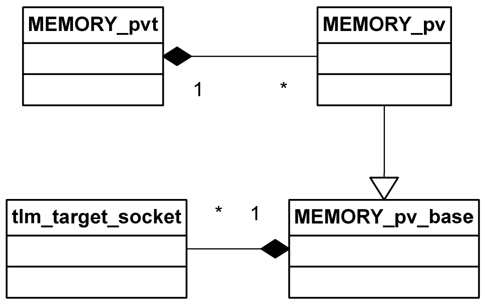
NC State University
ECE Department ECE 720 W. Rhett Davis

Homework #2 Solution

Problem 1)

The MEMORY_pvt class is related to the tlm_target_socket class through the classes MEMORY_pv and MEMORY_pv_base, as shown below. Note that the relationship between MEMORY_pvt and MEMORY_pv is considered composition, rather than aggregation, because the MEMORY_pv instance is created in the MEMORY_pvt constructor and deleted in its destructor.



Problem 2)

The member variables for the **traffic_generator** class are the following:
- const unsigned int
- sc_dt::uint64
- tg_queue_c
- bool
- sc_core::sc_port< sc_core::sc_fifo_out_if< gp_ptr > >
- sc_core::sc_port< sc_core::sc_fifo_in_if< gp_ptr > >

tg_queue_c is a locally defined class, so it won't be used in the **SimpleBusAT** class. Also, note that **gp_ptr** is a *typedef* for **tlm::tlm_generic_payload***.

The member variable for the **SimpleBusAT** class are the following:
- target_socket_type
- initiator_socket_type
- PendingTransactions
- tlm_utils::peq_with_get< transaction_type >
- sc_core::sc_event

Note the following typedefs in this class:
```
  typedef tlm_utils::simple_target_socket_tagged< SimpleBusAT > target_socket_type;
  typedef tlm_utils::simple_initiator_socket_tagged< SimpleBusAT > initiator_socket_type;
  typedef std::map< transaction_type*, ConnectionInfo > Pending Transactions;
  typedef tlm::tlm_generic_payload transaction_type;
```

ConnectionInfo is a locally defined struct, so it won't be used in the **traffic_generator** class.

Ultimately, there is only one attribute-dependency that both of these classes have in common. That is, both classes use the type tlm::tlm_generic_payload, though you wouldn't be able to tell unless you expanded the typedefs.

Problem 3)

Here are the contents of the modified instcount.cpp file. The lines in red are the modified ones.

```
int sc_main(int argc, char* argv[])
{

    map<string,module*> mods;
    string line,first,second,current_module;
    size_t pos;
    ifstream f("LMS_pipe.hier");


    while (f.good()) {
      getline(f,line);
      pos=line.find(' ');
      first = line.substr(0,pos);
      second = line.substr(pos+1);
      //cout << "\"" << first << "\"" << endl;
      //cout << "\"" << second << "\"" << endl;
      if (first == "module") {
        current_module = second;
        mods[current_module]=new module(current_module);
        //cout << "module " << second << endl;
      }
      else if (second != "") {
        mods[current_module]->addInstance(first);
      }

    }
    f.close();

    cout << mods[current_module]->countInstances(mods)
         << " total instances" << endl;

    //map<string,module*>::iterator it;
    //for (it=mods.begin(); it != mods.end(); it++) {
    //  cout << "module " << it->second->name << endl;
    //  it->second->print();
    //  }

    return 0;
}
```

Here is a portion of the modified module.h file. A new method called countInstances was added.

```
class module {
public:
    string name;
    vector<string> instances;
    module (string n) { name = n; }
    void addInstance(string modname) { instances.push_back(modname); }
    void print();
    int countInstances(map<string,module*> &mods);
};
```

Here is the definition of the countInstances() method in the module.cpp file.

```
int module::countInstances(map<string,module*> &mods) {
  int count = 0;
  vector<string>::iterator it;
  for (it=instances.begin(); it < instances.end(); it++) {
    //cout << '"' << *it << "\" " << mods[*it] << endl;
    if (mods[*it] != 0)
      count+= mods[*it]->countInstances(mods);
    else
      count++;
  }
  return count;
}
```

When running this script, 10581 Total instances should be reported for the LMS_pipe.hier file.

Problem 4)

The final program should print the following output:

```
back
dog's
lazy
the
over
jumpped
fox
brown
quick
The
```

The contents of the **reader.h** file are given below:

```
#pragma once

#include <list>
#include <string>

using namespace std;

class reader {
  list<string> words;
public:
  reader( string filename );
  void reversePrint();
};
```

The contents of the **reader.cpp** file are given below:

```
#include "reader.h"
#include <iostream>
#include <fstream>

using namespace std;

reader::reader(string filename) {
  string line;
  size_t pos;
  ifstream f(filename.c_str());

  while (f.good()) {
    getline(f,line);
    while (line.length() > 0) {
      //cout << '"' << line << '"' << endl;
      if (line[0] == ' ')
        line=line.substr(1);
      else {
        pos=line.find(' ');
        if (pos != line.npos) {
          words.push_back(line.substr(0,pos));
          line=line.substr(pos+1);
      }
        else {
          words.push_back(line);
        line="";
      }
      }
    }
  }
  f.close();
}

void reader::reversePrint() {
  words.reverse();
  list<string>::iterator it;
  for (it=words.begin(); it != words.end(); it++) {
    cout << *it << endl;
    }
}
```