Announcements

• Lab 3a
  – Due Today
  – No Lab Interview

• Lab 3b
  – Due 5/29

• Final Lab
  – Final Interview 6/4
  – Due 6/7
Lab 3b

- Implement control logic
- Test all instructions
- Run SuperGarbage Apps
- Performance Evaluation
Test All Instructions

• Write a simple program that uses all the instructions your ISA supports

• Make COE files and test
SuperGarbage

• Did you know?
  – SuperGarbage is a simple VM
  – If you properly implemented SuperGarbage benchmark, you can run any SuperGarbage application!
  – Enables performance comparisons across different ISAs

How to load a SuperGarbage App?
- A big code stub
- I/O interface
device.v

- Feed SuperGarbage Apps
  - select an app (‘out’ ch # 1)
    - 3 SuperGarbage apps will be provided
  - load the app (‘in’ ch # 1)

- Provide a counter
  - for performance measurements
  - ‘out’ ch #2
device.v

// read_addr (in channel):
// 0: reserved
// 1: data
// 2: counter
// others: reserved

// write_addr (out channel):
// 0: reserved
// 1: open a SuperGarbage app (0: app0, 1: app1, ...)
// 2: control the file - Xilinx does not support file control functions
// data 0: reserved
// data 1: set the file pointer to the beginning of the file
// data 2: close the file
// others: reserved. simply print with $display command

module device#(parameter D_WIDTH = 34, PA_WIDTH = 4)(
    input reset,
    input clk,

    input read_req,
    input write_req,
    input [PA_WIDTH-1 : 0] read_addr,
    input [PA_WIDTH-1 : 0] write_addr,
    input [D_WIDTH 1 : 0] din,
    output [D_WIDTH-1 : 0] dout,
    output read_ack,
    output write_ack
);
SuperGarbage Binary File

SuperGarbage(mem, startPC):
Loader

• Implement a loader in your ISA

```c
// SuperGarbage Loader
// app_num
// 0: test0.bin
// 1: test1.bin
// 2: test2.bin

word mem[1024];
word startPC;
word app_num = 1;

// basic code stub
set $SP;
set $GP;

// request an app with app_num
out(app_num, 0x1);

// load data from an external device while(addr != -1)
do {
    word addr, data;
    addr = in(0x1);
    data = in(0x1);

    if (addr == 0x3FFFFFFF) {
        startPC = data;
        break;
    }
    else {
        mem[addr] = data;
    }
} while(true);

// Finally, call SuperGarbage VM
SuperGarbage(mem, startPC);
```
Performance Evaluation

• Counter Feature
  – ‘in’ channel #2
  – return the current global clock counter

• SuperGarbage App
  – get global clock counters
    • A = counter at the beginning
    • B = counter at the end
  – exec cycles = B – A

Sample SuperGarbage App

```plaintext
in #2, 1 // get counter
......
......
<Main Program Here...>
......
......
in #2, 2 // get counter
sub 2, 1, 2
out #3, 2
<EOF>
```
Is SuperGarbage Difficult?

- **SuperGarbageSim**
  - **go**
    - execute instructions
  - **disassembly**
    - show disassembled instructions
  - **getmem**
    - print memory contents
  - **setmem**
    - set memory data
  - **setPC**
    - set the PC
  - **getPC**
    - get the current PC
## SuperGarbage

**Performance Evaluation**

<table>
<thead>
<tr>
<th></th>
<th>app0</th>
<th>app1</th>
<th>app2</th>
</tr>
</thead>
<tbody>
<tr>
<td># of cycles</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cycle time</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>exec time</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Any Questions?