Compiling core

We will be using one of the existing cores called *pearl* for this exercise.

- Login to one of the servers mentioned
- cd to ~/hive/examples/cores (here you have a list of available cores).
- cd pearl
- Following files are in this folder

```
-bash-3.00$ ll
total 28
-rwxr-xr-x  1 pd0801 pd 2752 Oct 16 15:11 bpe_pearl_is.tim
-rwxr-xr-x  1 pd0801 pd 2454 Oct 16 15:12 bpe_pearl.tim
-rwxr-xr-x  1 pd0801 pd 4569 Oct 16 15:10 bpse_pearl.tim
-rwxr--r--  1 pd0801 pd  511 Oct 16 14:55 Makefile
-rwxr--r--  1 pd0801 pd 4951 Oct 16 15:14 pearl.tim
```

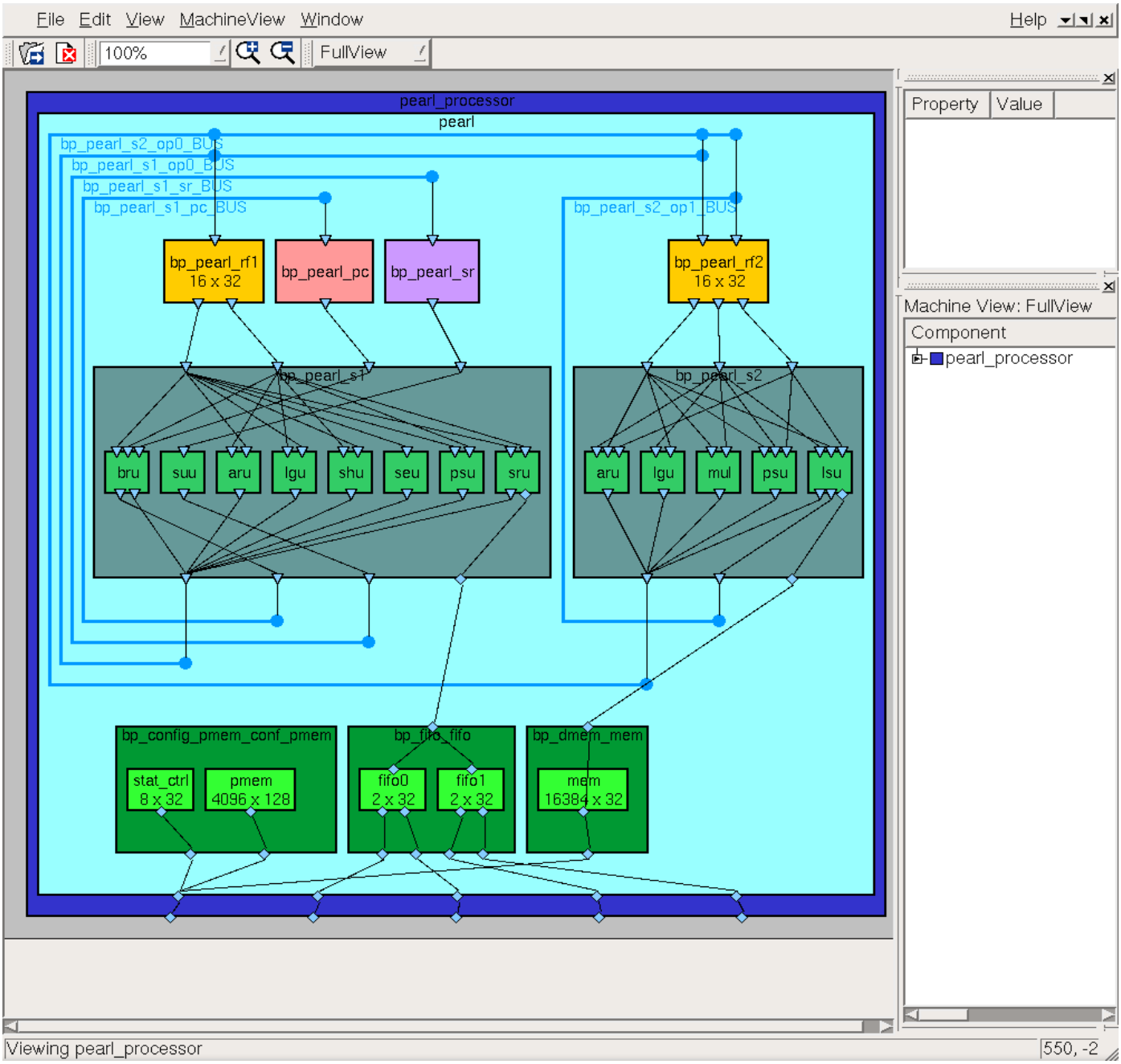
- You can inspect the pearl.tim using any text editor (e.g. vim or nedit). This file is the high-level view of the processor. You can inspect the other included files here.
- Compile this core by using the following command (gmake install). It should take about 2 minutes. You will see the output as follows

```
-bash-3.00$ gmake clean install
/bin/sh: ncroot: command not found
pearl Processor SDK installation started
Compiling machine description
.....
.....
Done building all libraries
pearl LIBC installation complete
pearl_system System SDK installation started
pearl_system System SDK installation complete
```

- A new folder Linux.pearl should have been made in your directory now.
- You can inspect the core by using the corebrowser (Note: you should have X-forwarding enabled for this). The compiled core is present in the path shown below.

```
-bash-3.00$ corebrowser ../../../Linux/cores/pearl/sdk/lib/processor_ast.so &
```

- If the above command gives the error, open the file from the File menu.
- The screen-shot of the corebrowser should look like this



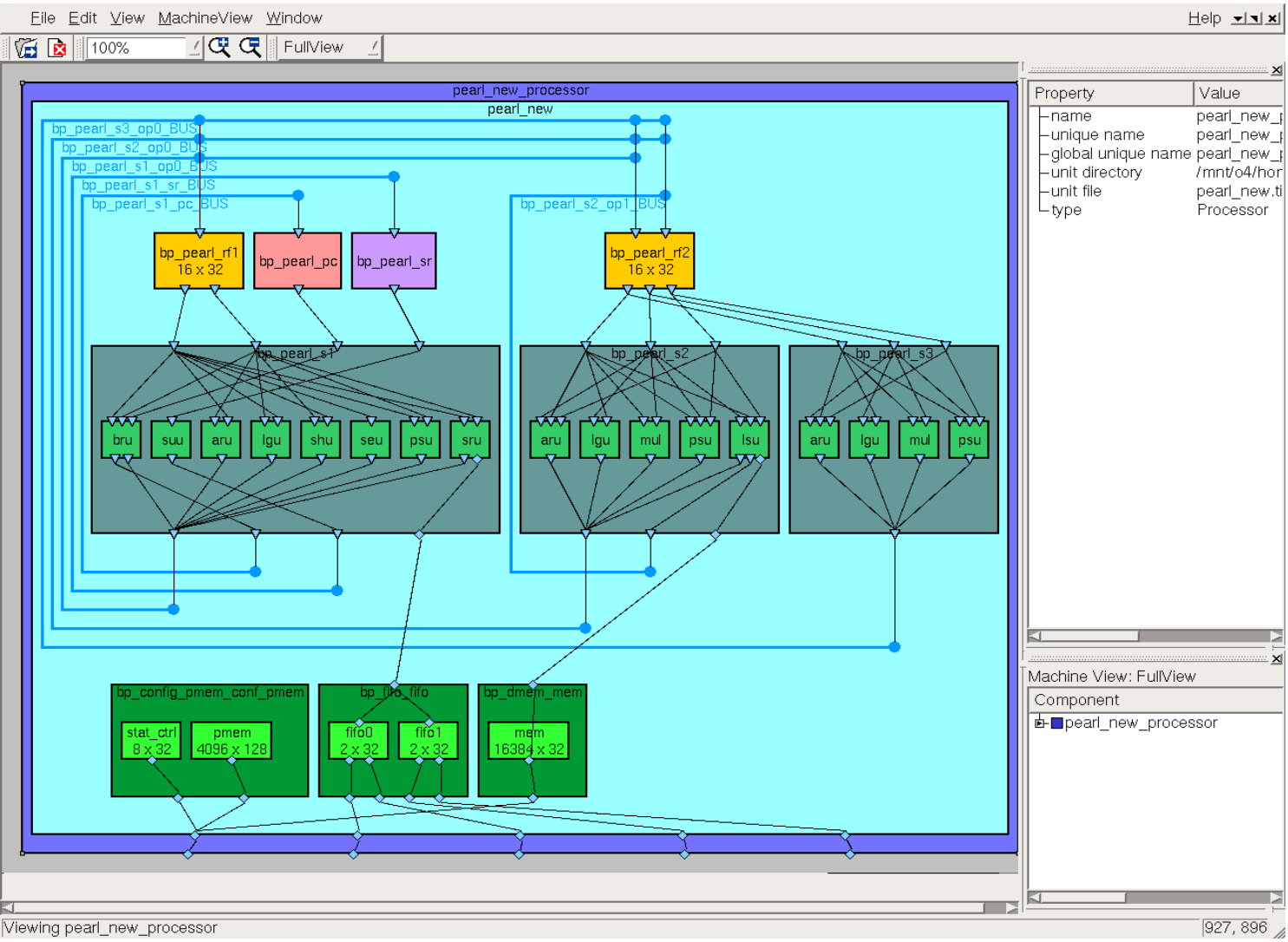
- Make yourself comfortable by clicking on different parts of the core, and seeing their properties in the top-right of corebrowser. Chapter 6 of HiveSDK.pdf gives details on how to use this tool. In particular, notice the operations provided by the different alu's in the two issue slots. For example, the 'mul' unit provides 4 operations – std_mul_iu, std_mul_i, std_mul_u and std_mul.

Question

1) What is the size of program memory and data memory (in bytes)?
2) What operations are supported by the lgu (logic unit)?

Modifying the core – add an issue slot

Let us now see how an issue slot can be added to the earlier core. Following is the screen-shot of the core that we want to have at the end of this exercise.



- Make a new folder in ~/hive/examples/cores/ called pearl_is3 as shown below, and rename **pearl.tim** file as follows:

```
-bash-3.00$ pwd
/home/pd/pd0801/hive/examples/cores
-bash-3.00$ cp -R pearl pearl_is3
-bash-3.00$ ll
total 8
drwxr-xr-x  3 pd0801 pd 4096 Oct 21 15:01 pearl
drwxr-xr-x  3 pd0801 pd 4096 Oct 21 15:04 pearl_is3
-bash-3.00$ cd pearl_is3
-bash-3.00$ mv pearl.tim pearl_is3.tim
-bash-3.00$ rm -rf Linux.pearl
-bash-3.00$ ll
total 28
-rwxr-xr-x  1 pd0801 pd 2752 Oct 21 15:04 bpe_pearl_is.tim
-rwxr-xr-x  1 pd0801 pd 2454 Oct 21 15:04 bpe_pearl.tim
-rwxr-xr-x  1 pd0801 pd 4569 Oct 21 15:04 bpse_pearl.tim
-rwxr--r--  1 pd0801 pd  511 Oct 21 15:04 Makefile
-rwxr--r--  1 pd0801 pd 4951 Oct 21 15:04 pearl_is3.tim
```

Make the following changes

- Makefile: Change all occurrences of **pearl** to **pearl_is3**
- Declare a new issue slot in **bpe_pearl_is.tim** by adding the following lines just before the last **#endif** in the file.

```
IS bpe_pearl_SLOT3
    < signed intWidth, signed immBits >
    ( portW<intWidth> ip0, ip1, ip2)
 -> ( portW<intWidth> op0)
{
  cfu_std_ARU_mod_i1        aru <intWidth, immBits>      ( ip1, ip0, ip2 );
  cfu_std_LGU_i1            lgu <intWidth, immBits>      ( ip1, ip0 );
  cfu_std_MPU_i0_L1         mul <intWidth, immBits>      ( ip0, ip1);

  cfu_std_PSU_3in_i0        psu <intWidth, immBits>      (ip0, ip1, ip2);

  op0 = { aru.op0, lgu.op0, mul.op1, psu.op0};
#if (!defined (BPSE_PEARL_S3_ISOLATE)) || (BPSE_PEARL_S3_ISOLATE==1)
  isolate({aru,1}, {psu,1}, {lgu,1}, {mul,1});
#endif
};
```

- Make the following changes to **bpe_pearl.tim**
  - Add a variable for the new issue slot (line 36):

```
signed s1ImmBits, signed s2ImmBits, signed s3ImmBits,
```

  - Connect the output of register file to the input of new issue slot (line 53,54)

```
RFwc1x2       rf1 <intWidth,rf1Cap>  ( {s1.op0, s2.op0, s3.op0, ip}                    );
RFwc2x3       rf2 <intWidth,rf2Cap>  ( {s1.op0, s2.op0, s3.op0, ip},  {s2.op0, s2.op1,
s3.op0} );
```

  - Declare the new issue slot (after line 57)

```
bpe_pearl_SLOT3 s3 <intWidth,s3ImmBits> (rf2.op0, rf2.op1, rf2.op2);
```

  - Connect the output of the new issue slot (modify line 60)

```
op          = { s1.op0, s2.op0, s3.op0 };
```

- Make the following changes to **bpse_pearl.tim**
  - Add a variable for the new issue slot (line 51):

```
signed s1ImmBits, signed s2ImmBits, signed s3ImmBits,
```

  - Add this variable in the declaration of bpse_pearl (line 70):

```
rf1Cap, rf2Cap, s1ImmBits, s2ImmBits, s3ImmBits,
```

- Make the following changes to **pearl_is3.tim**
  - Rename the processor to **pearl_is3.tim** (line 8):

```
Cell pearl_is3 <signed intWidth, signed llWidth, signed bpFifoWidth>
```

  - Add a variable for the new issue slot (after line 49):

```
signed bpS3ImmBits    := 5;
```

  - Add this variable in the declaration of bpse_pearl (line 64):

```
bpS1ImmBits, bpS2ImmBits, bpS3ImmBits,
```

  - More things to rename (line 79, 88-91):

```
Processor pearl_is3_processor
```

```
pearl_is3 pearl_is3 <intWidth, llWidth, bpFifoWidth> (sl_ip, st_ip0, st_ip1);
sl_op = pearl_is3.sl_op;
st_op0 = pearl_is3.st_op0;
st_op1 = pearl_is3.st_op1;
```

- Compile the processor pearl_is3. Following is the output you will see:

```
-bash-3.00$ gmake install
/bin/sh: ncroot: command not found
pearl_is3 Processor SDK installation started
Compiling machine description
/home/hive/sdk/bin/../include/pbb/semantics/std/sem_std_arith.tim:368:1: warning: std_sat has
mismatch (semantics vs STD_ARU implementation)
.....
.....
Done building all libraries
pearl_is3 LIBC installation complete
pearl_is3_system System SDK installation started
pearl_is3_system System SDK installation complete
```

- Browse the generated processor architecture using **corebrowser** again and answer the questions below.

Questions

3) How many register files do we have and how many registers are there in each?
4) What is the main difference between the second and third issue slots?
5) Can the third issue slot write data directly to the data memory?