

Praktikum Elektronische Systeme

Hochschule Osnabrück
Fakultät Ingenieurwissenschaften und Informatik
Labor für Digital- und Mikroprozessortechnik
Prof. Dr.-Ing. W. Gehrke

WS2425

Milestone 3: Audio-Filter mit AXI-Lite Schnittstelle

Lernziele

- AXI-Lite Schnittstelle
- Realisierung einfacher Schnittstellen
- Wiederholung Vivado-Projekte, VHDL und Synthese
- Arbeiten mit IP-Integrator

Aufgabenstellung

Das Audio-Filter-IP aus der Vorlesung soll um eine AXI-Lite Schnittstelle erweitert werden. So können die Koeffizienten zur Laufzeit geändert werden.

Das neue IP soll die folgenden AXIL-Register besitzen:

Adressoffset	Bezeichnung	Bitbelegung
0x00	Steuerregister	[31:1] – <i>ungenutzt</i> [0] – Run ; 0: IP inaktiv, 1: IP aktiv
0x04	Koeffizienten C0, C1, C2 und Divisor-Exponent E	[31:27] – <i>ungenutzt</i> [26:24] – Exponent E des Divisors ¹ (unsigned). [23:16] – C2 (signed) [15:8] – C1 (signed) [7:0] – C0 (signed)

¹ Ausgangswerte werden um E Bit rechtsverschoben, was einer Division durch 2^E entspricht.

Durchführung

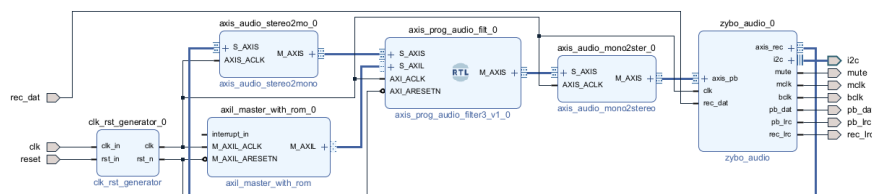
Filter IP

1. Erstellen Sie ein neues Vivado-Projekt mit dem Projektnamen *milestone3*. Fügen Sie die vorgegebene VHDL-Datei *axis_prog_audio_filter3.vhd* und die vorgegebene Constraints-Datei *milestone3.xdc* dem Projekt hinzu.
2. Die VHDL-Vorlage enthält nur die Entity. Zur Vereinfachung der Arbeiten können Sie die Architecture aus dem Audio-Filter-IP der Vorlesung aus dem Support-Ordner entnehmen. *Achten Sie darauf, dass in diesem Fall der Entity-Name angepasst werden muss.*
3. Ergänzen Sie den VHDL-Code um eine AXI-Lite Schnittstelle entsprechend der oben angegebenen Tabelle.

Die Filterkoeffizienten, der Divisor-Exponent und der Parameter *Run* sollen auch mit Hilfe der IP-Konfiguration wählbar sein. Daher sind entsprechende Generics in der VHDL-Entity vorgesehen. Übernehmen Sie während eines Resets des IPs die Werte der Generics in die AXIL-Register. Verwenden Sie für beide Schnittstellen (AXIS und AXIL) **ein** gemeinsames Resetsignal (AXI_ARESETN) und **ein** gemeinsames Taktsignal (AXI_ACLK).

Simulation

1. Fügen Sie das IP-Verzeichnis „IP“ (aus den Vorlesungsdaten) dem Projekt hinzu.
2. Erstellen Sie ein Block-Design *design_1* und fügen Sie den *axis_prog_audio_filter3* als RTL-Block hinzu. Bauen Sie eine Simulationsumgebung entsprechend des nachfolgenden Diagramms auf. (Tipp: im Anhang finden Sie eine größere Darstellung des Block-Designs)



3. In der Simulation und auf dem ZyBo-Board findet das IP „axil_master_with_rom“ Verwendung. Dieses IP kann AXIL-Master-Zugriffe durchführen, die in einem IP-internen Speicher abgelegt sind. Ein Script mit exemplarischen Master-Zugriffen und die zugehörige Binärdatei (stimuli.mem) finden Sie im Praktikumsordner. Kopieren Sie die .mem-Datei in den Projektordner (auf gleicher Ebene wie die .xpr-Datei).
4. Überprüfen Sie vor der Synthese den Pfad zur *mem*-Datei in der Konfiguration des AXIL-Master-IPs. Beispiel: Sie verwenden die Datei stimuli.mem und diese liegt (wie oben empfohlen) im Projektverzeichnis. Dann müsste folgendes in der Konfiguration des IPs eingetragen werden: `.././stimuli.mem`

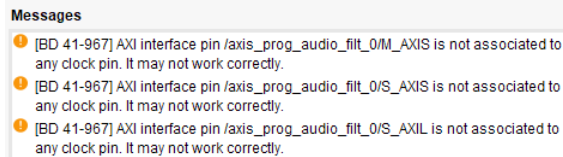
Erläuterung:

Die **Synthese** findet in einem Unter-Unter-Verzeichnis statt. Damit die .mem-Datei gefunden werden kann, muss der **Synthese** mitgeteilt werden, dass die Datei zwei Verzeichnisebenen weiter oben liegt. Dies geschieht mit `../..`.

Die **Simulation** findet sogar vier Verzeichnisebenen tiefer statt. Für die **Simulation** müsste man also „`.././.././stimuli.mem`“ eintragen. Da damit der Wechsel zwischen Simulation und Synthese anstrengend wird, ist das AXIL-Master-IP so realisiert, dass es bei der Simulation eigenständig zwei aufsteigende Verzeichnisebenen zu der angegebenen Datei hinzufügt. Falls Sie jetzt verwirrt sind, fragen Sie nach – die Sache ist einfacher als sie klingen mag.

5. Erstellen Sie den HDL-Wrapper. Kontrollieren Sie, ob der Wrapper unter *Simulation Sources* **und** unter *Design Sources* als Top-Level ausgewählt ist. Ist das nicht der Fall, korrigieren Sie dies.
6. Überprüfen Sie mithilfe der Simulation, ob die Schreibzugriffe korrekt ausgeführt werden und die von Ihnen erwarteten Daten von Ihrem IP übernommen werden. *Hinweis: Eine Simulation mit Audiodateien/-ausgabe ist mit dieser Testbench nicht möglich.*

Hinweis: Die nachfolgend dargestellte Warnmeldung kann ignoriert werden.



Synthese

1. Markieren Sie die AXIS- und AXIL-Anschlüsse Ihres IPs als Debug-Objekte für den ILA. Lassen Sie Vivado die „Connection Automation“ durchführen und ergänzen Sie anschließend manuell eventuell noch fehlende Verbindungen zum ILA-IP.
2. Konfigurieren Sie die Speichertiefe („Sample Data Depth“) des ILA mit 8192. So können Sie etwa vier aufeinanderfolgende Audiosamples mit dem ILA untersuchen.
3. Erzeugen Sie die Bitstream-Datei und laden Sie sie auf das ZyBo-Board.
4. Überprüfen Sie die AXIL-Übertragungen. Mögliches Vorgehen: ILA mit Triggerbedingung „scharf schalten“ und anschließend einen Reset (Btn0) auslösen.
5. Überprüfen Sie die Funktion des Filter-IPs, indem Sie die berechneten Ausgangswerte mit den Werten vergleichen, die sich rechnerisch aus den Eingangswerten ergeben. Verwenden Sie für diese Überprüfung die Werte, die Sie mit dem ILA aufgezeichnet haben.
6. Führen Sie eine akustische Überprüfung des Filter-IPs durch Anschließen einer Audioquelle und eines Headsets bzw. Lautsprechers durch.

ACHTUNG: Timing Problematiken

Schauen Sie nach der Synthese in der *Summary* unter dem Punkt „Timing“ nach, ob in Ihrem Design Timing-Verletzungen auftreten. Dies ist wahrscheinlich, wenn Sie alle Berechnungen in einem Taktzyklus ausführen. Timing-Probleme treten auf, wenn die Zeit zu lang ist, die ein Signal benötigt, um von einem Register zum nächsten Register zu gelangen. Versuchen Sie, den Signalweg zu verkürzen, indem Sie die Berechnung innerhalb des IPs in mehrere Schritte unterteilen, zum Beispiel in die Schritte: Multiplikationen, Addition der Produkte und Formatierung des Ergebnisses (Schiebeoperation). Wenn Sie diese Schritte mithilfe eines endlichen Automaten realisieren, kann es sinnvoll sein, dass Sie neben den drei Berechnungszuständen einen IDLE-Zustand (warten auf Eingangsdaten) und einen Zustand, der auf die Übernahme der Daten am Ausgang wartet, hinzufügen. Diese Lösung benötigt mehrere Taktzyklen je Audiosample. Das ist aber nicht dramatisch – wir haben ja 2083 Taktzyklen Zeit.

Wir können Sie hierbei mit einer Musterlösung unterstützen. Bevor Sie diese anfragen, machen Sie eigene Gehversuche. Vielleicht finden Sie auch allein eine Lösung...

Abgabe

Um den Speicherbedarf von Vivado-Projekten zu reduzieren, können Sie alle Verzeichnisse und Dateien mit Ausnahme der folgenden löschen: Verzeichnis `<projectname>.srcs`, und Projektdatei `<projectname>.xpr`

Für die Abgabe dieses Milestones archivieren Sie die beiden bereinigten Projektverzeichnisse in einer ZIP-Datei. Fügen Sie Screenshots der Simulation sowie des FPGA-Ressourcenverbrauchs hinzu.

Anhang

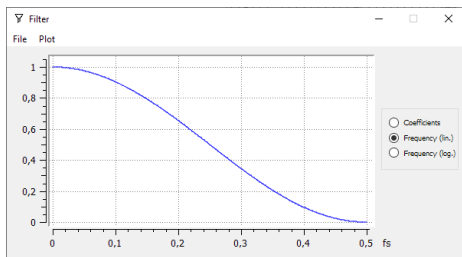
Koeffizienten

Die Koeffizienten liegen in dezimaler Form dar. Um Integer-Werte verwenden zu können, wurden die Koeffizienten mit einer Multiplikation von 64 ($= 2^6$) berechnet. Entsprechend muss das Ergebnis des Filters, wie im Video beschrieben, durch diesen Wert dividiert werden (Divisor-Exponent = 6).

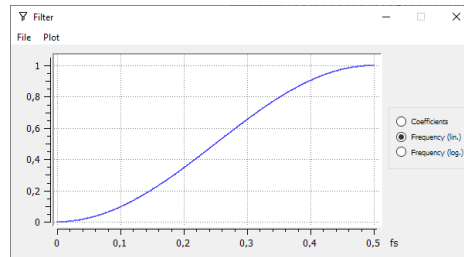
	C0	C1	C2
Tiefpass	16	32	16
Hochpass	-16	32	-16
Bandpass	32	0	-32
Bandsperr	32	0	32
Unverändert	0	64	0

Die Frequenzgänge der Filter sind in den nachfolgenden Diagrammen über Vielfache der Abtastfrequenz f_s im Bereich 0 bis $0,5 \cdot f_s$ aufgetragen.

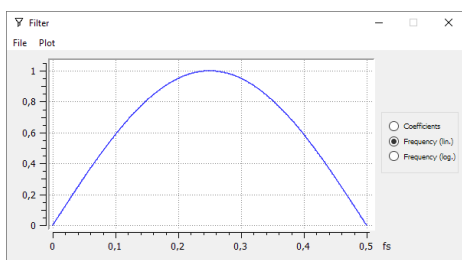
Beachten Sie die lineare (= nicht logarithmische) Darstellung.



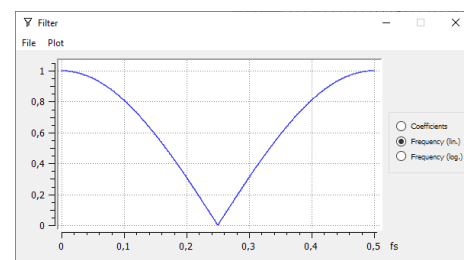
Frequenzgang Tiefpass



Frequenzgang Hochpass



Frequenzgang Bandpass



Frequenzgang Bandsperr

Klinkenstecker

Die Unterschiede zwischen einem 4-poligen (links) und einem 3-poligen (rechts) Klinkenstecker zeigen die nachfolgenden Bilder. Beide Steckertypen sind kompatibel mit dem 3-poligen Headphone-Anschluss des ZyBo-Boards. Sollten bei Ihnen wider Erwarten Probleme auftreten, sprechen Sie uns an.



Blockschaltbild des Systems (bereits mit ILA)