VHDL – Aufgabe 2

SD Karte

Aufgabenbeschreibung

Im zweiten Teil des VHDL Praktikums soll die SPI Schnittstelle aus der ersten Aufgabe eingesetzt werden um Daten von einer SD-Karte zu lesen.
Über Adressleitungen wird ein 512 Byte großer Block ausgewählt, von der SD Karte gelesen und in einem lokalen Speicherblock (M4k) des Cyclone II zwischengespeichert. Anschließend können die Daten aus diesem Speicherblock byteweise in beliebiger Reihenfolge und Häufigkeit gelesen werden (RAM).

SD Karte

Das im Praktikum verwendete DE2 Board von Altera stellt die komplette physikalische Schnittstelle zur SD Karte inklusive eines entsprechenden Slots zur Verfügung.
Das Protokoll der SD Karte sieht neben dem Standardmodus für hohe Geschwindigkeit den deutlich einfacheren SPI Modus vor. Dieser muss erst durch senden einer speziellen Befehlsfolge aktiviert werden. Anschließend können die Daten Blockweise (512 Byte) gelesen und geschrieben werden.
Eine einfache Einführung ins Protokoll ist unter folgender Adresse zu finden:
http://elm-chan.org/docs/mmc/mmc_e.html
Die offizielle Spezifikation befindet sich unter:
http://www.sdcard.org/developers/tech/sdcard/pls/Simplified_Physical_Layer_Spec.pdf
Hier ist im Wesentlichen Kapitel 7 (SPI Mode) interessant.

M4K Blöcke

Der Cyclone II FPGA unseres Testboards stellt 105 M4K Blöcke zur Verfügung. Hierbei handelt es sich um dedizierte Speicherblöcke mit einer Kappazität von je 4096 Bit. Am Einfachsten lassen sich diese durch Generierung eines Speicherblocks mit dem „Quartus Mega Wizzard“ verwenden (zum Beispiel als RAM).

Schnittstellenbeschreibung

In dieser Aufgabe ist keine bindende Vorgabe für die Schnittstelle gegeben. Die Schnittstelle ist so zu wählen, dass die Komponente im Rahmen des geforderten Funktionsumfangs möglichst einfach und universell eingesetzt werden kann.
Es folgt ein unverbindliches und unvollständiges Beispiel (es fehlen unter Anderem die SPI Leitungen):

```
-- Adresse des von der Speicherkarte zu lesenden Speicherblocks
blockAddr : in std_logic_vector(22 downto 0);
-- Startet den Lesevorgang von der Speicherkarte
readBlock : in std_logic;
-- Signalisiert, dass gerade von der Speicherkarte gelesen wird
busy : out std_logic;

-- Adresse des Bytes aus dem aktuell zwischengespeicherten Block
byteAddr : in std_logic_vector(8 downto 0);
-- Das mit byteAddr ausgewählte Byte
data : out std_logic_vector(7..0);
```

Test

Neben dem Ausführlichen Test mit Testbenches soll das Design bei dieser Aufgabe auch in die Hardware übertragen und dort mit einer echten SD Karte getestet werden.
Es ist eine entsprechende „Wrapper Entity“ zu entwerfen um die grundlegende Funktionalität zum Beispiel mit Schaltern und Tastern als Eingabe und LEDs zur Ausgabe zu testen. Die SD Karten können am PC mit dem Programm HxD im Rohmodus beschrieben werden. Das Programm befindet sich auf dem Transferlaufwerk. Alternativ kann es von folgender Seite geladen werden:
http://mh-nexus.de

Hinweise

Die Aufgabenstellung ist bewusst offen gestellt und fordert Eigeninitiative bei der Informationsbeschaffung und Ausarbeitung der Umsetzungsdetails. Für die erwartungsgemäß auftretenden Fragen stehe ich natürlich gerne zur Verfügung. Auch der Austausch von Ideen und Ansätzen unter den Praktikumsteilnehmen ist erwünscht.

Ablauf

Die Aufgabe soll in folgender Reihenfolge bearbeitet werden:

1. **Planungsphase:**
   Macht Euch anhand eines/einiger Blockdiagramme Gedanken zum Aufbau des Designs. Verwendet für die strukturierte Darstellung des Designs das Top-Down-Verfahren.

2. **Implementierungsphase:**
   Nach der Planungsphase implementiert Ihr das Design in VHDL. Hierfür steht in CAE2 die Entwicklungsumgebung Quartus II zur Verfügung. Achtet bitte darauf, dass Ihr möglichst früh anfangt einzelne Teile des Designs mit Modelsim zu testen!
3. **Simulationsphase:**
   Erstellt als Erstes eine Liste mit sinnvollen Testfällen um die Funktion des Designs möglichst vollständig zu prüfen. Es kann übrigens sinnvoll sein diese Liste bereits vor der Implementierungsphase zu erstellen um sicherzustellen, dass der Test nicht an die vorhandenen Fehler angepasst wird und um sich vorab die genaue gewünschte Funktion des Designs klar zu machen.
   Schreibt nun eine oder mehrere Testbench(-es), um das Design mit den erstellten Testfällen zu prüfen. Achtet darauf den Test möglichst übersichtlich zu gestalten. Hierbei hilft die Verwendung von Prozeduren, Funktionen und Schleifen. Für die automatische Auswertung der Tests lassen sich asserts verwenden.
   Das Design wird zunächst funktional und erst dann als post-synthese Simulation mit Timing getestet.
4. **Vorführung:**
   Das Design wird in das FPGA-Board übertragen und anschliessend auf Funktion getestet. Erfüllt das Design alle Erwartungen, lasst Euch die Aufgabe vom zuständigen Assistenten testieren und die nächste Aufgabe aushändigen.
5. **Dokumentation:**
   Die Dokumentation der Aufgabe ist **wichtiger** Bestandteil bei der Vergabe der Punkte für dieses Praktikum. Hierzu gehört neben einer schriftlichen Ausarbeitung in gedruckter Form eine CD mit allen selbst erstellten und für die Inbetriebnahme nötigen Dateien, sowie der Ausarbeitung in digitaler Form.
   In der Ausarbeitung müssen sich mindestens die folgende Punkte befinden:
   1. Planung:
      1. Problemanalyse
      2. Blockschaltbild(er)
      3. Beschreibung der Schnittstellen
   2. Implementierung:
      1. Erklärung wesentlicher Design-Teile
      2. Besonders wichtige Codeabsschnitte
      3. Zustandsdiagramme der Automaten
   3. Simulation:
      1. Beschreibung der Testfälle
      2. Beschreibung der Testbenches
      3. Testergebnisse (Timing-Diagramme, ...)
      4. Auswertung / Interpretation der Testergebnisse

Abgabe

Diese Aufgabe muss beim zuständigen Assistenten bis zum 30.06.09 erfolgreich vorgeführt und die Dokumentation abgegeben sein.