The highly efficient Imsys processor core

2012-09-10

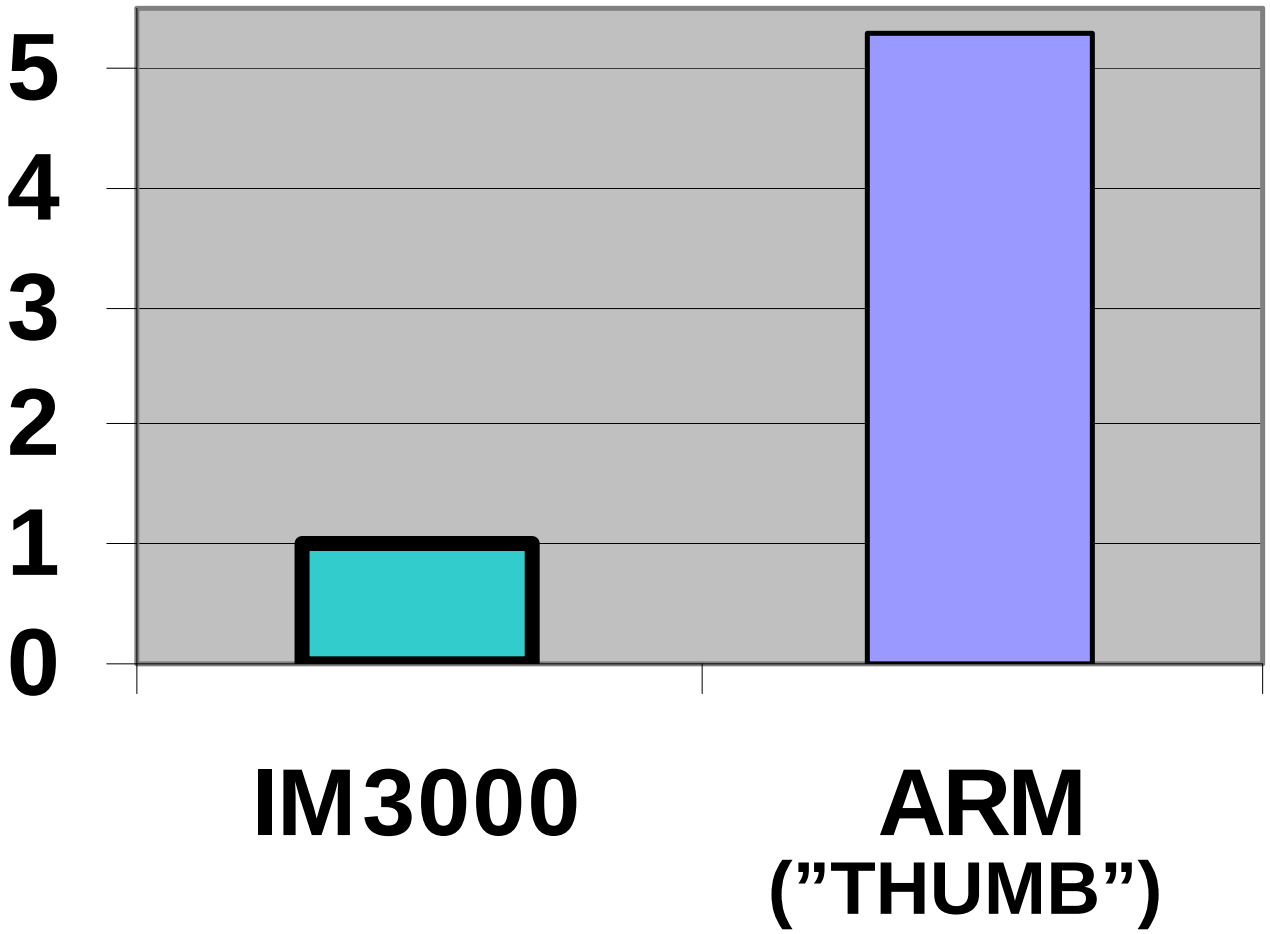
www.imsystech.com

Comparisons: Sources, Relevance

- Texas Instruments (TI) MSP430 is not really a competitor, since it cannot have large programs. However, comparing energy consumtion per benchmark with this processor is still relevant, since MSP430 is marketed as the most efficient of all architectures. TI's own benchmark programs have been used in the comparison of code size and efficiency for C code, and all non-Imsys results here have been published by TI *(SLAA205B – June 2005 – Revised July 2006. Available on the Internet.)*
- ARM7and ARM9 (and the corresponding newer Cortex M-3) is of particular interest since the ARM architecture is dominating in mobile phones and marketed as having energy and code size efficiency.
- Benchmarks for optimized DSP code were provided by a customer. Java comparison used benchmark programs published, along with the ARM9 results, by Systronix, Inc. *(Available on the Internet.)*

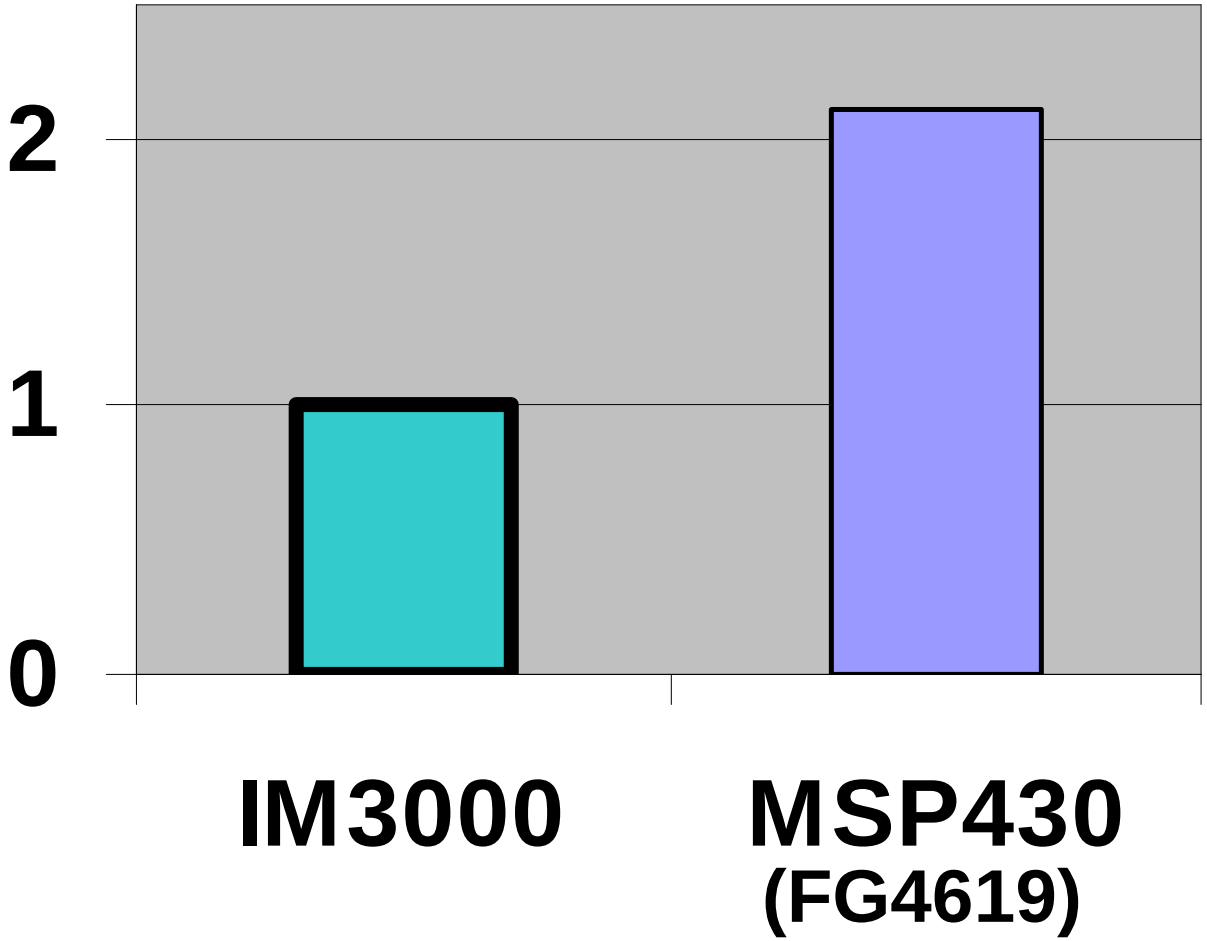
CODE SIZE

C CODE W/O OPTIMIZATION

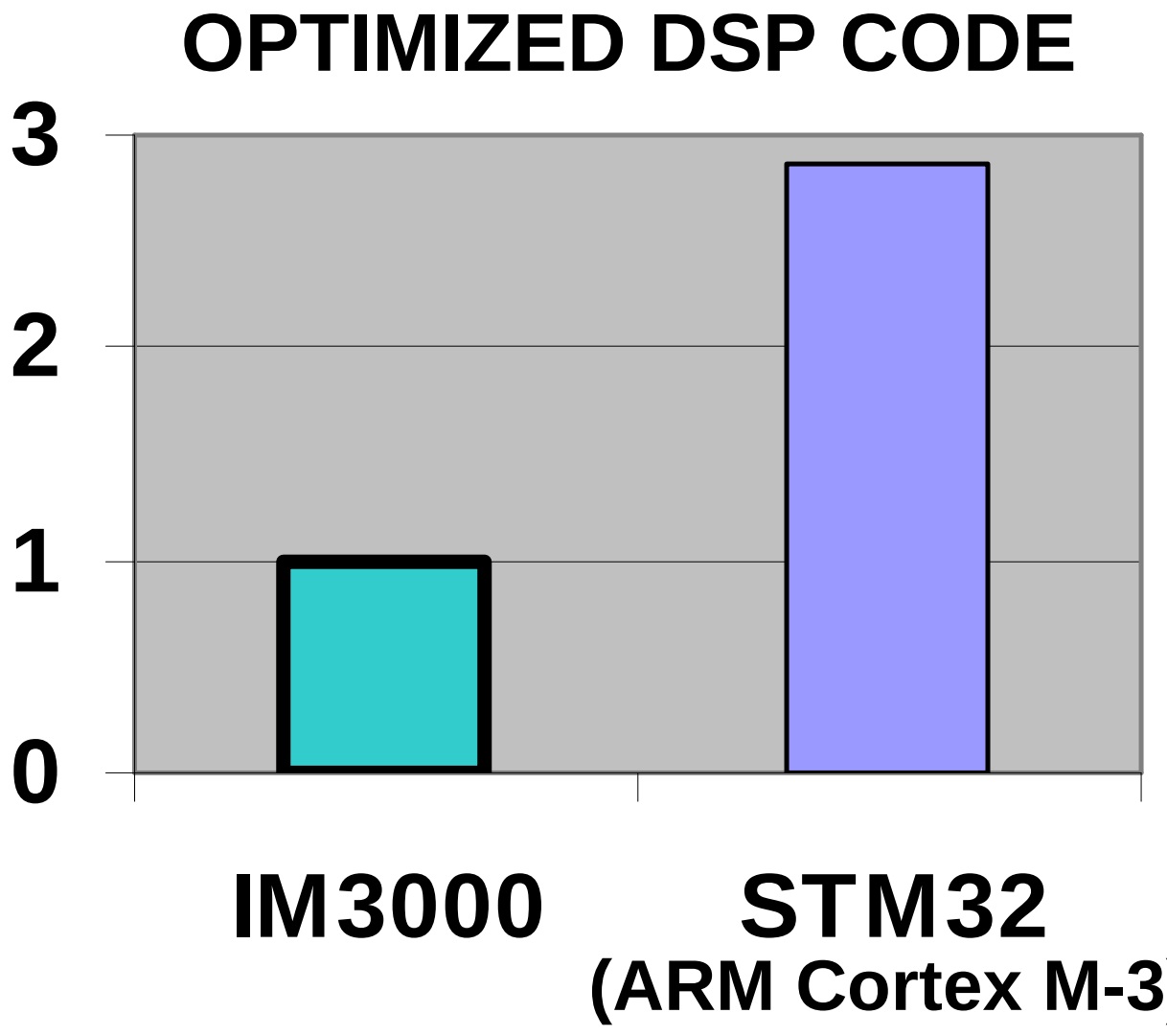


ENERGY CONSUMPTION

C CODE W/O OPTIMIZATION

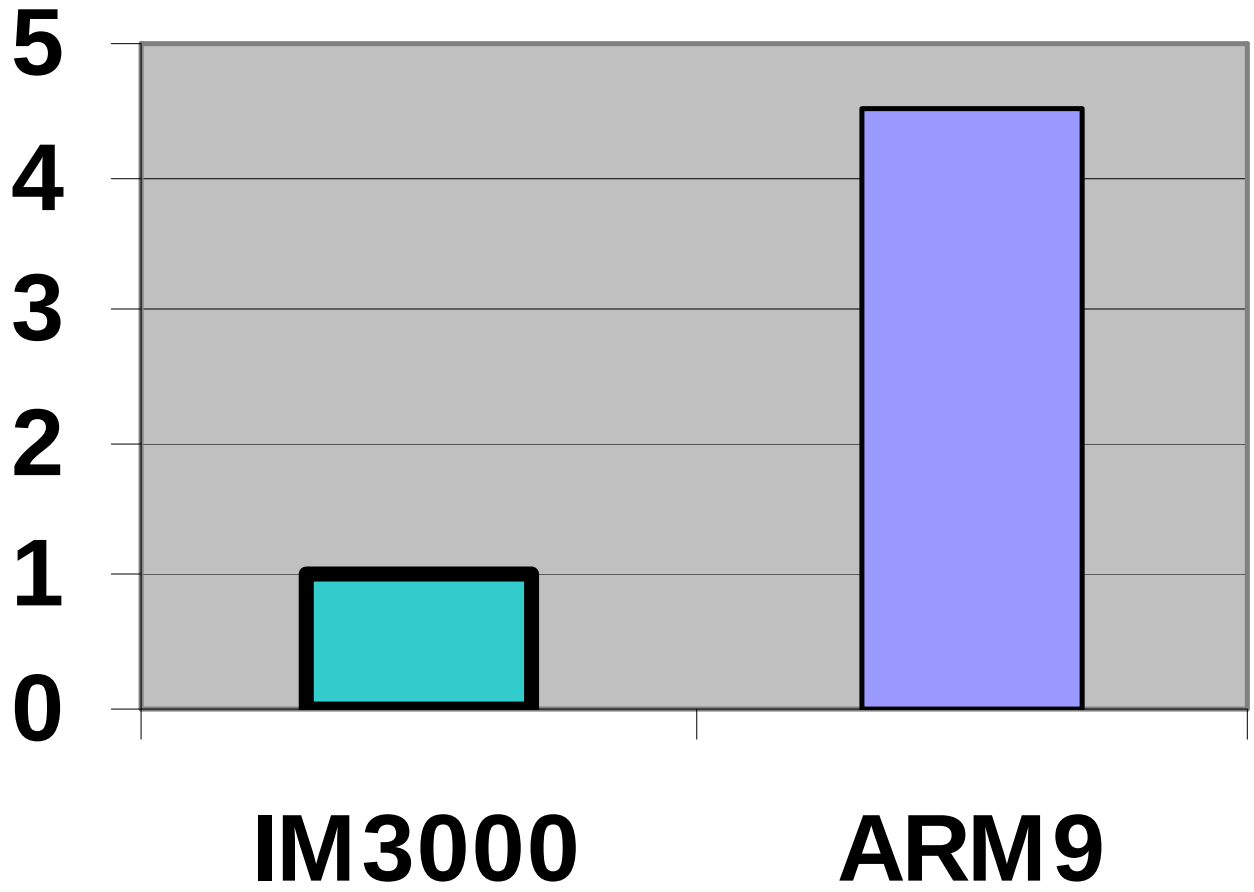


ENERGY CONSUMPTION



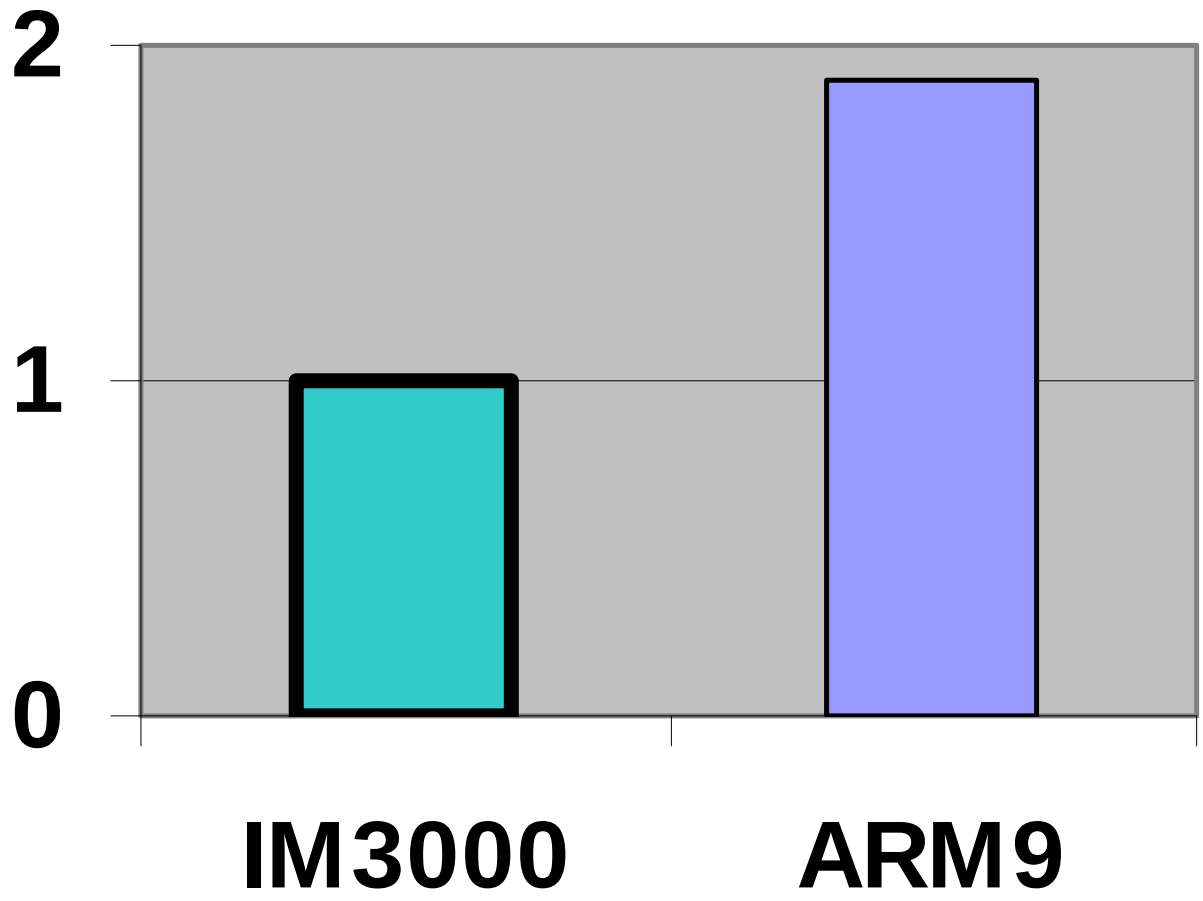
ENERGY CONSUMPTION

OPTIMIZED CRYPTO CODE (RSA,3DES)



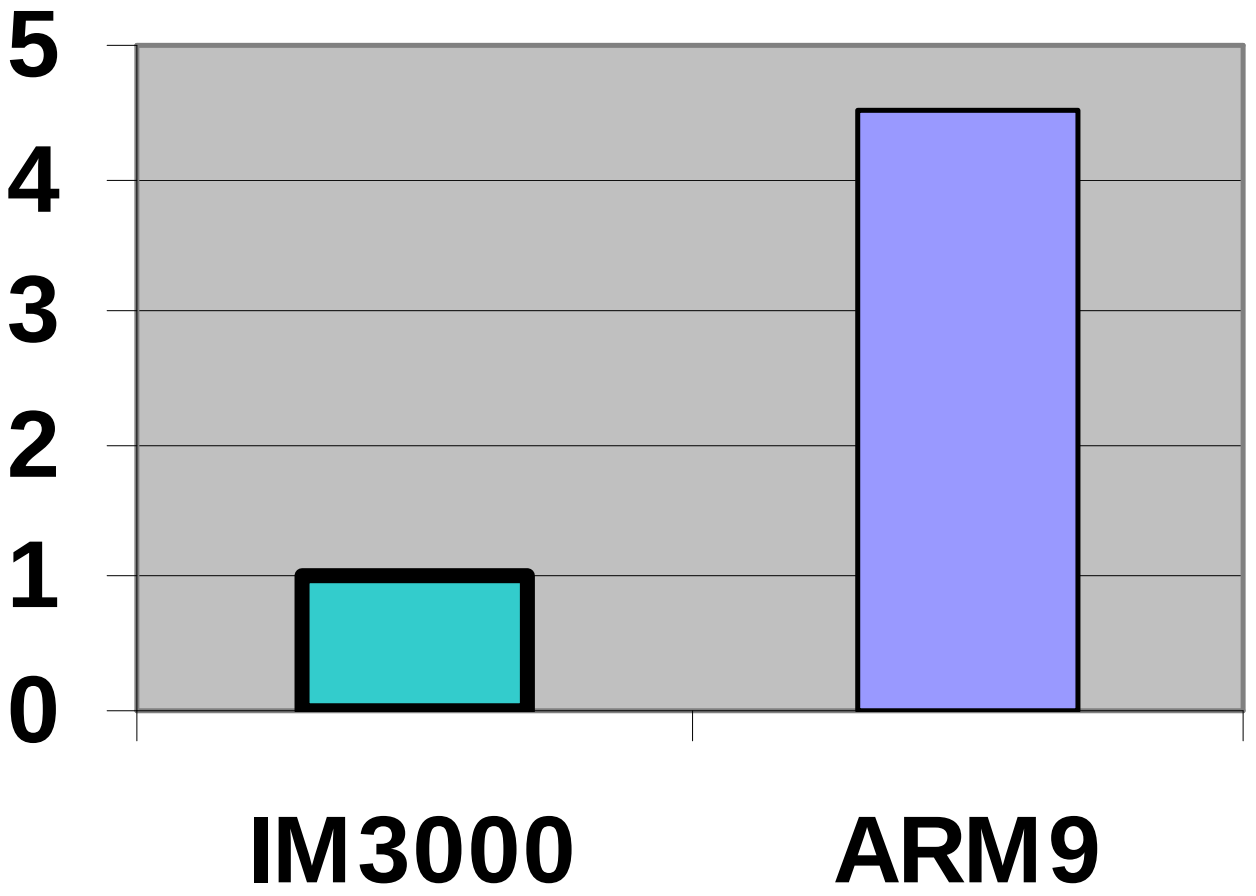
EXECUTION TIME

OPTIMIZED CRYPTO CODE (RSA,3DES)



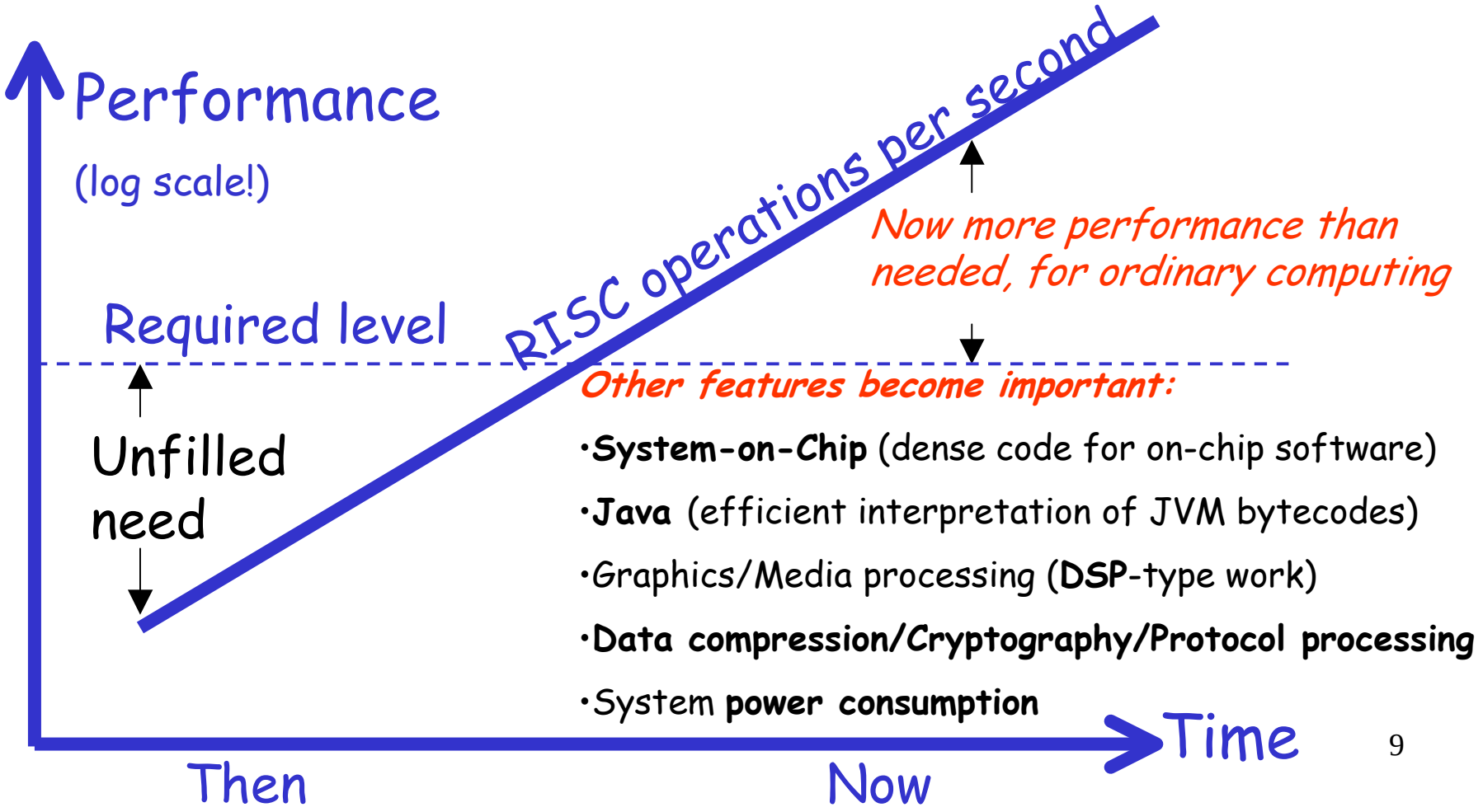
EXECUTION TIME

JAVA BYTECODES FOR FLOAT & DOUBLE ARITHMETIC



A reason for doing things differently

"RISC" processors were optimized for speed -- only.
But speed increases automatically, with Moore's law, as shown below.
Speed is now often sufficient, which may favor other principles



Word length and CMOS speed

- An 8-bit ALU needs 4 cycles to add 32-bit numbers
- This factor 4 corresponds to only 3 years of speed increase of CMOS technology (100-fold increase over the life of typical architectures)

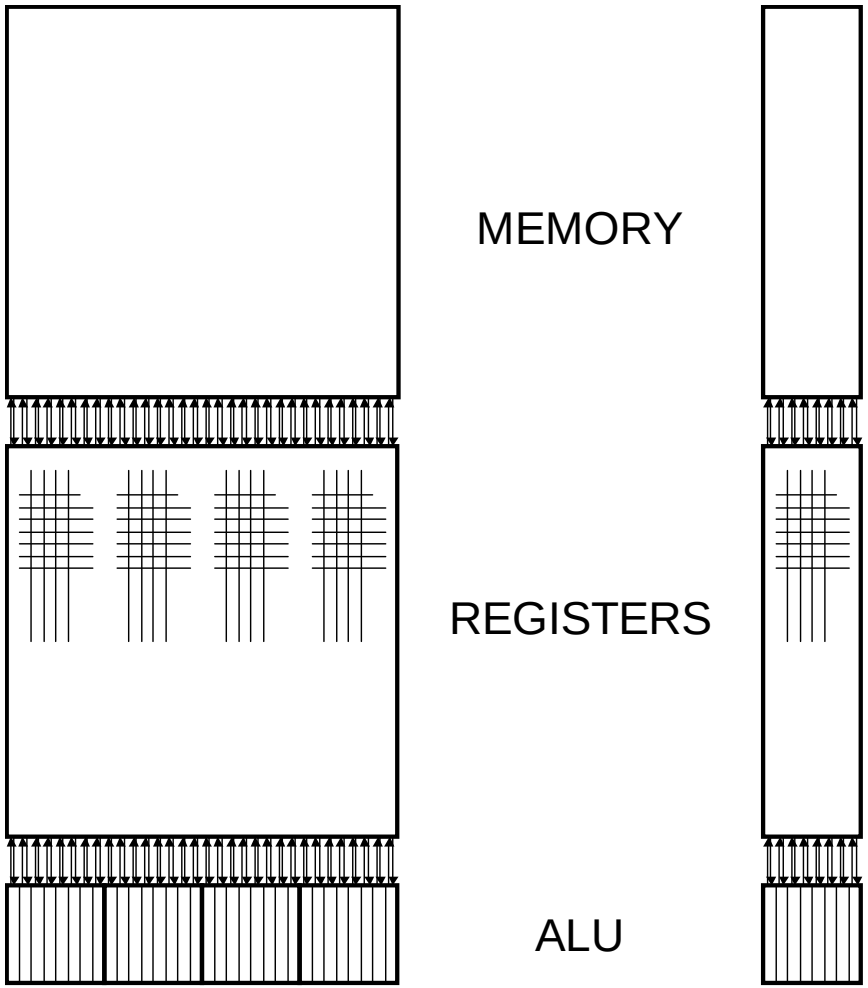
  => time reduces advantage of wide hardware
- Software usually does other things than adding 32bit data, which effectively reduces the factor
- Cost is always important
- Energy efficiency is getting much more important than the time needed for adding 32bit words

Note: Narrow is Better for Data Buses

- lower cost
- smaller
- more reliable
- less interference

Traditionally wider paths have been used for performance, but the reason for this have disappeared, since CMOS speed increases with every generation

The same applies for Processor Datapath!



- lower cost
- smaller
- more reliable
- less interference

Traditionally wider paths have been used for performance, but the reason for this is disappearing, since CMOS speed continues to increase with every generation

Implications of choosing 8-bit datapath when performance is needed

- Complex local control necessary => microcode
- Highly complex control is then available and can be used for important special functions*
- This calls for (partly) writable microcode
- Special functions can then be upgradeable and field configurable like software

*) Note that standard algorithms and virtual machines are getting more and more important

Microprogrammed 8-bit Microarchitecture

In addition to the

1) lower cost (fewer transistors)
2) higher reliability (fewer connections)
3) less interference (fewer parallel switching nodes)
4) higher flexibility (upgradeable internal operation)

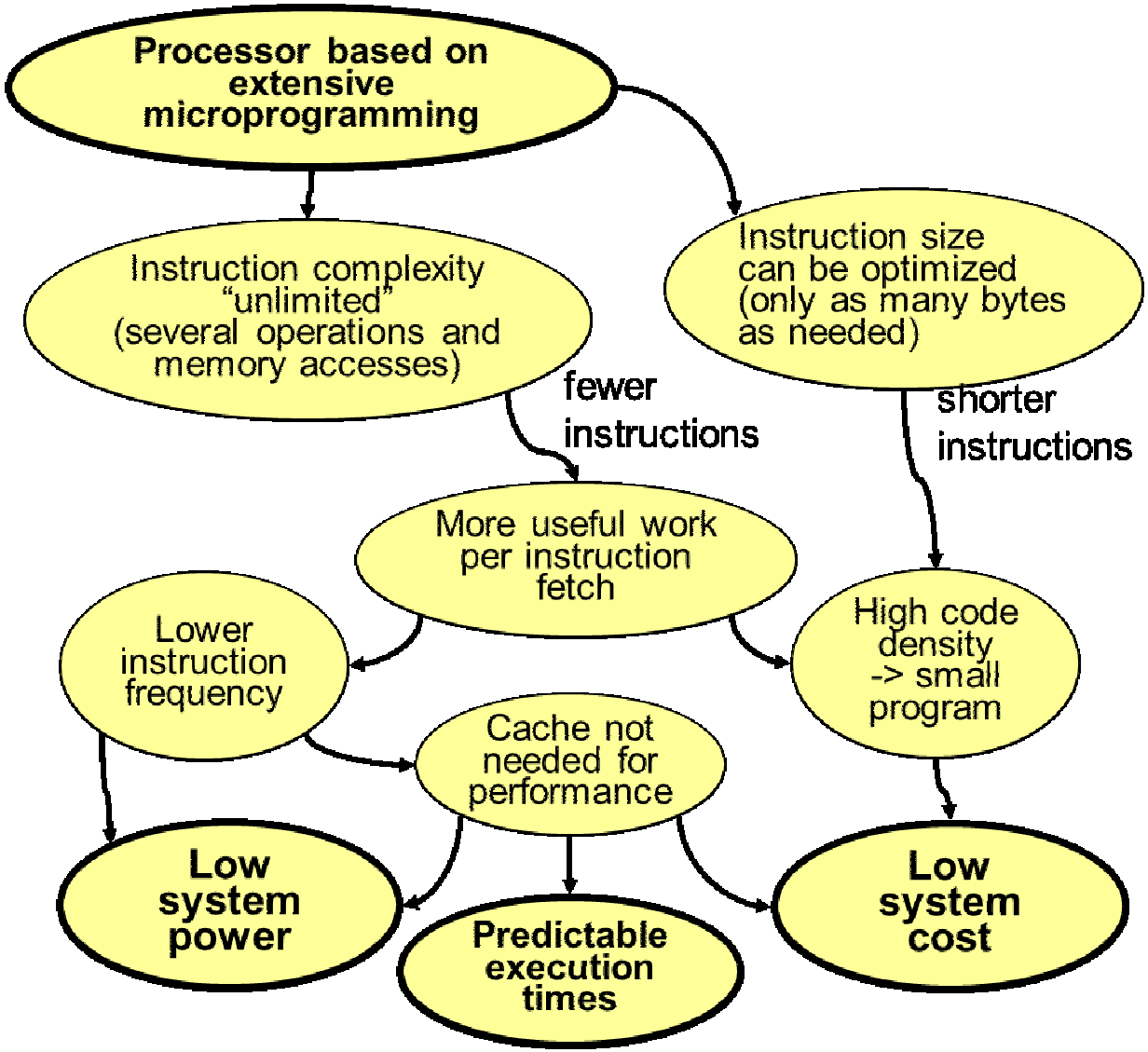
the architecture also has

5) higher efficiency and speed for important code --

e.g. VM interpretation, graphics, DSP, Peripheral I/O --

due to fewer dumb instructions and non-computing cycles

Soft microcode leads to low cost and high efficiency -- for any type of processing

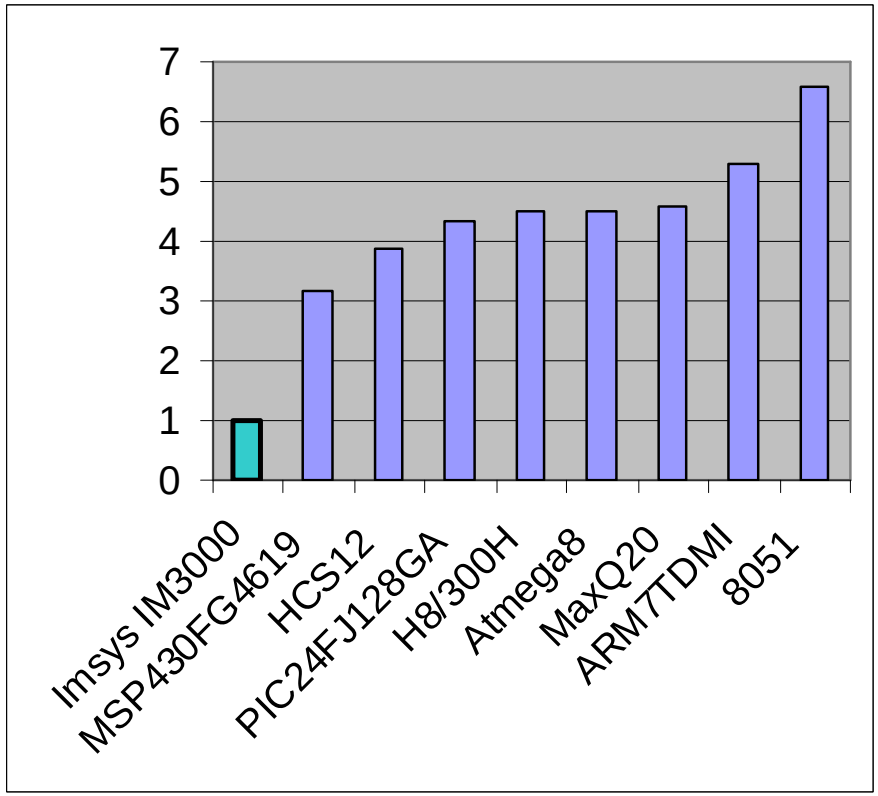


Microcode used in three ways

- Instruction execution
  - This is normal for CISC, but this ISA is closer to compiler and includes JVM bytecodes, for increased efficiency
- Special microcode for processing hotspots increases efficiency
  - Includes DSP, crypto, graphics, and JVM garbage collection
- Built-in peripheral control simplifies system
  - Autonomous microcode uses memory + core, sometimes also timers and I/O buffer memory, to handle e.g. Ethernet, LCD, and general DMA

Superior Code Density

- Imsys IM3000 Clear Winner in Non-Optimized C-Code
- Compared Against Both 8bit and 32bit ARM Processors



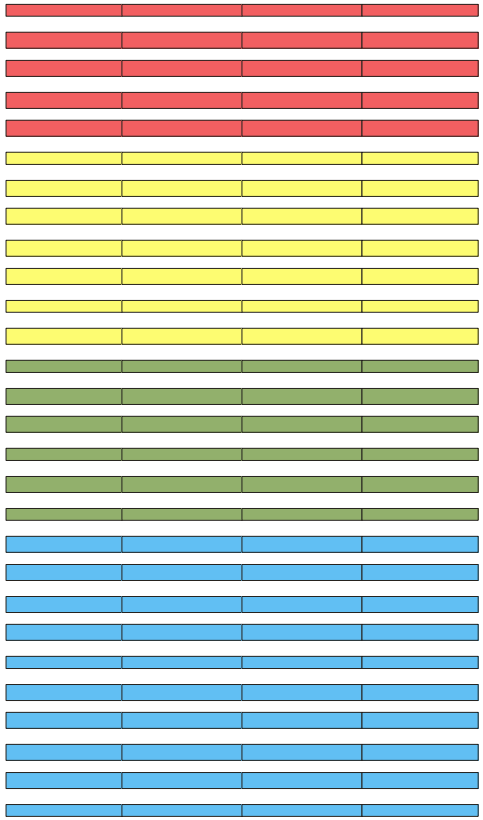
*ARM Uses Compact ”Thumb” Instruction Set

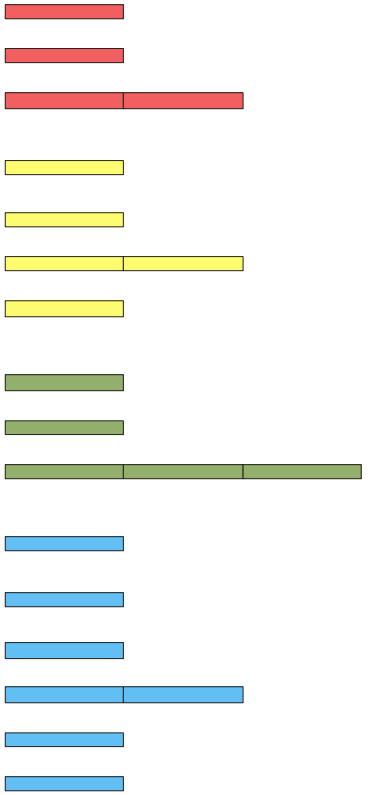
... therefore:

Smaller compiled program code

RISC

(Register oriented, fixed format, elementary instructions)





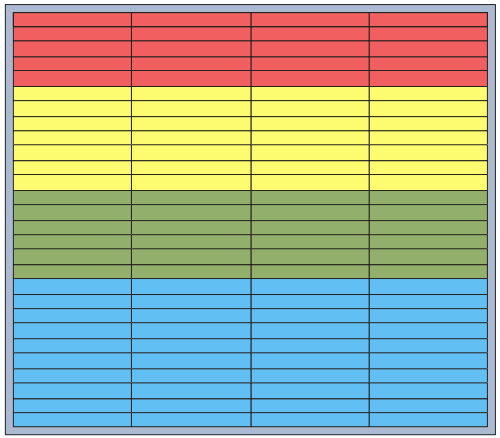
Imsys

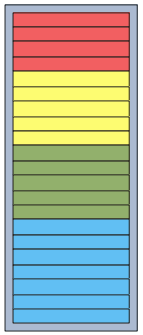
(stack oriented, variable length, complex instructions)

... therefore:

Smaller memory

RISC

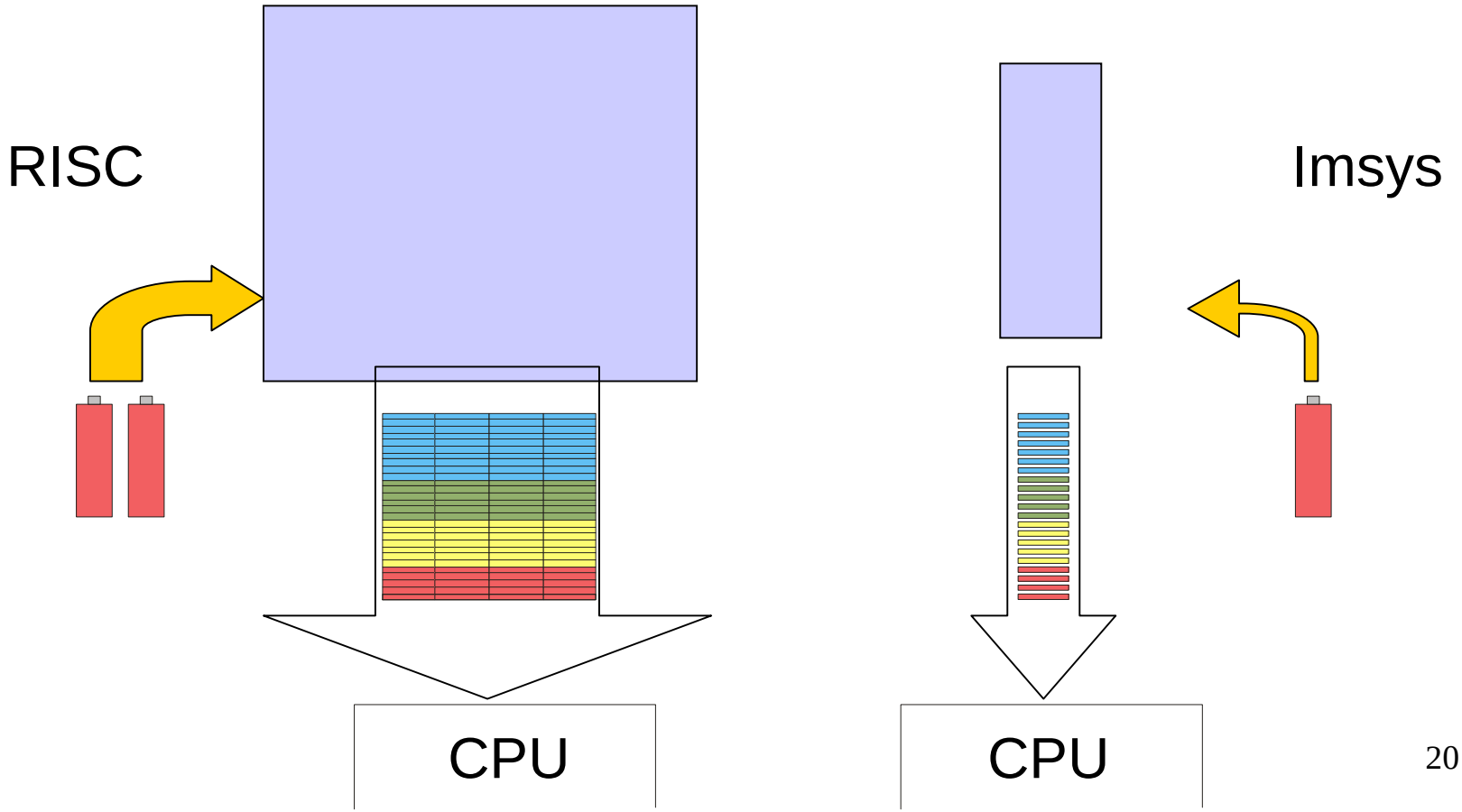




Imsys

... and:

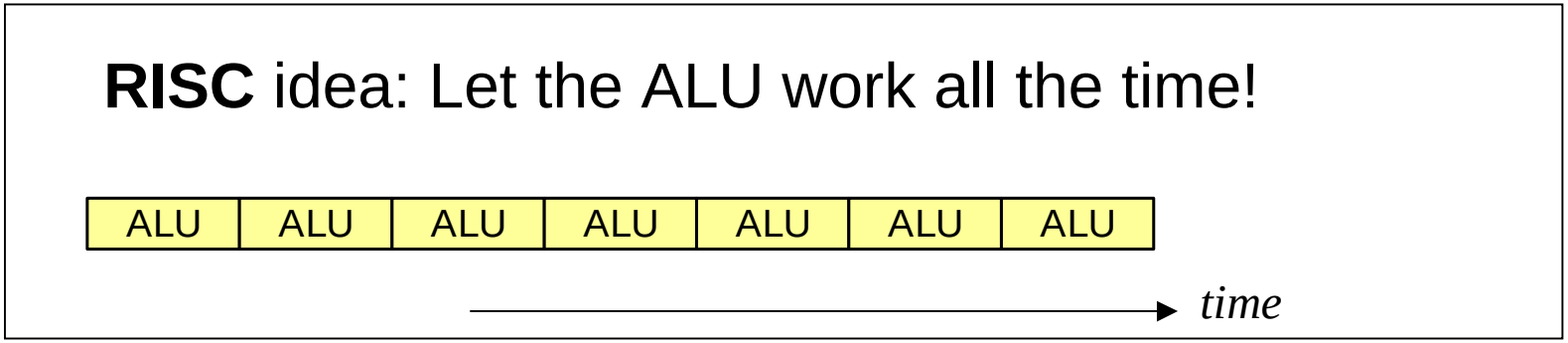
Lower activity => lower pwr consumption

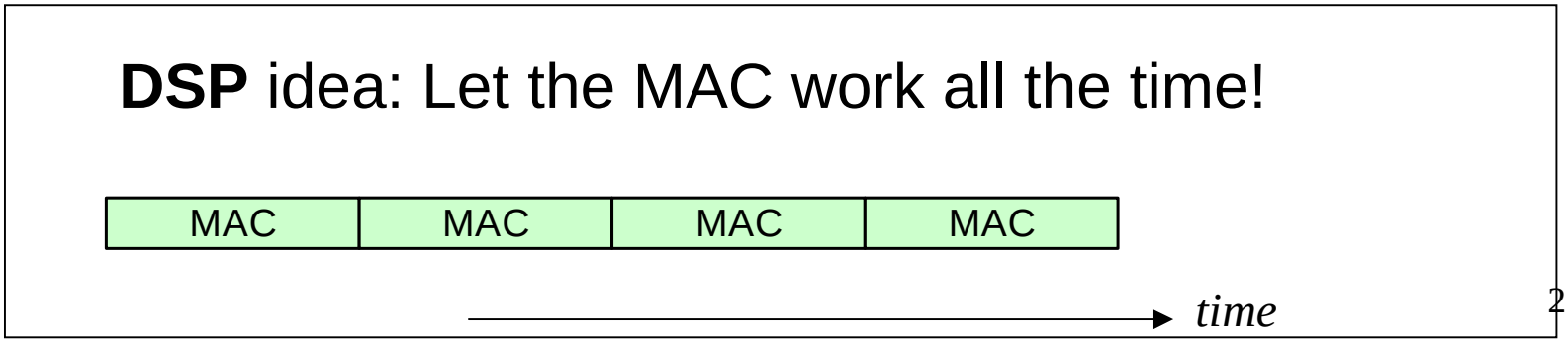


Optimization is usually centered on one execution resource - This limits flexibility

A RISC is good at adding 32bit integers – a DSP is good at calculating sums of products of 16bit fractions. They can't do each other's job efficiently.

Also, both are bad at interpretation of virtual machine instructions.

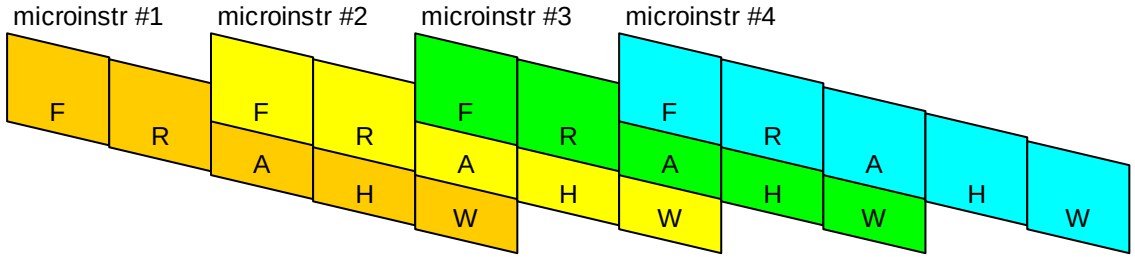




Imsys processor is optimized for efficient use of main memory, which nowadays more often is the limiting resource

- Stack orientation => register references disappear from instructions => reduce needed instruction bits
- Instructions more efficiently coded (short, different length depending on need) => reduce unused bits
- Complex instructions: may do several operations and memory accesses => reduce number of instructions
- => Reduced bandwidth => reduced cost and consumption
- Direct control of dynamic memory makes accessing more efficient:

Imsys' microinstructions are pipelined, like the instructions of RISC machines



-but each word is much wider than a RISC instruction and the total flow of control information is higher

- Furthermore, the microprogram often pipelines loops and sometimes executes different operation sequences in parallel (Example: Montgomery multiplication in RSA crypto routine can thereby utilize practically all time for multiplication)

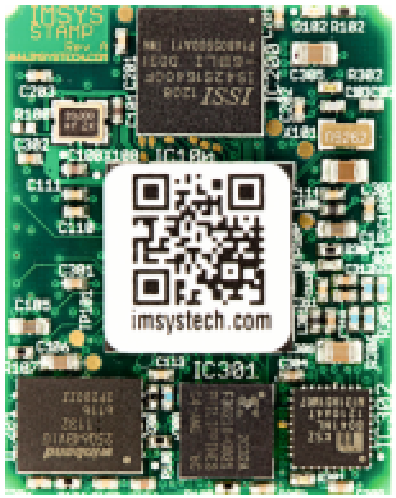
Velox Module

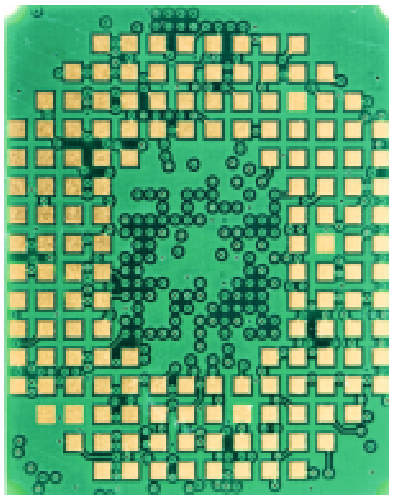
Compact module
for surface mounting

25.4 x 32 mm

156 pad LGA footprint

Delivered on tape for
automatic assembly





Complete with
8MB SDRAM,
2 or 8MB flash memory,
optional Ethernet PHY,
3 UARTs, SPI / I2C,
83MB/s I/O channel,
RTC with battery input,
8 timers,
8-ch ADC(16 bit)
2 DAC(16bit)

The typical customer PCB will be simplified and less costly than when using the bare IC, since the part requiring high density, multilayer PCB and attention to EMC issues is in cluded on the module.

Testing will also be simpler and yield will be higher, and Imsys firmware is already onboard (optionally also the customer application software).

Perhaps even more important: Any necessary changes of this core system, needed for ensuring a long commercial life of the customer product – e.g. design changes motivated by component availability issues – will also be taken care of, including firmware adaptation and maintenance.

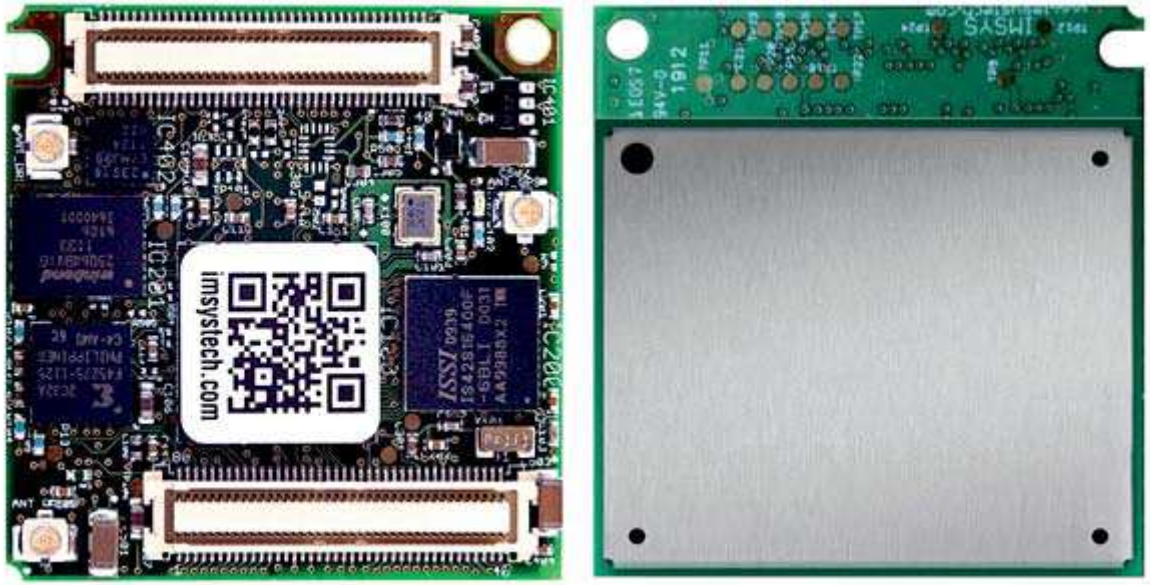
Aerius Module

IM3K-PHS8
Wireless Module

front and back shown
33,9 x 35 mm

Pin compatible with
Cinterion Tc65i

Adds 3G+ (5 band) to
existing designs



Adds more functionality in new designs:
•diversity antenna
•optional GPS
•SNAP (Simple Network Application Platform) with Java and Ethernet, UARTs, SPI, etc.
•Extra 80-pin I/O connector for the additional interfaces