

# Duke ECE152 – Spring 2012 – Instruction Set Architecture

## Duke152-S12-32

Instruction	Opcode	Type	Usage	Operation
add	00000	R	add \$rd, \$rs, \$rt	\$rd = \$rs + \$rt
sub	00001	R	sub \$rd, \$rs, \$rt	\$rd = \$rs – \$rt
and	00010	R	and \$rd, \$rs, \$rt	\$rd = \$rs AND \$rt
or	00011	R	or \$rd, \$rs, \$rt	\$rd = \$rs OR \$rt
sll	00100	R	sll \$rd, \$rs, \$rt	\$rd = \$rs shifted left by \$rt[4:0] zero-fill
srl	00101	R	srl \$rd, \$rs, \$rt	\$rd = \$rs shifted right by \$rt[4:0] zero-extend
addi	00110	I	addi \$rd, \$rs, N	\$rd = \$rs + N
lw	00111	I	lw \$rd, N(\$rs)	\$rd = Mem[\$rs + N]
sw	01000	I	sw \$rd, N(\$rs)	Mem[\$rs + N] = \$rd
beq	01001	I	beq \$rd, \$rs, N	if (\$rd == \$rs) then PC = PC+1+N
bgt	01010	I	bgt \$rd, \$rs, N	if (\$rd > \$rs) then PC = PC+1+N
jr	01011	I	jr \$rd	PC = \$rd
j	01100	J	j N	PC = N
jal	01101	J	jal N	\$r31 = PC+1; PC = N
input	01110	I	input \$rd	\$rd = keyboard input
output	01111	I	output \$rd	LCD output = \$rd (lower 8 bits)

Instruction Type	Instruction Format				
<b>R</b>	Opcode [31:27]	RD [26:22]	RS [21:17]	RT [16:12]	Zeroes [11:0]
<b>I</b>	Opcode [31:27]	RD [26:22]	RS [21:17]	Immediate [16:0]	
<b>J</b>	Opcode [31:27]	Target [26:0]			

I-type immediate field [16:0] is signed 2's complement and sign-extended to the full 32-bit word size.

J-type target field [26:0] is extended to the full 32-bit PC size using the upper bits from the current PC+1.

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 link address of a jump-and-link instruction.

Instructions that change control flow (beq, bgt, 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 December 24, 2011.*