Assignment 2 - Simulation Assignment Report

Anagh Singh

January 24, 2016

Question 1 [CACTI]

CACTI: Answer the following questions for a 32nm, 4 bank, 2-way Set Associative, 64B line size, 16KB cache:

(a) How many total sets per bank are there?

(b) What is the default Vdd value?

(c) What are the components of the access time parameter in Cacti? What was the value in your case?

(d) Amongst the Data array and the Tag array, which consumed the most dynamic and leakage power? What are the values?

(e) How large (in mm^2) is the 16KB cache?

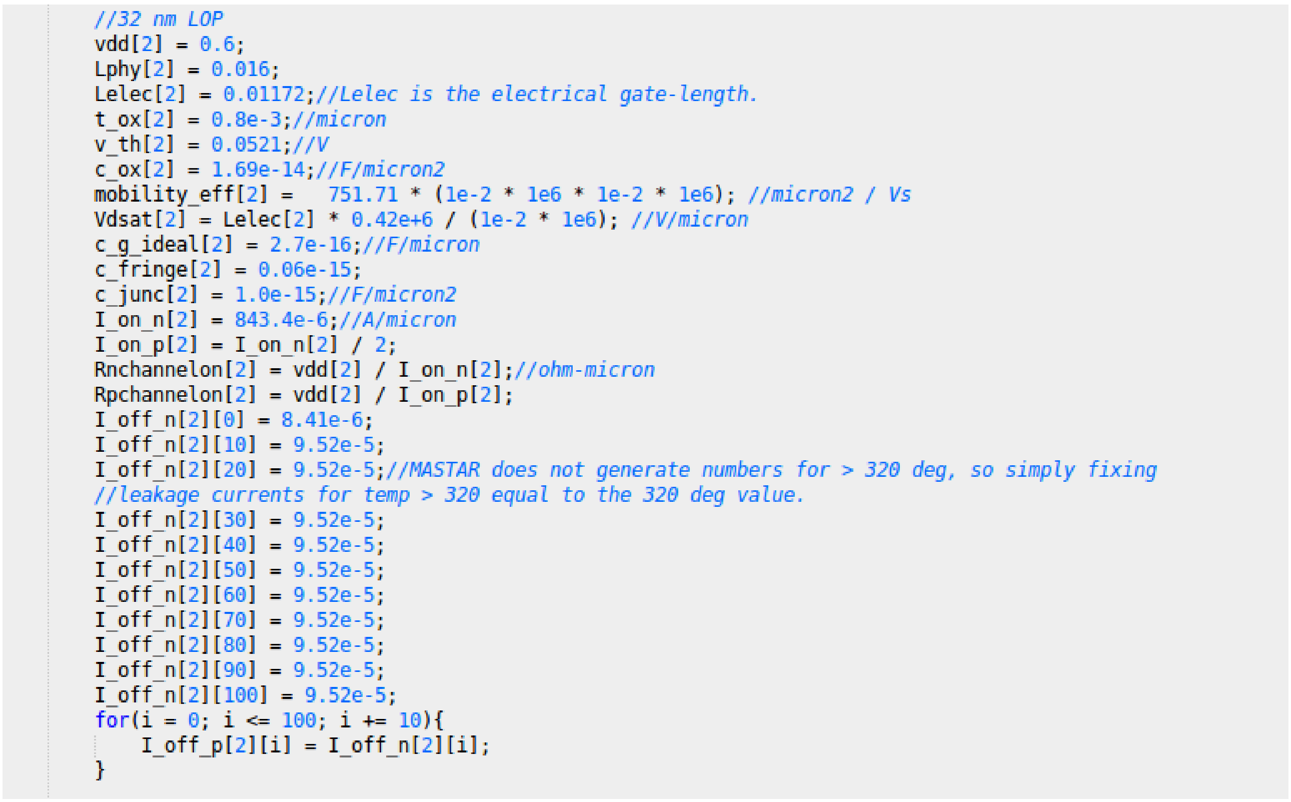
(f) Draw the architecture of a 4 bank, 2-way Set Associative, 64B line size, 16KB cache

Solution:

The CACTI simulator was proposed by researchers at the University of Utah. Most of the following answers have been based on the original paper published by the researchers at ISCA'07 outlining their simulator.

(a) For a given total cache size, we partition the cache into 2^N cache banks (N varies from 1 to 12) and for each N, we organize the banks in a grid with 2^M rows (M varies from 0 to N). Here, N=2, since there are 4 banks. Thus the grid is of size 2x2. Therefore the number of sets are 32.

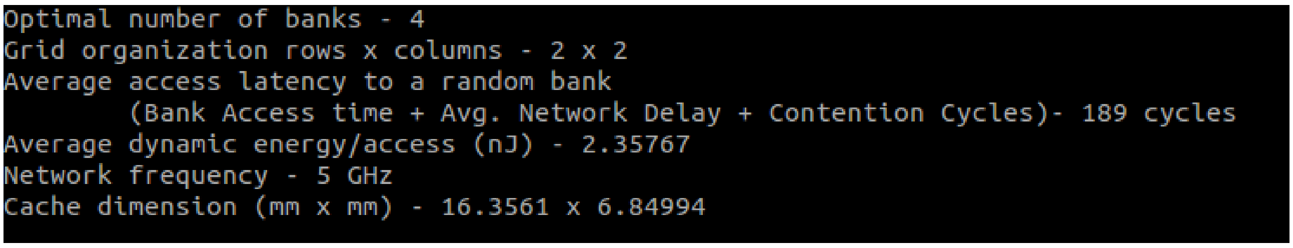
(b) The default Vdd value can be found in the ***technology.c*** file. The default value is set to 0.6V as can be seen in the following screenshot.



```
//32 nm LOP
vdd[2] = 0.6;
Lphy[2] = 0.016;
Lelec[2] = 0.01172;//Lelec is the electrical gate-length.
t_ox[2] = 0.8e-3;//micron
v_th[2] = 0.0521;//V
c_ox[2] = 1.69e-14;//F/micron2
mobility_eff[2] =   751.71 * (1e-2 * 1e6 * 1e-2 * 1e6); //micron2 / Vs
Vdsat[2] = Lelec[2] * 0.42e+6 / (1e-2 * 1e6); //V/micron
c_g_ideal[2] = 2.7e-16;//F/micron
c_fringe[2] = 0.06e-15;
c_junc[2] = 1.0e-15;//F/micron2
I_on_n[2] = 843.4e-6;//A/micron
I_on_p[2] = I_on_n[2] / 2;
Rnchannelon[2] = vdd[2] / I_on_n[2];//ohm-micron
Rpchannelon[2] = vdd[2] / I_on_p[2];
I_off_n[2][0] = 8.41e-6;
I_off_n[2][10] = 9.52e-5;
I_off_n[2][20] = 9.52e-5;//MASTAR does not generate numbers for > 320 deg, so simply fixing
//leakage currents for temp > 320 equal to the 320 deg value.
I_off_n[2][30] = 9.52e-5;
I_off_n[2][40] = 9.52e-5;
I_off_n[2][50] = 9.52e-5;
I_off_n[2][60] = 9.52e-5;
I_off_n[2][70] = 9.52e-5;
I_off_n[2][80] = 9.52e-5;
I_off_n[2][90] = 9.52e-5;
I_off_n[2][100] = 9.52e-5;
for(i = 0; i <= 100; i += 10){
    I_off_p[2][i] = I_off_n[2][i];
}
```

Figure 1: Default Vdd value

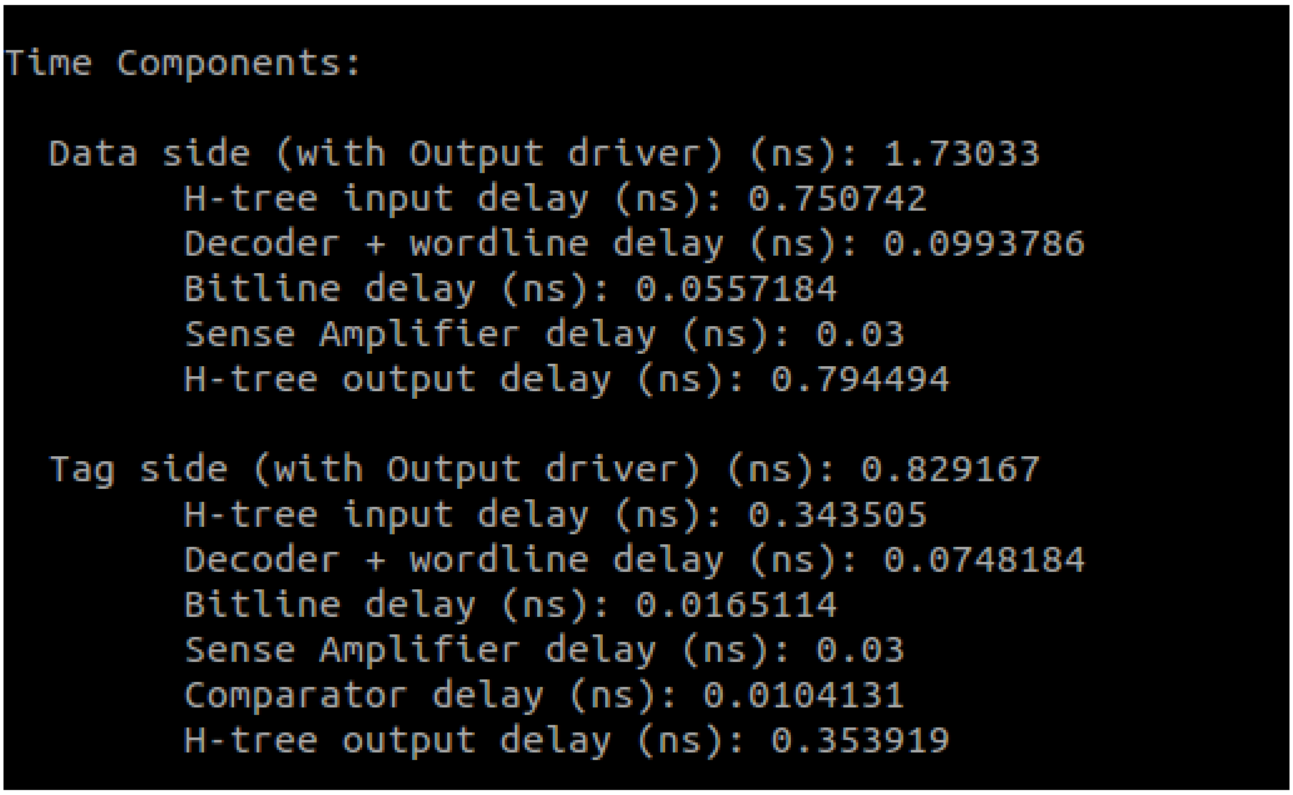
(c) The components of the access time parameter in Cacti are Bank Access time, Avg. Network Delay, and Contention Cycles. The value in this case was 189 cycles.



```
Optimal number of banks - 4
Grid organization rows x columns - 2 x 2
Average access latency to a random bank
        (Bank Access time + Avg. Network Delay + Contention Cycles)- 189 cycles
Average dynamic energy/access (nJ) - 2.35767
Network frequency - 5 GHz
Cache dimension (mm x mm) - 16.3561 x 6.84994
```

Figure 2: Access time components

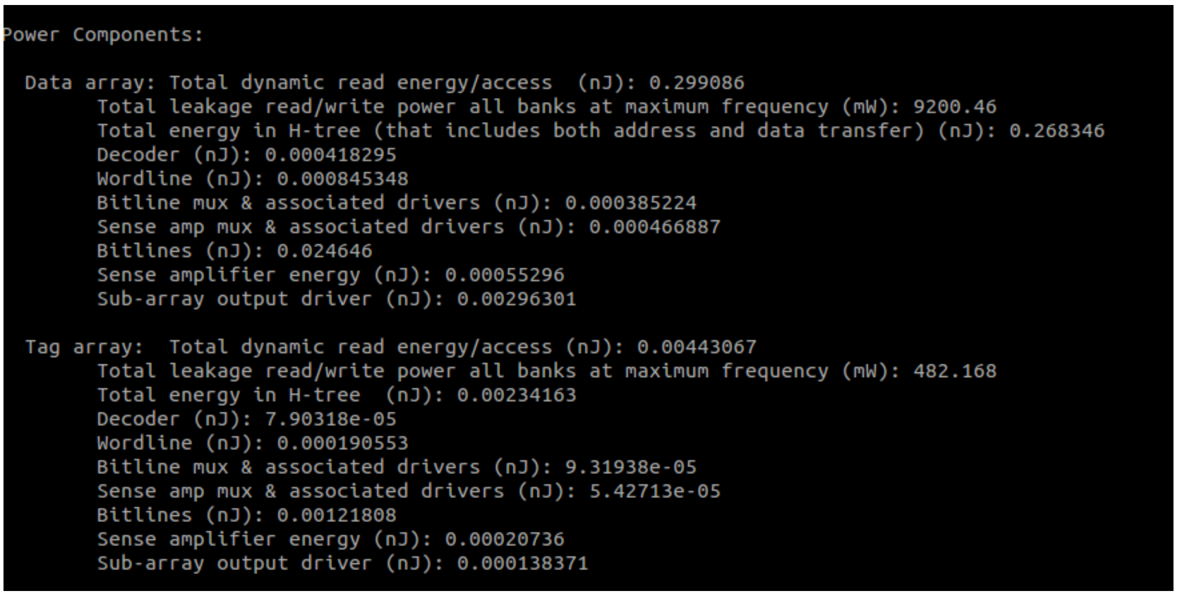
The bank access time can further be divided in the following manner, with the specific values being:



```
Time Components:

  Data side (with Output driver) (ns): 1.73033
        H-tree input delay (ns): 0.750742
        Decoder + wordline delay (ns): 0.0993786
        Bitline delay (ns): 0.0557184
        Sense Amplifier delay (ns): 0.03
        H-tree output delay (ns): 0.794494

  Tag side (with Output driver) (ns): 0.829167
        H-tree input delay (ns): 0.343505
        Decoder + wordline delay (ns): 0.0748184
        Bitline delay (ns): 0.0165114
        Sense Amplifier delay (ns): 0.03
        Comparator delay (ns): 0.0104131
        H-tree output delay (ns): 0.353919
```

Figure 3: Bank time components

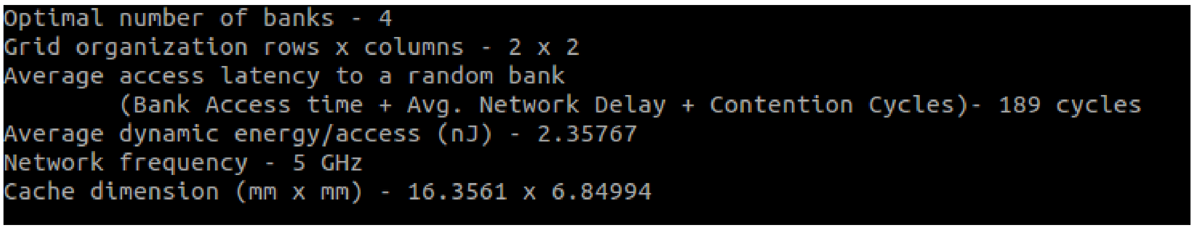
(d) For the Data Array, the dynamic power consumed was 0.299086 nJ, while leakage power consumed was 9200.46 mW. For the Tag Array, the dynamic power consumed was 0.00443067 nJ, while leakage power consumed was 482.168 mW.



```
Power Components:

  Data array: Total dynamic read energy/access  (nJ): 0.299086
        Total leakage read/write power all banks at maximum frequency (mW): 9200.46
        Total energy in H-tree (that includes both address and data transfer) (nJ): 0.268346
        Decoder (nJ): 0.000418295
        Wordline (nJ): 0.000845348
        Bitline mux & associated drivers (nJ): 0.000385224
        Sense amp mux & associated drivers (nJ): 0.000466887
        Bitlines (nJ): 0.024646
        Sense amplifier energy (nJ): 0.00055296
        Sub-array output driver (nJ): 0.00296301

  Tag array:  Total dynamic read energy/access (nJ): 0.00443067
        Total leakage read/write power all banks at maximum frequency (mW): 482.168
        Total energy in H-tree  (nJ): 0.00234163
        Decoder (nJ): 7.90318e-05
        Wordline (nJ): 0.000190553
        Bitline mux & associated drivers (nJ): 9.31938e-05
        Sense amp mux & associated drivers (nJ): 5.42713e-05
        Bitlines (nJ): 0.00121808
        Sense amplifier energy (nJ): 0.00020736
        Sub-array output driver (nJ): 0.000138371
```

Figure 4: Power usage by Data and Tag Arrays

(e) The cache has an area of 112.030125584 mm^2 as can be seen from the following screenshot.



```
Optimal number of banks - 4
Grid organization rows x columns - 2 x 2
Average access latency to a random bank
        (Bank Access time + Avg. Network Delay + Contention Cycles)- 189 cycles
Average dynamic energy/access (nJ) - 2.35767
Network frequency - 5 GHz
Cache dimension (mm x mm) - 16.3561 x 6.84994
```

Figure 5: Area of Cache

(f) The architecture of a 4 bank, 2-way Set Associative, 64B line size, 16KB cache is:

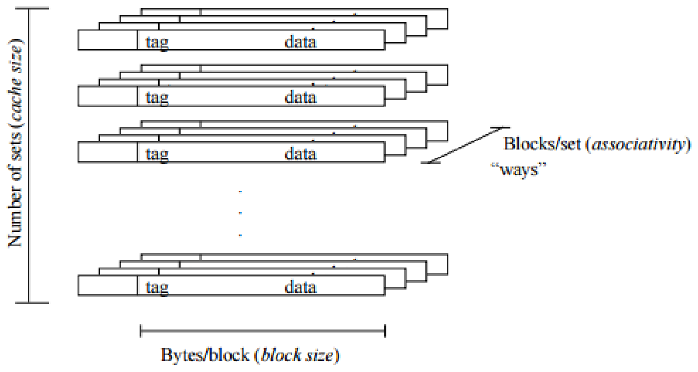

Number of sets (*cache size*)
tag
data
tag
data
tag
data
tag
data
Blocks/set (*associativity*)
"ways"
Bytes/block (*block size*)

Figure 6: Architecture of Cache

The number of bytes/block is called the block size , which is 64B over here. The blocks/set is the associativity of the cache, which is 2 here, while the number of sets is 32.

The cache.cfg file associated with this experiment is:

```
//-size (bytes) 4096
//-size (bytes) 16000
//-size (bytes) 1048576
//-size (bytes) 2097152
//-size (bytes) 4194304
//-size (bytes) 8388608
//-size (bytes) 16777216
//-size (bytes) 33554432
//-size (bytes) 134217728
-size (bytes) 67108864
//-size (bytes) 1073741824
//-block size (bytes) 8
//-block size (bytes) 128
-block size (bytes) 64

// To model Fully Associative cache, set associativity to zero
//-associativity 0
-associativity 2
//-associativity 4
//-associativity 8
//-associativity 16

-read-write port 1
-exclusive read port 0
-exclusive write port 0
-single ended read ports 0

// Multiple banks connected using a bus
-UCA bank count 1
// The following is the parameter that takes care of the X-nm process node technology
// 32 nm
-technology (u) 0.032
// 45 nm
//-technology (u) 0.045
// 68 nm
//-technology (u) 0.068
// 90 nm
//-technology (u) 0.090

// Bus width include data bits and address bits reqired by the decoder
//-output/input bus width 16
-output/input bus width 256

// 300-400 in steps of 10
-operating temperature (K) 350

// to model special structure like branch target buffers, directory, etc.
// change the tag size parameter
// if you want cacti to calculate the tagbits, set the tag size to "default"
-tag size (b) "default"
//-tag size (b) 45

// fast - data and tag access happen in parallel
// sequential - data array is accessed after accessing the tag array
// normal - data array lookup and tag access happen in parallel
//           final data block is broadcasted in data array h-tree
//           after getting the signal from the tag array
//-access mode (normal, sequential, fast) - "fast"
-access mode (normal, sequential, fast) - "normal"
//-access mode (normal, sequential, fast) - "sequential"

//NOTE: CACTI 5.0 assumes SRAM tags for a DRAM cache
//-cache type (SRAM - only data array <ex:register files, buffers etc.>, SRAM_CACHE - tag & data array, DRAM_CACHE) - "SRAM"
-cache type (SRAM - only data array <ex:register files, buffers etc.>, SRAM_CACHE - tag & data array, DRAM_CACHE) - "SRAM_CACHE"
//-cache type (SRAM - only data array <ex:register files, buffers etc.>, SRAM_CACHE - tag & data array, DRAM_CACHE) - "DRAM_CACHE"

// DESIGN OBJECTIVE for UCA (or banks in NUCA)
-design objective (weight delay, dynamic power, leakage power, cycle time, area) 100:100:0:0:0

// Percentage deviation from the minimum value
// Ex: A deviation value of 10:1000:1000:1000:1000 will try to find an organization
// that com mises at most 10% delay.
// NOTE: Try reasonable values for % deviation. Inconsistent deviation
// percentage values will not produce any valid organizations. For example,
// 0:0:100:100:100 will try to identify an organization that has both
// least delay and dynamic power. Since such an organization is not possible, CACTI will
// throw an error. Refer CACTI-6 Technical report for more details
-deviate (delay, dynamic power, leakage power, cycle time, area)
    20:100000:100000:100000:1000000

// Objective for NUCA
-NUCAdesign objective (weight delay, dynamic power, leakage power, cycle time, area)
    100:100:0:0:0
-NUCAdeviate (delay, dynamic power, leakage power, cycle time, area)
    10:10000:10000:10000:10000

// Set optimize tag to ED or ED^2 to obtain a cache configuration optimized for
// energy-delay or energy-delay sq. product
// Note: Optimize tag will disable weight or deviate values mentioned above
// Set it to NONE to let weight and deviate values determine the
// appropriate cache configuration
//-Optimize ED or ED^2 (ED, ED^2, NONE): "ED"
//-Optimize ED or ED^2 (ED, ED^2, NONE): "ED^2"
-Optimize ED or ED^2 (ED, ED^2, NONE): "NONE"



//-Cache model (NUCA, UCA)  - "UCA"
-Cache model (NUCA, UCA)  - "NUCA"

// In order for CACTI to find the optimal NUCA bank value the following
// variable should be assigned 0.
-NUCA bank count 4

// NOTE: for nuca network frequency is set to a default value of
// 5GHz in time.c. CACTI automatically
// calculates the maximum possible frequency and downgrades this value if necessary

// By default CACTI considers both full-swing and low-swing
// wires to find an optimal configuration. However, it is possible to
// restrict the search space by changing the signalling from "default" to
// "fullswing" or "lowswing" type.
-wire signalling (fullswing, lowswing, default) - "default"
//-wire signalling (fullswing, lowswing, default) - "lowswing"

// Contention in network (which is a function of core count and cache level) is one of
// the critical factor used for deciding the optimal bank count value
// core count can be 4, 8, or 16
//-Core count 4
-Core count 8
//-Core count 16
-Cache level (L2/L3) - "L2"

-Print level (DETAILED, CONCISE) - "DETAILED"
//-Print level (DETAILED, CONCISE) - "CONCISE"
```

Question 3

Consider the Vector add program (Z[i] = X[i] + Y[i]) and compile it with and without optimization for your machine. See the differences in the assembly code generated with and without the optimizations. Study the loop unrolling gcc can do. Find the optimal unrolling factor for an array size of 16384.

Solution:

The code for vector addition can is as follows:

Question 4 [McPAT]

For the processor on your laptop/smartphone generate two pie-charts: (1) Individual component Power consumption, (2) Individual component area. Modify the relevant XML file and present the table of parameters that you have modified in the input XML files. These parameters can be obtained from the datasheet of your processor.

Solution:

The data sheet of the processor we use is available here. Our machine has been listed as having a 12 way set associative cache, but that is flagged as a wrong input in McPat. So the set associativity is assumed to be 16 and the simulation conducted.

The output that we obtain from running the simulation for our processor is:

```
McPAT (version 1.3 of Feb, 2015) is computing the target processor...


McPAT (version 1.3 of Feb, 2015) results (current print level is 2, please increase print
    level to see the details in components):
*****************************************************************************************
  Technology 22 nm
  Using Long Channel Devices When Appropriate
  Interconnect metal projection= aggressive interconnect technology projection
  Core clock Rate(MHz) 800

*****************************************************************************************
Processor:
  Area = 53.7977 mm^2
  Peak Power = 1268.96 W
  Total Leakage = 1223.81 W
  Peak Dynamic = 45.154 W
  Subthreshold Leakage = 1223.69 W
  Subthreshold Leakage with power gating = 270.849 W
  Gate Leakage = 0.115284 W
  Runtime Dynamic = 18.9517 W

  Total Cores: 2 cores
  Device Type= ITRS low operating power device type
    Area = 46.4116 mm^2
    Peak Dynamic = 43.1819 W
    Subthreshold Leakage = 1222.28 W
    Subthreshold Leakage with power gating = 270.077 W
    Gate Leakage = 0.114317 W
    Runtime Dynamic = 18.1601 W

  Total L3s:
  Device Type= ITRS high performance device type
    Area = 6.70741 mm^2
    Peak Dynamic = 0.533555 W
    Subthreshold Leakage = 1.33954 W
    Subthreshold Leakage with power gating = 0.736577 W
    Gate Leakage = 0.000772985 W
    Runtime Dynamic = 0.114641 W

  Total NoCs (Network/Bus):
  Device Type= ITRS low operating power device type
    Area = 0.678706 mm^2
    Peak Dynamic = 1.43853 W
    Subthreshold Leakage = 0.0775544 W
    Subthreshold Leakage with power gating = 0.0348995 W
    Gate Leakage = 0.000193836 W
    Runtime Dynamic = 0.676954 W

*****************************************************************************************
Core:
      Area = 23.2058 mm^2
      Peak Dynamic = 21.591 W
      Subthreshold Leakage = 611.139 W
      Subthreshold Leakage with power gating = 135.039 W
      Gate Leakage = 0.0571584 W
      Runtime Dynamic = 18.1601 W

      Instruction Fetch Unit:
        Area = 1.66719 mm^2
        Peak Dynamic = 0.907901 W
        Subthreshold Leakage = 110.908 W
        Subthreshold Leakage with power gating = 25.8079 W
        Gate Leakage = 0.0103333 W
        Runtime Dynamic = 1.62892 W

      Renaming Unit:
        Area = 0.814187 mm^2
        Peak Dynamic = 2.45696 W
        Subthreshold Leakage = 72.704 W
        Subthreshold Leakage with power gating = 15.7724 W
        Gate Leakage = 0.00902435 W
        Runtime Dynamic = 4.72177 W

      Load Store Unit:
        Area = 1.23174 mm^2
        Peak Dynamic = 0.805525 W
        Subthreshold Leakage = 85.5795 W
        Subthreshold Leakage with power gating = 18.752 W
        Gate Leakage = 0.00961086 W
        Runtime Dynamic = 1.46138 W

      Memory Management Unit:
        Area = 0.806636 mm^2
        Peak Dynamic = 1.45602 W
        Subthreshold Leakage = 73.8247 W
        Subthreshold Leakage with power gating = 15.9461 W
        Runtime Dynamic = 2.97498 W

      Execution Unit:
        Area = 13.7949 mm^2
        Peak Dynamic = 14.1981 W
        Subthreshold Leakage = 133.725 W
        Subthreshold Leakage with power gating = 29.3968 W
        Runtime Dynamic = 6.83373 W

    L2
    Area = 1.57142 mm^2
    Peak Dynamic = 1.76641 W
    Subthreshold Leakage = 2.46711 W
    Subthreshold Leakage with power gating = 0.866448 W
    Gate Leakage = 0.000574075 W
    Runtime Dynamic = 1.07871 W

*****************************************************************************************
      L3
      Area = 6.70741 mm^2
      Peak Dynamic = 0.533555 W
      Subthreshold Leakage = 1.33954 W
      Subthreshold Leakage with power gating = 0.736577 W
      Gate Leakage = 0.000772985 W
      Runtime Dynamic = 0.114641 W

*****************************************************************************************
BUSES
      Area = 0.678706 mm^2
      Peak Dynamic = 1.43853 W
      Subthreshold Leakage = 0.0775544 W
      Subthreshold Leakage with power gating = 0.0348995 W
      Gate Leakage = 0.000193836 W
      Runtime Dynamic = 0.676954 W

      Bus:
        Area = 0.678706 mm^2
        Peak Dynamic = 1.43853 W
        Subthreshold Leakage = 0.0775544 W
        Subthreshold Leakage with power gating = 0.0348995 W
        Gate Leakage = 0.000193836 W
        Runtime Dynamic = 0.676954 W

*****************************************************************************************
```

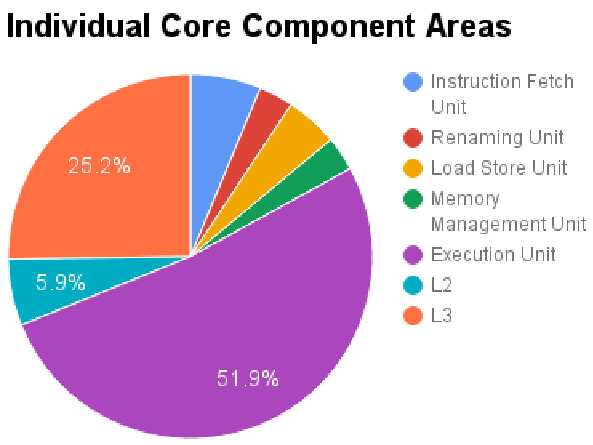
The parameters that we have modified as given by $diff$ are:

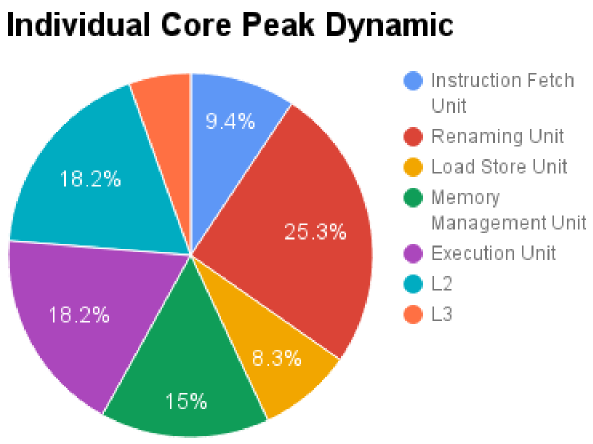
```
19,20c19,20
```

```
<       <param name=”core_tech_node” value=”22”/><!-- nm -->
<       <param name=”target_core_clockrate” value=”1600”/><!--MHz -->
---
>       <param name=”core_tech_node” value=”65”/><!-- nm -->
>       <param name=”target_core_clockrate” value=”3400”/><!--MHz -->
24c24
<       <param name=”device_type” value=”2”/><!--0: HP(High Performance Type); 1: LSTP(Low standby power) 2: LOP (Low Operating Power)  -->
---
>       <param name=”device_type” value=”0”/><!--0: HP(High Performance Type); 1: LSTP(Low standby power) 2: LOP (Low Operating Power)  -->
41c41
<           <param name=”clock_rate” value=”800”/>
---
>           <param name=”clock_rate” value=”3400”/>
51c51
<           <param name=”number_hardware_threads” value=”4”/>
---
>           <param name=”number_hardware_threads” value=”2”/>
67c67
<           <param name=”fp_issue_width” value=”4”/>
---
>           <param name=”fp_issue_width” value=”2”/>
72c72
<           <param name=”pipelines_per_core” value=”19,14”/>
---
>           <param name=”pipelines_per_core” value=”1,1”/>
77c77
<           <param name=”ALU_per_core” value=”4”/>
---
>           <param name=”ALU_per_core” value=”6”/>
79c79
<           <param name=”MUL_per_core” value=”2”/>
---
>           <param name=”MUL_per_core” value=”1”/>
237c237
<               <param name=”number_entries” value=”64”/><!--dual threads-->
---
>               <param name=”number_entries” value=”128”/><!--dual threads-->
265c265
<               <param name=”Dir_config” value=”64000,64,8,1,100,100, 8”/>
---
>               <param name=”Dir_config” value=”4096,2,0,1,100,100, 8”/>
267c267
<               <param name=”buffer_sizes” value=”128, 128, 128, 128”/>
---
>               <param name=”buffer_sizes” value=”8, 8, 8, 8”/>
308c308
<               <param name=”L2_config” value=”512000,64, 8, 8, 8, 23, 32, 1”/>
---
>               <param name=”L2_config” value=”1048576,32, 8, 8, 8, 23, 32, 1”/>
310c310
<               <param name=”buffer_sizes” value=”64, 64, 64, 64”/>
---
>               <param name=”buffer_sizes” value=”16, 16, 16, 16”/>
328c328
<               <param name=”L3_config” value=”3000000,64,16, 8, 16, 100,1”/>
---
>               <param name=”L3_config” value=”16777216,64,16, 16, 16, 100,1”/>
336c336
<               <param name=”buffer_sizes” value=”32, 32, 32, 32”/>
---
>               <param name=”buffer_sizes” value=”16, 16, 16, 16”/>
381,382c381,382
<           <param name=”type” value=”1”/> <!-- 1: low power; 0 high performance -->
<           <param name=”mc_clock” value=”1600”/><!--DIMM IO bus clock rate MHz-->
---
>           <param name=”type” value=”0”/> <!-- 1: low power; 0 high performance -->
>           <param name=”mc_clock” value=”200”/><!--DIMM IO bus clock rate MHz-->
395c395
<           <param name=”databus_width” value=”256”/>
---
>           <param name=”databus_width” value=”128”/>
411c411
<           <param name=”clockrate” value=”833”/>
---
```

```
>           <param name="clockrate" value="350"/>
425c425
<           <param name="type" value="1"/> <!-- 1: low power; 0 high performance -->
---
>           <param name="type" value="0"/> <!-- 1: low power; 0 high performance -->
431c431
<           <param name="num_channels" value="32"/> <!-- 2 ,4 ,8 ,16 ,32 -->
---
>           <param name="num_channels" value="8"/> <!-- 2 ,4 ,8 ,16 ,32 -->
```

The corresponding graphs for the individual power consumption and individual area are:

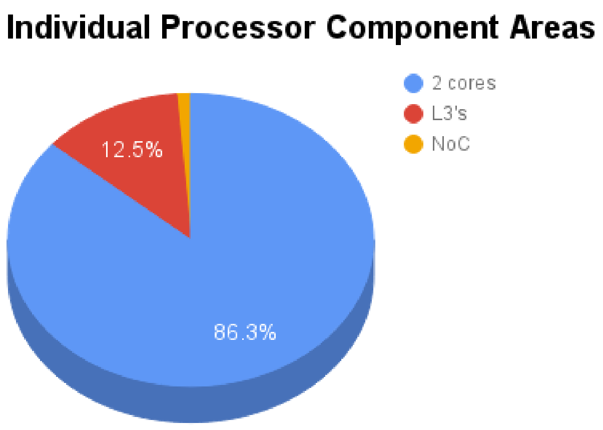


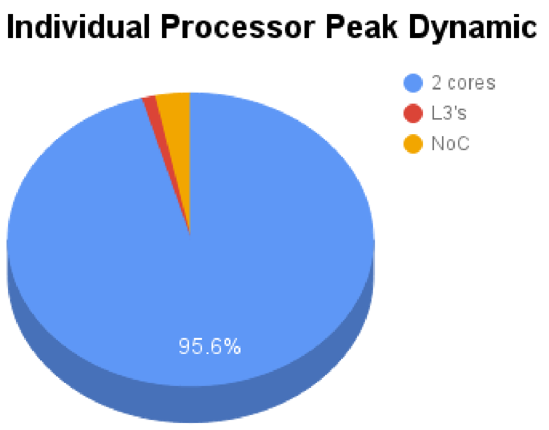


Question 5 [ORION]

Estimate power and area of standard on-chip network routers. Present the area and power results of the following runs.

- The default values from SIM port.h file.
- A Router implemented on 45nm technology, running at 0.9V, built with HVT transistors.





- The default router with 6 input ports, 6 output ports, 128 bit flit width.

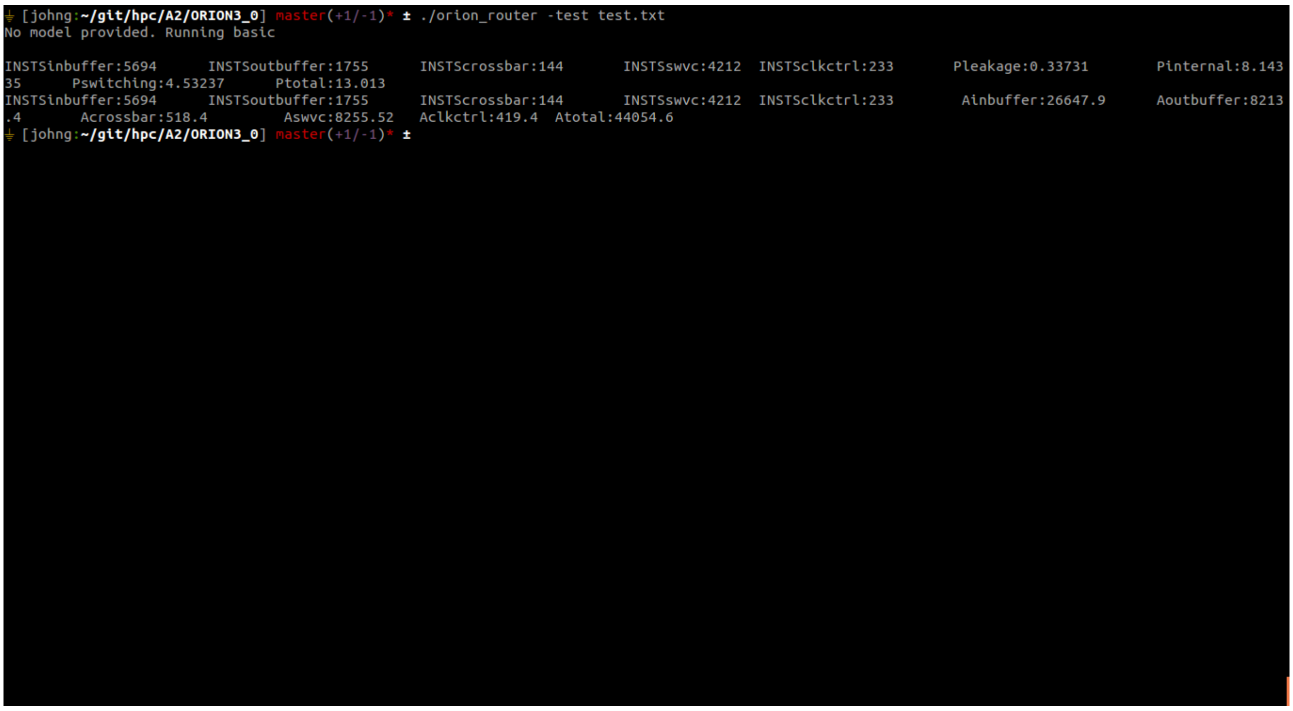
Solution:

The power and area of the on chip network routers in the scenarios are:

- Default values of SIM_port.h

  The associated SIM_port.h file:

```
/*************************************************************
*Unit :
*    voltage : V
*    frequency : Hz
*
*************************************************************/

#ifndef _SIM_PORT_H
#define _SIM_PORT_H
/*Technology related parameters */
#define PARM_TECH_POINT          65
#define PARM_TRANSISTOR_TYPE     NVT    /* transistor type, HVT, NVT, or LVT */
#define PARM_Vdd                 1.0
#define PARM_Freq                0.746e9
#define PARM_VDD_V               PARM_Vdd
```



```
⏚ [johng:~/git/hpc/A2/ORION3_0] master(+1/-1)* ± ./orion_router -test test.txt
No model provided. Running basic

INSTSinbuffer:5694      INSTSoutbuffer:1755      INSTScrossbar:144       INSTSswvc:4212  INSTSclkctrl:233       Pleakage:0.33731       Pinternal:8.143
35      Pswitching:4.53237      Ptotal:13.013
INSTSinbuffer:5694      INSTSoutbuffer:1755      INSTScrossbar:144       INSTSswvc:4212  INSTSclkctrl:233        Ainbuffer:26647.9      Aoutbuffer:8213
.4       Acrossbar:518.4         Aswvc:8255.52   Aclkctrl:419.4  Atotal:44054.6
⏚ [johng:~/git/hpc/A2/ORION3_0] master(+1/-1)* ±
```

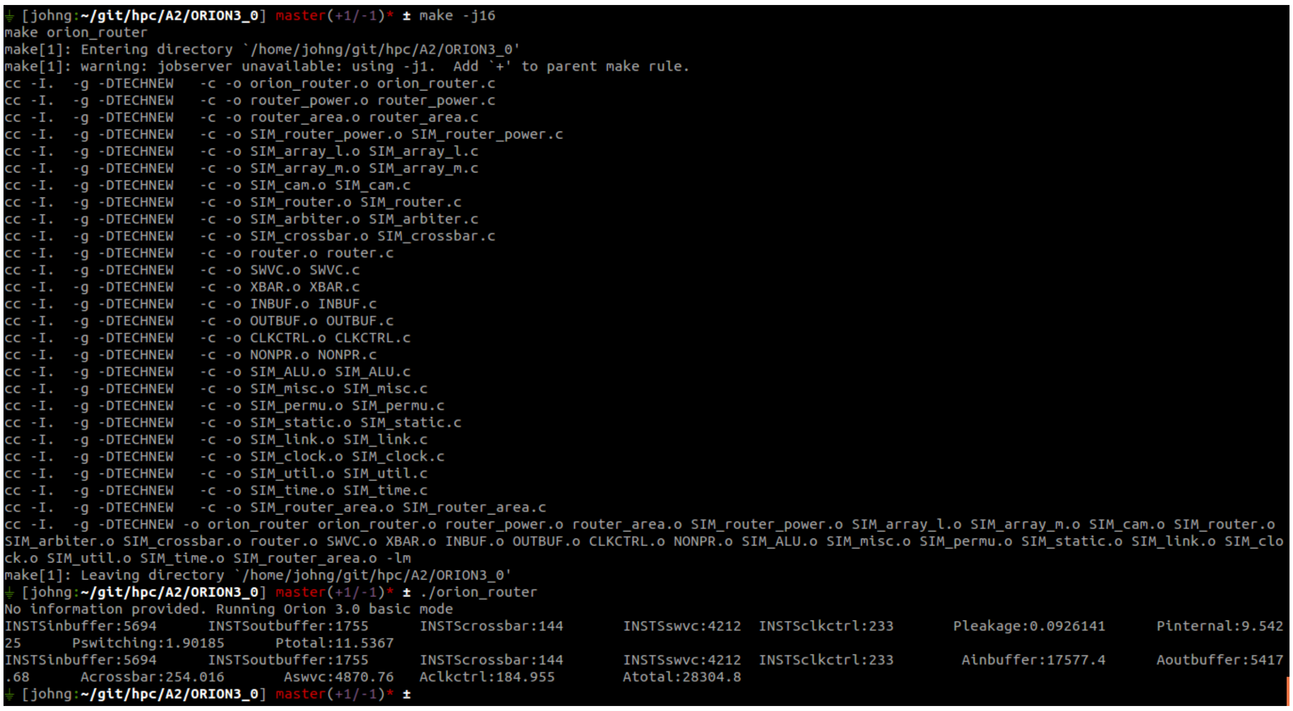
Figure 7: Default run

```
#define PARM_FREQ_Hz                PARM_Freq
#define PARM_tr                     .2

/* router module parameters */
/* general parameters */
#define PARM_in_port          3    /* # of router input ports */
#define PARM_cache_in_port    0    /* # of cache input ports */
#define PARM_mc_in_port       0    /* # of memory controller input ports */
#define PARM_io_in_port       0    /* # of I/O device input ports */
#define PARM_out_port         3
#define PARM_cache_out_port   0    /* # of cache output ports */
#define PARM_mc_out_port      0    /* # of memory controller output ports */
#define PARM_io_out_port      0    /* # of I/O device output ports */
#define PARM_flit_width       16   /* flit width in bits */

/* virtual channel parameters */
#define PARM_v_class          1    /* # of total message classes */
#define PARM_v_channel        7    /* # of virtual channels per virtual message class*/
#define PARM_cache_class      0    /* # of cache port virtual classes */
#define PARM_mc_class         0    /* # of memory controller port virtual classes */
#define PARM_io_class         0    /* # of I/O device port virtual classes */
/* ?? */
#define PARM_in_share_buf     0    /* do input virtual channels physically share buffers? */
#define PARM_out_share_buf    0    /* do output virtual channels physically share buffers? */
/* ?? */
#define PARM_in_share_switch      1    /* do input virtual channels share crossbar input ports?
    */
#define PARM_out_share_switch     1    /* do output virtual channels share crossbar output ports?
    */

/* crossbar parameters */
#define PARM_crossbar_model  TRISTATE_CROSSBAR    /* crossbar model type MATRIX_CROSSBAR,
    MULTREE_CROSSBAR, or TRISTATE_CROSSBAR (only for Orion3.0)*/
#define PARM_crsbar_degree   4                    /* crossbar mux degree */
#define PARM_connect_type    TRISTATE_GATE        /* crossbar connector type */
#define PARM_trans_type      NP_GATE              /* crossbar transmission gate type */
#define PARM_crossbar_in_len      0               /* crossbar input line length, if known */
#define PARM_crossbar_out_len     0               /* crossbar output line length, if known */
#define PARM_xb_in_seg            0
#define PARM_xb_out_seg           0
/* HACK HACK HACK */
#define PARM_exp_xb_model    SIM_NO_MODEL    /* the other parameter is MATRIX_CROSSBAR */
#define PARM_exp_in_seg      2
#define PARM_exp_out_seg     2
```

```

/* input buffer parameters */
#define PARM_in_buf          1        /* have input buffer? */
#define PARM_in_buf_set      2
#define PARM_in_buf_rport    1        /* # of read ports */
#define PARM_in_buffer_type REGISTER     /*buffer model type, SRAM or REGISTER*/

#define PARM_cache_in_buf        0
#define PARM_cache_in_buf_set    0
#define PARM_cache_in_buf_rport  0

#define PARM_mc_in_buf           0
#define PARM_mc_in_buf_set       0
#define PARM_mc_in_buf_rport     0

#define PARM_io_in_buf           0
#define PARM_io_in_buf_set       0
#define PARM_io_in_buf_rport     0

/* output buffer parameters */
#define PARM_out_buf             0
#define PARM_out_buf_set         2
#define PARM_out_buf_wport       1
#define PARM_out_buffer_type     SRAM         /*buffer model type, SRAM or REGISTER*/

/* central buffer parameters */
#define PARM_central_buf     0        /* have central buffer? */
#define PARM_cbuf_set        2560     /* # of rows */
#define PARM_cbuf_rport      2        /* # of read ports */
#define PARM_cbuf_wport      2        /* # of write ports */
#define PARM_cbuf_width      4        /* # of flits in one row */
#define PARM_pipe_depth      4    /* # of banks */

/* array parameters shared by various buffers */
#define PARM_wordline_model CACHE_RW_WORDLINE
#define PARM_bitline_model  RW_BITLINE
#define PARM_mem_model      NORMAL_MEM
#define PARM_row_dec_model  GENERIC_DEC
#define PARM_row_dec_pre_model  SINGLE_OTHER
#define PARM_col_dec_model  SIM_NO_MODEL
#define PARM_col_dec_pre_model  SIM_NO_MODEL
#define PARM_mux_model      SIM_NO_MODEL
#define PARM_outdrv_model   REG_OUTDRV

/* these 3 should be changed together */
/* use double-ended bitline because the array is too large */
#define PARM_data_end           2
#define PARM_amp_model          GENERIC_AMP
#define PARM_bitline_pre_model  EQU_BITLINE
//#define PARM_data_end         1
//#define PARM_amp_model        SIM_NO_MODEL
//#define PARM_bitline_pre_model    SINGLE_OTHER

/* switch allocator arbiter parameters */
#define PARM_sw_in_arb_model     RR_ARBITER  /* input side arbiter model type, MATRIX_ARBITER ,
    RR_ARBITER, QUEUE_ARBITER*/
#define PARM_sw_in_arb_ff_model NEG_DFF          /* input side arbiter flip-flop model type */
#define PARM_sw_out_arb_model    RR_ARBITER  /* output side arbiter model type, MATRIX_ARBITER
    */
#define PARM_sw_out_arb_ff_model     NEG_DFF      /* output side arbiter flip-flop model type */

/* virtual channel allocator arbiter parameters */
#define PARM_vc_allocator_type  TWO_STAGE_ARB    /*vc allocator type, ONE_STAGE_ARB,
    TWO_STAGE_ARB, VC_SELECT*/
#define PARM_vc_in_arb_model     RR_ARBITER  /*input side arbiter model type for TWO_STAGE_ARB.
    MATRIX_ARBITER, RR_ARBITER, QUEUE_ARBITER*/
#define PARM_vc_in_arb_ff_model      NEG_DFF      /* input side arbiter flip-flop model type */
#define PARM_vc_out_arb_model    RR_ARBITER  /*output side arbiter model type (for both
    ONE_STAGE_ARB and TWO_STAGE_ARB). MATRIX_ARBITER, RR_ARBITER, QUEUE_ARBITER */
#define PARM_vc_out_arb_ff_model    NEG_DFF      /* output side arbiter flip-flop model type */
#define PARM_vc_select_buf_type      REGISTER     /* vc_select buffer type, SRAM or REGISTER */

/*link wire parameters*/
#define WIRE_LAYER_TYPE             GLOBAL /*wire layer type, INTERMEDIATE or GLOBAL*/
#define PARM_width_spacing          DWIDTH_DSPACE    /*choices are SWIDTH_SSPACE, SWIDTH_DSPACE,
    DWIDTH_SSPACE, DWIDTH_DSPACE*/
```

```
#define PARM_buffering_scheme    MIN_DELAY        /* choices are MIN_DELAY, STAGGERED */
#define PARM_shielding           FALSE            /* choices are TRUE, FALSE */

/* clock power parameters*/
#define PARM_pipeline_stages     4       /*number of pipeline stages*/
#define PARM_H_tree_clock        0       /*1 means calculate H_tree_clock power, 0 means not
    calculate H_tree_clock*/
#define PARM_router_diagonal     442     /*router diagonal in micro-meter */

/* RF module parameters */
#define PARM_read_port  1
#define PARM_write_port 1
#define PARM_n_regs  64
#define PARM_reg_width  32

#define PARM_ndwl    1
#define PARM_ndbl    1
#define PARM_nspd    1

#define PARM_POWER_STATS      1

#endif  /* _SIM_PORT_H */
```

- HVT Transistors



```
⏚ [johng:~/git/hpc/A2/ORION3_0] master(+1/-1)* ± make -j16
make orion_router
make[1]: Entering directory `/home/johng/git/hpc/A2/ORION3_0'
make[1]: warning: jobserver unavailable: using -j1.  Add `+' to parent make rule.
cc -I.  -g -DTECHNEW   -c -o orion_router.o orion_router.c
cc -I.  -g -DTECHNEW   -c -o router_power.o router_power.c
cc -I.  -g -DTECHNEW   -c -o router_area.o router_area.c
cc -I.  -g -DTECHNEW   -c -o SIM_router_power.o SIM_router_power.c
cc -I.  -g -DTECHNEW   -c -o SIM_array_l.o SIM_array_l.c
cc -I.  -g -DTECHNEW   -c -o SIM_array_m.o SIM_array_m.c
cc -I.  -g -DTECHNEW   -c -o SIM_cam.o SIM_cam.c
cc -I.  -g -DTECHNEW   -c -o SIM_router.o SIM_router.c
cc -I.  -g -DTECHNEW   -c -o SIM_arbiter.o SIM_arbiter.c
cc -I.  -g -DTECHNEW   -c -o SIM_crossbar.o SIM_crossbar.c
cc -I.  -g -DTECHNEW   -c -o router.o router.c
cc -I.  -g -DTECHNEW   -c -o SWVC.o SWVC.c
cc -I.  -g -DTECHNEW   -c -o XBAR.o XBAR.c
cc -I.  -g -DTECHNEW   -c -o INBUF.o INBUF.c
cc -I.  -g -DTECHNEW   -c -o OUTBUF.o OUTBUF.c
cc -I.  -g -DTECHNEW   -c -o CLKCTRL.o CLKCTRL.c
cc -I.  -g -DTECHNEW   -c -o NONPR.o NONPR.c
cc -I.  -g -DTECHNEW   -c -o SIM_ALU.o SIM_ALU.c
cc -I.  -g -DTECHNEW   -c -o SIM_misc.o SIM_misc.c
cc -I.  -g -DTECHNEW   -c -o SIM_permu.o SIM_permu.c
cc -I.  -g -DTECHNEW   -c -o SIM_static.o SIM_static.c
cc -I.  -g -DTECHNEW   -c -o SIM_link.o SIM_link.c
cc -I.  -g -DTECHNEW   -c -o SIM_clock.o SIM_clock.c
cc -I.  -g -DTECHNEW   -c -o SIM_util.o SIM_util.c
cc -I.  -g -DTECHNEW   -c -o SIM_time.o SIM_time.c
cc -I.  -g -DTECHNEW   -c -o SIM_router_area.o SIM_router_area.c
cc -I.  -g -DTECHNEW -o orion_router orion_router.o router_power.o router_area.o SIM_router_power.o SIM_array_l.o SIM_array_m.o SIM_cam.o SIM_router.o
SIM_arbiter.o SIM_crossbar.o router.o SWVC.o XBAR.o INBUF.o OUTBUF.o CLKCTRL.o NONPR.o SIM_ALU.o SIM_misc.o SIM_permu.o SIM_static.o SIM_link.o SIM_clo
ck.o SIM_util.o SIM_time.o SIM_router_area.o -lm
make[1]: Leaving directory `/home/johng/git/hpc/A2/ORION3_0'
⏚ [johng:~/git/hpc/A2/ORION3_0] master(+1/-1)* ± ./orion_router
No information provided. Running Orion 3.0 basic mode
INSTSinbuffer:5694      INSTSoutbuffer:1755      INSTScrossbar:144       INSTSswvc:4212  INSTSclkctrl:233       Pleakage:0.0926141      Pinternal:9.542
25      Pswitching:1.90185      Ptotal:11.5367
INSTSinbuffer:5694      INSTSoutbuffer:1755      INSTScrossbar:144       INSTSswvc:4212  INSTSclkctrl:233        Ainbuffer:17577.4      Aoutbuffer:5417
.68      Acrossbar:254.016       Aswvc:4870.76   Aclkctrl:184.955        Atotal:28304.8
⏚ [johng:~/git/hpc/A2/ORION3_0] master(+1/-1)* ±
```

Figure 8: 45nm technology

The associated SIM_port.h file:

```
/*****************************************************************
* Unit :
*    voltage : V
*    frequency : Hz
*
*****************************************************************/

#ifndef _SIM_PORT_H
#define _SIM_PORT_H
/* Technology related parameters */
#define PARM_TECH_POINT          45
#define PARM_TRANSISTOR_TYPE     HVT    /* transistor type, HVT, NVT, or LVT */
#define PARM_Vdd                 0.9
#define PARM_Freq                0.746e9
#define PARM_VDD_V               PARM_Vdd
#define PARM_FREQ_Hz             PARM_Freq
```

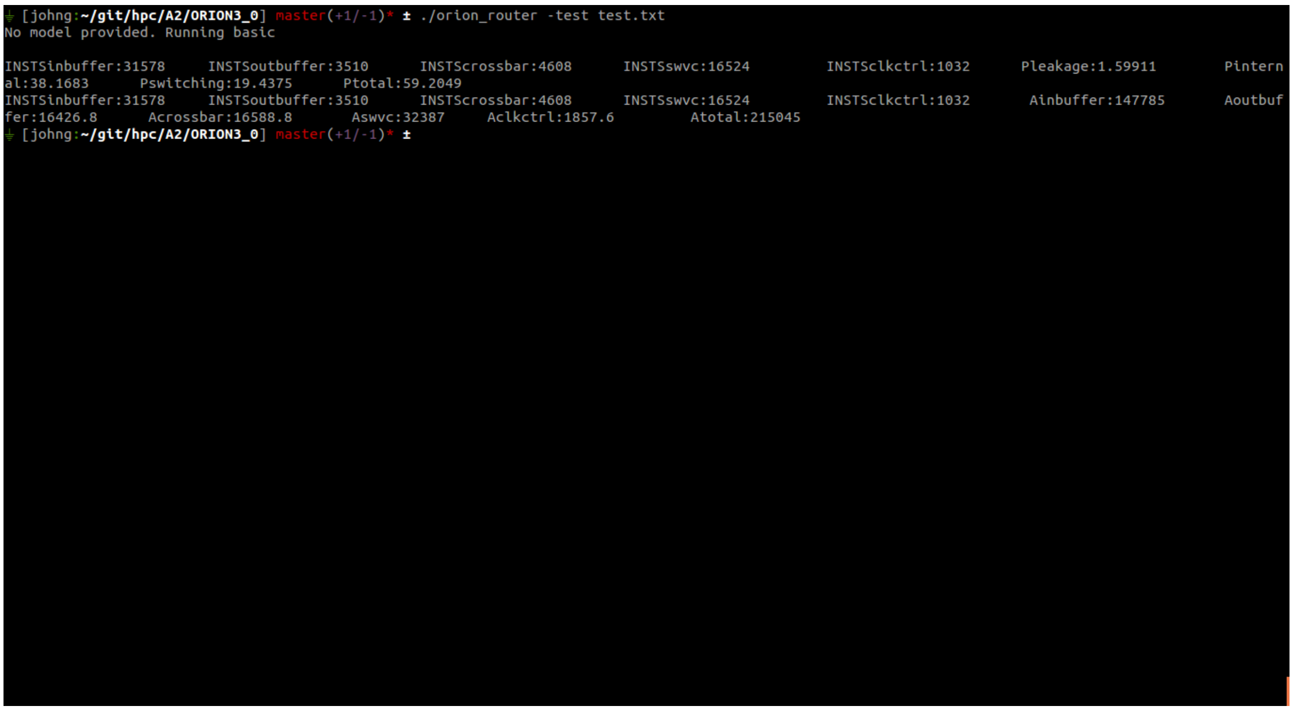
```
#define PARM_tr                  .2

/* router module parameters */
/* general parameters */
#define PARM_in_port         3    /* # of router input ports */
#define PARM_cache_in_port   0    /* # of cache input ports */
#define PARM_mc_in_port      0    /* # of memory controller input ports */
#define PARM_io_in_port      0    /* # of I/O device input ports */
#define PARM_out_port        3
#define PARM_cache_out_port  0    /* # of cache output ports */
#define PARM_mc_out_port     0    /* # of memory controller output ports */
#define PARM_io_out_port     0    /* # of I/O device output ports */
#define PARM_flit_width      16   /* flit width in bits */

/* virtual channel parameters */
#define PARM_v_class         1    /* # of total message classes */
#define PARM_v_channel       7    /* # of virtual channels per virtual message class*/
#define PARM_cache_class     0    /* # of cache port virtual classes */
#define PARM_mc_class        0    /* # of memory controller port virtual classes */
#define PARM_io_class        0    /* # of I/O device port virtual classes */
/* ?? */
#define PARM_in_share_buf    0    /* do input virtual channels physically share buffers? */
#define PARM_out_share_buf   0    /* do output virtual channels physically share buffers? */
/* ?? */
#define PARM_in_share_switch     1    /* do input virtual channels share crossbar input ports?
     */
#define PARM_out_share_switch    1    /* do output virtual channels share crossbar output ports?
     */

/* crossbar parameters */
#define PARM_crossbar_model TRISTATE_CROSSBAR    /* crossbar model type MATRIX_CROSSBAR,
      MULTREE_CROSSBAR, or TRISTATE_CROSSBAR (only for Orion3.0)*/
#define PARM_crsbar_degree  4                    /* crossbar mux degree */
#define PARM_connect_type   TRISTATE_GATE        /* crossbar connector type */
#define PARM_trans_type     NP_GATE              /* crossbar transmission gate type */
#define PARM_crossbar_in_len     0               /* crossbar input line length, if known */
#define PARM_crossbar_out_len    0               /* crossbar output line length, if known */
#define PARM_xb_in_seg           0
#define PARM_xb_out_seg          0
/* HACK HACK HACK */
#define PARM_exp_xb_model   SIM_NO_MODEL   /* the other parameter is MATRIX_CROSSBAR */
#define PARM_exp_in_seg     2
#define PARM_exp_out_seg    2

/* input buffer parameters */
#define PARM_in_buf         1        /* have input buffer? */
#define PARM_in_buf_set     2
#define PARM_in_buf_rport   1        /* # of read ports */
#define PARM_in_buffer_type REGISTER     /*buffer model type, SRAM or REGISTER*/

#define PARM_cache_in_buf        0
#define PARM_cache_in_buf_set    0
#define PARM_cache_in_buf_rport  0

#define PARM_mc_in_buf           0
#define PARM_mc_in_buf_set       0
#define PARM_mc_in_buf_rport     0

#define PARM_io_in_buf           0
#define PARM_io_in_buf_set       0
#define PARM_io_in_buf_rport     0

/* output buffer parameters */
#define PARM_out_buf             0
#define PARM_out_buf_set         2
#define PARM_out_buf_wport       1
#define PARM_out_buffer_type     SRAM         /*buffer model type, SRAM or REGISTER*/

/* central buffer parameters */
#define PARM_central_buf    0        /* have central buffer? */
#define PARM_cbuf_set       2560     /* # of rows */
#define PARM_cbuf_rport     2        /* # of read ports */
#define PARM_cbuf_wport     2        /* # of write ports */
#define PARM_cbuf_width     4        /* # of flits in one row */
#define PARM_pipe_depth     4    /* # of banks */

```

```
/* array parameters shared by various buffers */
#define PARM_wordline_model CACHE_RW_WORDLINE
#define PARM_bitline_model  RW_BITLINE
#define PARM_mem_model      NORMAL_MEM
#define PARM_row_dec_model  GENERIC_DEC
#define PARM_row_dec_pre_model  SINGLE_OTHER
#define PARM_col_dec_model  SIM_NO_MODEL
#define PARM_col_dec_pre_model  SIM_NO_MODEL
#define PARM_mux_model      SIM_NO_MODEL
#define PARM_outdrv_model   REG_OUTDRV

/* these 3 should be changed together */
/* use double-ended bitline because the array is too large */
#define PARM_data_end           2
#define PARM_amp_model          GENERIC_AMP
#define PARM_bitline_pre_model  EQU_BITLINE
//#define PARM_data_end         1
//#define PARM_amp_model        SIM_NO_MODEL
//#define PARM_bitline_pre_model    SINGLE_OTHER

/* switch allocator arbiter parameters */
#define PARM_sw_in_arb_model     RR_ARBITER  /* input side arbiter model type, MATRIX_ARBITER , RR_ARBITER, QUEUE_ARBITER*/
#define PARM_sw_in_arb_ff_model NEG_DFF          /* input side arbiter flip-flop model type */
#define PARM_sw_out_arb_model    RR_ARBITER  /* output side arbiter model type, MATRIX_ARBITER */
#define PARM_sw_out_arb_ff_model     NEG_DFF     /* output side arbiter flip-flop model type */

/* virtual channel allocator arbiter parameters */
#define PARM_vc_allocator_type  TWO_STAGE_ARB    /*vc allocator type, ONE_STAGE_ARB, TWO_STAGE_ARB, VC_SELECT*/
#define PARM_vc_in_arb_model     RR_ARBITER  /*input side arbiter model type for TWO_STAGE_ARB. MATRIX_ARBITER, RR_ARBITER, QUEUE_ARBITER*/
#define PARM_vc_in_arb_ff_model     NEG_DFF      /* input side arbiter flip-flop model type */
#define PARM_vc_out_arb_model    RR_ARBITER  /*output side arbiter model type (for both ONE_STAGE_ARB and TWO_STAGE_ARB). MATRIX_ARBITER, RR_ARBITER, QUEUE_ARBITER */
#define PARM_vc_out_arb_ff_model    NEG_DFF      /* output side arbiter flip-flop model type */
#define PARM_vc_select_buf_type      REGISTER    /* vc_select buffer type, SRAM or REGISTER */

/*link wire parameters*/
#define WIRE_LAYER_TYPE          GLOBAL /*wire layer type, INTERMEDIATE or GLOBAL*/
#define PARM_width_spacing       DWIDTH_DSPACE   /*choices are SWIDTH_SSPACE, SWIDTH_DSPACE, DWIDTH_SSPACE, DWIDTH_DSPACE*/
#define PARM_buffering_scheme    MIN_DELAY       /*choices are MIN_DELAY, STAGGERED */
#define PARM_shielding           FALSE           /*choices are TRUE, FALSE */

/*clock power parameters*/
#define PARM_pipeline_stages     4       /*number of pipeline stages*/
#define PARM_H_tree_clock        0       /*1 means calculate H_tree_clock power, 0 means not calculate H_tree_clock*/
#define PARM_router_diagonal     442     /*router diagonal in micro-meter */

/* RF module parameters */
#define PARM_read_port   1
#define PARM_write_port  1
#define PARM_n_regs 64
#define PARM_reg_width   32

#define PARM_ndwl    1
#define PARM_ndbl    1
#define PARM_nspd    1

#define PARM_POWER_STATS     1

#endif  /* _SIM_PORT_H */
```

- 128 bit flit width

  The associated SIM_port.h file:

```
/*******************************************************************
*Unit :
*   voltage : V
*   frequency : Hz
*
*******************************************************************/
```



```
⏚ [johng:~/git/hpc/A2/ORION3_0] master(+1/-1)* ± ./orion_router -test test.txt
No model provided. Running basic

INSTSinbuffer:31578     INSTSoutbuffer:3510      INSTScrossbar:4608      INSTSswvc:16524         INSTSclkctrl:1032       Pleakage:1.59911        Pintern
al:38.1683      Pswitching:19.4375      Ptotal:59.2049
INSTSinbuffer:31578     INSTSoutbuffer:3510      INSTScrossbar:4608      INSTSswvc:16524         INSTSclkctrl:1032        Ainbuffer:147785       Aoutbuf
fer:16426.8      Acrossbar:16588.8       Aswvc:32387     Aclkctrl:1857.6         Atotal:215045
⏚ [johng:~/git/hpc/A2/ORION3_0] master(+1/-1)* ±
```

Figure 9: 6 input and output ports

```

#ifndef _SIM_PORT_H
#define _SIM_PORT_H
/*Technology related parameters */
#define PARM_TECH_POINT          65
#define PARM_TRANSISTOR_TYPE    NVT    /* transistor type, HVT, NVT, or LVT */
#define PARM_Vdd                 1.0
#define PARM_Freq                0.746e9
#define PARM_VDD_V              PARM_Vdd
#define PARM_FREQ_Hz            PARM_Freq
#define PARM_tr                  .2

/* router module parameters */
/* general parameters */
#define PARM_in_port         6    /* # of router input ports */
#define PARM_cache_in_port   0    /* # of cache input ports */
#define PARM_mc_in_port      0    /* # of memory controller input ports */
#define PARM_io_in_port      0    /* # of I/O device input ports */
#define PARM_out_port        6
#define PARM_cache_out_port  0    /* # of cache output ports */
#define PARM_mc_out_port     0    /* # of memory controller output ports */
#define PARM_io_out_port     0    /* # of I/O device output ports */
#define PARM_flit_width      128 /* flit width in bits */

/* virtual channel parameters */
#define PARM_v_class         1    /* # of total message classes */
#define PARM_v_channel       7    /* # of virtual channels per virtual message class*/
#define PARM_cache_class     0    /* # of cache port virtual classes */
#define PARM_mc_class        0    /* # of memory controller port virtual classes */
#define PARM_io_class        0    /* # of I/O device port virtual classes */
/* ?? */
#define PARM_in_share_buf    0    /* do input virtual channels physically share buffers? */
#define PARM_out_share_buf   0    /* do output virtual channels physically share buffers? */
/* ?? */
#define PARM_in_share_switch     1    /* do input virtual channels share crossbar input ports?
    */
#define PARM_out_share_switch    1    /* do output virtual channels share crossbar output ports?
     */

/* crossbar parameters */
#define PARM_crossbar_model TRISTATE_CROSSBAR    /* crossbar model type MATRIX_CROSSBAR,
    MULTREE_CROSSBAR, or TRISTATE_CROSSBAR (only for Orion3.0)*/
#define PARM_crsbar_degree  4                    /* crossbar mux degree */
#define PARM_connect_type   TRISTATE_GATE        /* crossbar connector type */
```

```
#define PARM_trans_type       NP_GATE              /* crossbar transmission gate type */
#define PARM_crossbar_in_len     0                 /* crossbar input line length, if known */
#define PARM_crossbar_out_len    0                 /* crossbar output line length, if known */
#define PARM_xb_in_seg           0
#define PARM_xb_out_seg          0
/* HACK HACK HACK */
#define PARM_exp_xb_model     SIM_NO_MODEL   /* the other parameter is MATRIX_CROSSBAR */
#define PARM_exp_in_seg       2
#define PARM_exp_out_seg      2

/* input buffer parameters */
#define PARM_in_buf           1        /* have input buffer? */
#define PARM_in_buf_set       2
#define PARM_in_buf_rport     1        /* # of read ports */
#define PARM_in_buffer_type REGISTER     /*buffer model type, SRAM or REGISTER*/

#define PARM_cache_in_buf        0
#define PARM_cache_in_buf_set    0
#define PARM_cache_in_buf_rport  0

#define PARM_mc_in_buf           0
#define PARM_mc_in_buf_set       0
#define PARM_mc_in_buf_rport     0

#define PARM_io_in_buf           0
#define PARM_io_in_buf_set       0
#define PARM_io_in_buf_rport     0

/* output buffer parameters */
#define PARM_out_buf             0
#define PARM_out_buf_set         2
#define PARM_out_buf_wport       1
#define PARM_out_buffer_type     SRAM         /*buffer model type, SRAM or REGISTER*/

/* central buffer parameters */
#define PARM_central_buf      0        /* have central buffer? */
#define PARM_cbuf_set         2560     /* # of rows */
#define PARM_cbuf_rport       2        /* # of read ports */
#define PARM_cbuf_wport       2        /* # of write ports */
#define PARM_cbuf_width       4        /* # of flits in one row */
#define PARM_pipe_depth       4    /* # of banks */

/* array parameters shared by various buffers */
#define PARM_wordline_model CACHE_RW_WORDLINE
#define PARM_bitline_model  RW_BITLINE
#define PARM_mem_model      NORMAL_MEM
#define PARM_row_dec_model  GENERIC_DEC
#define PARM_row_dec_pre_model   SINGLE_OTHER
#define PARM_col_dec_model  SIM_NO_MODEL
#define PARM_col_dec_pre_model   SIM_NO_MODEL
#define PARM_mux_model      SIM_NO_MODEL
#define PARM_outdrv_model   REG_OUTDRV

/* these 3 should be changed together */
/* use double-ended bitline because the array is too large */
#define PARM_data_end            2
#define PARM_amp_model           GENERIC_AMP
#define PARM_bitline_pre_model   EQU_BITLINE
//#define PARM_data_end          1
//#define PARM_amp_model         SIM_NO_MODEL
//#define PARM_bitline_pre_model     SINGLE_OTHER

/* switch allocator arbiter parameters */
#define PARM_sw_in_arb_model     RR_ARBITER  /* input side arbiter model type, MATRIX_ARBITER , RR_ARBITER, QUEUE_ARBITER*/
#define PARM_sw_in_arb_ff_model NEG_DFF          /* input side arbiter flip-flop model type */
#define PARM_sw_out_arb_model    RR_ARBITER  /* output side arbiter model type, MATRIX_ARBITER */
#define PARM_sw_out_arb_ff_model     NEG_DFF      /* output side arbiter flip-flop model type */

/* virtual channel allocator arbiter parameters */
#define PARM_vc_allocator_type  TWO_STAGE_ARB   /*vc allocator type, ONE_STAGE_ARB, TWO_STAGE_ARB, VC_SELECT*/
#define PARM_vc_in_arb_model    RR_ARBITER  /*input side arbiter model type for TWO_STAGE_ARB. MATRIX_ARBITER, RR_ARBITER, QUEUE_ARBITER*/
#define PARM_vc_in_arb_ff_model     NEG_DFF      /* input side arbiter flip-flop model type */
```

```
#define PARM_vc_out_arb_model    RR_ARBITER  /*output side arbiter model type (for both ONE_STAGE_ARB and TWO_STAGE_ARB). MATRIX_ARBITER, RR_ARBITER, QUEUE_ARBITER */
#define PARM_vc_out_arb_ff_model    NEG_DFF      /* output side arbiter flip-flop model type */
#define PARM_vc_select_buf_type      REGISTER     /* vc_select buffer type, SRAM or REGISTER */

/*link wire parameters*/
#define WIRE_LAYER_TYPE           GLOBAL /*wire layer type, INTERMEDIATE or GLOBAL*/
#define PARM_width_spacing        DWIDTH_DSPACE   /*choices are SWIDTH_SSPACE, SWIDTH_DSPACE, DWIDTH_SSPACE, DWIDTH_DSPACE*/
#define PARM_buffering_scheme     MIN_DELAY        /*choices are MIN_DELAY, STAGGERED */
#define PARM_shielding            FALSE            /*choices are TRUE, FALSE */

/*clock power parameters*/
#define PARM_pipeline_stages     4        /*number of pipeline stages*/
#define PARM_H_tree_clock        0        /*1 means calculate H_tree_clock power, 0 means not calculate H_tree_clock*/
#define PARM_router_diagonal     442      /*router diagonal in micro-meter */

/* RF module parameters */
#define PARM_read_port   1
#define PARM_write_port  1
#define PARM_n_regs  64
#define PARM_reg_width   32

#define PARM_ndwl    1
#define PARM_ndbl    1
#define PARM_nspd    1

#define PARM_POWER_STATS      1

#endif  /* _SIM_PORT_H */
```