Instruction	**Opcode**	**Format**	**Meaning**
add	“00000”	[op][reg1][reg2][reg3]	reg1 = reg2 + reg3
sub	“00001”	[op][reg1][reg2][reg3]	reg1 = reg2 - reg3
sll	“00010”	[op][reg1][reg2][*reg3]	reg1 = reg2 << 1
srl	“00011”	[op][reg1][reg2][*reg3]	reg1 = reg2 >> 1
sra	“00100”	[op][reg1][reg2][*reg3]	reg1 = reg2 >>> 1
and	“00101”	[op][reg1][reg2][reg3]	reg1 = reg2 & reg3
or	“00110”	[op][reg1][reg2][reg3]	reg1 = reg2 | reg3
xor	“00111”	[op][reg1][reg2][reg3]	reg1 = reg2 ^ reg3
slt	“01000”	[op][reg1][reg2][reg3]	reg1 = (reg2 < reg3) ? 1 : 0
sgt	“01001”	[op][reg1][reg2][reg3]	reg1 = (reg2 > reg3) ? 1 : 0
seq	“01010”	[op][reg1][reg2][reg3]	reg1 = (reg2 == reg3) ? 1 : 0
send	“01011”	[op][reg1][*reg2][*reg3]	sendUART(reg1[7:0])
recv	“01100”	[op][reg1][*reg2][*reg3]	reg1 = x”00” & recvUART()
jr	“01101”	[op][reg1][*reg2][*reg3]	pc = reg1
wpix	“01110”	[op][reg1][reg2][*reg3]	framebuffer[reg1[11:0]] = reg2[15:0]
rpix	“01111”	[op][reg1][reg2][*reg3]	reg1 = framebuffer[reg2[11:0]]
beq	“10000”	[op][reg1][reg2][imm]	if(reg1 == reg2) pc = imm
bne	“10001”	[op][reg1][reg2][imm]	if(reg1 != reg2) pc = imm
ori	“10010”	[op][reg1][reg2][imm]	reg1 = reg2 | imm
lw	“10011”	[op][reg1][reg2][imm]	reg1 = dmem[reg2 + imm]
sw	“10100”	[op][reg1][reg2][imm]	dmem[reg2 + imm] = reg1
j	“11000”	[op][imm]	pc = imm
jal	“11001”	[op][imm]	ra = pc, pc = imm
clrscr	“11010”	[op][*imm]	framebuffer[all] = 0

*Note: * means operand ignored*

Register Index	**Nickname**	**Size (bits)**	**Special Behavior**
0	$zero	16	Resets to 0 every clock tick
1	$pc	16	Program Counter
2	$ra	16	Return Address for JAL
3-31	$r3-$r31	16	

Instruction Type	**Opcode (4 downto 3)**	**Format**
R	“00” or “01”	[op][reg1][reg2][reg3]
I	“10”	[op][reg1][reg2][imm]
J	“11”	[op][imm]