

Übung zur Vorlesung Einführung in Computer Microsystems

Prof. Dr. A. Koch
Thorsten Wink



TECHNISCHE
UNIVERSITÄT
DARMSTADT

Sommersemester 11
Übungsblatt 5

Aufgabe 5.1 Adressierung

Implementieren Sie als Verilog-Modul einen Adressdekoder für einen Bus mit einer CPU als Initiator/Master. Der Dekoder soll mit möglichst wenigen Gattern auskommen und für jeden der folgenden Slave-Teilnehmer ein separates SELECT-Signal erzeugen:

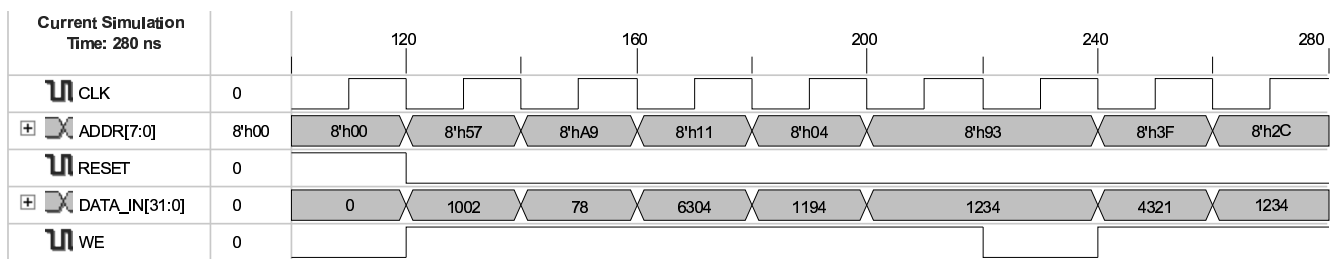
- RAM 64KB
- ROM 8KB
- ROM 8KB
- ROM 4KB
- 8 Register zu je 16 Bit
- 4 Register zu je 32 Bit

Jede Adresse adressiert ein Byte. Finden Sie eine geeignete Address-Map, Memory-Aliasing ist erlaubt. Es darf höchstens ein SELECT-Signal gleichzeitig aktiv sein. Prüfen Sie mit einer geeigneten Testbench, ob alle Adressen korrekt dekodiert werden.

Aufgabe 5.2 Transaktionen

Gegeben sind eine Waveform und ein Verilog-Modul.

Waveform:



Übung zur Vorlesung Einführung in Computer Microsystems

Verilog-Modul:

```
'timescale 1ns / 1ps
module memory_mapped_regs(CLK, RESET, ADDR, DATA_IN, WE, DATA_OUT);
  input CLK;
  input RESET;
  input [7:0] ADDR;
  input [31:0] DATA_IN;
  input WE;
  output [31:0] DATA_OUT;

  reg [31:0] A, B, C, D, E, F, G;

  assign DATA_OUT = ADDR[6] ? D : ADDR[5:4] == 2'b00 ? A :
                      ADDR[5:4] == 2'b01 ? B : ADDR == 42 ? E :
                      ADDR[7] ? C : ADDR[3:2] == 2 ? G : F;

  always @(posedge CLK or posedge RESET) begin
    if (RESET) begin
      A <= 1; B <= 2; C <= 3; D <= 4;
      E <= 5; F <= 6; G <= 7;
    end
    else
      case ({WE, ADDR[7], ADDR[4:2]})
        5'h17: A <= DATA_IN;
        5'h1F: B <= DATA_IN;
        5'h13: C <= DATA_IN;
        5'h14: D <= DATA_IN;
        5'h1A: E <= DATA_IN;
        5'h15: F <= DATA_IN;
        5'h1C: G <= DATA_IN;
        default: begin
          G <= 42;
          A <= DATA_IN;
        end
      endcase
    end
  end
endmodule
```

Die in der Waveform dargestellten Transaktionen werden als Stimulus in das Verilog-Modul eingegeben. Welche Werte enthalten danach die Register A bis G und auf welchen Adressen können sie jeweils ausgelesen werden? Hinweis: Das Modul enthält einige Designfehler beim Address-Decoding...

Diese Hausaufgaben müssen bis 1.7.11, 18:00 über das Moodle-System abgegeben werden.

Hausaufgabe 5.1 Adressierung (10 Punkte)

In dieser Aufgabe soll ein komplettes Bussystem mit mehreren Slave-Teilnehmern aufgebaut werden. Hierzu werden zuerst einige Module erstellt und dann verbunden.

Das System soll aus folgenden Slaves bestehen:

- RAM 32KB
- ROM 4KB
- ROM 8KB
- ROM 16KB
- 8 Register zu je 32 Bit

Jede Adresse adressiert ein Byte. Alle Datenleitungen sollen 32 Bit breit sein.

Hausaufgabe 5.1.1 Adressmap

Geben Sie eine gültige Adressmap an. Memory-Aliasing ist erlaubt. Achten Sie bereits darauf, dass eine möglichst effiziente Implementierung des Decoders möglich ist (siehe nachfolgenden Aufgabenteil).

Übung zur Vorlesung Einführung in Computer Microsystems

Hausaufgabe 5.1.2 Adressdecoder

Implementieren Sie als Verilog-Modul einen Adressdecoder für einen Bus mit einer CPU als Initiator/Master. Der Decoder soll mit möglichst wenigen Gattern auskommen und für jeden der folgenden Slave-Teilnehmer ein separates SELECT-Signal erzeugen.

Hausaufgabe 5.1.3 Speichermodul

Implementieren Sie den RAM-Speicher als Verilog-Modul. Es soll folgende Schnittstelle besitzen:

```
module ram(  
    input wire clk,           //Takt  
    input wire reset,         //synchroner Reset  
    input wire select,        //Select-Leitung  
    input wire we,            //Write-Enable, =1 wenn geschrieben werden soll  
    input wire [ ] addr,      //Adresse  
    input wire [31:0] datain,  //Schreibdaten  
    output wire [31:0] dataout); //Lesedaten  
  
endmodule
```

Hausaufgabe 5.1.4 Register

Implementieren Sie das Register-Modul in Verilog. Es soll folgende Schnittstelle besitzen:

```
module register(  
    input wire clk,           //Takt  
    input wire reset,         //synchroner Reset  
    input wire select,        //Select-Leitung  
    input wire we,            //Write-Enable, =1 wenn geschrieben werden soll  
    input wire [2:0] addr,     //Adresse  
    input wire [31:0] datain,  //Schreibdaten  
    output wire [31:0] dataout); //Lesedaten  
  
endmodule
```

Hausaufgabe 5.1.5 Testbench

Schreiben Sie eine Testbench, die alle Module verbindet und alle Funktionen testet. Überlegen Sie sich hierzu sinnvolle Adressreihenfolgen und dokumentieren Sie, was in jedem Testcase überprüft wird. Als ROM sollen Sie das Modul aus den Vorlesungsfolien 5-89 verwenden. Es darf nicht verändert werden! Die Testbench muss nicht synthetisierbar sein, die anderen Module müssen jedoch synthetisierbar sein. Laden Sie alle Module im Moodle hoch.

Hausaufgabe 5.1.6 Adressdecoder 2

Verändern Sie Ihren Adressdecoder so, dass kein Memory-Aliasing auftritt.

Plagiarismus

Der Fachbereich Informatik misst der Einhaltung der Grundregeln der wissenschaftlichen Ethik großen Wert bei. Zu diesen gehört auch die strikte Verfolgung von Plagiarismus. Weitere Infos unter www.informatik.tu-darmstadt.de/plagiarism