Programming Software Division MCST, Inc.

Sergey L. Shlykov
Lev E. Mukhanov
Vadim D. Gimpelson
Roman Yu. Rogov

Authors are fully responsible for the presented material.
For any questions you might have please refer to Sergey Shlykov (shlykov@mcst.ru)

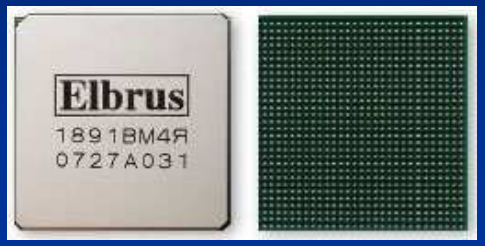
This presentation can be permanently accessed at MosSIGPLAN web-site http://mossigplan.acm.org/

Main fields of research and development

- Industrial optimizing compilers for computer system «Elbrus-3M1» with «Elbrus» microprocessor and computer system «Elbrus 90micro» with «MCST-R» microprocessor
- Automatic parallelization systems for shared and distributed memory
- Binary Translation System Intel x86 – Elbrus
- Verification and debugging technologies

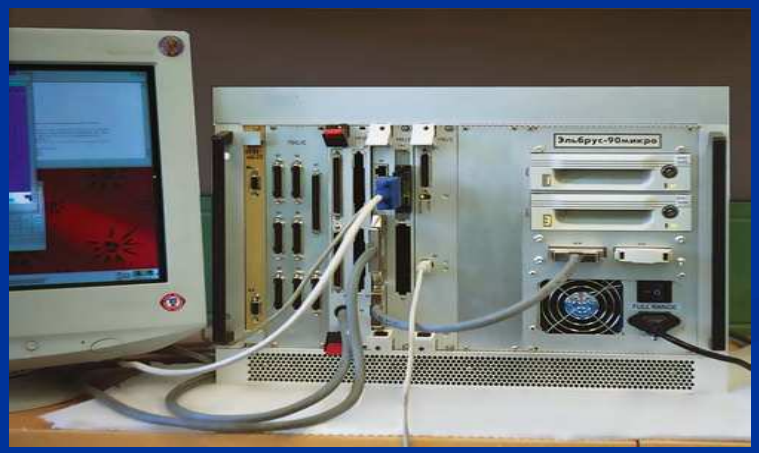
Computer Systems

«Elbrus-3M1» with microprocessor «Elbrus»

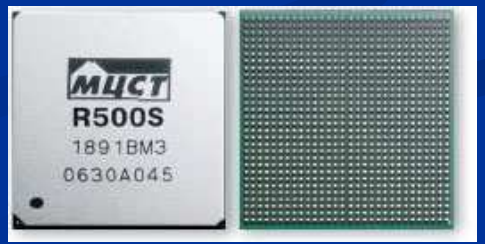


EPIC, original instruction set

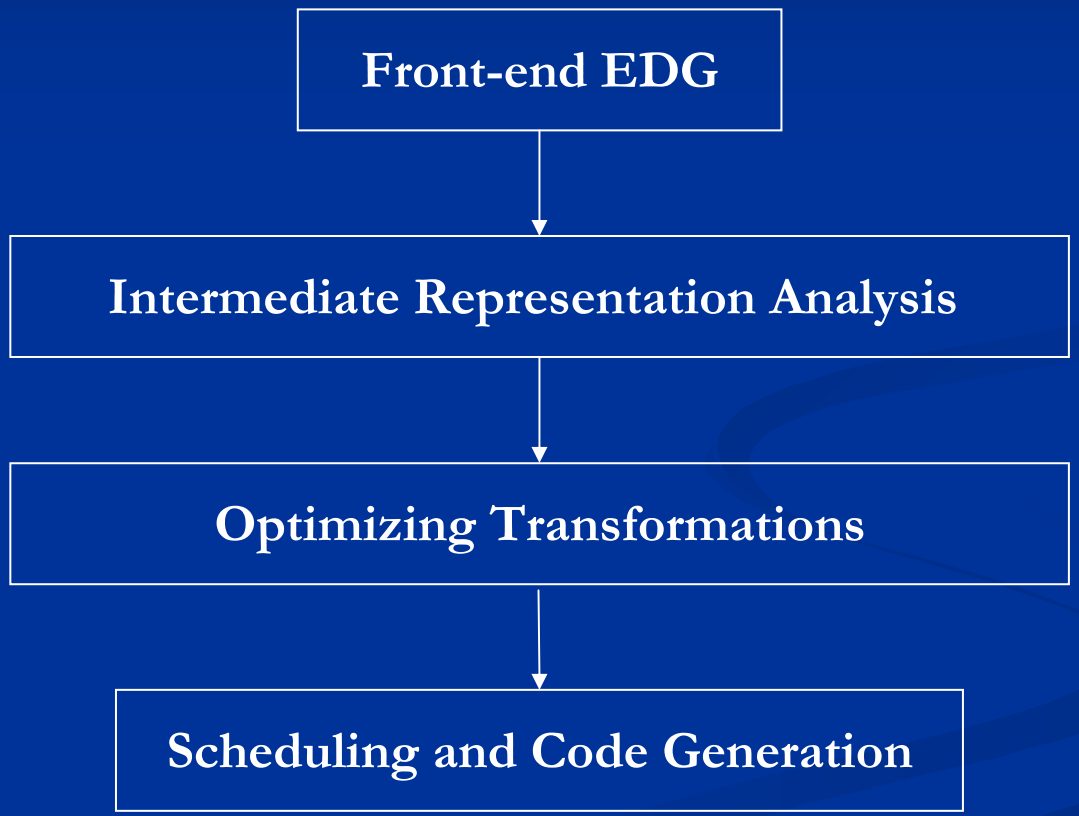
«Elbrus 90micro» with microprocessor «MCST-R500S»



RISC, SPARC V8 instruction set

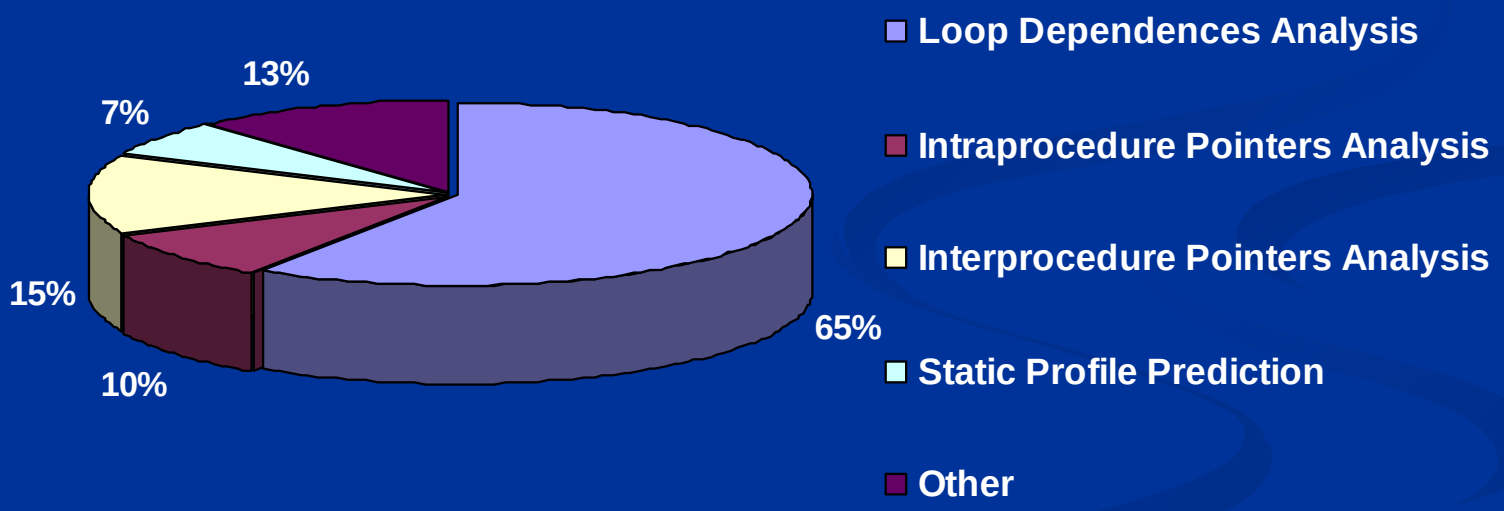


C/C++ Optimizing Compiler



C/C++ Optimizing Compiler

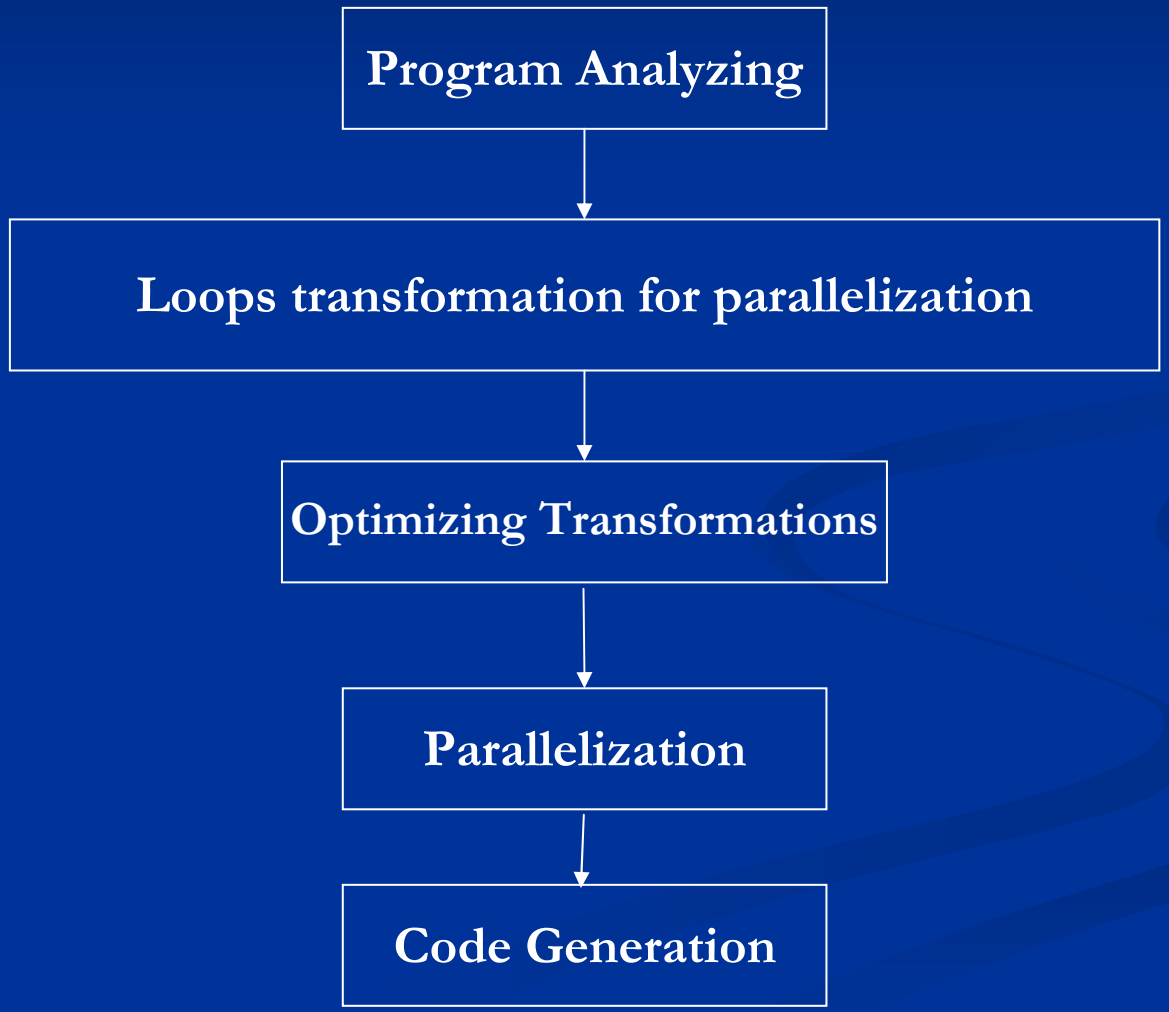
Intermediate Representation Analysis



C/C++ Optimizing Compiler

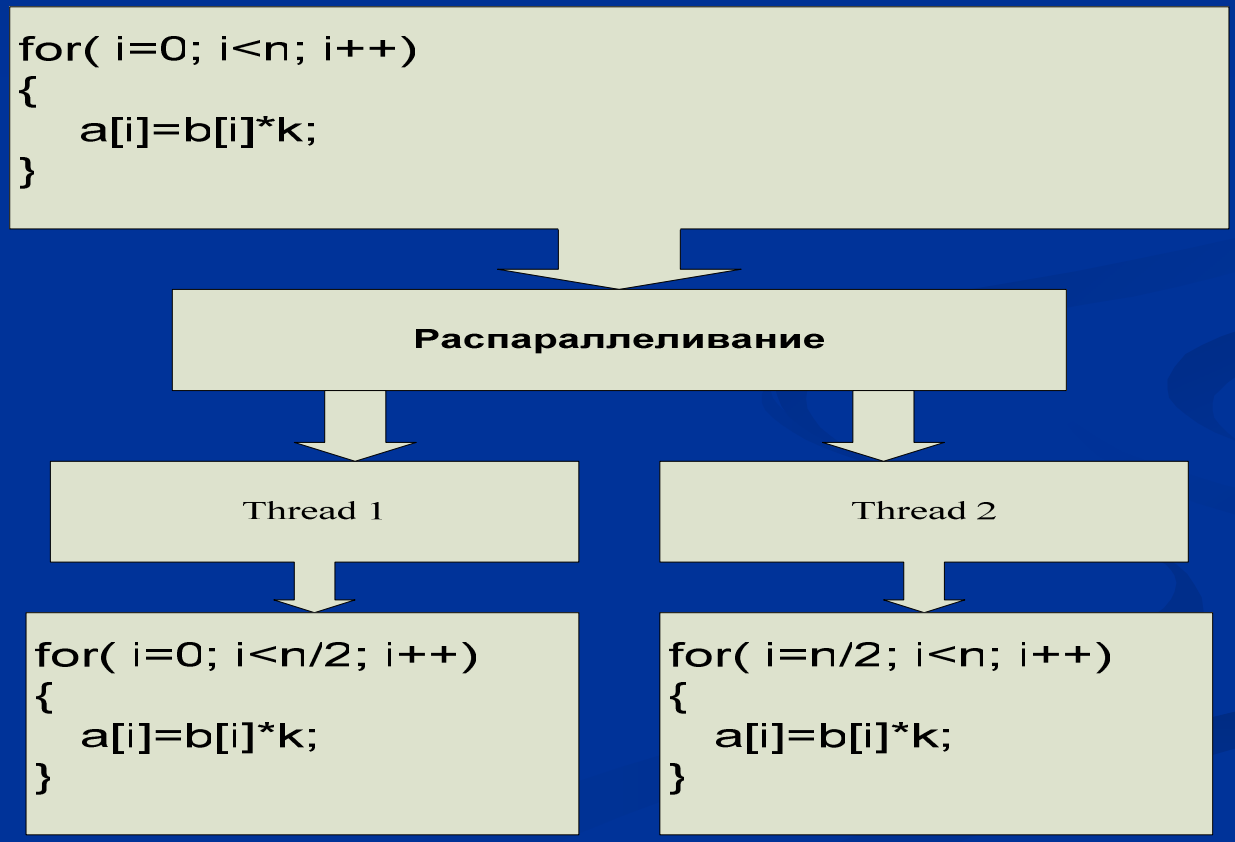
- Global Scheduling
- Data and Control Speculations
- Memory Latency Hiding Methods
- Data Reorganization
- Static Instruction Scheduling with Global Register Allocation

Automatic parallelization

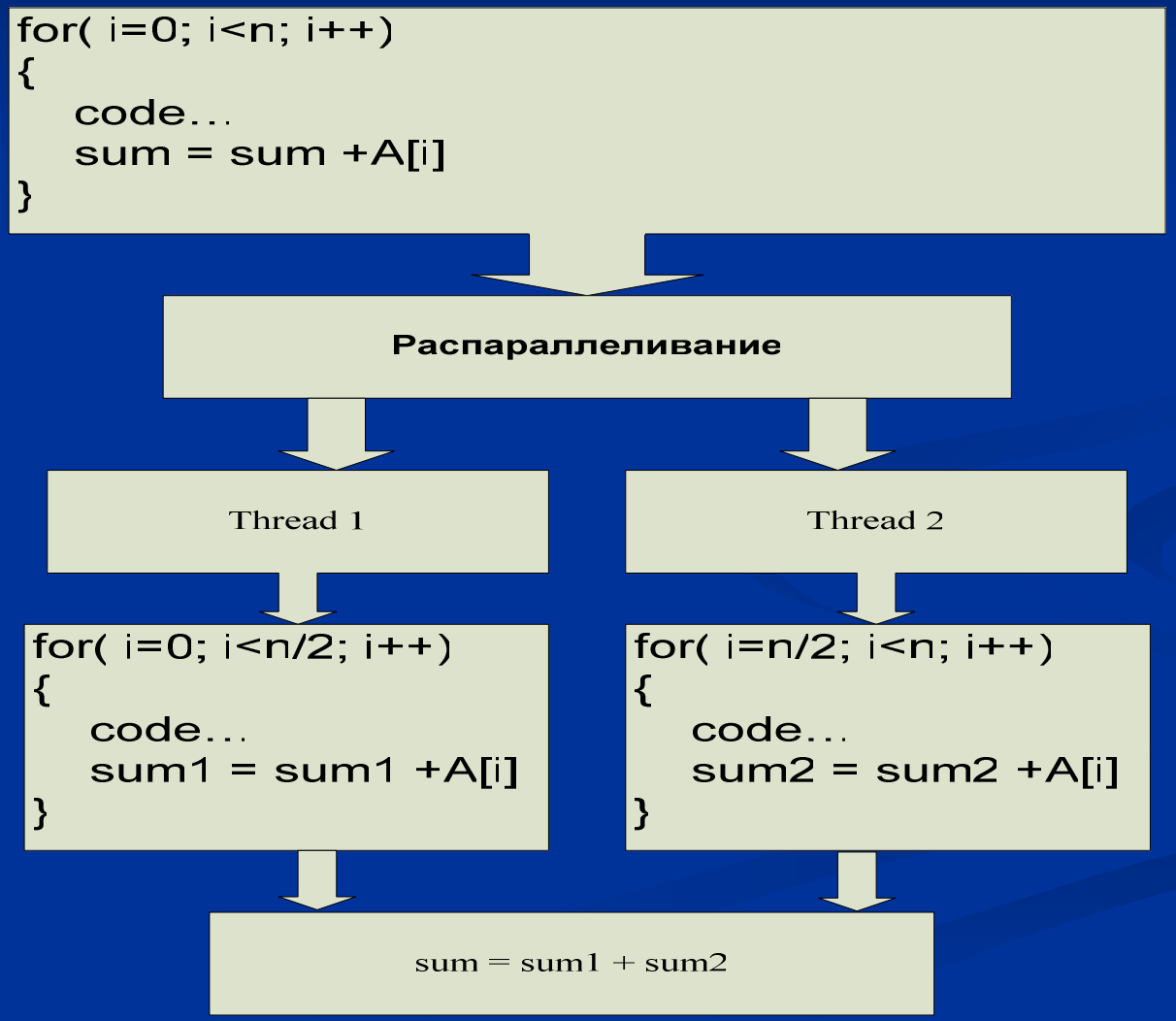


Automatic parallelization

Basic Parallelization Technique



Automatic parallelization



Automatic parallelization

Parallelization methods

- Basic Parallelization Technique
- Recurrence Parallelization

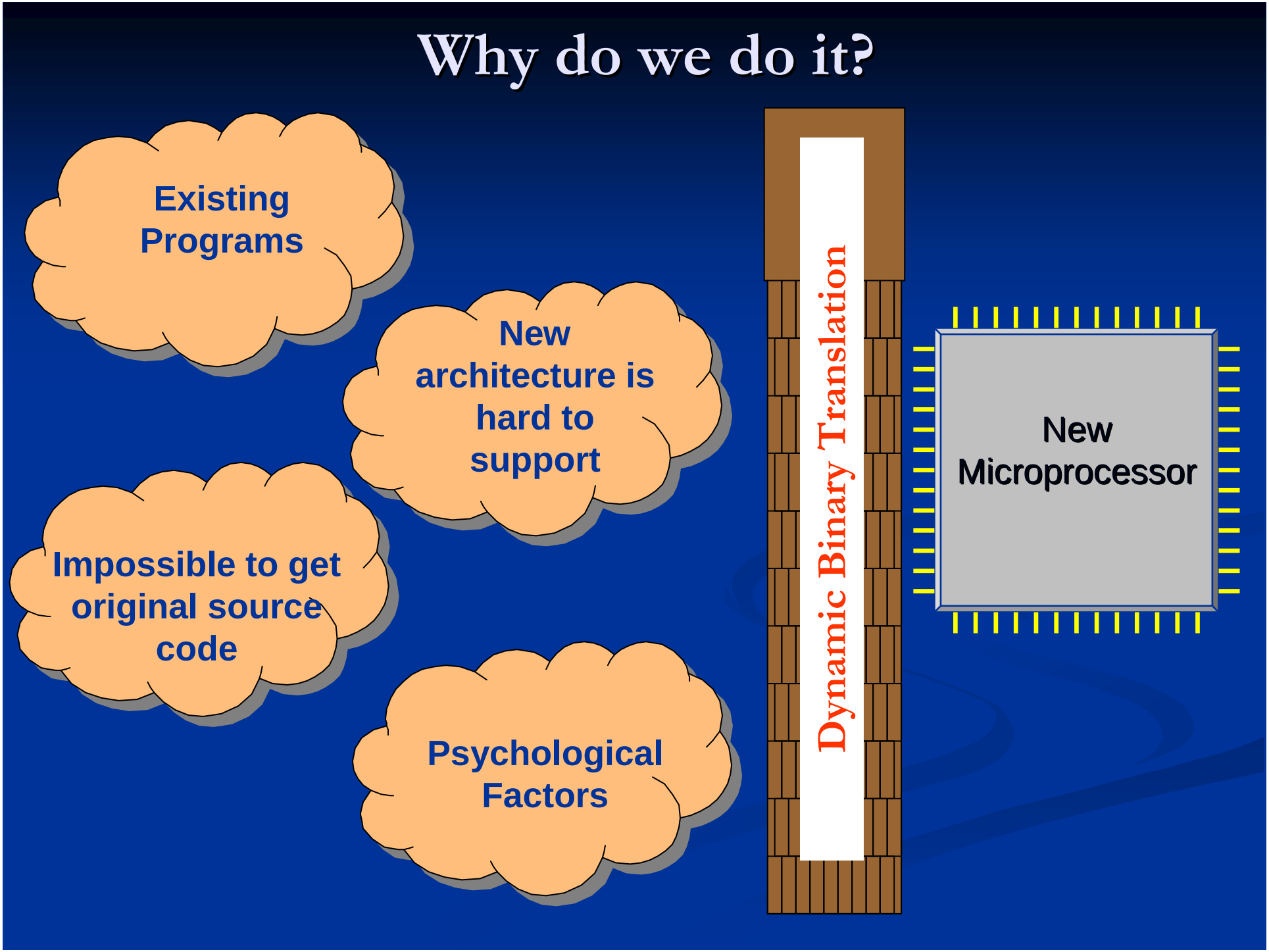
  R = oper(R,F[index]),

  oper=+, *, min, max, |, &, ^.
- Loops Transformation for Basic Parallelization Technique
- Data Prefetching Using Multiple Cores
- Decoupled Software Pipelining

Dynamic Binary Translation System

Binary Compatibility with Intel x86

Why do we do it?



How does it work?

x86-compatible processor
(visible for user)

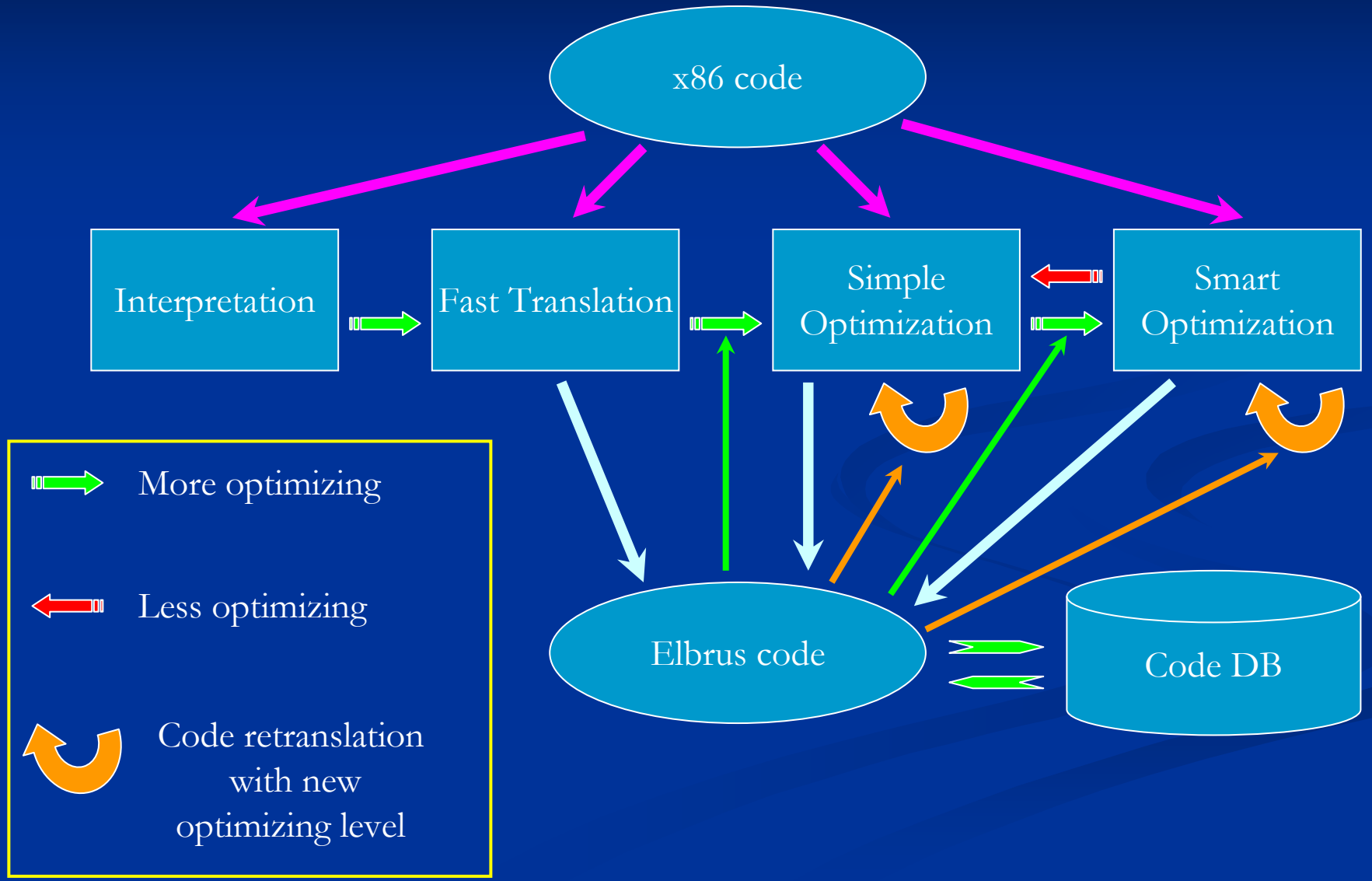
x86 code

Elbrus code

Dynamic Binary Translation System
(hidden from user)

Any operating systems and applications for x86

Dynamic Binary Translation System



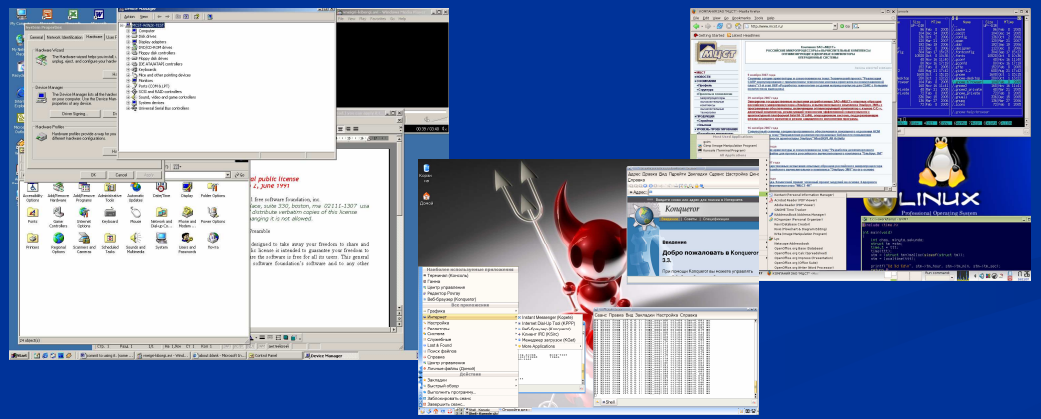
Verified for multiple applications

Computer System
«Elbrus-3M1»

20+
operating
systems

1000+
applications

5 000 000+
Random tests



```
01101001101000101...
add $0x20,%esp
call 805dd98
10100110011001100...
```

Verification and debugging technologies

- Verification vs Debugging
- Verification
  - Intraday Testing ~ 100 uses/day
  - Night Testing
  - Development Quality Assurance
  - Product Release
- Debugging
  - All available source code

Verification Process Control

- Multilevel Verification System
  - Tests Control
  - Resources Management
  - Multilevel Parallelism
  - Different Architectures Support
  - Monitoring utilities
- Permanent enhancement and improving

Test cases development

- Test cases
  - Regression test suites ~ 1500 cases
  - Conformity test suites ~ 50000 cases
    - ISO 9899:1999, ISO 14882:1998 etc.
  - Random test generation ~ 40000 cases/week
  - Functional test suites
  - Common user applications

Verification & Debugging Utilities

- Debugging Utilities
  - Tracing utilities
  - Trace and dump file parsers
  - Random test generators
  - Visualization utilities
- Development Quality Assurance
  - Performance monitoring
  - Source code metrics analysis
  - Source code coverage analysis

Concurrent Project Development

- Project Development Utilities
  - Project changes workflow utilities
  - Version Control Systems
- Development Groups Cooperation
  - Project management utilities
  - Bug tracking utilities
  - Internal documentation management utilities

Q?