Analog Devices SHARC

- by Russell Greenspan
- CS433 Spring 2005

Overview

- Processor History
- Physical packaging
- Data paths, register files, computational units
- Pipelining, timing information
- Memory
- Instruction Set Architecture (ISA)
- Applications targeted
- Systems employing the SHARC

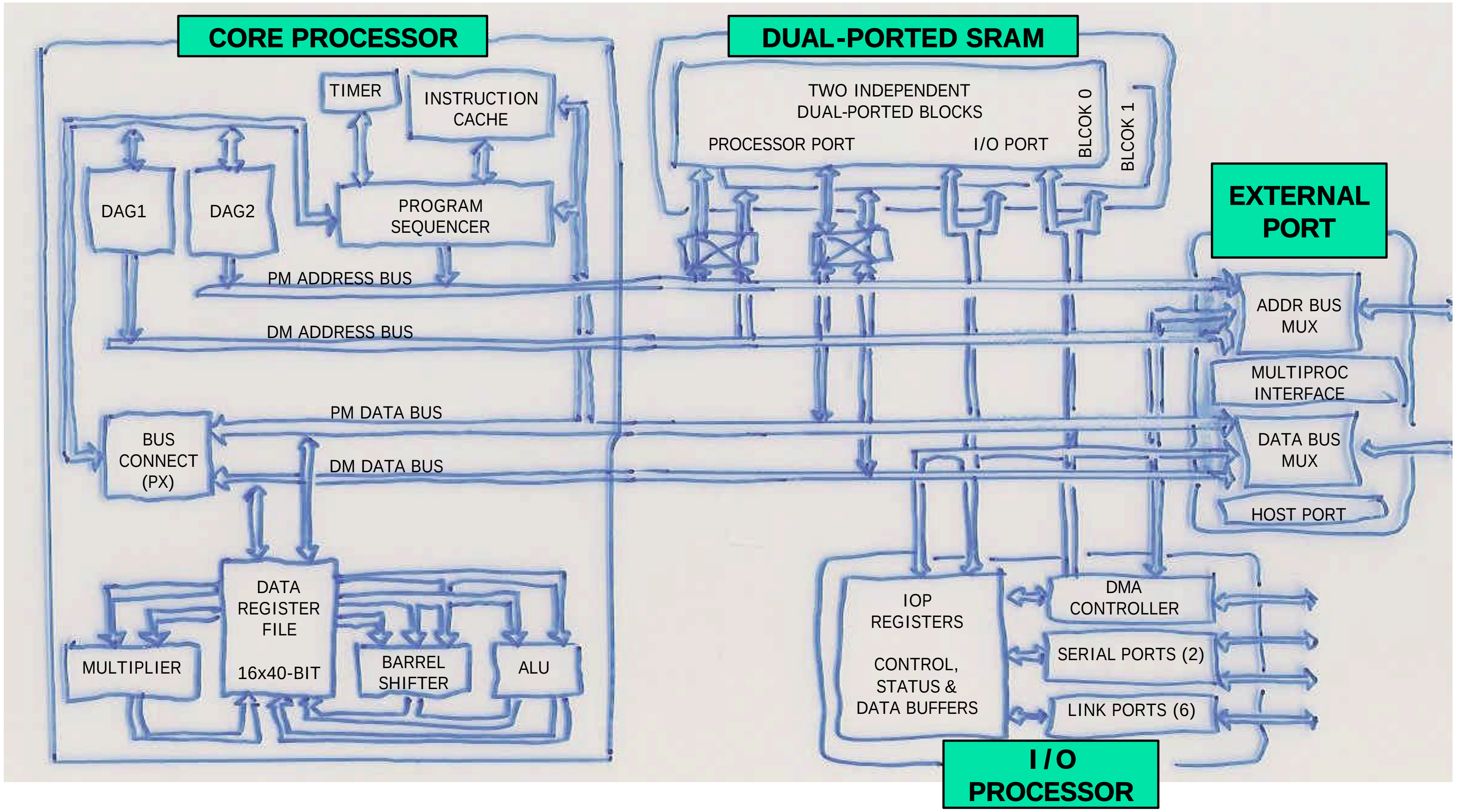
SHARC Features

- **S**uper **H**arvard **ARC**hitecture
  - Unique CISC architecture allows simultaneous fetch of two operands and an instruction in one cycle
- Combines high performance DSP core with integrated, on-chip system features
  - Dual-ported (processor and I/O) SRAM
  - DMA Controller
- Selective Instruction Cache
  - Cache only those instructions whose fetches conflict with program memory data accesses

SHARC Processor History

- ADSP-2106*x* (2000)
  - Single computational units based on predecessor ADSP-2100 Family
  - 40 MHz core
- ADSP-2116*x* (2001)
  - SIMD (Single-Issue Multiple-Data) dual computational unit architecture added
  - 150-200 MHz core, 1-2 MB RAM
- ADSP-2126*x*, ADSP-2136*x* (2003 – Future)
  - Integrated audio-centric peripherals (128-140db Sample Rate Conversion) added
  - 333-400 MHz core, 2-3 MB RAM

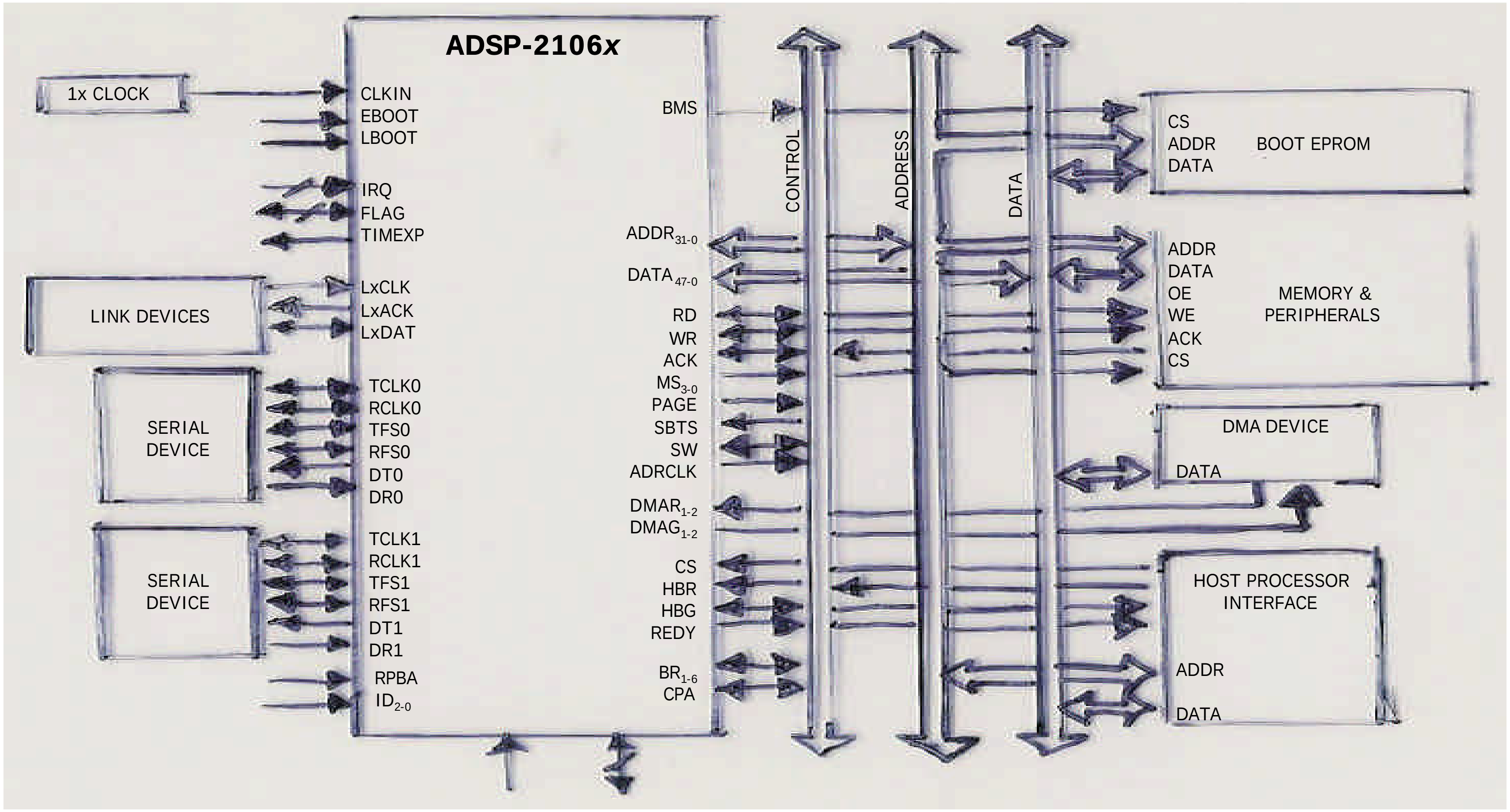
ADSP-2106*x* Overview



ADSP-2106x Core

- Computational Units
  - ALU, Multiplier, and Shifter can all perform independent operations in a single cycle
- Register File
  - Two sets (primary and alternate) of 16 registers, each 40-bits wide
- Program Sequencer and Data Address Generators
  - Allows computational units to operate independent of instruction fetch and program counter increment

ADSP-2106*x* Packaging



ADSP-2106x Key Pins

PIN	FUNCTION	NOTE
$ADDR_{31\text{-}0}$	External Bus Address	
$DATA_{47\text{-}0}$	External Bus Data	
$MS_{3\text{-}0}$	Memory Select Lines	Asserted (low) as chip selects memory bank
PAGE	DRAM Page Boundary	Asserted if a page boundary is crossed
DMAR(1-2)	DMA Request 1 and 2	
$IRQ_{2\text{-}0}$	Interrupt Request Lines	Edge-triggered or level-sensitive

ADSP-2106x Registers

- Data Registers
  - R15 – R0 (fixed-point), F15 – F0 (floating-point)
- Program Sequencer
  - PC (program counter), PCSTKP (PC stack pointer), FADDR (fetch address), etc.
- Data Address Generator
  - I7 – I0 (DAG1 index), M7 – M0 (DAG1 modify)
  - L7 – L0 (DAG1 length), B7 – B0 (DAG1 base)
- Bus Exchange, Timer, and System Registers

ADSP-2106x Buses

- Address
  - Program Memory Address – 24 bits wide
  - Data Memory Address – 32 bits wide
- Data
  - Program Memory Data – 48 bits wide
    - Stores instructions and data for dual-fetches
  - Data Memory Data – 40 bits wide
    - Stores data operands
- One PM Data bus and/or one DM Data bus register file access per cycle

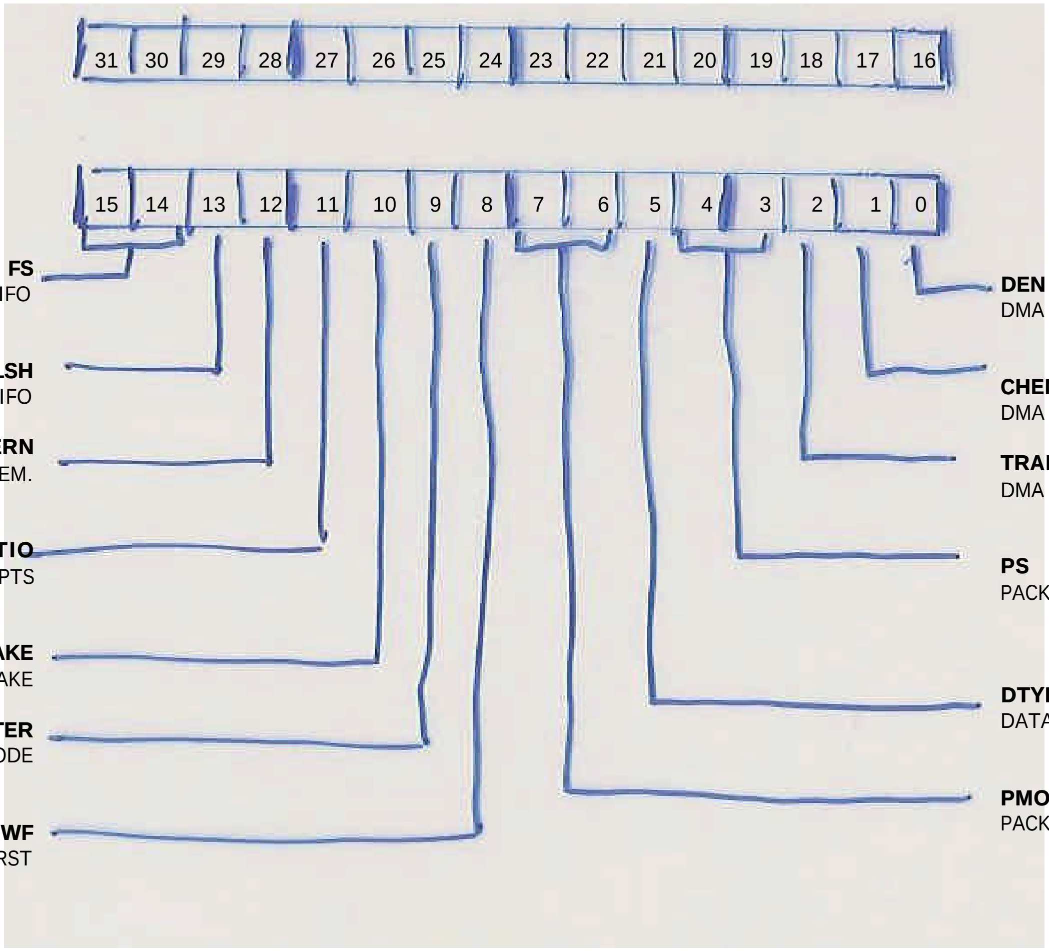
ADSP-2106*x* I/O

- Serial Ports
  - Operate at clock rate of processor
- DMA
  - Port data can be automatically transferred to and from on-chip memory

ADSP-2106*x* DMA

- I/O port block transfers (link/serial)
- External memory block transfers
- DMA Channel setup by writing memory buffer parameters to DMA parameter registers
  - Starting Address for Buffer
  - Address Modifier
  - Word Count
- Interrupt generated when transfer completes (i.e. Word Count = 0)

ADSP-2106*x* DMA Registers



FS
EXT. PORT FIFO

FLSH
FLUSH EXT. PORT FIFO

EXTERN
EXT. DEVICES TO EXT. MEM.

INTIO
SINGLE-WORD INTERRUPTS

HSHAKE
DMA HANDSHAKE

MASTER
DMA MASTER MODE

MSWF
MOST SIGNIFICANT WORD FIRST

DEN
DMA ENABLE

CHEN
DMA CHAINING ENABLE

TRAN
DMA CHANNEL DIRECTION

PS
PACKING STATUS

DTYPE
DATA TYPE

PMODE
PACKING MODE

ADSP-2106*x* Pipelining

- Three phases
  - Fetch
    - Read from cache or program memory
  - Decode
    - Generate conditions for instruction
  - Execute
    - Operations specified by instruction completed

ADSP-2106*x* Branching and Pipelining

- Branches
  - Delayed
    - Two instructions after branch are executed
  - Non-delayed
    - Program sequencer suppresses instruction execution for next two instructions

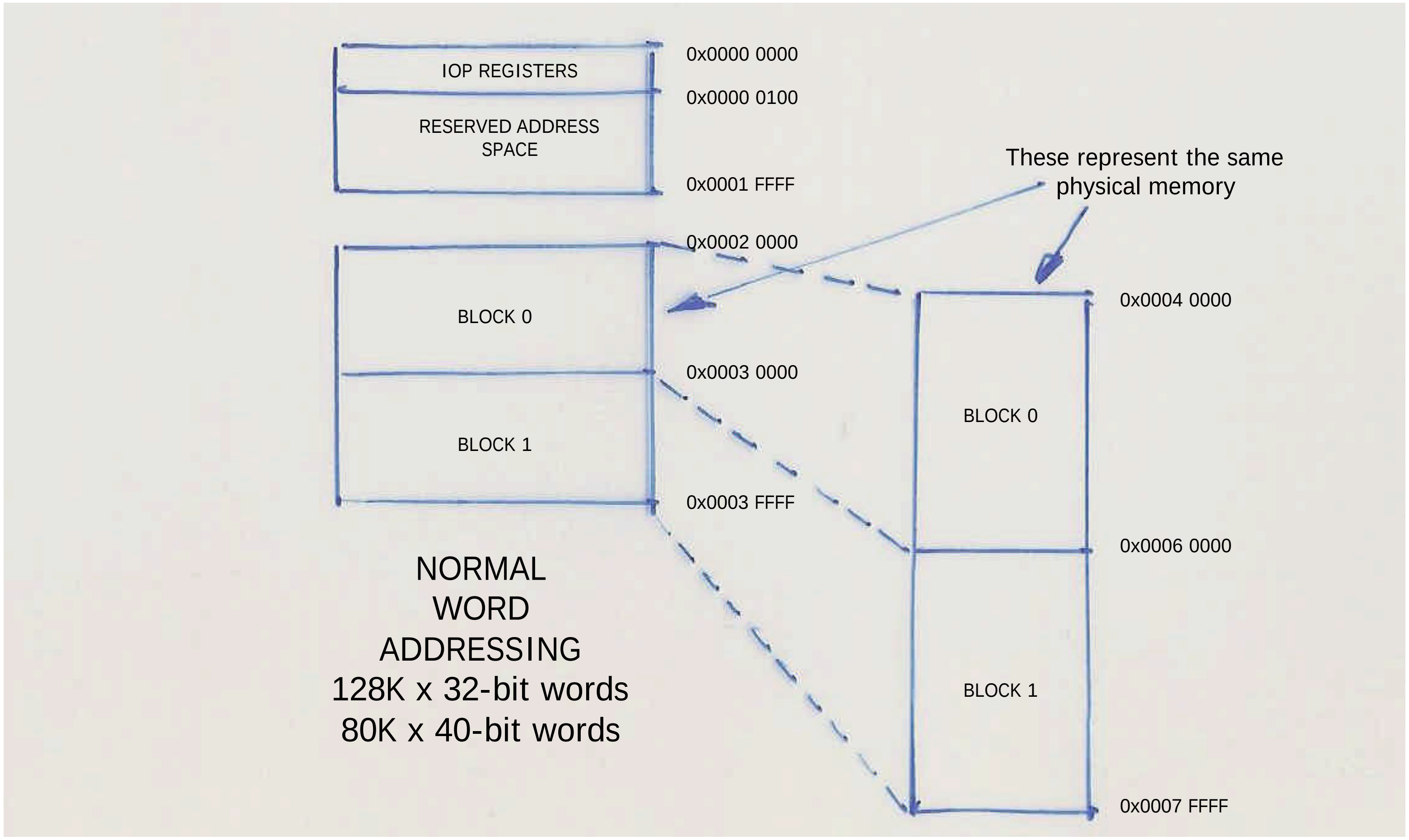
CLOCK CYCLES →				
Fetch	*n* + 2	j	*j* + 1	*j* + 2
Decode	*n* + 1	*n* + 2	*j*	*j* + 1
Execute	*n*	*no-op* *n* + 1	*no-op* *n* + 2	*j*
	Non-delayed *Delayed*			

ADSP-2106x Memory

On-Chip SRAM	ADSP-21060	ADSP-21062	ADSP-21061
Total Size	500KB	250KB	125KB

- On-chip support for:
  - 48-bit instructions (and 40-bit extended precision floating-point data)
  - 32-bit floating-point data
  - 16-bit short word data
- Off-chip memory up to 4 GB

ADSP-2106*x* Memory (2)



ADSP-2106*x* Memory (3)

- Memory divided into blocks
- Dual-ported (PM and DM bus share one port, I/O bus uses the other)
  - Allows independent access by processor core and I/O processor
  - Each block can be accessed by both in every cycle
- Typical DSP applications (digital filters, FFTs, etc.) access two operands at once, such as a filter coefficient and a data sample, so allowing single-cycle execution is a must

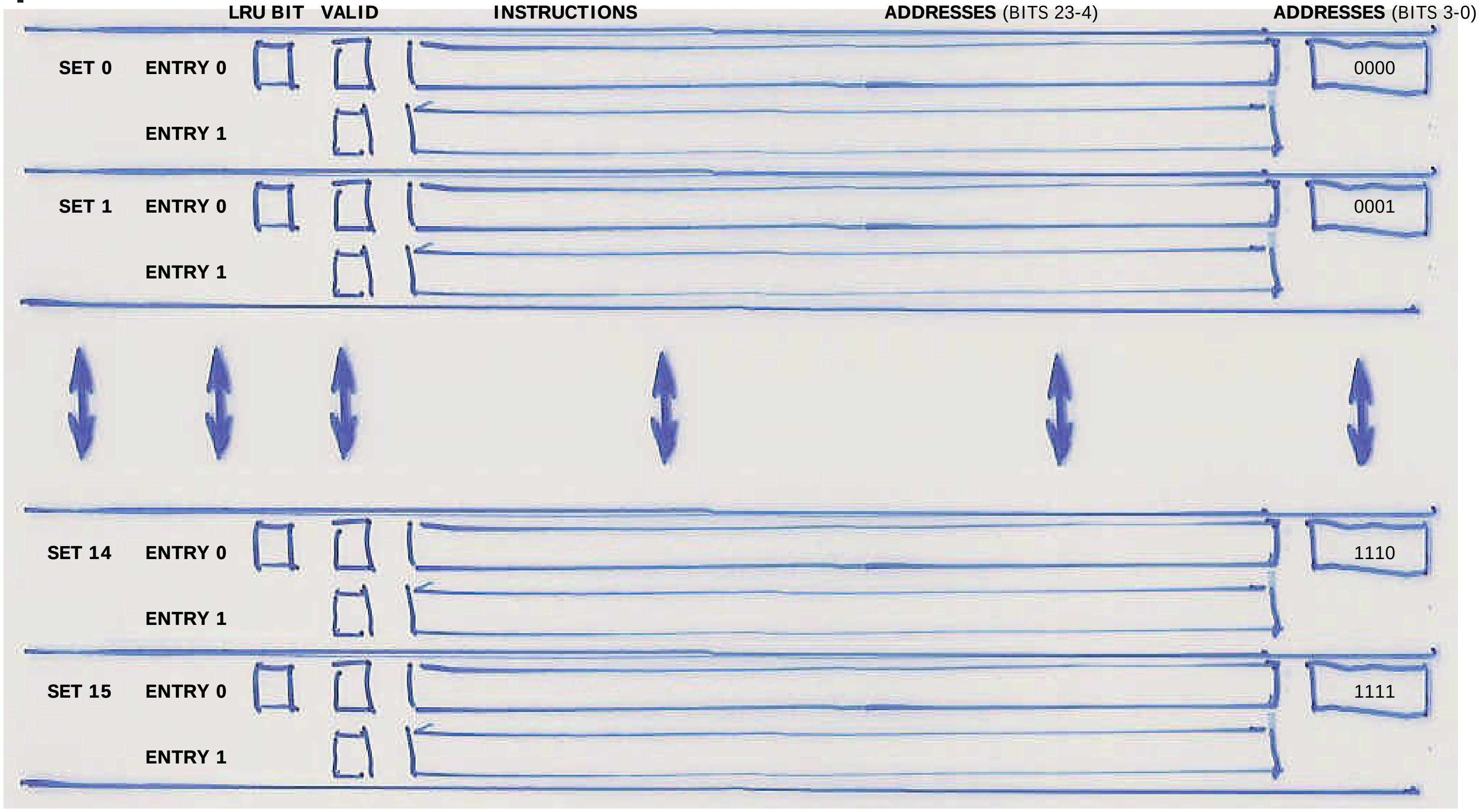
ADSP-2106x Shadow Write

- Due to need for high-speed operations, memory writes to a two-deep FIFO
- On write, data in FIFO from previous write is loaded to memory and new data enters FIFO
- Reads of last two written locations are intercepted and re-routed to the FIFO

ADSP-2106x Instruction Cache

- Sequencer checks instruction cache on every program memory data access
- Allows PM bus to be used for data fetches instead of being tied up with an instruction fetch
- When fetch conflict first occurs, instruction is cached to prevent the same delay from happening again

ADSP-2106*x* Instruction Cache (2)



ADSP-2106*x* ISA Overview

- 24 operations, although some have more than one syntactical form
- Instruction Types
  - Compute and Move
    - Compute operation in parallel with data moves or index register modify
  - Program Flow Control
    - Branch, Call, Return, Loop
  - Immediate Data Move
    - Operand or addressing immediate fields
  - Miscellaneous
    - Bit Modify and Test, No-op, etc.

ADSP-2106x ISA Compute and Move

- Instructions follow the format

    `IF condition op1, op2;`

- *`IF`* and *`condition`* are optional
- *`op1`* is an optional compute instruction
- *`op2`* is an optional data move instruction

ADSP-2106x ISA Compute Examples

- Single function
  - `F6 = (F2 + F3);`
- Multi-function
  - Distinct parallel operations supported
  - Parallel computations and data transfers
    - `R1 = R2 * R6, M4 = R0;`
  - Simultaneous multiplier and ALU operations
    - `R1 = R2 * R6, F6 = F2 + F3;`

ADSP-2106x ISA Single function Compute

22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	CU		OPCODE								RN				RX				RY			

- CU specifies
  - 00 – ALU
  - 01 – Multiplier
  - 02 – Shifter
- OPCODE indicates operation type (add, subtract, etc.)
- RN specifies result register
- RX and RY specify operand registers

ADSP-2106x ISA Multi-function Compute

- Parallel ALU and Multiplier operations

22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	OPCODE						RM				RA				RXM		RYM		RXA		RYA	

- Registers restricted to particular sets
  - Multiplier X: R3 – R0, Y: R7 – R4
  - ALU X: R11 – R8, Y: R15 – R12
- OPCODE specifies ALU op, for example:
  - 000100: `Rm = R3-0 * R7-4, Ra = R11-8 + R15-12;`
  - 011111: `Rm = R3-0 * R7-4, Ra = MIN(R11-8, R15-12);`

ADSP-2106x ISA
Program Flow Control

- Instructions follow the format

    `IF condition JUMP/CALL, ELSE op2;`

- *`IF`*, *`condition`*, *`ELSE`* are optional
- `JUMP/CALL` is a JUMP or CALL instruction
- *`op2`* is an optional compute instruction

ADSP-2106x ISA Program Flow Control (2)

- Instructions follow the format

```
DO <addr24> UNTIL termination;
```

- No optional fields
- `<addr24>` is the loop start address
- `termination` is the loop ending condition to check after each iteration

ADSP-2106x ISA Program Flow Examples

- Conditional Execution
  - `IF GT R1 = R2 * R6;`
  - `IF NE JUMP label2;`
- Also used for Call/Return

```
   main: CALL routine;
routine: ...
         RTS; /*return to main*/
```

ADSP-2106x ISA Immediate Data Move

- Instructions follow the format

```
ureg = <data32>;
DM(<data32>, Ia) = ureg;
PM(<data24>, Ia) = ureg;
```

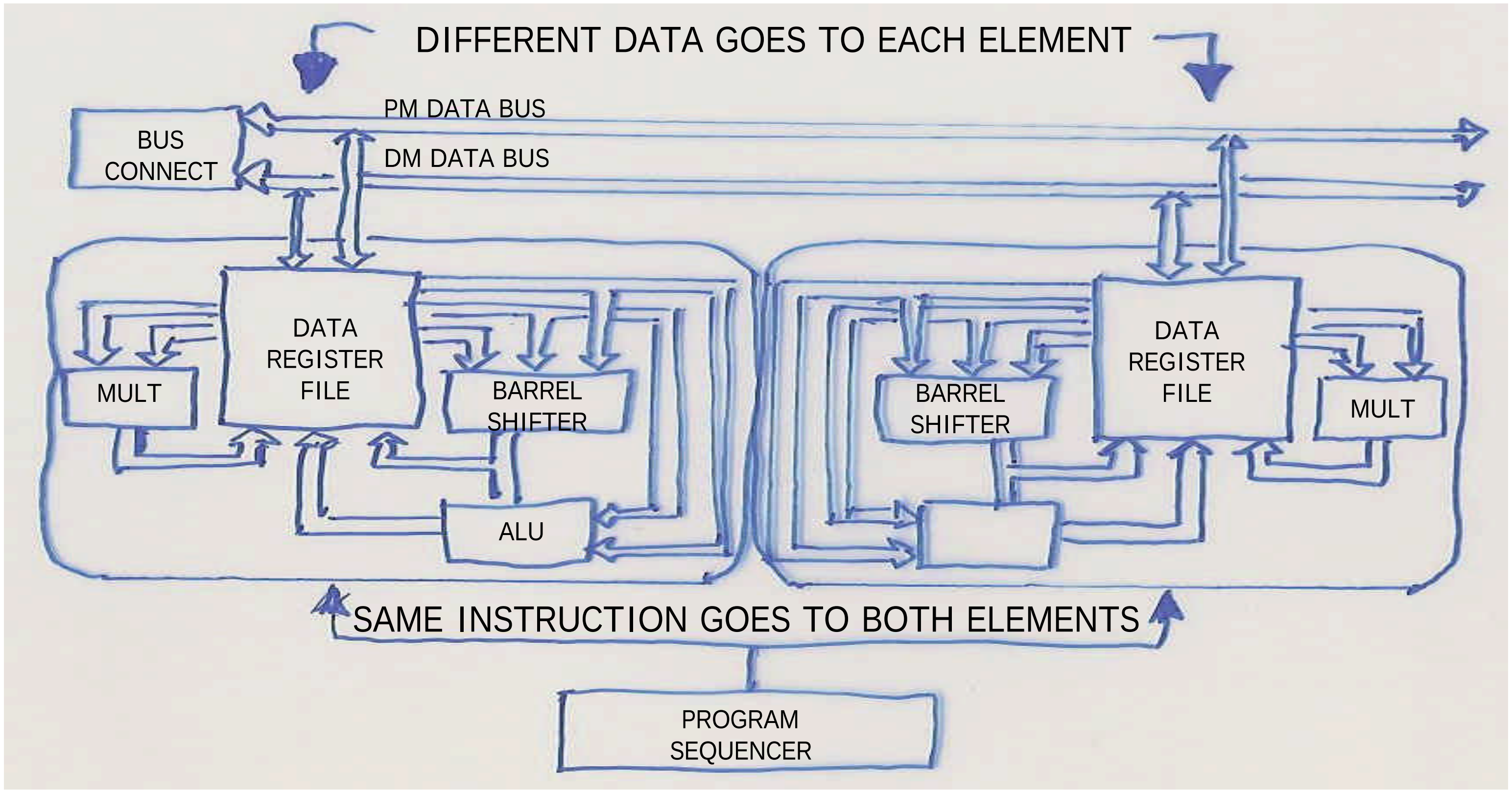
- `Ia` is an optional indirect addressor
- DM is a 32-bit data memory address
- PM is a 24-bit program memory address

ADSP-2106x ISA Addressing Examples

- Direct
  - `JUMP <data24>;`
- Relative to Program Counter
  - `JUMP (PC, <data24>);`
- Register Indirect (using DAG registers)
  - Pre-Modify (modification pre-address calculation)
    - `JUMP (M0, I0);`
  - Post-Modify (modification post-address calculation)
    - `JUMP (I0, M0);`

ADSP-2116*x* Overview

- Extension of 2106*x*, adding 150Mhz core and SIMD (Single-Issue Multiple-Data) support via dual hardware



ADSP-2116*x* SIMD Engine

- Dual hardware allows same instruction to be executed across different data
  - 2 ALUs, multipliers, shifters, register files
  - Two data values transferred with each memory or register file access
  - Very effective for stereo channel processing
- Can effectively double performance over similar algorithms running on ADSP-2106*x* processors

ADSP-2116*x* SIMD Engine (2)

- Enabled/disabled via `MODE1` bit
  - When disabled, processor simply acts in SISD mode
- Program sequencer must be aware of status flags set by each set of hardware elements
- Conditional compute operations can be specified on both, either, or neither hardware set
- Conditional branches and loops executed by AND'ing the condition tests on both hardware sets

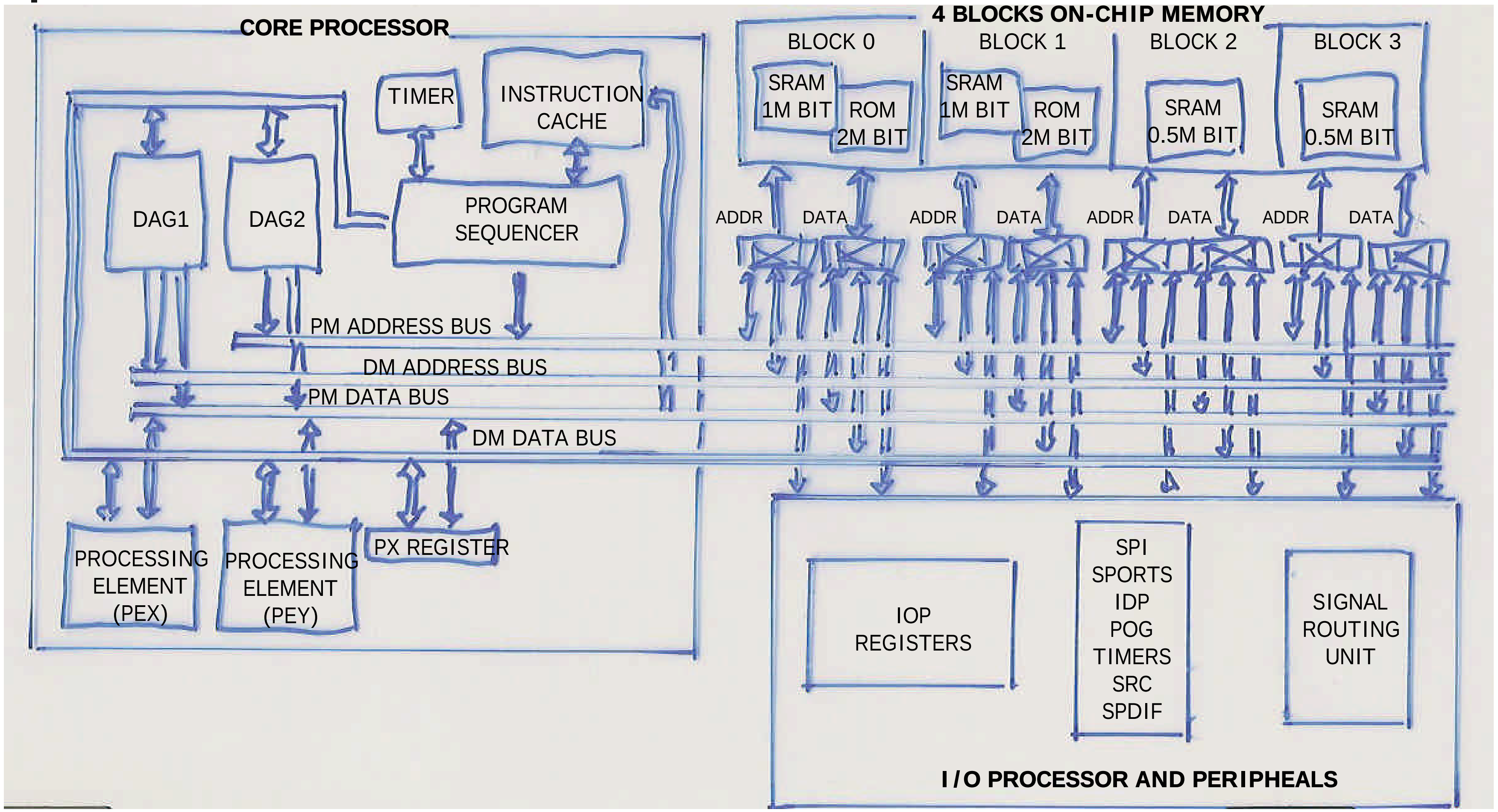
ADSP-2116x SIMD Engine (3)

Instruction	Mode	Transfer 1	Transfer 2
Rx = Ry;	SISD	Rx loaded from Ry	n/a
	SIMD	Rx loaded from Ry	Sx loaded from Sy
Sx = Sy;	SISD	Sx loaded from Sy	n/a
	SIMD	Sx loaded from Sy	Rx loaded from Ry
Rx = Sy;	SISD	Rx loaded from Sy	n/a
	SIMD	Rx loaded from Sy	Sx loaded from Ry
Sx = Ry;	SISD	Sx loaded from Ry	n/a
	SIMD	Sx loaded from Ry	Rx loaded from Sy

ADSP-2126*x* Overview

- Direct extension of 2116*x*, instructions are fully backward compatible
- Core increased to 150-200 MHz w/ 1MB SRAM
- Data buses increased from 32 to 64 bits
- Synchronous, independent serial ports increased from 2 to 6
- ROM-based security
  - Prevents piracy of code and algorithms
  - Prevents peripheral devices from reading on-chip memory

ADSP-2136x Overview



ADSP-2136*x* Overview (2)

- Direct extension of 2126*x*, instructions are fully backward compatible
- On-chip memory expanded from 2 to 4 blocks
- Digital Audio Interface (DAI) set of audio peripherals
  - Interrupt controller, interface data port, signal routing unit, clock generators, and timers
  - Different units contain S/PDIF receiver/transmitter, sample rate converters, or DTCP encrypting engine

SHARC Benchmarks

- Algorithm benchmarks supplied by manufacturer:

	2106*x*	**2116*x***	**2126*x***	**2136*x***
Clock Cycle	66 MHz	100 MHz	200 MHz	333 MHz
Instruction Cycle Time	15 ns	10 ns	6.67 ns	3 ns
MFLOPS Sustained	132 MFLOPS	400 MFLOPS	600 MFLOPS	1332 MFLOPS
MFLOPS Peak	198 MFLOPS	600 MFLOPS	900 MFLOPS	1998 MFLOPS
FIR Filter (per tap)	15 ns	5 ns	2.5 ns	1.5 ns
IIR Filter (per biquad)	61 ns	20 ns	10 ns	6 ns
Divide (y/x)	91 ns	30 ns	20 ns	9 ns

Applications Targeted

- SHARC designed to
  - Simplify Development
  - Speed time to Market
  - Reduce Product Costs
- Product targeted
  - A/V Receivers
    - 7.1 Surround Sound Decoding
  - Mixing Consoles
  - Digital Synthesizers
  - Automobiles

Systems Employing the SHARC

- SRS Circle Surround II
- Melody (w/ Auto Room Tuner)
- Metric Halo's Portable Pro Audio Hub
- Alacron FT-P5

SHARC in SRS Circle Surround II

- Full multi-channel surround sound from simple right/left stereo sound
- Encoding can be transmitted over standard stereo medium (broadcast television, radio, etc.) and maintains full backward compatibility

SHARC in SRS Circle Surround II (2)

- Output from each source is combined in constant phase filter banks and encoded in quadrature to prevent signal cancellation
- "Positional bias generator" analyzes ratios between left and right surround signals which multipliers then apply to the opposing left or right output
- Decoder uses this level imbalance to direct the surround information to the correct output

SHARC Melody

- “Optimized Surround Sound for the Mass Market”
- Core of high-fidelity audio decoders in Denon, Bose, and Kenwood products
- Auto Room Tuner (ART) integrated software simplifies setup of complex audio systems

SHARC Melody ART

- Automatically measures and corrects multi-channel sound system for room’s acoustics
- Corrects system deficiencies
- For each speaker, ART calculates:
  - Sound pressure level (SPL)
  - Distance of each speaker from listener
  - Frequency response

SHARC in Metric Halo's Portable Pro Audio Hub

- Portable FireWire-based recording device, used in live recordings applications by motion pictures and major recording artists like “No Doubt” and “Dave Mathews Band”
- Serial ports used to interface to digital and mixed-signal peripheral devices
- Initially implemented on SHARC ADSP-2106*x*, later upgraded to ADSP-2126*x*
- Future hybrid implementation will use a ADSP-2106*x* for FireWire processing coupled with a ADSP-2126*x* for audio processing

SHARC in Alacron FT-P5

- COTS (Commercial Off-The-Shelf) system for use in “distributed, compute intensive, high data rate applications” in commercial and military industries
- Supports 1 to 96 ADSP-2106*x* processors
- Makes extensive use of SHARC’s DMA through external PMC interface, supporting full-duplex communication in excess of 1 GB/sec
  - In-cabinet SAN clusters
  - Compute nodes in distributed systems

SHARC vs. RISC Processors

- RISC is…
  - Less costly to design, test, and manufacture, since processors are less specialized
- But…
  - Parallel (stereo) computation requires two or more interconnected processors accessing shared memory
  - Less performance

Conclusion

- SHARC offers great deal of computational power, with on-chip SRAM and SIMD architecture
- Variety of applications (especially audio processing) exploit it

Citations

- Processor details taken from product manuals and descriptions at http://www.analog.com