Codeerzeugung für eingebettete Prozessoren

(WS 2001/2002)

Dr.-Ing. Daniel Kästner

Lehrstuhl für Programmiersprachen und Übersetzerbau (Prof. Wilhelm)

fon: +49-681-302-5589

kaestner@cs.uni-sb.de

Organisatorisches

- Vorlesung: Do, 14-16
- Übungen:
- Vorlesungsfolien, Literaturlisten, Aktuelles unter

  http://www.cs.uni-sb.de/~kaestner/CG0102.html
- Folien auf Englisch. *Einwände?*

Today

- What is an embedded system?
- What do embedded applications look like?
- What makes embedded processors special?
- Why is the code generation problem important?
- Why is it a problem?

Embedded Systems: Definition

- Embedded processors are embedded in a physical environment and interact with it for measuring or controlling purposes.
- Characteristics of embedded applications:
  - complex interaction with environment (non-deterministic, contineous behavior; signals arrive at enormeous rates; signal processing and information filtering become key issues)
  - high dependability requirements
  - often real-time processing required

Special Case: Real-Time Systems

- In a real-time system, the correctness not only depends on the logical results but also on the timing of the applications.

- Definition (Oxford Dictionary of Computing):
  Any system in which the time at which output is produced is significant.

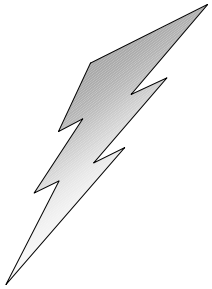
  This is usually because the input corresponds to some movement in the physical world, and the output has to relate to that same movement. The lag from input time to output time must be sufficiently small for acceptable timeliness.

Special Case: Real-Time Systems

- Distinction:
  - Hard real-time system: It is vital that the system satisfies the timing condition. Failure results in catastrophic consequences, e.g. the loss of lifes. Examples: flight control software, airbag control.
  - Soft real-time system: It is desirable that the system satisfies the timing conditions; otherwise the functioning of the system is negatively affected. Example: MP3-Player, telephone software.

Design of Embedded Systems: Goal Conflict

- High performance requirements
- Increasing software complexity
- High dependability



- Low cost
- Low power consumption
- Short product and development cycles

Software Complexity

- The amount of control software in consumer products is doubling roughly every 2 years [B98].

ROM Size/Year	1985	1990	1995	1998
High-end TV	10K	100K	800K	1.4M
Hifi Audio Set	-	10K	100K	256K
VCR	10K	100K	800K	1.2M
GSM phone	-	-	1M	2M
Set-top box	-	2K	128K	1M
Shaver	-	2K	16K	16K

Market for Embedded Processors

- Market for embedded processors estimated at almost $50 billion ($10^9$) in 1997; annual growth estimated at 35% per year (Micrologic Research).
- Worldwide semiconductor sales total $204 billion in 2000 (Semiconductor Industry Association).
- Embedded chips account for more than 90 % of all silicon processors sold [Leibson(Embedded Processor Forum), 2001]
- The global market for general-purpose DSPs grew 25.5% to $4.4 billion in 1999 [Tempe Research Company].

Today

- ✓ What is an embedded system?
- What do embedded applications look like?
- What makes embedded processors special?
- Why is the code generation problem important?
- Why is it a problem?

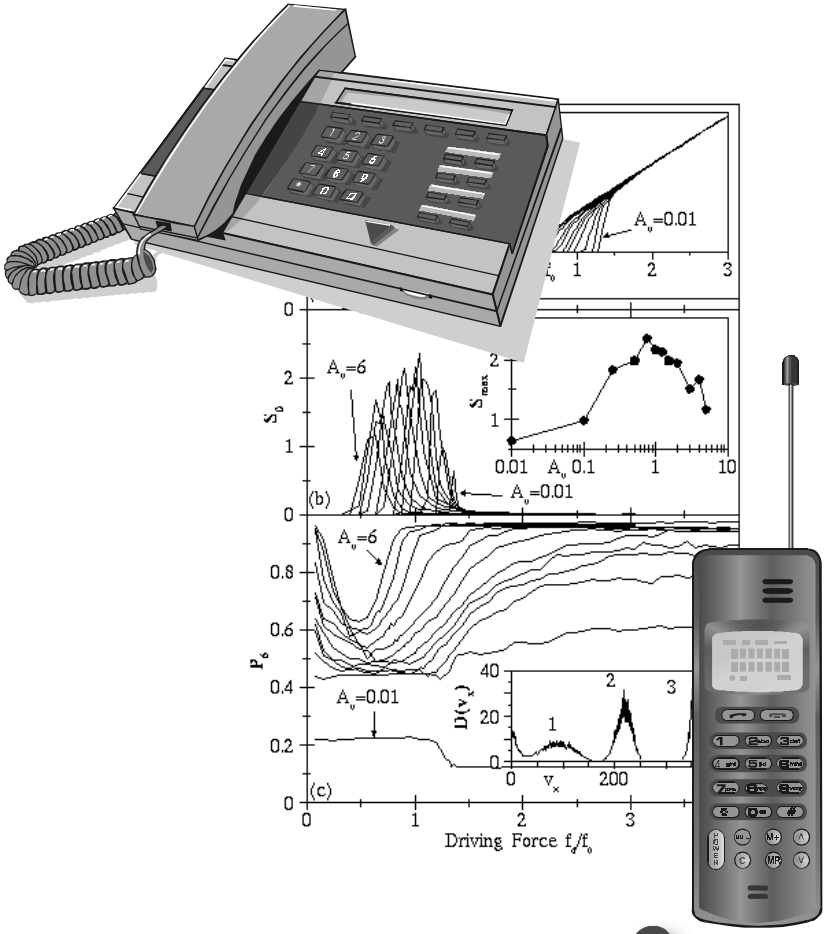
Examples: Consumer Electronics

- AV-R Receivers (e.g. Analog Devices SHARC)
- Smart Pen
- CD-player, DVD-player, MP3-player
- Organizer, PDAs
- Washing machines, microwave ovens, ...
- PC peripherals (hard disk control, graphic cards, ...)

Examples: Telecommunications

- Devices:
  - Cell phones
  - ISDN phones, fax
  - Answering machines, etc

- DSP-Applications:
  - voice and data compression
  - echo reduction
  - signal multiplexing
  - filtering

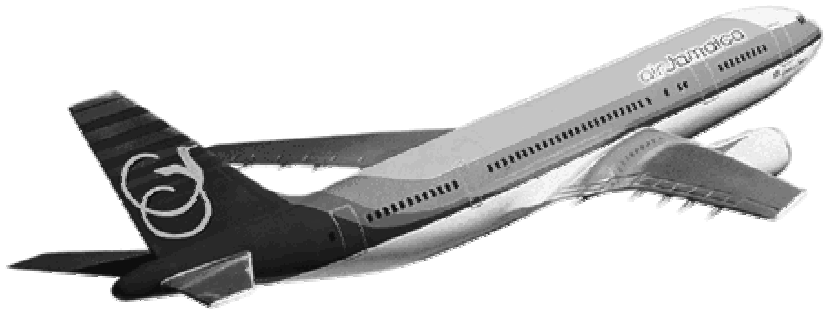


Examples: Automotive

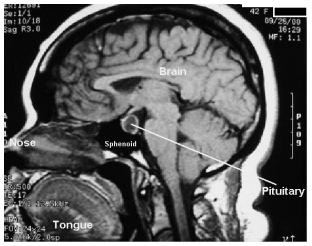
- An average passenger car has roughly the equivalent of six Pentium processors; currently there are up to 100 microprocessors per car.
- Engine control
- Active suspension
- Air-conditioning
- Airbag
- Navigation systems (GPS)
- Sound system, active noise cancellation
- …

Examples: Avionics and Healthcare Technology

- Pilot information systems
- Fly-by-wire
- Security control



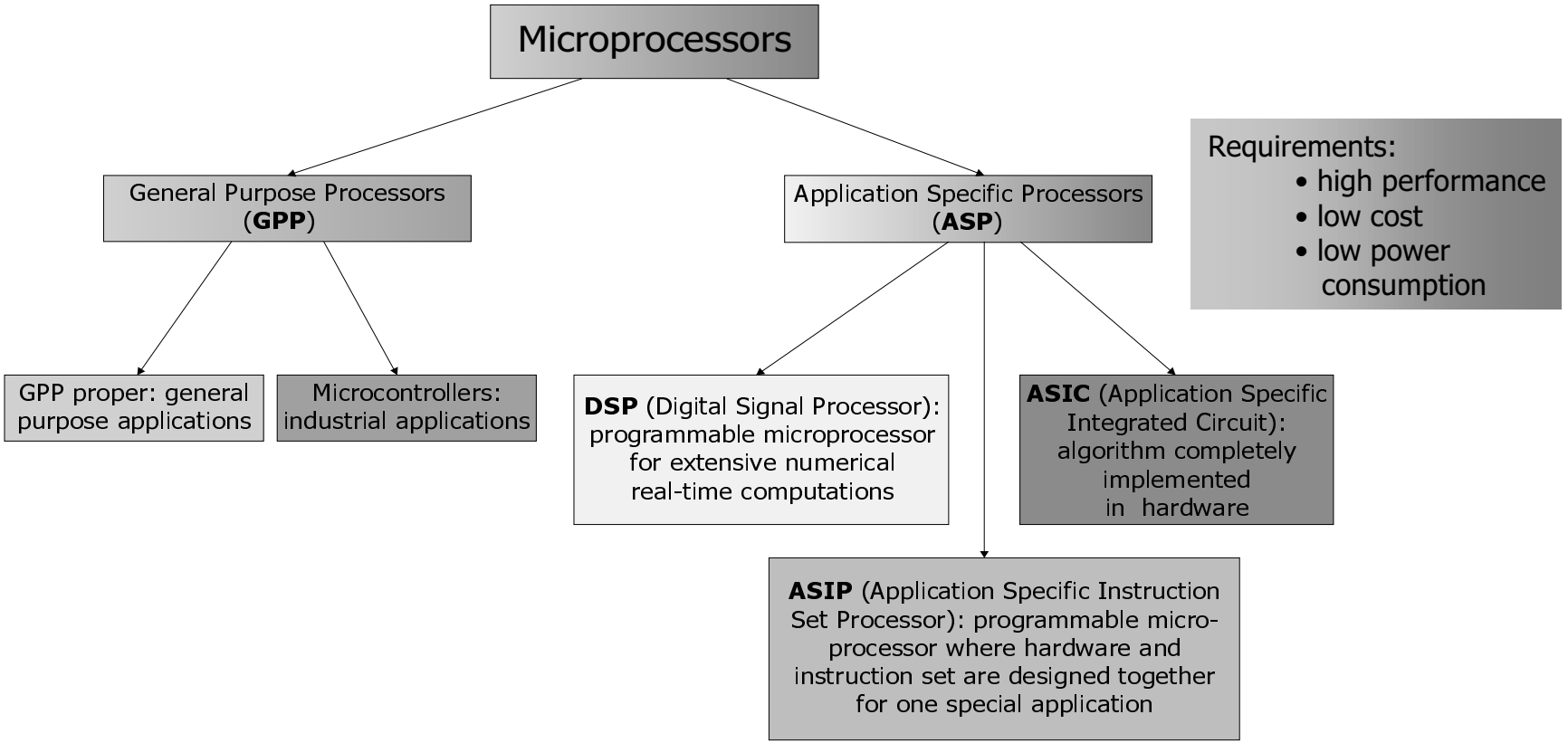
- Diagnostic imaging (Computed Tomography, Magnetic Resonance Imaging, ultrasound, etc)
- Electrocardiagram analysis
- Medical image storage/retrieval



Today

- ✓ What is an embedded system?
- ✓ What do embedded applications look like?
- What makes embedded processors special?
- Why is the code generation problem important?
- Why is it a problem?

Classification of Microprocessors



Architectural Valuation

- [Campbell, Northrop Grumman Corporation, 1998]
- More efficient architectures will use less energy to complete the same task on the same generation CMOS solid state technology
- Power consumption $P = CV^2 f\Delta N$
  - C : Capacitance
  - V : CPU Core Voltage
  - f : CPU clock frequency
  - ΔN : Number of gates changing state

Architectural Valuation

- Observations:
  - Higher performance by increasing the clock frequency does not change the performance per power ratio

$$\frac{Performance}{Power} = \frac{\frac{O}{Cycle} * \frac{Cycle}{Time}}{CV^2 f \Delta N} = \frac{Of}{CV^2 f \Delta N} = \frac{O}{CV^2 \Delta N}$$

  - A voltage decrease improves performance per power non-linearly

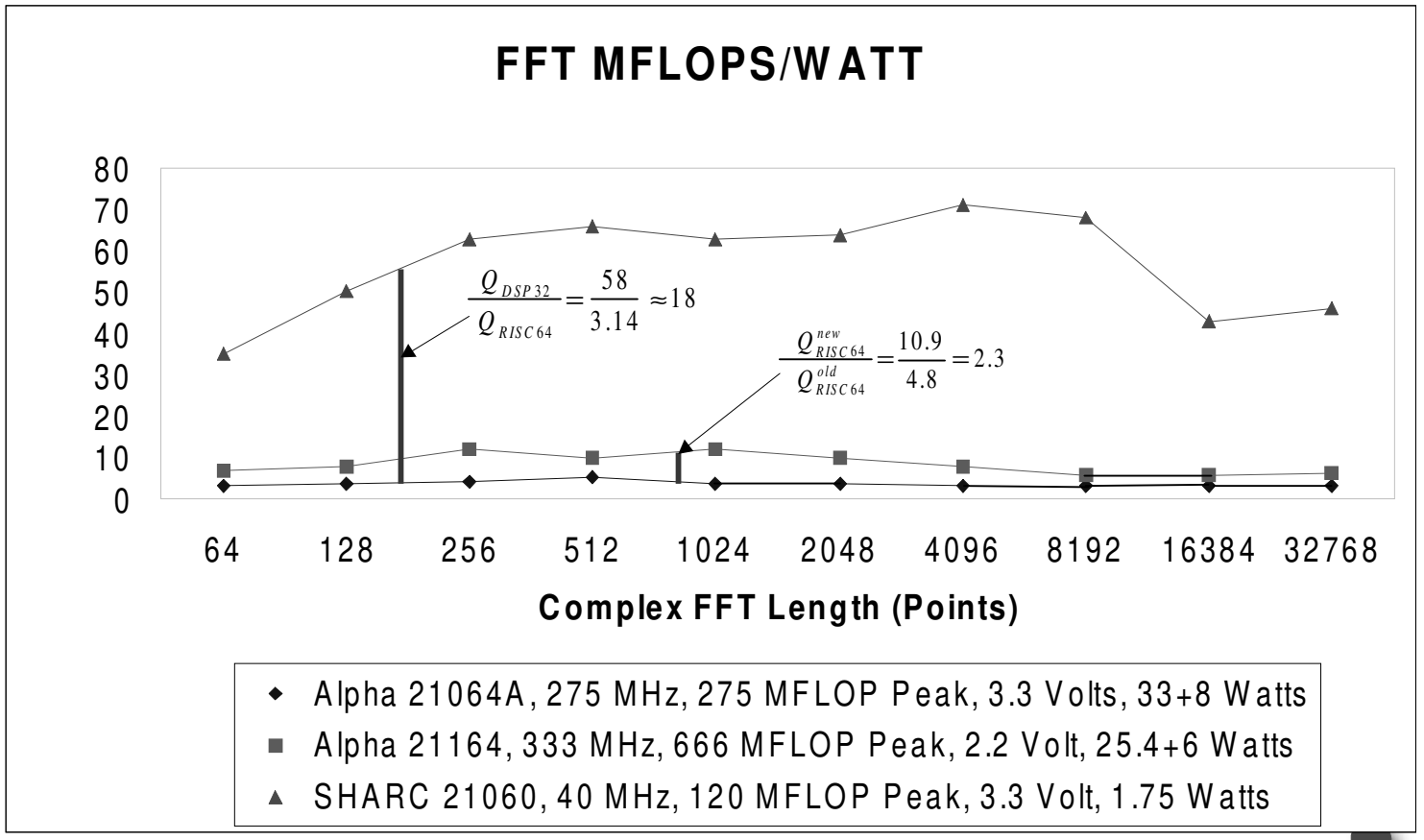
$$\frac{Performance}{Power} = \frac{O}{CV^2 f \Delta N}$$

Architectural Valuation

- Architectural specialization as a measure for how well the architecture fits a given target application.

- Estimation of architectural specialization:
  Performance per power.

Comparion of Performace Per Power Ratios



DSP vs. GPP: Some Figures

Die Size	
DSP	3,9mm^2 – 60mm^2
GPP	≈100mm^2 – 345mm^2 (HP PA-RISC 8000)

Prices (1997)		
Fixed-Point DSPs	Motorola DSP56812	7$
	TI TMS320C54x	25$ - 40$
Floating-Point DSPs	TI TMSC44	130$
	ADSP 2106x SHARC	64$ - 358$
General Purpose Processors	Pentium 200 MHz	509$
	PowerPC 604e 225 MHz	620$

Focus: Digital Signal Processors

- Programmable microprocessors specialized for applications of digital signal processing.
- Characteristics:
  - Multiply-accumulate units
  - Multiple access memory architecture
  - Specialized addressing modes: auto-modify addressing, circular addressing, bit-reverse addressing
  - Residual control / predicated execution
  - Hardware loops / zero-overhead loops
  - Restricted interconnectivity between registers and functional units
  - Encoding restrictions

Today

- ✓ What is an embedded system?
- ✓ What do embedded applications look like?
- ✓ What makes embedded processors special?
- Why is the code generation problem important?
- Why is it a problem?

Classical DSPs: Analog Devices ADSP-2106x SHARC

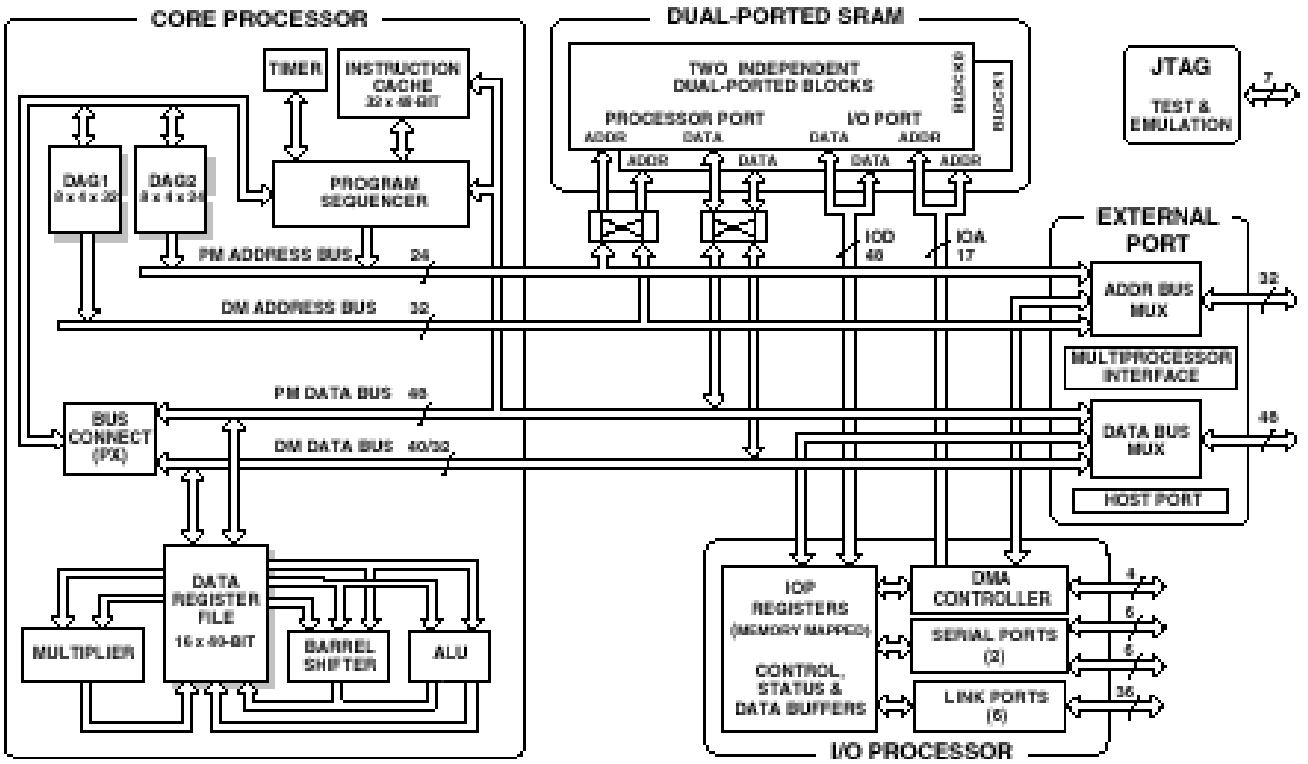
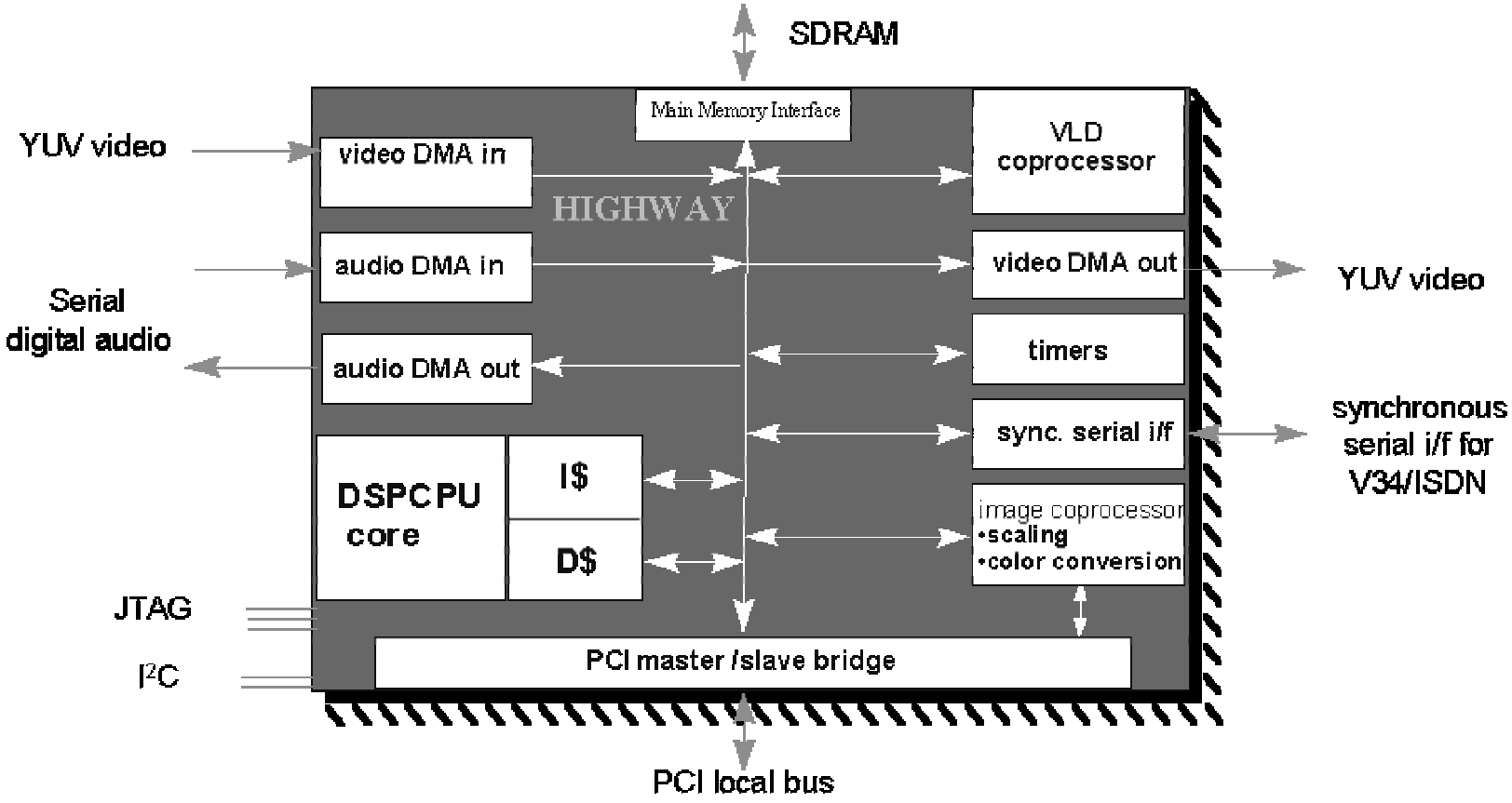
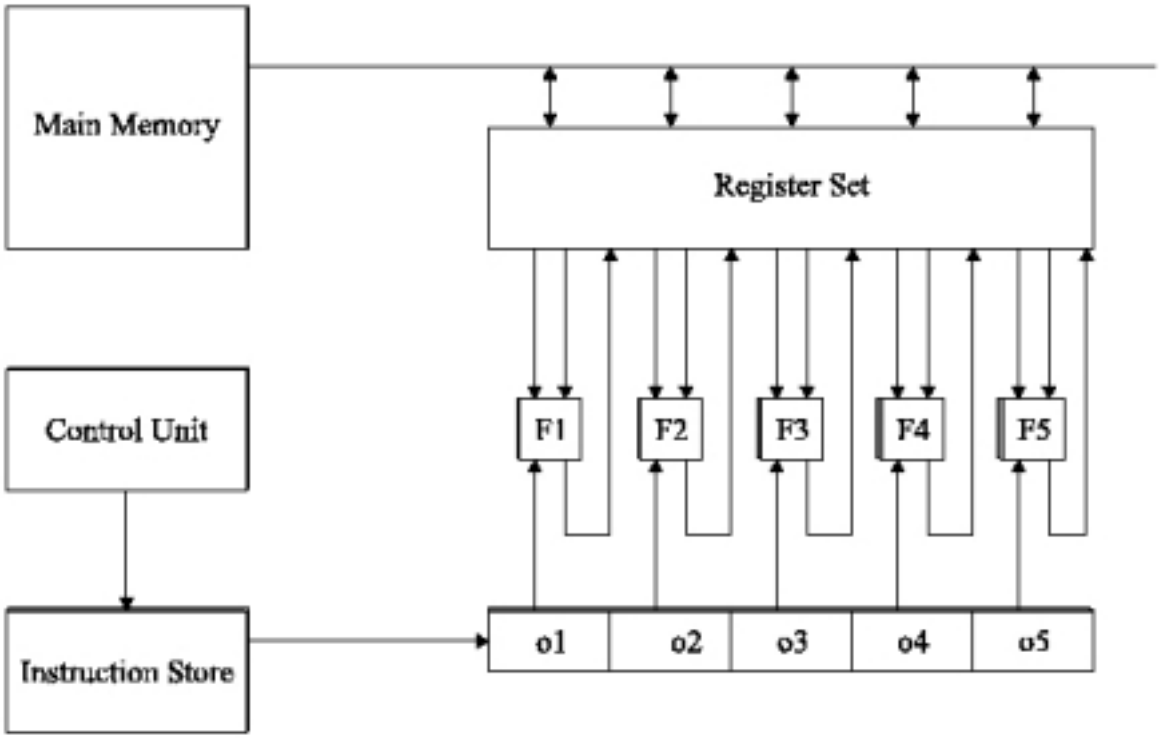


Figure 1. ADSP-21062/ADSP-21062L Block Diagram

Multimedia DSPs: TriMedia TM1000 (1)

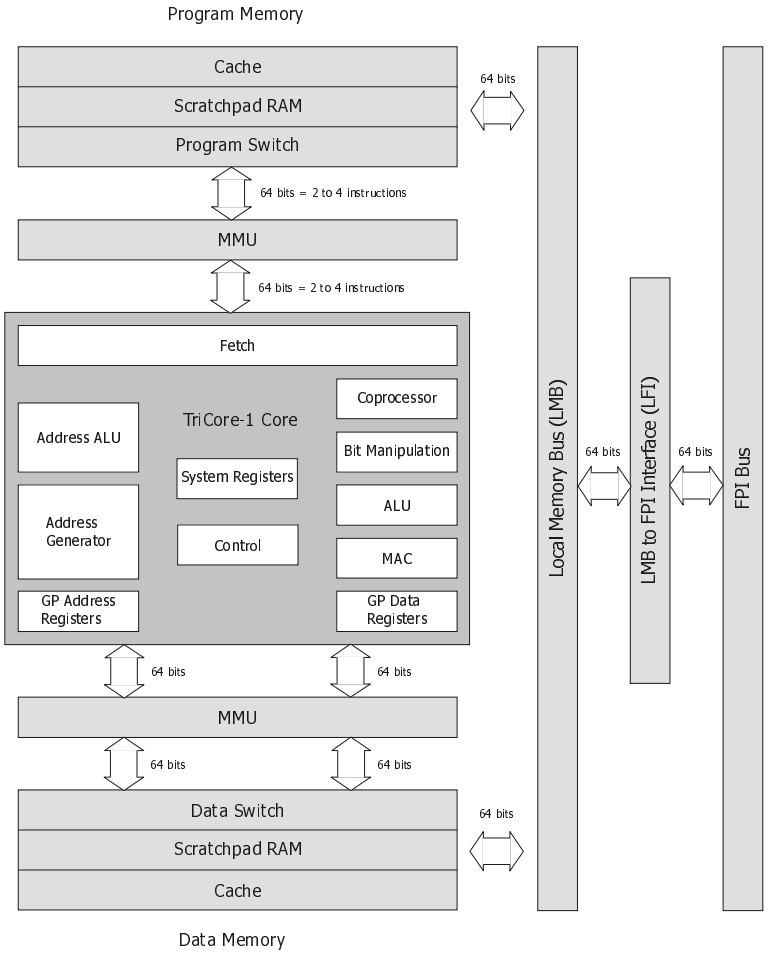


Multimedia DSPs: TriMedia TM1000 (2)

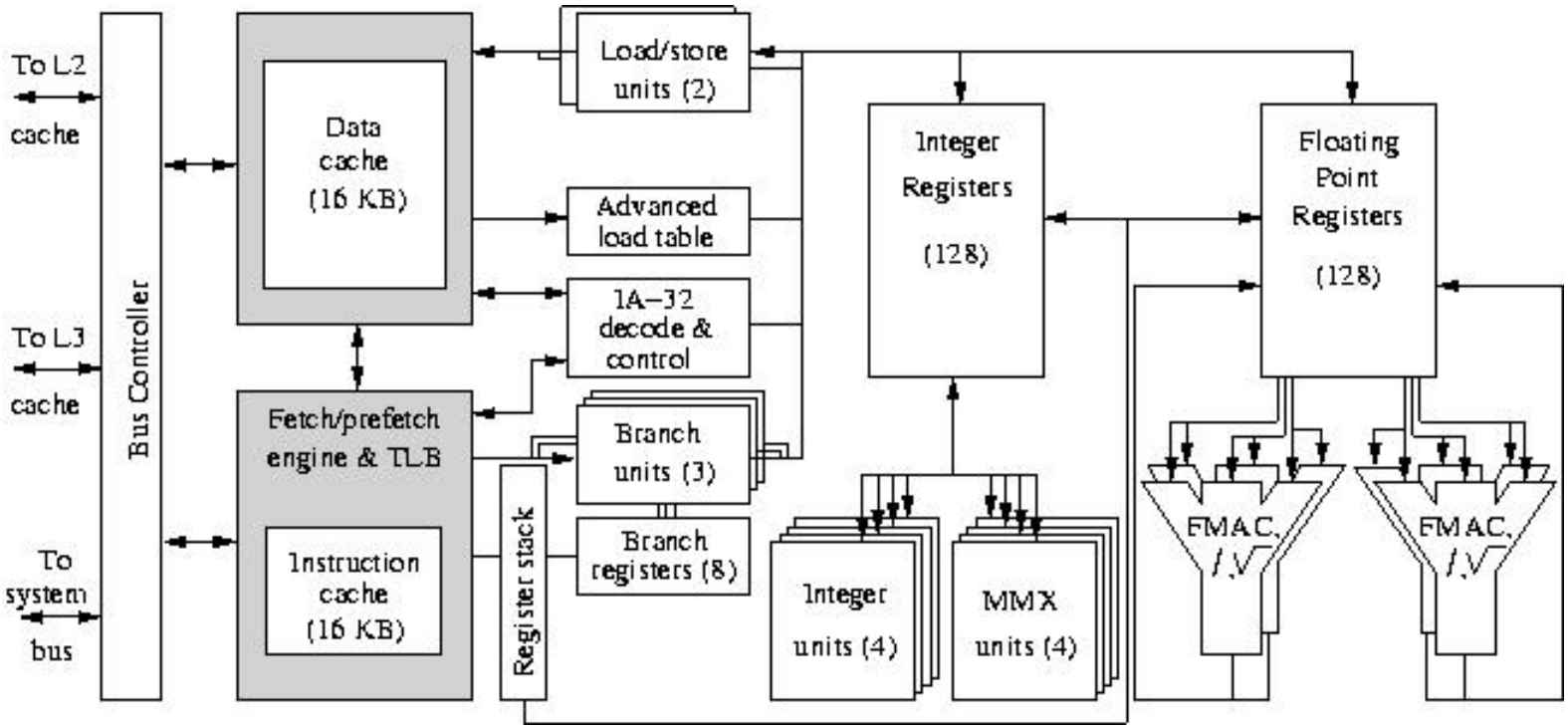


Generic VLIW Architecture

DSP & Microcontroller: Infineon TriCore



Intel IA64

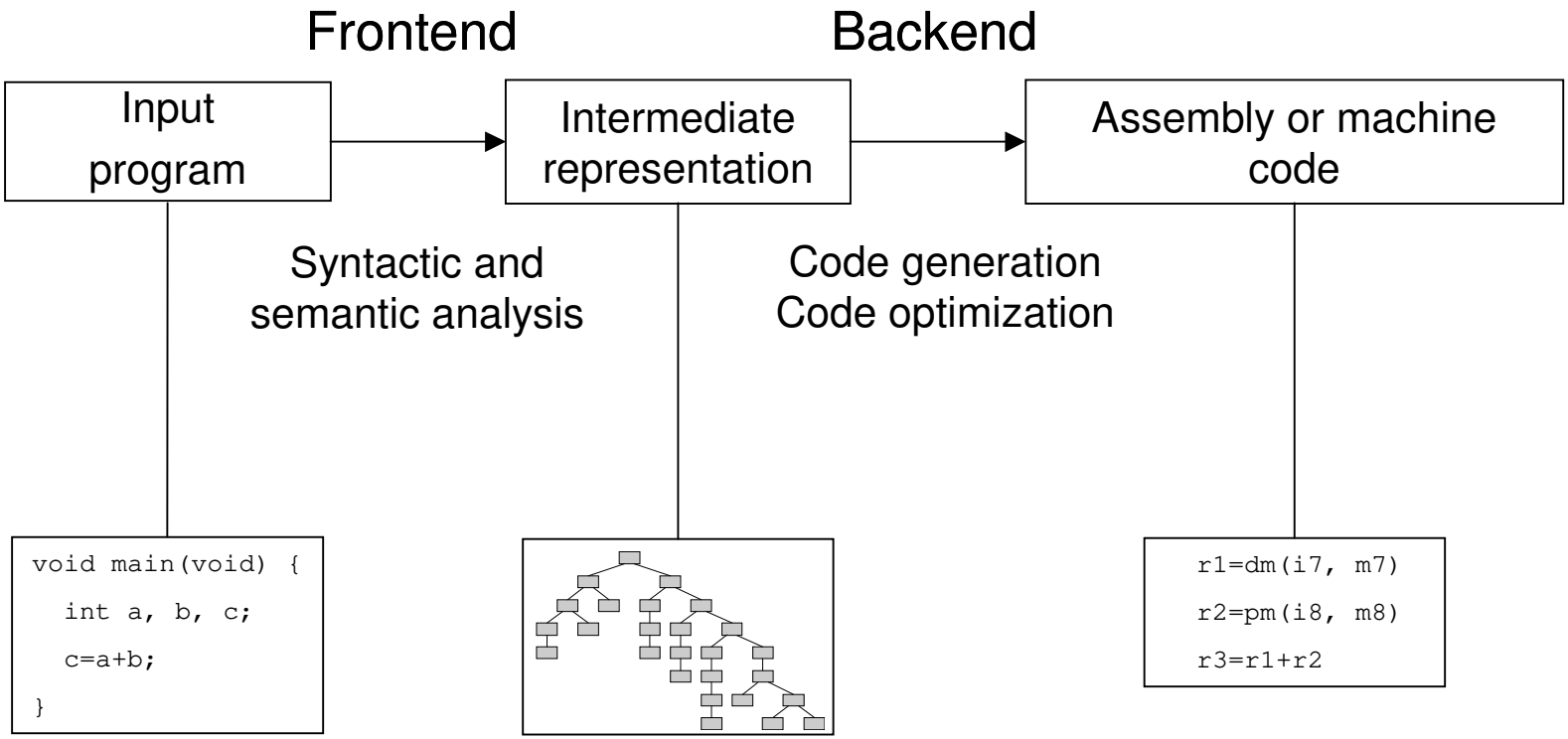


Block diagram of Intel Intanium.

Today

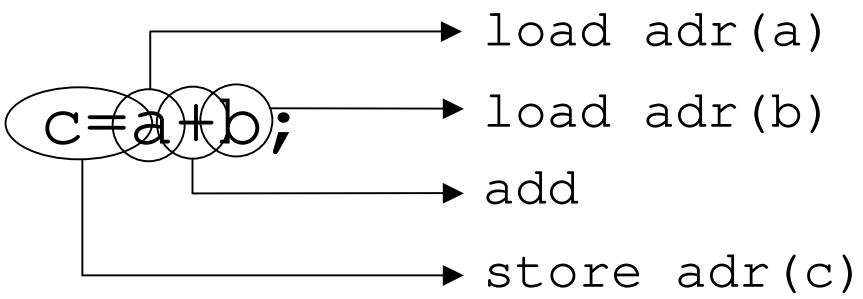
- ✓ What is an embedded system?
- ✓ What do embedded applications look like?
- ✓ What makes embedded processors special?
- ✓ Why is the code generation problem important?
- Why is it a problem?

Compiler Structure

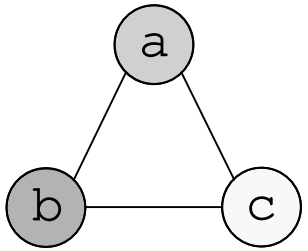


Code Selection, Register Allocation, and Instruction Scheduling

Code selection



Register allocation



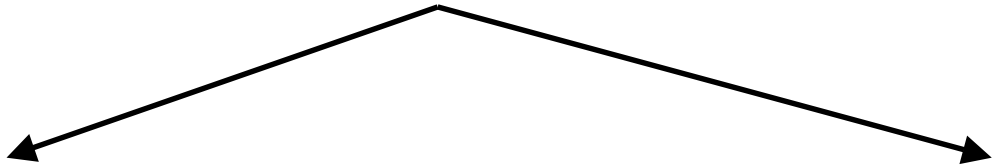
$a \mapsto r1$
$b \mapsto r2$
$c \mapsto r3$

Instruction scheduling

```
r1=load adr(a) || r2=load adr(b)
r3=add r1, r2
store adr(c), r3
```

Phase Coupling Problem

```
d1=s1*s1;
d2=s1+s2;
```



```
r3=r1*r1
store r3
r3=r1+r2
store r3
```

```
r3=r1*r1, r4=r1+r2
store r3
store r4
```

Optimize number of used registers

Optimize code speed and size

The Code Generation Problem

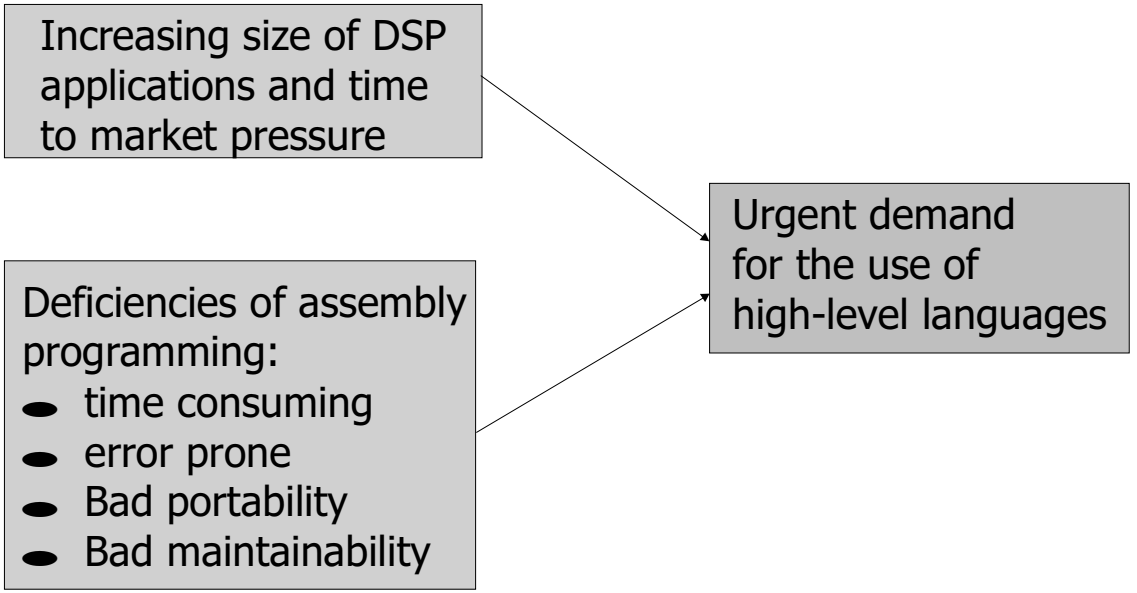
- Instruction scheduling, register allocation and code selection are NP complete problems.
- In classical approaches they are addressed by heuristic methods in separate phases.
- Unfortunately, all the code generation phases are interdependent, i.e. decisions made in one phase may impose restrictions to the other phases.
- Thus: often suboptimal combination of suboptimal partial results.
- Moreover: specific/irregular hardware features not well covered by standard code generation methods.

The DSPStone Study

- Evaluation of the performance of DSP compilers and joint compiler/processor systems.
- Evaluated: Analog Devices ADSP2101, AT&T DSP1610, Motorola DSP56001, NEC mPD77016, TI TMS320C51.
- Hand-crafted assembly code is compared to the compiler-generated code.
- Result: overhead between 100% and 1000% of compiler-generated code is typical !

Compiling for DSPs

- Code quality of traditional high-level language compilers is not satisfactory.
- Thus: Assembly programming.



Case Study: Infineon TriCore

C code (FIR Filter):

```
int i,j,sum;
for (i=0;i<N-M;i++) {
  sum=0;
  for (j=0;j<M;j++) {
    sum+=array1[i+j]*coeff[j];
  }
  output[i]=sum>>15;
}
```

Compiler-generated code (gcc):

```
.L21: add %d15,%d3,%d1
      addsc.a %a15,%a4,%d15,1
      addsc.a %a2,%a5,%d1,1
      mov %d4,49
      ld.h %d0,[%a15]0
      ld.h %d15,[%a2]0
      madd %d2,%d2,%d0,%d15
      add %d1,%d1,1
      jge %d4,%d1,.L21
```

Hand-written code (inner loop):

```
_8: ld16.w d5, [a2+]
    ld16.w d4, [a3+]
    madd.h e8,e8,d5,d4ul,#0
    loop a7,_8
```

- ✓ Zero-Overhead Loops
- ✓ SIMD Instructions
- ✓ Auto-modify addressing
- → **6 times faster!**
  (execution in SRAM)

Overview of Lecture

- Embedded systems and digital signal processing.
- Hardware characteristics of embedded processors.
- Code generation methods for specific hardware features.
- Code generation methods for the phase coupling problem.