

Echtzeitsysteme

Rekapitulation

Lehrstuhl Informatik 4

07. Februar 2013

Gliederung

1 Rückblick

- II Einleitung
- III-1 Objekt ↔ Rechensystem
- III-2 Struktureller Aufbau
- IV-1 Abarbeitung periodischer EZS
- IV-2 Ereignisgesteuerte Ablaufplanung
- IV-3 Zeitgesteuerte Ablaufplanung
- V-1 Grundlegende Abfertigung nicht-periodischer EZS
- V-2 Zustellerkonzepte
- VI Rangfolge
- VII Zugriffskontrolle
- VIII Verteilte Echtzeitsysteme
- IX Exkurs: WCET-Analyse

II Einleitung

Historischer Bezug von SAGE bis in die Gegenwart

- von hardware-lastigen, auf militärische Zwecke zugeschnittene Spezialsysteme hin zu software-geprägten Alltagsgegenständen

Echtzeitbetrieb eines Rechensystems in seiner Umgebung

- Operateur, **Echtzeitrechensystem**, **kontrolliertes Objekt**

Echtzeitbetrieb bedeutet **Rechtzeitigkeit**

- **Ereignisse**, **Ergebnisse** und **Termine**
- **schwache**, **starke** oder **strikte** Echtzeitbedingungen
- weiche vs. feste vs. harte Echtzeitsysteme

Gewährleistung der Rechtzeitigkeit als Herausforderung

- **Determiniertheit** und **Determinismus**
- **rein zyklischer** vs. **asynchroner/nicht vorhersagbarer Betrieb**

Abgrenzung des Vorlesungsstoffes

- **Rechtzeitigkeit**, nicht **Verlässlichkeit**

III-1 Physik. Objekt ↔ Kontrollierendes Rechensystem

Fallbeispiel Quadropter das physikalische Objekt

- hat **Objektdynamik** \mapsto Zeitparameter d^{object} , d^{rise}
- \leadsto zeitliche Ausrichtung des kontrollierenden Rechensystems
 - **Abtastperiode** f^{sample} , **Latenz** d^{cpu} , **Totzeit** d^{dead}

Lageregelung ist nur ein kleiner Teil des kontrollierenden Rechensystems

- Sensoren \rightarrow Mikrocontroller \rightarrow Betriebssystem \rightarrow Anwendung
- Anwendung \rightarrow Betriebssystem \rightarrow Mikrocontroller \rightarrow Aktoren
- \leadsto alles trägt zur Latenz d^{cpu} bei

Programmunterbrechung in synchroner oder asynchroner Ausprägung

- **Zustandssicherung**, Verwaltungsgemeinkosten des schlimmsten Falls
- **Interrupts** machen determinierte Programme nicht-deterministisch
 - nicht zu jedem Zeitpunkt ist bestimmt, wie weitergefahren wird
- **Unvorhersagbarkeit**, **Überlast**, **Verzögerung**, \dots , **Nebenläufigkeit**
- \leadsto Wie detailliert muss man diese Vorgänge betrachten?

III-2 Struktureller Aufbau von Echtzeitanwendungen

Strukturelle Elemente einer Echtzeitanwendung

- Ereignisse – event-trigger und time-trigger
- einfache vs. komplexe Aufgaben und Arbeitsaufträge
- Aufgabe vs. Arbeitsauftrag vs. Faden

Ablaufsteuerung \mapsto Strategie & Mechanismus

- zweiphasiger Prozess: statische und dynamische Abbildung
- Verwaltungsgemeinkosten dynamischer Ablaufsteuerung
- Einplanung ist die Strategie, Einlastung ist der Mechanismus

Arbeitsweise ist zeit- oder ereignisgesteuert: Einplanung & Einlastung

- entkoppelt im zeitgesteuerten System (Taktsteuerung)
- gekoppelt im ereignisgesteuerten System (Vorrangsteuerung)

Zeitparameter sind Punkte und Intervalle auf der Echtzeitachse

- (sporadische) Auslösezeit, (absoluter) Termin
- Latenz, Antwortzeit und Schlupfzeit

Planbarkeit Problemstellung und Optimalität

IV-1 Abarbeitung periodischer Echtzeitsysteme

Periodische Aufgaben genügen für viele Echtzeitanwendungen

- Periode, WCET, relativer Termin, Phase, Hyperperiode
- Restriktionen des periodischen Aufgabenmodells
- viele Anwendungsfälle kommen mit rein periodischen Aufgaben aus

Ereignisgesteuerte Abarbeitung periodischer Aufgaben

- feste und dynamische Prioritäten auf Job- bzw. Task-Ebene

Einplanung \mapsto ereignis-, vorrang-, prioritätengesteuert

- Einplanungsaufwand: Auslöse- vs. Auswahlzeitpunkt von Jobs
- Ablauf Tabellen, Ablaufliste, Multi-Level-Queue

Zeitgesteuerte Abarbeitung periodischer Aufgaben

- Probleme einer naiven *Busy-Loop*-Implementierung
- Vorabwissen ermöglicht die Bestimmung statischer Ablaufpläne

Einlastung und Laufzeitkontrolle im Abfrage- oder Unterbrecherbetrieb

- Taktzähler, Zeitgeber, Zeitkontrolle
- stapelbasierte Abarbeitung statischer Ablauf Tabellen

IV-2 Ereignisgesteuerte Ablaufplanung periodischer EZS

Gebräuchliche Verfahren mit statischen und dynamischen Prioritäten

- statisch: RM, DM, dynamisch: EDF

Prioritätsabbildung mangels Systemprioritäten bzw. Prioritätsebenen

- gleichmäßig oder ungleichmäßig, *constant ratio mapping*
- Einfluss auf die Planbarkeit (engl. *schedulability*) eines Systems

Optimalität und Nicht-optimalität der vorgestellten Verfahren

- Nicht-optimalität von statischen Prioritäten und Vorrangsteuerung

Planbarkeitsanalyse für ereignisgesteuerte Ablaufplanung

- CPU-Auslastung und Antwortzeitanalyse
- Komplexität der Problemstellung
- Notwendige und/oder hinreichende Planbarkeitskriterien

IV-3 Zeitgesteuerte Ablaufplanung periodischer EZS

Integration der Ablaufplanung in den Entwicklungsprozess

- Integration vs. Spezifikation des Zeitverhaltens

Struktur zyklischer Ablaufpläne \leadsto „gute Anordnung“, Determinismus

- bevorzugter Ansatz bei „manueller Ablaufplanung“
- Rahmen, Rahmenlänge, Scheiben; *major/minor cycle*
- Überwachungsfunktion: Auslösung und Zeitüberschreitung

Algorithmische Bestimmung statischer Ablauf Tabellen

- ... wenn eine manuelle Ablaufplanung nicht in Frage kommt
- List-Scheduling: b-Level und t-Level, z.B. HLFET, ISH, DLS, ...
- Branch&Bound-Prinzip: erschöpfende Suche, optimale Verfahren

Betriebswechsel entflechten die Ablauf Tabelle des schlimmsten Falls

- erfordern ein systemweit koordiniertes Vorgehen
- \leadsto Tabellenwechsel, Betriebsmittelfreigabe/-anforderung, Nachladen

V-1 Grundlegende Abfertigung nicht-periodischer EZS

Nicht-periodische Aufgaben unterstützen interaktive Anwendungen

- minimale Zwischenankunftszeit, WCET, Termin
- aperiodische und sporadische Aufgaben; Übernahmeprüfung

Periodische Zusteller behandeln periodisch nicht-periodische Aufgaben

- Ausführungsbudget, Auffüllperiode, Auffüllzeit, Auftragsüberhang

Abfragende Zusteller verlieren ihr Ausführungsbudget bei Untätigkeit

- hohe Antwortzeiten, Überlast durch hochfrequentes Abfragen

Unterbrecherbetrieb behandeln nicht-periodische Ereignisse **sofort**

- zu Lasten der Termintreue periodischer Aufgaben

↪ nur als kontrollierter Unterbrecherbetrieb verwendbar

Hintergrundbetrieb nutzt den Schlupf periodischer Aufgaben

- schlechte Antwortzeiten, keine Beeinflussung periodischer Aufgaben

Slack-Stealing ↪ Kompromiss: Unterbrecher- und Hintergrundbetrieb

- Verdrängung, solange noch Schlupf verfügbar ist
- einfach bei Taktsteuerung, zu komplex für Vorrangsteuerung

V-2 Zustellerkonzepte und Übernahmeprüfung

Bandweite-bewahrende Zusteller geben ihr Ausführungsbudget nicht auf

- Verbrauchs- und Auffüllregeln des Ausführungsbudgets
- aufschiebbarer Zusteller, SpSL Sporadic Server

POSIX Sporadic Server ↪ Implementierung des SpSL Sporadic Server

- Spezifikation ist defekt
- Anhäufung und verfrühte Auffüllung des Ausführungsbudgets
- unzureichende zeitliche Isolation

Übernahmeprüfung sporadischer Aufgaben

- dichte-basierter Akzeptanztest für nach EDF geplante Systeme
- schlupf-basierter Akzeptanztest mithilfe sporadischer Zusteller

VI Rangfolge

Rangfolge mündet in gerichteten Abhängigkeiten

- Datenabhängigkeiten ↪ Erzeuger-Verbraucher-Beziehung
- Beispiel: Datenabhängigkeiten im I4Copter
- notiert in Abhängigkeits- bzw. Aufgabengraphen

Koordinierung gerichteter Abhängigkeiten

- analytische durch Einplanung, konstruktiv durch Kooperation

Umsetzung gerichteter Abhängigkeiten in Echtzeitanwendungen

- Beispiel: byte-weiser Empfang von Nachrichten
- statische Anordnung im Quellcode

analytische Koordinierung ↪ geeignete Anordnung in einer Ablaufabelle

konstruktive Koordinierung ↪ Interprozesskommunikation

- direktes Aktivieren/Erzeugen von Ausführungssträngen
- konsumierbare Betriebsmittel: Semaphore, Nachrichten, ...

Ablaufplanung berücksichtigt gerichtete Abhängigkeiten

- Anpassung von Auslösezeit und Terminen

VII Zugriffskontrolle

Konkurrenz und Koordination nebenläufiger Aktivitäten

- Nebenläufigkeit, Kausalität, Kausalordnung
- Konfliktsituationen ↪ synchronisieren ohne Prioritätsumkehr

Verdrängungssteuerung ↪ verdrängungsfreie kritische Abschnitte

- benötigt kein à priori Wissen; Verklemmungsvorbeugung
- pragmatisch/effektiv, beeinträchtigt unabhängige Jobs

Prioritätsvererbung ↪ Priorität zeitweise erhöhen

- benötigt kein à priori Wissen
- direkte Blockierung, Blockierung durch Vererbung; transitiv

Prioritätsobergrenzen ↪ Priorität zeitweise deckeln

- benötigt à priori Wissen; Verklemmungsvorbeugung
- Grundmodell vs. (einfachere) stapelorientierte Variante

Ablaufplanung berücksichtigt Blockadezeiten

- Verzicht auf den Prozessor ermöglicht wiederholte Blockierung

VIII Verteilte Echtzeitsysteme

Erscheinungsform \mapsto **verteiltes Echtzeitrechnungssystem**

- Orientierung an der Verteilung des physikalischen Objekts
- Beherrschung von Komplexität, Erhöhung der Rechenleistung

Anforderungen an verteilte Echtzeitrechnungssysteme

- **Zusammensetzbarkeit** \mapsto Kompositionsproblem, Gerätegruppen
- **Skalierbarkeit** \mapsto Erweiterbarkeit (auch Schrumpfung), Komplexität
- **Verlässlichkeit** \mapsto Fehlereingrenzung, Replikation

Aufbau verteilter Echtzeitrechnungssysteme

- Knotenrechner, **Netzwerkschnittstelle**, **Kommunikationssteuerung**
- **externe vs. autonome Kontrolle**; **Ereignis-** vs. **Zustandsnachrichten**
- Netzübergangsknoten, Netzwerktopologien, Netzwerkföderation

Kommunikationssysteme für verteilte Echtzeitsysteme

- **Kapselung des Zeitverhaltens**, Verpflichtungen der Klienten
- **Ereignis- vs. Zeitsteuerung**, explizite vs. implizite **Flusskontrolle**
- Netzzugangsprotokolle: CSMA/CR vs. TDMA

VII Exkurs: WCET-Analyse

WCET-Analyse \leadsto „Die Suche nach dem e!“

- WCET ist unverzichtbare Eingabe für die Planbarkeitsanalyse
- Programmiersprachenebene vs. Maschinenprogrammenebene

Werkzeugkette für die statische WCET-Analyse

Pfadanalyse \leadsto findet auf Programmiersprachenebene statt

- **Timing Schema** verwendet den abstrakten Syntaxbaum
- **IPET** basiert auf dem Kontrollflussgraphen
 - Erzeugung eines **T-Graphen**, Ableitung eines **Flussproblems**
 - \leadsto ganzzahliges lineares Optimierungsproblem

Hardware-Analyse \leadsto Eingaben für die WCET-Berechnung

- Hauptaufgaben: **Cache-** und **Pipeline-Analyse**
- Beispiel: Datenflussanalyse für 4-fach assoziativen LRU-Cache
 - must-Approximation und may-Approximation

messbasierte WCET-Analyse \leadsto ein kleiner Fingerzeig!