HOLY CORE

REFERENCE TABLE

SINGLE CYCLE EDITION

&

FPGA EDITION

instruction name	type	OP	alu_op	imm_source	mem_write	reg_write	alu_source	write_back_source	branch	jump	second_add_source		Support ?
lw	I-Type	0000011	00	000	0	1	1	01	0	0	XX	YES	
sw	S-Type	0100011	00	001	1	0	1	XX	0	0	XX	YES	
add	R-Type	0110011	10	XXX	0	1	0	00	0	0	XX	YES	
and	R-Type	0110011	10	XXX	0	1	0	00	0	0	XX	YES	
or	R-Type	0110011	10	XXX	0	1	0	00	0	0	XX	YES	
beq	B-Type	1100011	01	010	0	0	0	XX	1	0	00	YES	
jal	J-Type	1101111	XX	011	0	1	X	10	0	1	00	YES	
addi	I-Type (ALU)	0010011	10	000	0	1	1	00	0	0	XX	YES	
lui	U-Type	0110111	XX	100	0	1	X	11	0	0	01	YES	
auipc	U-Type	0010111	XX	100	0	1	X	11	0	0	00	YES	
jalr	I-Type (Jump)	1100111	XX	000	0	1	X	10	0	1	10	YES	
bne	B-Type	1100011	01	010	0	0	0	XX	1	0	00	YES	
blt	B-Type	1100011	01	010	0	0	0	XX	1	0	00	YES	
bge	B-Type	1100011	01	010	0	0	0	XX	1	0	00	YES	
bltu	B-Type	1100011	01	010	0	0	0	XX	1	0	00	YES	
bgeu	B-Type	1100011	01	010	0	0	0	XX	1	0	00	YES	
lb	I-Type	0000011	01	010	0	1	1	01	0	0	XX	YES	
lh	I-Type	0000011	01	010	0	1	1	01	0	0	XX	YES	
lbu	I-Type	0000011	01	010	0	1	1	01	0	0	XX	YES	
lhu	I-Type	0000011	01	010	0	1	1	01	0	0	XX	YES	
sb	S-Type	0100011	00	001	1	0	1	XX	0	0	XX	YES	
sh	S-Type	0100011	00	001	1	0	1	XX	0	0	XX	YES	
slti	I-Type (ALU)	0010011	10	000	0	1	1	00	0	0	XX	YES	
sltiu	I-Type (ALU)	0010011	10	000	0	1	1	00	0	0	XX	YES	
xori	I-Type (ALU)	0010011	10	000	0	1	1	00	0	0	XX	YES	
ori	I-Type (ALU)	0010011	10	000	0	1	1	00	0	0	XX	YES	
andi	I-Type (ALU)	0010011	10	000	0	1	1	00	0	0	XX	YES	
slli	I-Type (ALU)	0010011	10	000	0	1	1	00	0	0	XX	YES	
srli	I-Type (ALU)	0010011	10	000	0	1	1	00	0	0	XX	YES	
srai	I-Type (ALU)	0010011	10	000	0	1	1	00	0	0	XX	YES	
sub	R-Type	0010011	10	XXX	0	1	0	00	0	0	XX	YES	
sll	R-Type	0010011	10	XXX	0	1	0	00	0	0	XX	YES	
slt	R-Type	0010011	10	XXX	0	1	0	00	0	0	XX	YES	
sltu	R-Type	0010011	10	XXX	0	1	0	00	0	0	XX	YES	
xor	R-Type	0010011	10	XXX	0	1	0	00	0	0	XX	YES	
srl	R-Type	0010011	10	XXX	0	1	0	00	0	0	XX	YES	
sra	R-Type	0010011	10	XXX	0	1	0	00	0	0	XX	YES	
pause	NOP												
fence	NOP												
fence.tso	NOP												
ecall	?												
ebrek	?												

ALU OP	f3	f7	op[5]	ALU_control	comment
		LOADS/STORES			
00	XXX	XXXXXXX	X	0000	lw,sw, ...
		BRANCHES			
01	000	XXXXXXX	X	0001	beq
	001	XXXXXXX	X	0001	bne
	100	XXXXXXX	X	0101	blt
	101	XXXXXXX	X	0101	bge
	110	XXXXXXX	X	0111	bltu
	111	XXXXXXX	X	0111	bgeu
		MATH : R-TYPES, I-Types ALU (general purpose arithmetic)			
10	000	XXXXXXX	0	0000	addi
	000	0000000	1	0000	add
	000	0100000	1	0001	sub
	001	0000000	X	0100	sll, slli
	010	XXXXXXX	X	0101	slt, slti
	011	XXXXXXX	X	0111	sltu, sltiu
	100	XXXXXXX	X	1000	xor, xori
	101	0000000	X	0110	srl, srli
	101	0100000	X	1001	sra, srai
	110	XXXXXXX	X	0011	or, ori
	111	XXXXXXX	X	0010	and, andi

ALU CONTROL ARITHMETIC CORRESPONDANCE TABLE	
0000	add
0001	substract
0010	and
0011	or
0100	shitft left logical
0101	set less than
0110	shitft right logical
0111	set less than unsigned
1000	xor
1001	shift right arithmetic (extends sign)

f3	instruction	alu_zero	alu_last bit	assert_branch
000	beq	1	X	1
001	bne	0	X	1
100	blt	X	1	1
101	bge	X	0	1
110	bltu	X	1	1
111	bgeu	X	0	1

	000	001	010	011	100	101	110
alu_op	loads & stores, result in an add for alu ctrl	branches realted, serve as source select for branch flag conditions (zero, less than, ...)	R-Types, alu then uses f3, f7 and op[5] to differenciate	XXX	XXX	XXX	XXX
imm_source	I-Type source, 12 bits signed extended	S-type 2 parts imm, 12 bits signed	B-Type scattered imm, 13 bits signed once a 0 is added at the end (half word aligned)	21 bits J-Type once 0 is added	20 bits upper immediate select	regular 12 bits I-Type immediate, except sign is NOT extended	XXX
mem_write	DO NOT write R2 in memory	write R2 in memory	XXX	XXX	XXX	XXX	XXX
reg_write	DO NOT Write "write_back_data" to R3	Write "write_back_data" to R3	XXX	XXX	XXX	XXX	XXX
alu_source	use R2 as second source for ALU	use IMM as second source for IMM	XXX	XXX	XXX	XXX	XXX
write_back_source	Use the ALU result as write_back_data	Use the memory read data as write_back_data	Use PC+4 as write_back_data	use the second_add result as write_back_data (whiwh can be "immediate" OR "immediate + PC", explore this table for more details)	XXX	XXX	XXX
branch	Not a B-type instruction : no branching possible (does not exclude jumping !)	B-Type instruction : possibility of branching if condition is met	XXX	XXX	XXX	XXX	XXX
jump	Not unconditional jump possible	Unditional jump : the PC_SOURCE signal will get set to 1	XXX	XXX	XXX	XXX	XXX
second_add_source	Sets the second add to PC+IMM	Sets the second add to IMM only (mainly for lui)	XXX	XXX	XXX	XXX	XXX
pc_source	The nex pc will be PC+4 an move on to the next instruction	The nex pc will be Second_add (for branches and jumps)	The next pc will be RD1 + IMM : Use RD1 as source. (for jalr)	XXX	XXX	XXX	XXX
ALU_CONTROL (cf alu decoder page)	see alu decoder sheet	see alu decoder sheet	see alu decoder sheet	see alu decoder sheet	see alu decoder sheet	see alu decoder sheet	see alu decoder sheet
sign (load_store_decoder / be_decoder)	The reader WILL sign-extend the data coming out of memory	The reader will NOT sign-extend the data coming out of memory	XXX	XXX	XXX	XXX	XXX