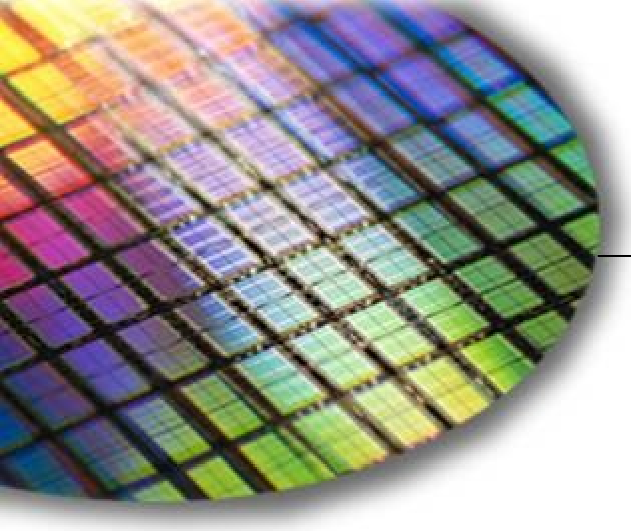


The World Leader in High Performance Signal Processing Solutions

SHARC® Digital Signal Processor Core Technology

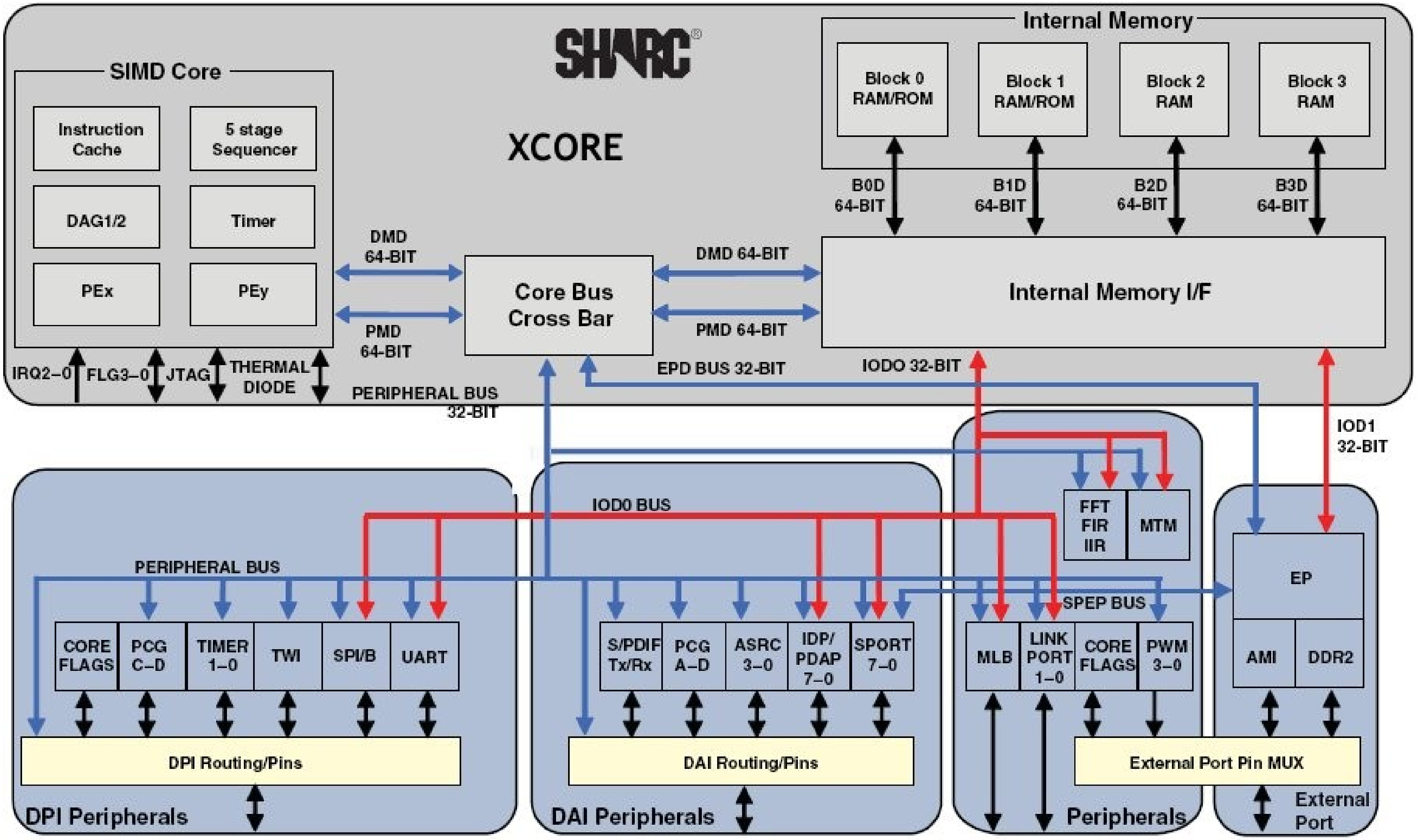
Abhijit Giri
Satya Simha
November 4th 2009

ANALOG DEVICES

Outline

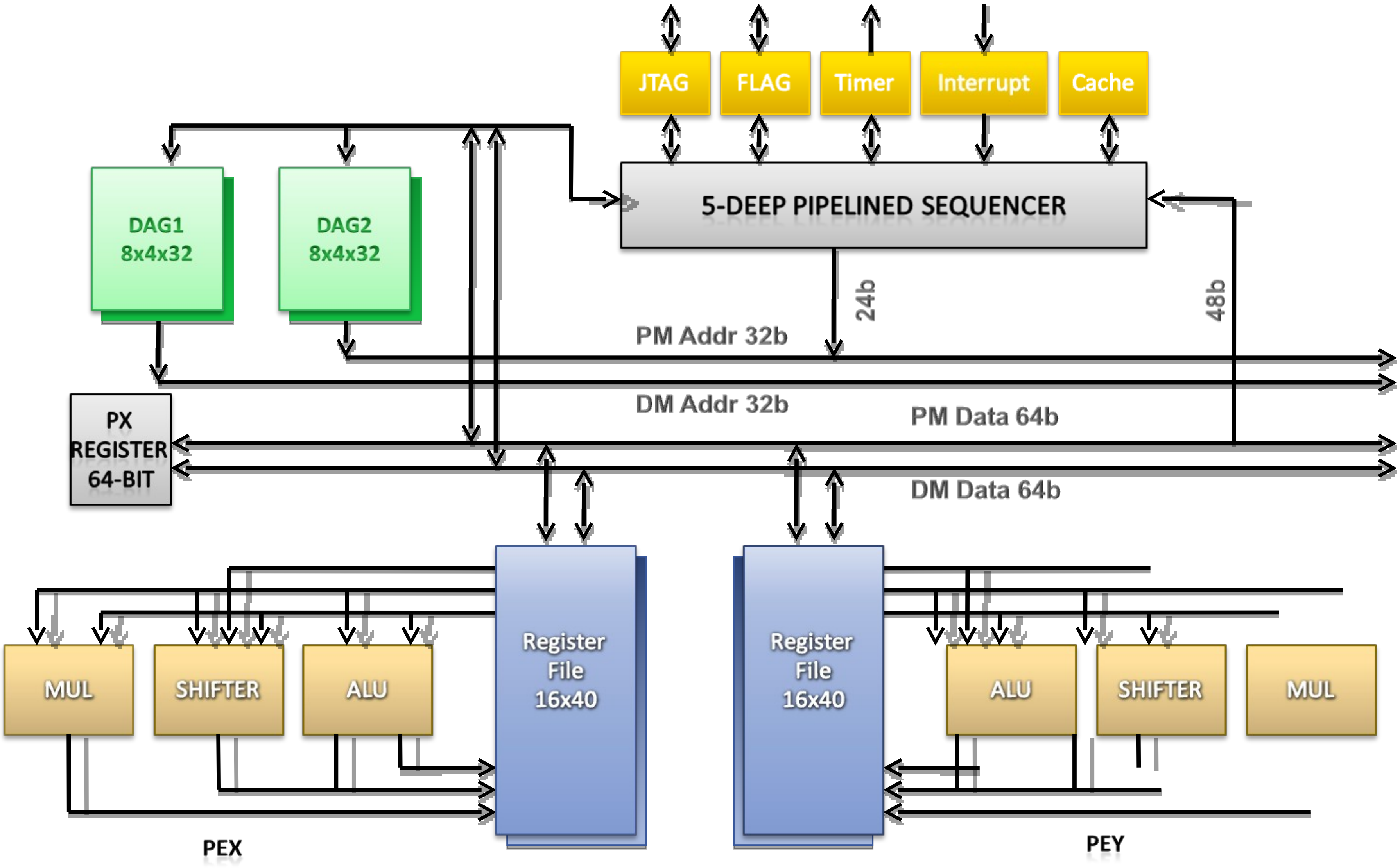
- Introduction to SHARC DSP – ADSP21469
- ADSP2146x Core
  - Compute unit
  - ISA
  - Memory Architecture
  - Connectivity
- Implementation Methodology
- ADSP 2146x Core as an IP

SHARC DSP & I/Os – ADSP21469



ANALOG DEVICES

SHARC ADSP2146x Core



SHARC® Architecture

- **High Performance IEEE-754 32-bit/40-bit Floating Point Processor**
- **Upward compatibility with the ADSP-21020 (SISD)**
- **Very deterministic architecture**
- **5 instruction pipeline stages (Protected)**
- **450 MHz (2.25ns) core instruction rate**
  - **Performs 2.7 GFLOPS / 900 MMACS**
- **Single-Instruction multiple-data (SIMD) computational architecture provides:**
  - **Two 32-bit floating point/ 32-bit fixed point/40-bit extended precision floating point computational units**
    - Each of two units has:
      - Multiplier
      - Arithmetic Logic Unit
      - Shifter
      - Register file
  - **Concurrent code execution**
  - **Single cycle execution of a Multiply or ALU operation**
  - **A dual memory read or write, and an instruction fetch**
  - **Transfers data between core & memory at a sustained 5.4GB/s bandwidth**

SHARC Instruction Set

- Multiple parallel operations packed in compact instructions
  - Variable length (16/32/48-bit long) instruction-encoding achieves compact code
- Almost all the instructions can be conditional; many also take ELSE clause
  - If..then..else constructs are compiled into compact and efficient code
- Branches can have delay slot
  - Minimizes wastage of cycles
- Hardware looping instructions
  - Zero-overhead looping
- Most instructions have a *compute* part
  - *Compute* can be single function or multi-function
- Algebric style instructions
  - Makes hand-coding easier

SHARC Instruction Set – contd..

- Multi-function compute packs multiply, add and subtract operations
  - Example: inner loop of a butterfly computation of FFT

```
f13 = f1*f4, f12 = f8+f12, f14 = f8-f12, f4 = dm(i2,m0), f1 = pm(i15,m9);
```

- Peak performance: 6*f MFLOPS (operation in SIMD)
  - 2.7 GFLOPS for 450MHz processor.

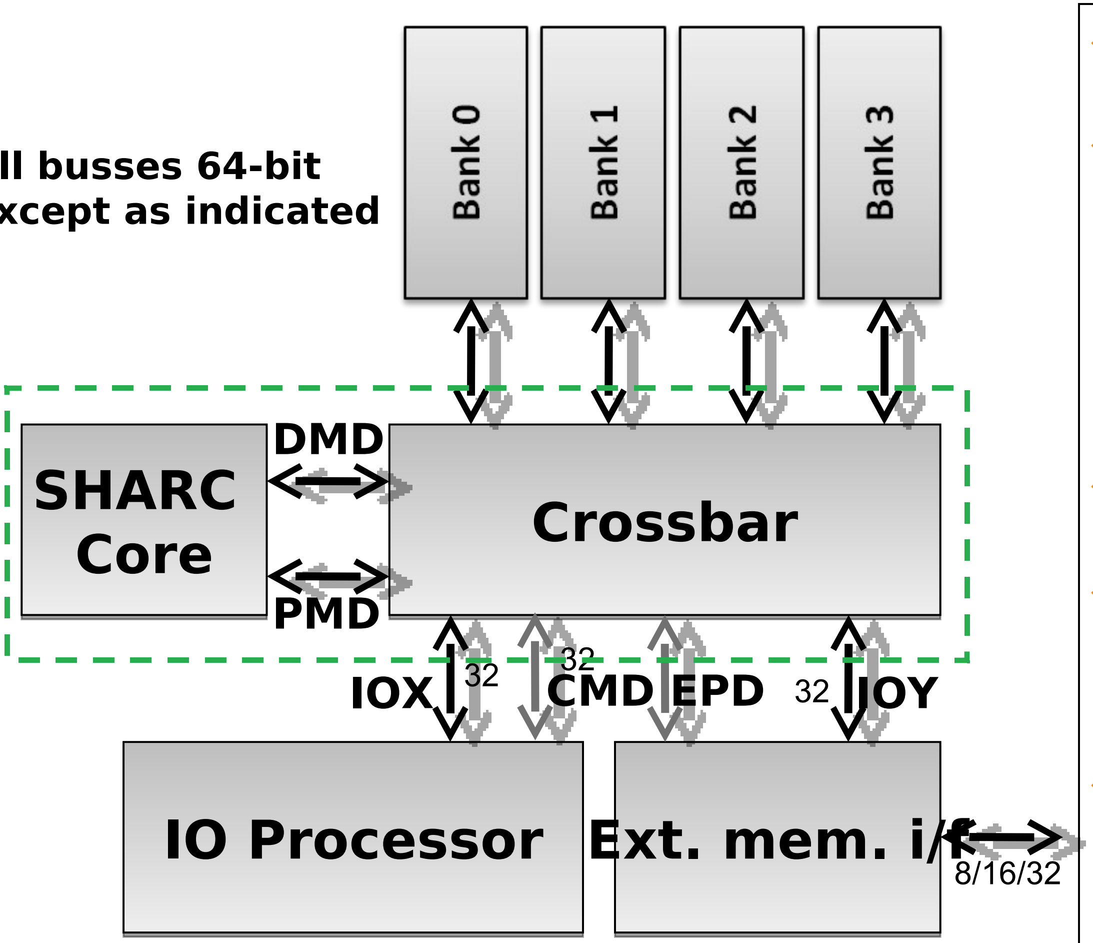
- Sustained peak MFLOPs is realizable due to
  - Single cycle multifunction compute
  - Parallel data load/store aided by DAGs from fast on-chip (L1) memory
  - Zero-overhead hardware looping
  - Shallow pipeline

Performance Benchmarks at 400 MHz

Benchmark Algorithm	Speed at 400 MHz
1024-Point Complex FFT (Radix 4, with Reversal)	23.25 us
FIR Filter (per Tap)	1.50 ns
IIR Filter (per Biquad)	5.00 ns
Matrix Multiply (Pipelined)	
[3 x 3] x [3 x 1]	11.25 ns
[4 x 4] x [4 x 1]	20.00 ns
Divide (y/x)	8.75 ns
Inverse Square Root	13.50 ns

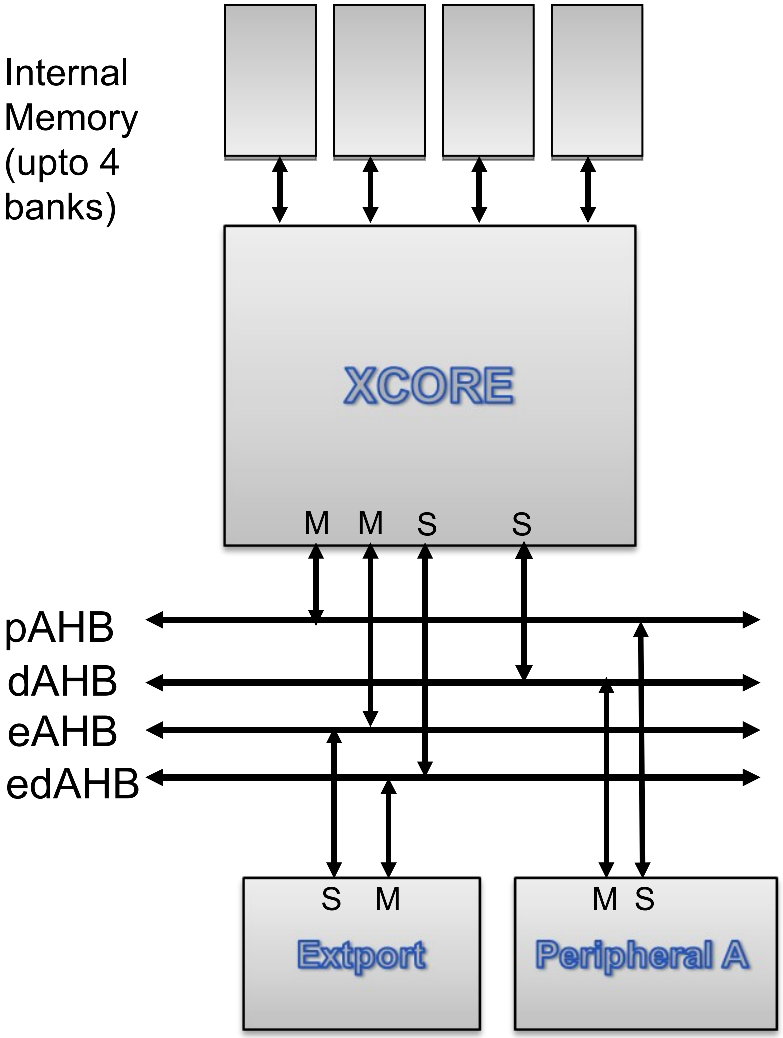
Memory System

All busses 64-bit except as indicated



- Harvard architecture – Instr and data busses
- 4-banked on-chip L1 at core speed
  - Fully addressable
  - Each bank 64-bit wide
  - Standard 1-deep pipelined interface
  - Can be compiler-generated
- Full Crossbar interconnect
  - 4 accesses if no conflict
- Supports large amount (16Mb) of on-chip memory
  - ADSP21469 populated with 5Mb RAM + 4Mb ROM
- Directly addressable off-chip memory
  - DMA transfer between on-chip and off-chip

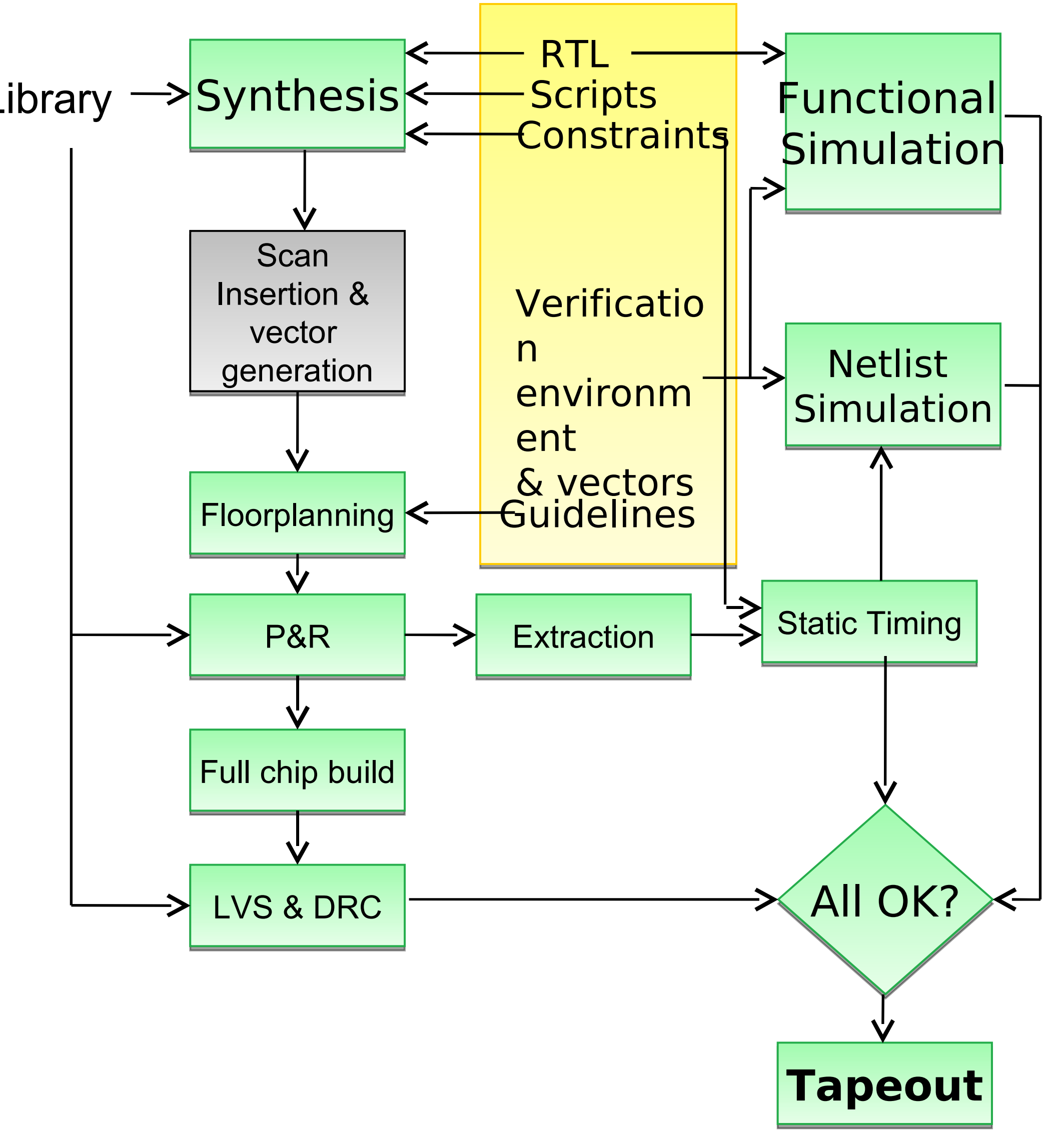
System Bus Interfaces



- **Core interfaces with IOP system over 4 AHB busses through appropriate bridges**
  - **pAHB – for MMR accesses**
  - **eAHB – for MMR accesses in ext. mem i/f, as well as direct off-chip access**
  - **edAHB – for DMA to/from ext. mem**
  - **dAHB – for DMA to/from all other peripherals**

M: Unit can master the bus
S: Unit is slave on the bus

Implementation Methodology



- Design in Verilog HDL
- ASIC standard design flow
  - Cell based Synthesis
    - Some custom cells used for performance
  - Auto P&R
    - Clock-tree synthesis
  - Static timing based timing sign-off
  - Coupling analysis/fixing
  - IR and EM checks/fixes
  - LVS/DRC etc..
- Verification
  - Self-checking directed tests
  - Constrained random test
  - Formal and semi-formal

SHARC Core

- 5 stage pipeline core (SIMD SHARC-V)
- Standard memory interface for internal memory
- AHB compliant interfaces for peripherals

- Flip-Flop based design with few latches
  - Design in synthesizable Verilog RTL
- Scan-ready
- Frequency – depends on choice of technology / implementation
  - 450MHz in a 65nm technology (optimized for high performance)

SHARC Core IP collateral

- **Core in Verilog RTL**
  - **Synthesizable Verilog RTL for simulation as well as synthesis**
  - **Simulation environment – standalone core environment**
    - Tests for design verification
  - **Synthesis scripts and guidelines for DCT (Synopsys)**
    - Clock descriptions, timing and other constraints, exceptions
- **Documentation of interfaces (memory, peripherals)**
- **C simulator of the core**
- **Documentation on Clocking guidelines (inside core), DFT, input clocking requirements of the core, power-on and reset.**
- **Physical design guidelines**
- **Design support as required**

ADSP 21469 I/O Peripherals

❑ Serial Ports

❑ SPDIF

❑ I2C®-compatible 2-wire interface

❑ UART

❑ SPI

❑ Timers

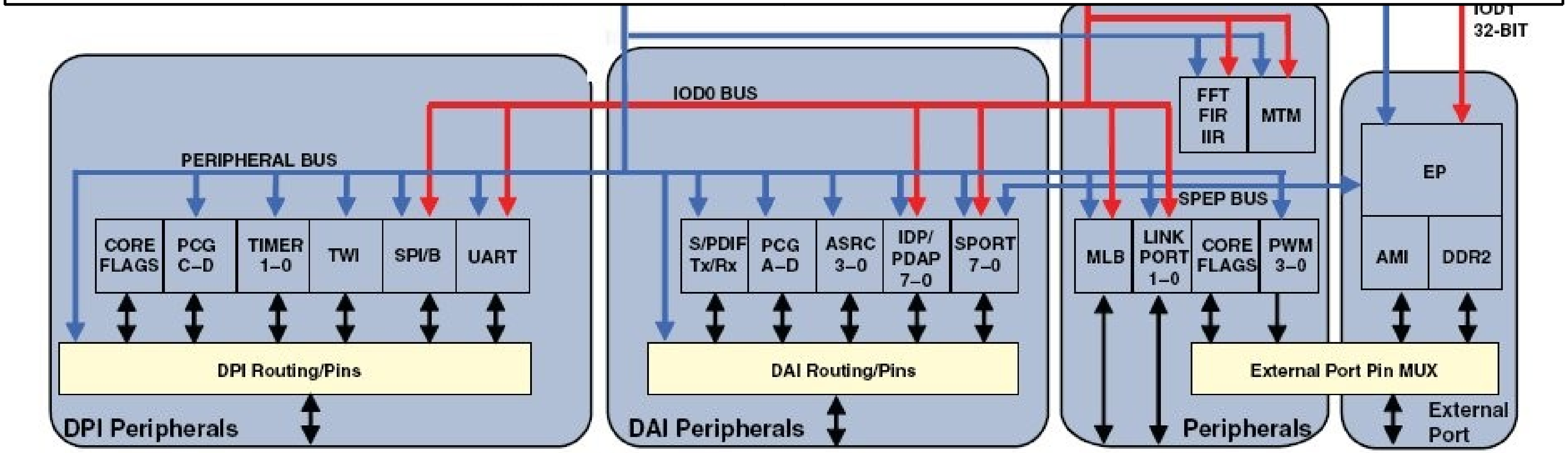
- Pulse with count
- PWM waveform generation

❑ Link Ports – 8-bit bi-directional port with Clock and ACK for fast link

❑ External memory interface

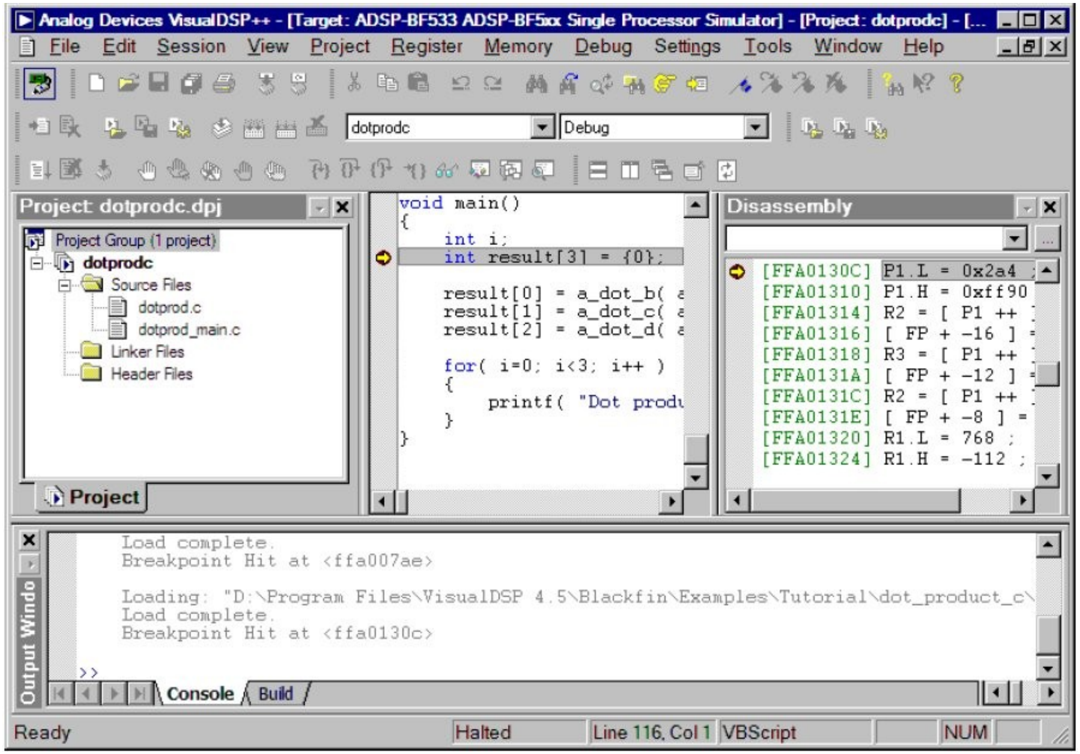
- DDR2 and AMI

❑ All are RTL based designs and are implemented in standard ASIC flow in one hierarchy.



ADI VDSP++ Development Tools

- VisualDSP++ provides an IDDE, which provides easy access to
  - Editor
  - Compilers C/C++
  - VDK RTOS/Kernel
  - Assembler
  - Linker
  - Simulator including MP
  - Emulator/debugger
- Plug-ins for easy programming of some of the peripherals


Analog Devices VisualDSP++ - [Target: ADSP-BF533 ADSP-BF5xx Single Processor Simulator] - [Project: dotprodc] - [...
File Edit Session View Project Register Memory Debug Settings Tools Window Help
dotprodc
Debug
Project: dotprodc.dpj
Project Group (1 project)
dotprodc
Source Files
dotprod.c
dotprod_main.c
Linker Files
Header Files
Project
Disassembly
Output Windo
Console
Build
Ready
Halted
Line 116, Col 1
VBScript
NUM

```
void main()
{
     int i;
     int result[3] = {0};

     result[0] = a_dot_b( a
     result[1] = a_dot_c( a
     result[2] = a_dot_d( a

     for( i=0; i<3; i++ )
     {
         printf( "Dot produ
     }
}
```

```
[FFA0130C] P1.L = 0x2a4 ;
[FFA01310] P1.H = 0xff90
[FFA01314] R2 = [ P1 ++ ]
[FFA01316] [ FP + -16 ] =
[FFA01318] R3 = [ P1 ++ ]
[FFA0131A] [ FP + -12 ] =
[FFA0131C] R2 = [ P1 ++ ]
[FFA0131E] [ FP + -8 ] =
[FFA01320] R1.L = 768 ;
[FFA01324] R1.H = -112 ;
```

```
Load complete.
Breakpoint Hit at <ffa007ae>

Loading: "D:\Program Files\VisualDSP 4.5\Blackfin\Examples\Tutorial\dot_product_c\
Load complete.
Breakpoint Hit at <ffa0130c>

>>
```

Summary

- ADSP 21469 is a modern 32-bit floating point DSP
- Delivers 2.7GFLOPS at 450MHz
- DSP Core can easily be integrated into any SoC
  - Synthesis-P&R-ready
  - Includes standard interfaces

Thank You

SHARC ADSP2146x Core - Summary

- 32-bit architecture
- Code compatibility with other SHARC family members at the assembly level
- Single instruction multiple data (SIMD) architecture provides
  - Two computational processing elements
  - Concurrent execution
- Compute units support
  - IEEE Single precision floating point (32-bit)
  - 40-bit extended precision floating point for 32-bit resolution in floating point computations
  - Also 32-bit fixed point
- Dual data address generators (DAGs)
  - modulo (for circular buffers) and bit-reverse (FFT) addressing
- Sequencer supports
  - Zero-overhead looping with single-cycle loop setup
  - Low-overhead branching
  - VISA (variable instruction set) execution support
- Parallelism in buses and computational units allows
  - Single cycle executions (with or without SIMD) of a multiply operation, an ALU operation, a dual memory read or write, and an instruction fetch
- 5-deep pipeline - fully interlocked