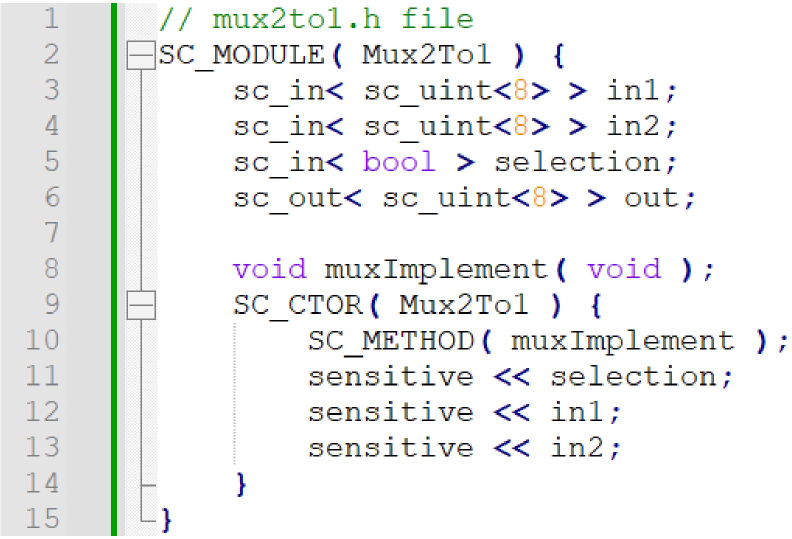
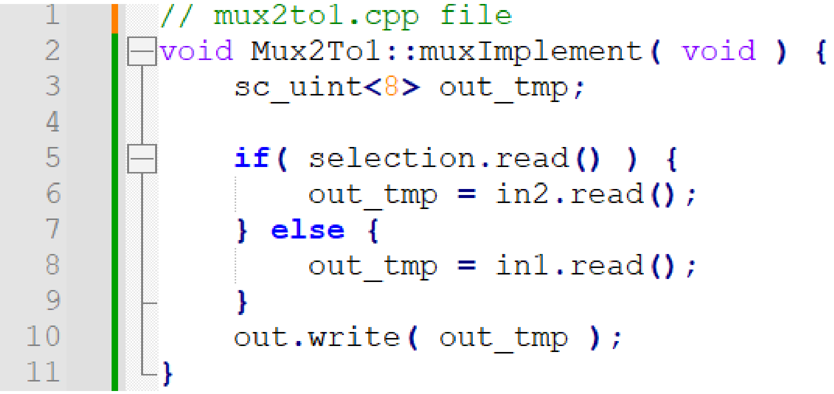


```cpp
// mux2to1.h file
SC_MODULE( Mux2To1 ) {
    sc_in< sc_uint<8> > in1;
    sc_in< sc_uint<8> > in2;
    sc_in< bool > selection;
    sc_out< sc_uint<8> > out;

    void muxImplement( void );
    SC_CTOR( Mux2To1 ) {
        SC_METHOD( muxImplement );
        sensitive << selection;
        sensitive << in1;
        sensitive << in2;
    }
}
```

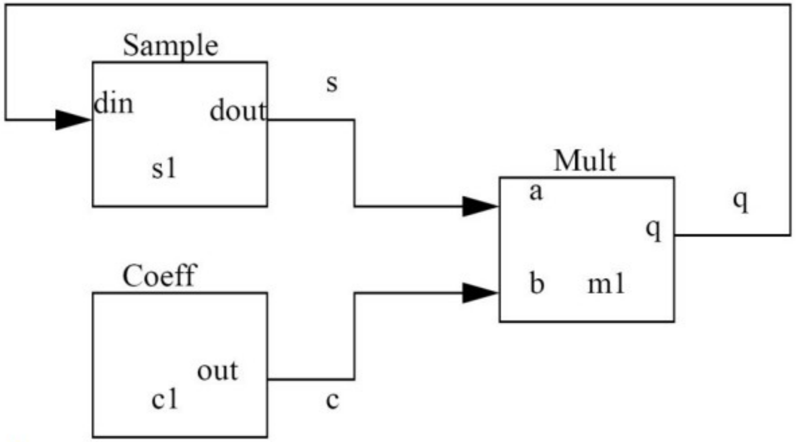


```cpp
// mux2to1.cpp file
void Mux2To1::muxImplement( void ) {
    sc_uint<8> out_tmp;

    if( selection.read() ) {
        out_tmp = in2.read();
    } else {
        out_tmp = in1.read();
    }
    out.write( out_tmp );
}
```

Named and Positional Connections

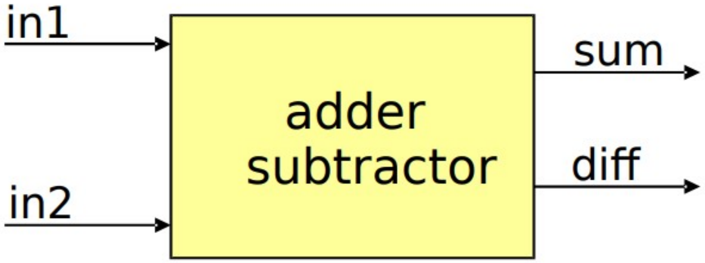
```
SC_MODULE(filter) {
   // Sub-modules: “components
   sample *s1;
   coeff     *c1;
   mult      *m1;
   sc_signal<sc_uint <32> > q,s,c;
    // Constructor :“architecture”
    SC_CTOR(filter) {
    //Sub-modules instantiation/mapping
       s1 = new sample (“s1”);
       s1->din(q);   // named mapping
       s1->dout(s);
       c1 = new coeff(“c1”);
       c1->out(c);   // named mapping
       m1 = new mult (“m1”);
       (*m1)(s, c, q)//positional mapping
   }
}
```



```
SC_MODULE(countsub)
{
   sc_in<double>  in1;
   sc_in<double>  in2;
   sc_out<double> sum;
   sc_out<double> diff;

   void addsub();

   // Constructor:
   SC_CTOR(countsub)
   {
// declare addsub as SC_METHOD
        SC_METHOD(addsub);
        sensitive << in1;
        sensitive << in2;
   }
};
```



```
// addsub method
void countsub::addsub()
{
   double a;
   double b;
   a = in1.read();
   b = in2.read();
   sum.write(a+b);
   diff.write(a-b);
};
```

Usage of SystemC types

```
sc_bit y; sc_bv<8> x;
y = x[6];

sc_bv<16> x; sc_bv<8> y;
y = x.range(0,7);

sc_bv<64> databus; sc_logic result;
result = databus.or_reduce();

sc_lv<32> bus2;
cout << “bus = “ << bus2.to_string();
```

✦ Example : Arbitrary Width Bit Type

```
#include <systemc.h>

int sc_main (int argc, char* argv[]) {
  sc_bv<8>  data_bus   ;
  sc_bv<16> addr_bus   ;
  sc_bit    parity     ;
  // Assign value to sc_bv
  data_bus = "00001011";
  cout <<"Value of data_bus : " << data_bus << endl;
  // Use range operator
  addr_bus.range(7,0) = data_bus;
  cout <<"Value of addr_bus : " << addr_bus << endl;
  // Assign reverse to addr bus using range operator
  addr_bus.range(0,7) = data_bus;
  cout <<"Value of addr_bus : " << addr_bus << endl;
  // Use bit select to set the value
  addr_bus[10] = "1";
  cout <<"Value of addr_bus : " << addr_bus << endl;
  // Use reduction operator
  parity  = data_bus.xor_reduce();
  cout <<"Value of parity   : " << parity << endl;

  return 1;
}
```

✦ Simulation : Arbitrary Width Bit Type

```
              SystemC 2.0.1 --- Oct  6 2006 19:17:37
         Copyright (c) 1996-2002 by all Contributors
                      ALL RIGHTS RESERVED
Value of data_bus : 00001011
Value of addr_bus : 0000000000001011
Value of addr_bus : 0000000011010000
Value of addr_bus : 0000010011010000
Value of parity   : 1
```