UIUC CS 433

Analog Devices Incorporated Blackfin

CS 433

Prof. Luddy Harrison

Processor Presentation Series

Note on this presentation series

- These slide presentations were prepared by students of CS433 at the University of Illinois at Urbana-Champaign
- All the drawings and figures in these slides were drawn by the students. Some drawings are based on figures in the manufacturer's documentation for the processor, but none are electronic copies of such drawings
- You are free to use these slides provided that you leave the credits and copyright notices intact

Introduction

- Blackfin combines 16-bit fixed point DSP features with general purpose microcontroller functions, eliminating the need for two processors in many applications.
- Design provides a balance between DSP performance, low power and low cost.
- First product to use the *Micro Signal Architecture* jointly developed by Analog Devices and Intel.

Introduction

- Targets embedded audio, video and communications applications.
- First generation BF535 released in 2001
- Second (current) generation has 5 members:
  - Single core BF531/BF532/BF533 from $4.95
  - Dual core BF561 from $19.95 (prices @10K units)
  - msp500 SoftFone for GSM/GPRS/EDGE
- Dual ALUs and 16x16 multipliers per core

Introduction

- 32-bit RISC Instruction Set w/3 issue VLIW
  - 64 bits: up one 32-bit and two 16-bit instructions
- 10-stage fully interlocked pipeline
- On-chip memory can be cache or data SRAM
- Integrates microcontroller style peripherals
- Wide range of speeds/voltages
  - From 100 MHz/0.7 V to 750 MHz/1.4 V
- Supervisor and User modes for protection

Instruction Set

- Algebraic assembly language
  - Ease of coding and readability
  - Optimized for C/C++ compilation
- Orthogonal instruction set
- Encoding assigns 16-bit op-codes to frequently use instructions. Complex DSP and arithmetic instructions are encoded into 32-bit op-codes.

Data Registers and Accumulators

32-bit Data Registers

	31 16	15 0
R0	R0.H	R0.L
R1	R1.H	R1.L
R2	R2.H	R2.L
R3	R3.H	R3.L
R4	R4.H	R4.L
R5	R5.H	R5.L
R6	R6.H	R6.L
R7	R7.H	R7.L

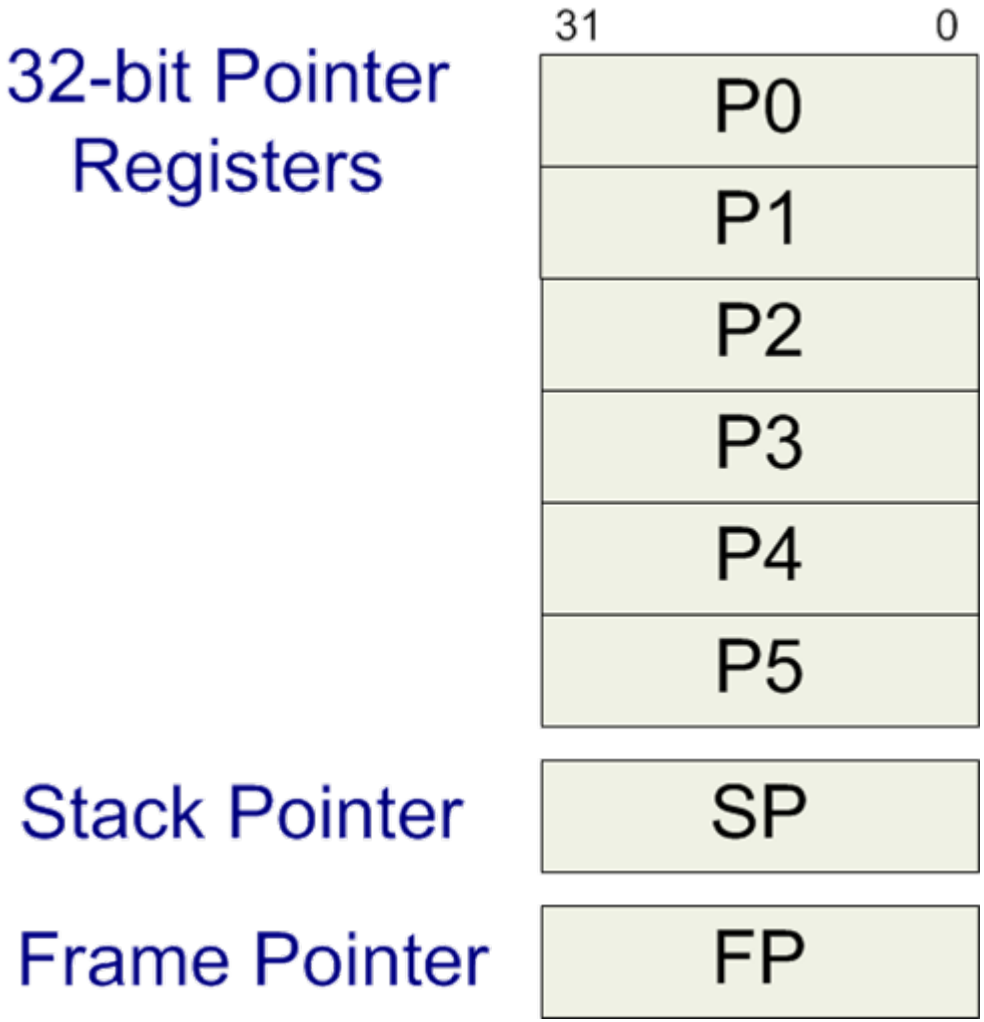
40-bit Accumulators

	39 32	31 0
A0	A0.X	A0.W
A1	A1.X	A1.W

Half-word access

- Data registers
  can be accessed
  by half words (high, low)
  - R0 = R1;
  - R3 = 0x7F11;
  - R8.H = R7.L;
  - R3.H = R2.L;

Pointer Registers



Instruction Set - Pointers

- Load/Store to Data or Pointer Reg

```
R3 = [P0++];
P3 = [P2];
[--P5] = R4;
```

- Push/Pop Single or Multiple Regs

```
[--SP] = R2;
[--SP] = (R7:4, P5:3);
(R7:4, P5:3) = [SP++];
R2 = [SP++];
```

Special Registers

Index Registers

31 0	31 0	31 0
I0	L0	B0
I1	L1	B1
I2	L2	B2
I3	L3	B3

Modify Registers

M0
M1
M2
M3

Instruction Set – Special Regs

- Index and Modify Special Purpose Registers
  - i Regs: Data structures
  - m Reg: Offset
  - Commonly used for circular buffers

```
i0 += m0;
i1 += m1 (brev);
```

Instruction Set - Addressing

- Register Indirect
```
R3 = [P3];
```
- Register indirect w/post increment/decrement
```
R3 = [P3++];
```
- Register indirect indexed with offset
```
R6 = [P2 + 12];
```
- Register indirect with pre-decrement (push)
```
[--SP] = R2;
```
- Register indirect with post-increment (pop)
```
R2 = [SP++];
```

Instruction Set - Logical

- & - Logical AND

```
R3 = R3 & R2;
```

- | - Logical OR
- ^ - Logical XOR (Exclusive OR
- ~ - Logical NOT (One's complement)

```
r5 = ~ r6;
```

- BXORSHIFT - Four Bit XOR w/Shift

```
r0.l = cc = bxorshift(a0, r1);
```

Instruction Set - Arithmetic

- Add

```
R3 = R1 + R2;
```

- Subtract

```
P2 = P1 – P0;
```

- Minimum / Maximum

```
r5 = min(r2, r3);
```

- Multiply

```
R3.L = R3.H * R2.H;
r6 = r3.h * r4.h; /* 32 bit result */
R3 *= R0; /* 32 bit multiply */
```

Instruction Set - Arithmetic

- Multiply/Accumulate (to Accumulator register)

```
A0 += R3.H * R2.H;
```

- Divide

```
DIVS(dividend_reg, divisor_reg);
DIVQ(dividend_reg, divisor_reg);
```

- As with many fixed point DSPs, many math operations support operation flags (covered below) to saturate or truncate, as well as to define fractional or integer operations.

Instruction Set – Op Flags

- Operation Flags

  (S) Saturate (NS) No saturation;
  (IS) Signed Integer (IU) Unsigned Integer
  (IH) Signed Integer, high word extract
  (ISS2) Signed integer with scaling
  (FU) Unsigned Fraction 0.16 * 0.16 → 0.32
  (TFU) Unsigned Fraction with truncation
  (T) Signed Frac. w/Trunc 1.15 * 1.15 → 1.31
  (S2RND) Signed fraction with scaling and rounding
  (M) Mixed mode multiply (BREV) Carry bit reverse

```
R0 = R5 - R3 (NS);
A1 = R0.H * R0.L (M);
```

Instruction Set – Flow Control

- JUMP – Load program counter with target address

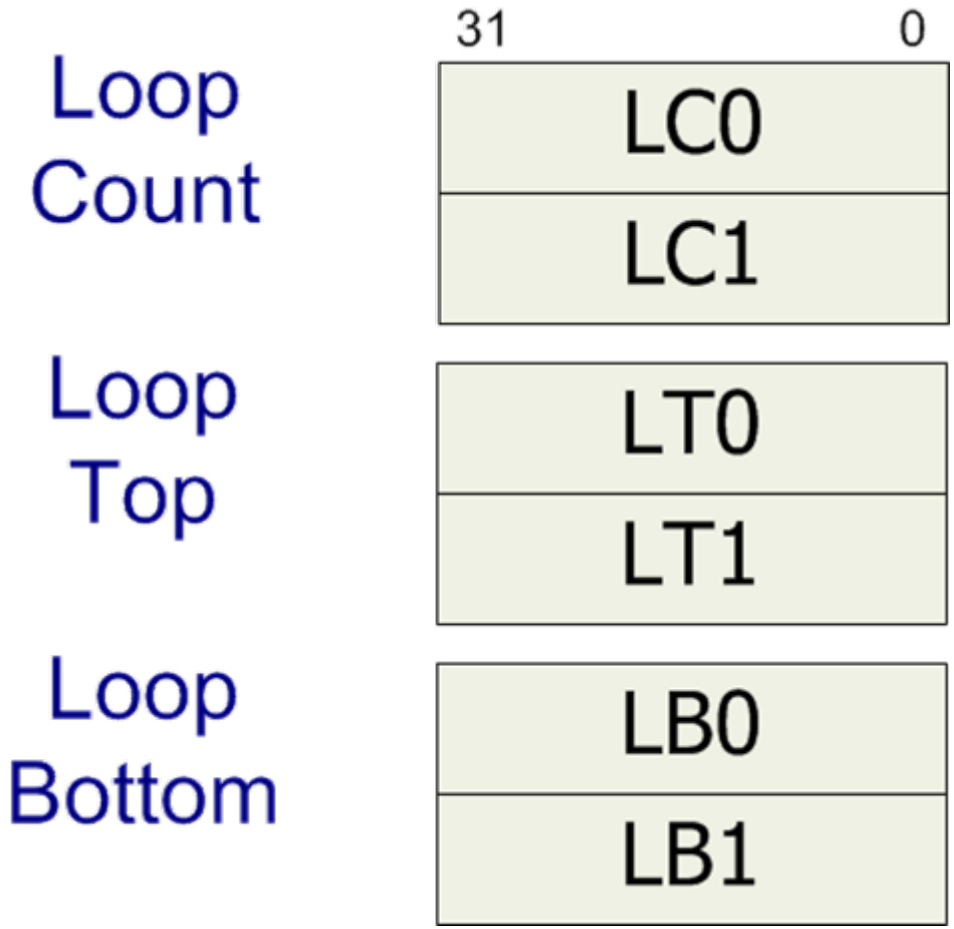
```
jump (p2); /* p2 contains target address */
jump (pc + p2); /* relative to program counter */
jump next_section; /* assembler resolves target*/
```

- CALL – Saves next instr. on RETS register and loads program counter with subroutine address

```
call (p5) ;
call 0x102030;
call next_subroutine;
```

- RTS – Returns from subroutine loading program counter with value of RETS register

Hardware Loop Registers



Instruction Set - Loops

```
LOOP loop_name loop_counter;
LOOP_BEGIN loop_name;
LOOP_END loop_name;

LSETUP (Begin_Loop, End_Loop)Loop_Counter;
```

Instruction Set - Conditionals

- Flags
  - CC – Control code
  - AZ – Zero
  - AC0 – ALU0 Carry
  - AV0 – A0 Overflow
  - V – Data Reg. Overflow
  - AN – Negative
  - AC1 – ALU1 Carry
  - AV1 – A1 Overflow
- Control code (CC) allows high code flexibility

```
cc = r6 <= r1;
if cc jump next_section;
```

Instruction Set – Shift/Rotate

- Logical shift

```
p0 = p1 << 2; /* left shift by 2 */;
r3.l = lshift r0.l by r2.l;
 /* sign of r2.l controls direction */
```

- Arithmetic shift

```
r4 = r2 >>> 20;
r4 = ashift r2 by r7.l;
```

- Shift with Add (Add with Shift also available)

```
r3 = (r1 + r2) << 2;
```

Instruction Set – Bit Ops

- Bit Set

```
bitset(r3, 6);
```

- Bit Clear

```
bitclr(r4, 0);
```

- Bit Toggle

```
bittgl(r2, 21);
```

- Bit Test – Result goes to Control Code (CC)

```
cc = ! bittst(r1, 18);
```

- Count One's

```
r0.l = ones r6;
```

Instruction Set – Pixel Ops

- Blackfin cores implement several operations designed to simplify 8-bit pixel manipulation.
- BYTEOP16P – Quad 8-bit Add

```
(r1,r2)= byteop16p(r3:2,r1:0);
```

- BYTEOP16M – Quad 8-bit Subtract

```
(r1,r2)= byteop16m(r3:2,r1:0);
```

- SAA - Quad 8-Bit Subtr-Absolute-Accumulate

```
saa(r1:0,r3:2);
```

- BYTEOP1P - Quad 8-Bit Average (Byte)

Instruction Set – Vector Ops

- Blackfin cores can perform simultaneous operations on multiple 16-bit values.
- Vector Add / Subtract

```
r6 = r4 +|+ r5;
r0 = r2 +|- r1;
```

- Vector Search

```
(r1,r0) = search r2(LE); /* Less Eq */
```

- Vector Compare/Select

```
r5 = vit_max(r3, r2)(asl);
```

Instruction Set – Parallel Issue

- Parallel Issue Syntax

32-bit ALU/MAC inst || *16-bit inst* || *16-bit inst*;

```
saa (r1:0, r3:2) || r0=[i0++] || r2=[i1++];
```

32-bit ALU/MAC inst || *16-bit inst*;

```
r6=(a0+=r3.h*r2.h)(fu) || i2-=m0 ;
```

32-bit NOP || *16-bit inst* || *16-bit inst*;

```
mnop || r1 = [i0++] || r3 = [i1++] ;
```

Instruction Set - Interrupts

- Interrupts use supervisory mode
- STI - Enable Interrupts
- CLI – Disable Interrupts
- SSYNC – System Synchronize
  - Forces all speculative and transient states to complete
- EXCPT - Force Exception
  - Used by applications for operating system calls

Instruction Set –Cache Control

- PREFETCH – Data cache prefetch

```
prefetch [p2];
```

- FLUSH – Data cache flush

```
flush [p2++];
```

- FLUSHINV – Data cache invalidate

```
flushinv [p2];
```

- IFLUSH – Instruction cache flush

```
iflush [p4];
```

Instruction Set - Encoding

- Frequently used control instructions are generally encoded in 16 bit.
- Arithmetic instructions are encoded in 16 or 32 bits depending on the arguments.
- Signal processing instructions are generally encoded as 32-bit values.
- Multi-issue 64-bit packets (VLIW) support up to one 32-bit and two 16-bit instructions.

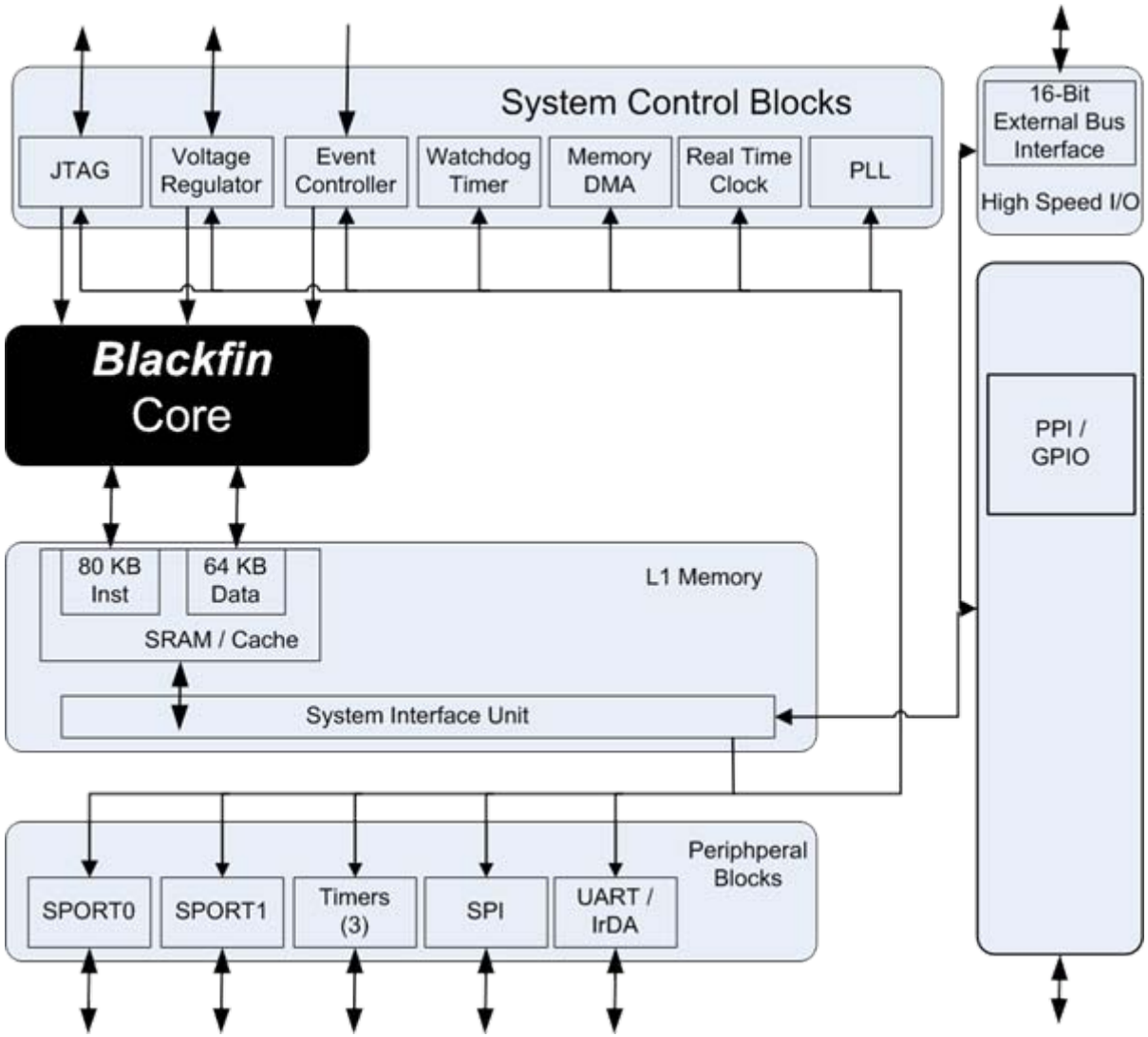
Instruction Set – Code Sample

```
// Portion of ADI Speech Audio code sample
__divfr_16 : // Signed fractional division of
             // 32bit numerator and 16b denominator
    P0 = 15;
    DIVS(R0,R1);

    LOOP .DIV_LP LC0 = P0;
    LOOP_BEGIN .DIV_LP;
    DIVQ(R0,R1);
    LOOP_END .DIV_LP;
    R0 = R0.L (X);
    RTS;

__divfr_16.end:
```

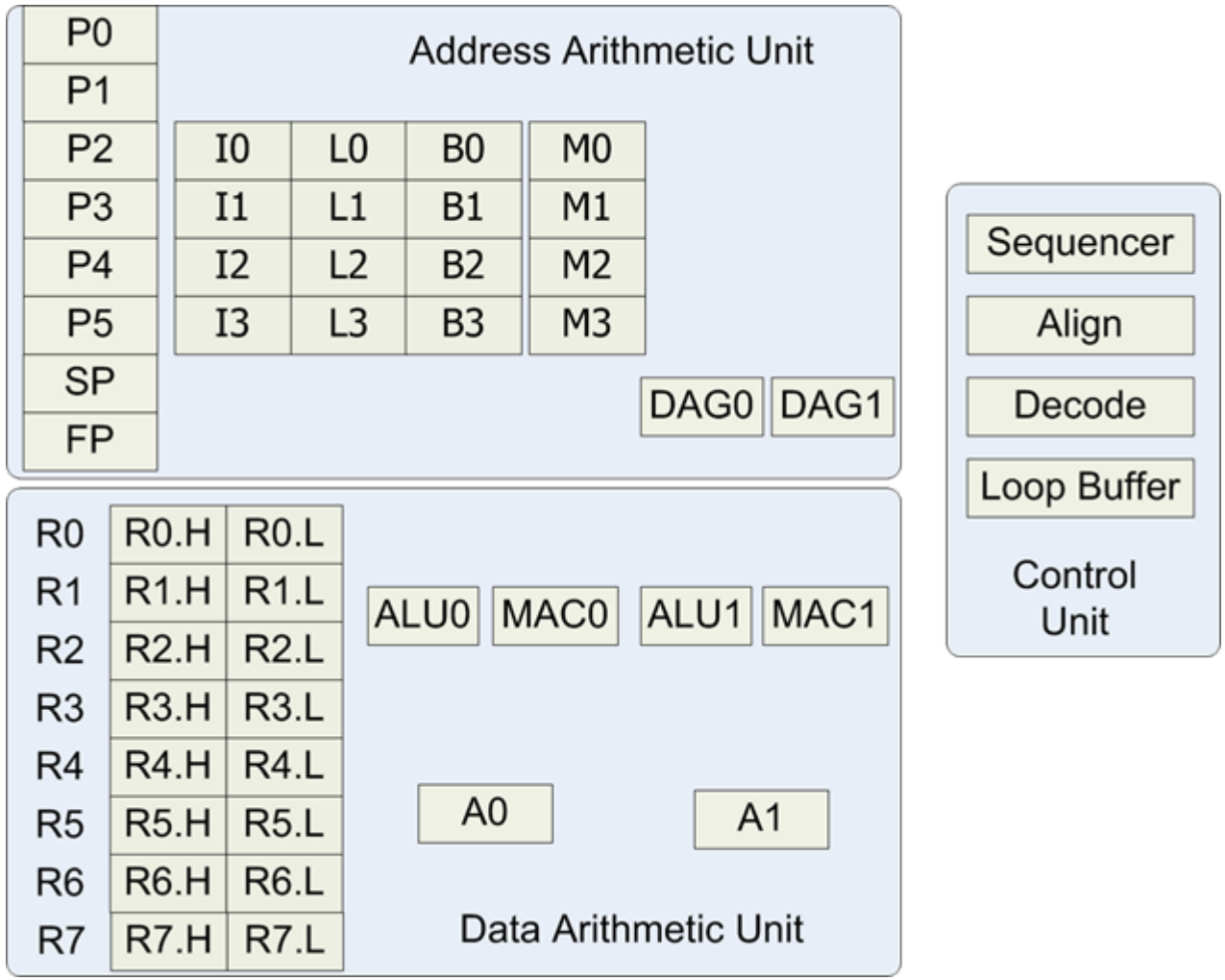
Instruction Set – Code Sample

```
__bpsk: // Portion of ADI QAM (Quadrature Amplitude Modulation) code sample
   L0 = 0;
   I0 = R1;              // Address of input bitstream in I0
   P1 = R0;              // counter
   P0 = R2;              // address of output bitstream in P0
   P2 = 16(Z);           // inner loop counter
   R3 = -1;              // Symbol for bit '0' in bitstream
   R0.L = W[I0++];

    LSETUP(OUT_LOOP_START,OUT_LOOP_END) LC0 = P1 >> 1;  // Load N/2
OUT_LOOP_START:
        LSETUP(IN_LOOP_START,IN_LOOP_END) LC1 = P2;
IN_LOOP_START:
            CC = BITTST(R0,0);  // Test the last bit
            R2 = 1(Z);
            IF !CC R2 = R3;     // If bit is zero output -1 else 1
IN_LOOP_END:
            R0 = R0 >> 1 || B[P0++] = R2;
OUT_LOOP_END:
        R0.L = W[I0++];
   RTS;
   NOP;
```

BF533 Block Diagram



Blackfin Core



Processor

- Architecture Overview
  - Load/Store architecture
  - Two fixed point data paths
    - Two 40-bit ALUs, two 16x16 multipliers (MACs)
    - Four 8-bit video ALUs and a barrel shifter
  - 32-bit RISC Instruction Set w/3 issue VLIW
  - SIMD (Single Instruction, Multiple Data)
    - Can use two ALUs or two MACs in parallel
    - Can’t use one ALU and one MAC in parallel

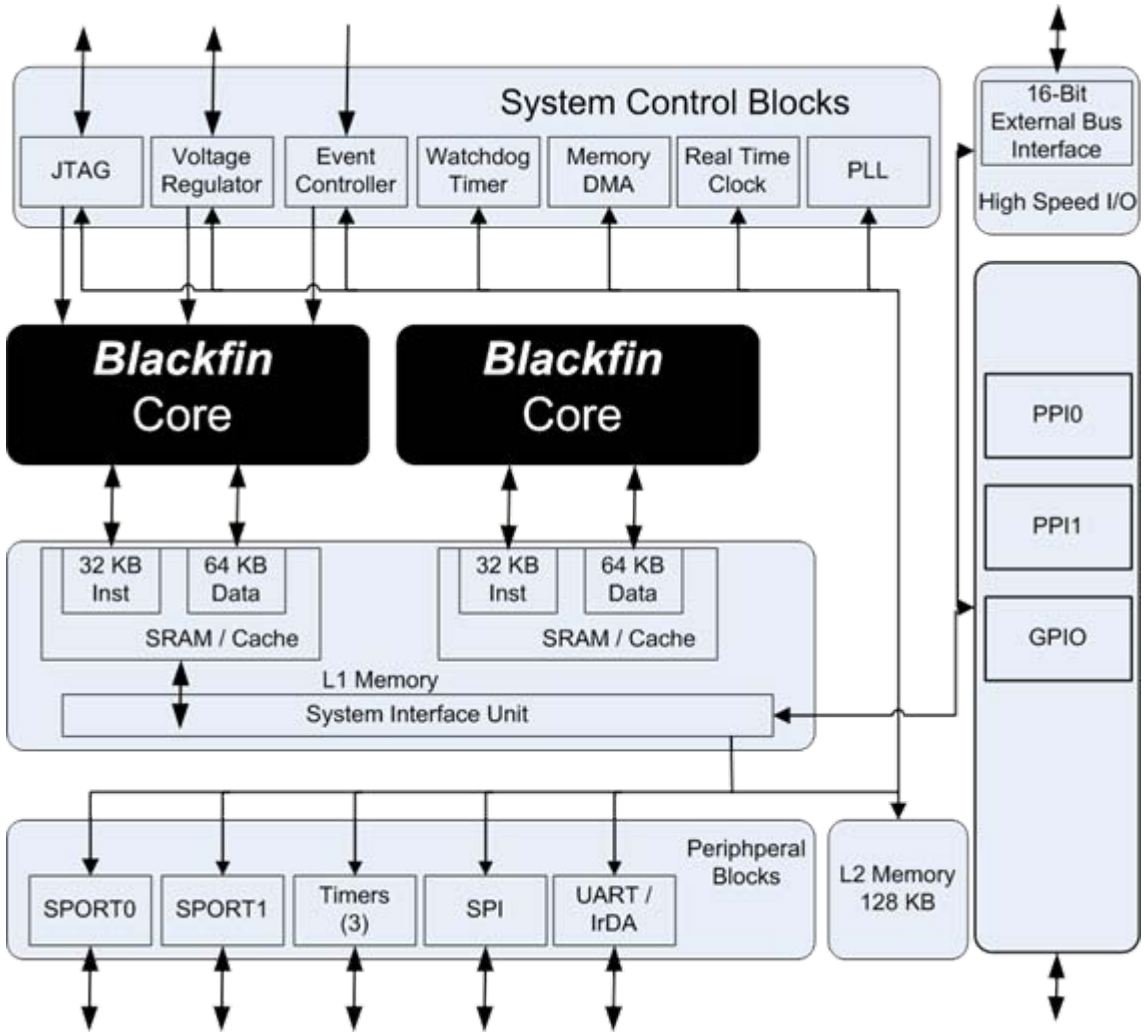
Processor – Bus Hierarchy

- Core Clock (CCLK) Domain
  - IDB - L1 Instruction (64-bit)
  - LD0 – Load Data 0 (32-bit)
  - LD1 – Load Data 1 (32-bit)
  - SD – Store Data (32-bit)

Processor – Bus Hierarchy

- System Clock (SCLK) Domain
  - DCB – DMA Core Bus
  - DAB – DMA Access Bus
  - DEB – DMA External Bus
  - PAB – Peripheral Access Bus
  - EAB – External Access Bus
  - EPB – External Port Bus (16-bit)

BF561 Dual Core Block Diagram



Processor – Operating Modes

Oper. Mode	Power Savings	PLL Status	PLL Bypass	CCLK	SCLK	DMA Allowed
Full On	None	Enabled	No	Enabled	Enabled	L1
Active	Medium	Enabled	Yes	Enabled	Enabled	L1
Sleep	High	Enabled	No	Disabled	Enabled	-
Deep Sleep	Maximum	Disabled	-	Disabled	Disabled	-

Processor – Mode Change

- To change frequency, write a new multiplier value to the appropriate system control register. After a stabilization time, the DSP can issue a command to reduce the core voltage. When the DSP receives the appropriate handshake from the power management chip, instructions can start executing at the new frequency and voltage.

```
CLI R2;         /* disable interrupts, copy IMASK to R2 */
R1.L = 0x3F00; /* set BYPASS bit */
W[P0] = R1;     /* and write to PLL_CTL */
IDLE; /* drain pipeline, enter idle, wait for PLL wakeup*/
STI R2;      /* after PLL wakeup, restore intr and IMASK */
...          /* processor is now in Active mode */
```

Processor Pipeline

Pipeline Stage	Description
IF1 – Instruction Fetch 1	Starts instruction memory access
IF2 – Instruction Fetch 2	Intermediate memory pipeline
IF3 – Instruction Fetch 3	Finishes L1 instruction memory access
DEC – Instruction Decode	Aligns instruction, starts instruction decode, and accesses Pointer register file
AC – Address Calculation	Calculates data addresses and branch target address
EX1 – Execute 1	Starts access of data memory
EX2 – Execute 2	Reads register file
EX3 – Execute 3	Finishes access of data memory and starts execution of dual cycle instructions
EX4 – Execute 4	Executes single-cycle instructions
WB – Write Back	Writes states to Data and Pointer Register Files and processes events

Processor Pipeline

- 10-stage fully interlocked pipeline
  - All data hazards hidden from the programmer
  - Stalls when necessary
- Static branch prediction on conditional branches

IF1	IF2	IF3	DEC	AC	EX1	EX2	EX3	EX4	WB	
	IF1	IF2	IF3	DEC	AC	EX1	EX2	EX3	EX4	WB

Memory

- Flat 32-bit (4 GB) address range
- Supports access of 8, 16 and 32 bit data
- External Bus Interface Unit (EBIU) interfaces to external memories without additional controllers.
  - PC100 and PC133 Synchronous DRAM (SDRAM)
  - Asynchronous SRAM, ROM, FIFOs, flash memory

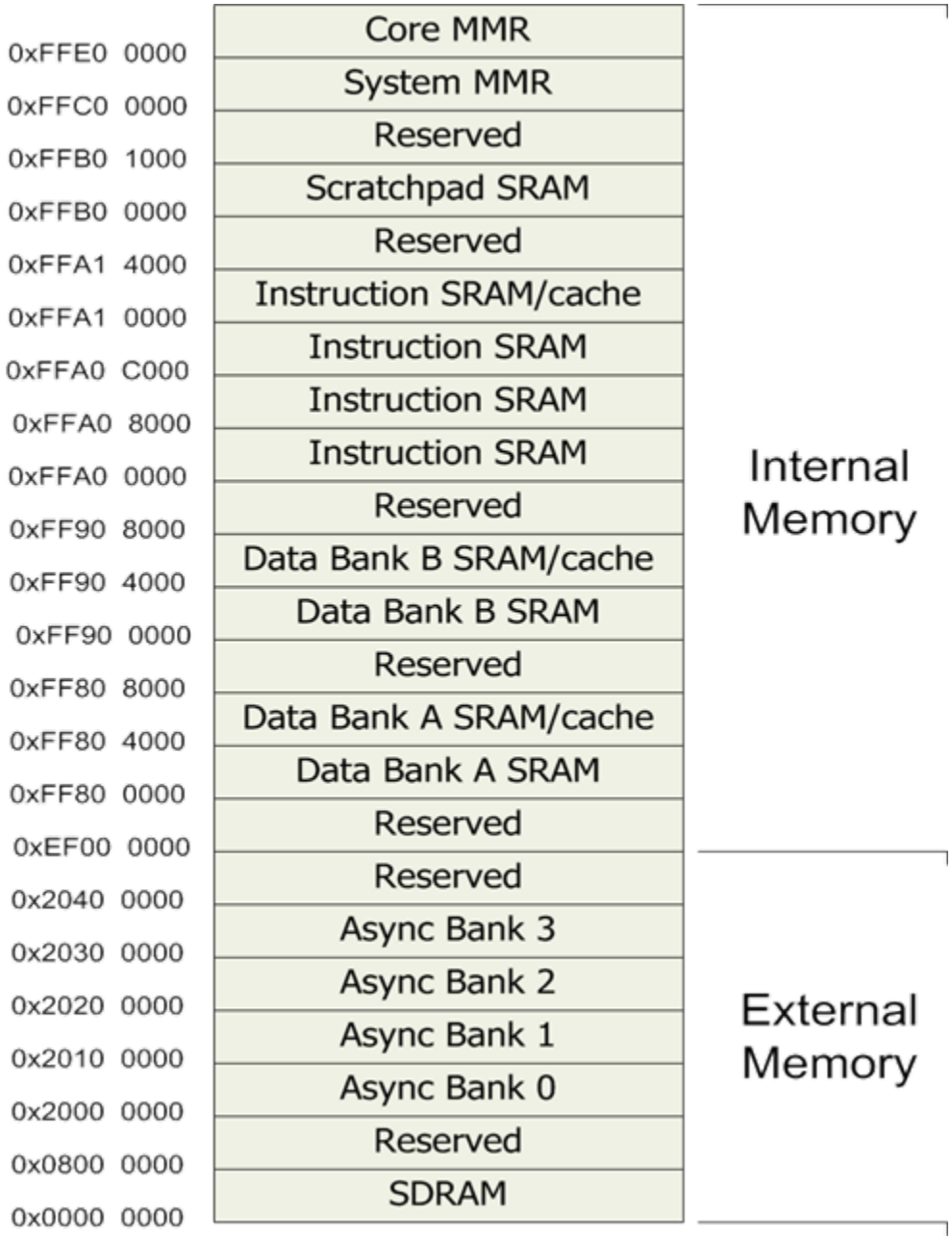
Memory

- Hierarchical memory model
  - Level 1 (L1) – on chip, faster
    - Separate instruction / data memories
    - 4-way set associative instruction cache
    - 64-bit data cache with fill/victim access
  - Level 2 (L2) - off-chip, longer latencies
  - External – off-chip, slower
  - Memory-mapped registers (MMRs)
  - Boot Read-Only Memory (ROM)

Memory

Memory Type	BF531	BF532	BF533
Instruction SRAM or Cache	16 KB	16 KB	16 KB
Instruction SRAM	16 KB	32 KB	64 KB
Data SRAM or Cache	16 KB	32 KB	32 KB
Data SRAM	-	-	32 KB
Data Scratchpad SRAM	4 KB	4 KB	4 KB
Total	84 KB	116 KB	148 KB

BF533 Memory Map



Peripherals

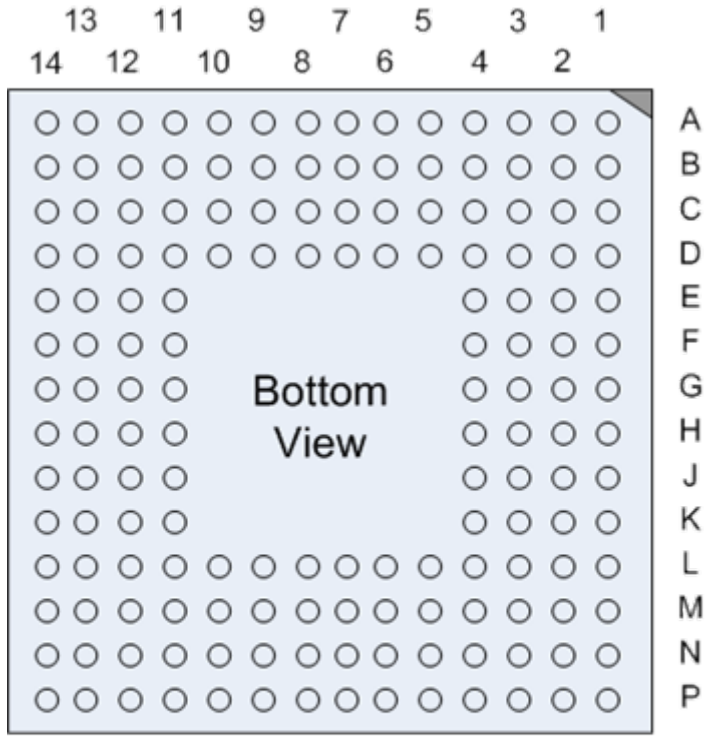
- Parallel Peripheral Interface (PPI)
  - Half-duplex, bidirectional port accommodating up to 16 bits of data.
- Timers
  - Three general-purpose timers, one core timer and one watchdog timer.
- Real-Time Clock (RTC)
  - Digital watch features: time of day, alarm, and stopwatch countdown

Peripherals

- Synchronous serial ports (SPORTs)
  - Synchronous serial data transfer
- Universal Asynchronous Receiver/ Transmitter (UART)
  - Supports the half-duplex IrDA® (Infrared Data Association) protocol
- External Bus Interface Unit (EBIU)
  - Glueless interfaces to external memories

Physical Packaging

- Single core processors
  - 160-lead MiniBGA
  - 176-lead LQFP
  - 169-lead PBGA
- Dual core processors
  - 256-lead MiniBGA
  - 297-lead PBGA



160-Lead Chip Scale Ball Grid Array (CSPBGA)

12 mm x 12 mm (144 mm^2)

Physical Packaging - Pinouts

Pin Name	I/O	Function	Pin Name	I/O	Function
Memory Interface			*Synchronous Memory Control*		
ADDR19-1	O	Address Bus	/SRAS	O	Row Address Strobe
DATA15-0	I/O	Data Bus	/SCAS	O	Column Address Strobe
/ABE1-0	O	Bytes enables	/SWE	O	Write Enable
/BR	I	Bus Request	SCKE	O	Clock Enable
/BG	O	Bus Grant	CLKOUT	O	Clock Output
/BGH	O	Bus Grant Hang	SA10	O	A10 Pin
Asynchronous Memory Control			/SMS	O	Bank Select
/AMS3-0	O	Bank Select	*Timers*		
ARDY	I	Hardware Ready Control	TMR0	I/O	Timer 0
/AOE	O	Output Enable	TMR1	I/O	Timer 1/PPI FrameSync 1
/ARE	O	Read Enable	TMR2	I/O	Timer 2/PPI FrameSync 2
/AWE	O	Write Enable	NOTE: / on pin name (NEG) implies active LOW.		

Single Core Blackfin Processors BF531 / BF532 / BF533

Software Tools

- ADI VisualDSP++
  - Integrated Development and Debugging Environment (IDDE)
- Green Hills MULTI® IDE
- GNU compiler (gcc) and debugger (gdb)
- uClinux for Blackfin
  - Microcontroller port of Linux 2.6 kernel
  - Blackfin Stamp PCB – hardware reference design
  - Boots from flash memory

Applications

- AeroStream Software Defined Radio (SDR)
  - Low cost, low power digital radio. High audio quality. Plans to add video and MPEG.
- Roku SoundBridge
  - Low cost, low power network music player. Streams digital music from a computer to a stereo system. Supports Ethernet or WiFi connections.
- SpotCell Adaptive PCS/Cellular Repeater
  - Improves cell phone coverage in buildings, and warehouses with concrete and metal construction.

Applications

- AuthenTec FingerLoc
  - Embedded fingerprint matching system used in biometric systems.
- AValon RF Miniature Wireless System
  - Carries one-way video, alarm, and telemetry, two-way audio and data, and cover frequencies from 56 MHz to 2.5 GHz. Used by military, government, private industry, and consumers.

Applications

- ADI Car Telematics Platform
  - Single-chip, software programmable platform based on the Blackfin DSP that can function as navigation, communication and entertainment system. Combines global positioning satellite (GPS) tracking and wireless-based services with voice activated control. Includes Java™ support.
- ADI Blackfin BRAVO Broadband Videophone
  - Chipset based on the Blackfin DSP delivers audio, streaming video, networked videophone capabilities with MPEG4/MPEG2/DivX/WMV9 30 fps full color, full motion video in CIF resolution over broadband networks.

Roadmap for Future Releases

- Third Generation (2005-2006) will offer increased speed, power efficiency and lower costs. New peripherals will include integrated USB 2.0, 10/100 Ethernet and CAN.
  - BF566 – Dual core, Ethernet, 4xPPI, 6xSerial
  - BF563 – Dual core, USB 2.0, 2xPPI, 4xSerial
  - BF539 – Single core, 10xSerial, CAN
  - BF536 – Single core, Ethernet, 5xSerial, CAN
  - BF534 – Single core, 5xSerial, CAN, 48xGPIO

Resources

- Analog Devices Blackfin home page

  http://analog.com/processors/processors/blackfin/
- Blackfin user forum

  http://blackfin.org/
- Microcontroller Linux (uClinux) for Blackfin

  http://blackfin.uclinux.org/