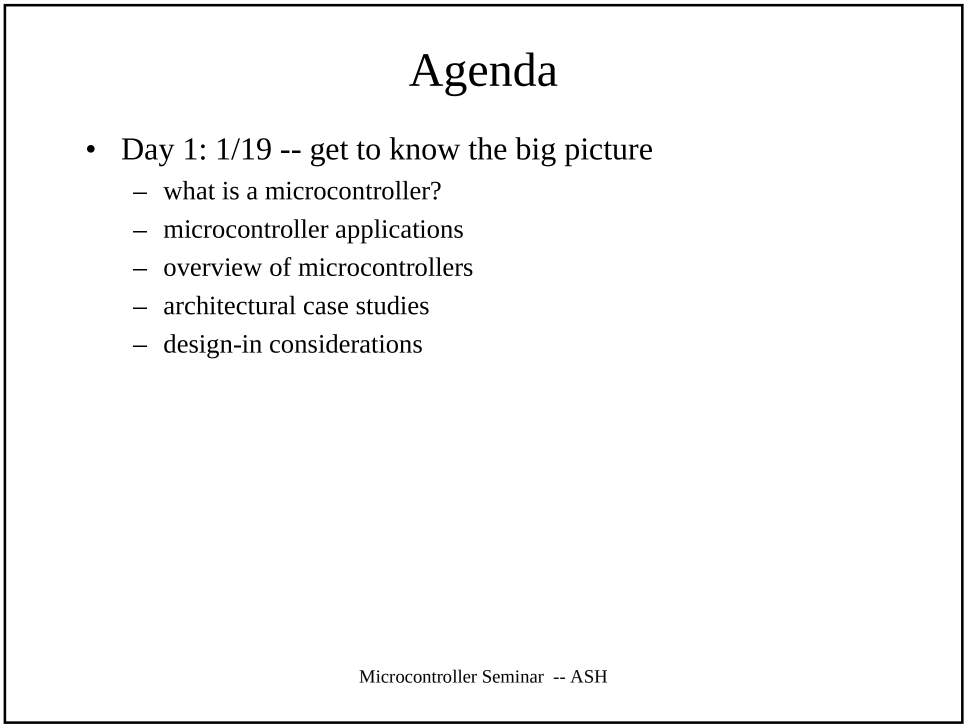
Advanced Embedded Microcontroller Seminar

Andrew Huang
bunnie@mit.edu

IAP 1999

References

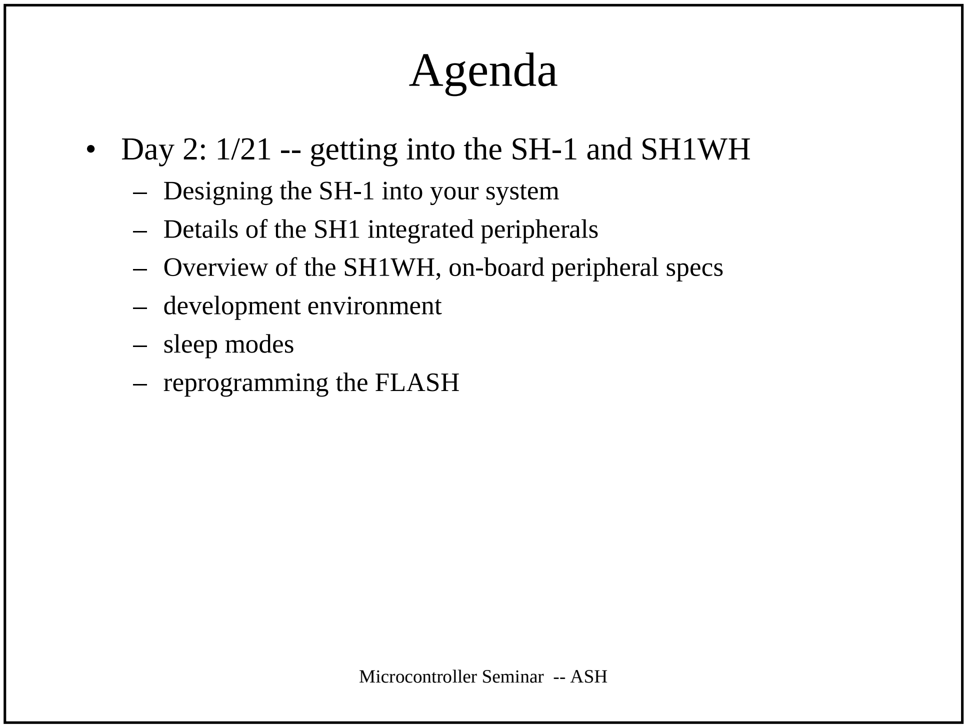
- Portions of these lecture slides come from
  - MIPS technical documentation
  - Hitachi tech docs
  - Microchip tech docs
  - ARM tech docs



Agenda

- Day 1: 1/19 -- get to know the big picture
  - – what is a microcontroller?
  - – microcontroller applications
  - – overview of microcontrollers
  - – architectural case studies
  - – design-in considerations

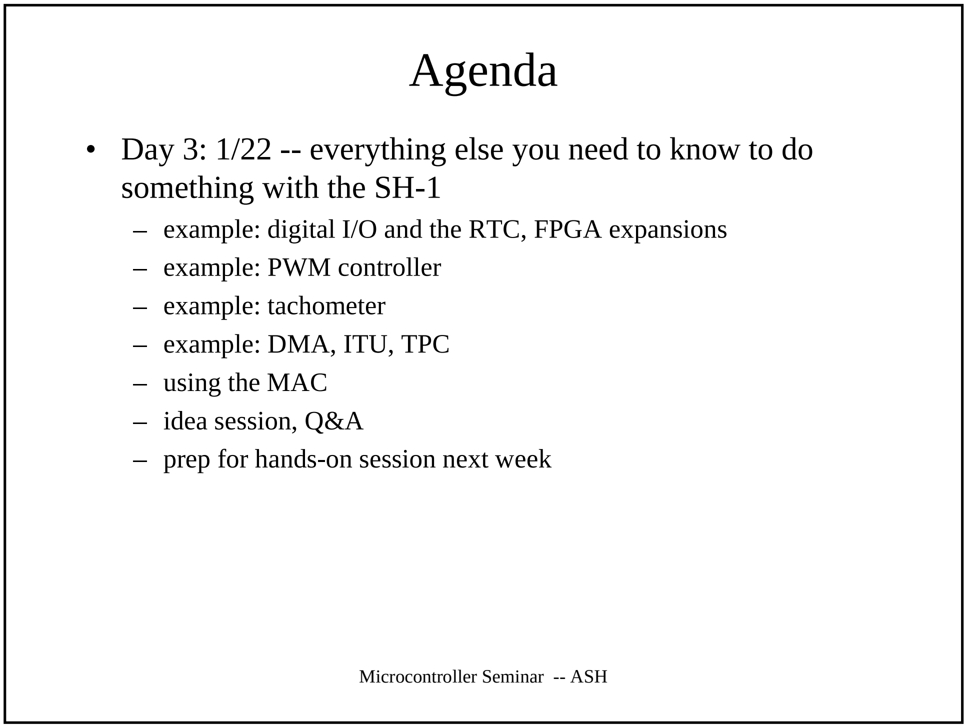
Microcontroller Seminar -- ASH



Agenda

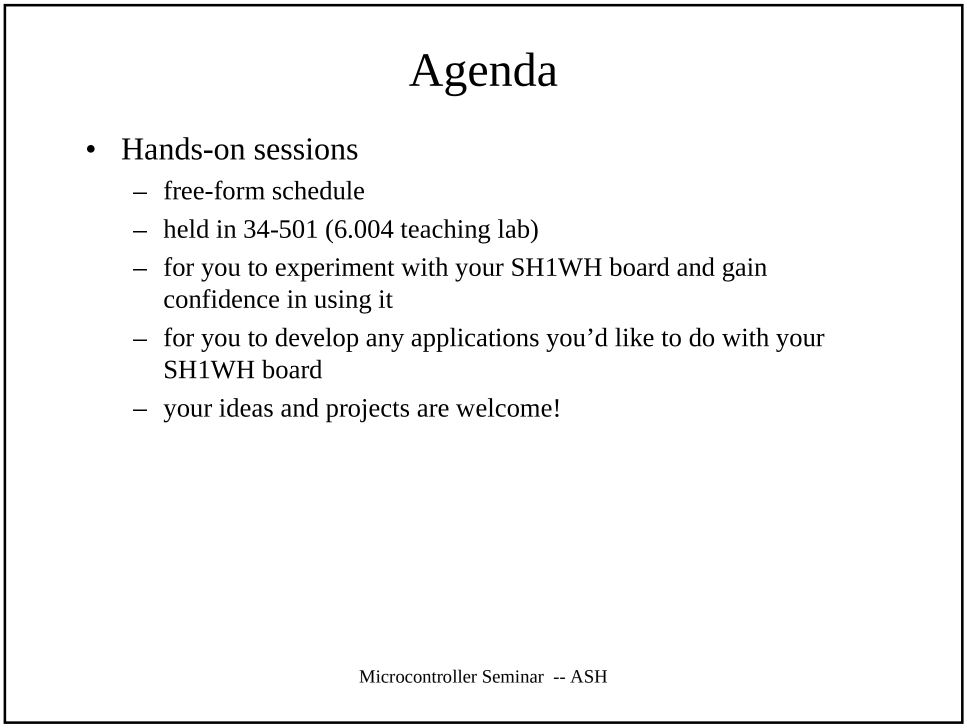
- Day 2: 1/21 -- getting into the SH-1 and SH1WH
  - – Designing the SH-1 into your system
  - – Details of the SH1 integrated peripherals
  - – Overview of the SH1WH, on-board peripheral specs
  - – development environment
  - – sleep modes
  - – reprogramming the FLASH

Microcontroller Seminar -- ASH



Agenda

- Day 3: 1/22 -- everything else you need to know to do something with the SH-1
  - example: digital I/O and the RTC, FPGA expansions
  - example: PWM controller
  - example: tachometer
  - example: DMA, ITU, TPC
  - using the MAC
  - idea session, Q&A
  - prep for hands-on session next week



Agenda

- Hands-on sessions
  - free-form schedule
  - held in 34-501 (6.004 teaching lab)
  - for you to experiment with your SH1WH board and gain confidence in using it
  - for you to develop any applications you'd like to do with your SH1WH board
  - your ideas and projects are welcome!

Using the Right Tool

- A Leatherman™ is great, but you wouldn’t want to cut down a tree or build a house with it
- On the same token, the PIC is a great processor, but its not the right tool for every job

Alphabet Soup

CISC

CPU

8085

SHARC

MIPS

VLIW

PowerPC

MCU

486

Dragonball

PIC

RISC

DSP

MPU

i960

ARM

SH-RISC

6811

Microcontrollers All

- Microcontroller
  - an application-specific processor
  - PIC vs. StrongArm: it all depends on your application
- Typically, microcontrollers...
  - are low cost, lightweight processors
  - require few support components for better system integration
  - come with integrated peripherals
- Let’s take a look at what other engineers use...

Some numbers...

High-end microcontroller applications:

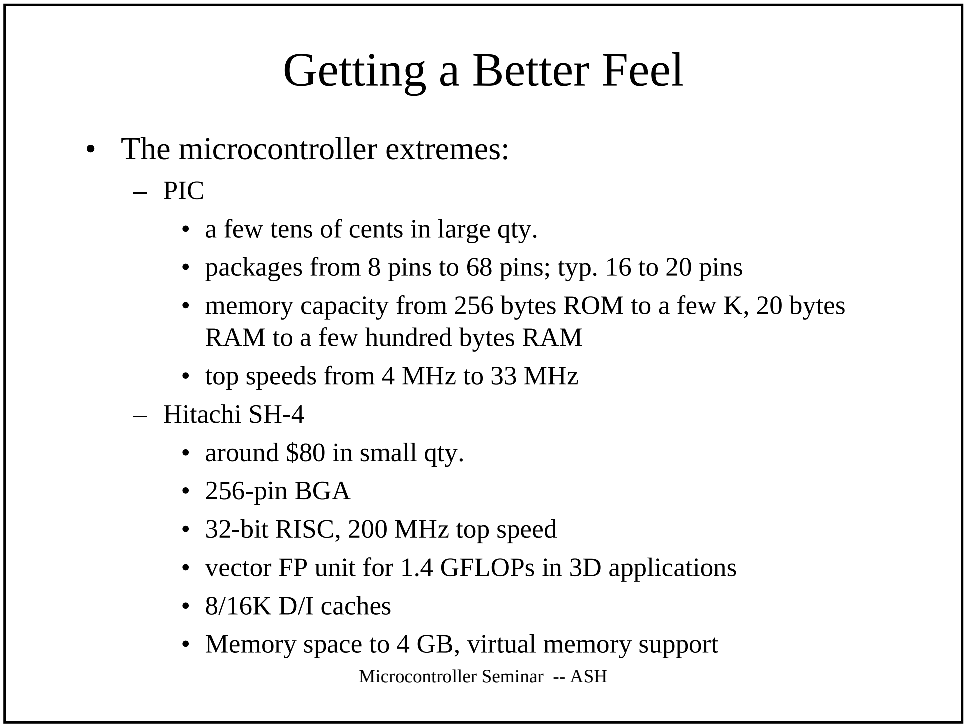
Product	Description	Processor	Speed	Memory
Sega Dreamcast	video game	SH-4, ARM for sound processing	200 MHz	16M main 8M vid 2M sound
Nintendo 64	video game	MIPS VR4300i	100 MHz	4M RAMBUS
Sony Playstation	video game	3000A MIPS core	33 MHz	2MB + 1MB vid, .5 MB snd
empeg	car stereo, MP3	SA	220 MHz	28 Gb HD storage
e-book	electronic book	ARM		can't read japanese
pocket station	PDA/playstation	ARM7T		128K RAM
QCP800	cell phone	80188 core	ND	not discloseable
HP4000 laserjet	printer	NEC MIPS 4300	100 MHz	4 MB RAM standard
LaserJet 6P	printer	i960JF	24 MHz	2 MB RAM standard
windows CE PDA	PDA	MIPS 3K and 4K series		2-8 MB RAM
LaserJet 5MP	printer	i960JF	35 MHz	4 MB RAM standard
E-11 Cassiopia	PDA	VR4111	70 MHz(?)	8 MB , windows CE
Cisco 7202	gateway	MIPS 4700	150 MHz	1MB SRAM/32MB DRAM
Palm Pilot III	PDA	Motorola Dragonball MC68328	16 MHz	2MB RAM
digital camera	casio	SH-2@20 MHz	20 MHz	

More numbers...

- 6811 and the 8085 series microcontrollers are popular in automotive industry
- PICs are popping up everywhere you require low cost and low performance, i.e. inside batteries, mice, toys, RFID, secure remotes
- 80188-class controllers are in hard drives
- Most japanese equipment use proprietary or “obscure” microcontrollers
  - NEC has a huge line of 4, 8, 16 and 32 bit microcontrollers that are great in consumer applications, but little known in the states
    - 16-bit K4
    - 32-bit V-800
    - 4-bit 75XL
    - 8-bit KO/S

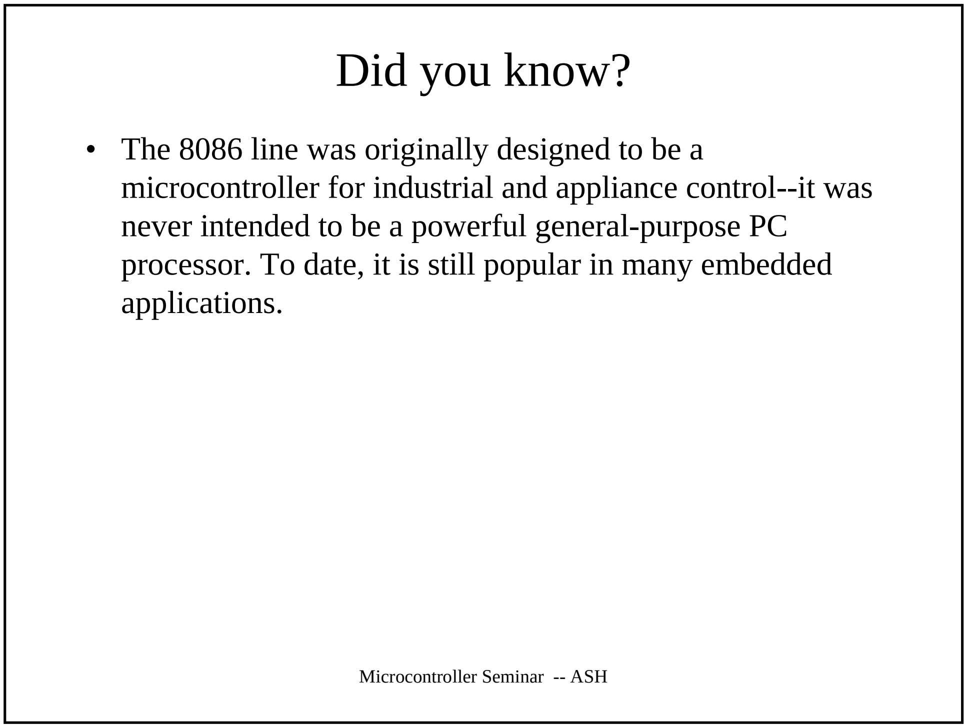
Where do you find a microcontroller?

- Cars
  - engine ignition, emissions control, ABS brakes, security
- Entertainment systems, consumer multimedia
  - front-panel displays, system functions
  - video game decks
- Appliances
  - timers, user interfaces, regulation of appliance functions
- Printers
  - image rasterizers, printer language interpretation
- Networking equipment
  - Routers, gateways
- Personal communications equipment
  - cell phones, cordless phones, smart cards
- Computers
  - IDE/SCSI hard drives, modems, high-end storage (I2O), keyboard
- Batteries
  - controls charging, discharge, gas guage



Getting a Better Feel

- The microcontroller extremes:
  - PIC
    - a few tens of cents in large qty.
    - packages from 8 pins to 68 pins; typ. 16 to 20 pins
    - memory capacity from 256 bytes ROM to a few K, 20 bytes RAM to a few hundred bytes RAM
    - top speeds from 4 MHz to 33 MHz
  - Hitachi SH-4
    - around $80 in small qty.
    - 256-pin BGA
    - 32-bit RISC, 200 MHz top speed
    - vector FP unit for 1.4 GFLOPs in 3D applications
    - 8/16K D/I caches
    - Memory space to 4 GB, virtual memory support



Did you know?

- The 8086 line was originally designed to be a microcontroller for industrial and appliance control--it was never intended to be a powerful general-purpose PC processor. To date, it is still popular in many embedded applications.

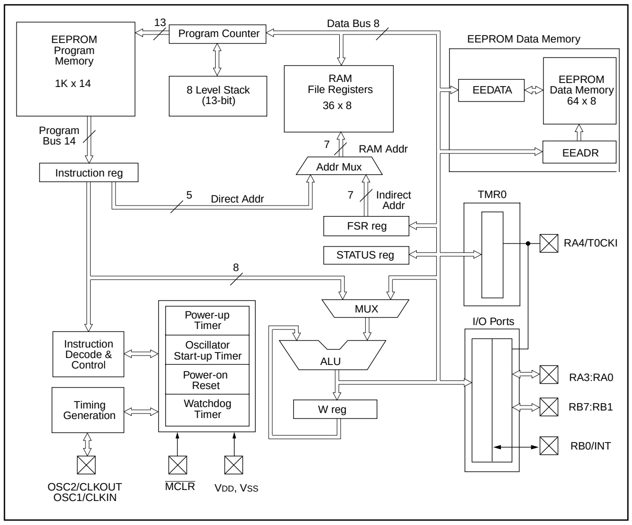
Architectural Case Studies

- PIC
  - low cost, low performance “penny micro”
- ARM
  - ARM is fabless, licenses IP and cores
- MIPS
  - ubiquitous, venerable line spanning several generations
- Intel i960
  - found in many computer peripherals, incl. printers and I2O (intelligent I/O) controllers
- Motorola 6811
  - legacy architecture, still popular in many low-end sectors
- SH-series
  - scalable family of microcontrollers

PIC

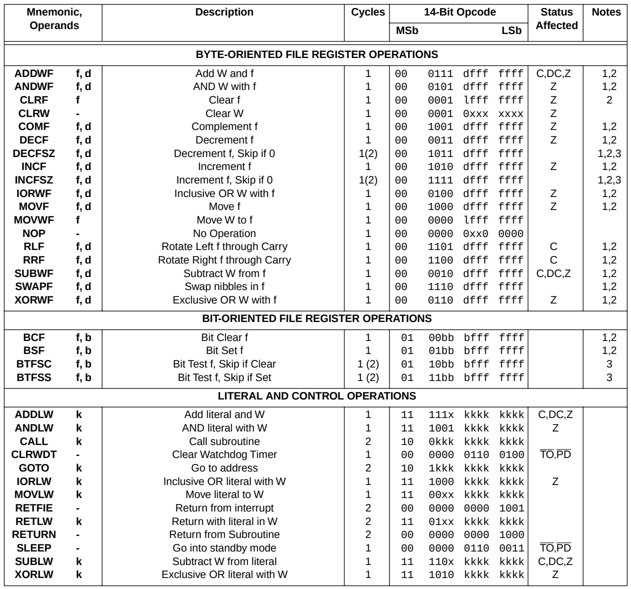
- Large family of CPUs
  - 8, 12, 14 and 16 bit variants
  - 4 MHz to 33 MHz
  - variety of embedded peripherals and low-cost package options
- Typical PIC: 16C84
  - 1Kx14 EEPROM, 36x8 RAM, 64x8 EEPROM data memory
  - 35 RISC-style instructions, 4-stage pipeline
  - semi-accumulator style ALU (“W” register, no register file per se)
  - 8-bit harvard (separate I/D mems) architecture
  - integrated watchdog timer and timer unit
  - 10 mW Pdiss
  - $7 single qty through digi-key

PIC16C84 block diagram



Microcontroller Seminar -- ASH

PIC 16CXX ISA



Mnemonic, Operands		Description	Cycles	14-Bit Opcode				Status Affected	Notes
				MSb			LSb		
BYTE-ORIENTED FILE REGISTER OPERATIONS									
ADDWF	**f, d**	Add W and f	1	00	0111	dfff	ffff	C,DC,Z	1,2
ANDWF	**f, d**	AND W with f	1	00	0101	dfff	ffff	Z	1,2
CLRF	**f**	Clear f	1	00	0001	lfff	ffff	Z	2
CLRW	**-**	Clear W	1	00	0001	0xxx	xxxx	Z	
COMF	**f, d**	Complement f	1	00	1001	dfff	ffff	Z	1,2
DECF	**f, d**	Decrement f	1	00	0011	dfff	ffff	Z	1,2
DECFSZ	**f, d**	Decrement f, Skip if 0	1(2)	00	1011	dfff	ffff		1,2,3
INCF	**f, d**	Increment f	1	00	1010	dfff	ffff	Z	1,2
INCFSZ	**f, d**	Increment f, Skip if 0	1(2)	00	1111	dfff	ffff		1,2,3
IORWF	**f, d**	Inclusive OR W with f	1	00	0100	dfff	ffff	Z	1,2
MOVF	**f, d**	Move f	1	00	1000	dfff	ffff	Z	1,2
MOVWF	**f**	Move W to f	1	00	0000	lfff	ffff		
NOP	**-**	No Operation	1	00	0000	0xx0	0000		
RLF	**f, d**	Rotate Left f through Carry	1	00	1101	dfff	ffff	C	1,2
RRF	**f, d**	Rotate Right f through Carry	1	00	1100	dfff	ffff	C	1,2
SUBWF	**f, d**	Subtract W from f	1	00	0010	dfff	ffff	C,DC,Z	1,2
SWAPF	**f, d**	Swap nibbles in f	1	00	1110	dfff	ffff		1,2
XORWF	**f, d**	Exclusive OR W with f	1	00	0110	dfff	ffff	Z	1,2
BIT-ORIENTED FILE REGISTER OPERATIONS									
BCF	**f, b**	Bit Clear f	1	01	00bb	bfff	ffff		1,2
BSF	**f, b**	Bit Set f	1	01	01bb	bfff	ffff		1,2
BTFSC	**f, b**	Bit Test f, Skip if Clear	1 (2)	01	10bb	bfff	ffff		3
BTFSS	**f, b**	Bit Test f, Skip if Set	1 (2)	01	11bb	bfff	ffff		3
LITERAL AND CONTROL OPERATIONS									
ADDLW	**k**	Add literal and W	1	11	111x	kkkk	kkkk	C,DC,Z	
ANDLW	**k**	AND literal with W	1	11	1001	kkkk	kkkk	Z	
CALL	**k**	Call subroutine	2	10	0kkk	kkkk	kkkk		
CLRWDT	**-**	Clear Watchdog Timer	1	00	0000	0110	0100	$\overline{TO}$,$\overline{PD}$	
GOTO	**k**	Go to address	2	10	1kkk	kkkk	kkkk		
IORLW	**k**	Inclusive OR literal with W	1	11	1000	kkkk	kkkk	Z	
MOVLW	**k**	Move literal to W	1	11	00xx	kkkk	kkkk		
RETFIE	**-**	Return from interrupt	2	00	0000	0000	1001		
RETLW	**k**	Return with literal in W	2	11	01xx	kkkk	kkkk		
RETURN	**-**	Return from Subroutine	2	00	0000	0000	1000		
SLEEP	**-**	Go into standby mode	1	00	0000	0110	0011	$\overline{TO}$,$\overline{PD}$	
SUBLW	**k**	Subtract W from literal	1	11	110x	kkkk	kkkk	C,DC,Z	
XORLW	**k**	Exclusive OR literal with W	1	11	1010	kkkk	kkkk	Z	

Microcontroller Seminar -- ASH

PIC architecture

- Very cost-oriented architecture
  - sacrifices scalability, performance, and expandability
  - excellent for simple control applications
- Tiny instruction set can make programming difficult
  - no hardware mult/div
  - small memory => no complex structures
  - assembly coding is more tedious
- Where might you use a PIC?
  - controller for appliances
  - controller for LCDs, other devices
- Incidentally:
  - for slightly more than the cost of a PIC16C84, you can buy a 5K-gate, 360 flip-flop FPGA that has performance to 100+ MHz and has 74 I/Os

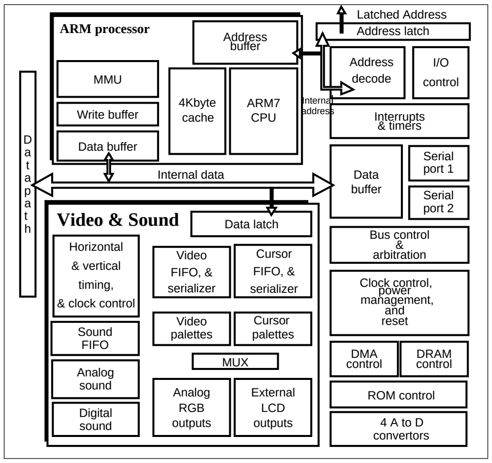
ARM

- ARM Holdings, plc. is a fabless company which defines and licenses the ARM architecture
  - joint venture of Apple, Acorn, and VLSI
- ARM is targetted at:
  - Portable market: digital cellular phones, pagers and personal organizers
  - Embedded market: modems, hard disc drives, printers and automotive applications
  - Consumer multimedia market: sound systems, games, internet access TV, set top box
- ARM tries to provide high MIPS/watt, good code density, and minimal area => minimal cost

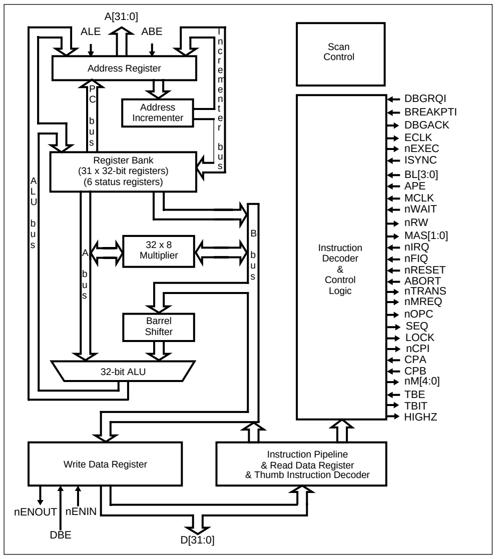
ARM

- 32-bit architecture, typ. harvard-style
- 31 general-purpose registers, 6 special-purpose regs
- Popular arch. variants include ARM7, ARM9, ARM10, and StrongARM
- Most ARM implementations include 8-16K separate I/D caches
- Representative ARM: ARM7500 implementation
  - 33 MHz processor, 4K cache
  - Integrated I/O: DMA, 2 serial, 4 A/D, 8 stereo sound channels, video controller, interrupt controller
  - multiply and barrel shift instructions in hardware
  - processor core consumes roughly 100 mW at top speed; peripherals additional

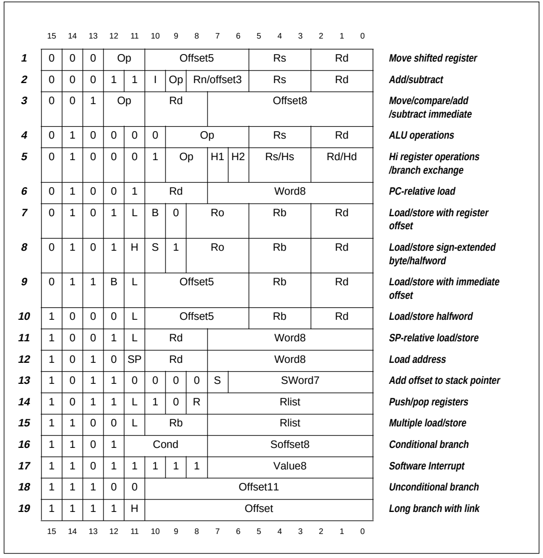
ARM7500 block diagram



ARM 7-series core diagram

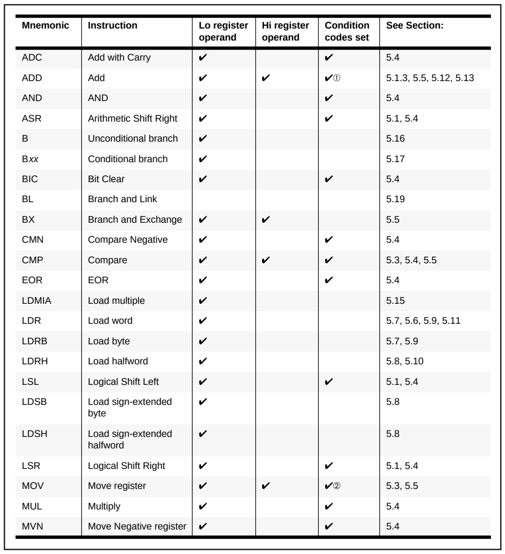


ARM7 ISA



	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
1	0	0	0	Op		Offset5					Rs			Rd			*Move shifted register*
2	0	0	0	1	1	I	Op	Rn/offset3			Rs			Rd			*Add/subtract*
3	0	0	1	Op		Rd			Offset8								*Move/compare/add /subtract immediate*
4	0	1	0	0	0	0	Op				Rs			Rd			*ALU operations*
5	0	1	0	0	0	1	Op		H1	H2	Rs/Hs			Rd/Hd			*Hi register operations /branch exchange*
6	0	1	0	0	1	Rd			Word8								*PC-relative load*
7	0	1	0	1	L	B	0	Ro			Rb			Rd			*Load/store with register offset*
8	0	1	0	1	H	S	1	Ro			Rb			Rd			*Load/store sign-extended byte/halfword*
9	0	1	1	B	L	Offset5					Rb			Rd			*Load/store with immediate offset*
10	1	0	0	0	L	Offset5					Rb			Rd			*Load/store halfword*
11	1	0	0	1	L	Rd			Word8								*SP-relative load/store*
12	1	0	1	0	SP	Rd			Word8								*Load address*
13	1	0	1	1	0	0	0	0	S	SWord7							*Add offset to stack pointer*
14	1	0	1	1	L	1	0	R	Rlist								*Push/pop registers*
15	1	1	0	0	L	Rb			Rlist								*Multiple load/store*
16	1	1	0	1	Cond				Soffset8								*Conditional branch*
17	1	1	0	1	1	1	1	1	Value8								*Software Interrupt*
18	1	1	1	0	0	Offset11											*Unconditional branch*
19	1	1	1	1	H	Offset											*Long branch with link*
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	

ARM 7 ISA detail



Mnemonic	Instruction	Lo register operand	Hi register operand	Condition codes set	See Section:
ADC	Add with Carry	✔		✔	5.4
ADD	Add	✔	✔	✔①	5.1.3, 5.5, 5.12, 5.13
AND	AND	✔		✔	5.4
ASR	Arithmetic Shift Right	✔		✔	5.1, 5.4
B	Unconditional branch	✔			5.16
B*xx*	Conditional branch	✔			5.17
BIC	Bit Clear	✔		✔	5.4
BL	Branch and Link				5.19
BX	Branch and Exchange	✔	✔		5.5
CMN	Compare Negative	✔		✔	5.4
CMP	Compare	✔	✔	✔	5.3, 5.4, 5.5
EOR	EOR	✔		✔	5.4
LDMIA	Load multiple	✔			5.15
LDR	Load word	✔			5.7, 5.6, 5.9, 5.11
LDRB	Load byte	✔			5.7, 5.9
LDRH	Load halfword	✔			5.8, 5.10
LSL	Logical Shift Left	✔		✔	5.1, 5.4
LDSB	Load sign-extended byte	✔			5.8
LDSH	Load sign-extended halfword	✔			5.8
LSR	Logical Shift Right	✔		✔	5.1, 5.4
MOV	Move register	✔	✔	✔②	5.3, 5.5
MUL	Multiply	✔		✔	5.4
MVN	Move Negative register	✔		✔	5.4

Microcontroller Seminar -- ASH

ARM 7 ISA detail

Mnemonic	Instruction	Lo register operand	Hi register operand	Condition codes set	See Section:
NEG	Negate	✔		✔	5.4
ORR	OR	✔		✔	5.4
POP	Pop registers	✔			5.14
PUSH	Push registers	✔			5.14
ROR	Rotate Right	✔		✔	5.4
SBC	Subtract with Carry	✔		✔	5.4
STMIA	Store Multiple	✔			5.15
STR	Store word	✔			5.7, 5.9, 5.11
STRB	Store byte	✔			5.7
STRH	Store halfword	✔			5.8, 5.10
SWI	Software Interrupt				5.17
SUB	Subtract	✔		✔	5.1.3, 5.3
TST	Test bits	✔		✔	5.4

Microcontroller Seminar -- ASH

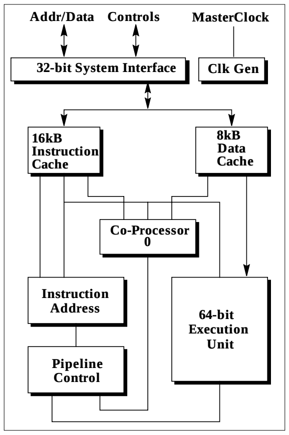
MIPS

- origins
  - defined by John Hennessey c. 1980
  - turned into a company by John Hennessey, Skip Stritter, and John Moussouris
- architecture spans several generations, most recent rev of ISA is MIPS IV
- MIPS is currently a fabless company who licenses technology, cores, and IP
  - Toshiba, NEC, IDT, LSI Logic, NKK, Philips, and QED (QED is also fabless)
- Range of MIPS varieties
  - 32 or 64-bit versions
  - 16 MHz to 300 MHz
  - 54 mW to 30W
  - FPUs up to 1.2 GFlop
  - 0.5u to 0.25u processes, 2mm^2 to 290 mm^2 implementations

MIPS

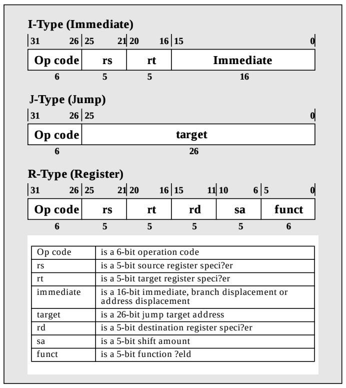
- originally targetted for workstation/supercomputer CPU
  - migrated to embedded applications, now one of the most popular embedded architectures
  - PowerPC is taking a similar track
- typical MIPS architecture: NEC VR4300
  - used in Nintendo 64
  - 64-bit, 167 MHz top speed
  - 1.4W @ 100 MHz
  - 16K I-cache, 8K D-cache
  - MIPS-III ISA
    - 32 general-purpose registers
    - 5-stage pipeline
  - internal FPU and MMU, no other integrated peripherals
  - $28.50, qty 1, Marshall

VR4300 block diagram



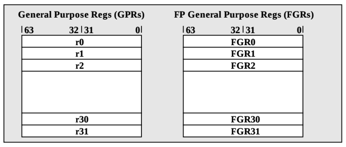
VR4300 ISA

Instruction Formats

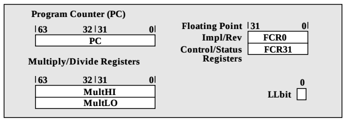


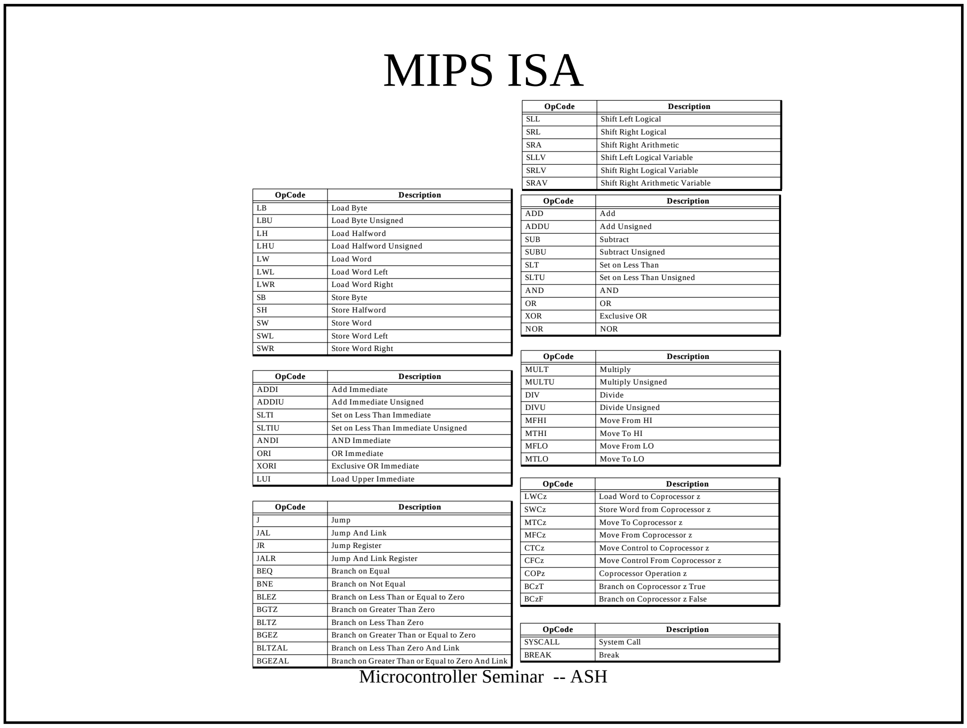
Op code	is a 6-bit operation code
rs	is a 5-bit source register speci?er
rt	is a 5-bit target register speci?er
immediate	is a 16-bit immediate, branch displacement or address displacement
target	is a 26-bit jump target address
rd	is a 5-bit destination register speci?er
sa	is a 5-bit shift amount
funct	is a 5-bit function ?eld

Register Set



Special Purpose Registers


Program Counter (PC)
63 32 31 0
PC
Multiply/Divide Registers
63 32 31 0
MultHI
MultLO
Floating Point Impl/Rev Control/Status Registers
31 0
FCR0
FCR31
0
LLbit



MIPS ISA

OpCode	Description
LB	Load Byte
LBU	Load Byte Unsigned
LH	Load Halfword
LHU	Load Halfword Unsigned
LW	Load Word
LWL	Load Word Left
LWR	Load Word Right
SB	Store Byte
SH	Store Halfword
SW	Store Word
SWL	Store Word Left
SWR	Store Word Right

OpCode	Description
ADDI	Add Immediate
ADDIU	Add Immediate Unsigned
SLTI	Set on Less Than Immediate
SLTIU	Set on Less Than Immediate Unsigned
ANDI	AND Immediate
ORI	OR Immediate
XORI	Exclusive OR Immediate
LUI	Load Upper Immediate

OpCode	Description
J	Jump
JAL	Jump And Link
JR	Jump Register
JALR	Jump And Link Register
BEQ	Branch on Equal
BNE	Branch on Not Equal
BLEZ	Branch on Less Than or Equal to Zero
BGTZ	Branch on Greater Than Zero
BLTZ	Branch on Less Than Zero
BGEZ	Branch on Greater Than or Equal to Zero
BLTZAL	Branch on Less Than Zero And Link
BGEZAL	Branch on Greater Than or Equal to Zero And Link

OpCode	Description
SLL	Shift Left Logical
SRL	Shift Right Logical
SRA	Shift Right Arithmetic
SLLV	Shift Left Logical Variable
SRLV	Shift Right Logical Variable
SRAV	Shift Right Arithmetic Variable

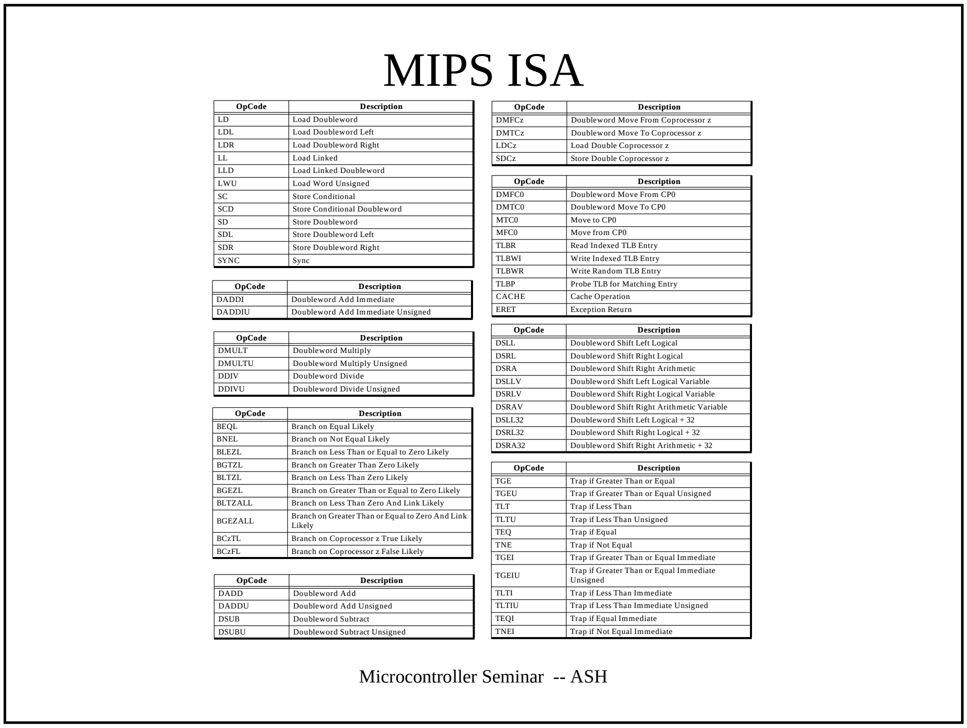
OpCode	Description
ADD	Add
ADDU	Add Unsigned
SUB	Subtract
SUBU	Subtract Unsigned
SLT	Set on Less Than
SLTU	Set on Less Than Unsigned
AND	AND
OR	OR
XOR	Exclusive OR
NOR	NOR

OpCode	Description
MULT	Multiply
MULTU	Multiply Unsigned
DIV	Divide
DIVU	Divide Unsigned
MFHI	Move From HI
MTHI	Move To HI
MFLO	Move From LO
MTLO	Move To LO

OpCode	Description
LWCz	Load Word to Coprocessor z
SWCz	Store Word from Coprocessor z
MTCz	Move To Coprocessor z
MFCz	Move From Coprocessor z
CTCz	Move Control to Coprocessor z
CFCz	Move Control From Coprocessor z
COPz	Coprocessor Operation z
BCzT	Branch on Coprocessor z True
BCzF	Branch on Coprocessor z False

OpCode	Description
SYSCALL	System Call
BREAK	Break

Microcontroller Seminar -- ASH



MIPS ISA

OpCode	Description
LD	Load Doubleword
LDL	Load Doubleword Left
LDR	Load Doubleword Right
LL	Load Linked
LLD	Load Linked Doubleword
LWU	Load Word Unsigned
SC	Store Conditional
SCD	Store Conditional Doubleword
SD	Store Doubleword
SDL	Store Doubleword Left
SDR	Store Doubleword Right
SYNC	Sync

OpCode	Description
DADDI	Doubleword Add Immediate
DADDIU	Doubleword Add Immediate Unsigned

OpCode	Description
DMULT	Doubleword Multiply
DMULTU	Doubleword Multiply Unsigned
DDIV	Doubleword Divide
DDIVU	Doubleword Divide Unsigned

OpCode	Description
BEQL	Branch on Equal Likely
BNEL	Branch on Not Equal Likely
BLEZL	Branch on Less Than or Equal to Zero Likely
BGTZL	Branch on Greater Than Zero Likely
BLTZL	Branch on Less Than Zero Likely
BGEZL	Branch on Greater Than or Equal to Zero Likely
BLTZALL	Branch on Less Than Zero And Link Likely
BGEZALL	Branch on Greater Than or Equal to Zero And Link Likely
BCzTL	Branch on Coprocessor z True Likely
BCzFL	Branch on Coprocessor z False Likely

OpCode	Description
DADD	Doubleword Add
DADDU	Doubleword Add Unsigned
DSUB	Doubleword Subtract
DSUBU	Doubleword Subtract Unsigned

OpCode	Description
DMFCz	Doubleword Move From Coprocessor z
DMTCz	Doubleword Move To Coprocessor z
LDCz	Load Double Coprocessor z
SDCz	Store Double Coprocessor z

OpCode	Description
DMFC0	Doubleword Move From CP0
DMTC0	Doubleword Move To CP0
MTC0	Move to CP0
MFC0	Move from CP0
TLBR	Read Indexed TLB Entry
TLBWI	Write Indexed TLB Entry
TLBWR	Write Random TLB Entry
TLBP	Probe TLB for Matching Entry
CACHE	Cache Operation
ERET	Exception Return

OpCode	Description
DSLL	Doubleword Shift Left Logical
DSRL	Doubleword Shift Right Logical
DSRA	Doubleword Shift Right Arithmetic
DSLLV	Doubleword Shift Left Logical Variable
DSRLV	Doubleword Shift Right Logical Variable
DSRAV	Doubleword Shift Right Arithmetic Variable
DSLL32	Doubleword Shift Left Logical + 32
DSRL32	Doubleword Shift Right Logical + 32
DSRA32	Doubleword Shift Right Arithmetic + 32

OpCode	Description
TGE	Trap if Greater Than or Equal
TGEU	Trap if Greater Than or Equal Unsigned
TLT	Trap if Less Than
TLTU	Trap if Less Than Unsigned
TEQ	Trap if Equal
TNE	Trap if Not Equal
TGEI	Trap if Greater Than or Equal Immediate
TGEIU	Trap if Greater Than or Equal Immediate Unsigned
TLTI	Trap if Less Than Immediate
TLTIU	Trap if Less Than Immediate Unsigned
TEQI	Trap if Equal Immediate
TNEI	Trap if Not Equal Immediate

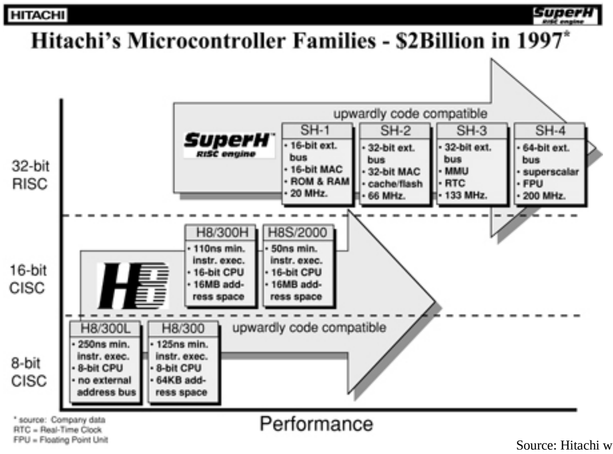
Microcontroller Seminar -- ASH

Hitachi SH

- Hitachi SH is one of many products made by Hitachi
  - for example, hitachi also makes nuclear reactors and the H8 series embedded FLASH controllers
- "Super-H" RISC engine
  - Scalable architecture, from SH-1 to SH-4
    - upward code compatible across entire family
    - applications from motion controllers to high-end game decks and windows CE computers
    - SH-1 is 20 MHz, highly integrated peripheral set, MAC, cost-sensitive apps
    - SH-2 is 66 MHz, highly integrated peripheral set, DSP version available, cache
    - SH-3 is 133 MHz, DSP version available, MMU, cache
    - SH-4 is 200 MHz, integrated FPU, 3-D vector unit, MMU, plus other peripherals, cache

Hitachi SH

- Targetted at low-power, cost-sensitive applications but still requiring high performance

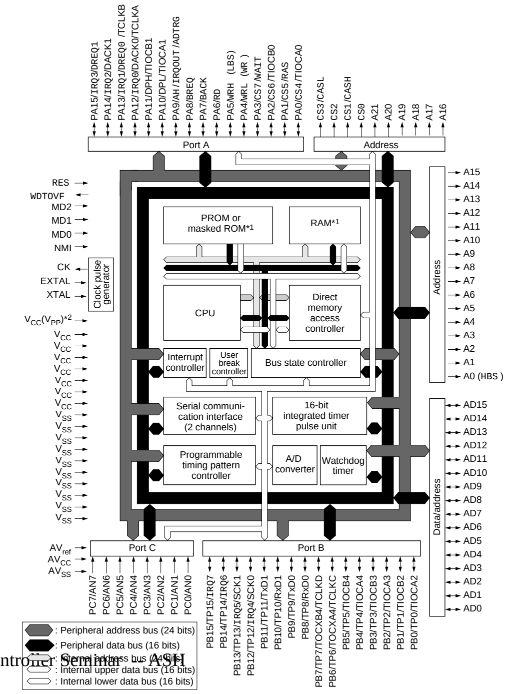


Source: Hitachi website

Hitachi SH

- Typical example: SH-1
  - 20 MHz @ 5V, 12.5 MHZ @ 3.3V
  - 32-bit RISC processor, 16 general purpose registers
  - 130 mW typ. @ 12.5 MHz
  - 5-stage pipeline
  - hardware multiply-accumulate unit
  - $25 qty. 1 for A-series die shrink
  - integrated peripheral set: interrupt, DMA, serial, timing controller, pulse generator, A/D, watchdog, bus I/F to DRAM, 4-8K RAM (user managed cache) plus masked ROM
  - 32 MB max external memory

SH-1 Block Diagram

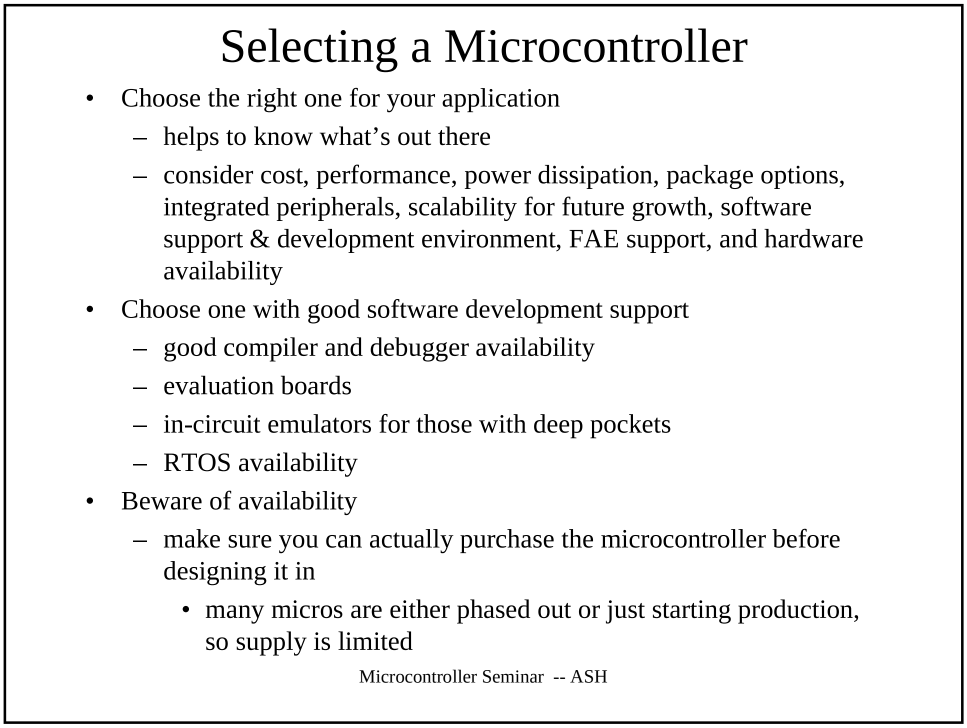


SH-1 ISA

Classification	Types	Operation Code	Function	Number of Instructions
Data transfer	5	MOV	Data transfer, immediate data transfer, supporting module data transfer, structure data transfer	39
		MOVA	Effective address transfer	
		MOVT	T bit transfer	
		SWAP	Swap of upper and lower bytes	
		XTRCT	Extraction of the middle of registers connected	
Arithmetic operations	17	ADD	Binary addition	28
		ADDC	Binary addition with carry	
		ADDV	Binary addition with overflow check	
		CMP/cond	Comparison	
		DIV1	Division	
		DIV0S	Initialization of signed division	
		DIV0U	Initialization of unsigned division	
		EXTS	Sign extension	
		EXTU	Zero extension	
		MAC	Multiplication and accumulation	
		MULS	Signed multiplication	
		MULU	Unsigned multiplication	
		NEG	Negation	
		NEGC	Negation with borrow	
		SUB	Binary subtraction	
		SUBC	Binary subtraction with carry	
		SUBV	Binary subtraction with underflow check	
Logic operations	6	AND	Logical AND	14
		NOT	Bit inversion	
		OR	Logical OR	
		TAS	Memory test and bit set	

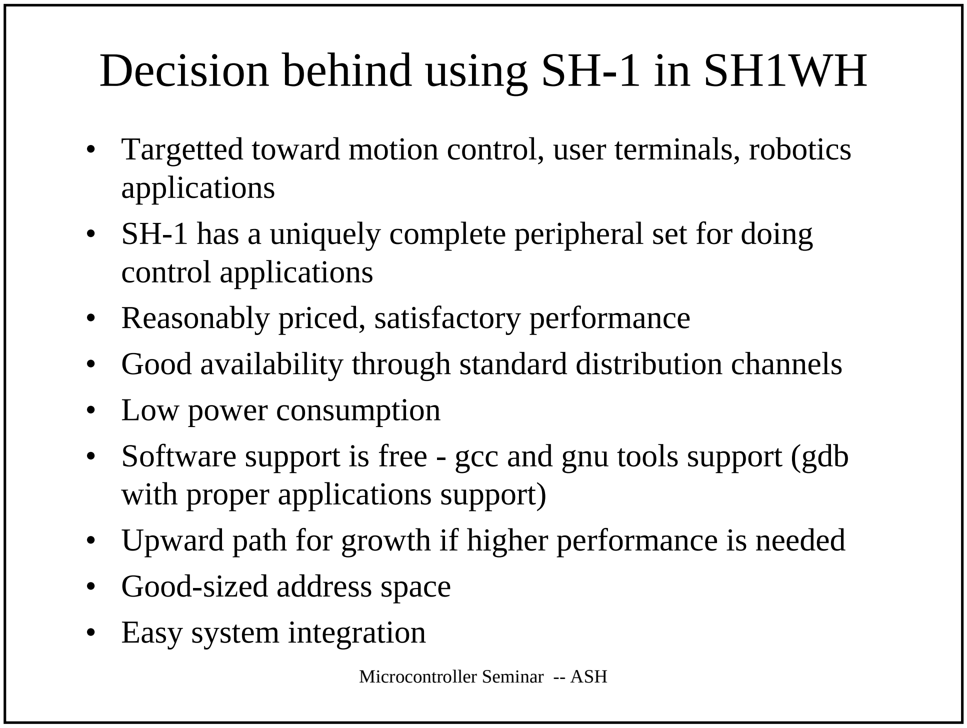
SH-1 ISA

Classification	Types	Operation Code	Function	Number of Instructions
Logic operations (cont)	6	TST	Logical AND and T bit set	14
		XOR	Exclusive OR	
Shift	10	ROTL	One-bit left rotation	14
		ROTR	One-bit right rotation	
		ROTCL	One-bit left rotation with T bit	
		ROTCR	One-bit right rotation with T bit	
		SHAL	One-bit arithmetic left shift	
		SHAR	One-bit arithmetic right shift	
		SHLL	One-bit logical left shift	
		SHLLn	n-bit logical left shift	
		SHLR	One-bit logical right shift	
		SHLRn	n-bit logical right shift	
Branch	7	BF	Conditional branch (T = 0)	7
		BT	Conditional branch (T = 1)	
		BRA	Unconditional branch	
		BSR	Branch to subroutine procedure	
		JMP	Unconditional branch	
		JSR	Branch to subroutine procedure	
		RTS	Return from subroutine procedure	
System control	11	CLRT	T bit clear	31
		CLRMAC	MAC register clear	
		LDC	Load to control register	
		LDS	Load to system register	
		NOP	No operation	
		RTE	Return from exception handling	
		SETT	T bit set	
		SLEEP	Shift into power-down mode	
		STC	Store control register data	
		STS	Store system register data	
		TRAPA	Trap exception handling	
Total	56			133



Selecting a Microcontroller

- Choose the right one for your application
  - helps to know what’s out there
  - consider cost, performance, power dissipation, package options, integrated peripherals, scalability for future growth, software support & development environment, FAE support, and hardware availability
- Choose one with good software development support
  - good compiler and debugger availability
  - evaluation boards
  - in-circuit emulators for those with deep pockets
  - RTOS availability
- Beware of availability
  - make sure you can actually purchase the microcontroller before designing it in
    - many micros are either phased out or just starting production, so supply is limited



Decision behind using SH-1 in SH1WH

- Targetted toward motion control, user terminals, robotics applications
- SH-1 has a uniquely complete peripheral set for doing control applications
- Reasonably priced, satisfactory performance
- Good availability through standard distribution channels
- Low power consumption
- Software support is free - gcc and gnu tools support (gdb with proper applications support)
- Upward path for growth if higher performance is needed
- Good-sized address space
- Easy system integration