

# ***MuViStar***

**SCIWORX**

---

Ein programmierbarer Videosignalprozessor  
für mobile Applikationen

Dr. Klaus Herrmann, sci-worx GmbH

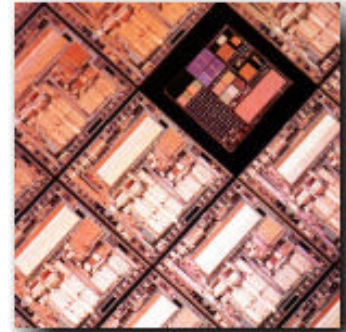
# Outline

- Introduction
- Processor Concept for Mobile Video Coding Applications
  - Instruction Set Architecture
  - Intelligent Direct Memory Access Controller
- MuViStar Processor Architecture
- Performance and Gatecount
- Summary

# Introduction

- Emerging Video Coding Applications on Mobile Devices
  - Digital TV for Mobile Phones (DVB-H, DMB)
  - Portable Media Players (PMP)
  - Multi Media Messaging (MMS)
  
- Required Video Coding Standards
  - DVB-H, DMB: ITU-T H.264, up to 352x288 pixel (CIF), 30 fps
  - PMP: MPEG-4, MPEG-2, DIVX, H.264, VC-1  
up to 720x576 pixel (D1), 25 fps
  - MMS: H.263, MPEG-4, up to 176x144 pixel (QCIF), 30fps
  
- Design and Implementation of a programmable Videosignal Processor  
-> **MuViStar (Multi Video Standard architecture)**

# Custom Design Targets



- Design Targets:
  - Multi standard encoding and decoding up to D1 @25fps on a single 200MHz processor core
  - Flexibility for all known standards
  - C-code programming, no assembler usage wanted
  - Scalable to higher resolutions with multiple cores

# Architecture Approach

- MuViStar is based on the Tensilica Xtensa LX core and the Xtensa core adaptation methodology

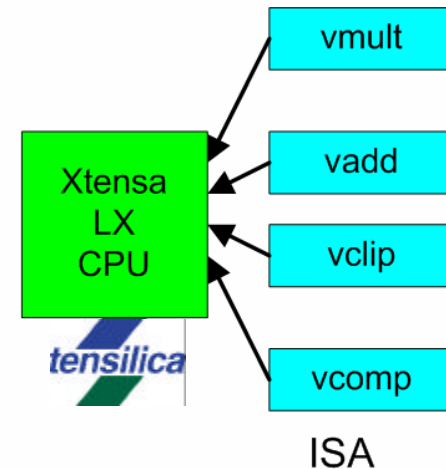
## (1) Exploration of parallelism in the desired classes of algorithms

- Data level, instruction level, and task level

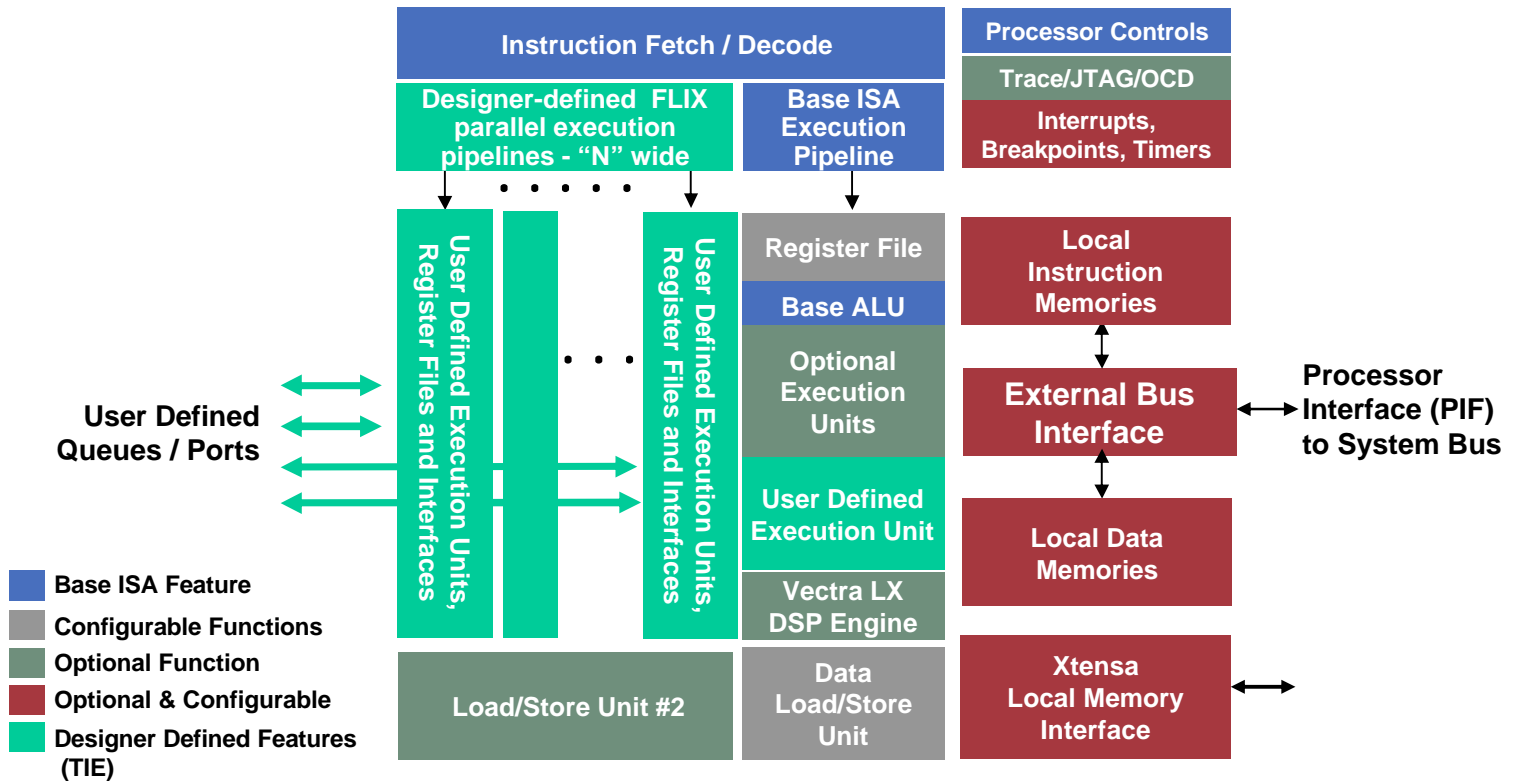
## (2) Mappings of this parallelism to custom extensions

- Vector instructions (SIMD)
- Complex scalar multi-operand instructions
- Slots (VLIW)
- Multiple cores, DMA

## (3) Selecting the optimum by efficiency consideration (speed delta versus area delta, retaining flexibility)

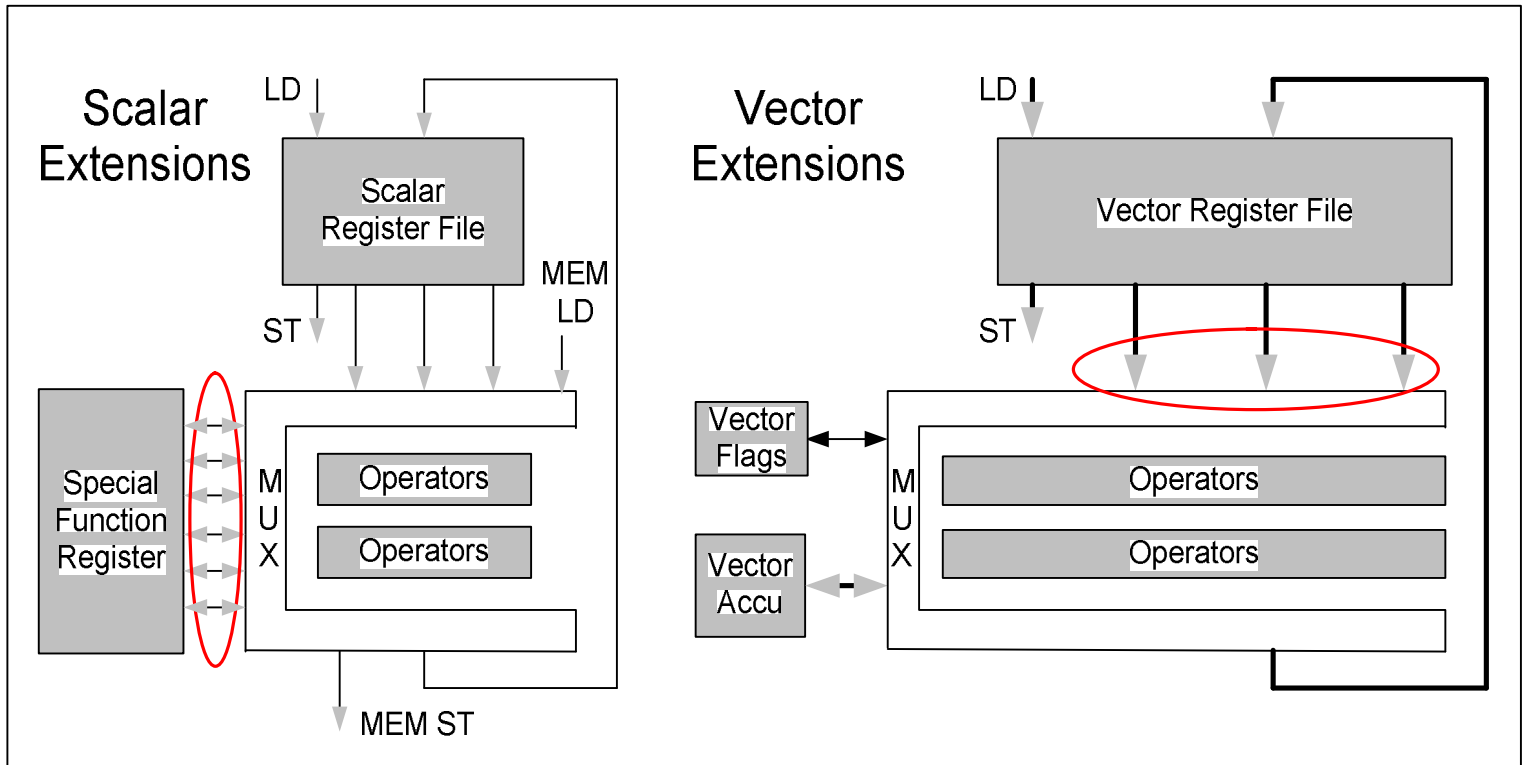


# Tensilica Xtensa LX



# Video Extensions Overview

- Video Extensions = SIMD Vector Extension + Scalar Extensions  
Basic structure, operand sources, and destinations:



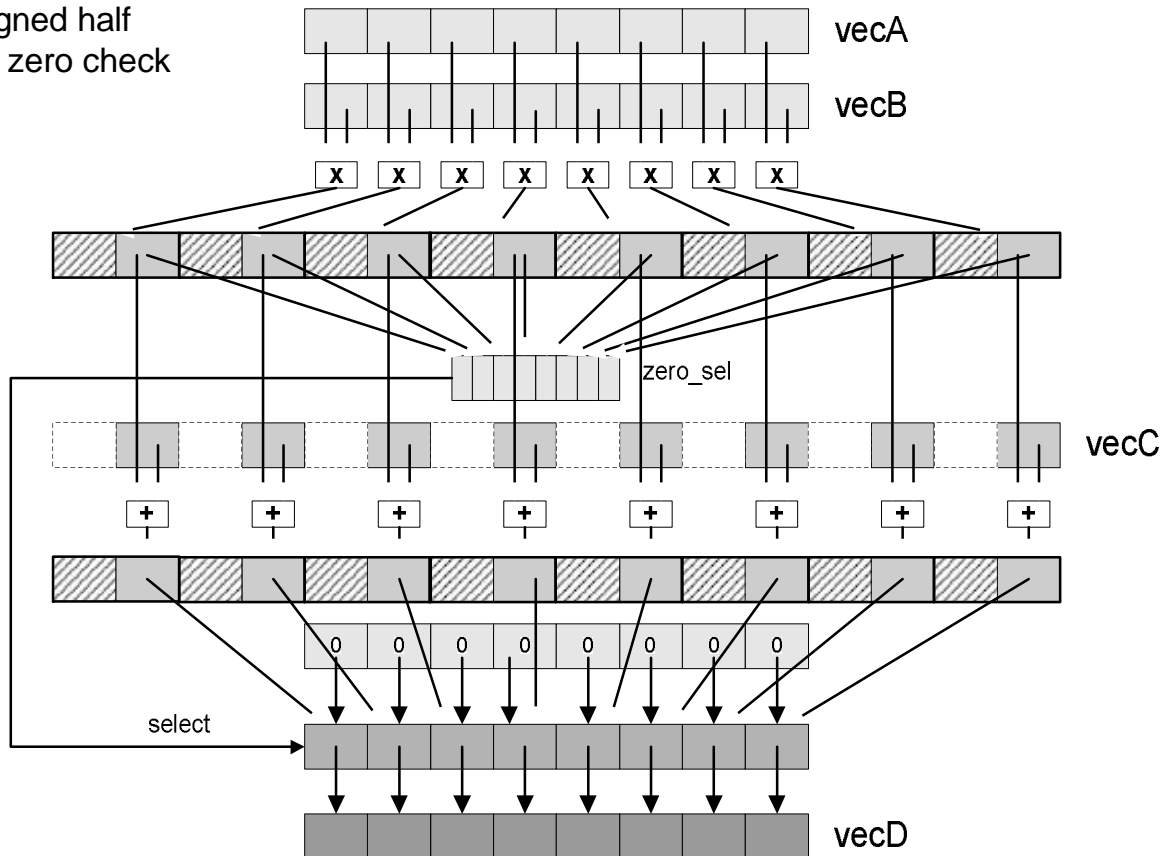
# Vector Extension Example

- SCImaddl\_sh\_mzd

Multiply low and add signed half word modulo with mult. zero check  
(multiplication with correction)

- Single cycle

- Usage: iQuant



# Scalar Instructions

- (1) Scalar instructions common for all standards (default)
  - Bit-stream handling
  - VLD: parsing @ 5–8 cycles per symbol lookup
    - ☞ **Code tables in memory**
  - VLC: code construction support
  - ~25k gates
  
- (2) Scalar instructions video-standard specific (optional)
  - VLD: parsing @ 1–2 cycles per symbol lookup
    - ☞ **Code tables hard-wired**
  - CAVLC support
  - ~5–15k gates per codec standard

VLD: Variable Length Decoding

VLC: Variable Length Coding

CAVLC: Context-adaptive variable-length (de)coding used in H.264

# Scalar Instruction Example

- MPEG4 8x8 block texture parsing (code tables in memory)
- 90% of an MPEG4 stream (runs & levels) are decoded with this C-code:

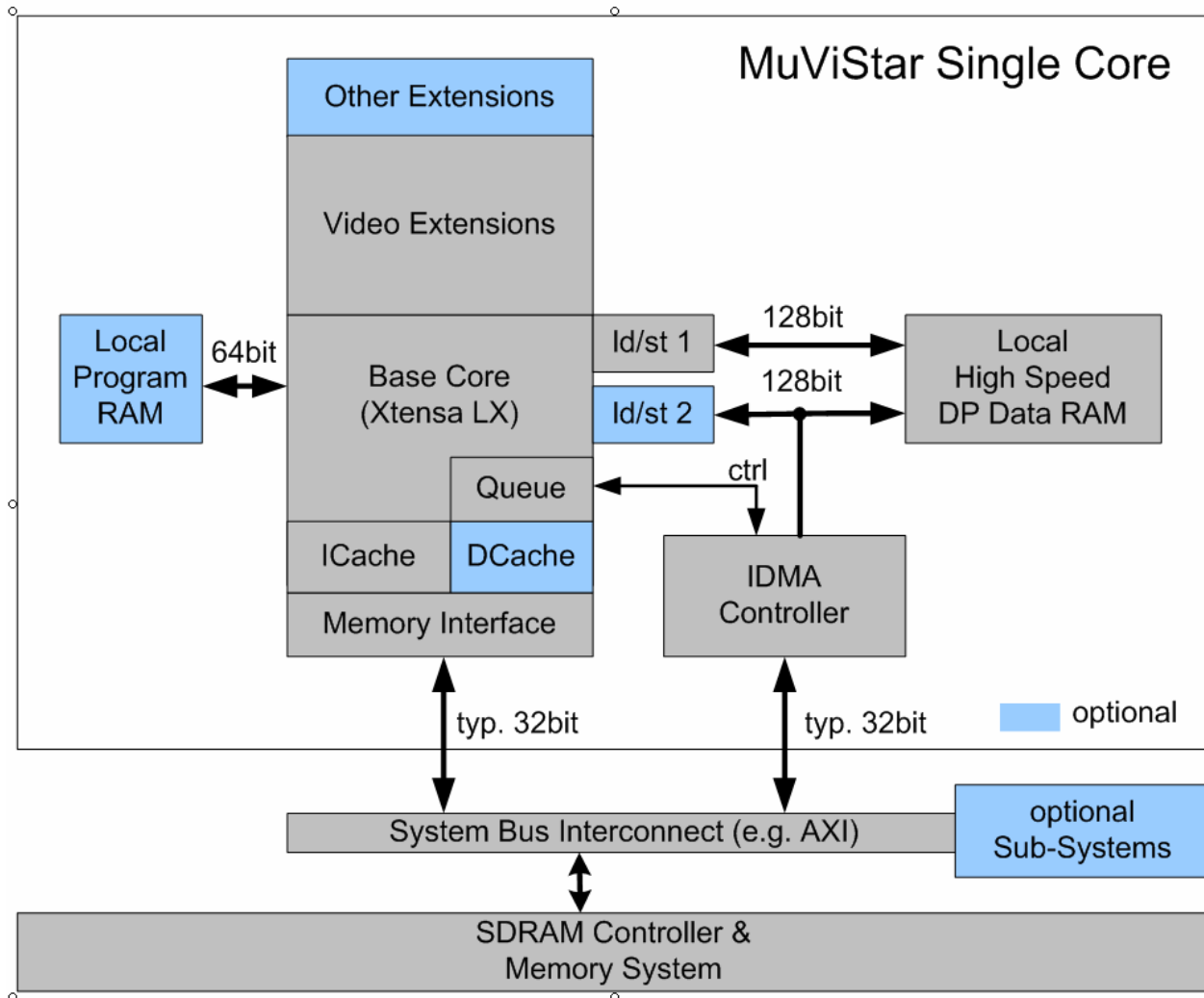
```
while (flag == 0) {  
    // Fast generic VLD  
    SCIsp_decvldi(tmp, vld_table); // VLD bits 0..3  
    SCIsp_decvld (tmp, vld_table); // VLD bits 3..7  
    SCIsp_decvld (tmp, vld_table); // VLD bits 8..11  
    SCIsp_decvld (tmp, vld_table); // VLD bits 12..16  
    run_level[i] = tmp; // store decoded symbol  
    SCIsp_flush_bits(); // flush used bits  
    flag = SCIsp_getfvld_ctrl(mode); // get loop control flag  
    i++; // increment store ptr  
}
```

- Loop performance improvement is more than 10x

# Intelligent DMA

- Factors that will easily degrade SIMD performance:
  - Vector misalignments, vector reordering, vector incompleteness
  
- Use IDMA for reordering and realignment “on the fly”:
  - Data remapping for single-cycle vector accesses
    - ◆ Several reorder formats (linear, vertical, Cb/Cr separated, 4x4, ...)
    - ◆ Several address modes (linear, x/y addressing, absolute, ...)
    - ◆ Out-of-picture padding support
  
  - Single-cycle IDMA start by queuing a pointer on a job descriptor
  - Single-cycle IDMA queue ready check
  
  - IDMA queue with 64 entries, job descriptors located in local memory

# Processor Architecture Overview (Single Core)



# Benchmarks (Single Core)

Standard	Direction	Resolution	Frame Rate	MHz
MPEG4	Decode	D1	30	<100
MPEG4	Encode	D1	25	<200
H263	Decode	D1	30	<100
H263	Encode	D1	25	<200
H264	Decode	D1	30	<200
H264	Encode	CIF	25	<150

- Gate counts

- Core 100k gates
- Custom extensions 195k gates
- IDMA 45k gates
- ◆ Total **340k gates**

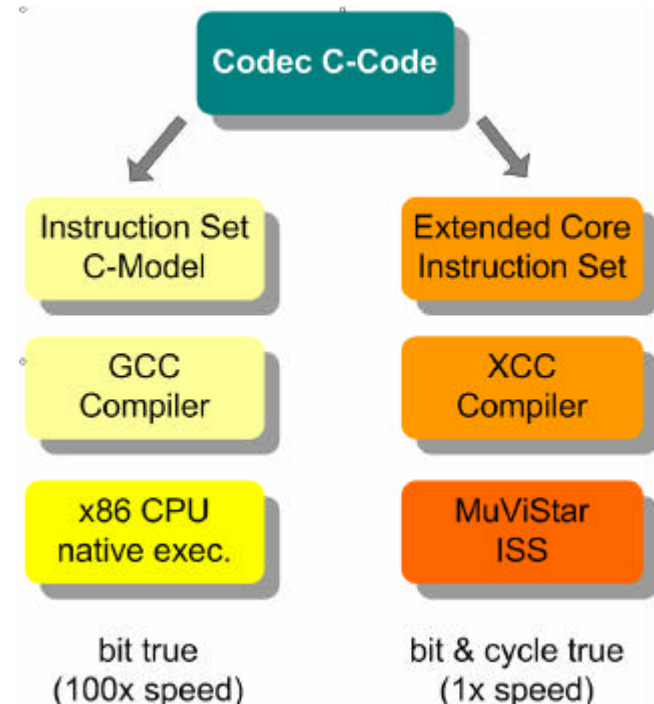
- SRAM size: 68Kbytes

- Estimated silicon area 6 mm<sup>2</sup> (TSMC 0.13G)

- Estimated power consumption 0.4 mW per MHz (typ. cond. 1.2V)

# Programming

- Video codecs written entirely in C
  - Use intrinsics but no inline assembler
  - Leave register management and slot assignment to the compiler
- Single code base compiles with
  - XCC (Xtensa C-Compiler) for MuViStar
  - GCC (GNU C-Compiler) for native x86 CPU
- All custom instructions are prototyped in C and immediately verified
  - Software guys write their own “instructions”
  - Hardware guys get “executable specifications”
- Software test and instruction verification at native speed on x86 CPU is ~100 times faster than using the instruction-set simulator (ISS)



# Summary

- MHz efficiency drove development
- More than 10x improvement in processing power with custom extensions
  
- New instructions are not codec specific
  - H.264, MPEG4, H.263, MPEG2, VC1, Real, XVID, DIVX, AVS, DIRAC, ...
- Video codecs written entirely in C
  
- Ultra-fast bit-true verification
- High scalability
  
- H.264 D1 decoder single core: 6 mm<sup>2</sup>, 60mW (TSMC 0.13G)

