Duke ECE152 – Spring 2011 – Instruction Set Architecture

Duke152-S11-32

Instruction	Opcode	Type	ALU Opcode	Usage	Operation
add	00000	R	00000	add $rd, $rs, $rt	$rd = $rs + $rt
sub	00000	R	00001	sub $rd, $rs, $rt	$rd = $rs – $rt
and	00000	R	00010	and $rd, $rs, $rt	$rd = $rs AND $rt
or	00000	R	00011	or $rd, $rs, $rt	$rd = $rs OR $rt
sll	00000	R	00100	sll $rd, $rs, shamt	$rd = $rs shifted left
sra	00000	R	00101	sra $rd, $rs, shamt	$rd = $rs shifted right arithmetic
custr1	00000	R	00110	custr1 $rd, $rs, $rt	student custom defined
custr2	00000	R	00111	custr2 $rd, $rs, $rt	student custom defined
custr3	00000	R	01000	custr3 $rd, $rs, $rt	student custom defined
custr4	00000	R	01001	custr4 $rd, $rs, $rt	student custom defined
custr5	00000	R	01010	custr5 $rd, $rs, $rt	student custom defined
custr6	00000	R	01011	custr6 $rd, $rs, $rt	student custom defined
custr7	00000	R	01100	custr7 $rd, $rs, $rt	student custom defined
custr8	00000	R	01101	custr8 $rd, $rs, $rt	student custom defined
addi	00001	I	N/A	addi $rd, $rs, N	$rd = $rs + N
lw	00010	I	N/A	lw $rd, N($rs)	$rd = Mem[$rs+N]
sw	00011	I	N/A	sw $rd, N($rs)	Mem[$rs+N] = $rd
bne	00100	I	N/A	bne $rd, $rs, N	if ($rd!=$rs) then PC = PC+1+N
blt	00101	I	N/A	blt $rd, $rs, N	if ($rd<$rs) then PC = PC+1+N
j	00110	J	N/A	j N	PC = N
jal	00111	J	N/A	jal N	$r31 = PC+1; PC = N
jr	01000	I	N/A	jr $rd	PC = $rd
input	01001	I	N/A	input $rd	$rd = keyboard input
output	01010	I	N/A	output $rd	LCD output = $rd (lower 8 bits)
custi1	01011	I	N/A	custi1 $rd, $rs, N	student custom defined
custi2	01100	I	N/A	custi2 $rd, $rs, N	student custom defined
custi3	01101	I	N/A	custi3 $rd, $rs, N	student custom defined
custi4	01110	I	N/A	custi4 $rd, $rs, N	student custom defined
custj1	01111	J	N/A	custj1 N	student custom defined
custj2	10000	J	N/A	custj2 N	student custom defined

Instruction Type	Instruction Format						
R	opcode(5)	rd(5)	rs(5)	rt(5)	shamt(5)	ALUop(5)	zeroes(2)
I	opcode(5)	rd(5)	rs(5)	immediate(17)			
J	opcode(5)	target(27)					

R-type instruction field shamt(5) is unsigned.
I-type immediate(17) is signed 2's complement and is sign-extended to the full 32-bit word size.
J-type target(27) is extended to the full PC size by using the upper bits from the current PC.
Register fields that are undefined are filled with zeroes by the assembler.
Register $r0 always equals zero. Registers $r1 through $r30 are general purpose. Register $r31 stores the return address of a jump-and-link instruction.
Instructions that change control flow (bne, blt, j, jal, jr) do not have a delay slot.
The Input instruction shall assert high on input_ack for the cycle only when the input is read from the keyboard controller; otherwise it shall assert low. The Output instruction shall assert high on LCD_wren for the cycle only when the data is outputted to the LCD controller; otherwise it shall assert low.

Memory is word-addressed. The instruction and data memory address spaces are separate. Static data begins at data memory address zero. Stack data begins at the end of the data memory and grows downwards. There is no preset boundary between the end of static data and the start of the upwards-growing heap; this is a property of the assembly program.

After a reset, all register values are zero and program execution begins from instruction memory address zero. The memories' contents are not reset.

Revised by John Ingalls on January 5, 2010.