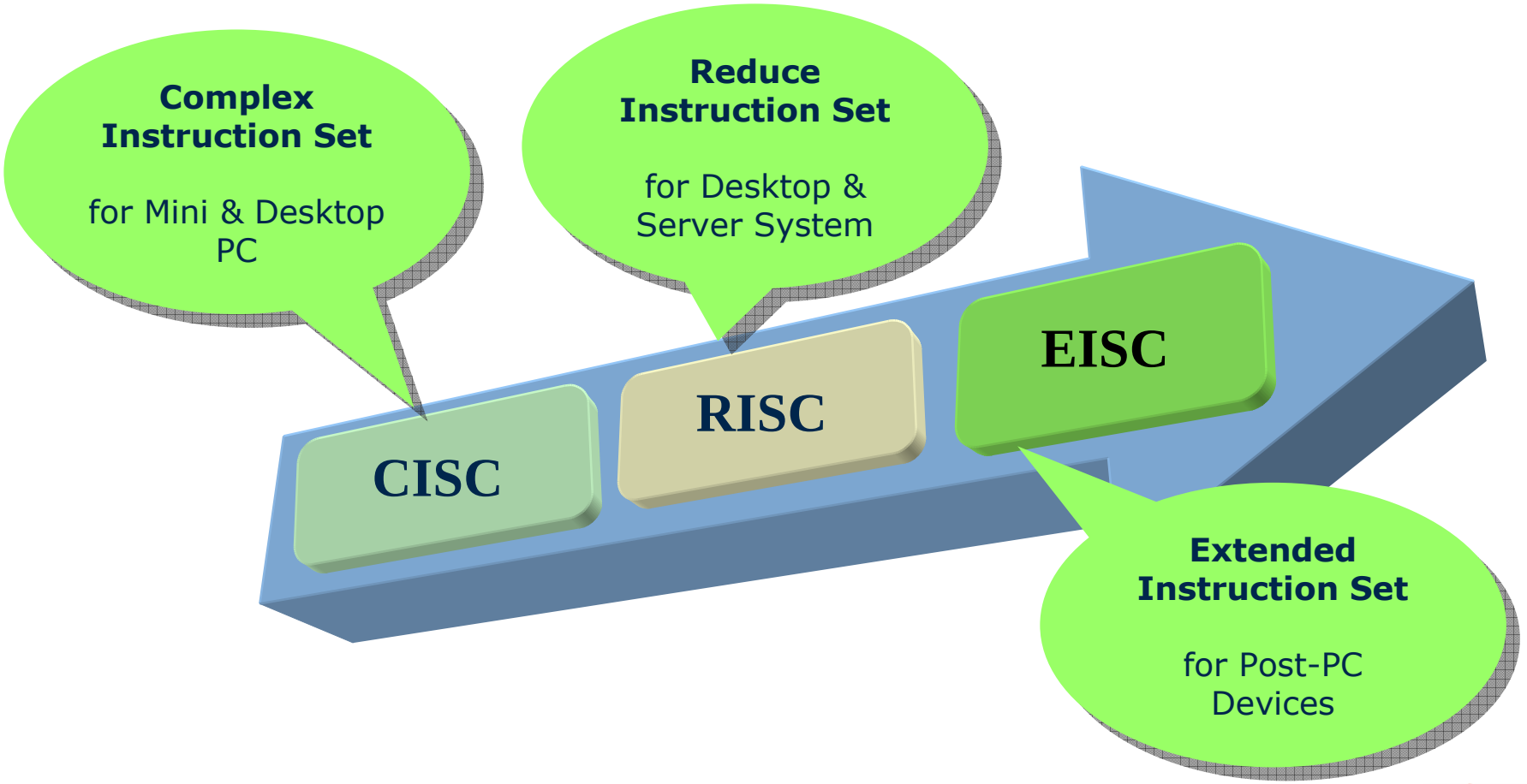
EISC Architecture:
Computer Architecture for Era of Post PC

Agenda

- EISC Processor Overview
  - Trends
  - EISC: Extendable Instruction Set Computer
  - EISC processors
- AE32000; 32-bit EISC processor
  - AE32000 Features
  - AE32000 Microarchitecture

Trends

❑ Development Cycle in Microprocessor



CISC

- Complex Instructions
  - Easy programming with Assembly Code
  - Many instructions for many situations
- Variable Length Instruction
  - Complex for hardware Implementation
  - Various execution time for each instruction

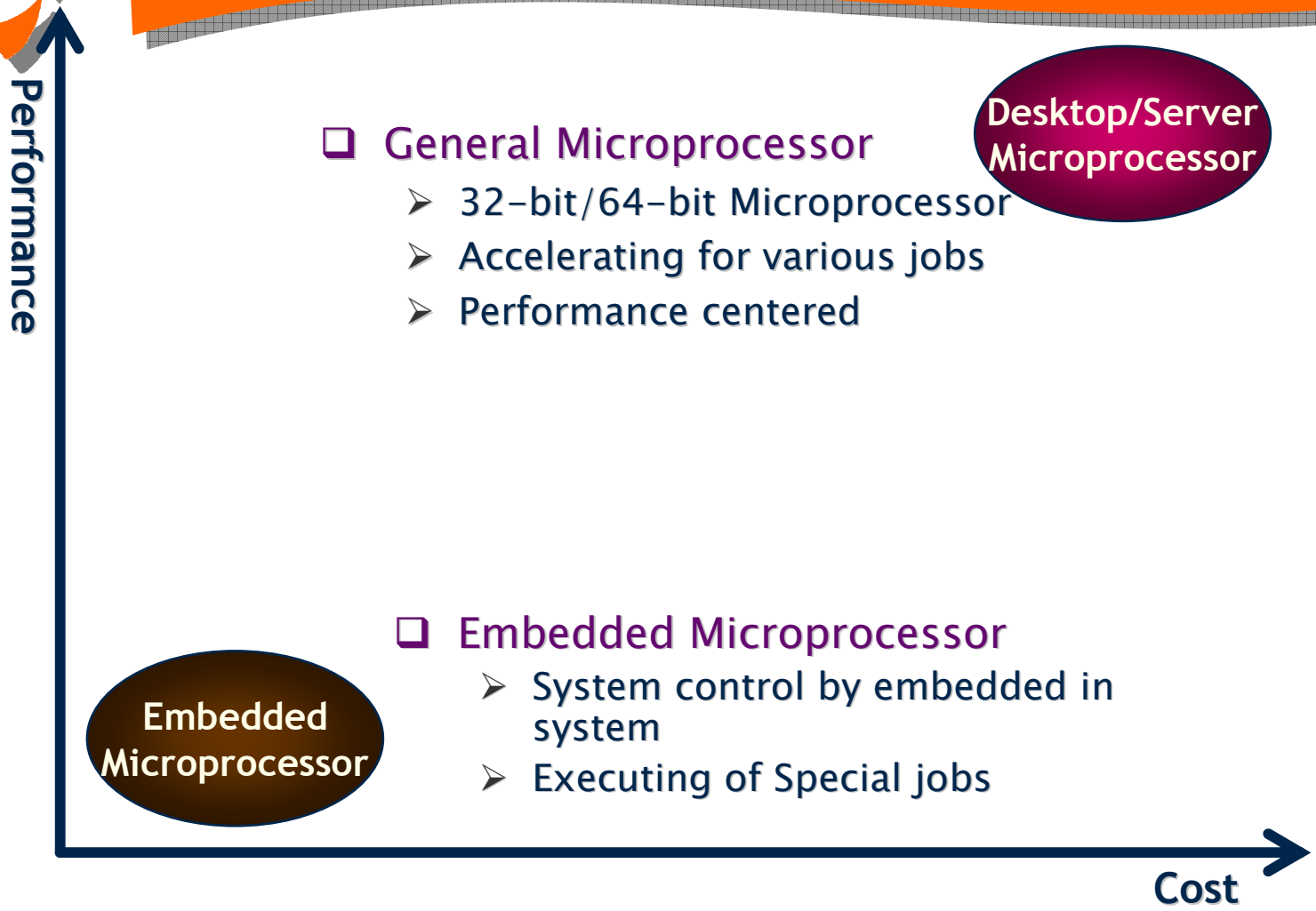
RISC

- Reduced Instruction Set
  - Weak point of CISC
    - ✓ Part of commands are used even though existing of many commands
    - ✓ Hardware consumption for rarely executing commands
- Make Common Case Faster!
  - Reduce # of Instructions
  - Take advantage of Compiler Technology and Pipeline architecture
  - Simple & Efficient Hardware

Post-PC Era.

- Devices
  - Emergence of various digital equipments
  - Embedded System!
    - ✓ "An **embedded system** is a special-purpose system in which the computer is completely encapsulated by the device it controls" [wikipedia].
  - Embedded microprocessor.
- Requirements
  - Low cost
  - Appropriate performance for required operation
  - High power efficiency

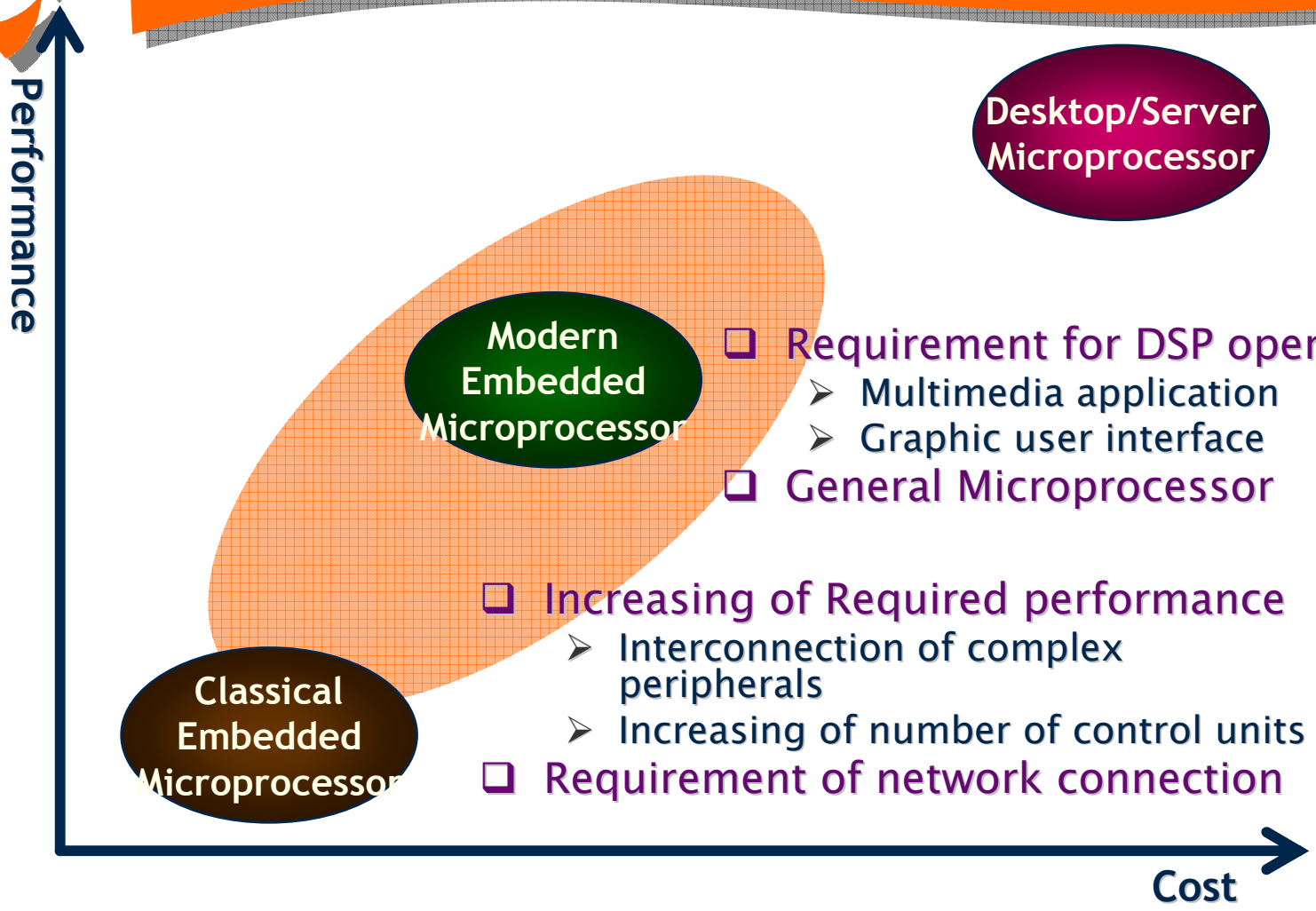
Embedded Microprocessor



- General Microprocessor
  - 32-bit/64-bit Microprocessor
  - Accelerating for various jobs
  - Performance centered

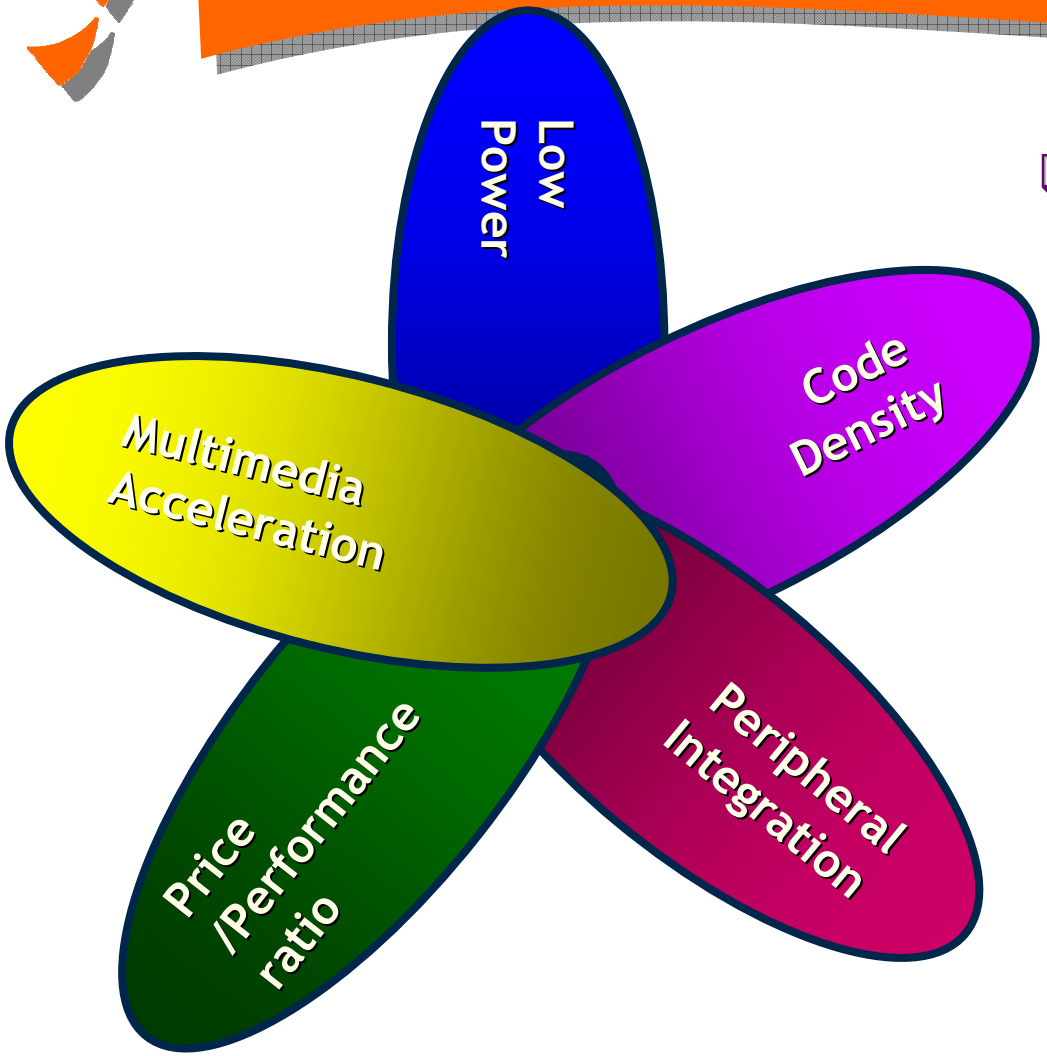
- Embedded Microprocessor
  - System control by embedded in system
  - Executing of Special jobs

Embedded Microprocessor



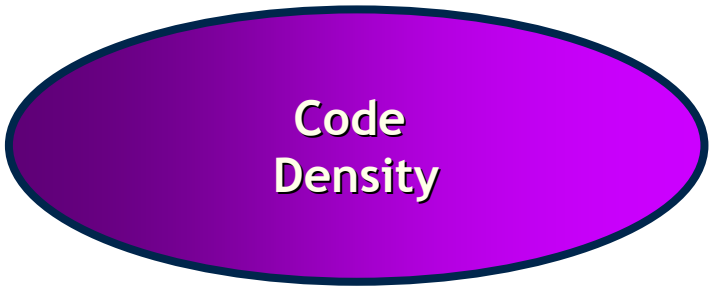
- Requirement for DSP operation
  - Multimedia application
  - Graphic user interface
- General Microprocessor
- Increasing of Required performance
  - Interconnection of complex peripherals
  - Increasing of number of control units
- Requirement of network connection

Embedded Microprocessor



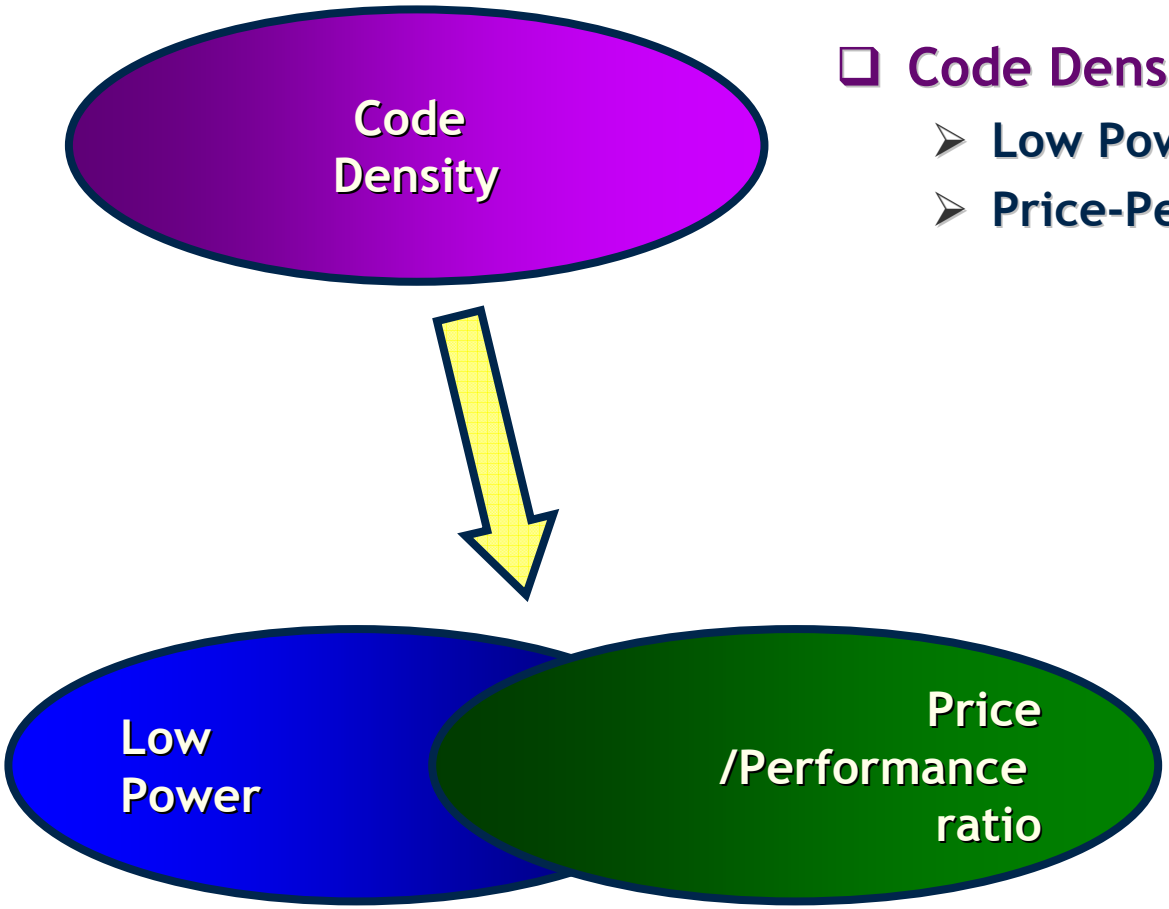
- 5 Criteria for Modern Embedded Microprocessors
  - M.Schlett

Embedded Microprocessor



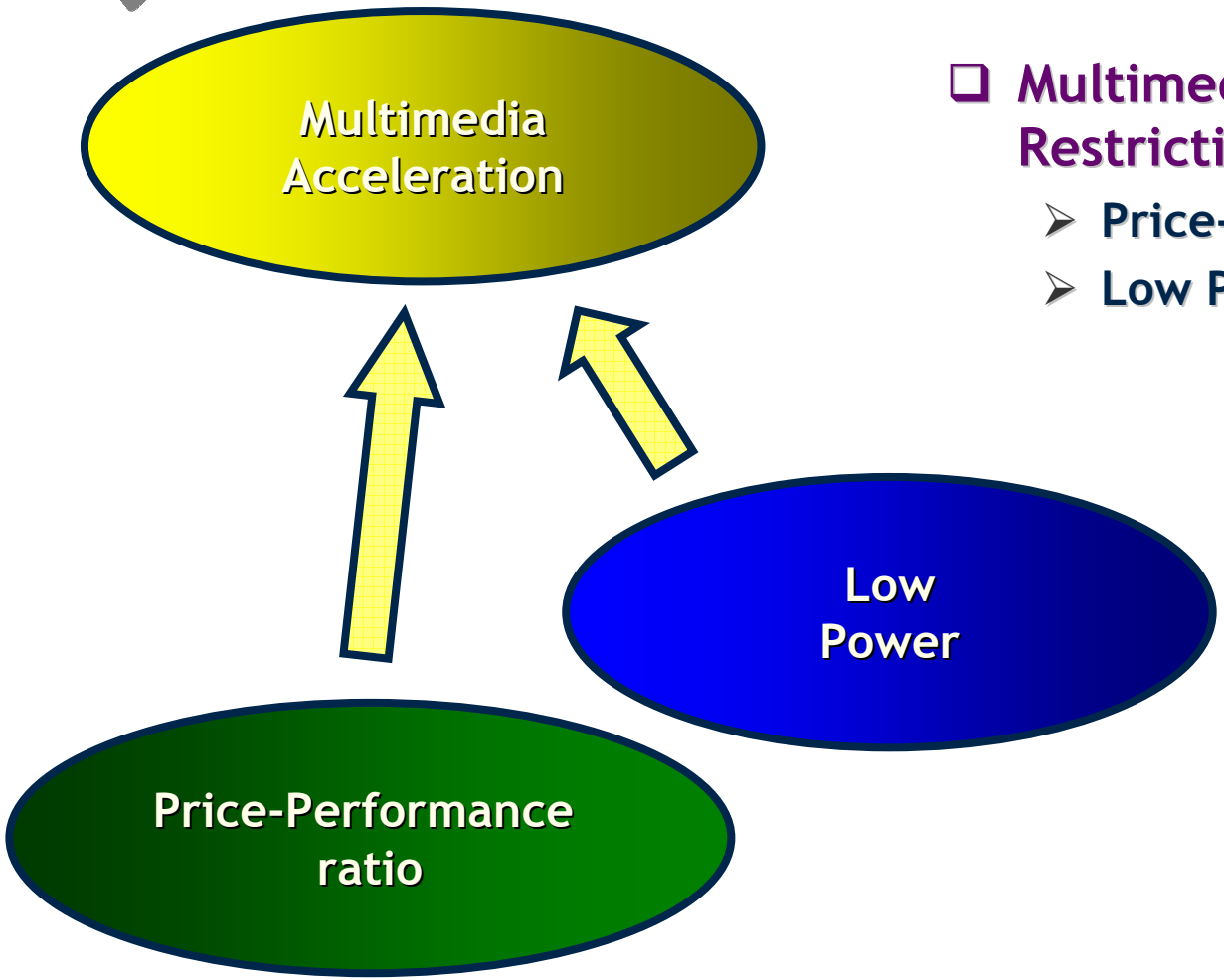
- **Code Density**
  - Inverse of static code size
  - Static code size
    - ✓ **Memory size for storing program**
    - ✓ **Code size + Initialized static data**
- **High code density**
  - Small memory size requirement for same program storing
    - ✓ Reducing memory size
  - Reducing power consumption for command call
  - Reducing performance degradation for memory access
    - ✓ Accessing time difference between memory and processor
  - High efficiency of instruction cache

Embedded Microprocessor



- Code Density
  - Low Power
  - Price-Performance Ratio

Embedded Microprocessor

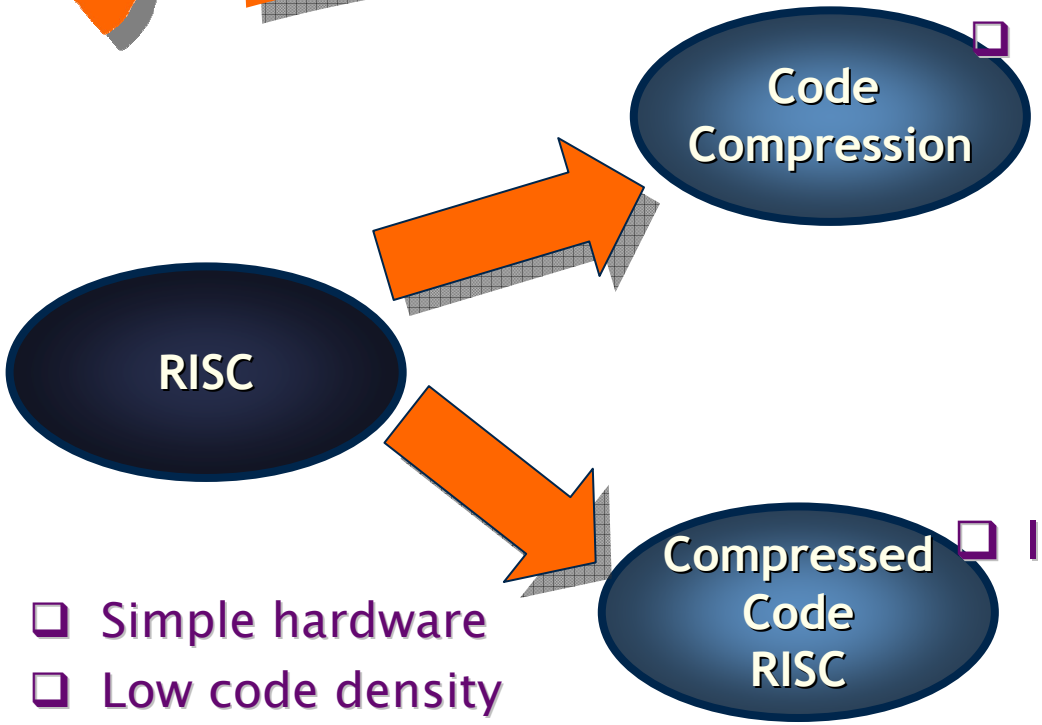


- Multimedia Acceleration; Restriction
  - Price-Performance Ratio
  - Low Power

Embedded Microprocessor

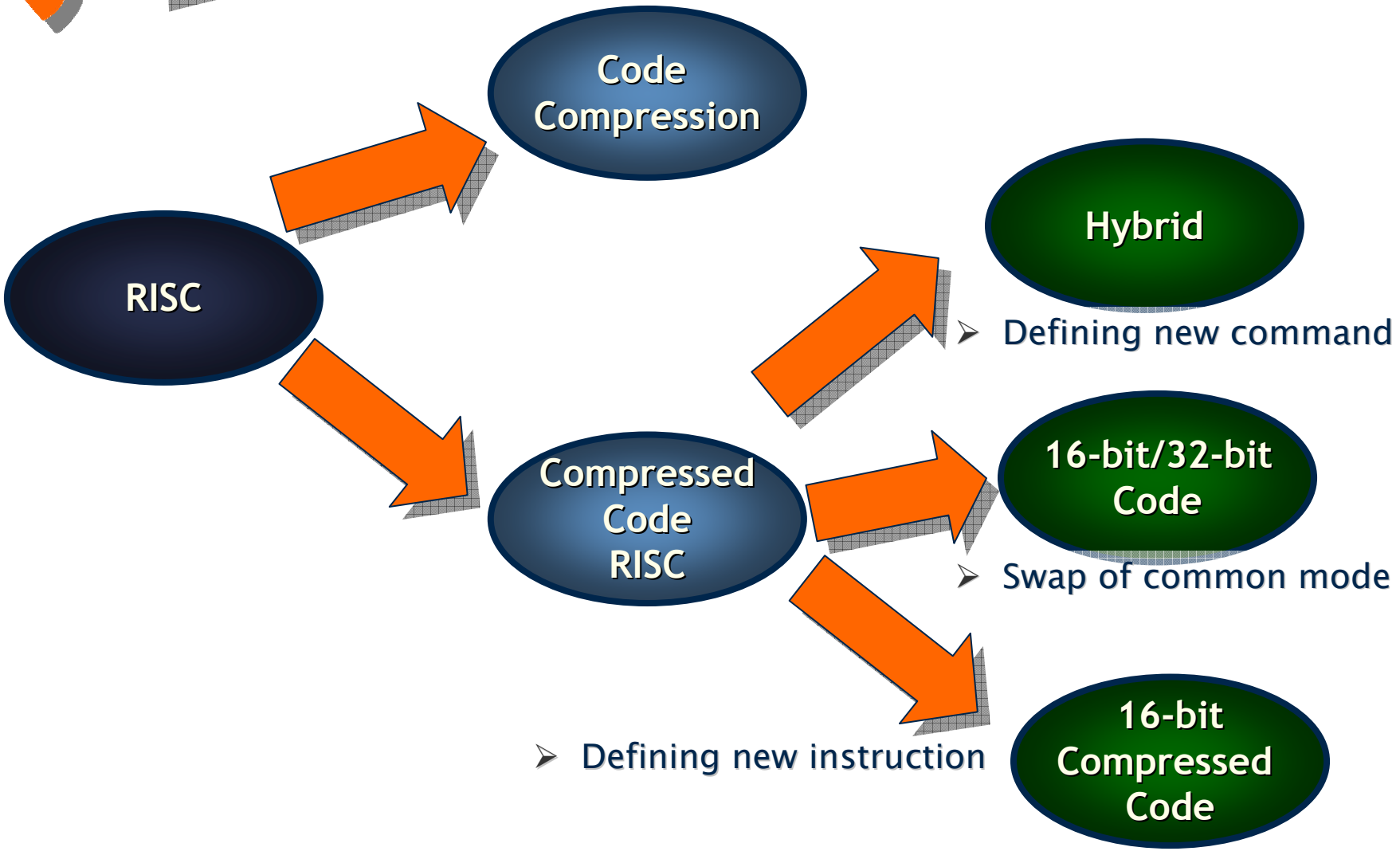
- Code Density
  - One of the most important metric
- CISC vs RISC, and...
  - CISC; Variable Length Code
    - ✓ Various instruction for each case
    - ✓ Various instruction length
      - ✓ High code density
  - RISC; 32-bit Fixed field instruction
    - ✓ Using of same code
      - ✓ Low code density

Improving Code Density



- Instruction Compressing
  - Using lossless compress method
  - Increasing hardware cost
  - High interoperability
- ISA using short instruction
  - Short-length Instruction
  - Problem of interoperability
- Simple hardware
- Low code density

Improving Code Density



Improving Code Density

- Compressed Code RISC
  - Long word control by short instruction
  - 16-bit/24-bit instruction set, 32-bit data
- Short Instruction
  - Restriction of size of bit for command encoding
- Command Encoding
  - OP-code + Operand

Improving Code Density

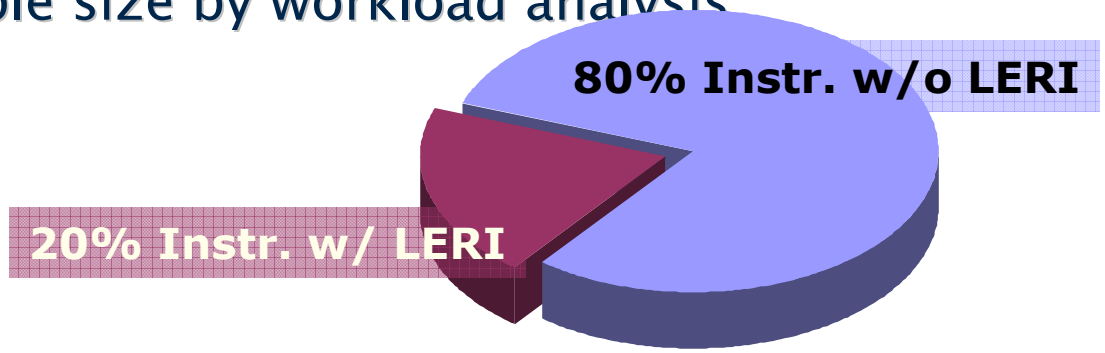
- Operand
  - Operation between Registers requires small bits
    - ✓ Same bit requirement for number of register
  - Immediate/Offset
    - ✓ Constant value for operation
    - ✓ Offset of indexed addressing
- Length of Immediate/Offset
  - Varying for instructions (up to 32bit)
  - High utilization of short immediate value, low utilization of long immediate value

Improving Code Density

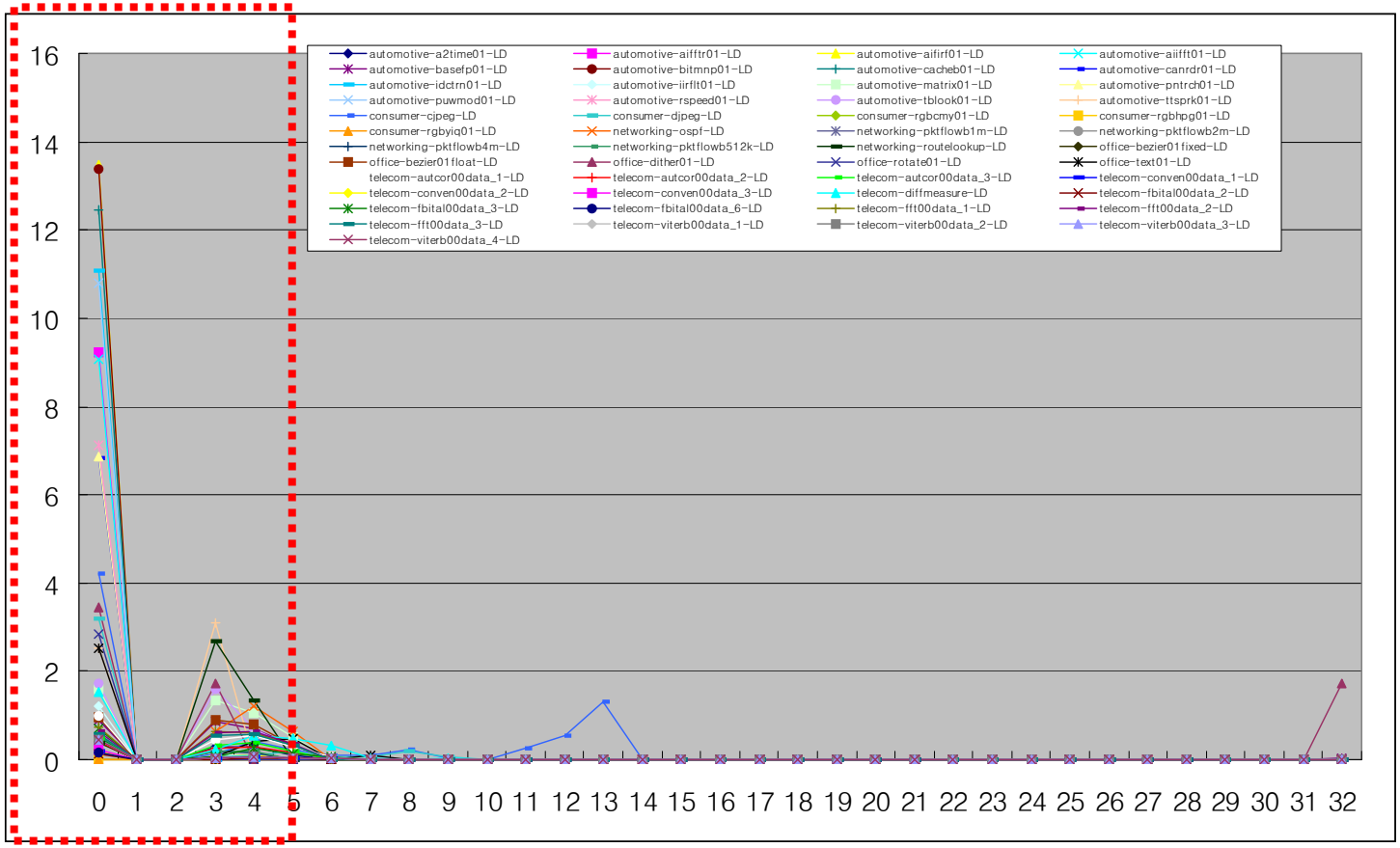
- Compressed code architecture
  - Lack of command filed for immediate value
- RISC based architecture
  - Case of long immediate value is required
    - ✓ Execute after moving immediate value to general register
    - ✓ Combination of multiple “LDI-SHIFT-ORI” commands
      - ✓ Complicate procedure
    - ✓ Load after locating immediate value to static data area
      - ✓ Require of data access
  - Reserve as many bits for immediate value

EISC

- EISC: Extendable Instruction Set Computer
  - Extend immediate value at Special purpose register or *Extension Register(ER)*
  - LERI Instruction; Command for load/store from/to ER
    - ✓ Elimination of general register using and combination of complex Instructions
- Immediate value of Instruction
  - Assigning suitable size by workload analysis
  - <20% instrs.



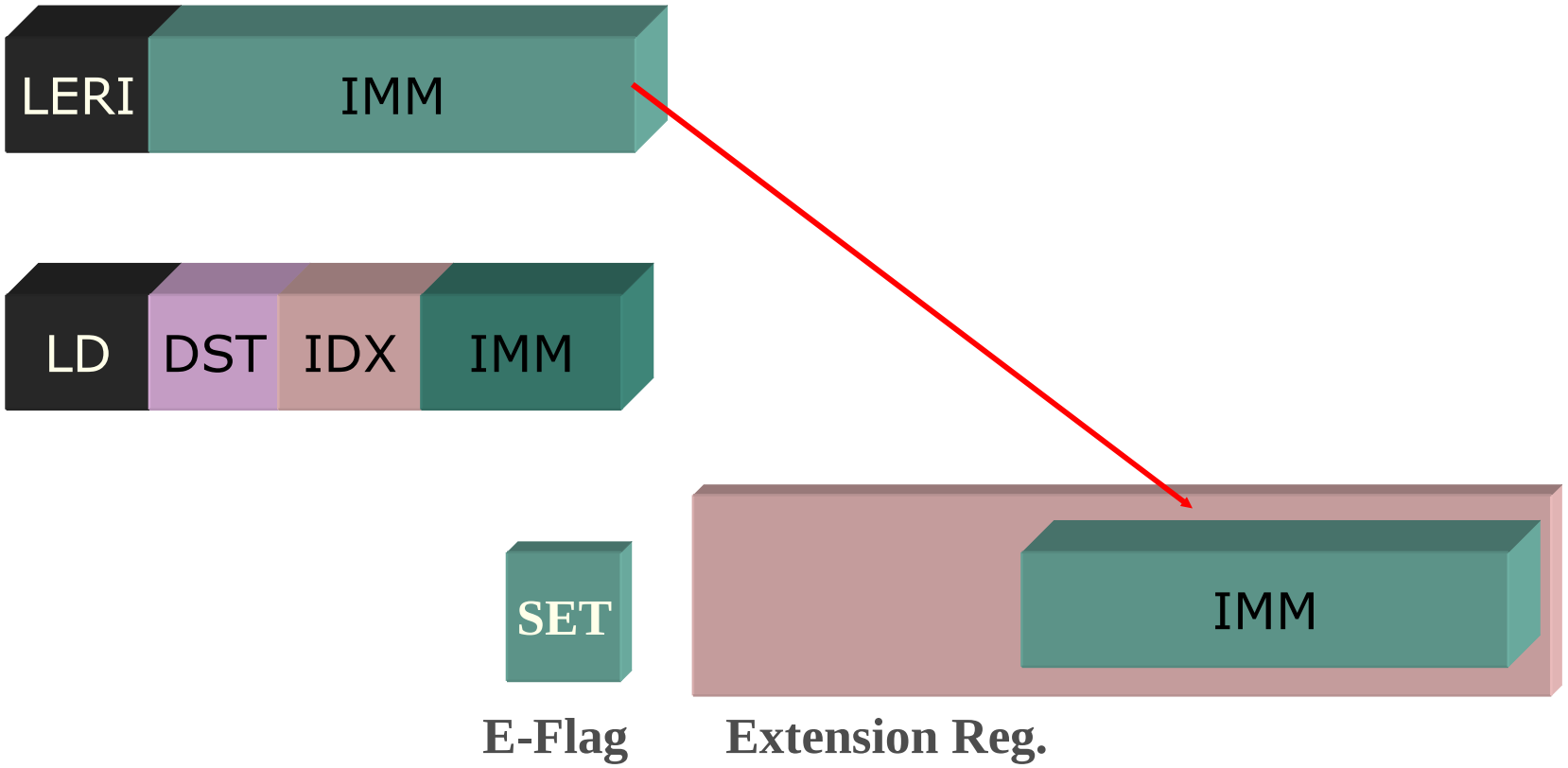
LD example


automotive-a2time01-LD
automotive-aifftr01-LD
automotive-aifirf01-LD
automotive-aiifft01-LD
automotive-basefp01-LD
automotive-bitmnp01-LD
automotive-cacheb01-LD
automotive-canrdr01-LD
automotive-idctrn01-LD
automotive-iirflt01-LD
automotive-matrix01-LD
automotive-pntrch01-LD
automotive-puwmod01-LD
automotive-rspeed01-LD
automotive-tblook01-LD
automotive-ttsprk01-LD
consumer-cjpeg-LD
consumer-djpeg-LD
consumer-rgbcmy01-LD
consumer-rgbhpg01-LD
consumer-rgbyiq01-LD
networking-ospf-LD
networking-pktflowb1m-LD
networking-pktflowb2m-LD
networking-pktflowb4m-LD
networking-pktflowb512k-LD
networking-routelookup-LD
office-bezier01fixed-LD
office-bezier01float-LD
office-dither01-LD
office-rotate01-LD
office-text01-LD
telecom-autcor00data_1-LD
telecom-autcor00data_2-LD
telecom-autcor00data_3-LD
telecom-conven00data_1-LD
telecom-conven00data_2-LD
telecom-conven00data_3-LD
telecom-diffmeasure-LD
telecom-fbital00data_2-LD
telecom-fbital00data_3-LD
telecom-fbital00data_6-LD
telecom-fft00data_1-LD
telecom-fft00data_2-LD
telecom-fft00data_3-LD
telecom-viterb00data_1-LD
telecom-viterb00data_2-LD
telecom-viterb00data_3-LD
telecom-viterb00data_4-LD
16
14
12
10
8
6
4
2
0
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32

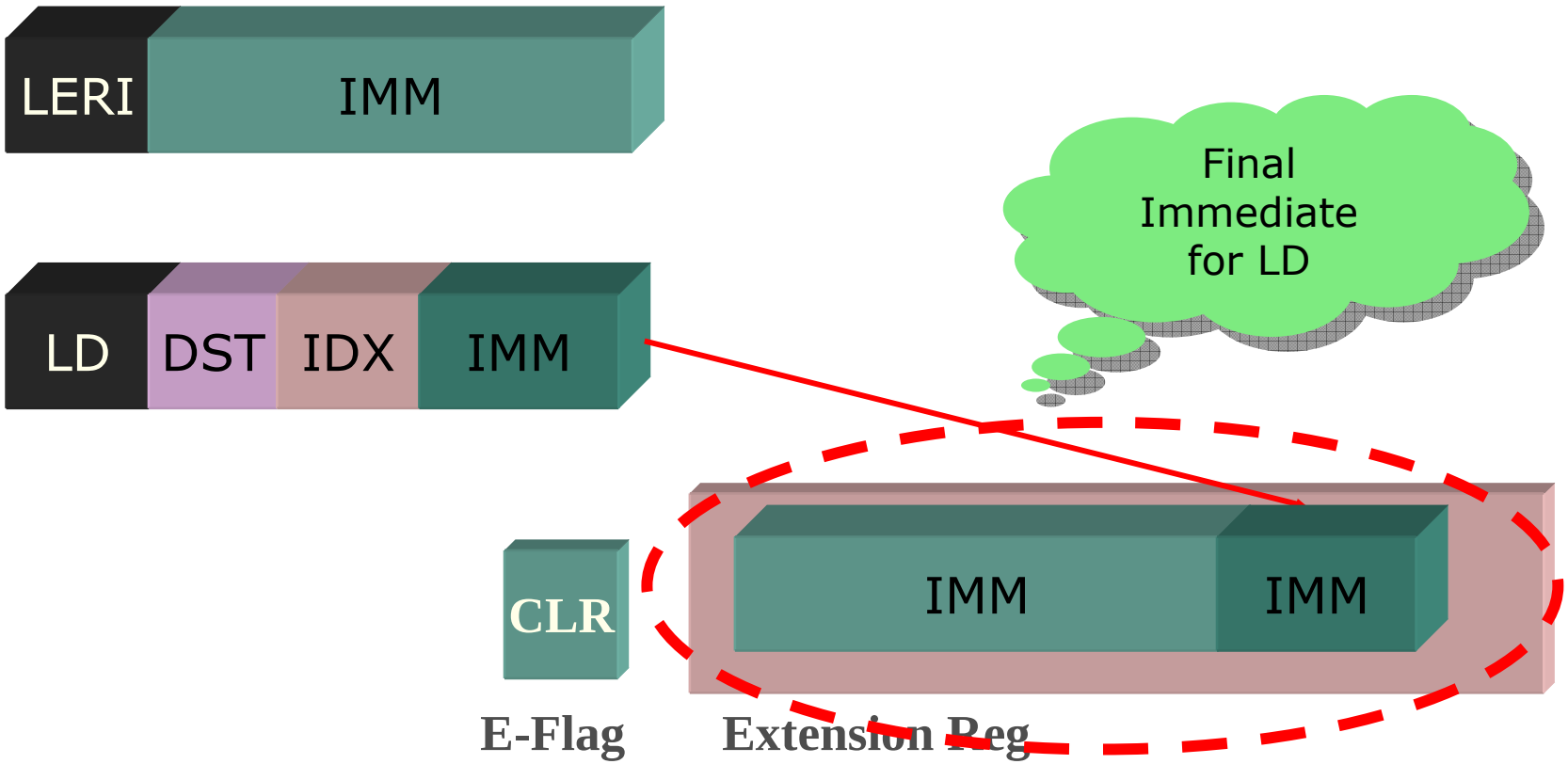
LERI makes long immediate

E-Flag Extension Reg.

LERI makes long immediate



LERI makes long immediate



EISC Architecture

- Assigning immediate by LERI
  - LERI itself is fixed length instruction
- Elimination of complex procedure for immediate value access
  - Low burden for long immediate
    - ✓ Assign relatively small size of bit for immediate
    - ✓ Increasing of bit size for opcode → Increasing number of instruction
    - ✓ Increasing of bit size for register indexing
  - Increasing available general register
    - ✓ 16 GPRs
    - ✓ ARM-THUMB; 8GPRs
  - No GPR for immediate value

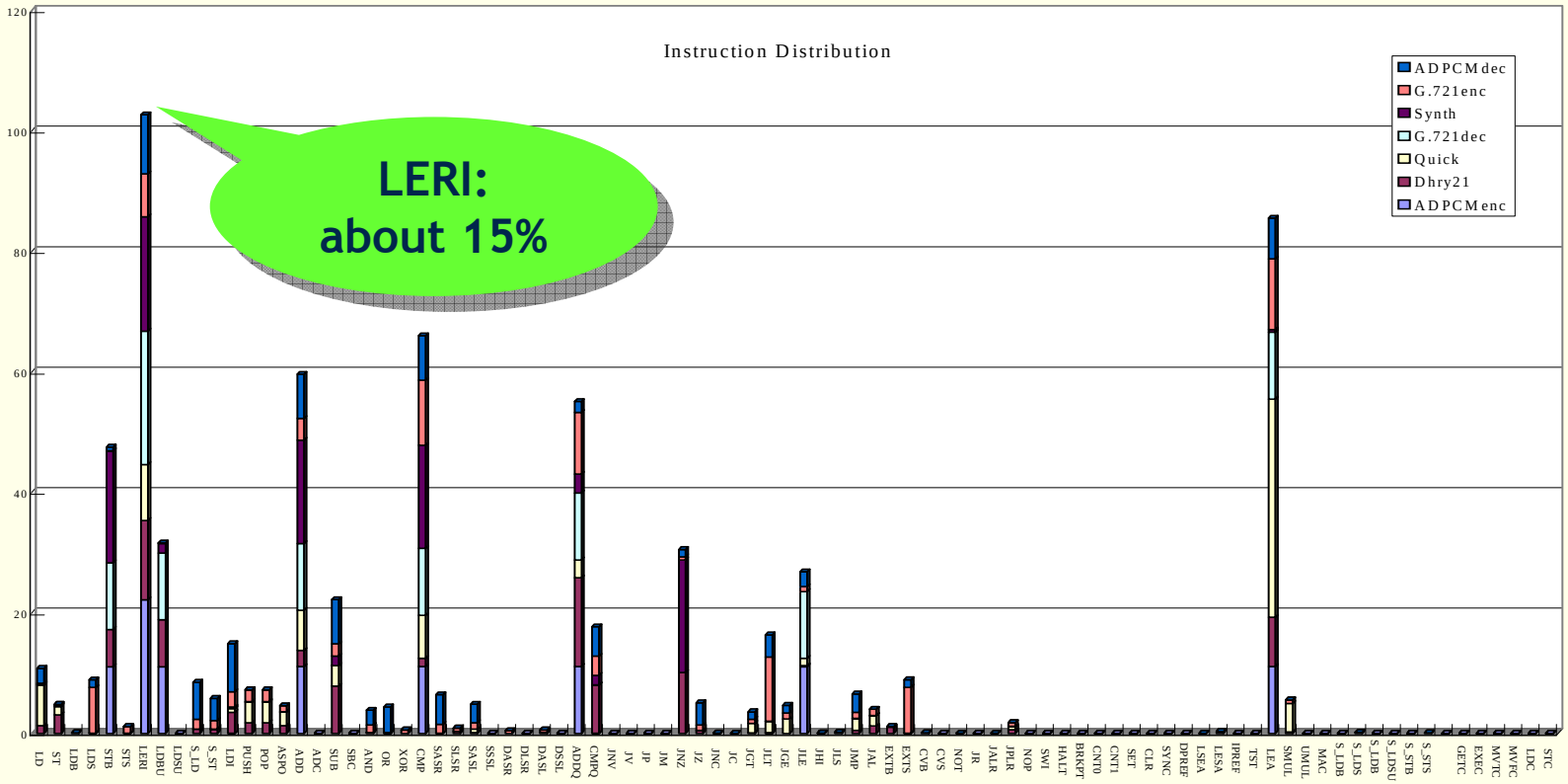
EISC Architecture

- Selection freedom for instruction type
  - Selection according to use LERI or not
    - ✓ Instruction such as *addq* is assigned if the short immediate values are frequently used
  - Good for long immediate operation
    - ✓ DSP

General *u*-proc.	EISC
sub : $Rd = $Ra – $Rd	sub : $Rd = $Ra – $Rd
subi : $Rd = $Ra – immediate	leri/sub : $Rd = $Ra – immediate

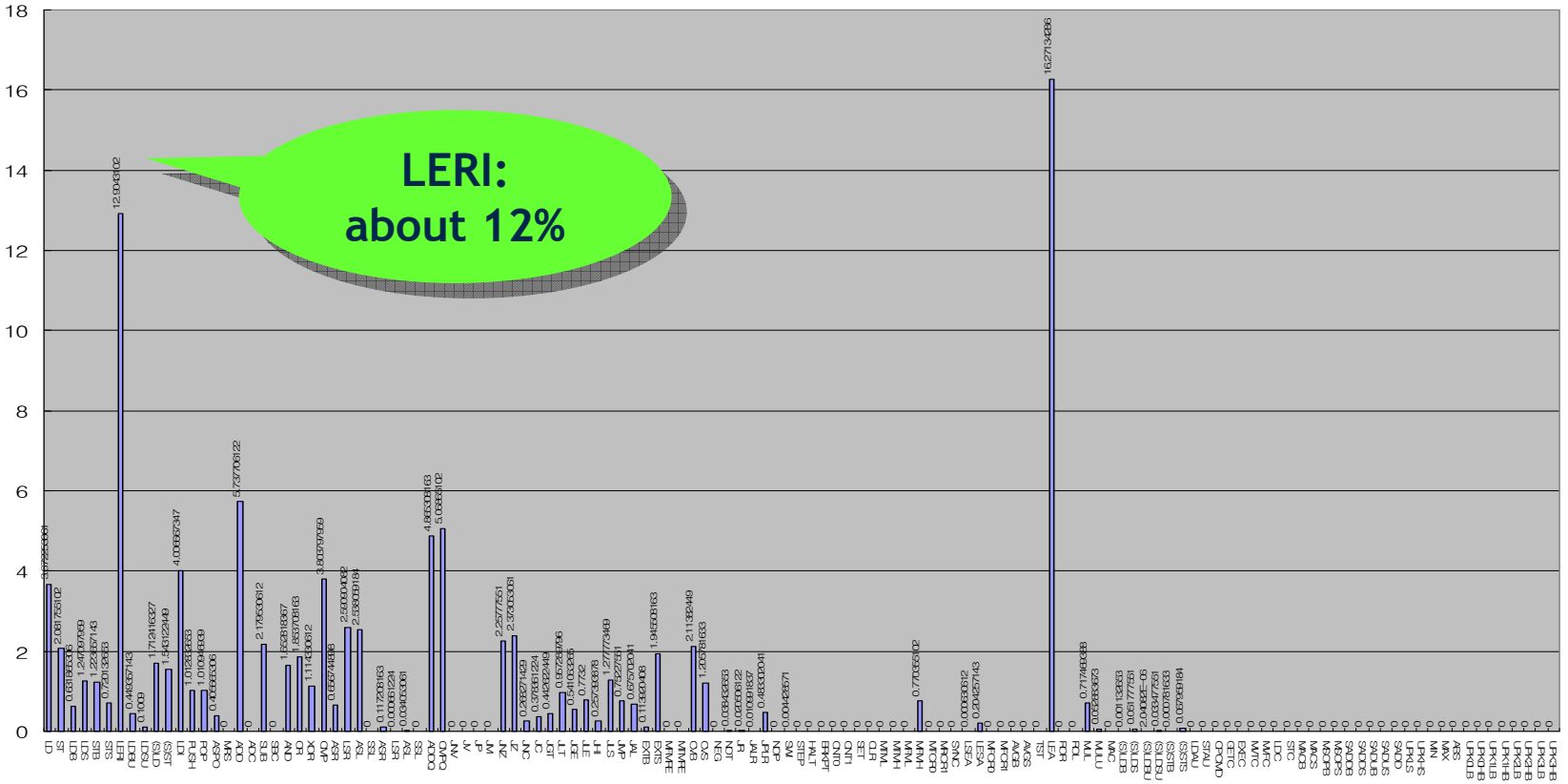
Impact of LERI

- Media benchmarks

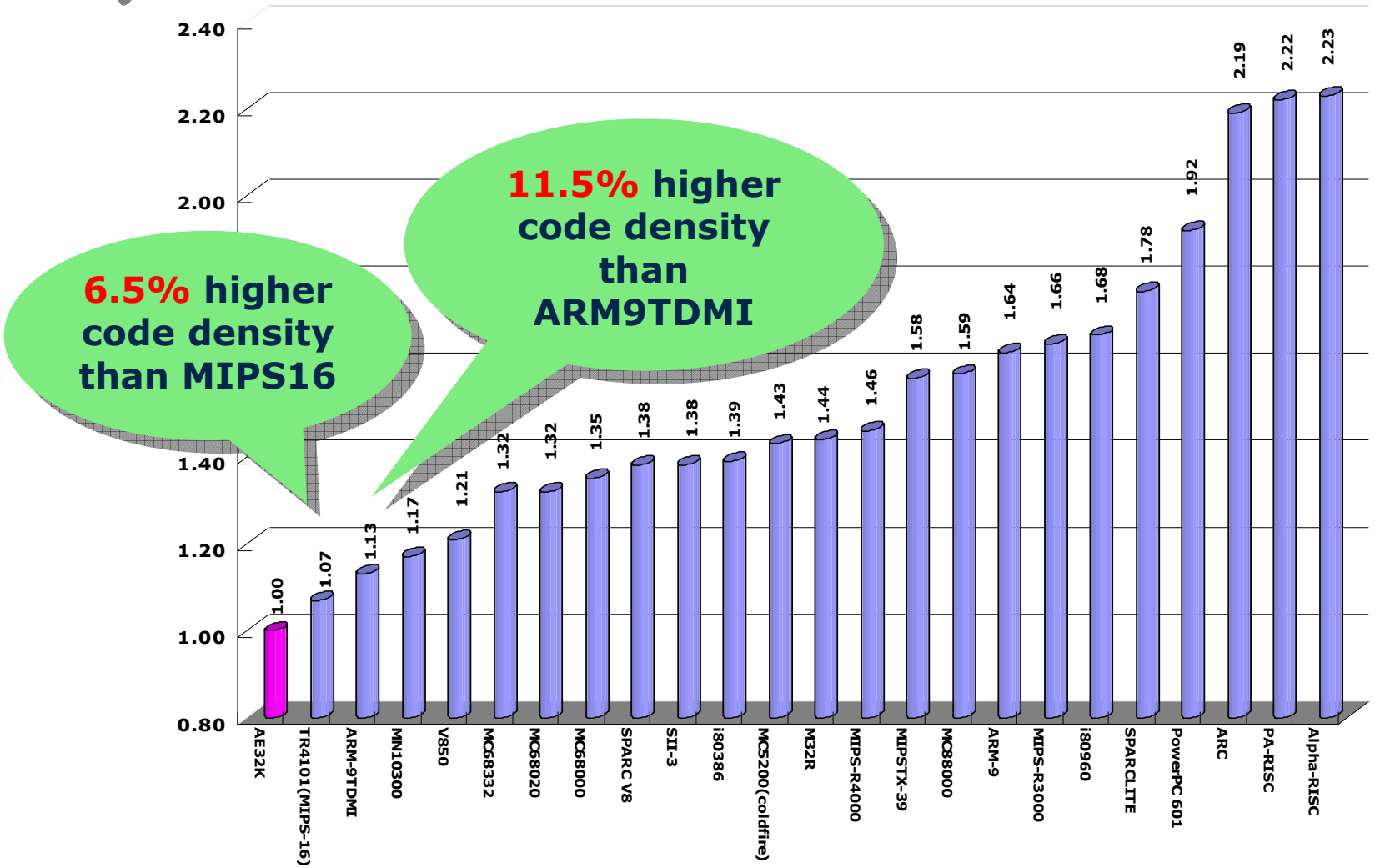


Impact of LERI

- EEMBC: Automotive, Consumer, Telecomm, Network, Office



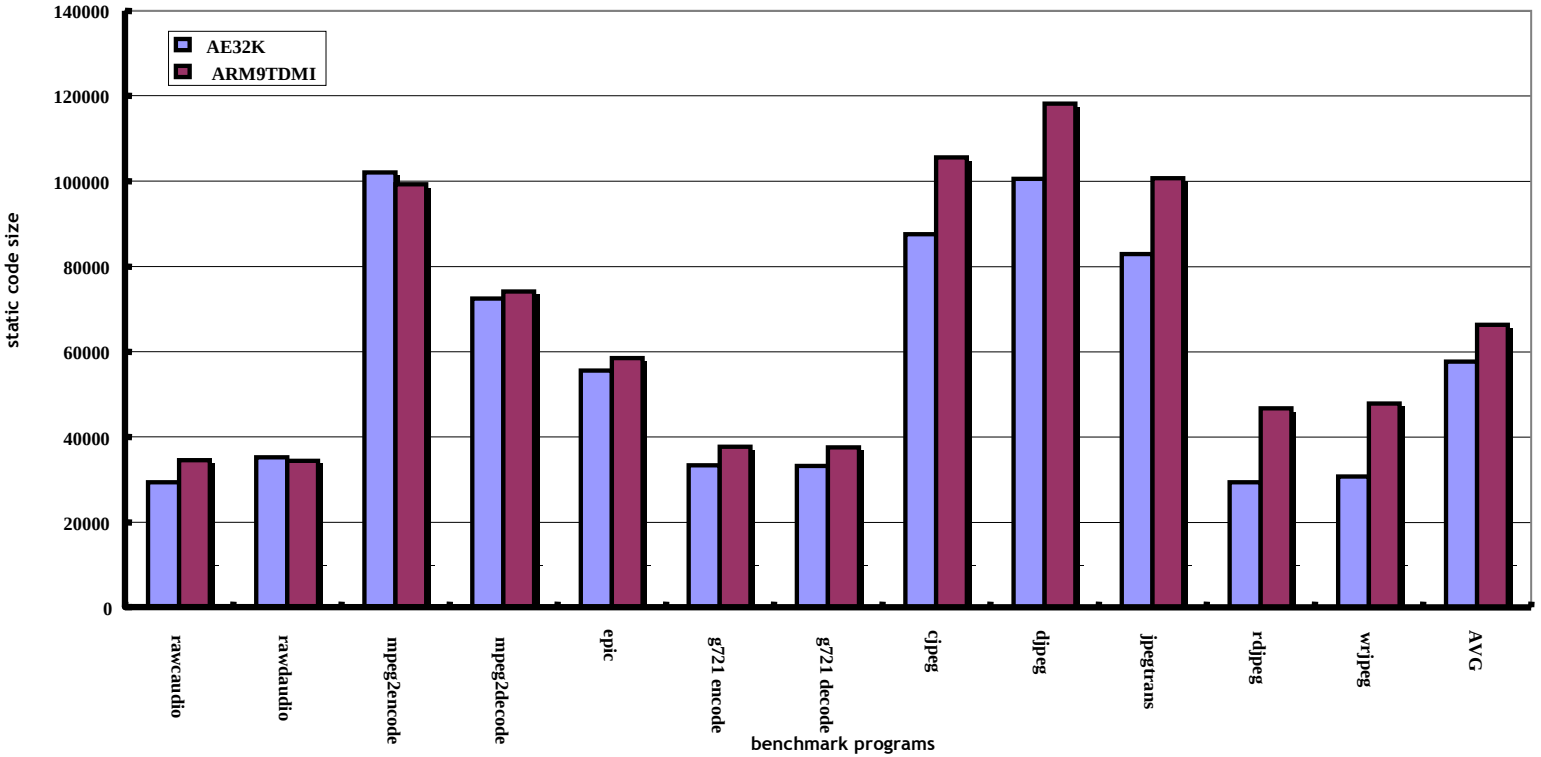
EISC Architecture



Benchmark: libc, libm, libstdc++

EISC Architecture

- Average18.9% higher code density
  - Mediabench, gcc3.2



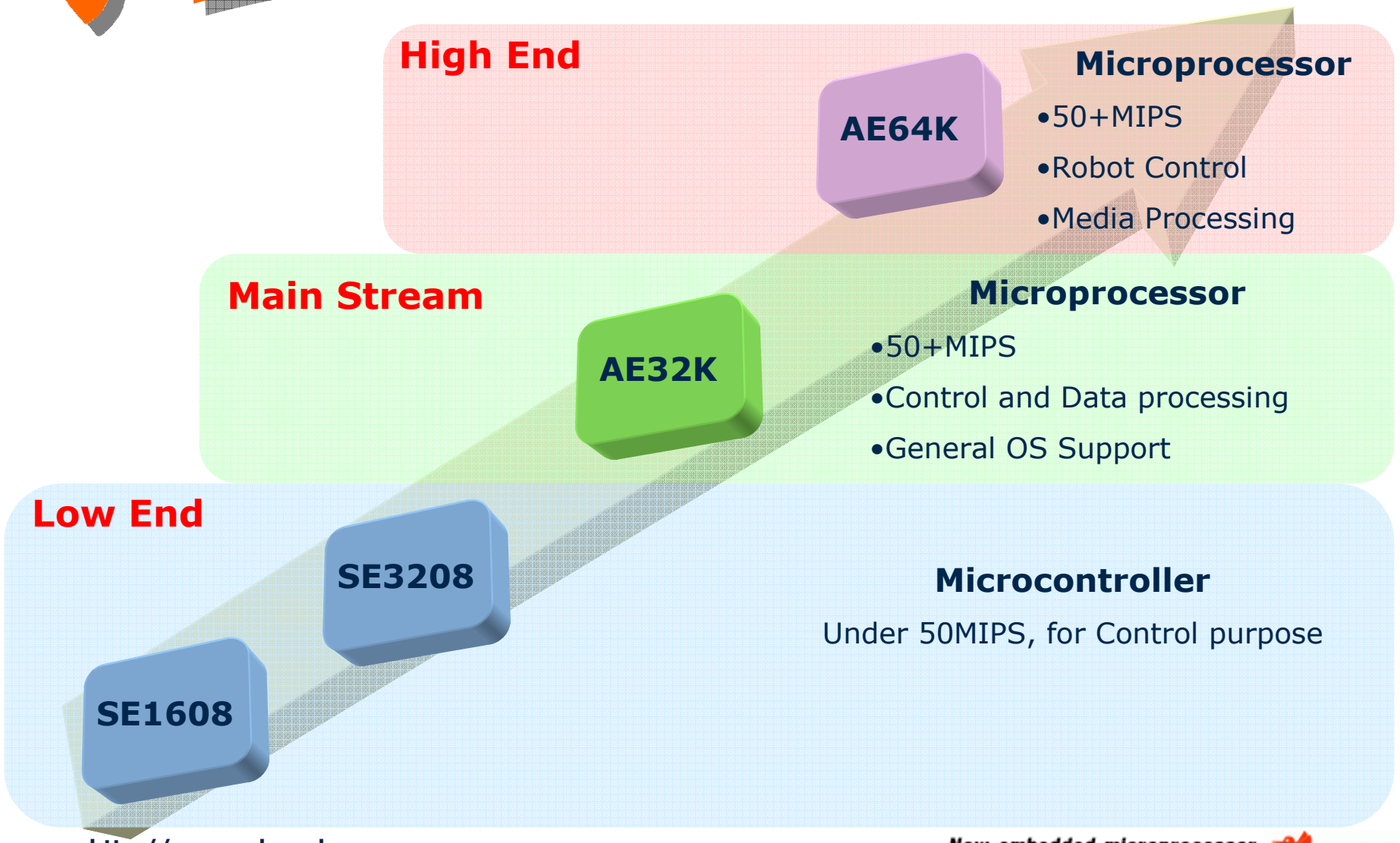
Memory Access

- 16GPRs
  - Most of Compressed Code RISC; 8 GPRs
- EISC can make long immediate easily
  - Most normal RISCs have multiple commands for 32bit immediate value
    - ✓ LUI, ORI sequence(MIPS)
    - ✓ memory LOAD

35% less data-memory access than ARM9TDMI & MIPS16

	AE32K	TR4101	ARM9TDMI
RCD	1.00	1.07	1.13
Memory Access Ratio	30.2%	48.4%	46.5%

Processor Family



Summary

- EISC architecture
  - We believe it is right ISA for post-PC market
  - LERI instruction
    - ✓ Flexibility
    - ✓ Makes instruction more compactor
  - Better code density

adc
advanced digital chips inc.