Universität Bremen – Fachbereich 3
Mathematik & Informatik

Prof. Dr. Rolf Drechsler, drechsler@informatik.uni-bremen.de, MZH 3510
Dipl.-Inf. Beate Muranko, bmuranko@informatik.uni-bremen.de, MZH 3550

2. Übungsblatt zur Vorlesung

Technische Informatik 1

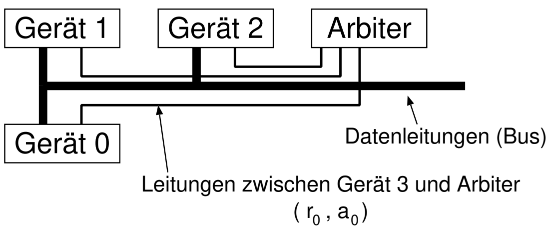
Aufgabe 1 (2 Punkte)

a) Beschreibe den Unterschied zwischen synchroner und asynchroner Datenübertragung auf einem Bus.
b) Welche Vorteile haben die jeweiligen Übertragungs-Schemata?

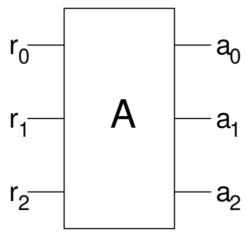
Aufgabe 2 (6 Punkte)
Ein Bus-Arbiter ist eine Komponente, die den Zugriff mehrerer Geräte auf einen Bus regelt. Abbildung 1(a) zeigt die schematische Darstellung eines Busses, auf den drei Geräte zugreifen können. Da zu jeder Zeit nur ein Gerät die Datenleitungen benutzen darf, wird der Arbiter benötigt. Bevor ein Gerät den Bus benutzt, signalisiert es die Anforderung (*Request*). Der Arbiter entscheidet dann, welchem Gerät der Bus zugeteilt wird und reagiert mit einer entsprechenden Bestätigung (*Acknowledge*).

Hier ist ein Bus-Arbiter für drei Geräte zu realisieren. Jedes Gerät hat eine Request-Leitung (r_0, r_1, r_2). Der Arbiter signalisiert einem Gerät über die Acknowledge-Leitung (a_0, a_1, a_2), ob es den Bus zugeteilt bekommt. Dabei dürfen keine zwei Geräte gleichzeitig ein Acknowledge erhalten.

Gib eine Boolesche Funktion $A : \mathbb{B}^3 \rightarrow \mathbb{B}^3$ durch eine Wahrheitstabelle und durch Boolesche Ausdrücke an, die diesen Arbiter realisiert. Dabei fordert Gerät i den Bus durch $r_i = 1$ an. Der Bus wird an Gerät i durch $a_i = 1$ zugeteilt. Priorität bei der Busvergabe hat das Gerät mit der kleinsten Zahl, d.h. wenn z.B. Gerät 1 und Gerät 2 den Bus zugleich anfordern, wird der Bus an Gerät 1 zugewiesen.



(a) Bus mit 4 Geräten und Arbiter



(b) Blockschaltbild des Arbiters

Abbildung 1: Bus-Arbiter

Aufgabe 3 (2 Punkte)
Zeige, dass es möglich ist, eine Zuweisung mittels
a) indirekter Adressierung (Rdest := Mem[Mem[Rsrc]])
b) indizierter Adressierung (Rdest := Mem[Rsrc+4])
nur unter Verwendung von absoluter Adressierung (Rdest := Mem[Rsrc]) zu realisieren.

Aufgabe 4 (6 Punkte)

Die *Fibonacci Folge* ist rekusiv definiert durch:

$$fib(n) = \begin{cases} 0 & , \quad n = 0 \\ 1 & , \quad n = 1 \\ fib(n-2) + fib(n-1) & , \quad n > 1 \end{cases}$$

Gib ein **iteratives** Assembler-Programm an, welches die n-te Fibonacci Zahl berechnet. Der Parameter n steht dabei zu Beginn in dem Register `$t0` zur Verfügung. Kommentiere die einzelnen Assemblerbefehle.

Aufgabe 5 (4 Punkte)

Es soll ein neuer Rechner für ein spezielles Einsatzgebiet entworfen werden. Die Rechengenauigkeit aller Operationen soll 16 Bit sein. Aufgrund der vorkommenden Operationen werden 182 Instruktionen benötigt. Der Rechner hat 40 Register bei einem Adressraum von maximal 250K. Es wird eine Load/Store-Architektur verwendet: Befehle, die auf den Speicher zugreifen, haben zwei Register als Operanden (Befehl 1: Rdest := Mem[Rsrc], Befehl 2: Mem[Rsrc] := Rdest). Kein Befehl hat mehr als drei Operanden, mindestens zwei Operanden sind Register, der dritte kann eine 8-Bit-Konstante oder ein Register sein.

a) Wie breit müssen die Register mindestens sein?

b) Wieviele Bit werden benötigt, um eine Instruktion zu codieren?
   Begründe Deine Überlegungen.

Abgabetermin: zu <u>Beginn</u> der Vorlesung am 15.05.2006