Embedded Systems Design: A Unified Hardware/Software Introduction

Chapter 1: Introduction

Embedded systems overview

• Computing systems are everywhere
• Most of us think of “desktop” computers
  – PC’s
  – Laptops
  – Mainframes
  – Servers
• But there’s another type of computing system
  – Far more common...

Outline

• Embedded systems overview
  – What are they?
• Design challenge – optimizing design metrics
• Technologies
  – Processor technologies
  – IC technologies
  – Design technologies
A “short list” of embedded systems

<table>
<thead>
<tr>
<th>Embedded Systems Design: A Unified Hardware/Software Introduction, (c) 2000 Vahid/Givargis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Antilock brakes</td>
</tr>
<tr>
<td>Auto-focus cameras</td>
</tr>
<tr>
<td>Automatic toll machines</td>
</tr>
<tr>
<td>Automatic toll systems</td>
</tr>
<tr>
<td>Automatic transmission</td>
</tr>
<tr>
<td>Avionic systems</td>
</tr>
<tr>
<td>Battery chargers</td>
</tr>
<tr>
<td>Car computers</td>
</tr>
<tr>
<td>Cell-phones</td>
</tr>
<tr>
<td>Cell-phone base stations</td>
</tr>
<tr>
<td>Cordless phones</td>
</tr>
<tr>
<td>Cruise control</td>
</tr>
<tr>
<td>Curbside check-in systems</td>
</tr>
<tr>
<td>Digital cameras</td>
</tr>
<tr>
<td>Disk drives</td>
</tr>
<tr>
<td>Electronic card readers</td>
</tr>
<tr>
<td>Electronic instruments</td>
</tr>
<tr>
<td>Electronic toy/games</td>
</tr>
<tr>
<td>Factory control</td>
</tr>
<tr>
<td>Fan machines</td>
</tr>
<tr>
<td>Fingerprint identifiers</td>
</tr>
<tr>
<td>Home security systems</td>
</tr>
<tr>
<td>Life-support systems</td>
</tr>
<tr>
<td>Medical testing systems</td>
</tr>
</tbody>
</table>

And the list goes on and on

Some common characteristics of embedded systems

- Single-functioned
  - Executes a single program, repeatedly
- Tightly-constrained
  - Low cost, low power, small, fast, etc.
- Reactive and real-time
  - Continually reacts to changes in the system’s environment
  - Must compute certain results in real-time without delay

Design challenge – optimizing design metrics

- Obvious design goal:
  - Construct an implementation with desired functionality
- Key design challenge:
  - Simultaneously optimize numerous design metrics
- Design metric
  - A measurable feature of a system’s implementation
  - Optimizing design metrics is a key challenge

An embedded system example -- a digital camera

- Single-functioned -- always a digital camera
- Tightly-constrained -- Low cost, low power, small, fast
- Reactive and real-time – only to a small extent
Design challenge – optimizing design metrics

- Common metrics
  - Unit cost: the monetary cost of manufacturing each copy of the system, excluding NRE cost
  - NRE cost (Non-Recurring Engineering cost): The one-time monetary cost of designing the system
  - Size: the physical space required by the system
  - Performance: the execution time or throughput of the system
  - Power: the amount of power consumed by the system
  - Flexibility: the ability to change the functionality of the system without incurring heavy NRE cost

Design metric competition -- improving one may worsen others

- Expertise with both software and hardware is needed to optimize design metrics
  - Not just a hardware or software expert, as is common
  - A designer must be comfortable with various technologies in order to choose the best for a given application and constraints

Time-to-market: a demanding design metric

- Time required to develop a product to the point it can be sold to customers
- Market window
  - Period during which the product would have highest sales
- Average time-to-market constraint is about 8 months
- Delays can be costly
NRE and unit cost metrics

- Costs:
  - Unit cost: the monetary cost of manufacturing each copy of the system, excluding NRE cost
  - NRE cost (Non-Recurring Engineering cost): The one-time monetary cost of designing the system
    - total cost = NRE cost + unit cost * # of units
    - per-product cost = total cost / # of units
    = (NRE cost / # of units) + unit cost
- Example
  - NRE=$2000, unit=$100
  - For 10 units
    - total cost = $2000 + 10*$100 = $3000
    - per-product cost = $2000/10 + $100 = $300

Amortizing NRE cost over the units results in an additional $200 per unit

NRE and unit cost metrics

- Compare technologies by costs -- best depends on quantity
  - Technology A: NRE=$2,000, unit=$100
  - Technology B: NRE=$30,000, unit=$30
  - Technology C: NRE=$100,000, unit=$2

- But, must also consider time-to-market

Losses due to delayed market entry

- Simplified revenue model
  - Product life = 2W, peak at W
  - Time of market entry defines a triangle, representing market penetration
  - Triangle area equals revenue
- Loss
  - The difference between the on-time and delayed triangle areas

- Area = 1/2 * base * height
  - On-time = 1/2 * 2W * W
  - Delayed = 1/2 * (W-D+W)*(W-D)
- Percentage revenue loss
  = ((on-time - delayed)/on-time)*100%
  = (D(3W-D)/2W^2)*100%
- Try some examples
  - Lifetime 2W=52 wks, delay D=4 wks
  - (4*(3*26 –4)/2*26^2) = 22%
  - Lifetime 2W=52 wks, delay D=10 wks
  - (10*(3*26 –10)/2*26^2) = 50%
  - Delays are costly!
The performance design metric

- Widely-used measure of system, widely-abused
  - Clock frequency, instructions per second – not good measures
  - Digital camera example – a user cares about how fast it processes images, not clock speed or instructions per second
- Latency (response time)
  - Time between task start and end
  - e.g., Camera’s A and B process images in 0.25 seconds
- Throughput
  - Tasks per second, e.g., Camera A processes 4 images per second
  - Throughput can be more than latency seems to imply due to concurrency, e.g., Camera B may process 8 images per second (by capturing a new image while previous image is being stored).
- Speedup of B over S = B’s performance / A’s performance
  - Throughput speedup = 8/4 = 2

Three key embedded system technologies

- Technology
  - A manner of accomplishing a task, especially using technical processes, methods, or knowledge
- Three key technologies for embedded systems
  - Processor technology
  - IC technology
  - Design technology

Processor technology

- The architecture of the computation engine used to implement a system’s desired functionality
- Processor does not have to be programmable
  - “Processor” not equal to general-purpose processor

General Purpose vs. Special Purpose

- Standard tradeoff

Embedded Systems Design: A Unified Hardware/Software Introduction, (c) 2000 Vahid/Givargis
Processor technology

- Processors vary in their customization for the problem at hand

```
for i = 1 to N loop
  total += M[i]
end loop
```

Datapath
- Controller
- Control logic and State register
- General ALU

Data memory
- Program memory
- Assembly code

```
total = 0
for i = 1 to N loop
```

Embedded Systems Design: A Unified Hardware/Software Introduction, (c) 2000 Vahid/Givargis

General-purpose processors

- Programmable device used in a variety of applications
  - Also known as “microprocessor”
- Features
  - Program memory
  - General datapath with large register file and general ALU
- User benefits
  - Low time-to-market and NRE costs
  - High flexibility
  - “Pentium” the most well-known, but there are hundreds of others

```
Controller
Datapath
Register file
General ALU
Program memory
Data memory
```

Single-purpose processors

- Digital circuit designed to execute exactly one program
  - a.k.a. coprocessor, accelerator or peripheral
- Features
  - Contains only the components needed to execute a single program
  - No program memory
- Benefits
  - Fast
  - Low power
  - Small size

```
Controller
Datapath
Control logic
Index
State register
Total
Data memory
```

Application-specific processors

- Programmable processor optimized for a particular class of applications having common characteristics
  - Compromise between general-purpose and single-purpose processors
- Features
  - Program memory
  - Optimized datapath
  - Special functional units
- Benefits
  - Some flexibility, good performance, size and power

```
Controller
Datapath
Control logic and State register
Custom ALU
Program memory
Data memory
```

Embedded Systems Design: A Unified Hardware/Software Introduction, (c) 2000 Vahid/Givargis
IC technology

• The manner in which a digital (gate-level) implementation is mapped onto an IC
  – IC: Integrated circuit, or “chip”
  – IC technologies differ in their customization to a design
  – IC’s consist of numerous layers (perhaps 10 or more)
    • IC technologies differ with respect to who builds each layer and when
    • Bottom layer = transistors
    • Middle layer = logic components
    • Top layer = connect components with wires

Full-custom/VLSI

• All layers are optimized for an embedded system’s particular digital implementation
  – Placing transistors
  – Sizing transistors
  – Routing wires
• Benefits
  – Excellent performance, small size, low power
• Drawbacks
  – High NRE cost (e.g., $300k), long time-to-market

Semi-custom

• Lower layers are fully or partially built
  – Designers are left with routing of wires and maybe placing some blocks
• Benefits
  – Good performance, good size, less NRE cost than a full-custom implementation (perhaps $10k to $100k)
• Drawbacks
  – Still require weeks to months to develop
PLD (Programmable Logic Device)

- All layers already exist
  - Designers can purchase an IC
  - Connections on the IC are either created or destroyed to implement desired functionality
  - Field-Programmable Gate Array (FPGA) very popular
- Benefits
  - Low NRE costs, almost instant IC availability
- Drawbacks
  - Bigger, expensive (perhaps $30 per unit), power hungry, slower

Moore’s law

- The most important trend in embedded systems
  - Predicted in 1965 by Intel co-founder Gordon Moore
    \[
    \text{IC transistor capacity has doubled roughly every 18 months for the past several decades}
    \]

Graphical illustration of Moore’s law

- Wow
  - This growth rate is hard to imagine, most people underestimate
  - How many ancestors do you have from 20 generations ago
    - i.e., roughly how many people alive in the 1500’s did it take to make you?
    - \(2^{20} = \) more than 1 million people
  - (This underestimation is the key to pyramid schemes!)

- Something that doubles frequently grows more quickly than most people realize!
  - A 2002 chip could hold about 15,000 1981 chips inside itself
  - Designers must work quickly to use all these transistors
Design Technology - Top Down Approach

- The manner in which we convert our concept of desired system functionality into an implementation

The co-design ladder

- In the past:
  - Hardware and software design technologies were very different
  - Recent maturation of synthesis enables a unified view of hardware and software
- Hardware/software “codesign”

The choice of hardware versus software for a particular function is simply a tradeoff among various design metrics, like performance, power, size, NRE cost, and especially flexibility; there is no fundamental difference between what hardware or software can implement.

Design productivity exponential increase

- Exponential increase over the past few decades

Independence of processor and IC technologies

- Basic tradeoff
  - General vs. custom
  - With respect to processor technology or IC technology
  - The two technologies are independent

General, providing improved:
  - Flexibility
  - Maintainability
  - NRE cost
  - Time-to-prototype
  - Time-to-market
  - Cost (low volume)

General-purpose processor

ASIP

Single-purpose processor

Customized, providing improved:
  - Power efficiency
  - Performance
  - Size
  - Cost (high volume)

PLD

Semi-custom

Full-custom
Design productivity gap

- While designer productivity has grown at an impressive rate over the past decades, the rate of improvement has not kept pace with chip capacity.

The mythical man-month

- The situation is even worse than the productivity gap indicates.
- In theory, adding designers to the team reduces project completion time.
- In reality, the productivity per designer decreases due to the complexities of team management and communication.
- The software community is known as "the mythical man-month." (Brooks 1975)
- At some point, it can actually lengthen project completion time! (“Too many cooks”)

- 10^6 trans, 1 designer = 5000 trans/month
- Each additional designer reduces capacity by 100 trans/month
- So 2 designers produce 9000 trans/month each

The mythical man-month

- 1981 Leading edge chip required 100 designer months
- 10,000 trans / 100 trans/month
- 2002 Leading edge chip requires 30,000 designer months
- 150,000,000 trans / 5000 trans/month
- Designer cost increase from $1M to $300M - few products justify this cost

Summary

- Embedded systems are everywhere.
- Key challenge: optimization of design metrics
- Design metrics compete with one another.
- A unified view of hardware and software is necessary to improve productivity.
- Three key technologies
  - Processor: general-purpose, application-specific, single-purpose
  - IC: Full-custom, semi-custom, PLD
  - Design: Compilation/synthesis, libraries/IP, test/verification