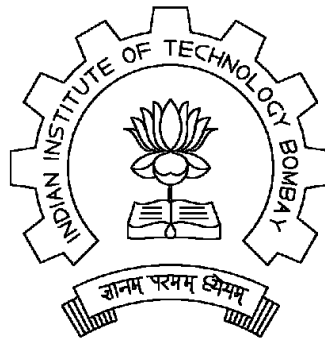


Embedded Systems (Software)

Prof. Kavi Arya



Embedded System Diversity

Beware of the computer!



- From computers to **embedded & networked SoCs ... IoT**
- Complete change in device interaction
- Growing number of **critical applications**

Common Design Metrics

- NRE (Non-recurring engineering) cost
- Unit cost
- Size (bytes, gates)
- Performance (execution time)
- Power (more power=> more heat & less battery time)
- Flexibility (ability to change functionality)
- Time to prototype
- Time to market
- Maintainability
- Correctness
- Safety (probability that system won't cause harm)

Apple “A” series SoC

- **Apple A4 (2010)**
 - for iPad ARM based SoC @1GHz w/integ. GPU (\$1Bn to devlp)
- **Apple A5 (2012)**
 - Based on dual-core ARM Cortex-A9 MPCore CPU
 - **\$4B development** facility by Samsung in Texas
 - Clocked at 1 GHz (auto adj. frequency to save battery)
 - ISP for face detection, wh.balance & automatic image stabilization
 - "EarSmart" unit from Audience for noise canceling
 - CPU portion 2x as powerful as the original iPad
 - GPU up to 7x as powerful A4
 - Cost 75% more than predecessor
- **Apple A6x (2013)**
 - 1.4 GHz Apple-designed [ARMv7](#) based [dual-core CPU](#) (Swift)
 - Integrated quad-core [PowerVR](#) SGX 554MP4 GPU @300 MHz
 - 2x computing power + graphics perf. of previous [Apple A5X](#)
 - 32 nm process => chip is 123 mm² large^[6] (26% larger than A6).

(Apple) Processor Trends

| Year | Model | Specs | Product | Spd | Gfx | Size | Tech. |
|------|-----------|---|---------------------------|----------|----------|---|-------------|
| 2012 | A6 | 1.3 GHz ARMv7 based dual-core CPU (Swift) | iPhone5 | 2x | 2x | 22% smaller | 32nM |
| 2013 | A7 | 1.3–1.4GHz 64-bit ARMv8-A dual-core CPU (Cyclone) integrated PowerVR G6430 GPU, 31x64bit GP regs, 32x128bit FP regs | iPhone 5S, iPad Mini2 & 3 | 2x | 2x | 1B transistors in 102sq.mm | 28nM |
| 2014 | A8 | 1.4GHz 64-bit ARMv8-A dual-core CPU & integrated PowerVR GX6450 GPU | iPhone6 & 6+ | 25% | 50% | 13% less in size, 2B transistors 89sq.mm | 20nM |
| 2015 | A9 | 64-bit ARM based system on a chip (SoC) | iPhone 6S & 6S+ | 70% more | 90% more | | 14nM |

(Apple) Processor Trends

| Year | Model | Specs | Product | Spd | Gfx | Size | Tech. |
|------|------------|---|----------------|------------------------------|-------------------------------|--------------------------|-------------|
| 2017 | A11 | 64-bit ARMv8-A 6-core CPU (Bionic) 2 high perf and 4 high efficiency | iPhone 8 | 25% faster | 70% faster | 4.3B transistors | 10nM |
| 2018 | A12 | 64-bit ARM 2+4 core CPU (Bionic) | iPhone XS & XR | 35% faster | 95% faster multi core | 6.9B transistors | 7nM |
| 2019 | A13 | 64-bit ARM 6 core with 2 high perf cores running at 2.65GHz (Lightning) with ML accelerators – AMX blocks; & 4 energy efficient cores (Thunder) | iPhone 11 | 20% faster with 30% less pwr | 20% faster with 40% lower pwr | 8.5B transistors | 7nM |
| 2020 | A14 | Apple A14 Bionic (hexa-core 64-bit ARM64 "mobile SoC", SIMD, caches) | iPhone 12 | 40% faster | 30% faster | 11.8B transistors | 5nM |
| 2022 | A16 | Apple A16 Bionic 6-core 64-bit ARMv8.6A SoC 6 core Neural Engine | iPhone 14 | 40% faster | 50% faster | 16.0B transistors | 4nM |

Challenges

...Decreasing

Mission & Safety-Critical Pressures

Increasing...

- Tolerance for defects
- Development Cycles
- Resource availability
- Ability to manage reqs
- Ability to ensure long term maintenance

- Safety critical requirements in
 - **Aerospace & defence, Energy Transportation, Industrial, Medical**
- Requirement changes, life span
- Application complexity
- Cost of code testing, validation & verification & certification
- Need for systems & software design reusability

- Packaging & ergonomics are key
- Mechatronics
- Mass deployment – less scope for error

Current Technology

Extrapolation of traditional software techniques

- Programs written in conventional languages
 - C subsets, MISRA C for automobile
 - Java for telephone / smartcards
- Glued together by OS services
 - A wide variety of embedded OS (VxWorks, OSEK)
- With some reuse and standardisation effort
- Classical software models largely inadequate
 - Too powerful, hard to verify
 - often subsets of rich languages=>doesn't make them simpler

Why Is Embedded Software Not Just Software On Small Computers?

- **Embedded = Dedicated**
- **Interaction with physical processes**
 - Sensors, actuators, processes
- **Critical properties are not all functional**
 - Real-time, fault recovery, power, security, robustness
- **Heterogeneity**
 - Hardware/software tradeoffs, mixed architectures
- **Concurrency**
 - Interaction with multiple processes
- **Reactivity**
 - Operating at the speed of the environment



These features look more like hardware!

Current Bottleneck

- Intrinsic application complexity grows rapidly
 - Analog / digital interface: more objects to control
 - Embedded algorithmics: signal, display, alarm, power...
 - Richer hardware architecture: \square P, DSP, ASIP, ASIC, FPGA
 - Performance adds complexity
 - Footprint / power minimization interferes with logical design
 - Technology independence is still difficult
- => Verification bottleneck
- Applications hard to verify off-site
 - Hardware / software interaction difficult

Software Engineering

(or, how do we build reliable systems?)

Things Have to Change!

- Pressure on productivity of design engineers working on complex systems.
- Time has come to design hardware using software engineering - rather than hw engg - methodologies.
- Complexity of system is the basic problem, and Moore's Law doubles complexity every 18 months.
- Advances in software engineering help produce complex systems **with more easily available design skills**, making large profits
- **We expect new designers, with/without hardware design skills, will design hardware in future**

Designer Productivity

- “The Mythical Man Month” by Frederick Brooks '75
 - More designers on team => lower productivity because of increasing communication costs between groups
 - Consider 1M transistor project:
 - Say, a designer has productivity of 5000 transistor/mth
 - Each extra designer => decrease of 100 transistor/mth productivity in group due to comm. costs
- | | | |
|---------------|----------------------------|---------|
| – 1 designer | $1M/5000 \Rightarrow$ | 200mth |
| – 10 designer | $1M/(10*4100) \Rightarrow$ | 24.3mth |
| – 25 designer | $1M/(25*2600) \Rightarrow$ | 15.3mth |
| – 27 designer | $1M/(27*2400) \Rightarrow$ | 15.4mth |
- Need new design technology to shrink design gap

Design Productivity Gap

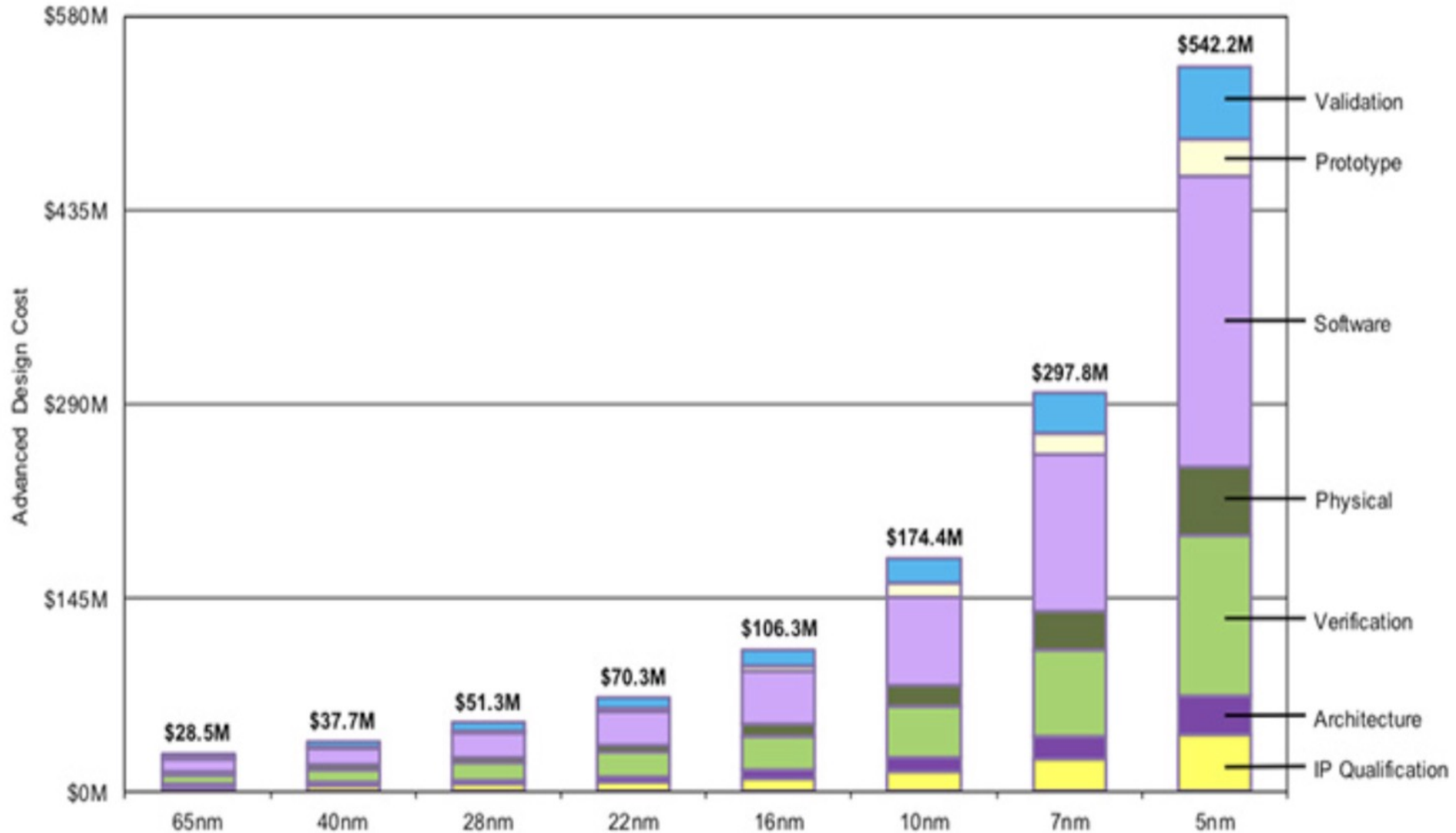
- Designer productivity grown over the last decade
- Rate of improvement has not kept pace with the chip-capacity growth
- 1981: leading edge chip:
 - 100 designers * 100 trans/mth => 10k trans complexity
- 2010: leading edge Intel chip using 45nm technology:
 - > 1B transistor complexity
- 2015: Leading edge Apple A9 using 14nm technology:
 - > 2B transistor complexity
- 2020: Apple A14 chip using 5nm technology:
 - > 11.8B transistor complexity
- 2023: Apple A16 Bionic chip (iPhone 14) using 3nm tech:
 - > 16-20B transistor complexity
- Designers at avg. of \$10k pm
=> cost of building leading edge chips has gone from \$1M ('81)-> \$300M (2002)-> \$1B (2010)-> \$5B (2020)-> \$10B+ (2023)



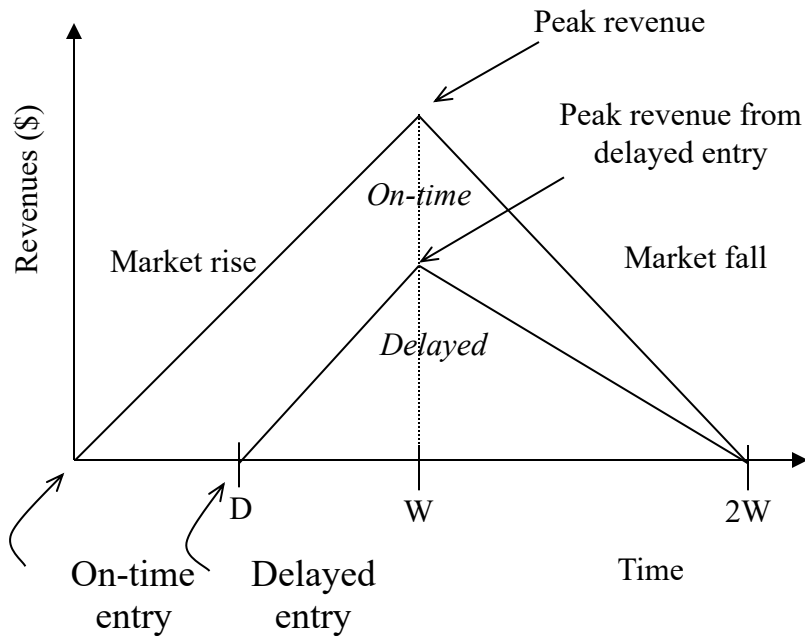
- **Need paradigm shift to cope with complexities**

Source: Embedded System Design: Frank Vahid/ Tony Vargis (John Wiley & Sons, Inc.2006) And others.

Chip Design Cost

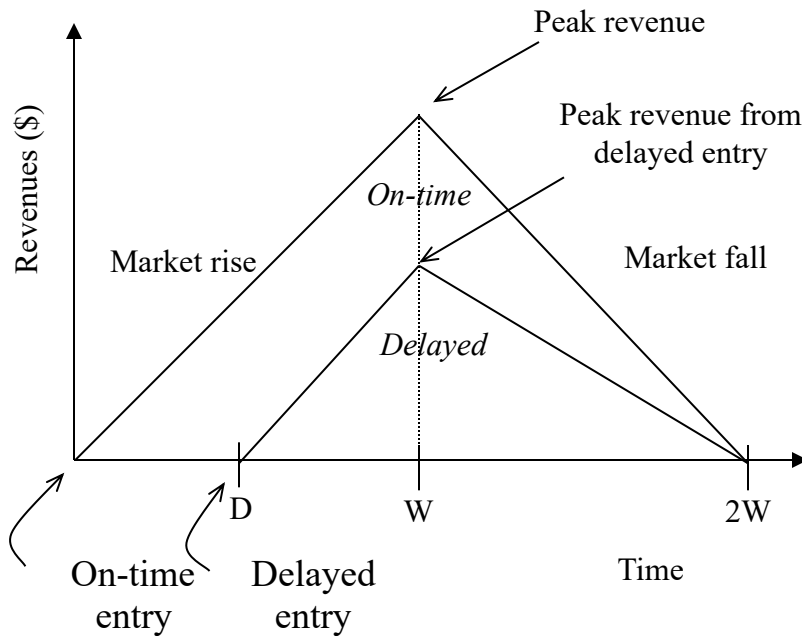


Time to Market Design Metric



- 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**
- Avg. time to market today = 8 mth
- 1 day delay may amount to \$Ms
 - **see Sony Playstation vs Xbox**

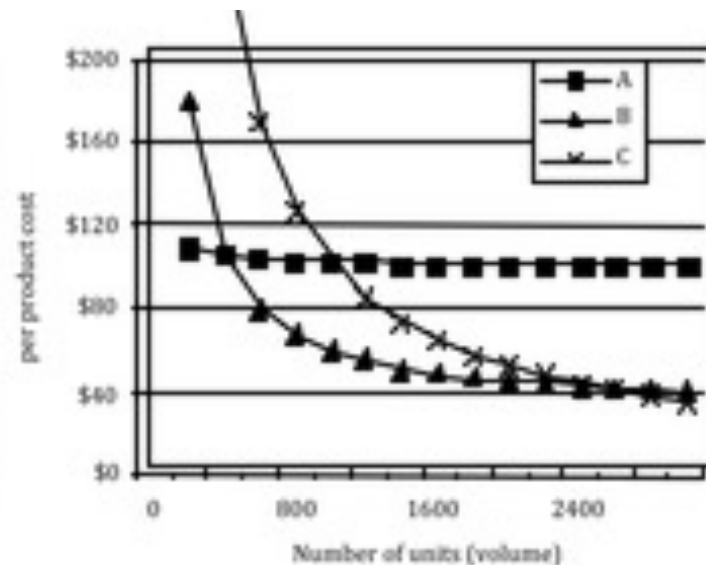
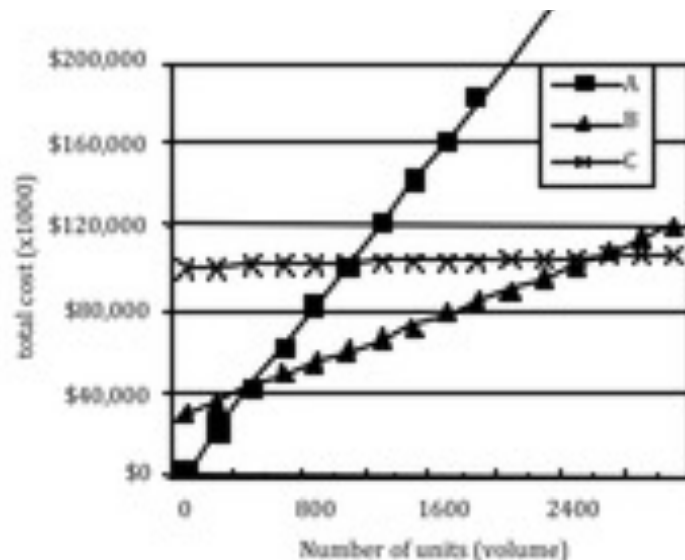
Losses due to delayed market entry



- Area = $1/2 * \text{base} * \text{height}$
 - On-time = $1/2 * 2W * W$
 - Delayed = $1/2 * (W-D+W)*(W-D)$
- Percentage revenue loss = $(D(3W-D)/2W^2)*100\%$
- 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!**

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

Trends (Moore's Law)

- **IC transistor capacity doubles every 18 mths**
 - 1981: leading edge chip had 10k transistors
 - 2002: leading edge chip has 150M transistors
 - 2010: Leading edge Intel processor has 0.5B trans in 107sq.mm
 - 2014: Leading edge Intel processor has 2.0B trans in 89sq.mm
 - 2019: Leading edge Intel processor has 8.5B trans in 83sq.mm
 - 2022: Apple A16 processor has 16B transistors
- **Why?**
 - Reducing transistor size, increasing chip size, clever circuits
 - Changing due to paradigm shifts: sys design tools, nanotech, ...

Trends (Designer Productivity)

- **Designer productivity improved due to better tools:**
 - Compilation/Synthesis tools
 - Libraries/IP
 - Test/verification tools
 - Standards
 - Languages and frameworks (Handel-C, Lava, Esterel, ...)
 - 1981: designer produced 100 transistors per month
 - 2002: designer produces 5000 transistors per month
 - 2022: designer produces ???

Software Engineering...

Why we need it?

Enemy number 1 : the bug

- Therac 25 : lethal irradiations
- Dharan's Patriot
- Ariane V
- Mars Explorer, Mars Polar Lander
- High-end automobile problems
- Pentium, SMP CPU networks
- Telephone and camera bugs



Bugs grow faster than Moore's law!



Other Important Issues

- Hardware / software partitioning

 - Hardware / software source code independence

 - Link between programming and performance analysis

- Operating Systems / scheduling

 - Well-studied field: rate-monotonic, earliest deadline first, etc.

 - Newer computation models need less explicit scheduling

- Fault tolerance

 - Software redundancy, voting algorithms, etc.

 - CRCs, TT networks

How to avoid or control bugs?

- Traditional : better verification by fancier simulation
- Next step : **better design** using **specific techniques**
 - Better and more reusable specifications
 - Simpler computation models, formalisms, semantics
 - Reduce architect / designer and customer / provider distance
 - Reduce hardware / software distance
- Requires **better tooling**
 - Higher-level models and synthesis
 - Formal property verification / program equivalence
 - Certified libraries

Anatomy of Embedded Applications

- **CC** : continuous control, signal processing
Differential equation solving, filters
Specs and simulation in Matlab / Scilab, manual or automatic code
- **FSM** : finite state machines, state transition systems
Discrete control, protocols, networking, drivers, security, etc.
Flat or hierarchical state machines, manual or automatic code
- **Calc** : calculation intensive
Navigation, security, etc.
C, manual + libraries
- **Web** : web-like navigation, audio / video streaming
Consumer electronics, infotainment systems
Data-flow networks, embedded Java

BMW 745i : Prelude To Complexity



Another Life Cycle
Example : **The
Software Error**

External view

The problem: software error, a desynchronization of the valvetronic motors



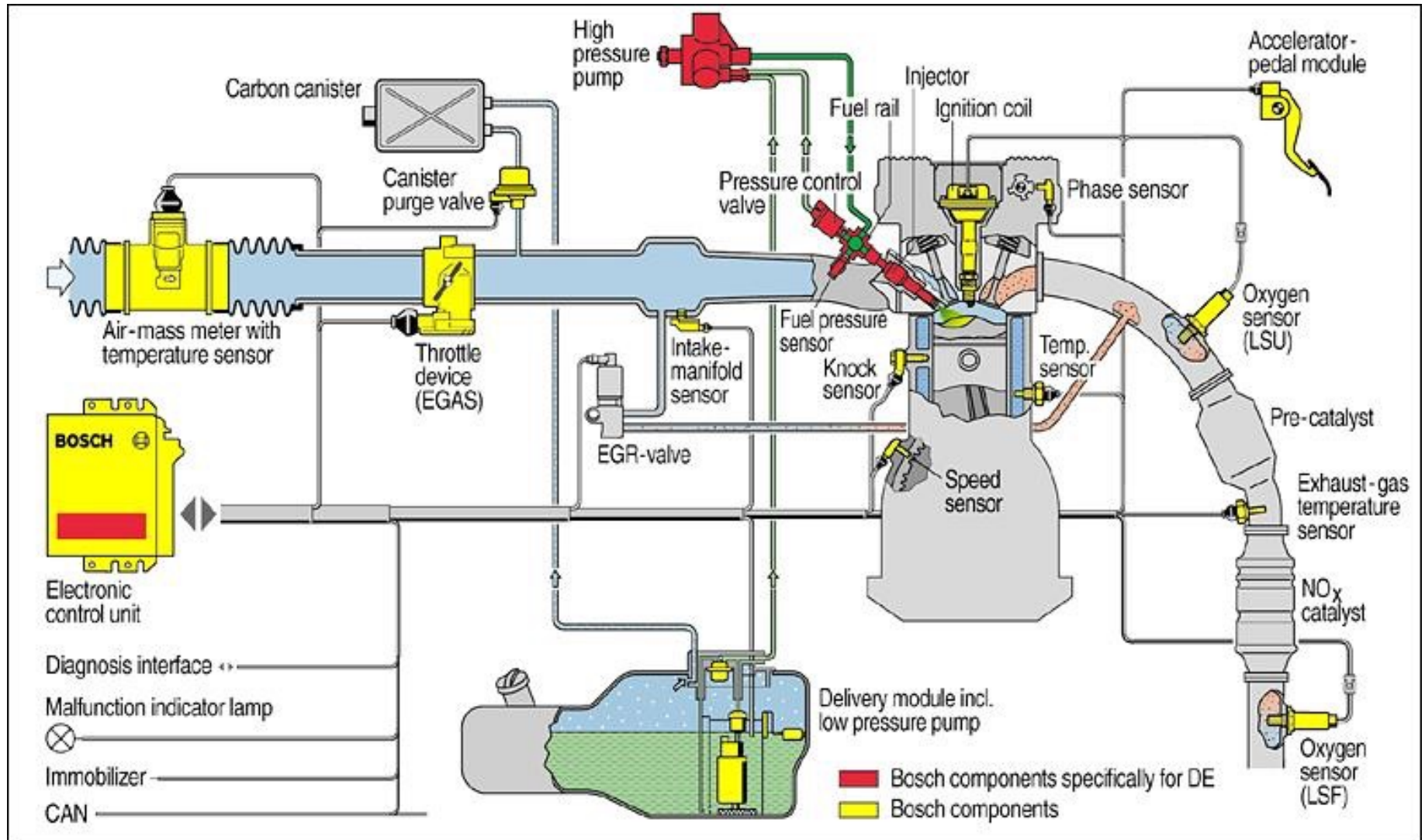
- Rough running engine, possibly stall
- Severity: 6 incidents in 5,470 cars with 2 rear endings
 - “alleged injury” of BMW passengers
 - Fault of drunk or inattentive following drivers

BMW Cost

- To repair: Reprogram ECU
- Recalls not uncommon in industry
 - BMW 5,470 cars @ \$68,500 = Rev \$372 mil
- Compare Cost: Recall BMW X5
 - 164,000 units @ \$66,800 = Rev \$10 bil.
 - ~\$5 Million
 - ~\$30 per SUV



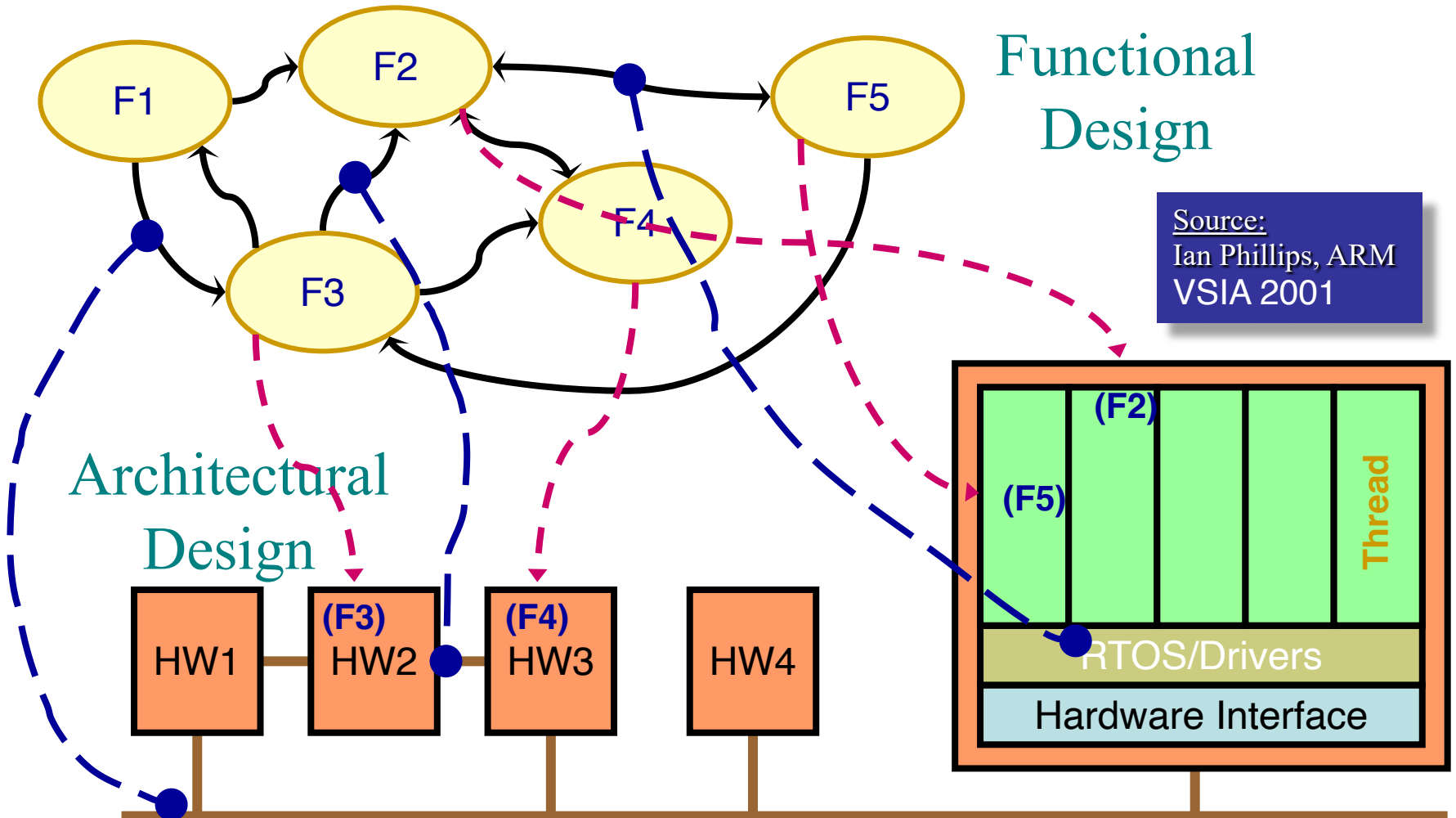
Bosch EMU For Four Wheeler (Multi Cylinder)



Design Issues

(How do we build these systems?)

Functional Design & Mapping



Synchronous languages

- Started in the 80's

Esterel : Ecole des Mines / INRIA, SyncCharts : U. Nice

Lustre : IMAG, Signal : INRIA Rennes

Lava : Chalmers, Xilinx

- Started in the mid-90's

- Handel-C: University of Oxford, Celoxica Inc.

- Industrial use in the 90's

Lustre / SCADE : nuclear plants (Schneider), avionics (Airbus)

Esterel : avionics (Dassault), telecom

=> Full development in the 2000's

avionics: Airbus, Dassault, Elbit, Eurocopter, SNECMA, Thales,...

automotive: AUDI, GM, PSA,...

hardware pilot projects / experiments: TI, STM, Xilinx, Intel, Thales

How do we get there?

KNOWLEDGE OR SKILLS REQUIRED

- **Design of Solutions**
- **Investigation**
- **Modern Tool Usage**
- **Individual & Team Work**
- **Communication**

Conclusion

- We have simultaneous optimisations of competing design metrics: speed, size, power, complexity, etc.
- Software engineering issues apply
 - Non-recurring engineering costs are critical
 - Design-productivity / time-to-market is paramount
- Traditional technologies unequipped to build complex embedded systems
 - Need unified view of hardware/ software co-design.
- Design focus at higher levels of abstraction =>
Abstract specs refined into programs
then into gates and logic