SPIM16 Reference Data

BASIC INSTRUCTION FORMATS

R	`oooo ssss tttt dddd`
	`0fff ssss tttt dddd`
RI	`oooo ssss tttt iiii`
I	`oooo ssss iiiiiiii`
J	`oooo aaaaaaaaaaaa`

opcode	`o`
function	`f`
rs	`s`
rt	`t`
rd	`d`
immediate	`i`
address	`a`

REGISTER FILE INFORMATION

NAME	NUMBER	USE	PRESERVED?
$zero	0	The Constant Value 0	N/A
$v0, $v1	1, 2	Value for Function Results and Expression Evaluation	No
$a0, $a1	3, 4	Function Argument Value	No
$t0-$t3	5-8	Temporary Values	No
$s0-$s3	9-12	Saved Values	Yes
$sp	13	Stack Pointer	Yes
$gp	14	Global Pointer	Yes
$ra	15	Return Address	Yes

CORE INSTRUCTION SET

NAME, MNEMONIC		FORMAT	OPERATION (in Verilog)		OPCODE / FUNCT
Or	`or`	R	R[rd] = R[rs] | R[rt]		0
Xor	`xor`	R	R[rd] = R[rs] ^ R[rt]		1
And	`and`	R	R[rd] = R[rs] & R[rt]		2
Add	`add`	R	R[rd] = R[rs] + R[rt]	(1)	3
Sub	`sub`	R	R[rd] = R[rs] - R[rt]	(1)	4
Shift Left Logical	`sll`	RI	R[rd] = R[rs] << immediate		5
Shift Right Logical	`srl`	RI	R[rd] = R[rs] >> immediate		6
Shift Right Arithmetic	`sra`	RI	R[rd] = R[rs] >>> immediate		7
Load Word	`lw`	RI	R[rt] = M[R[rs]+SignExtImm]	(2)	8
Store Word	`sw`	RI	M[R[rs]+SignExtImm] = R[rt]	(2)	9
Load Immediate	`li`	I	R[rs] = SignExtImm	(3)	A
Branch Not Zero	`bnz`	I	if (R[rs] != 0) PC = PC+2+BranchAddr	(4)	B
Branch Zero	`bz`	I	if (R[rs] == 0) PC = PC+2+BranchAddr	(4)	C
Set Less Than	`slt`	R	R[rd] = R[rs] < R[rt]		D
Jump	`j`	J	PC = JumpAddr	(5)	E
Jump Register	`jr`	R	PC = R[rs]		F

1 May cause overflow exception
2 SignExtImm = { 12{ immediate[3] }, immediate }
3 SignExtImm = { 8{ immediate[7] }, immediate }
4 BranchAddr = { 7{ immediate[7] }, immediate, 1'b0 }
5 JumpAddr = { PC+2[15:12], address, 1'b0 }

DATA ALIGNMENT

<table>
<tr><td colspan="4">WORD</td></tr>
<tr><td colspan="2">BYTE 0</td><td colspan="2">BYTE 1</td></tr>
<tr><td>NIBBLE</td><td>NIBBLE</td><td>NIBBLE</td><td>NIBBLE</td></tr>
</table>

MEMORY ALLOCATION

Addresses	Allocation
0x7FFF – 0x1000	Stack Data
0x1000 – 0x7FFF	Heap Data
0x0000 – 0x0FFF	Text/Static Data