Microblaze

Schritt 1:

Starten Sie Vivado und wählen Sie „New Project“ aus. Hierbei wählt man „RTL Project“ als Projekttyp, wobei der Haken bei „ Do not specify sources at this time“ gesetzt sein sollte.

//Hier Schritt1

Schritt 2:

Nun gilt es unter dem Tab „Boards“ das korrekte Board auszuwählen, in meinem Fall war es das Nexys4 DDR.

//Hier Schritt2

Anschließend klicken Sie auf „Finish“ und das Projekt wird erstellt.

Schritt3:

Im Flow Navigator auf der linken Seite wählt man nun „Create Block Design“ aus und gibt diesem Design einen beliebigen Namen.

Es öffnet sich das Block Design, bei dem der Tab „Board“ zu wählen ist. Hier werden nun alle Bausteine für das Nexys4 DDR-Board angezeigt.

//Hier Schritt 3

Schritt 4:

Durch doppeltes Klicken auf die Bausteine werden zunächst die „System Clock“, „USB UART“ und „DDR2 SDRAM“ hinzugefügt. Die folgenden Fenster werden jeweils mit OK bestätigt.

Im Design Diagram kann nun über das „+“-Symbol der Microblaze und der „AXI Timer“ eingefügt werden.

Schritt 5 :

Der „Clocking Wizard“ muss wie im folgenden Bild konfiguriert werden.

// Hier Schritt5

Dabei wird eine zweite Output Clock auf 200MHz gesetzt, sowie der Reset Type auf „Active Low“

Schritt 6:

Die bestehende Verbindung zwischen der Sys\_Clock und dem MIG muss gelöscht werden und eine neue zwischen dem Clocking Wizard und dem MIG erstellt werden.

//Hier Schritt6

Schritt 7:

Ab hier übernimmt zuerst die „Block Automation“, bei der die im nachfolgenden Bild angezeigten Einstellungen zu übernehmen sind. Besonders wichtig ist hier die Auswahl des Interrupt Controller.

//Hier Schritt 7

Im Anschluss wird die „Connection Automation“ gestartet und alle Bausteine ausgewählt.

Durch die Schaltfläche „Regenerate Layout“ kann das Diagram neu angeordnet werden.

Schritt 8:

Der angelegte Interrupt Controller ist mit einem Concat verbunden. Auf die Eingänge des Concats legt man nun die Interruptleitung des AXI Uartlite und des AXI Timers. Die Verbindungen sind im folgenden Bild orange gekennzeichnet.

//Hier Schritt 8

Schritt 9 :

Der Microblaze ist wie folgt durch einen Doppelklick zu konfigurieren.

//Hier Schritt 9

Die neuen Einstellungen werden durch das Klicken des OK-Buttons übernommen.

Schritt 10:

Der AXI Uartlite-Baustein muss ebenfalls angepasst werden. Hierbei gilt es im Tab „IP Configuration“ die Baudrate auf 115200 anzupassen. Diese Änderung ist mit OK zu bestätigen.

//Hier Schritt 10

Schritt 11:

Durch klicken der rechten Maustaste auf das Design im Source-Tab wird nun der HDL Wrapper erstellt.

Schritt 12 :

Im nächsten Schritt wird das Bitfile generiert. Dafür ist im Flow Manager der Button „Generate Bitstream“ vorgesehen. Die „Number of jobs“ erhöht dabei die Geschwindigkeit je nach Prozessor und verfügbaren Kernen. Im Anschluss wird über File , Export , Export Hardware das generierte Bitstream für die SDK, welche als nächstes gestartet wird, nutzbar gemacht.

Schritt 13:

Um den dazugehörigen Device Tree zu generieren, wird das Device-Tree Repository benötigt, welches aus dem Xilinx Github runtergeladen werden kann: <https://github.com/Xilinx/device-tree-xlnx.git>

Nach dem Entpacken wird der Ordner nun im SDK als Repository eingebunden. Dies geschieht über die Schaltfläche „Xilinx Tools“ beziehungsweise Repositories. Hier kann nun der Pfad des Ordners nur für ein bestimmtes Projekt oder für alle Projekte hinterlegt werden.

Schritt 14:

Das Erstellen des Device-Tree geschieht über „File“ und anschließend „New“ und „Board Support Package“. Dabei sollte nochmal die voreingestellten Parameter überprüft werden und als Board Support Package OS device\_tree ausgewählt werden.

//Hier Schritt 14

Das folgenden Fenster kann ohne Weiteres mit OK bestätigt werden.

Schritt 15:

Die entstandenen .dtsi beziehungsweise .dts-Dateien müssen gemerged werden und in einer .dts-Datei gespeichert werden. Hierzu sind beide Dateien im Texteditor zu öffnen, um die .dts

in die .dtsi-Datei zu kopieren. Die resultierende Datei wird, wie bereits erwähnt, als alleinstehende .dts-Datei gespeichert, um später in das buildroot-Tool eingefügt zu werden.

//Hier Schritt 15

**Erstellen des Linux Image**

Schritt 16:

Das bereits genannte Buildroot kann unter <https://buildroot.org/download.html> heruntergeladen werden.

Das in meiner Arbeit verwendete Buildtool besitzt die Version 2018.02 .

Neben dem Buildroot werden Linux Kernel defonfig-Dateien benötigt, welche von mir für das Nexys4 angepasst wurden und auf folgenden Github zur Verfügung stehen :

<https://github.com/HenW/microblaze/tree/master>

Die Dateien werden in das Verzeichnis /board/nexys/microblaze kopiert, welche vorher neu angelegt werden müssen. Es ist wichtig zu beachten, dass die Basisadresse des Kernels als 0x80000000 gesetzt ist und gegebenenfalls angepasst werden muss.

//Hier Schritt16

Die im vorherigen Schritt erstellte .dts-Datei muss ebenfalls in das erstellte Verzeichnis eingefügt werden.

Die Buildroot defconfig-Datei „nexys\_microblaze\_defconfig“ muss im „configs“ Ordner abgelegt werden.

Schritt 17:

Nachdem das System, wie in der vorherigen Schritt beschrieben, vorbereitet wurde, muss nun folgender Befehl mit Hilfe eines Terminals im Buildroot-Verzeichnis ausgeführt werden:

**make nexys\_microblaze\_defconfig**

Dieser Befehl erzeugt eine sogenannte .config Datei für das zugehörige buildroot.

Schritt 18:

Um den Kernel und das passende Filesystem zu erstellen, wird der Befehl

**make**

benötigt. Dieser Vorgang benötigt je nach Rechenleistung einige Minuten.

Nach erfolgreichem Abschluss des Vorgangs wird eine „simpleImage.artix7“-Datei erstellt, welche zum booten des Linux auf dem Nexys4 DDR Artix7 FPGA Modul benötigt wird.

**Ausführen von Linux auf dem Nexys4 DDR Artix7**

Das Linux Image muss jetzt in einen Ordner kopiert werden, welcher im Anschluss über die SDK Shell leicht zu erreichen ist. Das FPGA-Board wird nun mit dem Rechner über USB verbunden und eingeschaltet. Über das Xilinx SDK kann nun eine Verbindung über den korrekten Port mit der entsprechenden Baudrate (115200) hergestellt werden.

Schritt 19:

Um die Verbindung zu prüfen kann, nachdem das FPGA-Board programmiert wurde ein Hello World Programm ausgeführt werden. Andernfalls kann direkt im SDK über Xilinx Tools, beziehungsweise „Launch Shell“ ein Terminal geöffnet werden.

Mit dem Befehl:

**XMD**

wird eine Xilinx Microprocessor Debugger(XMD) Engine gestartet.

Anschließend wird über

**connect mb mdm**

eine Verbindung zum Microblaze Target(mb) als Microprocessor Debug Module Target(mdm) hergestellt.

Nun gilt es das Kernel Image auf das FPGA-Modul zu laden. Hierzu muss innerhalb des Shells in den korrekten Ordner navigiert werden.(Siehe Schritt 18)

Über die bestehende Verbindung kann nun mit Hilfe von

**dow simpleImage.artix7**

das Image heruntergeladen werden.