Philipp Holzinger
Thomas Schlögl

1 Daten- und Instruktionsspeicher

In dieser Übung soll der Datenspeicher sowie der Instruktionsspeicher erstellt werden. Da eine DRAM Ansteuerung für diese Übung zu aufwendig ist, werden werden sie mittels zwei separaten BRAMs (= Block SRAM) umgesetzt. Diese werden von Vivado durch eine passende VHDL Beschreibung *inferiert*. BRAM besitzt bei Xilinx Zynq Boards das folgende Verhalten:

- Array von std_logic_vector (*shared variable*)
- Maximal zwei verschiedene Adressbusse (Ports)
- Zugriffe auf eine Adresse können im selben Takt lesen und schreiben
- Ausschließlich taktsynchrones lesen
- Ausschließlich taktsynchrones schreiben
- Lässt sich nicht durch einen Reset zurücksetzen
- Können beim Laden des FPGA Images ein Mal initialisiert werden
- Besitzt je ein *Port enable*, um Lesen und Schreiben zu aktivieren
- Besitzt je ein *Port write enable*, um das Schreiben zu aktivieren

Beide Speicher sollen 256 32-Bit-Werte speichern können. Ein Testprogramm kann mit Hilfe des *MARS Simulators* geschrieben und die Instruktionen Hex-kodiert vom Simulator ausgeben werden. Anschließend kann das Programm via Array-Initialisierung fest in den Instruktionsspeicher der CPU geschrieben werden.

2 Aufgaben

- Erstellen sie die Datei *data_mem.vhd*, implementieren Sie den Datenspeicher und testen Sie ihn auf korrektes Verhalten.
- Synthetisieren Sie ihr Design und stellen Sie sicher, dass BRAM inferiert wird!
- Erstellen Sie die Datei *instr_mem.vhd* und implementieren Sie den Instruktionsspeicher.
- Laden Sie sich den MARS Simulator von Folgernder Webseite herunter:

  courses.missouristate.edu/kenvollmar/mars/

- Starten Sie den Simulator und schreiben Sie ein Assemblerprogramm welches die ersten Fibonacci Zahlen berechnet und an I/O Ports schreibt mit denen später die 8 LEDs verbunden werden. Gehen Sie davon aus, dass sich dieses Register an der Adresse *0x2000* befindet. Achten Sie auch darauf, dass die Zahlen auf der realen Hardware lange genug sichtbar sind! Benutzen Sie dabei nur MIPS Befehle, die von Ihrem Design unterstützt werden (AND, OR, ADD, LUI, XOR, SUB, SLT, NOR, ADDI, BEQ, LW, SW)! Testen Sie anschließend Ihr Programm im Simulator.
- Lassen Sie sich die Hex-kodierten Instruktionen vom Simulator ausgeben und initialisieren Sie damit Ihren Instruktionsspeicher.

Philipp Holzinger
Thomas Schlögl

- Fügen Sie den Programmcounter zu Ihrer Testbench hinzu und verbinden Sie ihn mit dem Instruktionsspeicher. Achten Sie dabei darauf, dass der Befehlszähler Byteadressen generiert, während der BRAM mit Wortadressen angesteuert wird!
- Synthetisieren und Implementieren Sie ihr Design. Überprüfen sie, ob diesmal BRAM inferiert wird oder nicht.