Parametric Superscalar Architecture & Time to Market

Faraydon Karim

ST Microelectronics

La Jolla, CA

faraydon.karim@st.com

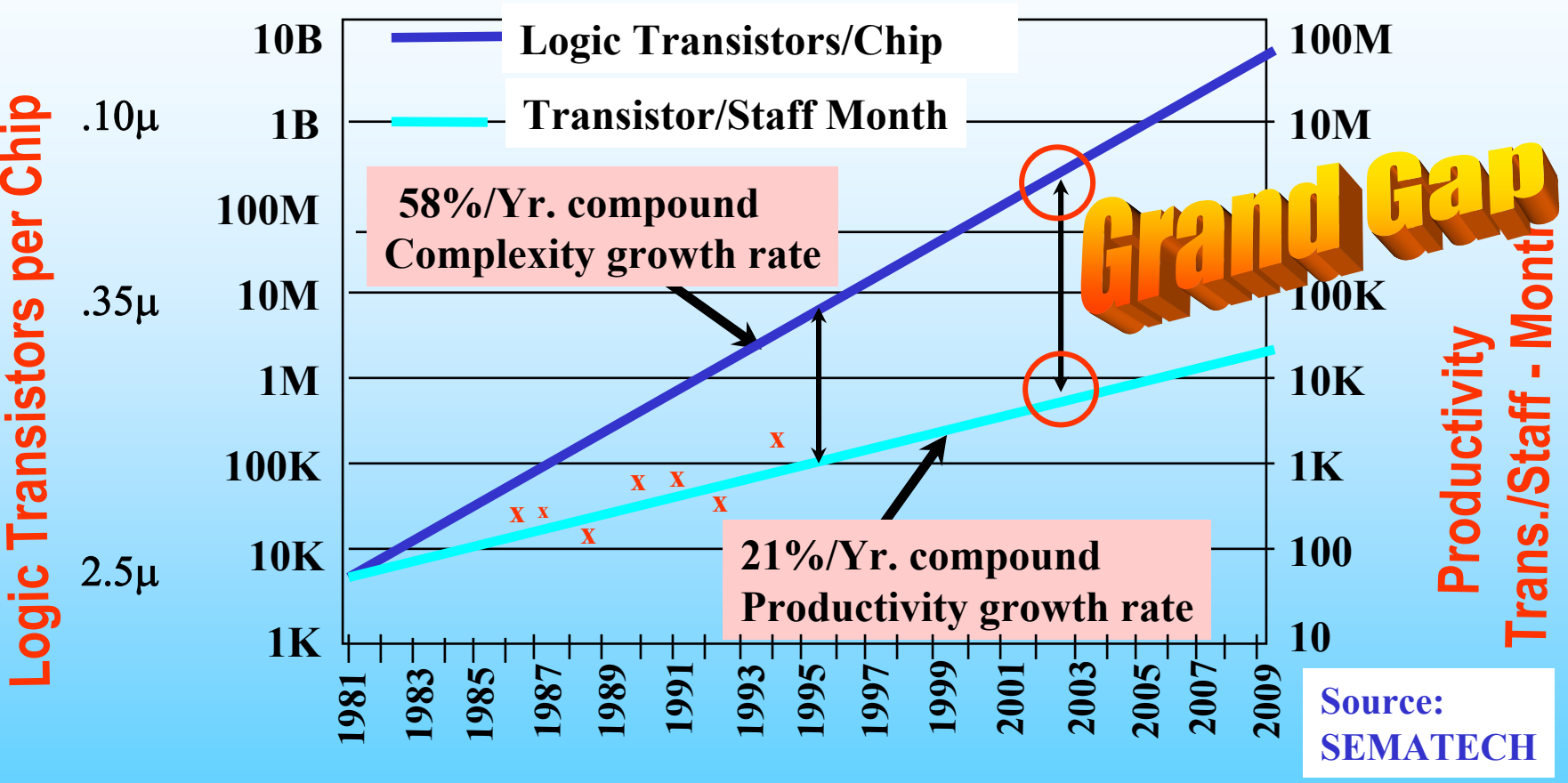
Outline

- Motivation
- Parametric Superscalar Architecture
- STARM Design
- Performance
- What work is left
- Summary

Motivation

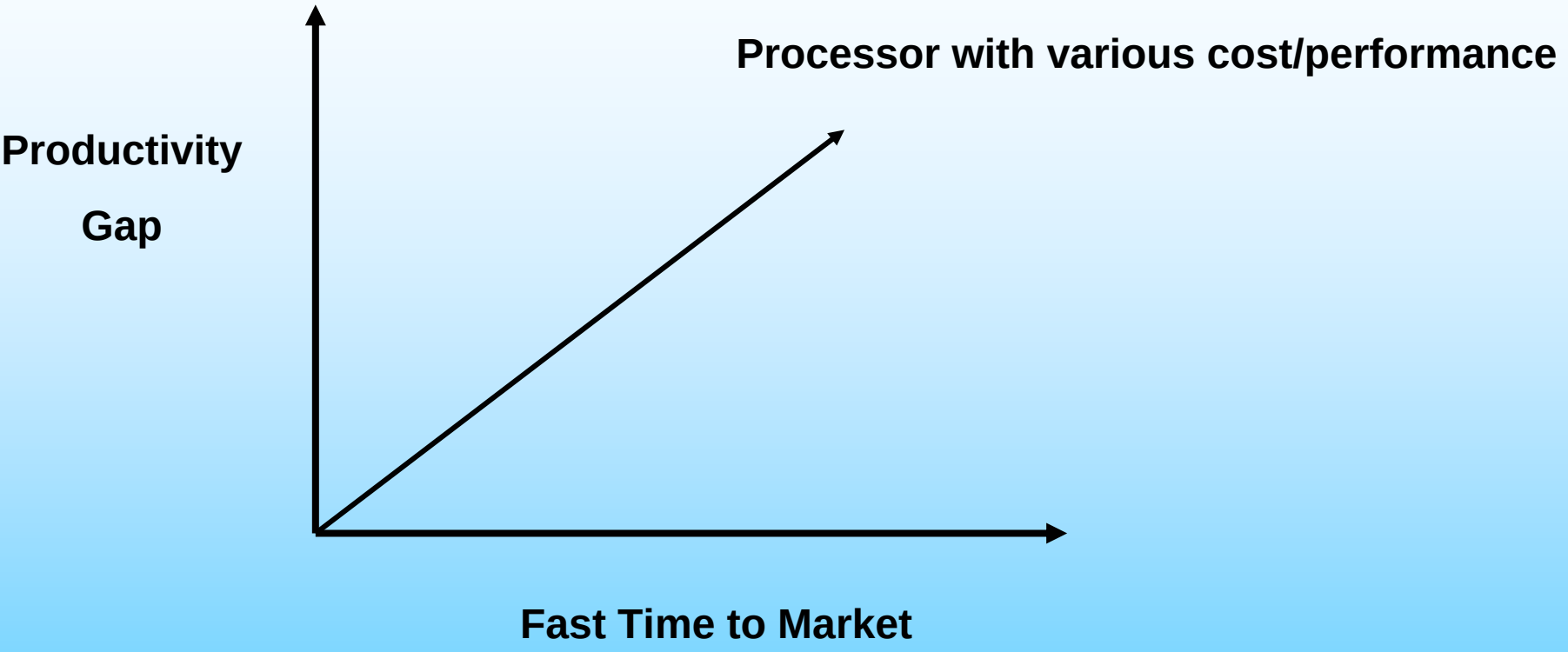
- One Size doesn't fit all.
  - Performance expectancies differ from system to another. The require different horse power from the processor
  - In systems with multiple processors
    - different performance and cost requirements exist

The Productivity Gap



- 100M logic gates in 90 nm -> 1000 ARM7's

Three Opposing Factors



What the Solution?

The Solution

- We Need a design tool that allows a parametric entries
- We need a parametric microarchitecture

Design Tool

- We used SystemC
  - It is high level and for any design block we can make it as a struct.
  - Each struct can be parameterized. Can be replicated us many times us desired. (Physical limitations must be considered)
  - SystemC is also synthesizable.

The tool is there

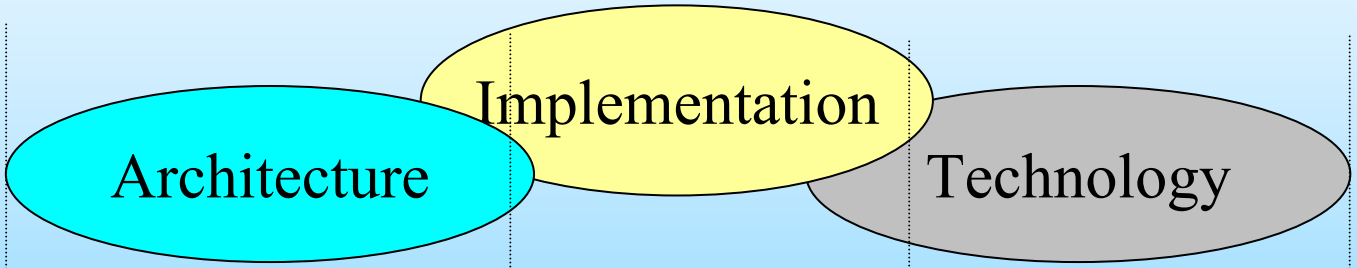
The Microarchitecture

- The most complicated part of the processor is the control logic.
  - If it is build parametrically the rest becomes easy
- The data dependency is also causes interlocking among various parts of design

Microarchitecture

- Superscalar performance is based on two major factors:
  - Code parallelism
    - Code must be able to supply data and instruction to utilize all the resources in efficient way
  - Microarchitecture that can solve
    - Data dependencies such as WAW, WAR, & RAW
    - Solve control dependencies by predicting and anticipating
    - Out of Order Execution to cover data dependency penalties
    - Other techniques

performance



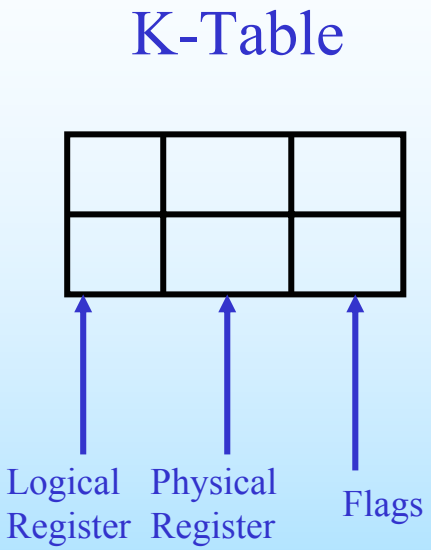
$$\text{Performance} = \frac{\text{Application}}{\#\ \text{Instructions}} \times \frac{\#\ \text{Instructions}}{\text{Cycles}} \times \frac{\text{Cycles}}{\text{Second}}$$

Data Dependencies

K-Table is made of a number of entries each dedicated for on instruction issued. It is initialized by the pointer to the rename register.

A machine with n instruction issue requires n entry in the K-Table

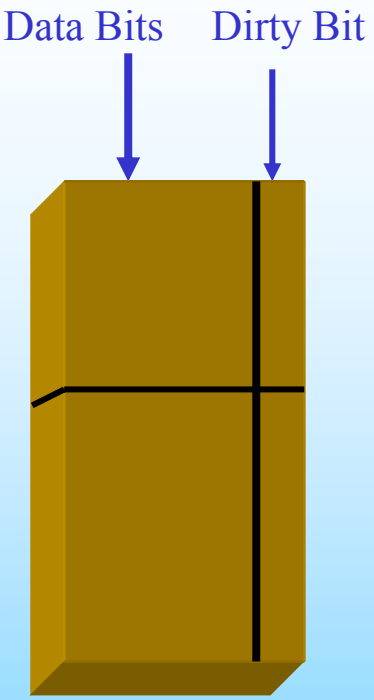
A processor with J number of pipe line stages requires J number of K-Tables



Total Rename Registers = J X n

Solving WAW & WAR

Data Dependencies -2

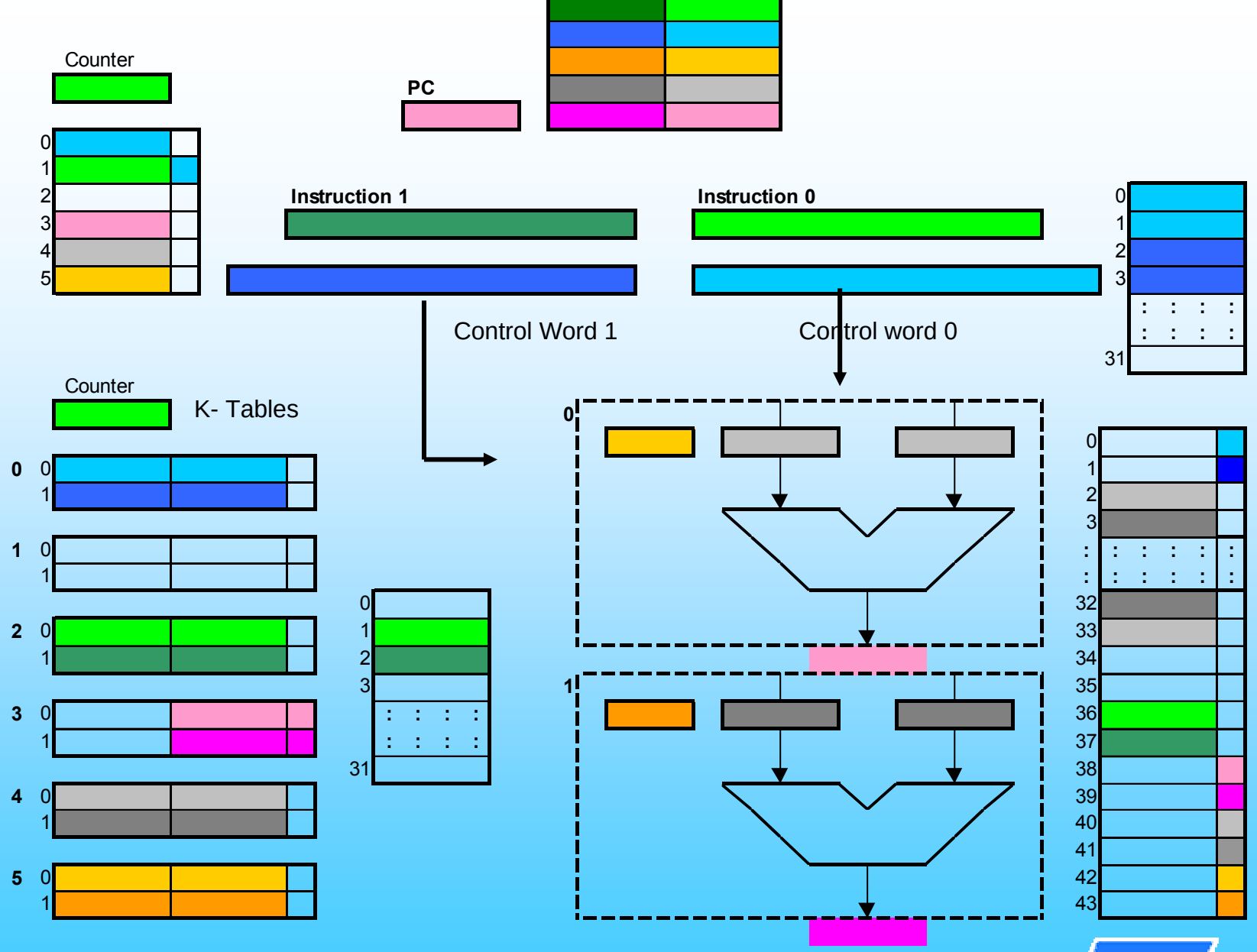


Each entry in the register File
is associated with a dirty bit
When the bit is on means the datum
is not there. when the data gets ready it turns
off. Thus RAW problem is solved.

This technique makes controls
simple and scales with zero
control complexity

Register File & Renaming

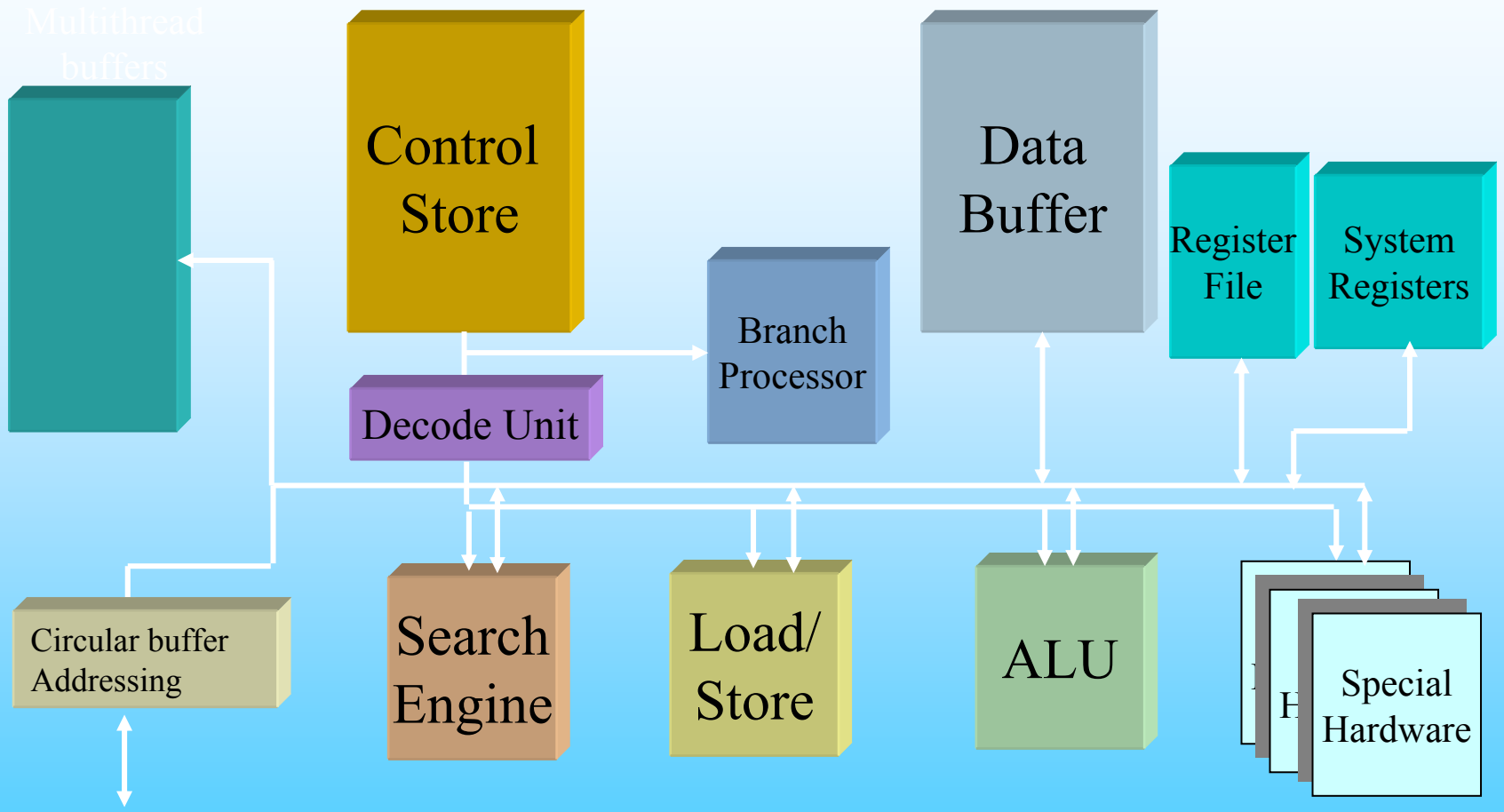
GPR and Rename is mixed. There is no need
for multiple data transfers and GPR ordering


Counter
PC
Instruction 1
Instruction 0
Control Word 1
Control word 0
Counter
K- Tables

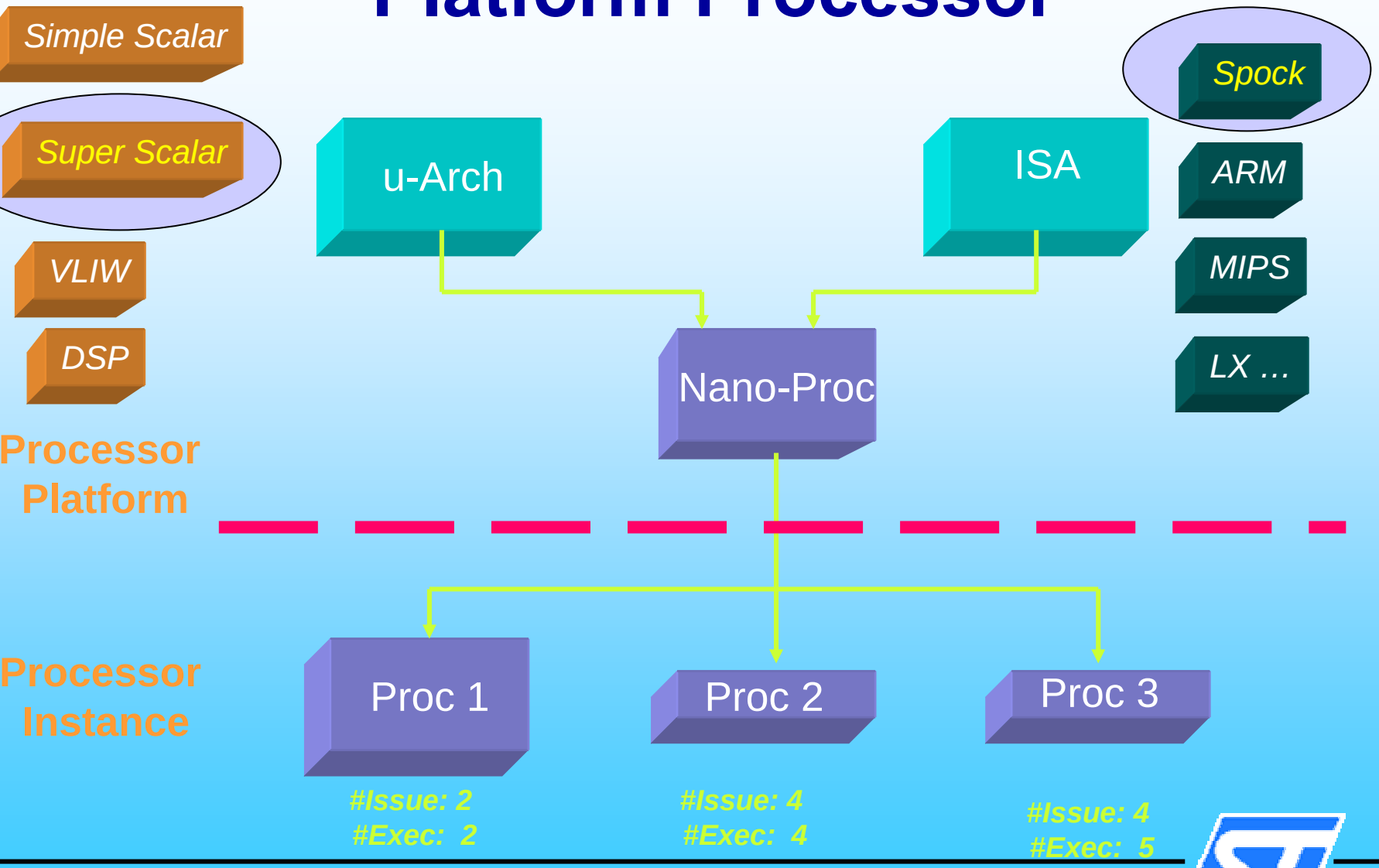
The Result

- Highly Efficient Superscalar with no complex control logic
  - Easy design
- The Parameter are measurable there will be no guess work
  - Using simple equations for decision making
- Create execution units independent of control logic
  - Allows for scaling and free addition or deletion of units

Nano-Processor Programming Model



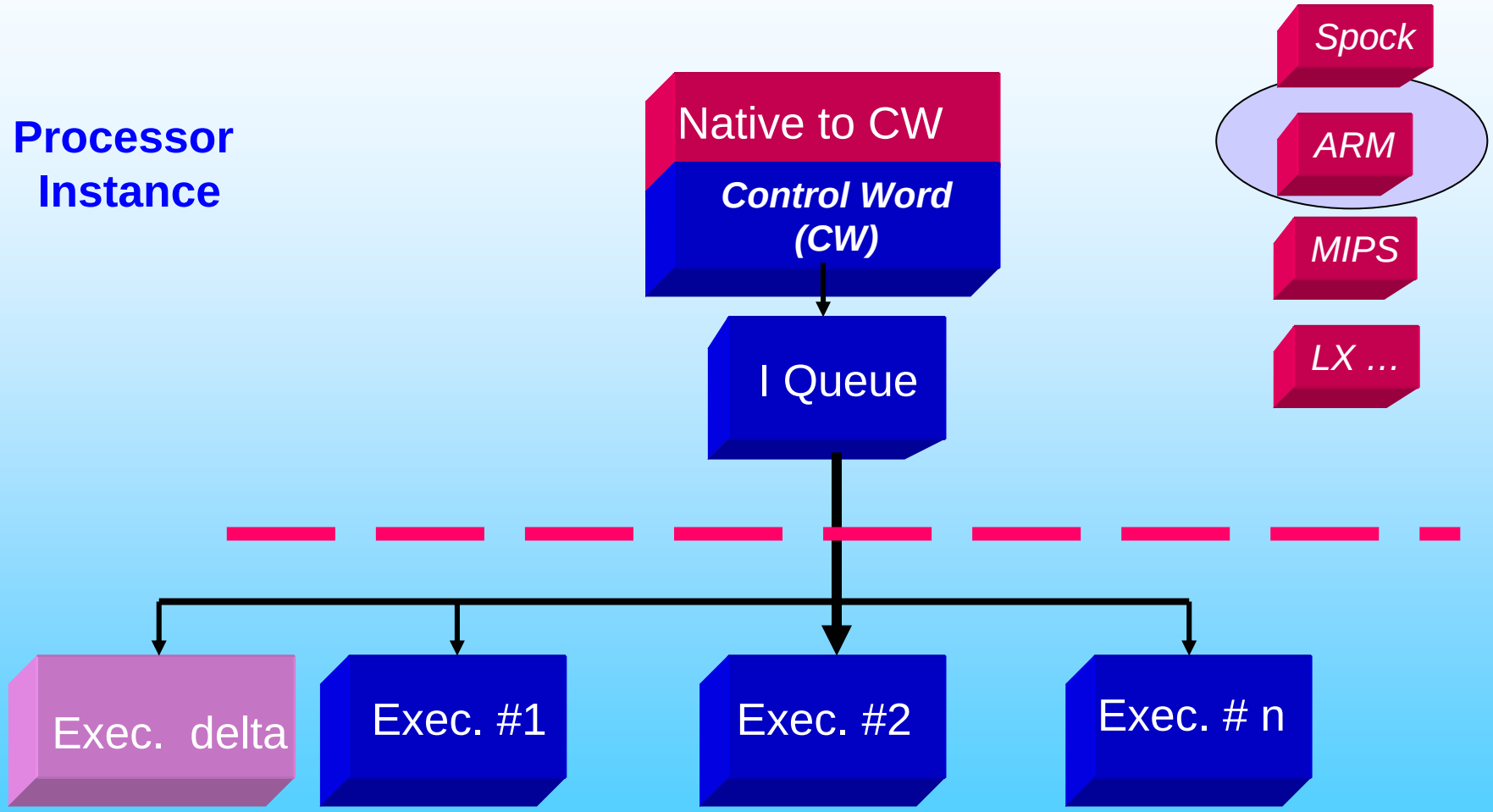
Platform Processor



ISA

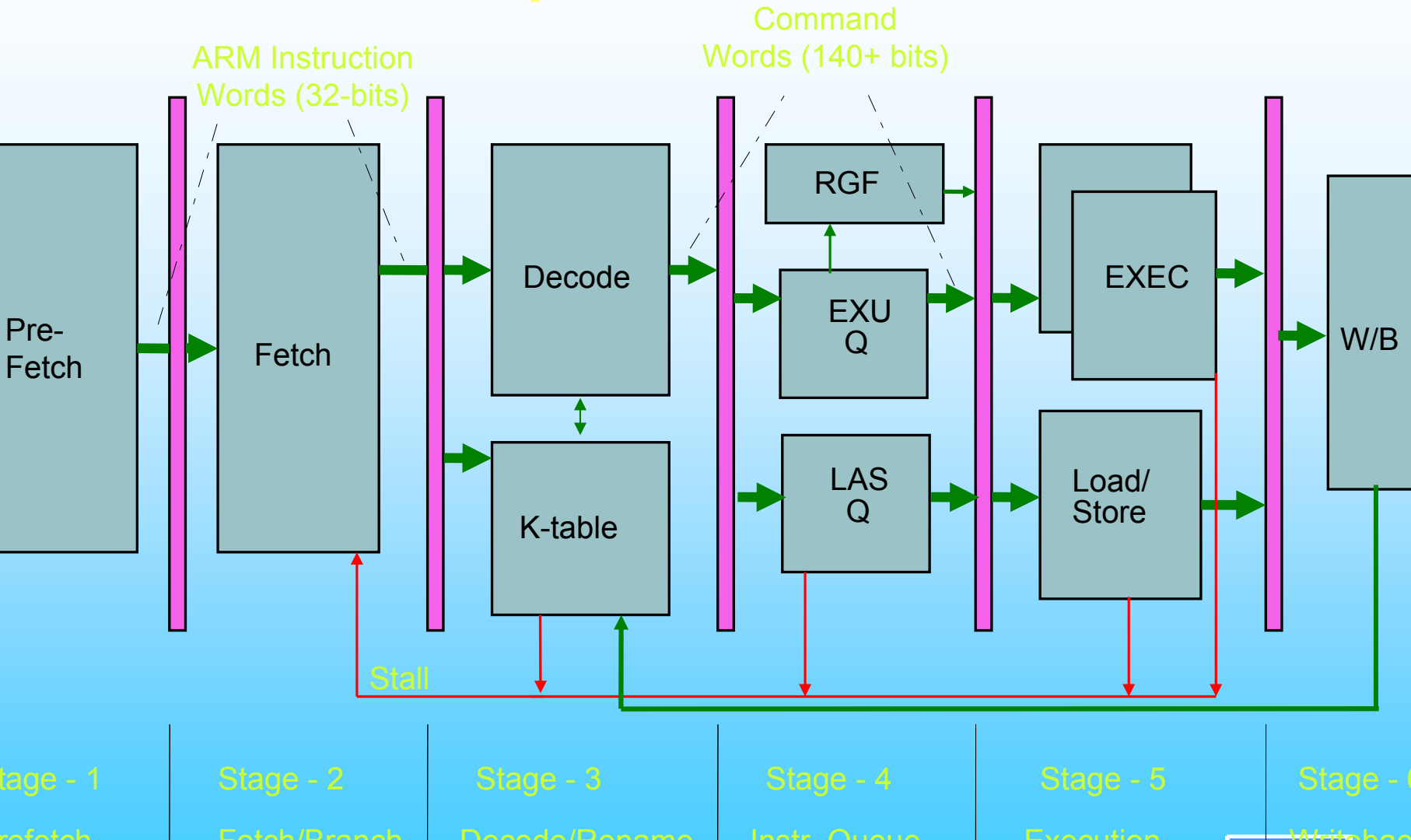
Platform Processor

Processor Instance



STARM

Superscalar ARM

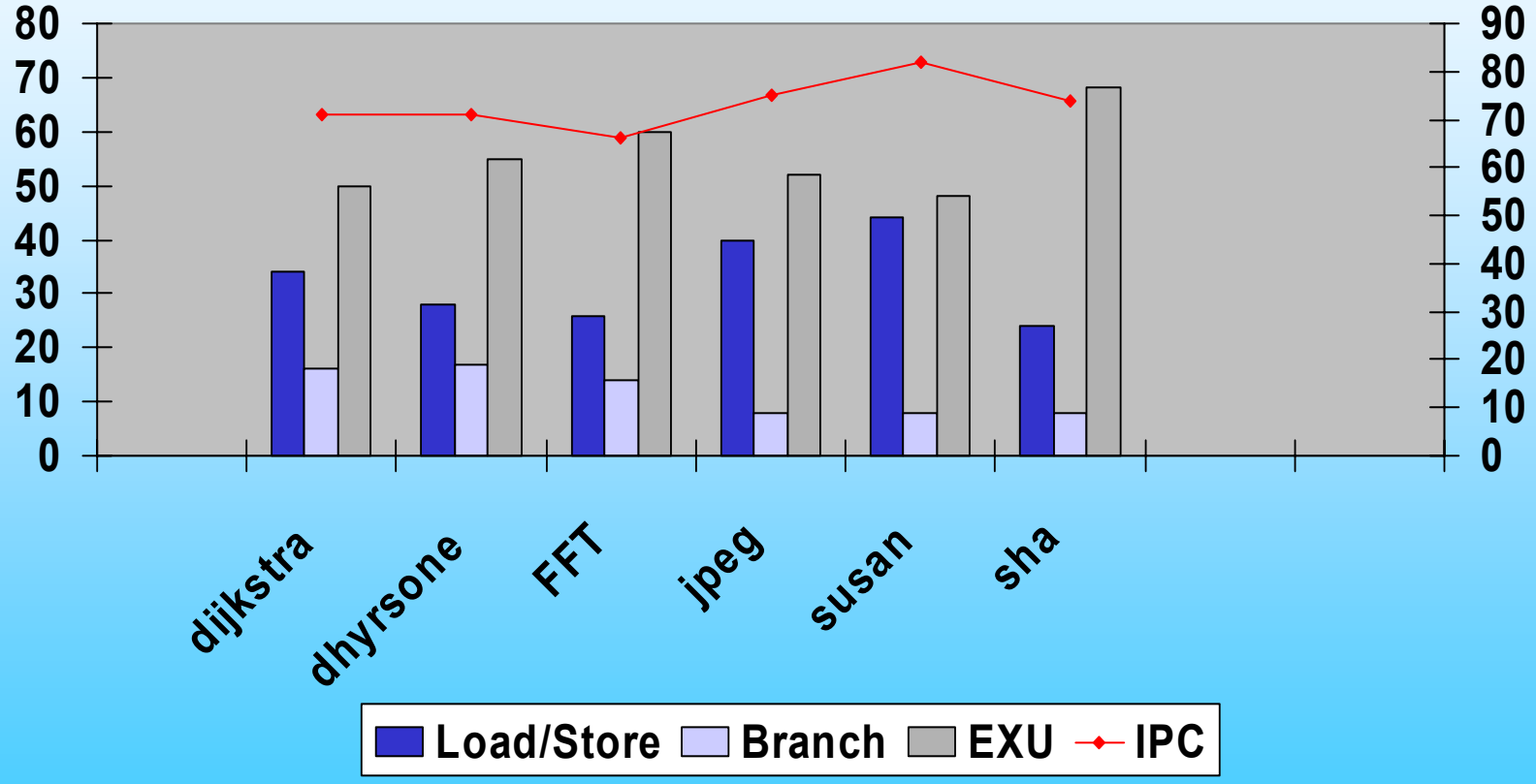


Verification Complexity

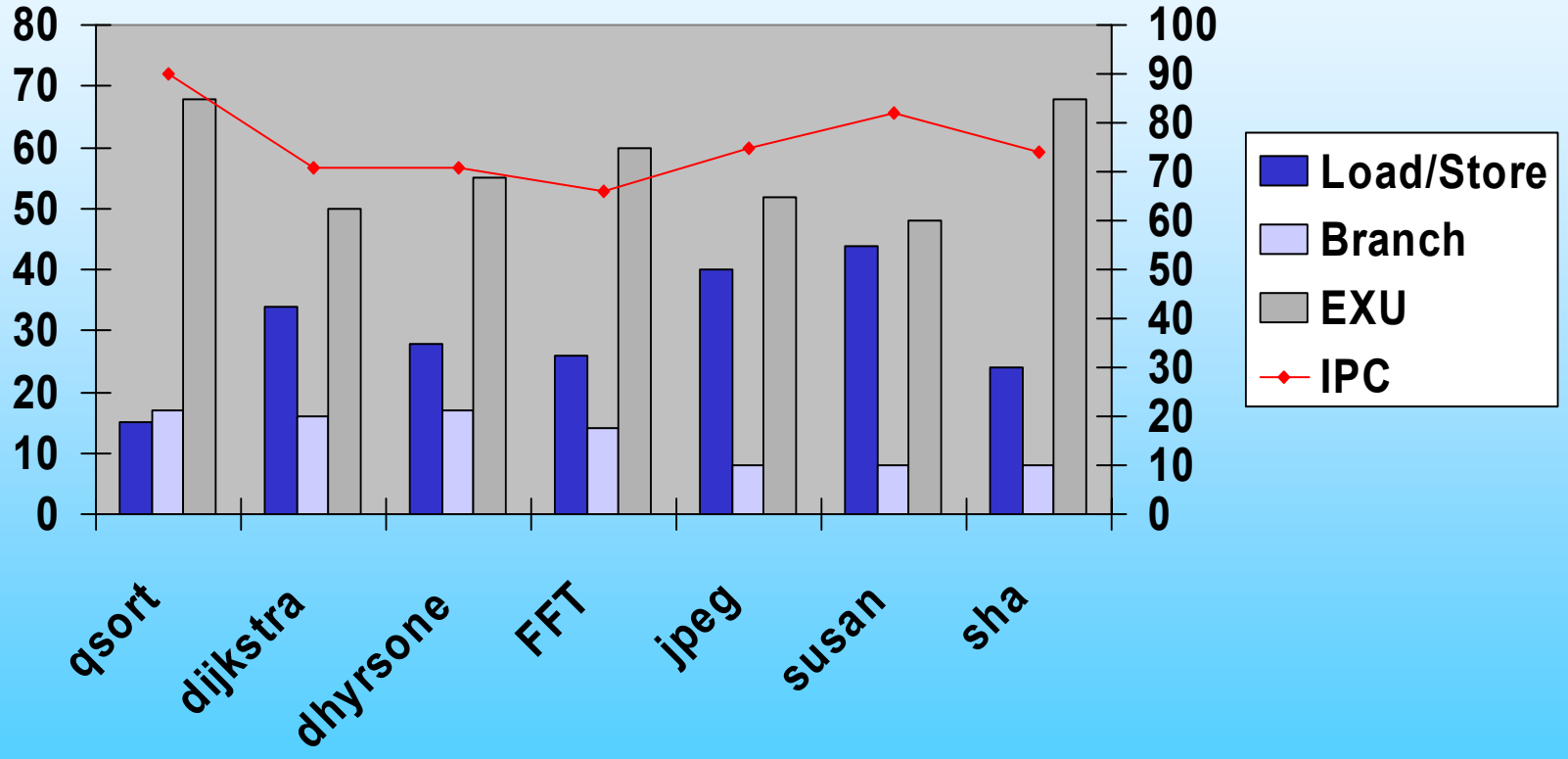
- Verification is divided to three levels:
  - Architectural Verification Program(IVP)
    - Verifies all the instructions and architectural parts that are observable
  - Implementation Verification Program (IVP)
    - All the testcases that were written for one block was capable to run on all the similar block in any combination.
      - Minimum development required

      Need improvements
  - Application Verification Program (AVP)

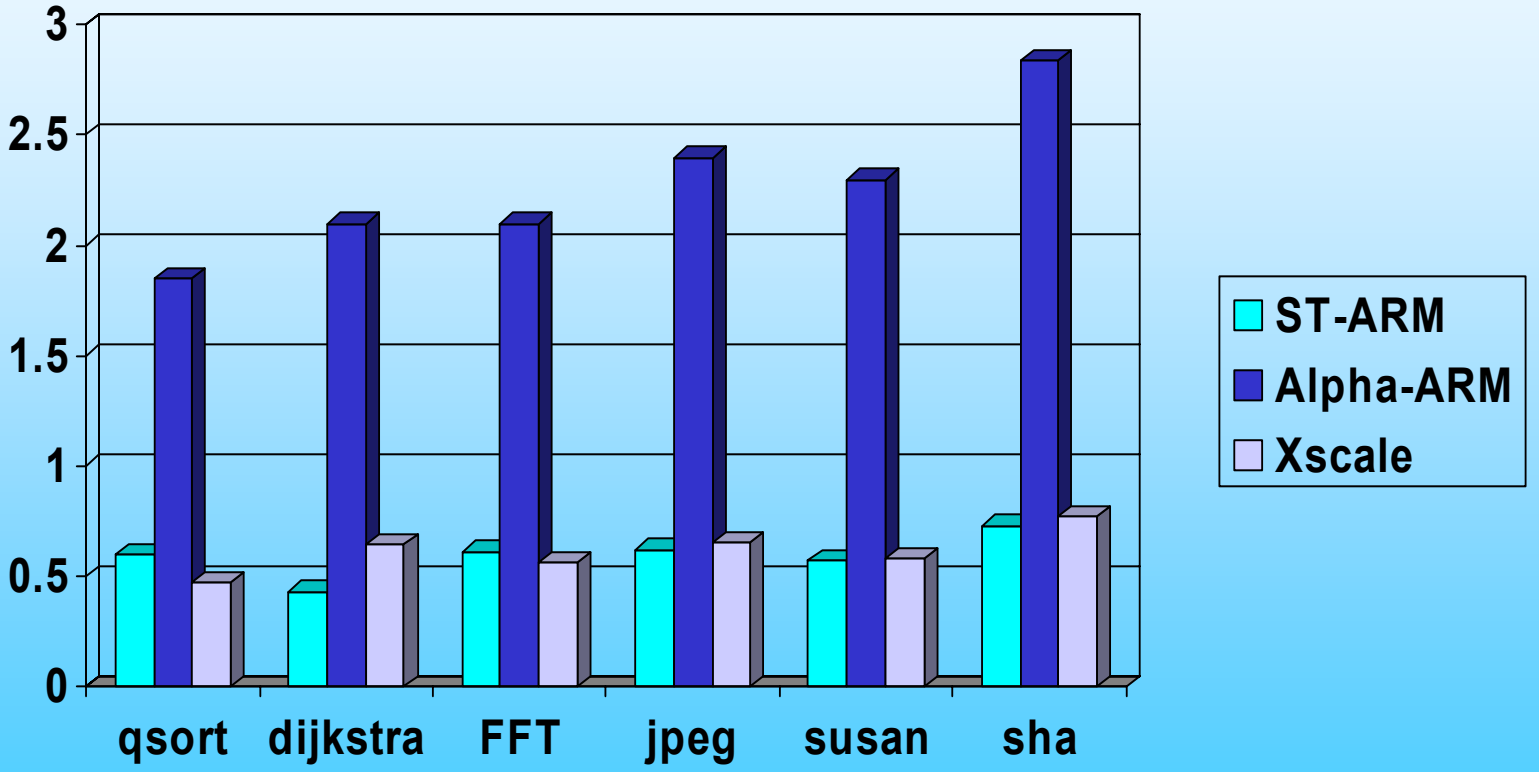
Instruction Distribution-IPC (Instruction count from armsd)



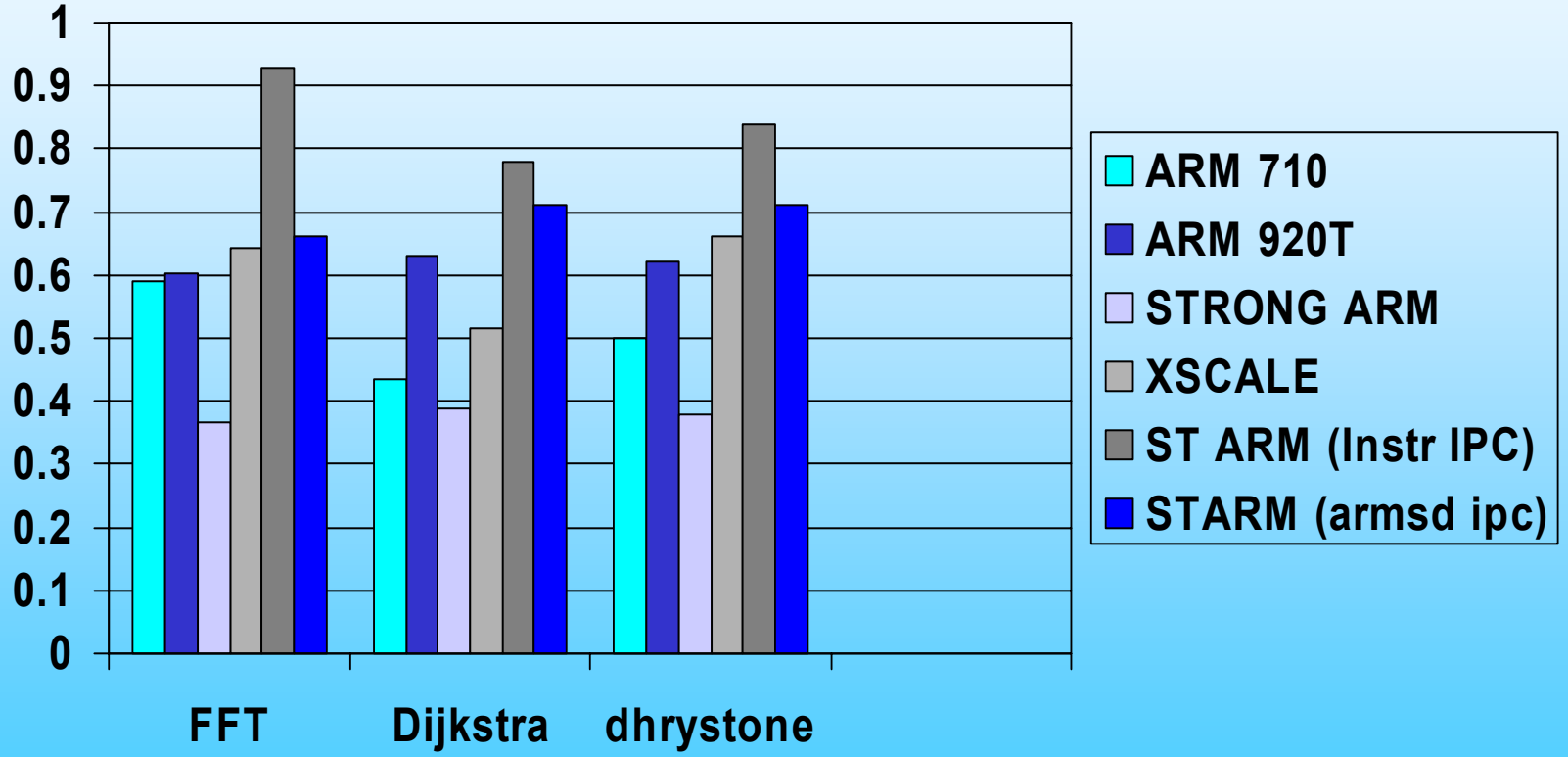
Instruction Distribution –IPC (Instrumented from model)



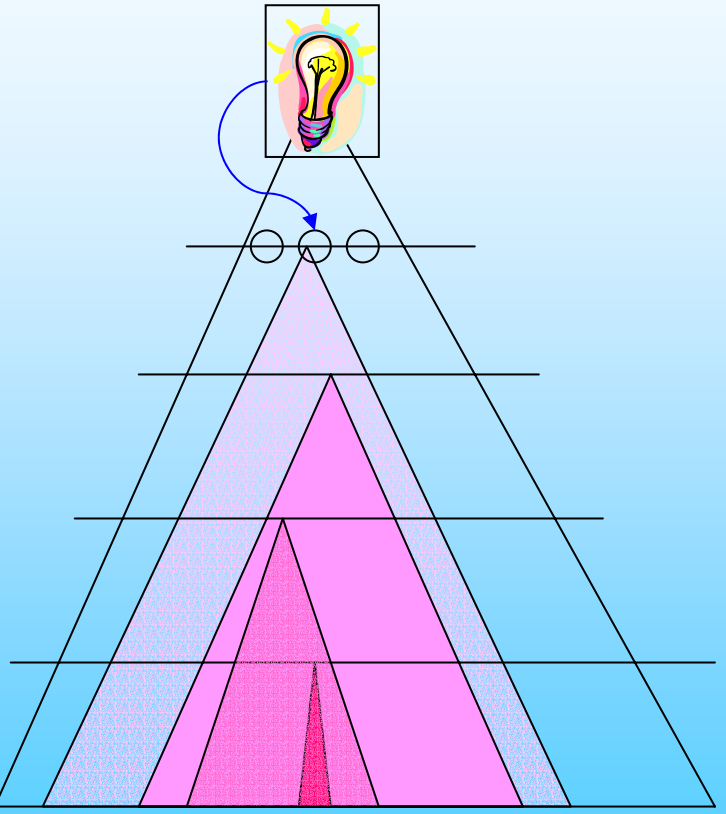
Comparison


3
2.5
2
1.5
1
0.5
0
qsort
dijkstra
FFT
jpeg
susan
sha
ST-ARM
Alpha-ARM
Xscale

ST-ARM – OTHER ARM IPC



What we added is a Fluid-IP



Abstraction Layer	Modeling Type	Re-use Model
Functional	Functional	Functiona
Architecture	Processes	Fluid-IP
RTL / u-Arch	Cycle-based	Soft-IP
Gate-Level	Boolean	Logic-IP
Silicon	Circuit	Hard-IP

Conclusion

- Combination of tool and and parametric architecture created a Fluid-IP.
- Fluid-IP gives the user ability to rapidly develop any variance of processor with minimum cost.
- Fluid-IP is fast path to the Market.
- Parametric Verification process requires more research.