Adaptive Computing

(... via Timing Error Avoidance)

Gus Uht
(w/ Rick Vaccaro)
Dept. of Electrical and Computer Engineering

Copyright © 2003-5, A. K. Uht, R. J. Vaccaro
Patent applied for.
Background

• Three typical goals (alone or combined):
  1. ~Maximize performance.
  2. ~Minimize power consumption.
  3. **Adapt** to current (or prior) conditions:
     a. **Environmental**, e.g., temperature. (current)
     b. **Operational**, e.g., power supply voltage. (current)
     c. **Manufacturing**, e.g., slower or faster chips. (prior)

• ~All adaptive methods applicable to all.
• “Better-than-worst-case design”
Outline

1. Adaptive Methods
2. Timing Error Toleration
3. Timing Error Avoidance
4. TEAtime – Prototype and Data
5. Adaptive Control Systems
6. TEAPC & TEA42
7. Demo – TEA42
8. Summary
Adaptive System Methods

• Timing Error *Toleration*:
  – Keep changing something (freq., volt.), then:
  – *Let error occur, then recover*.
  – Change the thing back, repeat.

• Timing Error *Avoidance*:
  – Keep changing something (freq., volt.), then:
  – *Stop just before error would occur*.
  – Change the thing back, repeat.
Timing Error Tolerance

- **Operation:**
  1. *Speed up clock* (resp. *reduce voltage*) until error occurs.
  2. *Slow down clock* (resp. *increase voltage*) for no error.
  4. *Repeat: GOTO 1.*

- **Examples:**
  - Uht 2000: *TIMERRTOL*: performance ~maximized; not pursued.

- **Plusses:**
  - True (or better than) perf. max., resp. power min. [*within* cycle]

- **Minuses:**
  - Can be costly (TIMERRTOL: 2x cost; Razor: little extra cost)
  - Is complex and hard to design.
  - Can lead to *increase* in cycles → perf. may NOT be true max.
**Timing Error Avoidance**

- **Operation:**
  1. *Speed up clock* (resp. *reduce voltage*) ‘til *just before* error occurs.
  2. *Slow down clock* (resp. *increase voltage*) for no error.
  3. Repeat: GOTO 1.  
     [No backtracking or repair needed.]

- **Examples:**
  - Uht 2003: *TEAtime*: ~performance maximization; prototype built.
    - One-bit wide slowest-path test logic always errs before real logic does.

- **Plusses:**
  - Close to true performance maximization, resp. power minimization.
  - Simple, easy to design and build. (TEAtime)
  - No increase in cycles.

- **Minuses:**
  - Shorter papers.  
    *(no, wait, that’s a plus...)*
Frequency & DAC setting vs. time (Temp. = 22° C.)
TEAtime Summary

- Performance of prototype improves by \( \geq 34\% \)
- TEAtime provides adaptable frequency control of any synchronous digital system
- Always \(~\)maximizes performance
- Very cheap
- Very easy to add to existing or future designs
- \(~\)Can be adapted to physically existing systems
- \textbf{It Works!!}
Adaptive Control Systems

- **TEAtime:**
  - Simple up/down control system – 0 delay.

- **Skadron, Bahar:**
  - Classic feedback control system.
  - In hardware.

- **TEAPC, TEA42:**
  - Big delays; used as opportunity, not problem.
  - State-space feedback control system.
  - In software; little overhead.
TEAPC, TEA42 Goals

• Motivating Goals:
  – Realize TEAtime characteristics in a real computer
    • Adaptive computing
    • Improved performance – “Better-than-Worst-Case”

• Additional Goals:
  1. Workload adaptation
  2. Reduced power consumption
  3. Improved reliability
  4. Disaster tolerance (always enabled)
  5. …and all in a real machine

• BUT: can’t redesign or build Pentium 4’s
• SO: use real IBM/Intel-standard PC
Current Control System

- Only input is CPU temperature (feedback line).
- Primary output is CPU frequency (N).
  [sometimes: $V_{core} = f'(N)$]

```
K_{NT} \left[ \frac{N}{\circ K} \right] \quad \Delta \circ K \quad \Delta N \quad \Delta N \quad N \quad N

K_G \quad \frac{1}{s} \quad \Sigma \quad \Sigma \quad \frac{K_{fN}}{\text{freq}/N}

K_{fN} \quad \frac{\text{freq}/N}{s + A}

B = 0.0364 \quad A = 0.03345

8.26e6 \quad 6.900e-9

K_{fN} \quad \frac{\text{freq}/N}{s + A}

\frac{K_Tf}{\circ C/\text{freq}}
```

Clock Synthesizer, CPU and CPU internal sensor

$T_{set}$ (°C)
TEAxx System Features

• **Hardware:**
  – No modifications → all COTS parts.
  – System: ~3.0 GHz Pentium 4; 800/533 MHz FSB; Intel chipset; 1 GB RAM.

• **Software (teapcwin program):**
  – Realizes feedback control system.
  – Standard MS Windows application.
  – (Standard OS: W2K SP4; no modifications [of course].)
  – Small: 1.1 megabytes.
  – Fast: < 1% CPU utilization (without display).
freq., Vcore unlinked < Load Adaptation > freq., Vcore linked
TEA42 Disaster Tolerance

• Example: CPU Fan dies….

• Changes (automatic, via feedback system):
  – Freq: 3.15 GHz $\rightarrow$ 1.5 GHz
  – Vcore: $\sim$1.5 V. $\rightarrow$ $\sim$1.1 V.
  $\rightarrow$ Power: $\sim$150 W. $\rightarrow$ $\sim$90 W. (40% savings)

• CPU temperature stabilizes at safe value (with this CPU).

• System still works.
...and now it's time for the:

DEMO
Overall Summary

• TEAtime, TEAPC & TEA42 realize:
  2. *Adaptive* operation to both environment and/or loading.
  3. Low-power, high-reliability operation.
  4. Disaster tolerance.

• Feedback-control great for a system, too.
• *Adaptive systems* are the way to go.
• *They Work!*
Pointers…

- My website: [www.ele.uri.edu/~uht](http://www.ele.uri.edu/~uht)
- Or µURI website: [www.ele.uri.edu/muri](http://www.ele.uri.edu/muri)
Adaptive Computing
(... via Timing Error Avoidance)

Gus Uht
(w/ Rick Vaccaro)
Dept. of Electrical and Computer Engineering

Copyright © 2003-5, A. K. Uht, R. J. Vaccaro
Patent applied for.

BARC: January 21, 2005
Appendix

21. TEAtime Block Diagram.
22. TEAPC Block Diagram.
23. TEAPC Components.
24. [TEAPC] Experiment Setup.
TEAtime Block Diagram

- Timing Error Avoidance system
  - Blue: TEAtime hardware
  - Green: on FPGA

NOTES: - w, x >> 1 ; - DAC: Digital-to-Analog Convertor ; - VCO: Voltage-Controlled Oscillator.
TEAPC Block Diagram

CPU
Intel P4

Northbridge
Intel 875P

Main Memory
1 GB Dual Channel
400 MHz
Ultra

FSB

Vcore

VID

CPU Vcore
Regulator
Control

POWER SUPPLY

Southbridge
I/O
Controller
Intel ICH5R

SMBUS - IIC Bus

LPC Bus

Super I/O
ITE 8712F

CPU Clock Memory Clock

CPU Clock

SMBUS - IIC Bus

Clock Synthesizer
ICS 952635

Memory Clock

CPU Vcore
Volt.

CPU Fan Speed

CPU core Temp.

CPU Vcore Volt.

Only directly relevant components and connections are shown.

(FSB - Front Side Bus) (LPC - Low Pin Count)
## TEAPC Components

<table>
<thead>
<tr>
<th>PC Component</th>
<th>Manufacturer</th>
<th>Part Number/Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Motherboard</td>
<td>Gigabyte</td>
<td>GA-8KNXP (Rev. 2); w/DPS regulator</td>
</tr>
<tr>
<td>CPU</td>
<td>Intel</td>
<td>P4 3.0 GHz, 800 MHz bus</td>
</tr>
<tr>
<td>Chipset</td>
<td>Intel</td>
<td>875P, ICH5R</td>
</tr>
<tr>
<td>Clock Synthesizer</td>
<td>ICS</td>
<td>ICS952635</td>
</tr>
<tr>
<td>Super I/O (Environment Mon.)</td>
<td>ITE</td>
<td>IT8712F V0.6</td>
</tr>
<tr>
<td>CPU Volt. Regulator Control</td>
<td>ITE</td>
<td>IT8206R V0.1</td>
</tr>
<tr>
<td><strong>Main Memory</strong></td>
<td>Ultra</td>
<td>U10-5903R; 2 x 512 MB; 400 MHz DDR, Dual Channel (Operated at 320 MHz.)</td>
</tr>
<tr>
<td>Operating System</td>
<td>Microsoft</td>
<td>Windows 2000 SP4, HT disabled</td>
</tr>
<tr>
<td>Disk System – RAID 0+1</td>
<td>ITE</td>
<td>GigaRAID IT8212F</td>
</tr>
<tr>
<td>Disks</td>
<td>Maxtor</td>
<td>4 x 6E040L0, 40 GB, 133MHz IDE</td>
</tr>
<tr>
<td>Equipment for experiments only</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fan Controller &amp; Temp. Mon.</td>
<td>Thermaltake</td>
<td>Hardcano 12; for 4 fans, 4 thermocouples</td>
</tr>
<tr>
<td>Power Meter</td>
<td>Electronic</td>
<td>watts up? PRO</td>
</tr>
<tr>
<td></td>
<td>Educational</td>
<td>(Note: this is the unit’s model name.)</td>
</tr>
<tr>
<td></td>
<td>Devices</td>
<td></td>
</tr>
<tr>
<td>CPU Fan Controller</td>
<td>custom</td>
<td>On/Off, control sel. (MOBO or Hardcano)</td>
</tr>
</tbody>
</table>
Experiment Setup