1 Einleitung

Datum: 16.9.2008, 10 Uhr 15
Prfer: Prof. J.K. Anlauf
Beisitzer: Armin Felke
Thema: Einführung i. d. technische Informatik, Rechnerorganisation
Dauer: 45 Minuten
Note: 1,0

2 Verlauf

Die Prüfung begann damit, dass Herr Anlauf zuerst im Stau stand und dann keinen Beisitzer fand – toll organisiert, die Rechnerorganisatoren. Seit dem Vordiplom in dem Fach habe ich es immerhin geschafft, meine Wartezeit vor der Prüfung um ganze 10 Minuten zu verkürzen auf eine Stunde. Verdammte abgelegene Informatiker.

- Nach dem üblichen Smalltalk, während Herr Anlauf auf seinen Beisitzer wartete, begann die Prüfung mit der Frage danach, was eine Normalform für eine Schaltfunktion ist (eine Konvention, um diese eindeutig darzustellen) und nach Beispielen (DNF, KNF, Wahrheitstabelle, ROBDD). Letzteres durfte ich ihm dann noch erklären (binärer Entscheidungsbaum, nach bestimmten Regeln reduzieren).

- Anschließend sollte ich ihm das Quine-McCluskey-Verfahren zur Bestimmung von Minimalformen von Schaltfunktionen erklären. Ich habs anhand eines Beispiels über ein KV-Diagramm gemacht – hier natürlich wichtig, dass man es im allgemeinen Fall über Tabellen macht. Schritt 1 besteht im Verschmelzen nebeneinanderliegender Einsen im KV-Diagramm (der entsprechenden Minterme in der Wahrheitstabelle), wobei man bei der Ausführung als Tabelle vorher die Minterme nach Anzahl der Einsen einteilt, um die Suche zu vereinfachen. Dies wird rekursiv angewandt, wobei mehrfach auftretende Blcke gestrichen werden. Das Ergebnis sind die sog. Primimplikanten.
  Schritt 2 besteht dann in der Suche nach wesentlichen Primimplikanten. Hier wollte er explizit das Stichwort „Primimplikantentafel“ hören sowie wie diese aufgebaut ist (zu überdeckende Minterme als Spalten, Primimplikanten als Zeilen).

- Weiterhin wollte er den Unterschied zwischen Schaltnetz und Schaltwerk erklärt haben, eine schematische Zeichnung eines allgemeinen Schaltwerks sowie die Begriffe Mealy- und Moore-Automat.

- Dann ging er über zur Rechnerorganisation. Die technische Informatik hat er ein wenig vernachlässigt, wohl auch weil ich im Vordiplom schon mal drüber geprüft wurde :-) Seine erste Frage hier war die, was denn eine ISA sei (Schnittstelle

zwischen low-level-Software und Hardware, aus Kompatibilitätsgrunden vereinheitlicht)

- Dann fragte er nach Besonderheiten von MIPS (alle Anweisungen außer load/store arbeiten ausschließlich auf Registern, Regularität, einheitliche Befehlslänge von 32 Bit – hier musste er viel nachbohren, weil mir nicht klar war, was er unter „Länge“ verstand, hab viel über Operandenanzahl geschwafelt ;-)).

- Anschließend sollte ich ihm detailliert den Unterschied zwischen Einzyklus- und Mehrzyklusdatenpfad erklären. Wichtig ist hier die Anzahl der funktionalen Einheiten (nur eine ALU, dafür zusätzliche Zwischenregister und Multiplexer im MZDP), die Steuerung (Schaltnetz im EZDP, Schaltwerk im MZDP) und die Taktzeit (im MZDP geringer als im EZDP, in unserem Beispiel mindestens 1/5, nämlich die Taktzahl des längsten Befehls (deutsche Sprache, schwere Sprache, es kommt nicht gut, wenn man geringere Taktzeit mit geringerer Verbesserung vertauscht und das dann auch noch ausspricht ;-)), praktisch ein wenig mehr wegen der zusätzlichen Multiplexer und Register). Warum ist der MZDP besser? (unterschiedlich lange Befehle)

- Nun zu Pipelining. Geht es vom Einzyklus- oder vom Mehrzykluspfad aus? Einzyklus, weil im MZDP funktionale Einheiten mehrmals genutzt werden, was beim Pipelinen dazu führen könnte, dass mehrere Befehle sich um eine funktionale Einheit streiten. Was brauchen wir an zusätzlicher Hardware? Pipelineregister, Register für Steuersignale, da eine Steuerung per Schaltwerk von 5 Befehlen abhängen müsste und daher viel zu aufwendig wäre. Weiterhin wurde noch nach Speedup gefragt (maximal Pipelinestufenanzahl, realistisch betrachtet weniger wegen Pipelineregistern und diversen Hazards)

- Der Rest der Prüfung war Ansätzen zur weiteren Beschleunigung von Prozessoren gewidmet – Superpipelining, macht eine Pipeline mit 1000 Stufen Sinn? Nein, weil funktionale Einheiten diskret sind und die Pipelineregister schließlich die Pipeline ausbremsen. Natürlich ist mir hier alles Andere als das Argument mit den Registern eingefallen, ich war schon bei der Stromversorgung, obwohl die Register als begrenzender Faktor vorher schon mal da waren.

- Superskalare Architekturen: Wie funktioniert's (Befehle in Gruppen aufteilen – in unserm Beispiel in R-Typ/branch und load/store – und je nach Gruppe durch eine eigene ALU schicken), wie sieht's mit Hardware aus (ein Registersatz, da nur ein Prozess, mehrere ALUs oder in der Praxis auch nur Addierer, da billiger und z.B. für die Adressberechnung bei load/store ausreichend), wo ist der Haken? (In unserm Beispiel z.B. lange Folgen von R-Typ-Befehlen, bei denen die zweite ALU nichts zu tun hat)

- Dynamic Pipeline Scheduling: Wie funktioniert's (mehrere ALUs, die die Befehle out of order (auf das Stichwort legt er Wert) bearbeiten), Steuerung (kompliziertes zentrales Steuerwerk, z.B. Scoreboard), Vorteile gegenüber superskalaren

Architekturen (siehe oben – bei DSP könnte man in obigem Beispiel eine nachfolgende load-Instruktion vorziehen oder so etwas)

- Hyperthreading: Was ist das (mehrere Prozesse auf mehreren logischen Prozessoren bei nur einem physikalischen Prozessor, z.B. parallel oder per Umschalten bei Zugriff auf Hauptspeicher), warum ist es sinnvoll (hier wollte er mich mittels einem Beispiel in die Ecke drängen, in dem zwei Threads gleichzeitig wegen eines Hauptspeicherzugriffs warten – so was wird natürlich fast immer durch die Existenz eines Caches verhindert), wie sieht die Hardware aus?
  Bei letzterer Frage hab ich arg gehangen, natürlich bei zwei Threads zwei PCs, auch zwei Registersätze – hier war ich mir nicht sicher, obwohl es im Nachhinein völlig einleuchtend ist – aber i.d.R. eine ALU (wenn nicht gerade mit DSP kombiniert, sonst hätten wir ja schon zwei physikalische Prozessoren) und ein Cache (flexibler als zwei Caches, arbeitet dann mit Process-ID).

Nach relativ kurzer Beratung, während der ich schon leicht nervös und nicht ganz sicher war, ob es denn gut gelaufen war, durfte ich mir dann meine Note abholen und mich freuen :-)

3 Fazit

Herr Anlauf ist als Prüfer sehr nett und nimmt auf die Nervosität seiner Prüflinge durchaus Rücksicht, stellt aber gerne Fragen, bei denen mir nicht ganz klar war, auf was es hinauslaufen soll. Dementsprechend gab es von mir bei vielen Fragen auch Antworten, die zwar halbwegs brauchbar waren, aber nicht der (für ihn) entscheidende Punkt. Offenbar hat er mir das aber nicht negativ angekreidet :-)
Was mich überrascht hat, ist, dass er sehr viel zu fortgeschrittenen Pipelinetechniken wissen wollte, obwohl der Bereich in der Vorlesung relativ am Rand stand, und dass er ganze Themekomplexe ausgelassen hat (große Teile der Technischen, Rechnerarithmetik, Cache). Unterm Strich kann ich ihn durchaus als Prüfer empfehlen.