ALBERT-LUDWIGS-UNIVERSITÄT FREIBURG
INSTITUT FÜR INFORMATIK

Dr. Jürgen Ruf

Freiburg, 17. April 2002

1. Übungsblatt zur Vorlesung

Rechnerarchitektur

Aufgabe 1

Gegeben sei folgende endliche Zustandsmaschine eines einfachen Getränkeautomaten:

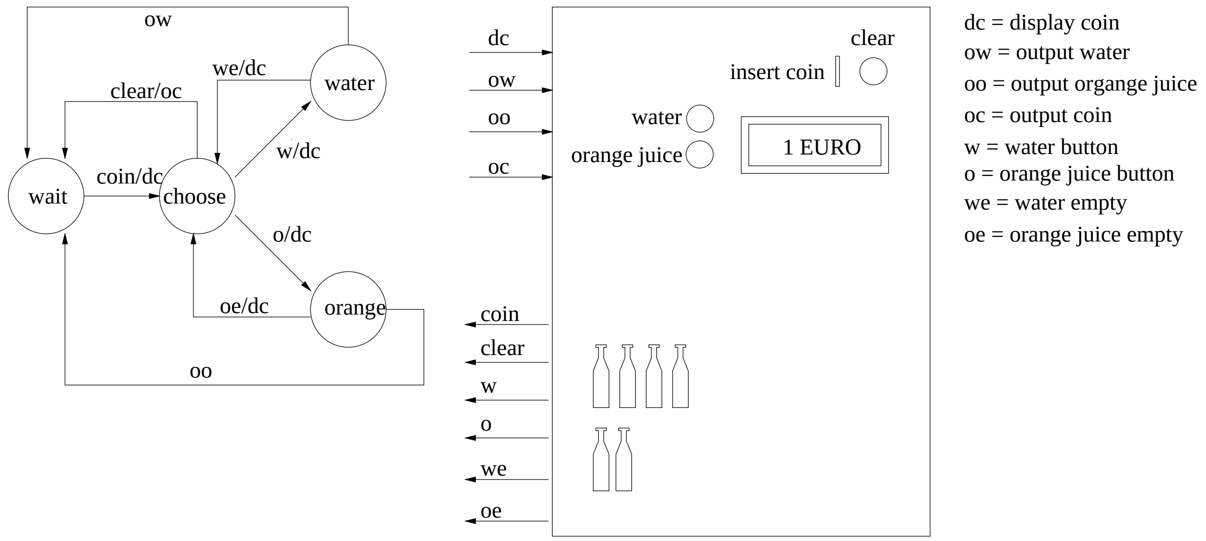

Abbildung 1: Getränkeautomat

Auf der linken Seite ist die Automatensteuerung. Nur die aktiven Signale (=high) sind eingezeichnet. Der Automat soll in seinem aktuellen Zustand verweilen, solange keine Bedingung einer ausgehenden Kante wahr wird, ausser in den Zuständen "water" und "orange", die sollen sofort nach ihrem Betreten wieder verlassen werden, je nach dem ob noch die entsprechenden Getränke im Automat sind oder nicht. Auf der rechten Seite ist der Automat mit seinen Funktionseinheiten und seiner Schnittstelle zur Steuerung gezeichnet.

a) Schreiben Sie ein Verilog Programm, das auf Verhaltensebene die Steuerung des Automaten repräsentiert.

b) Geben Sie einheitliche Richtlinien zur Notation von MEALY-Automaten mit Verilog an. Wie können Zustände kodiert und dargestellt werden? Wie wird die Zustandsübergangsfunktion repräsentiert? Wie kann die Ausgabefunktion angegeben werden?

c) Erzeugen Sie eine Testbench, um die Verhaltenbeschreibung zu validieren.

d) Führen Sie von Hand (mit den Techniken aus TI1 und TI2) eine Gattersynthese durch. Schreiben Sie nun ein zweites Verilog-Programm, das die Steuerung auf Strukturebene beschreibt, nutzen sie die *built-in primitives* für die logischen Gatter. Da es kein built-in primitive für ein Flipflop gibt, müssen Sie dieses als einfaches Verhaltensmodul beschreiben und dann für die Schaltungsbeschreibung instantiieren.

e) Welchen Unterschied sehen Sie zwischen der Verhaltenbeschreibung und der Gatternetzliste?

Aufgabe 2

a) Erweitern Sie den in der Vorlesung vorgestellten Streamline Codegenerierungsalgorithmus, so dass auch vollsynchrone Schaltungen (mit genau einem Taktsignal für alle Flip-Flops) simuliert werden können.

b) Wenden Sie Ihren in a) erstellten Algorithmus auf die strukturelle Automatenbeschreibung aus Aufgabe 1 an.

c) Moderne Simulatoren koppeln Streamline und Critical-Event-Scheduling, warum? Geben Sie einige Beispiele an, bei denen die Simulation mit Streamlinecode nicht mächtig genug ist.

Aufgabe 3

Als nächster Entwurfsschritt soll die Automatensteuerung aus Aufgabe 1 mit temporallogischer Modellprüfung verifiziert werden. Zu diesem Zweck müssen die zu verifizierenden Eigenschaften formal in Temporallogik angegeben werden.

a) Formulieren Sie folgende Sätze in LTL:

- Wenn Geld eingeworfen wurde und Wasser (Orangensaft) gewählt wurde und sich genügend Flaschen im Automaten befinden, dann wird innerhalb von 10 Zeitschritten Wasser (Orangensaft) ausgegeben.
- Es wird niemals Orangensaft und Wasser gleichzeitig ausgegeben.
- Wenn Geld im Automat ist und die clear-Taste für mindestens zwei Zeitschritte gedrückt wird, dann wird das Geld auch wieder ausgegeben.

b) Überlegen Sie sich weitere Spezifikationen und notieren Sie diese als LTL-Formeln.

Aufgabe 4

a) Für einen grafischen Coprozessor soll die Sinus-Funktion in HW berechnet werden. Schreiben Sie einen Algorithmus (C, Verilog, Pascal, ...), der bei gegebenem Eingabewert x die Funktion $\sin(x)$ annähert (Tipp: Taylorentwicklung). Führen Sie nun die Schritte der High-Level-Synthese durch, um diese Funktion in eine Registertransferebene zu übersetzen. Gehen Sie davon aus, dass es ein Divisionswerk, ein Multiplikationswerk, eine Inkrementiereinheit und eine ALU mit Addition, Komplementbildung (Multiplikation mit -1) und Vergleichsoperationen gibt. Alle Operationen ausser der Division benötigen eine Zeiteinheit. Die Division benötigt drei Zeiteinheiten.

- Stellen Sie den Problemgraphen für Ihren Algorithmus auf.
- Erzeugen sie einen Ablaufplan für den Problemgraphen mit den vorgegebenen Resourcenbeschränkungen.
- Bestimmen Sie die Anzahl der Register für Ihren Ablaufplan.
- Ordnen Sie Ihren Ablaufplan so um, dass die Zahl der benötigten Register kleiner wird.
- Skizzieren Sie den endlichen Automaten zur Steuerung der Funktion, so dass ersichtlich wird, welche Operation auf welchen Registern zu welchem Zeitpunkt auszuführen ist.

b) Welche programmiersprachlichen Konstrukte lassen sich nur sehr schwer in HW realisieren und werden deshalb von kommerziellen High-Level Synthesetools nicht unterstützt?