# Supplementary Material

Title of Dissertation: RESISTIVE RAM BASED MAIN-MEMORY SYSTEMS: UNDERSTANDING THE OPPORTUNITIES, LIMITATIONS, AND TRADEOFFS

Meenatchi Jagasivamani

Doctor of Philosophy, 2020

Dissertation directed by: Professor Bruce Jacob

Department of Electrical & Computer Engineering

## Summary of Contents

# Appendix A: Cadence Encounter Command File

The following is the final Cadence Encounter command file script used to implement the four-cluster blockage regions and perform the APR to generate the final layout.

```
##########################################
# Cadence Encounter Command File
# Design: Integrated ReRAM with Scaled VSCALE Processor
# Meenatchi Jagasivamani 2018

set_global _enable_mmmc_by_default_flow      $CTE::mmmc_default
suppressMessage ENCEXT-2799
getVersion
win
set ::TimeLib::tsgMarkCellLatchConstructFlag 1
set conf_in_tran_delay {120.0ps}
# allow for random io placement

set defHierChar {/}
set distributed_client_message_echo {1}
set init_assign_buffer {0}

set init_gnd_net {VSS}
set init_pwr_net {VDD}

# Read in Libraries
set init_lef_file
{/homes/mjagasiv/research/Nangate/NangateOpenCellLibrary_PDKv1_3_v2010_
12/Back_End/lef/NangateOpenCellLibrary.lef}

# Read in design netlist
set init_top_cell {vscale_core256}
set init_verilog {vscale_core256.vh}


set lsgOCPGainMult 1.000000
set opt_buf_footprint {buf}
set opt_delay_footprint {buf}
set opt_inv_footprint {inv}
set pegDefaultResScaleFactor 1.000000
set pegDetailResScaleFactor 1.000000
set timing_library_float_precision_tol 0.000010
set timing_library_load_pin_cap_indices {}
set tso_post_client_restore_command {update_timing ;
write_eco_opt_db ;}
```

```
set init_mmmc_file vscale_core256.view

####### Begin Initializing Design
init_design
getIoFlowFlag
setIoFlowFlag 0

# Specify Floorplan
floorPlan -site FreePDK45_38x28_10R_NP_162NW_34O -s 1000 1000  20 20 20
20

# Specify blockage for standard-cell placement and Metal Layers
# Four separate clusters
createPlaceBlockage -box        289.5   628     507.5   655
createPlaceBlockage -box        385     532.5   412     750.5
createRouteBlk -box     291.5   630     505.5   653      -layer 1 2 3
4 5 6 7 8 9 10
createRouteBlk -box     387     534.5   410     748.5    -layer 1 2 3
4 5 6 7 8 9 10

createPlaceBlockage -box        532.5   628     750.5   655
createPlaceBlockage -box        628     532.5   655     750.5
createRouteBlk -box     534.5   630     748.5   653      -layer 1 2 3
4 5 6 7 8 9 10
createRouteBlk -box     630     534.5   653     748.5    -layer 1 2 3
4 5 6 7 8 9 10

createPlaceBlockage -box        289.5   385     507.5   412
createPlaceBlockage -box        385     289.5   412     507.5
createRouteBlk -box     291.5   387     505.5   410      -layer 1 2 3
4 5 6 7 8 9 10
createRouteBlk -box     387     291.5   410     505.5    -layer 1 2 3
4 5 6 7 8 9 10

createPlaceBlockage -box        532.5   385     750.5   412
createPlaceBlockage -box        628     289.5   655     507.5
createRouteBlk -box     534.5   387     748.5   410      -layer 1 2 3
4 5 6 7 8 9 10
createRouteBlk -box     630     291.5   653     505.5    -layer 1 2 3
4 5 6 7 8 9 10

# Global VDD and VSS nets
clearGlobalNets
globalNetConnect VDD -type pgpin -pin VDD -inst * -all -override
globalNetConnect VDD -type tiehi -inst * -all -override
globalNetConnect VSS -type pgpin -pin VSS -inst * -all -override
```

```
globalNetConnect VSS -type tielo -inst * -all -override

set sprCreateIeRingNets {}
set sprCreateIeRingLayers {}
set sprCreateIeRingWidth 1.0
set sprCreateIeRingSpacing 1.0
set sprCreateIeRingOffset 1.0
set sprCreateIeRingThreshold 1.0
set sprCreateIeRingJogDistance 1.0

# Create power & ground rings
addRing -skip_via_on_wire_shape Noshape -skip_via_on_pin Standardcell -
center 1 -stacked_via_top_layer metal10 -type core_rin
gs -jog_distance 0.8 -threshold 0.8 -nets {VSS VDD} -follow core -
stacked_via_bottom_layer metal1 -layer {bottom metal9 top m
etal9 right metal10 left metal10} -width 5 -spacing 5 -offset 0.8

set sprCreateIeStripeNets {}
set sprCreateIeStripeLayers {}
set sprCreateIeStripeWidth 10.0
set sprCreateIeStripeSpacing 2.0
set sprCreateIeStripeThreshold 1.0

# Create power grid
addStripe -skip_via_on_wire_shape Noshape -block_ring_top_layer_limit
metal10 -max_same_layer_jog_length 1.6 -padcore_ring_bo
ttom_layer_limit metal9 -number_of_sets 3 -skip_via_on_pin Standardcell
-stacked_via_top_layer metal10 -padcore_ring_top_laye
r_limit metal10 -spacing 2 -merge_stripes_value 0.095 -layer metal10 -
block_ring_bottom_layer_limit metal10 -width 2 -nets {V
DD VSS} -stacked_via_bottom_layer metal9

# Connect Power and Ground nets
createPGPin -net VDD VDD
createPGPin -net VSS VSS
sroute -connect { blockPin padPin padRing corePin floatingStripe } -
layerChangeRange { metal1 metal10 } -blockPinTarget { nea
restTarget } -padPinPortConnect { allPort oneGeom } -padPinTarget
{ nearestTarget } -corePinTarget { firstAfterRowEnd } -floa
tingStripeTarget { blockring padring ring stripe ringpin blockpin
followpin } -allowJogging 1 -crossoverViaLayerRange { metal
1 metal10 } -nets { VDD VSS } -allowLayerChange 1 -blockPin useLef -
targetViaLayerRange { metal1 metal10 }

# Place standard cells
setPlaceMode -fp false
placeDesign
checkPlace
checkPinAssignment
```

```
# Route signals
trialRoute -maxRouteLayer 8 -highEffort

# Apply timing constraints
####### timing route -- preCTS
create_constraint_mode -name SDCvscale_core256 -sdc_files
{vscale_core256.sdc}
create_analysis_view -name typ -constraint_mode {SDCvscale_core256} -
delay_corner {default}
set_analysis_view -setup {typ} -hold {typ}
timeDesign -preCTS -idealClock -pathReports -drvReports -slackReports -
numPaths 50 -prefix v_core256_preCTS -outDir timingRep
orts
setEndCapMode -reset
setEndCapMode -boundary_tap false

######### clock tree synthesis
createClockTreeSpec -bufferList {CLKBUF_X1 CLKBUF_X2 CLKBUF_X3 BUF_X1
BUF_X16 BUF_X2 BUF_X32 BUF_X4 BUF_X8 } -output clock.sp
ec -routeClkNet
specifyClockTree -clkfile clock.spec
setCTSMode -engine ck
clockDesign -specFile clock.spec -outDir clock_report -
fixedInstBeforeCTS

######## RC extraction
setDrawView place
timeDesign -reportOnly -pathReports -drvReports -slackReports -numPaths
50 -prefix v_core256 -outDir timingReports
setNanoRouteMode -quiet -timingEngine {}
setNanoRouteMode -quiet -routeWithTimingDriven 1
setNanoRouteMode -quiet -routeWithSiDriven 1
setNanoRouteMode -quiet -routeWithSiPostRouteFix 0
setNanoRouteMode -quiet -drouteStartIteration default
setNanoRouteMode -quiet -routeTopRoutingLayer default
setNanoRouteMode -quiet -routeBottomRoutingLayer default
setNanoRouteMode -quiet -drouteEndIteration default
setNanoRouteMode -quiet -routeWithTimingDriven true
setNanoRouteMode -quiet -routeWithSiDriven true
#setNanoRouteMode routeWithECO true

snapFPlan -guide -block -stdCell -pinBlk -ptnCore -placeBlk -macroPin -
pin
globalDetailRoute

# Add filler cells and perform parasitic extraction
getFillerMode -quiet
addFiller -cell FILLCELL_X2 FILLCELL_X4 FILLCELL_X8 FILLCELL_X16
FILLCELL_X32 -prefix FILL_
extractRC
timeDesign -postRoute -pathReports -drvReports -slackReports -numPaths
50 -prefix v_core256_postRoute -outDir timingReports

########## power measurement
```

```
set_power_analysis_mode -reset
set_power_analysis_mode -method static -corner max -create_binary_db
true -write_static_currents true -honor_negative_energy
true -ignore_control_signals true
set_power_output_dir -reset
set_power_output_dir ./
set_default_switching_activity -reset
set_default_switching_activity -input_activity 0.2 -period 10.0
read_activity_file -reset
set_power -reset
set_powerup_analysis -reset
set_dynamic_power_simulation -reset
report_power -rail_analysis_format VS -outfile .//v_core256.rpt

############## verify connectivity (LVS)
verifyConnectivity -type all -noAntenna -error 1000 -warning 50

##########  verify geometry (DRC)
setVerifyGeometryMode -area { 0 0 0 0 } -minWidth true -minSpacing true
-minArea true -sameNet true -short true -overlap true
 -offRGrid false -offMGrid true -mergedMGridCheck true -minHole true -
implantCheck true -minimumCut true -minStep true -viaEn
closure true -antenna false -insuffMetalOverlap true -pinInBlkg false -
diffCellViol true -sameCellViol false -padFillerCellsO
verlap true -routingBlkgPinOverlap false -routingCellBlkgOverlap true -
regRoutingOnly false -stackedViasOnRegNet false -wireE
xt true -useNonDefaultSpacing false -maxWidth true -maxNonPrefLength -1
-error 1000

verifyGeometry

######## Rerun detailed route with ECO mode
setNanoRouteMode -quiet -routeWithEco 1
setNanoRouteMode -quiet -drouteStartIteration default
setNanoRouteMode -quiet -routeTopRoutingLayer default
setNanoRouteMode -quiet -routeBottomRoutingLayer default
setNanoRouteMode -quiet -drouteEndIteration default
setNanoRouteMode -quiet -routeWithTimingDriven true
setNanoRouteMode -quiet -routeWithSiDriven true
routeDesign -globalDetail
verifyGeometry
setLayerPreference violation -isVisible 1
violationBrowser -all -no_display_false
verifyConnectivity -type all -noAntenna -error 1000 -warning 50

######### Timing report
timeDesign -reportOnly -pathReports -drvReports -slackReports -numPaths
50 -prefix v_core256 -outDir timingReports

###### Stream out and save design
streamOut v_core256.gds -
mapFile ../FreePDK45/osu_soc/lib/files/gds2_encounter.map -libName
libJuly5 -structureName v_core256
 -stripes 1 -units 1000 -mode ALL
```

```
saveNetlist v_core256_PR.v

saveDesign v_core256.enc

summaryReport -noHtml -outfile summaryReport.rpt

reportGateCount -level 5 -limit 100 -outfile v_core256.gateCount
reportNetStat
##########################################
```

# Appendix B: MAC Javascript Source Code

The following is the Javascript source code for the Monolithic Architecture Calculator (MAC) mentioned in Section **Error! Reference source not found..**

```
<html>
<head>
   <title>Monolithic Architecture Calculator</title>
<style>
p {
    font-family: Tahoma, Geneva, sans-serif;
}

table, th, td {
    margin-left:15px;
    border: 1px solid black;
    border-collapse: collapse;
}
th, td {
    padding: 10px;
}
* {
    box-sizing: border-box;
     font-family: Tahoma, Geneva, sans-serif;
}

.row {
    display: flex;
    border-style: double;
    border-width: thick;
}
.input_form {
    padding:0px;
    margin:0px;
```

```
    margin-left: 25px;
}



/* Create two equal columns that sits next to each other */
.column {
    flex: 50%;
    padding: 10px;
    border-style: double;
    border-width: thin;

}
.slidecontainer {
    margin-left: 25px;
}

.slider {
    -webkit-appearance: none;
    width: 50%;
    height: 10px;
    //border-radius: 5px;
    background: #d3d3d3;
    outline: none;
    opacity: 0.7;
    -webkit-transition: .2s;
    transition: opacity .2s;
}

.slider:hover {
    opacity: 1;
}

.slider::-webkit-slider-thumb {
    -webkit-appearance: none;
    appearance: none;
    width: 25px;
    height: 25px;
   // border-radius: 50%;
    background: #537d9b;
    cursor: pointer;
}

.slider::-moz-range-thumb {
    width: 25px;
    height: 25px;
 //   border-radius: 50%;
    background: #537d9b;
    cursor: pointer;
}
```

```
</style>
<script language=javascript type="text/javascript">


function round(value, precision) {
    var multiplier = Math.pow(10, precision || 0);
    return Math.round(value * multiplier) / multiplier;
}
function calculate(){
    var memSelected = document.getElementById('memory_type');
    var networkSelected = document.getElementsByName('networkT');
    var procSelected = document.getElementsByName('procT');
    var procT = document.getElementById('procT');
    var diesize = document.getElementById("diesize");
    var numMCs = document.getElementById("Input_numMC").value;
    var slideCol = document.getElementById("cache_ratio");
    var cache_ratio = slideCol.value;
    var l2l3_ratio = slideCol2.value;

      //var cache_ratio = document.getElementById("cache_ratio").value;

    var availArea=diesize.value;
      //calculate available area based on memory type selected
    var CPoverhead = 0;
      if (memSelected.value == "NVR") {
        //charge pump % area -> typical flash ratio
        //Reference:  7% of 16nm Micron NAND flash diesize of 173 mm^2
        CPOverhead=0.07*173;
        availArea = availArea-CPoverhead;
      } else if (memSelected.value == "dram") {
        //no charge pump, but external Memory controller interface logic
        // Typical 10% area
        //Reference:  10% of 16nm Micron NAND flash diesize of 173 mm^2
        CPOverhead=0.1*173;
        availArea = availArea-CPoverhead;
      } else if (memSelected.value == "VR") {
        //charge pump % area -> half of typical flash ratio
        //Reference:  3.5% of 16nm Micron NAND flash diesize of 173 mm^2
        CPOverhead=0.035*173;
        availArea = availArea-CPoverhead;
      }

    //calculate NoC overhead
      if (networkSelected[0].checked) {
```

```
        //Mesh -- upto 4 links per node
        //TODO placeholder 10% overhead
        availArea = availArea*0.9;
      } else if (networkSelected[1].checked) {
        //Hoffman-Singleton -- upto 3 links per node
        //TODO placeholder 7% overhead
        availArea = availArea*0.93;
      }

    //Calculate available SRAM Cache area
      //Reference:  IEDM 2013, TSMC 6T SRAM bit cell area for 16nm: 0.07
sq. micron
      //Using typical value of 70% array efficiency for overhead circuitry
      var SRAMcellsize=0.07;
      var SRAMarea=(1-cache_ratio)*availArea*1000000*0.7;
      var SRAMsize = SRAMarea/(SRAMcellsize*1024*1024*8);
      document.getElementById("Total_Cache").innerHTML            =
round(SRAMsize,1)+" MB";
      document.getElementById("L2").innerHTML          =          "L2:
"+round(SRAMsize*l2l3_ratio,1)+" MB";
      document.getElementById("L3").innerHTML          =          "L3:
"+round(SRAMsize*(1-l2l3_ratio),1)+" MB";

      //Fixed  32kB  size  for  Queuing  buffer  of  Memory  controller  --
supports 512 depth of 512 bits
      var MCBuffer_area=(SRAMcellsize*32*1024*8/1000000)/0.7;

    //set processor area
    var procSize = 0.0284077;
    if (procSelected[0].checked) {
        //Raven-3 RISC-V: Scaled 0.46x from 28nm node
        procSize = 0.46*1.19; //mm^2
      } else if (procSelected[1].checked) {
        //Fujitsu Sparc64 X+: Scaled 0.46x from 28nm node
        procSize = 0.46*10.907; //mm^2
      } else if (procSelected[2].checked) {
        //Intel Skylake-X CPU core (14nm)
        procSize = 16.9; //mm^2
      } else if (procSelected[3].checked) {
        //Intel Xeon-Phi core -- knights landing (14nm)
        procSize = 3.13; //mm^2
      }

      var Procarea= cache_ratio*availArea;
```

```
    var numProcessors = Procarea/procSize; //initial value ignoring ReRAM
congestion penalty
    //calculate Main-Memory bandwidth, latency
    //for ReRAM (default), bw=8*4 bits per memory controller
    var bw = 32*numMCs;
    var rLatency = 200;
    var wLatency = 1000;

      //Power numbers derived based on McPAT memory access model
      //Crossbar for ReRAM cell energy number
      var rPwr=4.24; //W
      var wPwr=41.86; //W

   //Calculate # of Mem Controllers under SRAM
    if (memSelected.value == "dram") {
        //charge pump % area -> typical flash ratio
        //Reference:  7% of 16nm Micron NAND flash diesize of 173 mm^2
        document.getElementById("Total_ReRAM").innerHTML = "External MM
";
        document.getElementById("Total_ReRAM2").innerHTML = " ";
        document.getElementById("Total_ReRAM8").innerHTML = " ";

            //128 bits per memory controller
        bw = 128*numMCs;
        rLatency = 55;
        wLatency = 55;
            rPwr=0; //W
            wPwr=0; //W

      }  else if (numMCs>0) {

            //calculate ReRAM storage amount
            //Assumptions:  50% array efficiency
            //Cell-size at 16nm: 0.011236 sq-um at 45nm --> scales down
by 10x at 16nm
            //Reference: Crossbar ReRAM
            // Avail area =  with 25% periphery overhead +  congestion
penalty

            var MMArea = availArea*0.5;  //50% array efficiency
            var ReRAMcellsize=0.011236/10;
            var
ReRAMsize=MMArea*1000000/(ReRAMcellsize*1024*1024*1024*8);
            //check if numMemory Controllers entered is invalid
            var MMGranularity=ReRAMsize*2*1024/numMCs;

            /////////////Calculat           congestion           penalty
////////////////////
            var congestion = 0.12;  //congestion penalty is function of
# of MCs
```

```
            //remove RERAM periphery circuit area -- applies to Proc and
Cache
            availArea = availArea*(1-0.5*0.25);
            //remove Memory-Controller queuing buffer area
            availArea = availArea-(numMCs*MCBuffer_area);

            if (numMCs <= (availArea/procSize)) {
                  congestion = 0.12*(numMCs/(availArea*0.88/procSize));
//numMCs/numProcTiles (approx)
            } else {
                  congestion                                           =
congestion+0.04*Math.log2((numMCs/(availArea/procSize)));
//numMCs/numProcTiles (approx)
                  var   tmpArea     =   availArea*(1-congestion);   //not
available for either proc or sram
                  congestion                                           =
congestion+0.04*Math.log2((numMCs/(tmpArea/procSize)));
//numMCs/numProcTiles (approx)
            }
            if                                            (congestion<0)
document.getElementById("debug").innerHTML = " **** ERROR:    INVALID
Congestion Penalty parameter **** " ;
            //document.getElementById("debug").innerHTML      =      "
numproctiles="+(availArea*0.88/procSize)+" MMarea="+MMArea;
            //document.getElementById("debug").innerHTML      =      "
MCBuffer_area="+MCBuffer_area*numMCs;


            //recalculate after removing wasted area for congestion
            availArea  =  availArea*(1-congestion);  //not  available  for
either proc or sram
            ReRAMsize=MMArea*1000000/(ReRAMcellsize*1024*1024*1024*8);
            MMGranularity=ReRAMsize*2*1024/numMCs;

            if (MMGranularity > 1) {
                  document.getElementById("Total_ReRAM").innerHTML    =
round(ReRAMsize,1)+" GB ";//congestion="+congestion;
                  document.getElementById("Total_ReRAM2").innerHTML   =
round(ReRAMsize*2,1)+" GB for 2-layer stack ";
                  document.getElementById("Total_ReRAM8").innerHTML   =
round(ReRAMsize*8,1)+" GB for 8-layer stack";
            } else {
                  document.getElementById("Total_ReRAM").innerHTML    =
"ERROR -- too high # of Memory Controllers";
                  document.getElementById("Total_ReRAM2").innerHTML = "
";
                  document.getElementById("Total_ReRAM8").innerHTML = "
";
```

```
                bw=0;
                rLatency=0;
                wrLatency=0;
            }

            //update num processors after considering reram congestion
penalty
            Procarea= cache_ratio*availArea;
            numProcessors = Procarea/procSize;
    } else {
        document.getElementById("Total_ReRAM").innerHTML = "0 GB ";
        document.getElementById("Total_ReRAM2").innerHTML = "0 GB ";
        document.getElementById("Total_ReRAM8").innerHTML = "0 GB ";
            bw=0;
            rLatency=0;
            wLatency=0;
      }

      if (availArea <= 0) {
            document.getElementById("NumMC").innerHTML = "ERROR --
Invalid Configuration";
            document.getElementById("NumProc").innerHTML = " ";
      } else {
            document.getElementById("NumMC").innerHTML =
round(numMCs,0);
            document.getElementById("NumProc").innerHTML =
round(numProcessors,0);
      }
    //calculate number of NoC controllers: 1 per tile
    var numNoCs = numMCs;
      if (networkSelected[0].checked) {
        //Mesh -- upto 4 links per node
        //TODO placeholder 10% overhead
        numNoCs = numMCs;
      } else if (networkSelected[1].checked) {
        //Hoffman-Singleton -- upto 3 links per node
        //TODO placeholder 7% overhead
        numNoCs = numMCs*0.75;
      }
    // document.getElementById("NumNoc").innerHTML = round(numNoCs,0);


    //add NoC Latency
      if (networkSelected[0].checked) {
        //Mesh -- upto 4 links per node
        //TODO placeholder 10% overhead
        //rLatency = rLatency * 1.1;
        //wLatency = wLatency * 1.1;
```

```
      } else if (networkSelected[1].checked) {
        //Hoffman-Singleton -- upto 3 links per node
        //TODO placeholder 7% overhead
        //rLatency = rLatency * 1.07;
        //wLatency = wLatency * 1.07;
      }
    //convert to bytes
    bw=round(bw,0)
    bw = bw/8;
    document.getElementById("bw").innerHTML = round(bw,2);
    document.getElementById("latency").innerHTML = round(rLatency,2)+"
ns & "+round(wLatency,2);

    //calculate power consumption for Processor, MM, SRAM
    pwr = 0; // unit mW/Hz
    performance = 0;
    //for Processor:
    if (procSelected[0].checked) {
        //RISC-V Ref IEEE Micro 2016: 961MHz, 173mW, 34GFLOPS/w
        //pwr = 0.173*numProcessors/961; //w/MHz
        pwr = 0.173*numProcessors; //w
        performance = 34; //GFLOPS/W
      } else if (procSelected[1].checked) {
        //Sparc Ref: Fujitsu Sparc64 X+ wikipedia 392W, 448GFLOPS, 3.5GHz
        //pwr = 392*(numProcessors/16)/3500;  //w/MHz
        pwr = 392*(numProcessors/16);  //w
        performance = 448/392; //GFLOPS/W at 1.4GHz
      } else if (procSelected[2].checked) {
        //Intel Skylake-X CPU core (14nm) spec: 165W, 1152 GFLOPS at
4.4GHz (turbo)
        //pwr = 165*(numProcessors/18)/4400; //w/MHz
        pwr = 165*(numProcessors/18); //w
        performance = 1152/165; //GFLOPS/W at 1GHz
      } else if (procSelected[3].checked) {
        //Intel Knights Landing/Mill Xeon Phi  (14nm) spec: 260W, 3456
GFLOPS at 1.7GHz (turbo)
        //pwr = 260*(numProcessors/72)/1700;  //w/MHz
        pwr = 260*(numProcessors/72);  //w
        performance = 3456/260; //GFLOPS/W at 1GHz
      }
    document.getElementById("pwr").innerHTML = "Processor:
"+round(pwr,3)+" w";
      if (memSelected.value == "dram")
            document.getElementById("pwr_MM").innerHTML = " ";
      else
```

```
            document.getElementById("pwr_MM").innerHTML = "Main-Memory: Read: "+round(rPwr,2)+" W & Write: "+round(wPwr,2)+" W";
    document.getElementById("performance").innerHTML = round(performance,2)+" GFLOPS/W ";

}
</script>

</head>

<body>
<h2>Monolithic Architecture Calculator (MAC)</h2>

<b>Purpose:</b>  Preliminary estimates on what can "fit" in a given chip dimension and assess architectural tradeoffs with various design options on a Memory-Processors System.

<br> Process Node: 16nm<br><br>


<div class="row" >

<div class="column"  >
<form name="calc" action="post">
<p>
1) Main Memory Type:
<select id="memory_type">
    <option value=NVR>Non-Volatile ReRAM </option>
    <option value=dram>DRAM</option>
    <option value=VR>Volatile ReRAM</option>
</select>
<br>
<br>

2) Die-Size [mm^2]: <input type=text id="diesize" size=10 value="686">
<br>
  <div class="input_form"></div>

3) Select Processor Core Type:
  <div class="input_form">
    <input type="radio" id="riscv"
     name="procT" value="riscv">
    <label for="riscv">Raven-3 RISC-V w/56kB L1 per core </label><br>

    <input type="radio" id="sparc"
     name="procT" value="sparc">
    <label for="sparc">Sparc64 XII w/128kB L1 per core </label><br>

    <input type="radio" id="gpu"
     name="procT" value="gpu">
    <label for="gpu">Intel Skylake-X w/64kB L1 per core  </label><br>

    <input type="radio" id="knl"
     name="procT" value="knl" checked="checked">
```

```
    <label for="gpu">Intel Xeon Phi (Knights Landing) w/32kB L1 per core
</label><br>
  </div>
  <br>

4) Core-to-Cache Area ratio for SRAM Cache on Chip (0 to 1):<br>
  (0:all SRAM; 1: all Processor, no SRAM): <br>
<div class="slidecontainer">
   <input  type="range"  min="0"  max="1"  step="0.05"  value="0.85"
id="cache_ratio" class="slider">
  Value: <span id="f" style="font-weight:bold;"> </span>
<script>
var slideCol = document.getElementById("cache_ratio");
var y = document.getElementById("f");
y.innerHTML = slideCol.value;

slideCol.oninput = function() {
    y.innerHTML = this.value;
}

</script>

</div>
<br>

5)  Number  of  Memory  Controllers:  <input  type=text  id="Input_numMC"
size=10 value="131"> <br>
  <div class="input_form"></div>
  <br>


6) Network Topology:
  <div class="input_form">

    <input type="radio" id="mesh"
     name="networkT" value="mesh" checked="checked">
    <label for="mesh">Mesh network</label><br>
     <!---
    <input type="radio" id="singleton"
     name="networkT" value="singleton" disabled>
    <label for="singleton">Hoffman-Singleton Graph</label><br>
      -->
  </div>
  <br>

-------------------------------------------------------<p
align="center">
<input type=button value="CALCULATE" onClick="calculate()"></p>
</form>
</div>
<div class="column" >
 Final Architecture Configuration for given die-size and components:<br>
<a id="debug"></a>
```

```
<table>
  <tr>
      <th>Parameter</th>
      <th>Value</th>
  </tr>
  <tr>
      <td>Total Main Memory Storage Size </td>
      <td><a  id="Total_ReRAM"></a><br><a  id="Total_ReRAM2"></a>  <br><a
id="Total_ReRAM8"></a> <br> </td>
  </tr>
  <tr>
      <td>Total SRAM Cache Size
            <p>L2-to-L3 ratio: <p>
<div class="slidecontainer">
   <input type="range" min="0" max="1" step="0.05" value="1" id="l2_l3"
class="slider"> <p>
    <span id="l2l3" style="font-weight:bold;">  </span>
<script>
var slideCol2 = document.getElementById("l2_l3");
var y2 = document.getElementById("l2l3");
y2.innerHTML = slideCol2.value;

slideCol2.oninput = function() {
    y2.innerHTML = this.value;
}

</script>

</div>
      </td>
      <td>
      <a id="Total_Cache"></a> <br>
      <a id="L2"></a> <br>
      <a id="L3"></a>

</td>
  </tr>
  <tr>
      <td>Number of Processors <br>
    Number of Memory Controllers <br>
  <!--  Number of NoC Controllers <br> -->
    </td>
      <td><a id="NumProc"></a> <br>
        <a id="NumMC"></a> <br>
   <!--     <a id="NumNoc"></a> <br> -->
    </td>
  </tr>
  <tr>
      <td>Main Memory (MM) Bandwidth </td>
      <td><a id="bw"></a> bytes per access   </td>
  </tr>
  <tr>
      <td>MM Read & Write Latency </td>
```

```
        <td><a id="latency"></a> ns  </td>
    </tr>
    <tr>
        <td>Power Consumption </td>
        <td><a id="pwr"></a>   <br>
        <a id="pwr_MM"></a>    <br>
        </td>
    </tr>
    <tr>
        <td>Energy Efficiency </td>
        <td><a id="performance"></a>  </td>
    </tr>
</table>
</div>


</div>


<hr>

<div align="right"><small>
   University of Maryland, College Park, MD<br>
   Electrical and Computer Engineering<br>
   Memory Systems Research Group<br> <i>
  <br>Posted by: Meenatchi Jagasivamani
  <br><a href="mjagasiv@terpmail.umd.edu">
  mjagasiv@terpmail.umd.edu</a>.</i>
</small></div>

</body>

</html>
```