Cheatography

HEPIA-RISC Assembler Cheat Sheet

by Krrag via cheatography.com/205339/cs/43813/

Registers

Register	Purpose
R0	Function Argument 1 return value
R1	Function Argument 2
R2	Function Argument 3
R3	Function Argument 4
R4	Interrupt Vector Register
R5	Interrupt Vector Register customasm macros
R6	Systick
R7	Register Link
R8	General purpose
R9	General purpose
R10	General purpose
R11	General purpose
R12	General purpose
R13	General purpose
R14	General purpose
R15	Stack Pointer
IRL	Interrupt Register Link
PC	Program Counter
SR	Status Register

IRL, PC & SR are not addressable

Size

Word (16-bits)	w
Byte (8-bits)	b

Instructions default to word if no size specified

Conditions

Mnemonic	Meaning
VC	No overflow
VS	Overflow
HS	Unsigned higher or same
LO	Unsigned lower
MI	Negative
PL	Positive
EQ	Equal

Conditions (cont)

NE	Not equal
HI	Unsigned higher
GT	Signed greater
GE	Signed greater or equal
LT	Signed less
LE	Signed lower or equal
LS	Unsigned lower or equal

Function call and jumps

Instruction	Example	Result
bl	bl func	R7 = PC, PC=func
bx	bx	PC = R7
b	b #-2	PC = PC - 2
b{cond}	bgt loop	PC = loop if z=0 and n=v

Both version of "b" can use labels or relative values

Load and store instructions

Instruction	Example	Result
ldr	ldr r0 [r1], #3	r0 = [0x4003]
str	str r0, [r1]	[0x4000] = r0

Assume r1 = 0x4000
[value] indicates a data bus address

Arithmetic Instructions

Instruction	Example	Result
mov	mov r3, #3	r3 = 3
add	add r3, r0, r1	r3 = 1 + 2
sub	sub r2, r0	r2 = 3 - 1
lsl	lsl r3, r0, #2	r3 = 1 << 2 = 4
lsr	lsr r3, r0, #2	r3 = 1 >> 2 = 0
asr	asr r3, r0, #2	r3 = 1 >>> 3 = 0

Arithmetic Instructions (cont)

and	and r3, r1, r0	r3 = (1 and 2) = 0
orr	orr r3, r1, r0	r3 = (1 or 2) = 3
not	not r3, r1	r3 = ~r1 = -3

assume r0 = 1, r1 = 2, r2 = 3

Pseudo-ops

Instruction	Example	True Operation
nop	nop	r0 = r0 and r0
eor	eor r0, r1, r2	r0 = (r1 or r2) - (r1 and r2)
ib{cond}	ibgt #2	ble (branch if not gt)
cmp	cmp r0, r1	Set flags using sub
cmn	cmn r0, r1	Set flags using add
{op} {cond}	addeq r0, r1, r2	r0 = r1 + r2 if z=0
psh	psh r0	[SP] = r0
pop	pop r0	r0 = [SP]
zero	zero r0	r0 = 0

push and pop do not automatically increment/decrement SP.