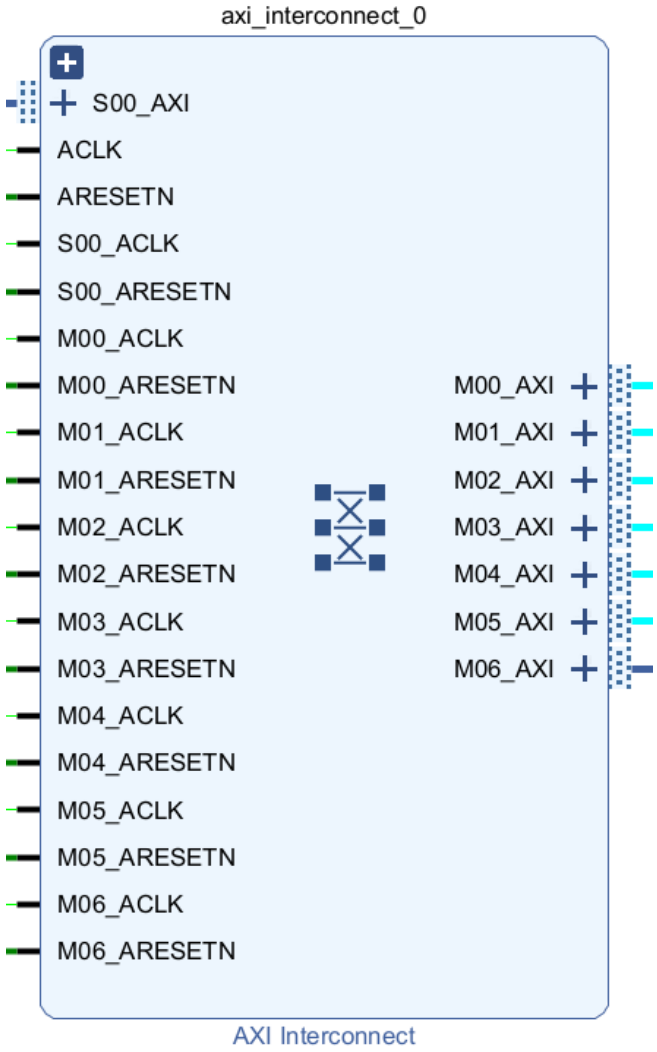

Figure 4: AXI Interconnect

Nucleul AXI Interconnect poate fi adăugat numai la un design de bloc integrator Vivado® IP în Vivado Design Suite. Nucleul Interconnect IP reprezintă un bloc de proiectare ierarhic care conține mai multe instanțe LogiCORE™ IP (nuclee de infrastructură) care devin configurate și conectate în timpul sesiunii de proiectare a sistemului. Fiecare dintre nucleele de infrastructură poate fi, de asemenea, adăugat direct la un design bloc (în afara nucleului AXI Interconnect) sau selectat direct din Catalogul Vivado IP și configurat pentru utilizare într-un design HDL. Nucleul AXI Interconnect permite orice amestec de master AXI și dispozitivele slave care urmează să fie conectate la acesta, care pot varia între ele în ceea ce privește lățimea datelor, domeniul de clock și sub-protocolul AXI (AXI4, AXI3 sau AXI4-Lite). Când caracteristicile interfeței oricărui dispozitiv master sau slave conectat diferă de cele ale comutatorului cu bară transversală din interiorul interconexiunii, nucleele infrastructurii adecvate sunt deduse automat și conectate în cadrul interconexiunii pentru a efectua conversiile necesare. Miezul AXI Interconnect IP

este un modul versatil care conectează mai multe dispozitive master și slave mapate cu memorie AXI. Iată o explicație detaliată a fiecărui pin din diagrama bloc furnizată:

1. **S00_AXI:** Aceasta este interfața slave AXI primară. Acesta permite unui master AXI să inițieze tranzacții de citire și scriere la interconectare.

2. **ACLK:** Acesta este semnalul de ceas global pentru interconectarea AXI. Sincronizează toate operațiunile din interconectare.

3. **ARESETN:** Acesta este semnalul global de resetare activ-low. Afirmarea acestui semnal resetează întreaga interconexiune AXI.

4. **S00_ACLK:** Acesta este semnalul de ceas specific interfeței slave S00_AXI. Sincronizează tranzacțiile pe această interfață specifică.

5. **S00_ARESETN:** Acesta este semnalul de resetare activ-scăzut pentru interfața slave S00_AXI. Inițializează această interfață specifică atunci când este afirmată.

6. **M00_ACLK la M06_ACLK:** Acestea sunt semnalele de ceas pentru fiecare interfață master respectivă (M00_AXI la M06_AXI). Ei sincronizează tranzacțiile pentru interfețele lor principale respective.

7. **M00_ARESETN la M06_ARESETN:** Acestea sunt semnalele de resetare activ-low pentru fiecare interfață master respectivă (M00_AXI la M06_AXI). Ele inițializează interfețele master respective atunci când sunt afirmate.

8. **M00_AXI la M06_AXI:** Acestea sunt interfețele master AXI care se conectează la dispozitivele slave AXI. Fiecare interfață gestionează tranzacțiile de citire și scriere către dispozitivele slave conectate.

Detalii suplimentare:

- **Interfețe master și slave:** Interconectarea AXI facilitează comunicarea între unul sau mai mulți master AXI și unul sau mai mulți slave AXI. Acesta asigură că datele sunt direcționate corect de la master la slave și invers.

- **Domenii de ceas:** Semnalele separate de ceas (ACLK) și de resetare (ARESETN) pentru fiecare interfață permit AXI Interconnect să gestioneze diferite domenii de ceas. Acest lucru este util în special în sistemele complexe în care diferite componente pot funcționa la frecvențe de ceas diferite.

- **Semnale de resetare:** existența unor semnale de resetare individuale (S00_ARESETN, M00_ARESETN etc.) pentru fiecare interfață permite un control fin asupra inițializării și resetării diferitelor părți ale interconexiunii.

Nucleul AXI Interconnect IP este foarte configurabil, permițându-i să fie adaptat nevoilor specifice ale unui design, inclusiv numărul de interfețe master și slave, lățimi de date și alți parametri.

2.4.4 AXI Traffic Generator

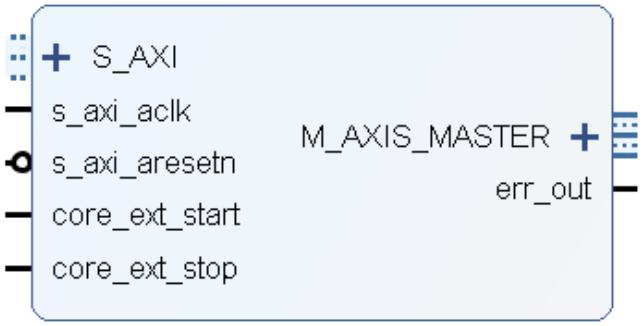

Figure 5: AXI Traffic Generator

Generatorul de trafic AXI AMD LogiCORE™ IP generează trafic prin interconectarea AXI4 și AXI4-Stream și alte periferice AXI4 din sistem. Acesta generează o mare varietate de tranzacții AXI bazate pe programarea de bază și modul de operare selectat.
Caracteristici:

- Interfață AXI4 pentru acces la registru și transfer de date.
- Operare multimod (AXI4 Master, AXI4-Lite Master și AXI4-Stream Master).
- Capacitate flexibilă de lățime a datelor (32/64 biți) la ieșire AXI4 Slave, (32 /64/128/256/512-bit) la ieșire Interfața AXI4 Master.
- Capacitate flexibilă de lățime a adresei de la 32 la 64 de biți pe interfața Master AXI4.
- Capacitate flexibilă de lățime a datelor de la 8 biți la 1.024 de biți în multipli de opt ieșiri Interfață AXI4-StreamMaster/Slave.
- Suportă interfața AXI4-Lite Master pentru inițializarea sistemului în sistemul fără procesor.
- Suport întrerupătoare pentru indicarea încheierii pentru generarea traficului.

- Pin de întrerupere de eroare care indică eroare apărută în timpul funcționării de bază. Registrele de eroare pot fi citite pentru a înțelege eroarea apărută. Acceptat numai în modul Avansat.
- Suportul de inițializare prin fișierele de inițializare a memoriei în memoria RAM internă (CMDRAM,PARAMRAM și MSTRAM) vă permite să inițializați conținutul tuturor RAM-urilor pentru un profil de trafic dorit.
- Pornire/oprire globală externă pentru a sincroniza mai mulți generatori de trafic AXI în sistemul și activarea AXI Traffic Generator fără intervenția procesorului.
- Acceptă generarea de trafic la nivel înalt pentru diferite profiluri de trafic.

AXI Traffic Generator IP de la Xilinx este un nucleu versatil conceput pentru a genera trafic AXI4 pentru a testa și a solicita diverse module și interconectări dintr-un sistem. O explicație detaliată a fiecărui pin din diagrama bloc furnizată:

1. **S_AXI Interfață:** Aceasta este interfața slave pentru tranzacțiile AXI4, permițând generatorului de trafic să primească comenzi și configurații.
2. **s_axi_aclk:** Acesta este semnalul de ceas pentru interfața S_AXI. Sincronizează transferul de date și semnalele de control între generatorul de trafic și sistemul conectat.
3. **s_axi_aresetn:** Acesta este semnalul de resetare activ-low pentru interfața S_AXI. Inițializează generatorul de trafic și asigură că începe de la o stare cunoscută atunci când este afirmată.
4. **core_ext_start:** Acest semnal de intrare este utilizat pentru a începe procesul de generare a traficului. Când este afirmat, declanșează generatorul de trafic să înceapă să trimită tranzacții AXI.
5. **core_ext_stop:** Acest semnal de intrare este folosit pentru a opri procesul de generare a traficului. Când este afirmată, oprește generatorul de trafic să trimită mai multe tranzacții.
6. **M_AXIS_MASTER Interfață:** Aceasta este interfața principală pentru tranzacțiile AXI4-Stream. Generatorul de trafic trimite date AXI4-Stream către această interfață, care poate fi conectată la alte periferice compatibile AXI4-Stream.
7. **err_out:** Acest semnal de ieșire indică o condiție de eroare în generatorul de trafic. Când este afirmat, înseamnă că a apărut o eroare în timpul procesului de generare a traficului, iar registrele de erori pot fi citite pentru a determina problema specifică.

Acești pini permit generatorului de trafic AXI să simuleze diferite scenarii de trafic, ajutând astfel la evaluarea performanței și la testarea de stres a sistemelor bazate pe AXI.

2.4.5 AXI Direct Memory Access (DMA)

Figure 6: AXI Direct Memory Access

Miezul Xilinx® LogiCORE™ IP AXI Direct Memory Access (AXI DMA) este un nucleu Xilinx IP moale pentru utilizare cu Xilinx Vivado® Design Suite. AXI DMA oferă acces direct la memorie de bandă mare între memorie și perifericele țintă AXI4-Stream. Capacitățile sale opționale de împrăștiere/adunare descarcă și sarcinile de mișcare a datelor din Unitatea Centrală de Procesare (CPU).

Nucleul IP AXI Direct Memory Access (AXI DMA) oferă acces direct la memorie de bandă mare între memoria mapată AXI4 și interfețele IP AXI4-Stream. Capacitățile sale opționale de colectare a dispersării descarcă sarcinile de mișcare a datelor de la Unitatea Centrală de Procesare (CPU) în sistemele bazate pe procesor. Registrele de inițializare, stare și gestionare sunt accesate printr-o interfață slave AXI4-Lite.

Mișcarea primară a datelor DMA de mare viteză între memoria sistemului și ținta fluxului se face prin intermediul AXI4 Read Master la AXI4 memorie mapată la stream Master (MM2S) și AXI stream la memorie mapată Slave (S2MM) la AXI4 Write Master. AXI DMA permite, de asemenea, până la 16 canale multiple de mișcare a datelor pe căile MM2S și S2MM în modul scatter/gather.

Canalul MM2S și canalul S2MM funcționează independent. AXI DMA oferă protecție la limitele adresei de 4 KB (atunci când este configurat în non-Micro DMA), mapare automată în rafală, precum și posibilitatea de a pune în așteptare mai multe cereri de transfer folosind aproape capacitatea de lățime de bandă completă a magistralelor AXI4-Stream. În plus, AXI DMA oferă realinierea datelor la nivel de octeți, permițând citirea și scrierea memoriei începând din orice locație de decalaj de octeți.

Canalul MM2S acceptă un flux de control AXI pentru trimiterea datelor aplicației utilizator către IP-ul țintă. Pentru canalul S2MM, este furnizat un flux de stare AXI pentru primirea

datelor aplicației utilizator de la IP-ul țintă.

Scatter/Gather preia și actualizează descriptori buffer din memoria sistemului prin interfața AXI4 Scatter Gather Read/Write Master.

Caracteristici:

- AXI4 compatibil.
- Suport opțional Independent Scatter/Gather Direct Memory Access (DMA).
  - Oferă descărcarea activității de gestionare a DMA de la CPU.
  - Oferă preluarea și actualizarea descriptorilor de transfer independent de magistrala de date primară.
  - Permite plasarea descriptorilor în orice locație mapată în memorie separată din bufferele de date. De exemplu, descriptorii pot fi plasați în blocul RAM.
  - Oferă funcționare ciclică opțională.
- Modul opțional de înregistrare directă (fără suport pentru împrăștiere/adunare) O performanță mai scăzută, dar mai puțin intensivă în resurse FPGA poate fi activată prin excluderea motorului Scatter Gather. În acest mod, transferurile sunt comandate prin setarea unei adrese sursă (pentru MM2S) sau a unei adrese de destinație (pentru S2MM) și apoi prin specificarea unui număr de octeți într-un registru de lungime.
- Suport primar pentru lățimea datelor AXI4-Stream de 8, 16, 32, 64, 128, 256, 512 și, 1.024 biți.
- Engine opțional de realiniere a datelor pentru o lățime a fluxului de date de până la 512 biți.

Nucleul IP AXI Direct Memory Access (DMA) facilitează transferurile de date cu lățime de bandă mare între memorie și perifericele AXI4-Stream. O explicație detaliată a fiecărui pin din diagrama bloc furnizată:

1. **Interfață S_AXI_LITE:**
   - **s_axi_lite_aclk:** Acesta este semnalul de ceas pentru interfața de control AXI-Lite, utilizat pentru configurarea și registrele de stare.
   - **S_AXI_LITE:** Această interfață este utilizată pentru controlul și configurarea nucleului AXI DMA. Acesta permite unui procesor să configureze operația DMA prin scrierea în registrele sale de control și să citească informațiile de stare.

2. **Interfață S_AXIS_S2MM:**

   - **s_axi_s2mm_aclk:** Acesta este semnalul de ceas pentru interfața AXI4-Stream to memory-mapped (S2MM). Sincronizează transferul de date de la sursa AXI4-Stream în memorie.
   - **S_AXIS_S2MM:** Această interfață acceptă date de la o sursă AXI4-Stream și le scrie într-o destinație mapată în memorie. Include semnale precum TREADY, TDATA, TKEEP, TLAST etc., care controlează transferul de date în flux.

3. **Interfață M_AXI_S2MM:**

   - **m_axi_s2mm_aclk:** Acesta este semnalul de ceas pentru interfața principală mapată cu memorie AXI4 utilizată în direcția S2MM.
   - **M_AXI_S2MM:** Această interfață principală scrie date într-o destinație mapată în memorie. Include semnale precum AWADDR (adresă de scriere), WDATA (date de scriere) și BREADY (răspuns de scriere).

4. **axi_resetn:** Acesta este semnalul de resetare activ-low pentru miezul AXI DMA. Inițializează nucleul și asigură că începe de la o stare cunoscută atunci când este afirmată.

5. **s2mm_prmry_reset_out_n:** Acest semnal de ieșire oferă un semnal de resetare activ-scăzut către perifericele din aval, indicând faptul că miezul AXI DMA a fost resetat.

6. **s2mm_introut:** Acest semnal de ieșire de întrerupere este afirmat pentru a indica faptul că un transfer de date s-a încheiat sau a apărut o eroare. Poate fi folosit pentru a notifica procesorului că trebuie să gestioneze transferul DMA sau starea de eroare.

7. **axi_dma_tstvec[31:0]:** Acesta este un vector pe 32 de biți utilizat în scopuri de testare și depanare. Acesta permite utilizatorului să testeze diverse funcții și căi în cadrul nucleului AXI DMA.

Nucleul AXI DMA poate funcționa în două moduri principale: modul DMA simplu și modul DMA Scatter-Gather. Acceptă canale multiple, realinierea datelor opționale și adresare pe 64 de biți, făcându-l versatil pentru o gamă largă de aplicații care necesită transfer eficient de date între memorie și periferice.

2.4.6 First-In, First-Out (FIFO)

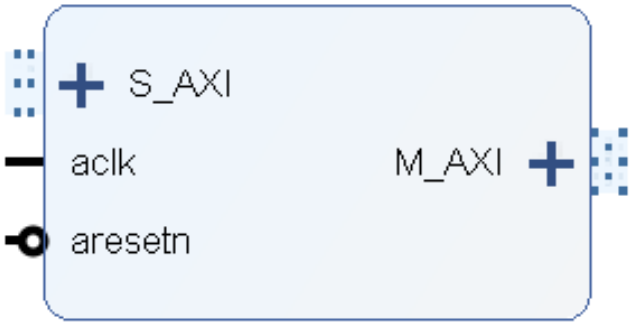

Figure 7: First-In, First-Out (FIFO)

Nucleul LogiCORE IP AXI4-Stream FIFO permite accesul mapat cu memorie la o interfață AXI4-Stream. Nucleul poate fi folosit pentru interfața cu IP-urile AXI4-Stream, similar cu nucleul FIFO AXI4-Stream, fără a fi nevoie să utilizați o soluție DMA completă. Funcționarea principală a acestui nucleu permite scrierea sau citirea pachetelor de date către sau de la un dispozitiv fără a se preocupa de semnalizarea interfeței AXI4-Stream. Puteți gestiona cu ușurință interfețele AXI4-Stream deoarece sunt transparente.

Miezul AXI Data FIFO IP este o componentă crucială concepută pentru stocarea în buffer a tranzacțiilor AXI pentru a îmbunătăți debitul și a gestiona fluxul de date între diferite domenii de ceas. O explicație detaliată a fiecărui pin din diagrama bloc furnizată:
Caracteristici:

1. **S_AXI Interfață:** Aceasta este interfața slave pentru tranzacțiile AXI4, permițând masterilor externi să scrie date în FIFO.

2. **aclk:** Acesta este semnalul de ceas principal pentru interfața AXI. Sincronizează toate tranzacțiile și operațiunile AXI4 în cadrul FIFO.

3. **aresetn:** Acesta este semnalul de resetare activ-scăzut pentru interfața AXI. Afirmarea acestui semnal inițializează FIFO și asigură că pornește într-o stare cunoscută. ”n” de la sfârșit înseamnă că resetarea este activă la nivel scăzut.

4. **Interfață M_AXI:** Aceasta este interfața principală pentru tranzacțiile AXI4, care permite FIFO să trimită date în buffer către slave AXI.

AXI Data FIFO IP este configurabil pentru a îndeplini cerințele specifice de proiectare, cum ar fi adâncimea FIFO, lățimea datelor și dacă să folosiți anumite canale AXI (cum ar fi canale de scriere de răspuns sau de citire a adresei).

2.4.7 AXI SmartConnect

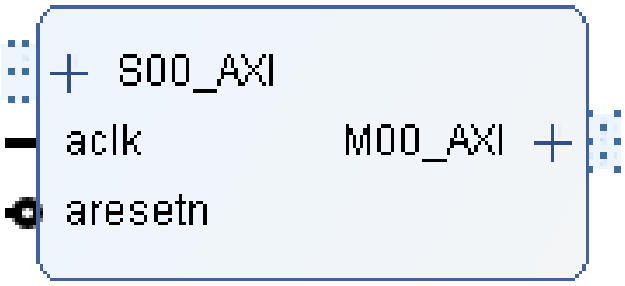

Figure 8: AXI SmartConnect

Nucleul Xilinx® LogiCORE™ IP AXI SmartConnect conectează unul sau mai multe dispozitive master mapate cu memorie AXI la unul sau mai multe dispozitive slave mapate cu memorie.

Nucleul AXI SmartConnect IP este o soluție de interconectare sofisticată pentru conectarea dispozitivelor master și slave mapate cu memorie AXI. O explicație detaliată a fiecărui pin din diagrama bloc furnizată:

1. **S00_AXI:** Aceasta este interfața slave AXI primară. Se conectează la un dispozitiv master AXI, permițându-i să inițieze tranzacții de citire și scriere prin SmartConnect.

2. **aclk:** Acesta este semnalul de ceas global pentru nucleul AXI SmartConnect. Sincronizează toate tranzacțiile și operațiunile din SmartConnect.

3. **aresetn:** Acesta este semnalul global de resetare activ-low. Când este afirmat (scăzut), resetează întregul nucleu AXI SmartConnect, asigurându-se că pornește dintr-o stare cunoscută.

4. **M00_AXI:** Aceasta este interfața principală AXI. Se conectează la un dispozitiv slave AXI, facilitând rutarea tranzacțiilor de la masterul AXI (conectat la S00_AXI) către acest slave.

2.4.8 General-Purpose Input/Output (AXI GPIO)

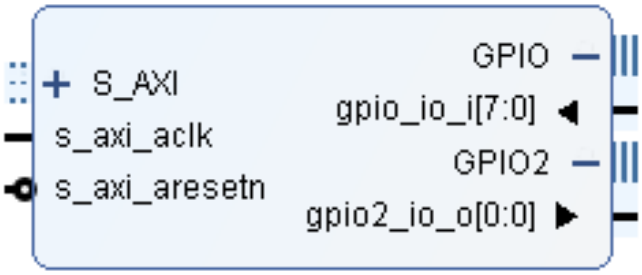

Figure 9: General-Purpose Input/Output (AXI GPIO)

Designul AXI GPIO oferă o interfață de intrare/ieșire de uz general către o interfață AXI4-Lite. AXI GPIO poate fi configurat fie ca dispozitiv cu un singur canal, fie ca dispozitiv cu două canale. Lățimea fiecărui canal este configurabilă independent. Porturile sunt configurate dinamic pentru intrare sau ieșire prin activarea sau dezactivarea bufferului cu 3 stări. Canalele pot fi configurate pentru a genera o întrerupere atunci când are loc o tranziție la oricare dintre intrările lor.

IP AXI GPIO (General Purpose Input/Output) de la Xilinx este o componentă versatilă utilizată pentru a interfața semnalele de intrare și ieșire de uz general cu o interfață AXI4-Lite. O explicație detaliată a fiecărui pin din diagrama bloc furnizată:

1. **S_AXI Interfață:** Aceasta este interfața slave pentru tranzacțiile AXI4-Lite. Acesta permite nucleului AXI GPIO să fie configurat și controlat de un master AXI4-Lite (de obicei un procesor).

   - **s_axi_aclk:** Acesta este semnalul de clock pentru interfața S_AXI. Sincronizează toate tranzacțiile și operațiunile AXI4-Lite din nucleul GPIO.
   - **s_axi_aresetn:** This is the active-low reset signal for the S_AXI interface. It initializes the GPIO core, ensuring it starts from a known state when asserted.

2. **Interfață GPIO:** Aceste interfețe sunt utilizate pentru semnale de intrare și ieșire de uz general. Nucleul AXI GPIO poate avea unul sau două canale GPIO, fiecare configurabil ca intrare, ieșire sau ambele.

   - **gpio_io_i[7:0]:** Acesta este un port de intrare pe 8 biți pentru primul canal GPIO. Permite citirea semnalelor externe de nucleul AXI GPIO.
   - **gpio2_io_o[0:0]:** This is a 1-bit output port for the second GPIO channel. It allows the AXI GPIO core to drive external signals.

2.4.9 AXI Timer

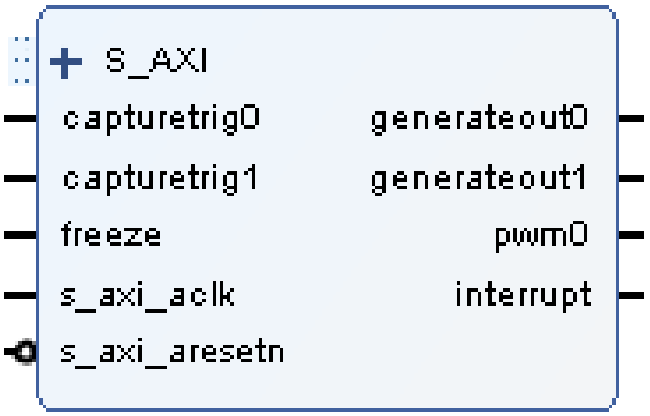

Figure 10: AXI Timer

Temporizatorul AXI este organizat ca două module de cronometru identice. Fiecare modul de cronometru are asociat un registru de încărcare care este utilizat pentru a păstra fie valoarea inițială a contorului pentru generarea de evenimente, fie o valoare de captare, în funcție de modul temporizatorului. Valoarea de generare, care este valoarea încărcată în registrul de încărcare, este utilizat pentru a genera o singură întrerupere la expirarea unui interval sau o serie continuă de întreruperi cu un interval programabil. Valoarea de captare este valoarea temporizatorului care a fost blocată la detectarea unui eveniment extern. Frecvența de ceas a modulelor de cronometru este s-axi-aclk (nu se efectuează o prescalare a ceasului). Toate întreruperile temporizatorului/contorului sunt combinate OR pentru a genera un singur semnal de întrerupere extern. Rutina de serviciu de întrerupere citește registrele de control/stare pentru a determina sursa întreruperii.

Nucleul AXI Timer IP oferă un modul de temporizator/contor flexibil cu o interfață AXI4-Lite, folosită în mod obișnuit în proiectele FPGA pentru diferite operațiuni de sincronizare. O explicație detaliată a fiecărui pin din diagrama bloc furnizată:

1. **Interfață S_AXI:**

   - **S_AXI:** Interfața slave AXI4-Lite utilizată pentru comunicarea cu procesorul gazdă. Permite configurarea, controlul și monitorizarea stării cronometrului.
   - **s_axi_aclk:** Semnalul de ceas pentru interfața AXI4-Lite. Sincronizează toate operațiunile și tranzacțiile pe interfața S_AXI.
   - **s_axi_aresetn:** Semnalul de resetare activ-scăzut pentru interfața AXI4-Lite. Afirmarea acestui semnal inițializează cronometrul, asigurându-se că pornește într-o stare cunoscută.

2. **Semnale de control și declanșare:**

   - **capturetrig0:** Un semnal de intrare utilizat pentru a declanșa funcția de captare a primului temporizator. Când acest semnal este afirmat, valoarea curentă a temporizatorului este capturată și stocată pentru o utilizare ulterioară.
   - **capturetrig1:** Similar cu capturetrig0, acest semnal de intrare declanșează funcția de captare a celui de-al doilea temporizator.

3. **Control operational:**

   - **freeze:** Un semnal de intrare folosit pentru a opri funcționarea temporizatorului. Când este afirmat, oprește procesul de numărare, care poate fi util în timpul depanării sau a unor stări operaționale specifice în care sincronizarea ar trebui întreruptă.

4. **Semnale de ieșire:**

   - **generateout0:** Un semnal de ieșire generat de primul temporizator. Poate fi folosit pentru a declanșa alte evenimente sau pentru a indica faptul că a trecut un anumit interval de timp.
   - **generateout1:** Similar cu generateout0, acest semnal de ieșire este generat de al doilea temporizator și servește același scop.

5. **Ieșire PWM (Pulse Width Modulation):**

   - **pwm0:** Un semnal de ieșire utilizat pentru modularea lățimii impulsurilor (PWM). Caracteristica PWM poate genera un semnal de impuls cu o frecvență și un ciclu de lucru specificate, util pentru controlul dispozitivelor precum motoare și LED-uri.

6. **Semnal de întrerupere:**

   - **întrerupere:** un semnal de ieșire care indică a avut loc un eveniment de cronometru. Poate fi conectat la un controler de întrerupere pentru a notifica procesorului evenimente legate de temporizator, cum ar fi expirarea temporizatorului sau evenimentele de captare.

În aplicațiile practice, AXI Timer este folosit pentru generarea de intervale precise de sincronizare, controlul dispozitivelor bazate pe PWM și măsurarea timpului dintre evenimente prin funcționalitatea sa de captare. Capacitatea sa de a gestiona mai multe moduri de operare și furnizarea de semnale de control esențiale îl fac o componentă robustă și flexibilă în sistemele bazate pe FPGA.

2.4.10 AXI Timebase Watchdog Timer

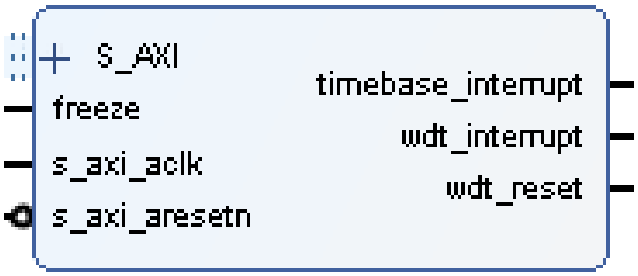

Figure 11: AXI Timebase Watchdog Timer

Xilinx® LogiCORE™ IP AXI4-Lite Timebase Watchdog Timer (WDT) este un periferic pe 32 de biți care oferă o bază de timp de 32 de biți cu funcționare liberă și un timer watchdog.

Nucleul IP AXI Watchdog Timer (WDT) este conceput pentru a ajuta la prevenirea blocărilor sistemului prin resetarea sistemului dacă software-ul nu funcționează corect. Iată o explicație detaliată a fiecărui pin din diagrama bloc furnizată:

1. **Interfață S_AXI:**
   - **S_AXI:** Aceasta este interfața slave AXI, care se conectează la un dispozitiv master AXI. Acesta permite maestrului să configureze timer-ul watchdog, să seteze timeout-uri și să gestioneze întreruperi.
   - **s_axi_aclk:** Acesta este semnalul de ceas pentru interfața AXI4-Lite. Sincronizează toate operațiunile și tranzacțiile pe interfața S_AXI.
   - **s_axi_aresetn:** Acesta este semnalul de resetare activ-low pentru interfața AXI4-Lite. Afirmarea acestui semnal inițializează temporizatorul watchdog, asigurându-se că pornește într-o stare cunoscută.
2. **Semnale de control:**
   - **freeze:** Acest semnal de intrare este folosit pentru a îngheța temporizatorul watchdog. Când este afirmată, oprește temporar numărătoarea inversă, ceea ce poate fi util în timpul depanării sau în modurile operaționale specifice în care watchdog-ul nu ar trebui să resetați sistemul.
3. **Întreruperi și resetare ieșiri:**
   - **timebase_interrupt:** Acest semnal de ieșire este o întrerupere generată de contorul de bază de timp. Contorul de bază de timp este un contor cu rulare liberă care poate genera întreruperi la intervale regulate, care poate fi folosit pentru sarcini periodice din sistem.

- **wdt_interrupt:** Acest semnal de ieșire este o întrerupere generată de temporizatorul watchdog. Semnalizează că temporizatorul watchdog a atins perioada de expirare și poate fi folosit pentru a alerta sistemul sau pentru a lua măsuri corective înainte de o resetare completă.
- **wdt_reset:** Acest semnal de ieșire este afirmat când timer-ul watchdog expiră și este necesară o resetare a sistemului. Este folosit pentru a reseta sistemul la o stare cunoscută, asigurând recuperarea dintr-o posibilă blocare.

Detalii suplimentare:

1. **Funcția Timer Watchdog:** Timer-ul Watchdog numără invers de la o valoare prestabilită și generează o resetare sau o întrerupere dacă ajunge la zero înainte de a fi repornit. Este de obicei folosit pentru a recupera de la erori software în cazul în care funcționarea regulată a sistemului este compromisă.
2. **Timebase Counter:** Acesta este un contor cu rulare liberă care poate genera întreruperi periodice, care sunt utile pentru cronometrarea evenimentelor sau verificări regulate ale sistemului.

Utilizare:

1. **Configurare:** AXI master configurează timer-ul watchdog scriind în registrele sale de control prin interfața S_AXI. Aceasta include setarea perioadei de expirare și activarea sau dezactivarea cronometrului.
2. **Funcționare:** temporizatorul watchdog trebuie resetat periodic de software pentru a preveni atingerea zero. Dacă software-ul nu reușește să resetați temporizatorul (de exemplu, din cauza unui blocaj), temporizatorul va genera o întrerupere sau va reseta sistemul.
3. **Funcția de freeze:** Semnalul de înghețare poate fi folosit pentru a opri temporar timer-ul watchdog, util în timpul depanării sau în anumite stări operaționale în care watchdog-ul nu ar trebui să resetați sistemul.

2.4.11 ILA (Integrated Logic Analyzer)

Figure 12: ILA (Integrated Logic Analyzer)

Miezul IP personalizabil Integrated Logic Analyzer (ILA) este un analizor logic care poate fi utilizat pentru a monitoriza semnalele interne ale unui design. Nucleul ILA include multe caracteristici avansate ale analizoarelor logice moderne, inclusiv ecuații booleene de declanșare și declanșatoare de tranziție de margine. Deoarece nucleul ILA este sincron cu designul monitorizat, toate constrângerile de ceas de proiectare care sunt aplicate designului dumneavoastră sunt aplicate și componentelor nucleului ILA.

Nucleul IP AXI Integrated Logic Analyzer (ILA) este un instrument puternic utilizat pentru depanarea și monitorizarea în timp real a semnalelor interne în modelele FPGA. O explicație detaliată a fiecărui pin din diagrama bloc furnizată:

1. **SLOT_0_AXI:** Acesta este slotul de interfață AXI principal utilizat pentru a conecta AXI ILA la o magistrală AXI4 sau AXI4-Lite. Acesta permite ILA să monitorizeze tranzacțiile AXI pe autobuz. Această interfață captează și analizează datele, adresa și semnalele de control ale magistralei AXI în scopuri de depanare.

2. **clk:** Aceasta este intrarea de ceas pentru nucleul ILA. ILA funcționează sincron cu semnalul de ceas furnizat aici. Este esențial să furnizați un semnal de ceas care să se potrivească cu ceasul operațional al semnalelor monitorizate pentru a asigura capturarea și analizarea exactă a datelor.

Nucleul AXI ILA captează și analizează semnalele interne și tranzacțiile în cadrul unui FPGA. Include funcții avansate, cum ar fi ecuații de declanșare booleene, declanșatoare de tranziție la margine și capabilități de stocare a datelor. Miezul ILA poate fi configurat cu diferite lățimi și adâncimi de sondă pentru a se potrivi cerințelor specifice de debugg-ing ale designului.

Utilizat în principal pentru verificarea pe cip, AXI ILA permite proiectanților să monitorizeze și să depaneze comportamentul semnalului intern în timp real. Această capacitate este deosebit de utilă pentru depanarea proiectelor complexe, deoarece captează semnale și condiții specifice, ajutând proiectanții să identifice și să depaneze problemele din FPGA. De asemenea, ajută la analiza performanței prin monitorizarea momentului și corectitudinii tranzițiilor interne ale semnalului și verifică funcționalitatea blocurilor de proiectare prin verificarea stărilor interne ale semnalului față de valorile așteptate în timpul funcționării.

Pentru a utiliza AXI ILA, de obicei începeți prin a instanția nucleul ILA în proiectarea dvs. utilizând catalogul Vivado IP. Apoi, conectați semnalele pe care doriți să le monitorizați la sondele ILA. Apoi configurați condițiile de declanșare și adâncimea de captare în Vivado, implementați și rulați designul pe FPGA și, în final, utilizați managerul hardware al Vivado pentru a vizualiza datele capturate și a depana designul.

2.5 Ce execută Software-ul (pe procesorul ARM)

Calculul Modelului C: Software-ul implementează modelul de calcul C pentru blocurile de construcție ale procesării streaming. Aceasta înseamnă executarea algoritmilor specifici aplicației care procesează datele de intrare într-o manieră secvențială sau bazată pe evenimente.

Suport pentru Întreruperi: Gestionarea întreruperilor hardware pentru a răspunde rapid la evenimente externe sau interne, cum ar fi finalizarea transferurilor DMA, semnale de la periferice sau timeri expirați.

Configurarea și Controlul DMA: Inițializarea și configurarea transferurilor DMA pentru a optimiza transferul de date între memoria sistemului și FPGA, reducând astfel sarcina procesorului și întârzierile.

Configurarea și controlul cronometrului AXI Timebase Watchdog: Inițializați și configurați Timebase Watchdog AXI pentru a monitoriza în mod continuu sistemul, prevenind blocările și detectând stările de eroare. Permite resetarea automată a sistemului în cazul unui timeout, asigurând o funcționare robustă și fiabilă.

Configurarea și controlul cronometrului AXI: Configurați și gestionați cronometrul AXI pentru a oferi funcționalități precise de sincronizare necesare în aplicațiile în timp real. Configurați-l pentru a genera întreruperi la intervale de timp predefinite, asigurând sincronizarea corectă a operațiunilor și optimizând fluxul de lucru al sistemului.

Configurarea și controlul AXI Traffic Generator: Configurați AXI Traffic Generator pentru a testa și valida performanța magistralei AXI prin generarea de trafic de date controlat. Acest lucru ajută la evaluarea capacității de transfer și a latenței magistralei, contribuind la optimizarea arhitecturii sistemului.

Configurarea și controlul AXI GPIO0, AXI GPIO1, AXI GPIO2: Configurați și gestionați IP-urile AXI GPIO pentru a oferi interfețe flexibile de intrare/ieșire digitale. Configurați pinii pentru diferite funcții de control și monitorizare, permițând interacțiunea cu dispozitivele externe și facilitând integrarea în aplicații complexe.

2.6 Ce execută FPGA

Funcționalități Cheie: Implementează funcționalități specifice aplicației, cum ar fi hold, soft reset, timestamping și watchdog, care sunt esențiale pentru gestionarea stării sistemului și a datelor.

Logica de Integrare: Include FIFO de intrare și ieșire pentru bufferizarea datelor, logica calculată de software pentru prelucrarea datelor de ieșire.

Procesarea Datelor Streaming: Realizează prelucrarea de bază și filtrarea datelor de intrare în timp real, utilizând capacitățile de procesare paralelă ale FPGA-ului. Acest lucru este crucial pentru aplicațiile care necesită prelucrarea rapidă a volumelor mari de date, cum ar fi procesarea semnalului sau a imaginii.

Comunicarea prin AXI Stream: Generează transferul de date în timp real în cadrul sistemului, folosind interfețe AXI Stream pentru a conecta diferite blocuri de procesare și memorie în FPGA.

Cerințele de cost și throughput pentru un sistem de procesare de tip streaming bazat pe platforma AMD ZYNQ 7000 depind foarte mult de specificul aplicației și de design-ul sistemului. Totuși, putem discuta în termeni generali despre cerințele pentru modurile de operare diferite, cum ar fi **single frame, multiple frame** și **multiple frame cu repeat,** privind capacitatea sistemului de a procesa un anumit număr de biți pe secundă.

Single Frame Mode În modul single frame, sistemul procesează un singur cadru de date la un moment dat. Throughput-ul necesar va depinde de mărimea cadrelor și de rata la

care acestea trebuie procesate. De exemplu, dacă un cadru este de 1 MB (megabyte) și cerința este de a procesa 30 de cadre pe secundă (fps), throughput-ul necesar va fi:

Throughput = Marimea Cadru × Rate Cadre = 1 MB × 30 = 30 MB/s

Conversia în biți: 30 MB/s × 8 = 240 Mbps

Multiple Frame Mode În modul multiple frame, sistemul procesează mai multe cadre simultan. Acest lucru necesită un throughput semnificativ mai mare, deoarece capacitatea sistemului trebuie să suporte sumarea ratelor de procesare pentru fiecare cadru individual. Dacă sistemul ar trebui să proceseze, de exemplu, 3 fluxuri diferite la 30 fps fiecare, cu aceeași mărime a cadrelor de 1 MB, throughput-ul necesar ar fi triplat: Throughput = 3 × 240 Mbps = 720 Mbps

Multiple Frame Mode cu Repeat Modul multiple frame cu repeat presupune reprocesarea aceluiași set de cadre multiple ori. Dacă fiecare cadru trebuie repetat de două ori înainte de procesarea următorului set, atunci throughput-ul necesar se dublează față de cel calculat pentru multiple frame mode: Throughput = 720 Mbps × 2 = 1440 Mbps

2.7 Descrierea detaliată a sistemului și integrarea nucleelor AXI IP pentru procesarea datelor ZedBoard

Această teză prezintă un design detaliat și o integrare a mai multor nuclee IP AXI pentru manipularea datelor și controlul sistemului pe un ZedBoard cu un sistem de procesare (PS) Zynq-7000. Sistemul încorporează un generator de trafic AXI, AXI DMA, AXI Data FIFO, AXI SmartConnect și alte periferice AXI pentru a gestiona fluxul de date, a gestiona resetările sistemului și pentru a oferi feedback în timp real prin LED-uri. Această configurație demonstrează un mecanism robust de transfer de date de la un generator de trafic AXI la Zynq-7000 PS și include mecanisme pentru funcții de resetare și watchdog, precum și feedback interactiv prin componente hardware de pe ZedBoard.

Arhitectura sistemului

Arhitectura sistemului este împărțită în mai multe subsisteme, fiecare responsabil pentru funcționalități specifice, interconectate pentru a forma un sistem coeziv pentru generarea, procesarea și controlul datelor.

1. **Subsistemul de generare și transfer de date.**
   Componente implicate:

   - **Generator de trafic AXI**

- **AXI DMA**
- **AXI Data FIFO**
- **AXI SmartConnect**
- **Zynq-7000 PS**

AXI Traffic Generator generează trafic de date sintetice, care sunt trimise către AXI DMA pentru operațiuni de acces direct la memorie. AXI DMA facilitează transferurile de date de mare viteză de la generatorul de trafic la AXI Data FIFO, care memorează temporar datele. Datele stocate în tampon sunt apoi direcționate prin AXI SmartConnect către sistemul de procesare (PS) Zynq-7000 pentru procesare și stocare ulterioară. Acest lanț asigură generarea și transferul eficient de date de la generatorul de trafic la Zynq-7000 PS, făcându-l potrivit pentru aplicații de înaltă performanță.

2. **Resetarea subsistemul de control.**
   Componente implicate:

   - **Buton de pe ZedBoard**
   - **axi_gpio_0**
   - **Generator de trafic AXI**

   Un buton de pe ZedBoard este conectat la axi_gpio_0, care monitorizează starea butonului. Când butonul este apăsat, acesta declanșează un semnal de resetare prin intermediul GPIO la pinul de resetare al generatorului de trafic AXI. Acest lucru permite generatorului de trafic să repornească și să trimită date din nou către Zynq-7000 PS.

3. **Subsistem de afișare LED bazat pe timp.**
   Componente implicate:

   - **AXI Timer**
   - **axi_gpio_1**
   - **LED-uri pe ZedBoard**

   Pinul generateout0 al temporizatorului AXI emite un semnal la fiecare secundă, care este trimis către axi_gpio_1. GPIO conduce apoi LED-urile de pe ZedBoard, actualizându-le pentru a afișa timpul scurs în formă binară, arătând numărul de secunde trecute.

4. **Subsistemul de management al întreruperii.**
   Componente implicate:

   - **AXI Timer**

- **Zynq-7000 PS IRQ_F2P[1:0]**

Pinul de întrerupere a temporizatorului AXI este conectat la pinii IRQ_F2P[1:0] de pe Zynq-7000 PS. Această conexiune permite temporizatorului să genereze întreruperi, informând Zynq-7000 PS despre evenimentele de sincronizare, care pot fi folosite pentru gestionarea sarcinilor critice de timp în sistem.

5. **Subsistemul de resetare a temporizatorului Watchdog.**
   Componente implicate:

   - **AXI Timebase Watchdog Timer**
   - **Generator de trafic AXI**

   Pinul wdt_reset al temporizatorului Watchdog AXI Timebase este conectat la pinul de resetare al generatorului de trafic AXI. La fiecare 10 secunde, watchdog-ul trimite un semnal de resetare la generatorul de trafic, asigurându-se că repornește și continuă să trimită date periodic.

6. **Watchdog Timer Interrupt Management.**
   Componente implicate:

   - **AXI Timebase Watchdog Timer**
   - **Zynq-7000 PS IRQ_F2P[1:0]**

   Pinul wdt_interrupt al temporizatorului Watchdog AXI Timebase este conectat la pinii IRQ_F2P[1:0] de pe Zynq-7000 PS. Această conexiune permite watchdog-ului să genereze întreruperi la PS, indicând când timer-ul watchdog-ului a expirat, permițând sistemului să ia măsurile corespunzătoare.

7. **Watchdog Timer Freeze Control.**
   Componente implicate:

   - **Comutator ZedBoard**
   - **axi_gpio_2**
   - **AXI Timebase Watchdog Timer**

   Un comutator de pe ZedBoard este conectat la axi_gpio_2, care monitorizează starea comutatorului. Acest GPIO controlează pinul de înghețare al temporizatorului Watchdog AXI Timebase. Când comutatorul este comutat, întrerupe sau reia temporizatorul watchdog, oferind control manual asupra funcționalității watchdog.

Integrarea acestor nuclee AXI IP demonstrează un sistem complex, dar eficient pentru manipularea, controlul și monitorizarea datelor pe ZedBoard cu un Zynq-7000 PS. Această configurație prezintă aplicarea practică a perifericelor bazate pe AXI în gestionarea fluxului de date, gestionarea resetărilor, furnizarea de feedback în timp real și menținerea fiabilității sistemului prin mecanisme de supraveghere. Această arhitectură detaliată nu numai că exemplifică capacitățile nucleelor AXI IP, dar servește și ca o bază solidă pentru îmbunătățiri și aplicații viitoare în sistemele bazate pe FPGA.

2.8 Fluxul de lucru al proiectului in Vivado

Crearea unui proiect în Vivado implică câțiva pași detaliați pentru a asigura un flux de lucru fără probleme, de la configurarea proiectului până la generarea și exportul fluxului de biți. Mai jos este o listă cuprinzătoare a pașilor necesari privind utilizarea Vivado. Acesta listă este menită să ofere o abordare detaliată și structurată a gestionării unui proiect Vivado, care include crearea unui proiect, validarea designului, generarea fluxului de biți și exportul platformei.

1. **Crearea unui proiect nou.**
   - Lansați Vivado și creați un nou proiect.
   - Setați numele, locația și tipul proiectului (de exemplu, Proiect Licenta).
   - Specificați sursele (fișiere RTL) și constrângeri (fișiere XDC).
   - Alegeți dispozitivul sau placa FPGA țintă.
2. **Adăugați și configurați surse.**
   - Adăugați fișiere RTL și constrângeri suplimentare după cum este necesar.
   - Definiți modulul superior al designului dvs.
   - Adăugați nuclee IP din Catalogul IP dacă este necesar.
3. **Design și proiectare bloc.**
   - Creați și configurați un design de bloc folosind IP Integrator.
   - Conectați blocurile IP și validați designul blocului.
4. **Synthesis.**
   - Rulați sinteza pentru a compila RTL într-o listă de net la nivel de poartă.
   - Examinați raportul de sinteză pentru utilizarea resurselor și a timing-ului.
5. **Implementation.**

- Rulați implementarea pentru a plasa și ruta proiectul.
- Examinați rapoartele de implementare și rezolvați orice problemă de sincronizare.

6. **Bitstream Generation.**

   - Generați fluxul de biți pentru configurația FPGA.
   - Asigurați-vă că generarea fluxului de biți se finalizează fără erori.

7. **Export Platform.**

   - Exportați designul hardware, inclusiv fluxul de biți și fișierul de definiție hardware (HDF/XSA), pentru dezvoltarea software.

8. **Software Development.**

   - Utilizați SDK sau Vitis pentru a crea un proiect software.
   - Importați hardware-ul și dezvoltați software pentru proiectare.

9. **Testare si Validare.**

   - Implementați fluxul de biți în FPGA.
   - Testați și validați sistemul complet pe hardware.

Fiecare pas din acest proces implică sarcini detaliate care asigură crearea, validarea și implementarea cu succes a proiectului in Vivado. Proiectul începe cu crearea unui nou proiect, adăugarea de surse, crearea de design-uri de bloc și continuă prin simulare, sinteză și implementare. Validarea și generarea fluxului de biți asigură că designul este pregătit pentru implementarea hardware, iar pașii finali implică exportul hardware-ului și integrarea software-ului.

2.9 Fluxul de lucru al proiectului in Vivado

Pașii pentru începerea unui nou proiect în Vitis folosind fișierul .xsa generat de Vivado, inclusiv crearea unei platforme, construirea unei aplicații și implementarea acesteia pe un ZedBoard:

Pași pentru a începe un nou proiect în Vitis.

1. **Instalați Vitis și Dependențe.**

   - Asigurați-vă că sunt instalate Vitis si Vivado.

2. **Generați fișierul .XSA în Vivado.**

- Generare completă a fluxului de biți în Vivado.
- Export hardware: Clic pe File - Export - Export Hardware.
- Asigurați-vă că includeți fluxul de biți este bifat. Salvați fișierul .xsa într-o locație cunoscută.

Pași pentru a crea o platformă în Vitis.

3. **Creați un nou proiect de platformă Vitis.**

- Deschideți Vitis.
- Creare proiect platforma: Clic pe File - New - Platform Project.
- Setați numele și locația proiectului.
- Alegeți Creare din hardware (XSA) și navigați la fișierul .xsa exportat din Vivado.
- Setați un nume pentru platformă și selectați ZedBoard.
- Clic Next -¿ Finish.
- Faceți clic dreapta pe proiectul platformei și selectați Build.
- Asigurați-vă că platforma se build-uie cu succes.

Pași pentru a crea o aplicație utilizând platforma.

4. **Creați un nou proiect de aplicație.**

- Deschideți Vitis.
- Creare Application Project: Clic pe File - New - Application Project.
- Introduceți un nume pentru proiectul aplicației.
- Alegeți platforma creată în pașii anteriori.
- Clic Next.
- Alegeți configurația sistemului cu un singur procesor.
- Selectați un template (de exemplu, Hello World, Empty Application).
- Clic Finish.

Acești pași oferă o cale clară de la generarea fișierului .xsa în Vivado, crearea unei platforme în Vitis, dezvoltarea unei aplicații și implementarea acesteia pe ZedBoard. Asigurați-vă că fiecare pas este urmat cu precizie pentru a eficientiza fluxul de lucru al proiectului și pentru a rula cu succes aplicații pe hardware-ul dumneavoastră FPGA.

2.10 Documentatia .h si .c a componentelor IP, platformei si main-ul proiectului

Documentația pentru fișierele de inițializare a platformei

- **Fișier header:** platform_config.h, platform.h
- **Fișier sursă:** platform.c

Aceste fișiere sunt responsabile pentru inițializarea și curățarea platformei, gestionarea configurațiilor cache-ului și configurarea UART pentru comunicare.

Descrierea macro-urilor și a funcțiilor folosite

1. **Macros**

   platform_config.h

   STDOUT_IS_PS7_UART: Indică faptul că ieșirea standard este direcționată către PS7 UART.

   UART_DEVICE_ID: Definește ID-ul dispozitivului pentru UART.

   Valoare: 0

2. **Funcțiile**

   - **void enable_caches:** Activează memoria cache pentru instrucțiuni și date pe baza arhitecturii procesorului țintă.
   - **void disable_caches:** Dezactivează memoria cache de instrucțiuni și date pe baza arhitecturii procesorului țintă.
   - **void init_uart:** Inițializează UART pe baza configurației din **platform_config.h**.
   - **void init_platform:** Inițializează platforma activând cache-urile și instalând UART.
   - **void cleanup_platform:** Curăță platforma dezactivând memoria cache.

Documentație pentru AXI Traffic Generator IP

- **Fișier header:** axi_traffic_gen.h
- **Fișier sursă:** axi_traffic_gen.c

Descrierea macro-urilor și a funcțiilor folosite

1. **Macros**

   AXI_TRAFFIC_GEN_BASEADDR:

   Aceast macro definește adresa de bază pentru IP-ul AXI Traffic Generator. Este folosit ca adresă de pornire pentru toate interacțiunile cu registrul.

   Valoare: 0x43C00000 Adresa de bază unde IP-ul AXI Traffic Generator este mapat în memorie.

2. **Funcțiile**

   - **void AXI_TrafficGen_Start:** Această funcție pornește AXI Traffic Generator.
   - **void AXI_TrafficGen_Stop:** Această funcție oprește generatorul de trafic AXI.
   - **void AXI_TrafficGen_SetPattern:** Această funcție setează modelul de trafic pentru AXI Traffic Generator.
   - **void AXI_TrafficGen_Reset:** Această funcție resetează AXI Traffic Generator.

Documentație pentru AXI DMA IP

- **Fișier header:** axi_dma.h
- **Fișier sursă:** axi_dma.c

Descrierea macro-urilor și a funcțiilor folosite

1. **Macros**

   Fișierele nu definesc în mod explicit niciun macro. Toate configurațiile și definițiile necesare sunt gestionate prin apeluri de funcție și constante **xparameters.h.**

2. **Funcțiile**

   - **int AXI_DMA_Init:** Inițializează hardware-ul AXI DMA.
   - **int AXI_DMA_S2MM_Transfer:** Inițiază un transfer Stream-to-Memory-Mapped (S2MM) folosind AXI DMA.
   - **int AXI_DMA_MM2S_Transfer:** Inițiază un transfer Memory-Mapped-to-Stream (MM2S) folosind AXI DMA.

Documentație pentru AXI Timebase Watchdog Timer (WDT) IP.

- **Fișier header:** axi_timebase_wdt.h

- **Fișier sursă:** axi_timebase_wdt.c

Descrierea macro-urilor și a funcțiilor folosite

1. **Macros**

   AXI_TIMEBASE_WDT_DEVICE_ID:

   Aceast macro definește ID-ul dispozitivului pentru IP-ul AXI Timebase Watchdog Timer (WDT).

   Valoare: XPAR_AXI_TIMEBASE_WDT_0_DEVICE_ID ID-ul dispozitivului IP WDT definit în fișierul **xparameters.h (0U)**.

2. **Funcțiile**

   - **int AXI_Timebase_WDT_Init:** Inițializează hardware-ul AXI Timebase WDT.
   - **void AXI_Timebase_WDT_Start:** Pornește AXI Timebase WDT.
   - **void AXI_Timebase_WDT_Stop:** Oprește AXI Timebase WDT.

Documentație pentru AXI Timer IP.

- **Fișier header:** axi_timer.h
- **Fișier sursă:** axi_timer.c

Descrierea macro-urilor și a funcțiilor folosite

1. **Macros**

   AXI_TIMER_DEVICE_ID:

   Definește ID-ul dispozitivului pentru IP-ul AXI Timer.

   Valoare: XPAR_AXI_TIMER_0_DEVICE_ID ID-ul dispozitivului al IP-ului temporizatorului AXI definit în fișierul **xparameters.h (0U)**.

   TIMER_INTERRUPT_ID:

   Definește ID-ul de întrerupere pentru IP-ul temporizatorului AXI.

   Valoare: XPAR_FABRIC_AXI_TIMER_0_INTERRUPT_INTR ID de întrerupere definit în fișierul **xparameters.h (62U)**.

2. **Funcțiile**

   - **int AXI_Timer_Init:** Inițializează cronometrul AXI.
   - **void AXI_Timer_Start:** Pornește temporizatorul AXI.

- **void AXI_Timer_Stop:** Oprește temporizatorul AXI.
- **void AXI_Timer_Intr_Handle:** Gestionează întreruperea temporizatorului.

Documentație pentru AXI GPIO IP (axi_gpio_0).

- **Fișier header:** axi_gpio.h
- **Fișier sursă:** axi_gpio.c

Descrierea macro-urilor și a funcțiilor folosite

1. **Macros**

   AXI_GPIO_DEVICE_ID:

   Definește ID-ul dispozitivului pentru IP-ul AXI GPIO (axi_gpio_0).

   Valoare: XPAR_AXI_GPIO_0_DEVICE_ID ID-ul dispozitivului IP-ului AXI GPIO (axi_gpio_0) definit în **xparameters.h (0)**.

2. **Funcțiile**

   - **int AXI_GPIO_Init:** Inițializează AXI GPIO (axi_gpio_0).
   - **u32 AXI_GPIO_Read:** Citește date de la AXI GPIO (axi_gpio_0).

Documentație pentru AXI GPIO IP (axi_gpio_1).

- **Fișier header:** axi_gpio_1.h
- **Fișier sursă:** axi_gpio_1.c

Descrierea macro-urilor și a funcțiilor folosite

1. **Macros**

   AXI_GPIO_1_DEVICE_ID:

   Definește ID-ul dispozitivului pentru IP-ul AXI GPIO (axi_gpio_1).

   Valoare: XPAR_AXI_GPIO_1_DEVICE_ID ID-ul dispozitivului IP-ului AXI GPIO (axi_gpio_1) definit în **xparameters.h (1)**.

2. **Funcțiile**

   - **int AXI_GPIO_1_Init:** Inițializează AXI GPIO (axi_gpio_1).

- **void AXI_GPIO_1_Write:** Scrie date pe AXI GPIO (axi_gpio_1).

Documentație pentru AXI GPIO IP (axi_gpio_2).

- **Fișier header:** axi_gpio_2.h
- **Fișier sursă:** axi_gpio_2.c

Descrierea macro-urilor și a funcțiilor folosite

1. **Macros**

   AXI_GPIO_2_DEVICE_ID:
   Definește ID-ul dispozitivului pentru IP-ul AXI GPIO (axi_gpio_2).
   Valoare: XPAR_AXI_GPIO_2_DEVICE_ID ID-ul dispozitivului IP-ului AXI GPIO (axi_gpio_2) definit în **xparameters.h (2)**.

2. **Funcțiile**

   - **int AXI_GPIO_2_Init:** Inițializează AXI GPIO (axi_gpio_2).
   - **u32 AXI_GPIO_2_Read:** Scrie date pe AXI GPIO (axi_gpio_2).

Documentație pentru main.c.

Fișierul main.c servește drept cod primar de control și demonstrație pentru interfața cu diverse periferice AXI, inclusiv generator de trafic, DMA, GPIO, timer și watchdog de pe ZedBoard. Inițializează aceste periferice, setează întreruperi și demonstrează transferul de date folosind aceste nuclee IP.

- **Fișier sursă:** main.c

Descrierea macro-urilor și a funcțiilor folosite

1. **Macros**

   BUFFER_SIZE:
   Dimensiunea buffer-ului utilizat pentru transferul de date.
   Valoare: 1024.

   PATTERN:
   Model de trafic pentru generatorul de trafic AXI.
   Valoare: 0xA5A5A5A5.

DMA_BUFFER_BASE:
Adresă de bază pentru tamponul DMA din memorie.
Valoare: XPAR_PS7_DDR_0_S_AXI_BASEADDR + 0x01000000 (0x00100000 + 0x01000000).

WDT_INTERRUPT_ID:
Valoare: XPAR_FABRIC_AXI_TIMEBASE_WDT_0_WDT_INTERRUPT_INTR (61U).

2. **Funcțiile**

- **void start_traffic_gen :** Configurați și pornește AXI Traffic Generator.
- **void transfer_data :** Transferă date folosind DMA de la generatorul de trafic într-un buffer.
- **void wait_for_button_press :** Așteaptă apăsarea unui buton pentru a reporni generatorul de trafic.
- **void Timer_Intr_Handler :** Gestionează întreruperile temporizatorului.
- **void WDT_Intr_Handler :** Se ocupă de întreruperile timer-ului watchdog.
- **int SetupInterruptSystem :** Configurați sistemul de întrerupere pentru temporizator și watchdog.
- **int main :** Funcția principală inițializează perifericele, setează întreruperi și rulează bucla principală a aplicației.

Această documentație acoperă codul de control primar din main.c pentru inițializarea și gestionarea perifericelor AXI pe ZedBoard. Fișierul integrează generarea de trafic AXI, DMA, GPIO, timer și timer watchdog pentru a demonstra transferul de date, a gestiona întreruperile și a interacționa cu hardware-ul prin apăsarea butoanelor și LED-urilor.

2.11 Considerații de Cost

Costul Hardware: Capacitatea de throughput mai mare necesită hardware mai performant, care poate crește costul sistemului. FPGA-urile mai avansate și cu o capacitate mai mare de procesare sunt, de obicei, mai scumpe.

Costul Dezvoltării: Implementarea logicii necesare pentru a suporta throughput-uri mari poate fi mai complexă și necesită mai mult timp și resurse, ceea ce, de asemenea, influențează

costul total al proiectului.

În contextul unui sistem bazat pe AMD ZYNQ 7000 destinat validării unui sistem de procesare de tip streaming cu interfață AXI stream, programabilitatea prin software se referă la capacitatea de a configura, controla și modifica funcționarea hardware-ului și a logicii sistemului fără a fi necesare modificări fizice ale hardware-ului. Aceasta include o gamă largă de funcționalități, de la configurarea parametrilor de sistem până la actualizarea algoritmilor de procesare.

3 CONCLUZIE

Acest proiect a integrat cu succes mai multe periferice AXI pe ZedBoard folosind Xilinx Vivado și Vitis, culminând cu un sistem funcțional bazat pe FPGA. Începând cu descrierea hardware în Vivado, am creat o platformă robustă în Vitis, care a facilitat dezvoltarea unei aplicații software care folosește AXI Traffic Generator, DMA, GPIO, Timer și Timebase Watchdog Timer.

AXI Traffic Generator a furnizat un flux de date pentru testare, în timp ce AXI DMA a gestionat eficient transferurile de date. AXI GPIO-urile au permis interacțiunile utilizatorului, iar temporizatorul AXI a gestionat operațiuni precise de sincronizare. Timebase Watchdog Timer a adăugat fiabilitate prin monitorizarea sănătății sistemului și resetarea generatorului de trafic după cum este necesar.

Gestionarea întreruperilor a fost esențială pentru îmbunătățirea capacității de răspuns a sistemului, gestionanții pentru Timer și Watchdog Timer gestionând în mod eficient evenimentele critice. Integrarea hardware-ului și software-ului a fost exemplificată în aplicația Vitis, demonstrând designul FPGA practic și controlul sistemului în timp real.

Acest proiect a evidențiat importanța unei abordări coordonate hardware-software și utilitatea instrumentelor Xilinx în gestionarea proiectelor complexe FPGA. Perspectivele obținute oferă o bază solidă pentru explorarea ulterioară în dezvoltarea FPGA, deschizând calea pentru procesarea avansată a datelor, optimizarea sistemului și funcționalitatea extinsă în aplicațiile viitoare.

4 BIBLIOGRAFIE