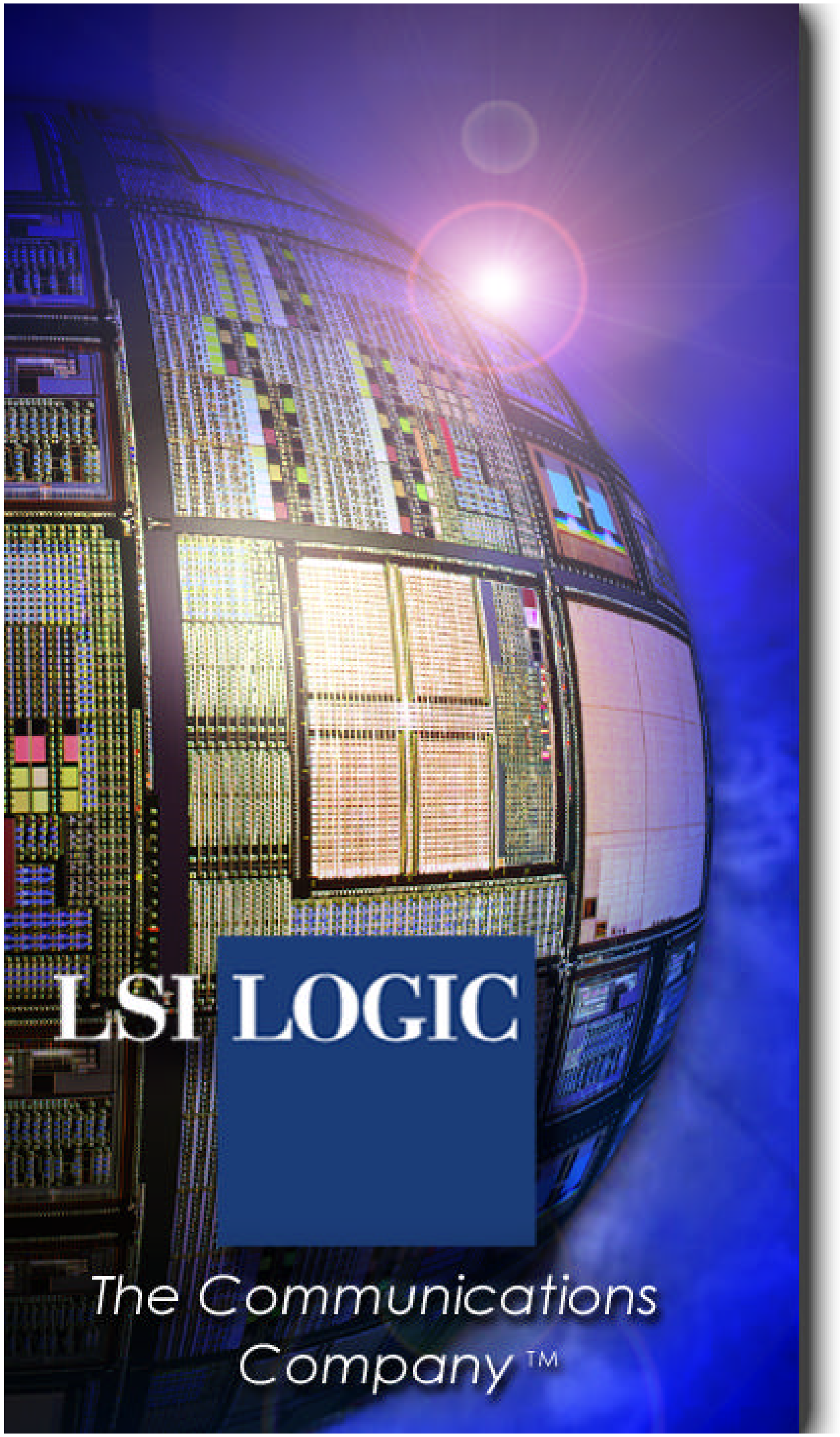


ZSP™ G2

DSP Freedom

The Second Generation

Presented by

Vijay Angarai

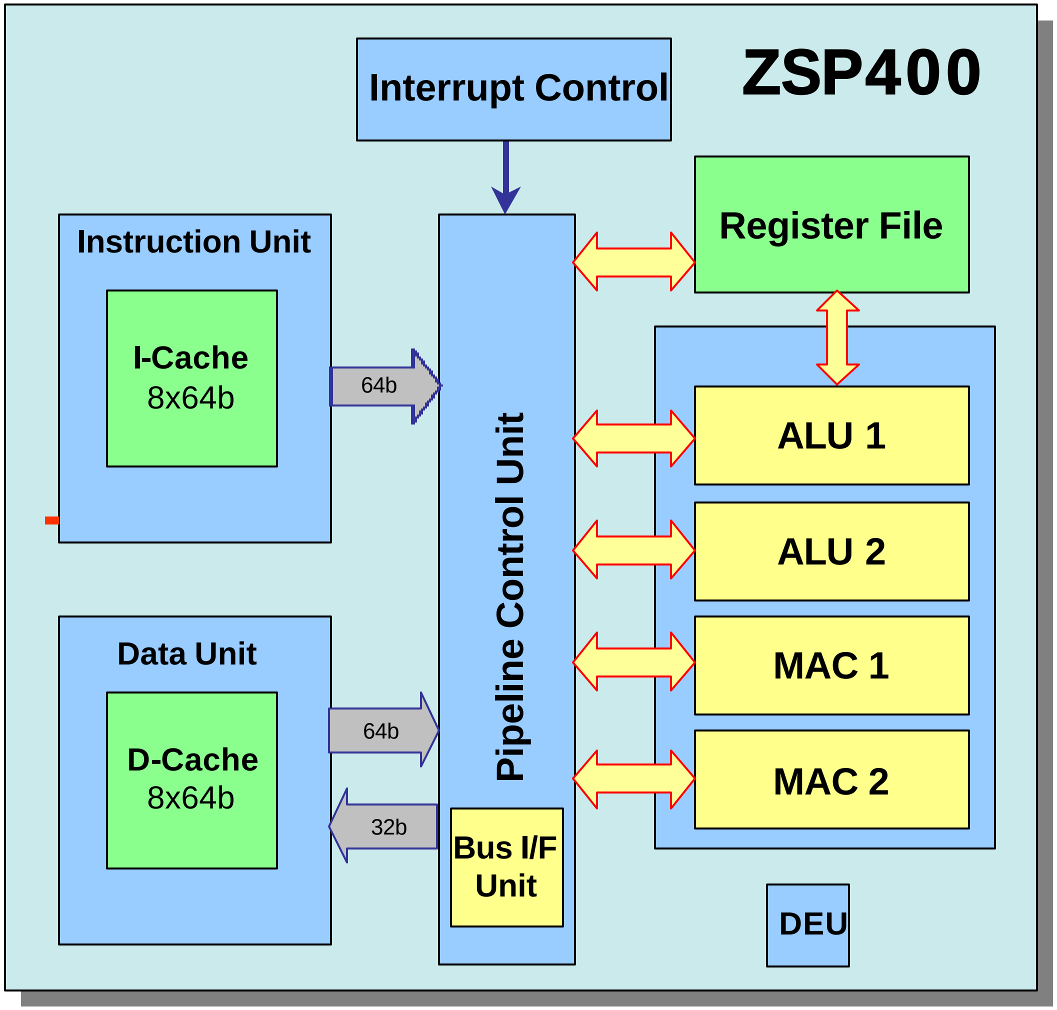
For questions, contact:
dsp-support@lsil.com

Introducing “G2”

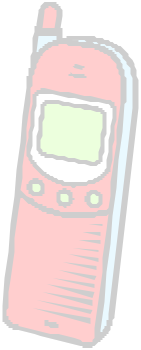
- Extending the performance of the ZSP400
- Sixteen bit, superscalar RISC architecture
- Agenda
  - G1/ZSP400 Overview
  - Design goals and focus
  - Design process
  - Resource, pipeline, and ISA enhancements
  - G2 implementations : ZSP500/ZSP600
  - Summary

G1/ZSP400 Features

- 5-stage hardware controlled pipeline
- High Code Density
- Compiler friendly ISA
- 16 General Purpose Registers
  - Can be combined into 32-bit registers
  - 40-bit multiply/accumulate results available from MAC
- Address generation by ALU
- 16-bit Address Space
- Bus Interface to External Peripherals
- Memory Interface



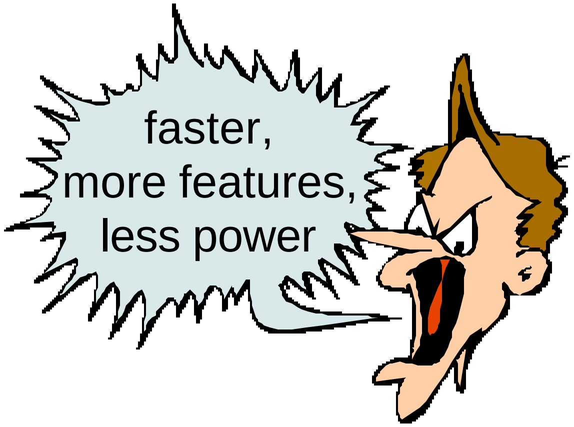
Design Goals

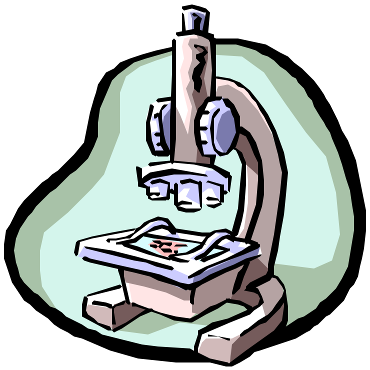


- Extend the ZSP roadmap
- Maintain instruction set compatibility
- Broaden the application space
- Enable scalable implementations
- Support wider address space
- Enhance instruction set architecture
- Simplify system integration

Design Process

- Substantial direction from customers
- Feedback from developers (int. & ext.)
- Detailed analysis of existing architecture
- Architectural simulations
- Iterative feedback process with licensees





G2 : Overview

- Sixteen bit, superscalar RISC architecture

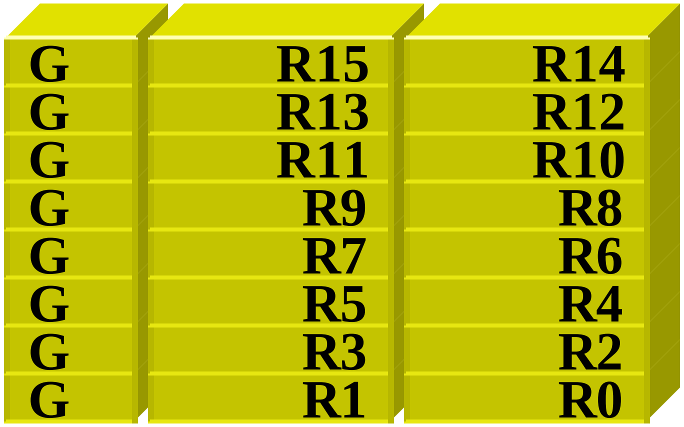
- Scalable in terms of
  - Arithmetic resources
  - Data bandwidth

- Multiple Implementations share same
  - Instruction Set Architecture (ISA)
  - Register Resources

G2 : Register Resources

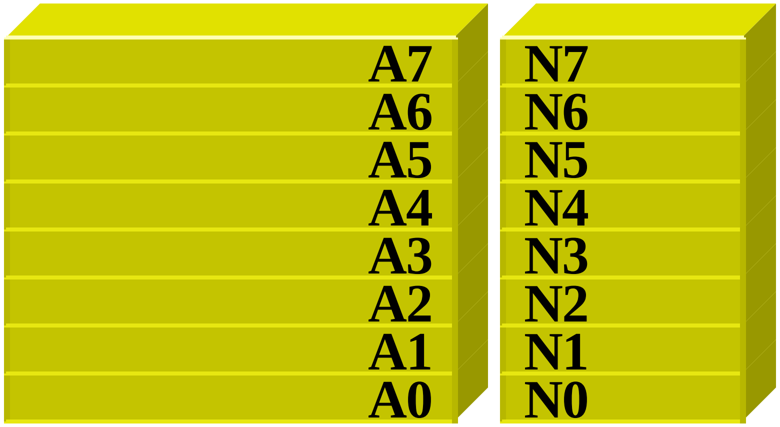
- Sixteen 16b general purpose registers
- **New**: Eight guard bytes
- **New**: Eight 32b address registers
- **New**: Eight 16b index registers

General Purpose RF



G	R15	R14
G	R13	R12
G	R11	R10
G	R9	R8
G	R7	R6
G	R5	R4
G	R3	R2
G	R1	R0

Address Register File



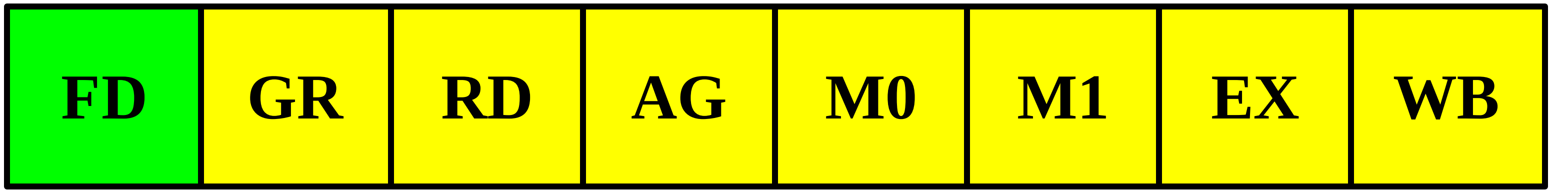
A7	N7
A6	N6
A5	N5
A4	N4
A3	N3
A2	N2
A1	N1
A0	N0

G2 : Arithmetic Resources

- Two Address Generation Units **(AGU)**
  - Up to two 64-bit load/store requests per cycle
  - No alignment restrictions
- Two 16-bit ALUs
  - Combine for 32-bit arithmetic
- 40-bit Multiply/Arithmetic Units **(MAU)**
  - Perform 16-bit, 32-bit, and 40-bit arithmetic
  - Scalable resource → Quad MAC execution
  - Full Viterbi butterfly per cycle

G2 : Pipeline

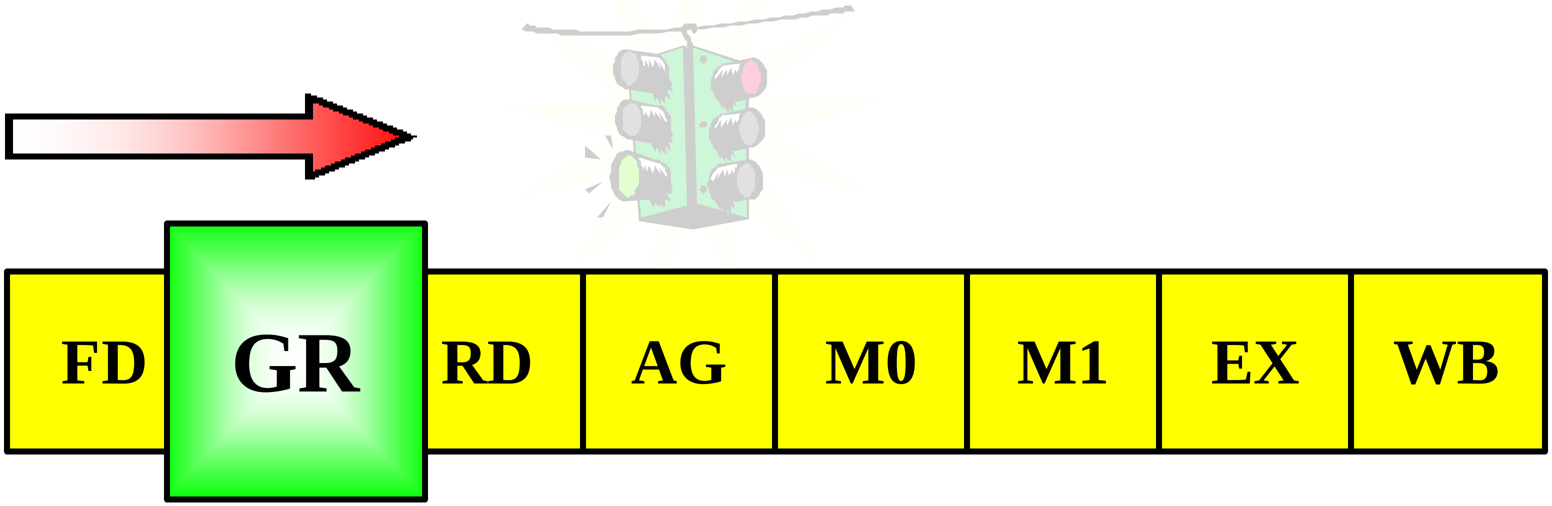
- Eight stage pipeline
- Hardware controlled pipeline protection
- **FD** - Fetch/decode
  - Up to six instructions per cycle



FD	GR	RD	AG	M0	M1	EX	WB

G2 : Pipeline

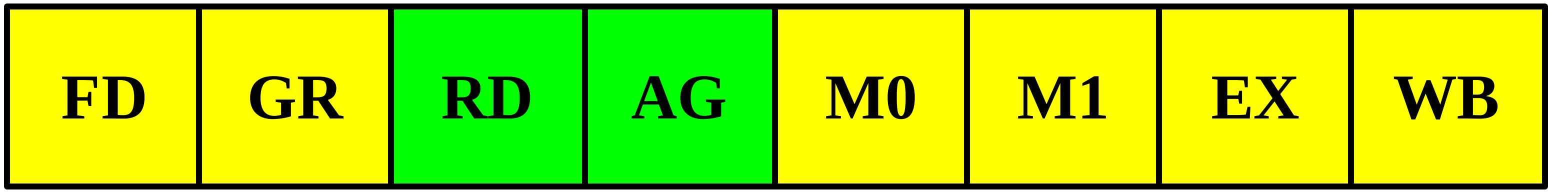
- **GR** – Grouping
  - Gate-keeper of the pipeline
  - Responsible for ALL pipeline protection
  - In-order issue of instruction groups



FD GR RD AG M0 M1 EX WB

G2 : Pipeline

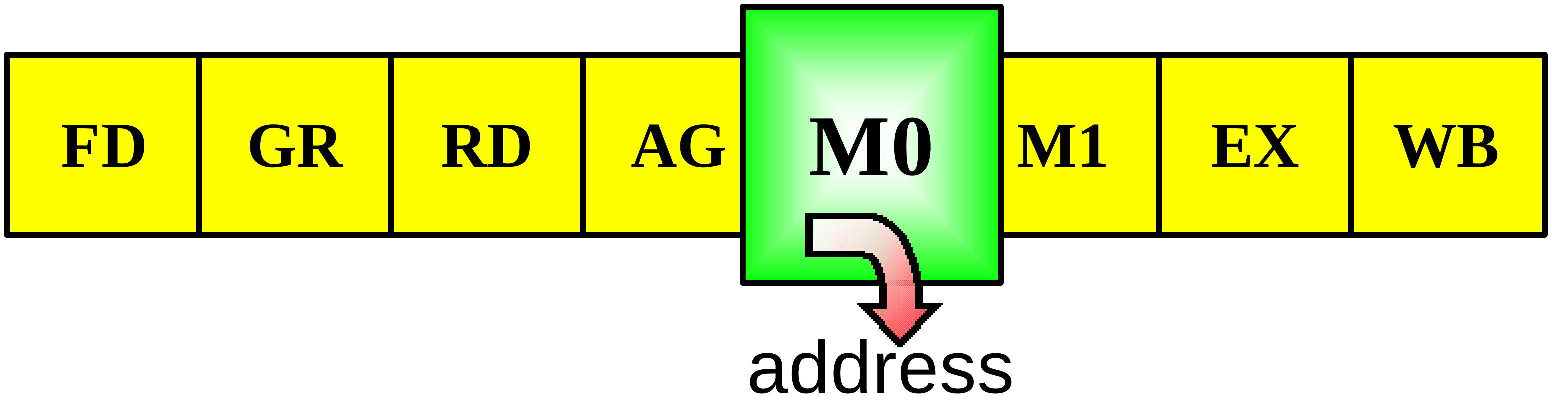
- **RD** – Operand Read
  - Read operand for address generation
- **AG** – Address Generation
  - Perform address arithmetic
  - Generate load/store request(s)



FD	GR	RD	AG	M0	M1	EX	WB

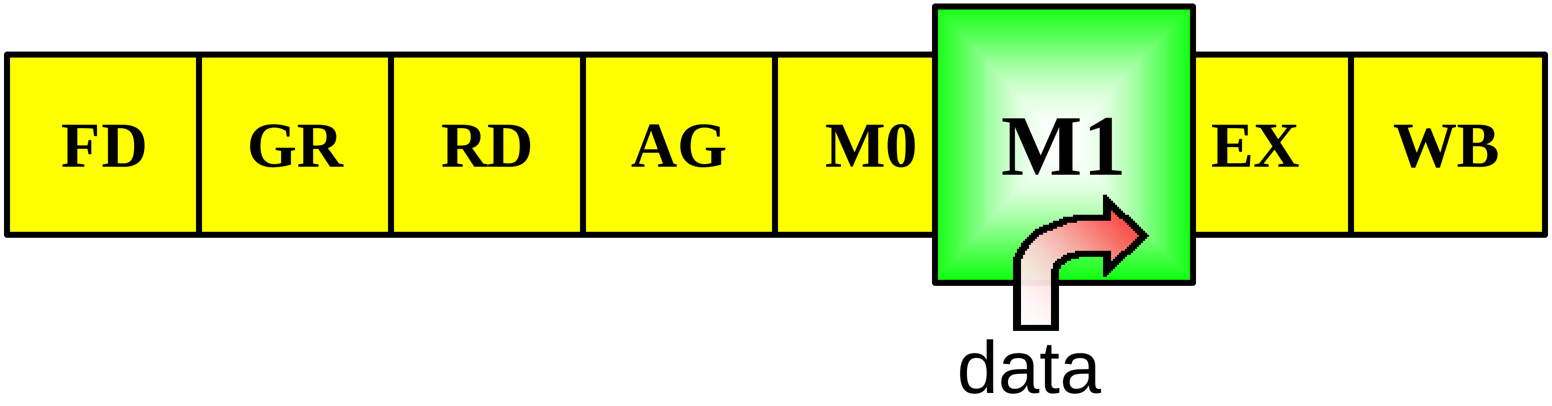
G2 : Pipeline

- **M0** – Memory Access Stage 0
  - First stage allocated for Memory Subsystem
  - Load and store requests issued to MSS
  - MSS performs address decode



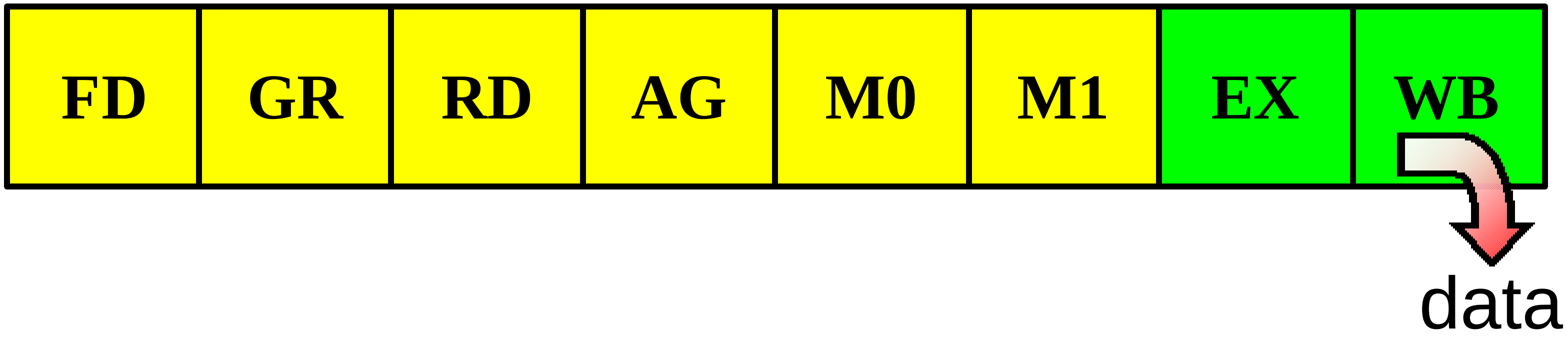
G2 : Pipeline

- **M1** – Memory Access Stage 1
  - MSS performs data access
  - Load data returns to core
  - Operand read stage for ALUs and MAUs



G2 : Pipeline

- **EX** - Execution
  - Result forwarding occurs
  - ALU operations are executed and completed
  - Data written to register files
- **WB** – Write Back
  - Store data sent to MSS



Instruction Set Enhancements

- ISA based on a 16-bit instruction word
  - Facilitates high code density
  - Assembly level compatible with first generation
- Prefix allows paired instruction words
  - Improves support for immediate operands
  - Allows more effective use of the 16-bit opcodes
  - Improves code density
  - Improves compiler efficiency

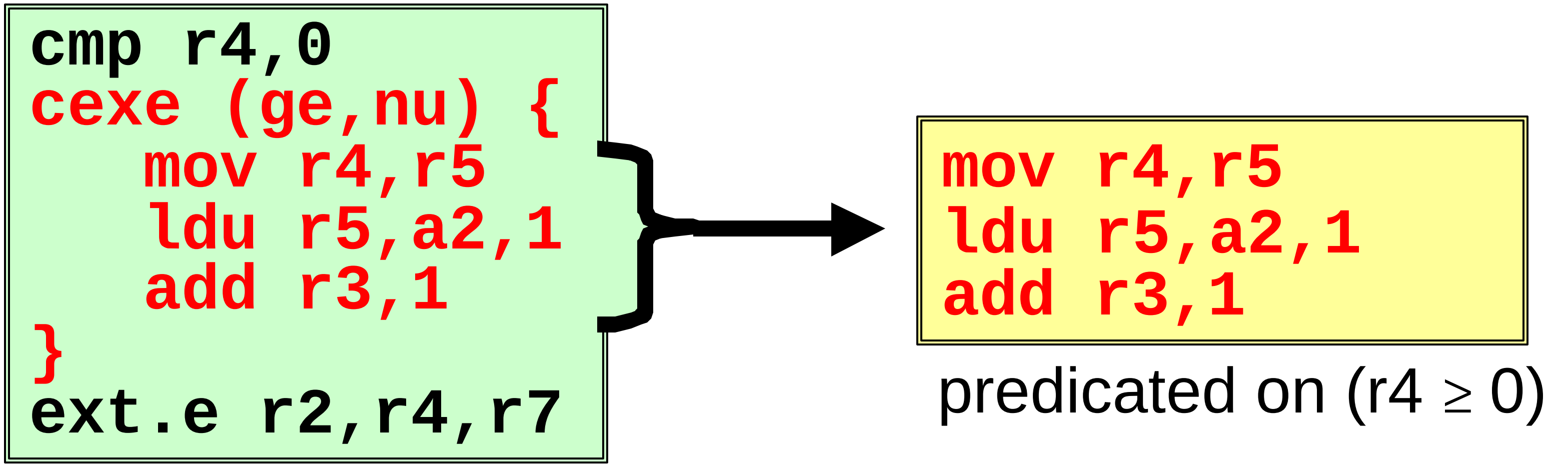
Instruction Set Enhancements

- ANSI C compliant operations
- Arithmetic operations that ignore modes.
- Loads/stores with immediate offsets
- Signed & unsigned MUL/MAC instructions
- SIMD operations
- Bit field operations
- Forty-bit arithmetic
- Division assist

New!
16-bit

Instruction Set Enhancements

- Conditional Execution
  - Implemented as a single instruction: **cexe**
  - Programs define variable width **cexe packets**
  - Entire **cexe packet** is predicated
  - Multiple **cexe** variations supported



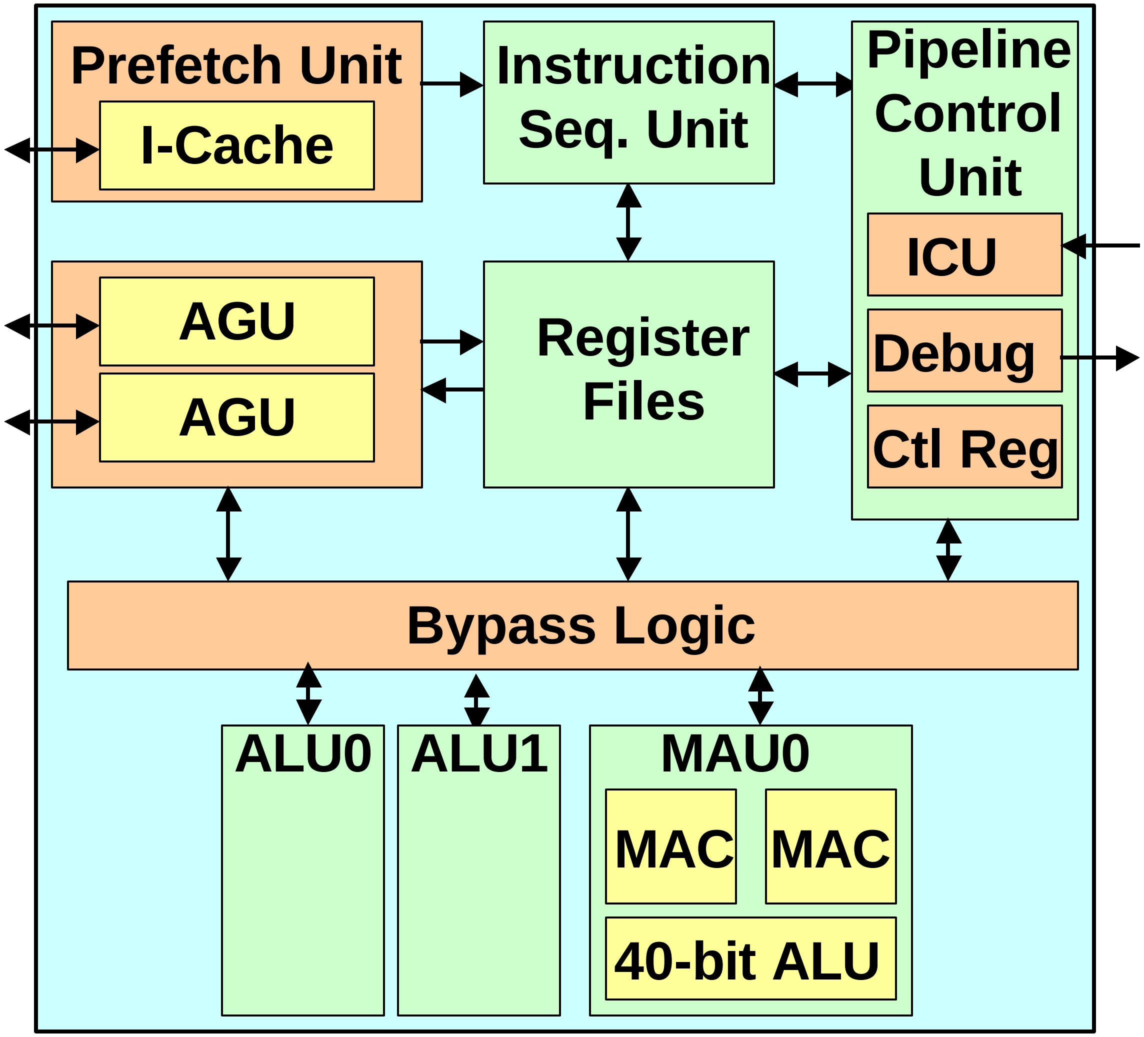
```
cmp r4,0
cexe (ge,nu) {
   mov r4,r5
   ldu r5,a2,1
   add r3,1
}
ext.e r2,r4,r7
```

```
mov r4,r5
ldu r5,a2,1
add r3,1
```

predicated on ($r4 \geq 0$)

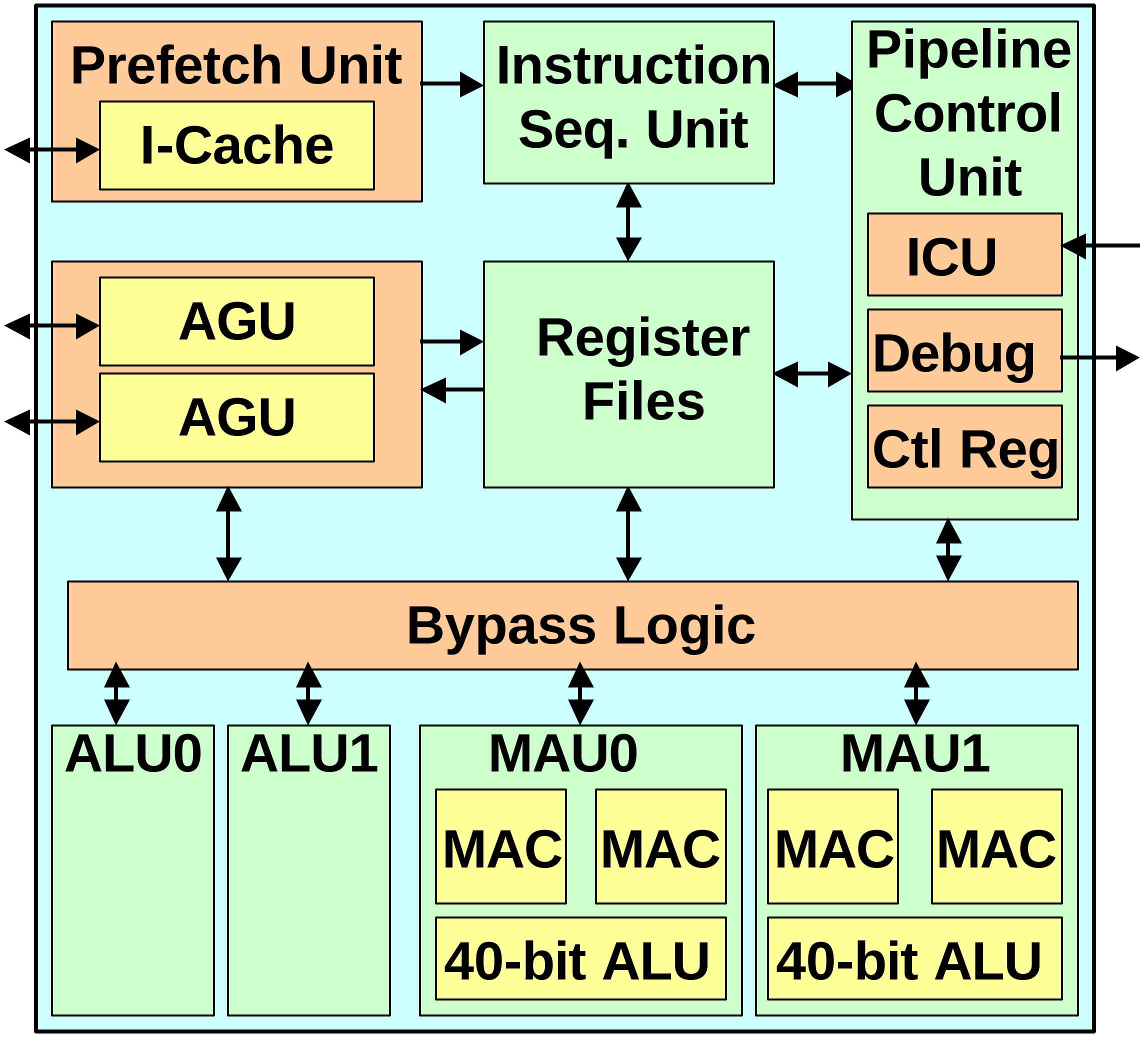
ZSP500 Implementation

- Single MAU unit capable of dual MAC operations
- 32-bit L/S ports
- 128-bit I-fetch
- 8x8 I-cache



ZSP600 Implementation

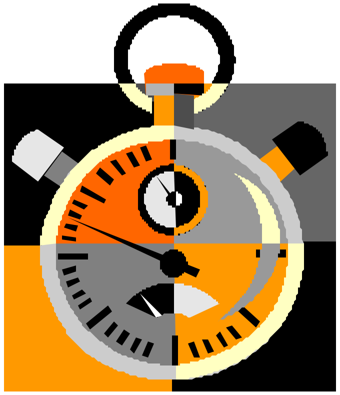
- Dual MAU units facilitate Quad MAC operations
- 64-bit L/S ports
- 128-bit I-fetch
- 8x8 I-cache



Performance

Benchmark	Cycle Formula
Real FIR (Buffer), T-tap, S=size	((S/2)*(T+1)) + 15
LMS FIR (One Sample), T-tap	5*(T/4) + 12
FFT, P-points	$\log_2(P)*(P+20)-P/4$

- Two-cycle FFT butterfly
- Full Viterbi butterfly per cycle
- 300 MHz, 0.13 μm



Summary "G2"

- High performance extension of the ZSP400
- Licensable, fully synthesizable RTL

Scalable

Compatible

Superscalar

- G2 implementations: ZSP600 & ZSP500

  Same ISA ; Binary Compatible

4.5X Boost

300MHz
0.13µm

Quad-MAC