```
<!-- Copyright 2015, Stacy Bridges -->
<!-- index.html -->
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">

<head>
    <!-- domain: mipsops -->
    <title>MIPS Operations</title>
    <link rel="stylesheet" type="text/css" href="css/styles.css">

    <script language="JavaScript1.2" type="text/javascript">

    /*********************************************************************
    *  function init()
    *  this function is used to reset all radio buttons upon page refresh
    *
    *********************************************************************/
    function init(){
        var oR = document.getElementsByName("opRadio");
          var mR = document.getElementsByName("modeRadio");
          var sR = document.getElementsByName("speedRadio");

        // check the first radio button by default
        oR[0].checked=true;
        mR[0].checked=true;
        sR[3].checked=true;
    }

    //  BEGIN SCRIPTS  ===================================================================

    // global variables  -------------------------------------------------------------
    var operationQflag = 0;                  // flag to toggle helpBox from 'show' to 'hide'
    var operandsQflag  = 0;                  // flag to toggle helpBox from 'show' to 'hide'
    var modeQflag      = 0;                  // flag to toggle helpBox from 'show' to 'hide'
    var speedQflag     = 0;                  // flag to toggle helpBox from 'show' to 'hide'
    var validFlag      = false;              // flag to indicate valid operands have been entered
    var simOperation   = "multiplication";   // operation value from SIM SETTINGS interface
    var simOp1         = 0;                  // operand1 value from SIM SETTINGS interface
    var simOp2         = 0;                  // operand2 value from SIM SETTINGS interface
    var simMode        = "continuous";       // mode value from SIM SETTINGS interface
    var simSpeed       = 4;                  // speed value from SIM SETTINGS interface
    var underFlag      = false;              // flag to toggle underLED
    var overFlag       = false;              // flag to toggle overLED
    var pArrow         = 0;                  // flag to toggle graphics groups
    var cArrow         = 0;                  // flag to toggle graphics groups
    var aluArrow       = 0;                  // flag to toggle graphics groups
    var rtArrow        = 0;                  // flag to toggle graphics groups
```

```
var graphicFlag    = 0;                  // flag to toggle graphics on and off
var count          = 0;                  // this counter allows pauses between SIM steps
var myVar;                               // object for setInterval() / clearInterval()
var binMultiplicand;                     // initial multiplicand register
var binMultiplier;                       // initial multiplier register
var binProduct;                          // initial product register
var newProduct;                          // product + multiplicand
var addFlag        = 0;                  // flag to toggle correct product to shift right
var startFlag      = 0;                  // flag to allow run/stop control over sim
var abacus         = 0;                  // 32-bit counter to control operations
var turboFlag      = 0;
var logMessage;
var logCount       = 0;
var logFlag        = 0;                  // flag to toggle log (show/hide)

// ------------------------------------------------------------------------------------
// VIEWER FUNCTIONS
// ------------------------------------------------------------------------------------

 /*********************************************************************
*  funtion toggleLog()
*  this function is used to 'show' or 'hide' the log feature
*
*********************************************************************/
function toggleLog(){
    if( logFlag == 0 ){
         document.getElementById("viewerDiv").style.visibility="visible";
        logFlag = 1;
      }
    else if( logFlag == 1 ){
        document.getElementById("viewerDiv").style.visibility="hidden";
        logFlag = 0;
    }
}


/*********************************************************************
*  function movViewer()
*  this function is used to move the viewer in the direction indicated
*  by the argument
*
*********************************************************************/
function movViewer(direction){
    if( direction == 0 ) vImgNum --;
    if( direction == 1 ) vImgNum ++;
```

```
    if( vImgNum < vAdder ) vImgNum = vImgArray.length - 1 + vAdder;
    if( vImgNum > vImgArray.length - 1 + vAdder ) vImgNum = vAdder;

    document.getElementById("viewerPic").src = vFolder + vImgNum + "fs.jpg";
    document.getElementById("viewerCaption").innerHTML = vAltTag[ vImgNum - vAdder ];
}

//  BEGIN FLOATING LAYER FUNCTIONS  ----------------------------------------------------
//  the base code for the floating layer was provided as a free-use tool from
//  by http://tools.seochat.com/tools/floating-layer/
isIE=document.all;
isNN=!document.all&&document.getElementById;
isN4=document.layers;
isActive=false;

function MoveInit(e){
    topOne=isIE ? "BODY" : "HTML";
    whichOne=isIE ? document.all.FloatingLayer : document.getElementById("viewerDiv");
    ActiveOne=isIE ? event.srcElement : e.target;

    while (ActiveOne.id!="moveZone"&&ActiveOne.tagName!=topOne){
        ActiveOne=isIE ? ActiveOne.parentElement : ActiveOne.parentNode;
    }
    if (ActiveOne.id=="moveZone"){
        offsetx=isIE ? event.clientX : e.clientX;
        offsety=isIE ? event.clientY : e.clientY;
        nowX=parseInt(whichOne.style.left);
        nowY=parseInt(whichOne.style.top);
        MoveEnabled=true;
        document.onmousemove=Move;
    }
}

function Move(e){
    if (!MoveEnabled) return;
    whichOne.style.left=isIE ? nowX+event.clientX-offsetx : nowX+e.clientX-offsetx;
    whichOne.style.top=isIE ? nowY+event.clientY-offsety : nowY+e.clientY-offsety;
    return false;
}

function MoveN4(whatOne){
    if (!isN4) return;
    N4=eval(whatOne);
    N4.captureEvents(Event.MOUSEDOWN|Event.MOUSEUP);
    N4.onmousedown=function(e){
        N4.captureEvents(Event.MOUSEMOVE);
```

```
        N4x=e.x;
        N4y=e.y;
    }
    N4.onmousemove=function(e){
        if (isActive){
            N4.moveBy(e.x-N4x,e.y-N4y);
            return false;
        }
    }
    N4.onmouseup=function(){
        N4.releaseEvents(Event.MOUSEMOVE);
    }
}

document.onmousedown=MoveInit;
document.onmouseup=Function("MoveEnabled=false");

//  END FLOATING LAYER FUNCTIONS  ----------------------------------------------


/**********************************************************************
*  function toggleHelp()
*  this function is used to turn the tool-tips on and off by means of a
*  switch statement, which determines 'off' or 'on' status based on two
*  indicators:
*          1) the name of the tool-tip as passed into the function
*          2) the current state of the corresponding flag ( 0 or 1)
*
**********************************************************************/
function toggleHelp(objQ){
    switch(objQ.name){
        case "operationQ":
            if( operationQflag == 0){
                document.getElementById("operationHelp").style.visibility="visible";
                document.getElementById("q1").innerHTML=" X ";
                operationQflag=1;
            }
            else{
                document.getElementById("operationHelp").style.visibility="hidden";
                document.getElementById("q1").innerHTML=" ? ";
            operationQflag=0;
            }
            break;
        case "operandsQ":
            if( operandsQflag == 0){
                document.getElementById("operandsHelp").style.visibility="visible";
```

```
                document.getElementById("q2").innerHTML=" X ";
                operandsQflag=1;
            }
            else{
                document.getElementById("operandsHelp").style.visibility="hidden";
                document.getElementById("q2").innerHTML=" ? ";
                operandsQflag=0;
            }
            break;
        case "modeQ":
            if( modeQflag == 0){
                document.getElementById("modeHelp").style.visibility="visible";
                document.getElementById("q3").innerHTML=" X ";
                modeQflag=1;
            }
            else{
                document.getElementById("modeHelp").style.visibility="hidden";
                document.getElementById("q3").innerHTML=" ? ";
                modeQflag=0;
            }
            break;
        case "speedQ":
            if( speedQflag == 0){
                document.getElementById("speedHelp").style.visibility="visible";
                document.getElementById("q4").innerHTML=" X ";
                speedQflag=1;
            }
            else{
                document.getElementById("speedHelp").style.visibility="hidden";
                document.getElementById("q4").innerHTML=" ? ";
                speedQflag=0;
            }
            break;
    }
}// end toggleHelp()


/*********************************************************************
*  function validate()
*  this function is used to validate the operands submitted by the user
*  in the SIM SETTINGS panel;
*  the validations include:
*          - the operands are not NULL
*          - the operands are not negative
*          - the operands are integers
*          - the operands do not contain text or special chars
```

```
*          - the operands are not floating point values
*            (note: a floating val is valid if all decimal points are 0)
*
**********************************************************************/
function validate(){
    var message = "";
    validFlag = true;

    //  get the operand values from the SIM SETTINGS text-fields
    var simOp1 = document.getElementById("operand1").value;  // multiplicand, dividend
    var simOp2 = document.getElementById("operand2").value;  // multiplier, divisor

    //  check that the operands are not empty
    if( simOp2 == "" && simOp1 == "" ){
        validFlag=false;
        message = message + "The operands are empty. ";
    }
    else{
        if( simOp1 == "" && simOp2 != "" ){
            validFlag=false;
            message = "The first operand is empty. ";
        }
        if( simOp2 == "" && simOp1 != "" ){
            validFlag=false;
            message = "The second operand is empty. ";
            }
        }

    //  check that operands are not fractions (ie, operand%1) or text (ie, isNaN(operand))
    if( ( eval(simOp1%1) != 0 && eval(simOp2%1) != 0) || ( isNaN(simOp1) && isNaN(simOp2) ) ){
        validFlag=false;
        message = message + "The operands are not integers. ";
    }
    else{
        if( eval(simOp1%1) != 0 || isNaN(simOp1) ){
            validFlag=false;
            message = message + "The first operand is not an integer. ";
        }
        if( eval(simOp2%1) != 0 || isNaN(simOp2)){
            validFlag=false;
            message = message + "The second operand is not an integer. ";                                    }
    }

    //  check that operands are not negative
    if( simOp1 < 0 && simOp2 < 0 ){
        validFlag=false;
```

```
        message = message + "The operands are negative. ";
    }
    else{
        if( simOp1 < 0 ){
            validFlag=false;
            message = message + "The first operand is negative.";
        }
        if( simOp2 < 0 ){
            validFlag=false;
            message = message + "The second operand is negative.";
        }
    }

    //  display validation results
    if( validFlag ){
        document.getElementById("validLED").style.backgroundColor="green";
        document.getElementById("validationMessage").style.color="green";
        document.getElementById("validationMessage").innerHTML="Validation successful.<br>Run when
                                                                ready.";
    }
    else{
        document.getElementById("validLED").style.backgroundColor="red";
        document.getElementById("validationMessage").style.color="red
        document.getElementById("validationMessage").innerHTML=message;
    }
}// end validate()


function fieldClick(field, code){
    //  update validation message
    document.getElementById("validationMessage").style.color="brown";
    document.getElementById("validationMessage").innerHTML="Awaiting valid operands.";

    //  if user clicked operand1 field, then reset it
    if( field.id == "operand1" ){
        document.getElementById("operand1").value = "";
        document.getElementById("validLED").style.backgroundColor = "rgb(250,250,150)";
        field1Flag = 0;
    }// end if

    //  if user clicked operand2 field, then reset it
    else{
        document.getElementById("operand2").value = "";
        document.getElementById("validLED").style.backgroundColor = "rgb(250,250,150)";

        field2Flag = 0;
```

```
    }// end else
}// end fieldClick()

function changeFieldColor(field, code){
    if( (field.id == "operand1" && field1Flag == 0) || (field.id == "operand2" && field2Flag == 0) ){
        if( code == 1 ){
            // onmouseover behavior
            document.getElementById(field.id).style.backgroundColor="white";
        }
        else if ( code == 0 ){
            // onmouseout behavior

            document.getElementById(field.id).style.backgroundColor="rgb(255,255,230)";
   }
    }
}// end changeFieldColor


function clearSettings(){
    document.getElementById("operand1").value = "";
    document.getElementById("operand2").value = "";
    document.getElementById("operand1").style.backgroundColor="rgb(255,255,230)"; // light yellow
    document.getElementById("operand2").style.backgroundColor="rgb(255,255,230)"; // light yellow
    document.getElementById("validLED").style.backgroundColor="rgb(250,250,150)"; // amber
    document.getElementById("validationMessage").style.color="brown";
    document.getElementById("validationMessage").innerHTML="Awaiting valid operands.";
}// end clearSetttings()


function getSimSettings(){
    // get the operation value
    var i; // array index
    var valOperation = document.getElementsByName("opRadio");
    for( i = 0; i < valOperation.length; i++ ){
        if( valOperation[i].checked ){
            simOperation = valOperation[i].value;
        }
    }

    // get the operand values
    simOp1 = document.getElementById("operand1").value;  // multiplicand, dividend
    simOp2 = document.getElementById("operand2").value;  // multiplier, divisor

    // get the mode value
    var valMode = document.getElementsByName("modeRadio");
    for( i = 0; i < valMode.length; i++ ){
```

```
        if( valMode[i].checked ){
            simMode = valMode[i].value;
        }
    }

    // get the speed value
    var valSpeed = document.getElementsByName("speedRadio");
    for( i = 0; i < valSpeed.length; i++ ){
        if( valSpeed[i].checked ){
            simSpeed = valSpeed[i].value;
        }
    }
}// end getSimSettings()


function startSim(){
    if( startFlag == 0){
        // don't start the SIM unless the operands have passed validation
        validate();

        if( !validFlag ){
            alert("The SIM cannot be run due to invalid operands.");
            return;
        }

        // fetch the values from the SIM SETTINGS interface
        getSimSettings();

        // convert operands to binary
        document.getElementById("messageBox").innerHTML="converting the operands to binary. . . ";
        binMultiplicand = convertToBinary(simOp1);   // convert multiplicand
        binMultiplier   = convertToBinary(simOp2);   // convert multiplier
        binProduct      = convertToBinary(0);        // convert product

        // set the speed of the SIM based on speedRadio value
        if( turboFlag == 0 ){
            speed = Number(simSpeed * 200);
        }
        else if (turboFlag == 1){
            speed = Number(simSpeed * 10);
        }

        // clear the log
        for(var i = 0; i <= 65; i++){
            document.getElementById("log"+i).innerHTML="";
```

```
            // clear all counters
            count = 0;
            addFlag = 0;
            abacus = 0;
            logCount = 0;
            logFlag = 0;
        }// end for
    }// end if

    document.getElementById("simStateMessage").innerHTML="the sim is running. . . ";

    // run the SIM on a timed interval basis
    myVar = setInterval( runSim, speed );
}// end startSim()


function turboDrive(){
    if( turboFlag == 0 ){
        turboFlag = 1;
        document.getElementById("x10button").style.backgroundColor="green";
        document.getElementById("x10button").style.color="rgb(232,200,160)";
    }
    else if( turboFlag == 1 ){
        turboFlag = 0;
        document.getElementById("x10button").style.backgroundColor="transparent";
        document.getElementById("x10button").style.color="brown";
    }
}// end turboDrive()


function stopSim(x){
    if( count > 0 ){ // if count == 0, the sim has not been started yet, so
                     // no need to run stop instructions
        clearInterval(myVar);
        startFlag = x;
        if( startFlag == 1){
            document.getElementById("simStateMessage").innerHTML="the sim is stopped. . . ";
        }
        if( startFlag == 0 ){
            document.getElementById("simStateMessage").innerHTML="the sim run is finished. ";
        }
    }
}//end stopSim()


function resetSIM(){
```

```
    location.reload(true);
}// end resetSIM()


function runSim(){
    if( simOperation == "multiplication" ) {
        // ================================================================  MULTIPLY
        // WAIT ******************
        if( count == 0 ){
        // load the operands into the registers
            document.getElementById("messageBox").innerHTML="initializing registers. . . ";
            document.getElementById("controlMessage").innerHTML="initialize registers";
        }

        // WAIT ******************
        if( count == 1 ){
            document.getElementById("multiplicandReg").style.color="orange";
            document.getElementById("multiplierReg").style.color="orange";
            document.getElementById("productReg").style.color="orange";
            document.getElementById("multiplicandReg").style.backgroundColor="white";
            document.getElementById("multiplierReg").style.backgroundColor="white";
            document.getElementById("productReg").style.backGroundcolor="white";

        }

        // WAIT ******************
        if( count == 2 ){
            document.getElementById("multiplicandReg").innerHTML=binMultiplicand;
            document.getElementById("multiplierReg").innerHTML=binMultiplier;
            document.getElementById("productReg").innerHTML=binProduct;

            // write to the log
            document.getElementById("log" + logCount).innerHTML = "start    " +" "
                                                                  " + binMultiplicand + "  " +
                                                                   binProduct +
                                                                   "  " + binMultiplier;
            logCount++;
        }

        // WAIT ******************
        if( count == 3 ){
            document.getElementById("multiplicandReg").style.color="brown";
            document.getElementById("multiplierReg").style.color="brown";
            document.getElementById("productReg").style.color="brown";
            document.getElementById("multiplicandReg").style.backgroundColor="rgb(200,255,255)";
            document.getElementById("multiplierReg").style.backgroundColor="rgb(200,255,255)";
```

```
    document.getElementById("productReg").style.backGroundcolor="rgb(200,255,255)";
}

// REM: 32-Cycle Multiplication starts here  ---------------------------------

// WAIT ******************
if( count == 4 ){
    // compare lsb to 1  -------------------------------------
    // show messages and graphics
    abacus++; // the abacus will be incremented 32 times (once for each bit in the register)
    document.getElementById("messageBox").innerHTML="comparing the multiplier lsb to 1. . . ";
    document.getElementById("controlMessage").innerHTML="compare lsb to 1";
}

// WAIT ******************
if( count == 5 ){
    document.getElementById("aluLeftInput").style.visibility="visible";
    document.getElementById("aluLeftInput").innerHTML="  1  ";
    document.getElementById("aluLeftInput").style.border="1px solid brown";
    document.getElementById("aluRightInput").style.visibility="visible";
    document.getElementById("aluRightInput").innerHTML="  lsb  ";
    document.getElementById("aluRightInput").style.border="1px solid brown";
    document.getElementById("lsbArrow1").style.visibility="visible";
    document.getElementById("pArrow11").style.visibility="visible";

    var g = document.getElementsByName("plierLsb");
    for( var i = 0; i < g.length; i++ ){
        g[i].style.visibility="visible";
    }
}

// WAIT ******************
if( count == 6 && binMultiplier[31] == 0 ){
    // if false, no add
    document.getElementById("aluOut0").style.visibility="visible";
    document.getElementById("trueFalse").style.visibility="visible";
    document.getElementById("trueFalse").style.textAlign="right";

    document.getElementById("trueFalse").innerHTML="     
                                                         FALSE";
    addFlag = 0;
}

// WAIT ******************
if( count == 7 && binMultiplier[31] == 0 ){
    // 'lsb is not 1, so no add'
```

```
    document.getElementById("sumBitString").style.visibility="visible";

    document.getElementById("sumBitString").innerHTML="     
                                                        lsb is not 1, so 'no add'";                }

// WAIT ******************
if( count == 8 && binMultiplier[31] == 0 ){
    // show controlBox arrow
    document.getElementById("messageBox").innerHTML="no add ";
    document.getElementById("controlMessage").innerHTML="  no add";
    var g = document.getElementsByName("cArrow");
    for( var i = 0; i < g.length; i++ ){
        g[i].style.visibility="visible";
    }

    // write to the log
    document.getElementById("log" + logCount).innerHTML = "no add   " + " " +
                                                           binMultiplicand + "  " +
                                                           binProduct + "  " +
                                                           binMultiplier;
    logCount++;
}

// WAIT ******************
if( count == 9 && binMultiplier[31] == 0 ){
    // hide graphics
    document.getElementById("aluLeftInput").style.visibility="hidden";
    document.getElementById("aluRightInput").style.visibility="hidden";
    document.getElementById("lsbArrow0").style.visibility="hidden";
    document.getElementById("lsbArrow1").style.visibility="hidden";
    document.getElementById("pArrow11").style.visibility="hidden";
    document.getElementById("aluOut0").style.visibility="hidden";
    document.getElementById("trueFalse").style.visibility="hidden";
    document.getElementById("sumBitString").style.visibility="hidden";

    var g = document.getElementsByName("cArrow");

    for( var i = 0; i < g.length; i++ ){
        g[i].style.visibility="hidden";
    }

    g = document.getElementsByName("plierLsb");
    for( var i = 0; i < g.length; i++ ){
        g[i].style.visibility="hidden";
    }
```

```
            count = 18; // this count increase allows a jump to SHIFT RIGHT
        }// end if statements for 'if false, no add'

        // WAIT  ******************
        if( count == 6 && binMultiplier[31] == 1 ){
            // if true, add
            document.getElementById("aluOut0").style.visibility="visible";
            document.getElementById("trueFalse").style.visibility="visible";
            document.getElementById("trueFalse").style.textAlign="right";
            document.getElementById("trueFalse").innerHTML="     
                                                                 TRUE";
            addFlag = 1;
        }

        // WAIT  ******************
        if( count == 7 && binMultiplier[31] == 1 ){
            // 'lsb is 1, so add'
            document.getElementById("sumBitString").style.visibility="visible";
document.getElementById("sumBitString").innerHTML="     
                                                       lsb is 1, so 'add'";          }

        // WAIT ******************
        if( count == 8 && binMultiplier[31] == 1 ){
            // show controlBox arrow
            document.getElementById("messageBox").innerHTML="add";
            document.getElementById("controlMessage").innerHTML="  add";

            var g = document.getElementsByName("cArrow");
            for( var i = 0; i < g.length; i++ ){
                g[i].style.visibility="visible";
            }
        }

        // WAIT ******************
        if( count == 9 && binMultiplier[31] == 1 ){
            // hide graphics
            document.getElementById("aluLeftInput").style.visibility="hidden";
            document.getElementById("aluRightInput").style.visibility="hidden";
            document.getElementById("lsbArrow0").style.visibility="hidden";
            document.getElementById("lsbArrow1").style.visibility="hidden";
            document.getElementById("pArrow11").style.visibility="hidden";
            document.getElementById("aluOut0").style.visibility="hidden";
            document.getElementById("trueFalse").style.visibility="hidden";
            document.getElementById("sumBitString").style.visibility="hidden";
```

```
    var g = document.getElementsByName("cArrow");

    for( var i = 0; i < g.length; i++ ){
        g[i].style.visibility="hidden";
    }

    g = document.getElementsByName("plierLsb");
    for( var i = 0; i < g.length; i++ ){
        g[i].style.visibility="hidden";
    }
}

// WAIT ******************
if( count == 10 && binMultiplier[31] == 1  ){
    // convert and add multiplicand and product strings
    newProduct = convertAndAdd(binProduct, binMultiplicand);

    // show messages and graphics
    document.getElementById("controlMessage").innerHTML = "add multiplicand to product";




    // write to the log
    document.getElementById("log" + logCount).innerHTML = "add      "
                                                          + " " + binMultiplicand +
                                                           " + newProduct + "  " +
                                                          binMultiplier;
logCount++;                                                                                    }

// WAIT ******************
if( count == 11 && binMultiplier[31] == 1 ){
    document.getElementById("aluLeftInput").style.visibility="visible";
    document.getElementById("aluLeftInput").innerHTML="  product  ";
    document.getElementById("aluLeftInput").style.border="1px solid brown";
    document.getElementById("aluRightInput").style.visibility="visible";
    document.getElementById("aluRightInput").innerHTML="multiplicand";
    document.getElementById("aluRightInput").style.border="1px solid brown";
    document.getElementById("lsbArrow0").style.visibility="visible";
    document.getElementById("lsbArrow1").style.visibility="visible";
    graphicToggle("pArrow");
}

// WAIT ******************
if( count == 12 && binMultiplier[31] == 1 ){
```

```
    // show binary addition inputs
    document.getElementById("aluOut0").style.visibility="visible";
    document.getElementById("trueFalse").style.visibility="visible";
    document.getElementById("trueFalse").innerHTML=binProduct;
    document.getElementById("addBitString").style.visibility="visible";
    document.getElementById("addBitString").innerHTML=binMultiplicand;
}

// WAIT ******************
if( count == 13 && binMultiplier[31] == 1 ){
    // show binary addition result
    document.getElementById("sumBitString").style.visibility="visible";
    document.getElementById("sumBitString").innerHTML=newProduct;
}

// WAIT ******************
if( count == 14 && binMultiplier[31] == 1 ){
    // load result into product register -> 1st, turn off graphics
    document.getElementById("controlMessage").innerHTML="load result to product register";
document.getElementById("aluOut0").style.visibility="hidden";
    document.getElementById("trueFalse").style.visibility="hidden";
    document.getElementById("addBitString").style.visibility="hidden";
    document.getElementById("aluLeftInput").style.visibility="hidden";
    document.getElementById("aluRightInput").style.visibility="hidden";
    graphicToggle("lsbArrow");
    graphicToggle("pArrow");
    graphicToggle("pArrow");
}

// WAIT ******************
if( count == 15 && binMultiplier[31] == 1 ){
    document.getElementById("aluOut1").style.visibility="visible";
}

// WAIT ******************
if( count == 16 && binMultiplier[31] == 1 ){
    // load result into product register -> 2nd, load new value into register
    document.getElementById("productReg").style.color="red";
    document.getElementById("productReg").style.backgroundColor="white";
}

// WAIT ******************
if( count == 17 && binMultiplier[31] == 1 ){
    document.getElementById("productReg").innerHTML=newProduct;
}
```

```
// WAIT ******************
if( count == 18 && binMultiplier[31] == 1 ){
    // turn off graphics and cBox message
    document.getElementById("sumBitString").style.visibility="hidden";
    document.getElementById("aluOut1").style.visibility="hidden";
    document.getElementById("productReg").style.color="brown";
    document.getElementById("productReg").style.backgroundColor="rgb(200,255,255)";
    document.getElementById("controlMessage").innerHTML="";

}// end if statements for 'if true, add'

// WAIT ****************** SHIFT RIGHT
if( count == 19 ){
    // shift registers to the right -> 1st, show graphics
    document.getElementById("controlMessage").innerHTML="shift right";
    graphicToggle("rtArrow");
}

// WAIT ******************
if( count == 20 ){
    // shift registers to the right -> 2nd, compute new reg values and load to regs
    document.getElementById("productReg").style.color="red";
    document.getElementById("productReg").style.backgroundColor="white";
    document.getElementById("multiplierReg").style.color="red";
    document.getElementById("multiplierReg").style.backgroundColor="white";
}

// WAIT ******************
if( count == 21 ){
    if( addFlag == 1 ){
        shiftRegsRight(newProduct, binMultiplier);
    }
    else if ( addFlag == 0 ){
        shiftRegsRight(binProduct, binMultiplier);
    }

    // write to the log
    document.getElementById("log" + logCount).innerHTML = "shift    " + "
                                                          " + binMultiplicand +
                                                          "  " + binProduct + "  " +
                                                          binMultiplier;
    logCount++;

}

// WAIT ******************
```

```
        if( count == 22 ){
            // restore screen to starting position
            graphicToggle("rtArrow");
            document.getElementById("productReg").style.color="brown";
            document.getElementById("productReg").style.backgroundColor="rgb(200,255,255)";
            document.getElementById("multiplierReg").style.color="brown";
            document.getElementById("multiplierReg").style.backgroundColor="rgb(200,255,255)";
            document.getElementById("controlMessage").innerHTML="";
        }

        count++;

    }// end multiplication

    if( count == 23 && abacus < 32 ){
        // reset wait count for next loop
        count = 4;
    }

    if( count == 23 && abacus == 32){
        // end operation loop
        document.getElementById("messageBox").innerHTML="operation complete!"
        count = 0;
        abacus = 0;
        stopSim(0);
    }

    // ========================================================================  DIVIDE
    if( simOperation == "division" ){
        alert("divide");
        stopSim();
    }
}// end runSim()


function shiftRegsRight(prod, mult){
    // convert the string arguments into array
    var prodReg = [];
    var multReg = [];

    for( var i = 0; i < 32; i++ ){ // populate array with string values
        var x = i+1;
        prodReg[i] = parseInt( prod.substring(i,x) );
        multReg[i] = parseInt( mult.substring(i,x) );
    }
```

```
    // shift the multiplier register
    for( var i = 31; i >= 0; i-- ){
        var h = i - 1;
        if( i == 0 ){
            multReg[i] = prodReg[31];
        }
        else{
            multReg[i] = multReg[h];
        }
    }

    // shift the product register
    for( var i = 31; i >= 0; i-- ){
        var h = i - 1;
        if( i == 0 ){
            prodReg[i] = 0;
        }
        else{
            prodReg[i] = prodReg[h];
        }
    }

    var nuProd = prodReg.join("");
    var nuMult = multReg.join("");

    document.getElementById("productReg").innerHTML=nuProd;
    document.getElementById("multiplierReg").innerHTML=nuMult;

    binProduct = nuProd;
    binMultiplier = nuMult;
}// end shiftRegsRight()


function convertAndAdd(prod, cand){
    // convert the string arguments into arrays
    var prodArray = [];   // create new prodarray

    var candArray = [];   // create new prodarray
    var nuProd = []

    for( var i = 0; i < 32; i++ ){ // populate array with string values
        var x = i+1;
        prodArray[i] = parseInt( prod.substring(i,x) );
        candArray[i] = parseInt( cand.substring(i,x) );
        nuProd[i] = 0;
    }
```

```
    // add the arrays together
    nuProd = binaryAddition(prodArray, candArray);

    // convert the new product array into a string and return it to the caller
    nuProd = nuProd.join("");

    return nuProd;
}// end convertAndAdd()


function convertToBinary(num){
    // set a flag to indicate if argument will need two's complement conversion
    var signFlag = false;
    if( num < 0 ){
        signFlag = true;
        num = Math.abs(num);
    }

    // create an array to hold the binary version of the operand
    var binArray = [];

    // initialize the array with 0 values
    for( var i = 0; i < 32; i++ ){
        binArray[i] = 0;
    }// end for loop

    // convert the passed-in argument to a bit string and store in array
    var dividend = num;
    for( var i = 31; i >= 0; i-- ){
        if( dividend == 0 ){
            binArray[i] = 0;
        }
        else{
            binArray[i] = ( dividend | 0 ) % 2;
            dividend    = ( dividend / 2 ) | 0;
        }
    }// end for loop

    // do two's complement conversion if needed
    if( signFlag == true ){
        var bitString = convertToTwos(binArray);
        return bitString;
    }

    // else, convert the unsigned array to a string and return it to the caller
```

```
    else{
        var bitString = binArray.join("");
        return bitString;
    }
}// end convertToBinary()

function convertToTwos(binArray){
    // flip the bits
    for( var i = 0; i < 32; i++ ){
        if( binArray[i] == 0 ){
            binArray[i] = 1;
        }
        else{
            binArray[i] = 0;
        }
    }// end for

    // add one
    // first, create an array to hold binary value of 1
    var oneArray = [];
    for( var i = 0; i < 32; i ++ ){
        if( i == 31 ){
            oneArray[i] = 1;
        }
        else{
            oneArray[i] = 0;
        }
        }// end for

    // next, pass the binArray and the oneArray to binaryAddition()
    binArray = binaryAddition(binArray, oneArray);

    // next, convert the two's complement number to a bit string and return to caller
    var bitString = binArray.join("");
    return bitString;
}// end convertToTwos


function binaryAddition(binArray, oneArray){
    var carryBit = 0;
    var nuBin = [];
    var nuOne = [];

    // convert array contents to integers
    for( var i = 0; i < 32; i ++ ){
        nuBin[i] = parseInt(binArray[i]);
```

```
        nuOne[i] = parseInt(oneArray[i]);
    }

    // perform addition
    for( var i = 31; i >= 0; i-- ){
        var h = Number(nuBin[i] + nuOne[i] + carryBit);
        if( h == 0 ){
            nuBin[i] = 0;
            carryBit = 0;
        }
        if( h == 1 ){
            nuBin[i] = 1;
            carryBit = 0;
        }
        if( h == 2 ){
            nuBin[i] = 0;
            carryBit = 1;
        }
        if( h == 3 ){
            nuBin[i] = 1;
            carryBit = 1;
        }

    }// end for

    // set underFlag if carryBit == 1
    if( carryBit == 1 ){
        underFlag = true;
    }

    // return the converted array to caller
    return nuBin;
}// end binaryAddition()


function showLog(){
    alert("log");
}// end showLog()


function changeOperandLabels(code){
    if( code == 0 ){
        document.getElementById("op1_label").innerHTML="multiplicand:";
        document.getElementById("op2_label").innerHTML="multiplier:";
    }
    else{
```

```
            document.getElementById("op1_label").innerHTML="dividend:";
            document.getElementById("op2_label").innerHTML="divisor:";
            alert("The division operation is disconnected. The operation will be set to 'multiplication'");
            document.getElementById("multRadio").checked="yes";
            document.getElementById("op1_label").innerHTML="multiplicand:";
            document.getElementById("op2_label").innerHTML="multiplier:";
        }
    }// end changeOperandLabels()


    function modeClick(code){
        if( code == 1 ){// continuous behavior
            document.getElementById("pauseButton").style.visibility="visible";
            document.getElementById("resumeButton").style.visibility="visible";
        }
        else if ( code == 0 ){ // step behavior
            document.getElementById("pauseButton").style.visibility="hidden";
            document.getElementById("resumeButton").style.visibility="hidden";
        }
    }// end modeClick()


    function graphicToggle(a){
        if( graphicFlag == 1){
            var g = document.getElementsByName(a);
            for( var i = 0; i < g.length; i++ ){
                g[i].style.visibility="hidden";
            }
            graphicFlag = 0;
        }
        else if( graphicFlag == 0){
            var g = document.getElementsByName(a);
            for( var i = 0; i < g.length; i++ ){
                g[i].style.visibility="visible";
            }
            graphicFlag = 1;
        }
    }// end graphicToggle()

    // END SCRIPTS  ================================================================
    </script>
</head>

<body onload="init();">
    <header>
        MIPS Ops
```

```
    <a href="docs/htmlCode.pdf" target="_blank" id="htmlCode" class="deliverables" title="deliverable">
        <img src="images/pdfIconSmall.png" style="position:relative;top:6px;">
        html/js
    </a>

    <a href="docs/cssCode.pdf" target="_blank" id="cssCode" class="deliverables" title="deliverable">
        <img src="images/pdfIconSmall.png" style="position:relative;top:6px;">
        css
    </a>
    <a href="docs/mipsOpsUserGuide.pdf" target="_blank" class="deliverables" id="userGuide"
     title="deliverable">
        <img src="images/pdfIconSmall.png" style="position:relative;top:6px;">
        user guide
    </a>
    <div style="width:1138px;height:4px;border-bottom:1px solid brown;
                margin-top:20px;margin-bottom:20px;"></div>
</header>

<!-- begin user front end ////////////////////////////////////////////////// -->
<div id="mainDiv">
    <table id="mainTable" cellspacing="0" cellpadding="0">
        <tr>
            <td style="font-weight:bold;"> SIM SETTINGS:</td>
            <td colspan="16" style="height:20px;font-weight:bold;"> SIM VIEWER:</td>
            <td></td>
        </tr>
        <tr>
            <td colspan="17" style="border-bottom:1px solid brown;height:10px;"></td>
            <td></td>
        </tr>
        <tr>

            <td rowspan="20" valign="top" style="background-image:url(images/simSettingsBG3.jpg);
                                                 border-left:1px solid brown;
                                                 border-right:1px solid brown;">
                <!-- start simSettings//////////////////////////////////////////////////// -->

                <div id="simSettings" style="font-size:1.2em;">
                    <table id="simSettingsTable" style="margin-left:8px;color:brown;
                                                        font-size:.75em;color:brown;width:232px;">
                        <tr>
                            <td>
                                <br>
                                <a style="font-weight:bold;">operation:</a> 
                                <a class="help" style="font-weight:bold;margin-left:80px;"
                                                 onclick="toggleHelp(this)"
```

```
            name="operationQ" id="q1" title="info"> ? </a>
        <br><br> 
        <a class="radio">
            <input type="radio" class="buttons" name="opRadio"
             value="multiplication" checked="yes"
             onclick="changeOperandLabels(0)" id="multRadio">
            multiplication
        </a>
        <br> 
        <a class="radio">
            <input type="radio" class="buttons" name="opRadio"
             value="division" onclick="changeOperandLabels(1)">
            division
        </a>
        <br><br>
        <hr>
    </td>
</tr>
<tr>
    <td align="left">
        <a style="font-weight:bold;">
            operands:
        </a> 

        <form>
            <fieldset style="border:0px;">

                <a class="help" style="font-weight:bold;margin-left:80px;
                                       position:relative;left:98px;top:-26px;"
                   onclick="toggleHelp(this)" name="operandsQ" id="q2"
                   title="info">
                     ? </a>
                <br>

                <article>
                    <a id="op1_label" style="font-style:italic;
                                             font-size:.95em;">
                        multiplicand:
                    </a>

                    <br>
                    <input type="text" name="first" value="" id="operand1"
                     autocomplete = "off" class="textField"
                           onmouseout="changeFieldColor(this, 0)"
                           onmouseover="changeFieldColor(this, 1)"
                           onclick="fieldClick(this, 1)">
```

```
                    <br><br>
                    <a id="op2_label" style="font-style:italic;
                     font-size:.95em;">
                        multiplier:
                    </a>

                    <br>
                    <input type="text" name="second" value="" id="operand2"
                     autocomplete = "off" class="textField"
                     onmouseout="changeFieldColor(this, 0)"
                     onmouseover="changeFieldColor(this, 1)"
                     onclick="fieldClick(this, 1)">

                    <br><br>
                </article>

                <article id="validationMessage">
                    Awaiting valid operands.
                </article>
            </fieldset>
        </form>
    </td>
</tr>
<tr>
    <td align="center">
        <article align="center">
            <button class="buttons" onclick="validate()">validate</button>
            <a style="background-color:rgb(250,250,150);
                      border:1px brown;" id="validLED"> </a>

            <button class="buttons"
             onclick="clearSettings()">  clear  </button>

        </article>
        <br>
        <hr>
    </td>
</tr>
<tr>
    <td>
        <a style="font-weight:bold;">mode:</a>
        <a class="help" style="font-weight:bold;margin-left:122px;"
         onclick="toggleHelp(this)" name="modeQ" id="q3" title="info">
```

```
             ? </a>


        <br><br> 
        <a class="radio">
         <input type="radio" class="buttons" value="continuous"
             name="modeRadio" checked="yes"
             onclick="modeClick(0)">
             continuous
        </a>
        <br><br>
        <hr>
    </td>
</tr>
<tr>
    <td>
        <a style="font-weight:bold;">
            speed:</a> 
        <a class="help" style="font-weight:bold;margin-left:112px;"
           onclick="toggleHelp(this)" name="speedQ" id="q4" title="info">
             ? </a>


        <br><br>
        <a id="disabledMessage"></a>
        <div id="speedDiv" style="visibility:visible;">



             
            <a class="radio">
                <input type="radio" name="speedRadio" value="9" title="25%">
            </a>

                


            <a class="radio">
                <input type="radio" name="speedRadio" value="6" title="50%">
            </a>

                


            <a class="radio">
                <input type="radio" name="speedRadio" value="3" title="75%">
```

```
                                </a>

                                    


                                <a class="radio">
                                    <input type="radio" name="speedRadio" value="1" title="100%"
                                     checked="yes">
                                </a>

                                <br>
                                <a style="font-style:italic;font-size:.95em;"> slow
                                </a>

                                           
                                          
                                <a style="font-style:italic;font-size:.95em;">
                                    fast
                                </a>
                                <br>
                                <div id="x10div" style="font-size:.75em;text-align:center;">
                                    <button value="10" class="buttons"
                                     id="x10button"onclick="turboDrive()">
                                        x10
                                    </button>
                                </div>
                            </div>
                        </td>
                    </tr>
                </table><!-- end simSettings table -->
            </div><!-- end simSettings div -->
        </td>
    </tr>
    <!-- start simViewer////////////////////////////////////////////////////// -->

    <tr>
        <td rowspan="2"></td>
        <td colspan="4" class="label"><br></td>
        <td colspan="4"></td>
        <td colspan="6"></td>
        <td></td>
        <td rowspan="19" style="border-left:1px solid brown"></td>

    </tr>
    <tr>
        <td colspan="8" id="messageBox" title="this box displays the current sim step">idle</td>
```

```
    <td colspan="7" class="label" align="center">
        <ul id="navMenu" style="font-style:normal;font-size:1.0em;">
            <li title="run the sim" onclick="startSim()" id="runOrStep">run</li>
            <li title="stop the sim" onclick="stopSim(1)" id="stop">stop</li>
            <li title="view log of current sim run" onclick="toggleLog()">log</li>
            <li title="reset the SIM" onclick="resetSIM()">reset</li>
        </ul>

    </td>
</tr>
<tr>
    <td colspan="16" style="height:20px;border-bottom:1px solid brown;"></td>
</tr>
<tr>
    <td colspan="16" style="height:20px;"></td>
</tr>
<tr>
    <td rowspan="2"></td>
    <td colspan="7" class="label">multiplicand register (32 bits):</td>
    <td rowspan="2"></td>
    <td colspan="6" rowspan="2" id="simStateMessage" valign="bottom"
     style="font-style:italic;">
        <a>the sim is idle. . . </a>
    </td>
    <td rowspan="2"></td>
</tr>
<tr>
    <td colspan="7" class="register">
        <a id="multiplicandReg">00000000000000000000000000000000</a>
    </td>
</tr>
<tr>
    <td><img src="images/arrowElbow0.jpg" id="pArrow8" name="pArrow" class="graphics"></td>
    <td colspan="2" align="center"><img src="images/arrowHorz.jpg"  id="pArrow9" name="pArrow"
     class="graphics"></td>
    <td><img src="images/arrowElbow1.jpg" id="pArrow10" name="pArrow" class="graphics"></td>
    <td></td>
    <td></td>
    <td></td>
    <td><img src="images/arrowShft0.jpg" id="lsbArrow0" name="lsbArrow" class="graphics"></td>
    <td></td>
    <td></td>
    <td colspan="5"></td>
    <td></td>
</tr>
<tr>
```

```
    <td><img src="images/arrowShft1.jpg" id="pArrow7" name="pArrow" class="graphics"></td>
    <td></td>
    <td colspan="3" align="center"><a id="aluLeftInput" class="aluInputs">product</a></td>
    <td></td>
    <td colspan="3" align="center"><a id="aluRightInput"
     class="aluInputs">multiplicand</a></td>
    <td><img src="images/arrowHorz9bb.jpg" name="plierLsb" class="graphics"></td>
    <td colspan="2"><img src="images/arrowHorz9a.jpg" name="plierLsb" class="graphics"></td>
    <td><img src="images/arrowHorz9b.jpg" name="plierLsb" class="graphics"></td>
    <td colspan="2"><img src="images/arrowHorz9a.jpg" name="plierLsb" class="graphics"></td>

    <td align="bottom"><img src="images/arrowElbow0b.jpg" name="plierLsb"
     class="graphics"></td>
</tr>
<tr>
    <td><img src="images/arrowShft0.jpg" id="pArrow6" name="pArrow" class="graphics"></td>
    <td></td>
    <td></td>
    <td valign="bottom"><img src="images/arrowDown.jpg" id="pArrow11" name="pArrow"
     class="graphics"></td>
    <td></td>
    <td align="center" class="label" valign="bottom">alu:</td>
    <td></td>
    <td valign="bottom"><img src="images/arrowDown.jpg" id="lsbArrow1" name="lsbArrow"
     class="graphics"></td>
    <td></td>
    <td colspan="6" class="label" valign="bottom">
        <a style="margin-left:32px;">control:</a>
    </td>
    <td><img src="images/arrowShft0.jpg" name="plierLsb" class="graphics"></td>
</tr>
<tr>
    <td><img src="images/arrowShft2.jpg" id="pArrow5" name="pArrow" class="graphics"></td>
    <td></td>
    <td colspan="7" align="center" class="label">
        <img src="images/alu.jpg" style="margin-left:-14px;">
    </td>
    <td colspan="6"><a id="controlBox"><p id="controlMessage"></p></a></td>
    <td><img src="images/arrowShft2.jpg" name="plierLsb" class="graphics"></td>
</tr>
<tr>
    <td><img src="images/arrowShft0.jpg" id="pArrow4" name="pArrow" class="graphics"></td>
    <td colspan="4"></td>
    <td valign="top" align="center"><img src="images/arrowDown.jpg" id="aluOut0"
     name="aluArrow" class="graphics"></td>
    <td colspan="2"></td>
```

```
    <td></td>
    <td></td>
    <td></td>
    <td></td>
    <td valign="bottom"><img src="images/arrowUp.jpg" id="cArrow4" name="cArrow" class="graphics"></td>

    <td colspan="2" rowspan="4"></td>
    <td><img src="images/arrowShft0.jpg" name="plierLsb" class="graphics"></td>

</tr>
<tr>
    <td rowspan="2" valign="top">
        <img src="images/arrowShft4.jpg" id="pArrow3" name="pArrow" class="graphics"></td>
    <td colspan="7" align="center" valign="bottom">
        <a class="binAdd" id="trueFalse">00000000000000000000000000000000</a>
    </td>
    <td colspan="2" valign="bottom" style="font-style:italic;font-size:.85em;">
        <a id="(product)"></a>
    </td>
    <td></td>
    <td></td>
    <td rowspan="2"><img src="images/arrowShft4.jpg" id="cArrow3" name="cArrow" class="graphics"></td>
    <td rowspan="2"><img src="images/arrowShft4.jpg" name="plierLsb" class="graphics"></td>
</tr>
<tr>
    <td colspan="7" align="center" valign="center">
        <!-- need to add a plus sign-->
        <a class="binAdd" id="addBitString" style="border-bottom:1px solid brown;">
            00000000000000000000000000000000
        </a>
    </td>
    <td colspan="2" style="font-style:italic;font-size:.85em;"><a id="(multiplicand)"></a></td>

    <td></td>
    <td></td>
</tr>
<tr>
    <td><img src="images/arrowShft5.jpg" id="pArrow2" name="pArrow" class="graphics"></td>
    <td colspan="7" align="center">
        <a class="binAdd" id="sumBitString">
            00000000000000000000000000000000
        </a>
    </td>
    <td colspan="2" align="right" valign="top" id="Horz2cell">
        <img src="images/arrowHorz2.jpg" id="cArrow0" name="cArrow" class="graphics" valign="top">
    </td>
```

```
    <td colspan="2" valign="top">
        <img src="images/arrowHorz2.jpg" id="cArrow1" name="cArrow" class="graphics">
    </td>
    <td valign="top">
        <img src="images/arrowElbow3.jpg" id="cArrow2" name="cArrow" class="graphics">
    </td>
    <td><img src="images/arrowShft5.jpg" name="plierLsb" class="graphics"></td>

</tr>
<tr>
    <td valign="top"><img src="images/arrowShft0.jpg" id="pArrow1" name="pArrow" class="graphics"></td>

    <td colspan="4"></td>
    <td valign="bottom">
        <img src="images/arrowDown.jpg" id="aluOut1" name="aluArrow" class="graphics">
    </td>
    <td colspan="2" align="center" valign="bottom">
        <img src="images/arrowRt.jpg" id="rtArrow0" name="rtArrow" class="graphics">
    </td>
    <td colspan="5"></td>
    <td colspan="2" align="center" valign="bottom">
        <img src="images/arrowRt.jpg" id="rtArrow1" name="rtArrow" class="graphics">
    </td>
    <td align="right"><img src="images/arrowShft0.jpg" name="plierLsb" class="graphics"></td>

</tr>
<tr>
    <td rowspan="2"><img src="images/arrowElbow2.jpg" id="pArrow0" name="pArrow" class="graphics"></td>
    <td colspan="7" class="label">product register (32 bits):</td>
    <td colspan="7" class="label">multiplier register (32 bits):</td>
    <td rowspan="2"><img src="images/arrowElbow2b.jpg" name="plierLsb" class="graphics"></td>

</tr>
<tr>
    <td colspan="7" class="register" style="border-right:1px dotted brown;">
        <a id="productReg">00000000000000000000000000000000</a>
    </td>
    <td colspan="7" class="register" style="border-left:1px dotted brown;">
        <a id="multiplierReg">00000000000000000000000000000000</a>
    </td>
</tr>
<tr>
    <td colspan="16" align="center" class="label">64 bits</td>
</tr>
<tr style="color:transparent;">
    <td style="width: 0px;"> 1</td>
```

```
            <td style="width: 0px;"> 2</td>
            <td style="width: 0px;"> 3</td>
            <td style="width: 0px;"> 4</td>
            <td style="width:50px;"> 5</td>
            <td style="width: 0px;"> 6</td>
            <td style="width:55px;"> 7</td>
            <td style="width:55px;"> 8</td>
            <td style="width:50px;"> 9</td>
            <td style="width:50px;">10</td>
            <td style="width: 0px;">11</td>
            <td style="width: 0px;">12</td>
            <td style="width:50px;">13</td>
            <td style="width: 0px;">14</td>
            <td style="width: 0px;">15</td>
            <td style="width:50px;">16</td>
            <td style="width: 0px;">17</td>

        </tr>
        <tr>
            <td colspan="18" style="border-top:1px solid brown;
                                    text-align:center;font-size:.75em;">
                <br><br>
                &copy; 2015,  Stacy Bridges
            </td>
        </tr>
    </table><!-- end mainTable -->

</div><!-- end mainDiv -->

<!-- end user front end ////////////////////////////////////////////////////// -->

<!-- begin help descriptions //////////////////-->
<div id="operationHelp"
 style="position:relative;top:-774px;left:232px;width:30%;visibility:hidden;" name="helper">
    <table class="helpBox">
        <tr>
            <td style="padding:0px 0px 0px 8px;">
                <a>Select an integer operation<br>for the SIM to run.</a>

            </td>
        </tr>
    </table>
</div>

<div id="operandsHelp" style="position:relative;top:-706px;left:232px;width:30%;visibility:hidden;" name="helper">
    <table class="helpBox">
```

```
        <tr>
            <td style="padding:0px 0px 0px 8px;">
                <a>Enter two integer operands (non-negative)<br>to use in the SIM operation.</a>

            </td>
        </tr>
    </table>
</div>

<div id="modeHelp" style="position:relative;top:-440px;left:228px;width:30%;visibility:hidden;" name="helper">
    <table class="helpBox">
        <tr>
            <td style="padding:0px 0px 0px 8px;">
                <a>The SIM is set to run continuously<br>
                   from beginning to end, but you may<br>
                   use the "run" and "stop" links in<br>
                   the nav menu for more control.
                </a>
            </td>
        </tr>
    </table>
</div>

<div id="speedHelp" style="position:relative;top:-426px;left:228px;width:30%;visibility:hidden;" name="helper">
    <table class="helpBox">
        <tr>
            <td style="padding:0px 0px 0px 8px;">
                <a>Select a speed to run the SIM from<br>
                   slow (25%) to fast (100%). You may<br>
                   hover over each radio button to see<br>
                   its speed. Click "x10" to increase<br>
                   the speed ten times. Once selected, the<br>
                   speed is set until the SIM is reset.
                </a>
            </td>
        </tr>
    </table>
</div>
<!-- end help descriptions -->

<!-- LOG FLOATING LAYER -->
<!-- Start Floating Layer -->
<div id="viewerDiv"
 style="position:absolute;left:620px;top:300px;visibility:hidden;overflow:scroll;height:600px;">
    <table style="font-size:.70em;background-color:white;border:1px solid brown;">
        <tr>
```

```
    <td id="moveZone"> Ops Log
        <a style="position:relative;left:720px;" class="help" onclick="toggleLog()">&nbspX </a>
    </td>
</tr>
<tr>
    <td id="loga">
        step            
        multiplicand     
                                
                      
        product                  
                                
        multiplier
    </td>
</tr>
<tr>
    <td id="logb">
        -------  
        -------------------------------- 
        -------------------------------- 
        --------------------------------
    </td>
</tr>
<tr><td id="log0"></td></tr>
<tr><td id="log1"></td></tr>
<tr><td id="log2"></td></tr>
<tr><td id="log3"></td></tr>
<tr><td id="log4"></td></tr>
<tr><td id="log5"></td></tr>
<tr><td id="log6"></td></tr>
<tr><td id="log7"></td></tr>
<tr><td id="log8"></td></tr>
<tr><td id="log9"></td></tr>
<tr><td id="log10"></td></tr>
<tr><td id="log11"></td></tr>
<tr><td id="log12"></td></tr>
<tr><td id="log13"></td></tr>
<tr><td id="log14"></td></tr>
<tr><td id="log15"></td></tr>
<tr><td id="log16"></td></tr>
<tr><td id="log17"></td></tr>
<tr><td id="log18"></td></tr>
<tr><td id="log19"></td></tr>
<tr><td id="log20"></td></tr>
<tr><td id="log21"></td></tr>
<tr><td id="log22"></td></tr>
```

```
        <tr><td id="log23"></td></tr>
        <tr><td id="log24"></td></tr>
        <tr><td id="log25"></td></tr>
        <tr><td id="log26"></td></tr>
        <tr><td id="log27"></td></tr>
        <tr><td id="log28"></td></tr>
        <tr><td id="log29"></td></tr>
        <tr><td id="log30"></td></tr>
        <tr><td id="log31"></td></tr>
        <tr><td id="log32"></td></tr>
        <tr><td id="log33"></td></tr>
        <tr><td id="log34"></td></tr>
        <tr><td id="log35"></td></tr>
        <tr><td id="log36"></td></tr>
        <tr><td id="log37"></td></tr>
        <tr><td id="log38"></td></tr>
        <tr><td id="log39"></td></tr>
        <tr><td id="log40"></td></tr>
        <tr><td id="log41"></td></tr>
        <tr><td id="log42"></td></tr>
        <tr><td id="log43"></td></tr>
        <tr><td id="log44"></td></tr>
        <tr><td id="log45"></td></tr>
        <tr><td id="log46"></td></tr>
        <tr><td id="log47"></td></tr>
        <tr><td id="log48"></td></tr>
        <tr><td id="log49"></td></tr>
        <tr><td id="log50"></td></tr>
        <tr><td id="log51"></td></tr>
        <tr><td id="log52"></td></tr>
        <tr><td id="log53"></td></tr>
        <tr><td id="log54"></td></tr>
        <tr><td id="log55"></td></tr>
        <tr><td id="log56"></td></tr>
        <tr><td id="log57"></td></tr>
        <tr><td id="log58"></td></tr>
        <tr><td id="log59"></td></tr>
        <tr><td id="log60"></td></tr>
        <tr><td id="log61"></td></tr>
        <tr><td id="log62"></td></tr>
        <tr><td id="log63"></td></tr>
        <tr><td id="log64"></td></tr>
        <tr><td id="log65"></td></tr>
    </table>
</div>
<!-- end log (floating layer) -->
```

```
</body>
</html>
```