Prüfungsfragen Hardware-Software-Co-Design 2004

Hard- und Software Codesign
Prof. Teich
April 2004

Bemerkungen zu Prüfung und Prüfer

- Insgesamt was die Prüfung in Ordnung, Herr Teich ist sehr nett und hilft einem, wenn man die Frage nicht gleich versteht. Auch über die Benotung war ich zufrieden. Allerdings sollte man scheinbar alle Formeln auswendig wissen.
- Die Prüfung war in 4 Aufgaben untergliedert, die Fragen waren nicht strikt nach Fach getrennt

Das Doppeldachmodell

- Welche unterschiedlichen Abstraktionsebenen gibt es bei Hardware und bei Software?
  → System- Architektur -Logik bzw. Modul -Block, Verhalten/ Struktur, Verfeinerungsstufen
- Was hat man bei der jeweiligen Stufe bei der Verhaltenssicht, was bei der Struktursicht?
  → System: Verhalten: Leistungsanforderungen an Funktion, Zeitverhalten, Kosten usw.
  → Struktur: Kommunizierende Komponenten, Funktionale Einheiten, Busse, Prozessoren
  Das ganze dann noch für Architektur -Logik bzw. Modul -Block
- Zwischenfrage: Nennen Sie ein zeitkritisches und ein leistungskritisches eingebettetes System:
  → Airbagsteuerung, tragbarer Mp3 Player

Petrinetze

- Herr Teich zeichnet ein Petrinetz.
- Was ist was?
  → Stellen, Transitionen usw.
- Welche besondere Form von Petrinetz ist das Gezeichente?
  → Ein Datenflussgraph (jede Stelle hat genau einen Vor- und Nachbereich, Kantengewichte alle eins
- Kann es da einen Konflikt geben?
  → Nein, siehe Definition von Konflikt
- Wann ist ein Datenflussgraph lebendig?
  → Wenn jeder Zyklus mindestens eine Marke enthält

Ablaufplanung

- Ablaufplanung Herr Teich zeigt ein Blatt mit Problemgraph und Ressourcegraph
- Was bedeutet was?
  → Allokation, Berechnungszeit, Problemgraph und Ressourcegraph, Datenabhängigkeiten
- Was sind die Grundaufgaben der Synthese?
  → Allokation, Ablaufplanung, Bindung
- Wie kann man modellieren dass die Multiplikationen auch auf der Alu laufen können?
  → Einfach alle Multiplikationsknoten mit der Alu verbinden
- Welche Ablaufplanungsverfahren gibt es hier?
  → Für Ablaufplanung mit Ressourcenbeschränkung gibt es List Scheduling , Force Directed Scheduling, Ganzzahlige lineare Programmierung und erweiterten Asap/Alap
- Führen Sie mal einen beliebigen vor?
  → Schlau wie ich bin nehm ich "Ganzzahlige lineare Programmierung" War keine sonderlich gute Idee, weil des voll schwierig ist, aber wenigstens musste ich's nicht lösen sondern nur sagen welche Gleichungen man aufstellt. List Scheduling wäre viel einfacher gewesen...
- Welche Verfahren gibt es ohne Ressourcenbeschränkung?
  → Asap/ Alap, Bellman Ford
- Ordnen sie die 4 Algorithmen nach der Qualität des Ergebnisses?
  → Ganzzahlige lineare Programmierung (exakt, braucht aber ewig) Force Directed Scheduling(berechnet dynamisch Prioritäten) List Scheduling(berechnet statisch Prioritäten) erweiterten Asap/Alap(macht schlechte Pläne)

Markov-Ketten

- Ein Server, 4 Prozesse: der Server wählt in jedem Zeitschritt eine Prozess zufällig aus- wenn der Prozess läuft, wird er deaktiviert, ist er deaktiviert, wird er aktiviert. Das lässt sich als stochastischer Automat mit 5 Zuständen modellieren (0 laufen bis 4 laufen)
- Zeichnen Sie die übergangswahrscheinlichkeiten?
  → Grübel grübel aha gar nicht so schwer
- Und wie kann man jetzt sehn ob es einen stationären Zustand gibt?
  → Irgendwie kann man des sehn, so ne Formen mit GgT, aber ich wusste nur noch, dass man's auch berechnen kann: Pi= Pi mal übergangsmatrix, Summe über alle Elemente von Pi gleich 1
- Dann schreiben Sie mal die übergangsmatrix hin?
  → Hab mich 3465 mal vertan, weil's mit P00 und nicht mit P11 angeht

Eingebettete Systeme, Ereignisgesteuerte Systeme, Prof. Dr. Jürgen Teich, Beisitzer: Dipl.-Ing. Cornelia Grabbe, April 2004

Bemerkungen zu Prüfung und Prüfer

- Ergebnis: 1,7
- Freundlicher Prüfer
- Lässt auswählen, mit welchem Teilgebiet man beginnen möchte
- Erwartet, dass man Antworten auf Papier aufschreibt
- Kein lockeres Gespräch, sondern Prüfer hatte Fragen und Aufgaben vorbereitet
- Fragen klar und ausschließlich auf Stoff bezogen
- Legt Wert auf exakte Ausdrucksweise (Definitionen lernen)
- Erwartet, dass man Dinge lückenlos aufzählen kann (z.B. wenn einem von 5 behandelten Schedulingverfahren nur 3 einfallen ist das ein Fehler)
- Übungsaufgaben mindestens genauso wichtig wie Stoff des Skripts
- Kompliziertere Formeln und Algorithmen wurden aber nicht gefragt

Fragen

Eingebettete Systeme

- Anforderungen eingebetter Systeme? (Echtzeitfähigkeit, Kosten, ...)
- Diagramm zum Entwurfsparadigma (Dreieck) hinzeichnen und erklären
- Verfeinerungsstufen bei Hard- und Software? (System, Modul-, Block-, Logikebene, ...)
- Warum kann nicht alles in Software realisiert werden? (Geschwindigkeit, Energieverbrauch, ...)
- Allgemeine Aufgaben bei Synthese? (Allokation, Ablaufplanung, Bindung)
- Welche Verfahren zur Ablaufplanung bei Hardware gibt es? (ASAP, ALAP, Listscheduling, ...)
- Prüfer gibt Sequenzgraphen und Ressourcegraphen vor. Was ist das? Dann musste ich ASAP und Listscheduling am Graphen durchführen (wie in Übungsaufgabe)
- Prüfer gibt Petri-Netz vor. Was ist das? (markierter Graph) Ist in diesem Graph ein Deadlock möglich und warum?
- Allgemeine Fragen und Definitionen zu Erreichbarkeit, Lebendigkeit, Aktivierbarkeit, Beschränktheit bei Petrinetzen
- Erreichbarkeitsgraph eines Petrinetzes erklären

Ereignisgesteuerte Systeme

- Wie definieren sich ereignisgesteuerte Systeme (nicht zeitgesteuert, ...)
- Was ist eine Zufallsvariable? (Definition)
- Was ist ein stochastischer Prozess? (Definition)
- Was ist eine Markovkette? (Definition)
- Aufgabe zu Markovketten: Prüfer legt drei Münzen auf den Tisch. Bei jedem Zeitschritt wird eine zufällig ausgewählte Münze umgedreht.
- Zustandsübergangsdiagramm zeichnen
- Übergangsmatrix aufstellen
- Ist der Graph reduzierbar? (nein)
- Kann man eine stationäre Analyse machen? (nein, weil Graph periodisch ist)
- Wie sieht man das an der Übergangsmatrix? (alle Diagonalelemente sind 0)

Scheinprüfung Hareware-Software-Co-Design Prof. Dr. Jürgen Teich, SS 2004

- 1.
  - Welche Typen von Prozessoren gibt es?

    General Purpose, Microcontroller, Dsps, Asips, Fpgas, Asics
  - Mit welchen Kriterien kann man die Prozessoren vergleichen?
    Speicher, Befehlssatz, Wortbreite, Taktrate, Datendurchsatz, Pipelining, Flexibilität, Grösse, Leistungsaufnahme
  - Ordnen Sie die Prozessoren ein nach diesen Kriterien?
  - Was bedeuten die Abkürzungen?
- 2. Pareto
  Diagramm in das Punkte eingetragen sind, Skalen sind: Ausführungszeit und Kosten Die Punkte sind mögliche Entwürfe.
  - Welche Punkte würden Sie auswählen?
    Pareto-Punkte
  - Was sind Pareto-Punkte?
  - Was heisst dominieren?
  - Jetzt kommen 2 neue Punkte hinzu, was sind nun die Pareto-Punkte?
  - Stellen Sie eine Funktion auf, die berechnet, ob ein Punkt einen anderen dominiert.
- 3.
  - Welche grundsätzlichen Probleme gibt es bei der Codegenerierung, oder allgemein bei der Synthese?
    Allokation, Bindung, Ablaufplanung
  - Ist die Allokation bei der Codegenerierung nötig?
    Wenn der Prozessor vorgegeben ist, dann nicht. Wenn er selbst entworfen wird, dann schon.
  - Welche speziellen Aufgeben hat man bei der Codegenerierung?
    Registervergabe, Befehlsauswahl, Befehlsreihenfolge
  - Wo ist die Befehlsauswahl wichtiger und schwieriger, bei Risc oder Cisc?
    Bei Cisc, sehr spezielle Befehle (Complex Instruction Set)
  - Welche Repräsentationsmöglichkeiten für Zwischencode gibt es?
    3-Address-Code, DAG
  - Was ist der Vorteil von 3-Address-Code?
    Die Befehlsreihenfolge ist schon vorgegeben
- 4.
  - Das Partitionierungsproblem, was ist das?
    Abbildung von n Individuen auf m Blöcke, M hoch n Möglichkeiten, NP-vollständig
  - Welche Lösungsverfahren gibt es?
    Simulated Annealing, Tabu Search, Random Mapping, Evolutionäre Algorithmen, Hierarchisches Clustern

  - Was ist das Hauptproblem beim Hierarchischen Clustern?
    Finden einer Closeness-Fkt.

- 5.
  - Was ist das Register-Transfer-Graph-Kriterium?
    Wenn 2 Register mit dem gleichen Befehl zu einem dritten Register führen, und ein Zyklus zwischen den ersten beiden existiert, dann muss es in jedem Zyklus einen Speicherknoten geben.
  - Wenn das erfüllt ist, was folgt daraus?
    Optimaler Code kann in linearer Zeit O(n) erzeugt werden.

Hardware-Software-Co-Design und Eingebettete Systeme Prof. Teich Oktober 2004

Bemerkungen zu Prüfung und Prüfer

- Entspannte Prüfungsatmosphäre, Herr Teich sehr freundlich.
- Allerdings hatte er die Fragen vorbereitet und stur abgearbeitet.
- Im Gegensatz zu den meisten anderen Prüfungen musste man viel vorrechnen.
- Kommt man nicht weiter, hilft Herr Teich einem; auch kein Problem, wenn man eine Frage mal nicht gleich versteht.
- Die Fragen gingen sehr in die Tiefe, einige herausgepickte Themen wurden intensiv besprochen.
- Die Fragen gingen nicht unangemessen über den Vorlesungsstoff hinaus, es genügt, das Buch (Hardware-Software-Co-Design von Teich) teilweise zu lesen und das Skript zu lernen.

Eingebettete Systeme

- Welche Abstraktionsebenen gibt es beim Entwurf von eingebetteten Systemen? →Doppeldachmodell hinzeichnen und erläutern.
- Welche Kriterien sollen beim Entwurf von eingebetteten Systemen optimiert werden? →Kosten, Leistungsverbrauch, Entwicklungszeit, Gewicht, Größe, Sicherheitsanforderungen...
- Herr Teich zeigt drei Bilder: Ein Gatternetz (And- und Or-Gatter mit Verbindungen), ein Netz aus Prozessoren und Bussen und ein Netz mit ALU und anderen Komponenten. Wo sind diese Bilder im Doppeldachmodell einzuordnen? Welche Ebene, welche Sicht? →Gatter: HW-Logik-Ebene, Struktursicht; Proz.+Busse: Systemebene, Struktursicht; ALU etc: HW-Architekturebene, Struktursicht.
- Was versteht man unter iterativer Ablaufplanung?
- Herr Teich zeigt einen Problemgraphen mit Kantengewichten und einen Ressourcengraph. Was ist das, was bedeuten die Kantengewichte? → Kantengewichte bedeuten Datenabhängigkeiten zwischen den Iterationen.
- Warum erstellt man nicht einen sequentiellen Ablaufplan für eine Iteration und lässt diesen immer wieder laufen? → Keine optimale Ressourcenauslastung.
- Was macht man stattdessen? → Parallele Berechnung von Aufgaben unterschiedlicher Iterationen. Funktionale Fließbandverarbeitung.
- Die zu dem Problemgraphen gehörenden Abhängigkeiten als Funktionen formulieren. → $x_1[n] = f_1(x_1[n-1], x_3[n-2])$...
- Kann für dieses Beispiel ein iterativer Ablaufplan bestimmt werden? → Ja, da die Summe der Kantengewichte für alle gerichteten Zyklen größer null ist.
- Was wäre die kleinstmögliche Periode? → Maximales Zyklengewicht. Formel hinschreiben.
- Dann sollte ich für den Beispielgraphen das maximale Zyklengewicht bestimmen. → Lese Berechnungszeiten der Knoten aus dem gegebenen Ressourcengraphen. Dann einsetzen aller elementaren Zyklen in die Formel.

- Wie viele Ressourcen braucht man mindestens, um die minimale Periode erzielen zu können. → Zähle Berechnungszeiten zusammen und teile durch die minimale Periode (das maximale Zyklengewicht).
- Wie könnte man das iterative Ablaufplanungsproblem lösen? → Mit Integer-linearer Programmierung (musste ich aber die Formeln zum Glück nicht mehr hinschreiben).
- Herr Teich zeichnet einen Synchronen Datenflussgraph.
- Was ist das? Wie funktioniert ein Synchroner Datenflussgraph? → Knoten wie Transitionen bei einem Petri-Netz, können feuern, wenn genügend Marken auf den Eingangskanten liegen und produzieren Marken auf die Ausgangskanten.
- Wie heißen die Knoten bei Synchronen Datenflussgraphen? → Aktoren
- Für welche Anwendungsgebiete wird er bevorzugt verwendet? → Datenflussdominante Anwendungen.
- Datenflussdominante Anwendungen laufen oft periodisch ab. Wie lässt sich das auf den Graph übertragen? → Die Anfangsmarkierung muss wieder erreicht werden können, der Graph muss also konsistent sein.
- Wie lässt sich feststellen, ob der Graph konsistent ist? → Besitzt die Topologiematrix Vollrang, ist der Graph nicht konsistent.
- Ist der gegebene Graph konsistent? → Er hatte keine Zyklen, also ja. (?)
- Berechnung des minimalen Feuerungsvektors. → Lösen des Gleichungssystems: $C^T * \lambda = 0$.
- Looped schedule formulieren.
- Wie kann man den Programmspeicher minimieren? → Möglichst wenige Knoten im looped schedule; im Beispiel kam jeder Knoten nur einmal vor, der war also minimal.
- Wie groß ist hier der Datenspeicheraufwand? → Maximal mögliche Anzahl von Markierungen ist ein Maß für den Datenspeicheraufwand.
- Wie kann dieser reduziert werden?

Hardware-Software-Co-Design

- Wie ist ein Compiler aufgebaut? → Lexikalische Analyse, Syntaktische Analyse, Semantische Analyse, Zwischencodegenerierung, Optimierung, Codegenerierung, Optmierung. Jeweils Erläuterung der Phasen.
- Welche Aufgaben müssen im Rahmen von Allokation, Bindung und Ablaufplanung erledigt werden? → Allokation: Bestimmen von Registeranzahl, Befehlssatz..., Bindung: Zuweisen der symbolischen Variablen zu den Registern, Abbildung der Hochsprachenbefehle auf die Maschinenbefehle (oftmals uneindeutig vor allem bei CISC), Ablaufplaung: Reihenfolge der Befehle festlegen.
- Was ist wichtig bei der Codegenerierung von Eingebetteten Systemen? → Der Code muss möglichst klein und effizient sein. Die Übersetzungszeit spielt keine Rolle.
- Wie wird Code generiert? → Hab ihm dann was über dynamische Programmierung erzählt, aber glaub nicht, dass er das hören wollte. Aber wo wir schonmal beim Thema waren:
- Welche Voraussetzungen müssen erfüllt sein, um die dynamische Programmierung zur Codegenerierung verwenden zu können? → Alle Register müssen gleichberechtigt sein, es darf nicht vorkommen, dass bestimmte Befehle nur mit bestimmten Registern gehen.
- Welche Aufgaben der Synthese werden dadurch erledigt? → Ablaufplanung und Bindung.
- (Jetzt wieder allgemein) Wie werden die Register vergeben? → Einige Register sind global für Schleifenvariablen o.ä. reserviert. Restliche Register z.B. durch Graphfärben (Aufstellen des Registerkonfliktgraphen) vergeben.
- Herr Teich zeichnet einen Problemgraph, nennt mir die verfügbaren Prozessoren, Busse und Zeiten, die die Knoten auf den einzelnen Prozessoren benötigen, sowie Kommunikationszeiten. Erstellen Sie den Architekturgraphen und den Spezifikationsgraphen.

- Auf welcher Ebene befinden wir uns? (System) Was sind die Aufgaben der Systemsynthese? → Zuweisung der Aufgaben auf die verschiedenen Hardware-Einheiten, Entscheidung, welche Einheiten verwendet werden sollen. HW-SW-Partitionierung.

- Welches wäre für den gegebenen Spezifikationsgraphen die kostenoptimalste Partitionierung? → Verwende nur einen Prozessor (es gab einen, auf dem alle Aufgaben ausgeführt werden könnten).

- Welches ist die zeitoptimalste Partitionierung? → Ich war zwar etwas verblüfft, aber sollte dann alle Möglichkeiten aufzählen und mich für die günstigste entscheiden (unter Einbeziehung der Kommunikationskosten). In dem Fall liefen zwei Tasks auf einem Prozessor und ein Task auf dem Spezialprozessor.

Hardware-Software-Codesign, Parallele Systeme
Teich, Hannig
Oktober 2004

Bemerkungen zu Prüfung und Prüfer

- Die Reihenfolge der beiden Fächer durfte ich selber bestimmen. Wie erwartet hat das erste dann mehr Zeit in Anspruch genommen. Was mich sehr gewundert hat, war dass zu Partitionierung gar keine Frage kam, obwohl das in der HSCD-Vorlesung ja doch der Schwerpunkt war.

- Faire Benotung, mit sehr gutem Ergebnis:)

Fragen

Hardware-Software-Codesign (Fragen von Herrn Teich):

- Welche Arten von Prozessoren werden in eingebetteten Systemen benutzt? Mikrocontroller: kontrollflussdominante Anwendungen, Register meist im Speicher -¿ schneller Kontextwechsel DSP: datenflussdominate Anwendungen, spezial Befehle, extra Datentypen/Register

- Was sind ASIP? (hätte ich das bei der ersten Frage auch erwähnen sollen?) Prozessoren mit speziellem Befehlssatz für die Anwendung

- Warum keine GP-Prozessoren? stromverbrauch, Timing nicht vorhersagbar (Pipelining, Caches)-¿nicht geeignet für Echtzeit-Anwendungen

- Sind Mikrocontroller und DSP eher RISC oder CISC Prozessoren im Vergleich zu GP-Prozessoren? Mikrocontroller: CISC, RAM/ROM ist das teure → möglichst hohe Codedichte DSP: CISC wegen spezial Befehle, z.B. Mult&Acc (meine Antwort), aber auch RISC damit alle Befehle gleich lang (seine Idee; scheint's beides gültig)

- Code, geschrieben in gegebenen Befehlssatz (nix spezielles): Grundblöcke markieren, Kontrollflußgraph der Grundblöcke malen, Worst-case execution time bestimmen (inkl ILP Bedingungen aufstellen); Vorhersage wegen Caches und Pipelining schwierig

Parallel Systeme (Fragen von Herrn Hannig):

- Unterscheidung von Parallelsystemen nach Flynn + kurze Erklärung: SISD, SIMD, MISD, MIMD

- Unterscheidung von Parallelsystemen nach Speicherarchitektur: Multiprozessor: Shared-Memory; UMA (physikalisch ein Speicher), NUMA (physikalisch hat jeder Prozessoer seinen Speicher, logisch ein Addressraum), COMA

- Multicomputer: Distributed-Memory; jeder Knoten eigener Computer, private Addressräume typische Anwendung für beide (ganz konkretest Beispiel gefragt): Multiprozessor: Bildverarbeitung (für Multicomputer fiel/fällt mir nix ein)

- Regelmäßiger Algorithmus gegeben: Signalflußgraph (2 Knoten, 4 Kanten, Gewichte 2-dimensional)+ Entfalltung malen; Schnittmengentransformation, so dass auf allen Kanten ein Register, auch algebraisch angeben; Prozessorfeld malen (eine Dimension wurde als Zeit definiert)

- Hier war die Zeit zu Ende. Eigentlich war wohl noch eine Frage geplant.