FCL examples & SystemC assertions

Ex-1 implementation

```
#include "tvm_func_cov.h"

COVER_GROUP_START(Arith)

   int add;
   int sub;
   int mul;
   int type;

   bool b_t;
   bool b_a;
   bool b_s;
   bool b_m;


  COVER_POINT_ST(TypeID)
    BINS(type,"[0:3]")
  COVER_POINT_END

  COVER_POINT_ST(ADD)
    BINS(add,"(0)")
  COVER_POINT_END

  COVER_POINT_ST(SUB)
    BINS(sub,"(0)")
  COVER_POINT_END

  COVER_POINT_ST(MUL)
    BINS(mul,"(0)")
  COVER_POINT_END
```

Sampling Variables

Variables for conditional coverage

-

```
  ADD_COVER_POINT
    COVER_POINT(TypeID,type,2)
    COVER_POINT(ADD,add,1)
    COVER_POINT(SUB,sub,1)
    COVER_POINT(MUL,mul,1)
  END_COVER_POINT

  SAMPLE_COVER_POINT
  if(b_t){
    SAMPLE(TypeID)
    b_t = false;
  }
  if(b_a){
    SAMPLE(ADD)
    b_a = false;
  }
  if(b_s){
    SAMPLE(SUB)
    b_s = false;
  }
  if(b_m){
    SAMPLE(MUL)
    b_m = false;
  }
END_SAMPLE_COVER_POINT

COVER_GROUP_END
```

COVER_POINT(CP, sampling variable, width)

Conditional coverage:
sample TypeID only if b_t is enabled

Ex-1 Cov Collection

```
class Arith;


template <class T>
class ArithmeticOps{
  ArithmeticOps(){}
  public:
  T a;
  T b;
  Arith cg;
  ArithmeticOps(std::string name):
    cg("Arith"){

    cg.b_t = false;
    cg.b_a = false;
    cg.b_s = false;
    cg.b_m = false;
    name = "file." + name;
    cg.set_db_name(name.c_str());
    cg.add_cover_point();
    if(typeid(a)==typeid(int)){
      cg.type = 0;
      cg.b_t = true;
      cg.sample();
    } else if(typeid(a)==typeid(long)){
      cg.type = 1;
      cg.b_t = true;
      cg.sample();
    } else if(typeid(a)==typeid(float)){
      cg.type = 2;
      cg.b_t = true;
      cg.sample();
    } else if(typeid(a)==typeid(double)){
      cg.type = 3;
      cg.b_t = true;
      cg.sample();
    }
  }

  T add(){
    cg.b_a = true;
    cg.add = 0;
    cg.sample();
    return (a+b);
  }

  T sub(){
    cg.b_s = true;
    cg.sub = 0;
    cg.sample();
    return ((a-b)>(b-a)?(a-b):(b-a));
  }

  T mul(){
    cg.b_m = true;
    cg.mul = 0;
    cg.sample();
    return (a*b);
  }

  void dump(){
    cg.display();
  }
};
```

Setting file name for coverage file and setting Cover points

Sampling the CG

Writing out the sampled coverage

Here the coverage collection is implemented directly in our DUT itself

Ex-2 implementation

```
#include "tvm_func_cov.h"

COVER_GROUP_START(Data)

   int i1;
   int i2;
   int op;

  COVER_POINT_ST(Input1)
    EX_AUTO_BINS(ip1)
  COVER_POINT_END


  COVER_POINT_ST(Input2)
    EX_AUTO_BINS(ip2)
  COVER_POINT_END


  COVER_POINT_ST(Output)
    EX_AUTO_BINS(opt)
  COVER_POINT_END

  CROSS_COVERPOINT(Input1_Input2)
    CROSS(2,"Input1","Input2")
  CROSS_COVERPOINT_END

  ADD_COVER_POINT
    COVER_POINT(Input1,i1,8)
    COVER_POINT(Input2,i2,8)
    COVER_POINT(Output,op,8)
    CROSS_COVER_POINT(Input1_Input2)
  END_COVER_POINT

  SAMPLE_COVER_POINT
    SAMPLE(Input1)
    SAMPLE(Input2)
    SAMPLE(Output)
    SAMPLE_CROSS(Input1_Input2)
  END_SAMPLE_COVER_POINT

COVER_GROUP_END
```

Will create an exhaustive bin where each value possible will be a separate bin

Crossing 2 Cover Points. N-way possible

Ex-2 Cov Collection

```
int main(){
  Data datas("data");
  datas.set_db_name("file2.data");
  datas.add_cover_point();
  int x = 100;
  int y = 200;

  ArithmeticOps<int> ar("int_type");
  ar.a = x;
  ar.b = y;
  datas.i1 = x;
  datas.i2 = y;
  datas.op = ar.add();
  datas.sample();

  x = 5;
  y = 3;
  ar.a = x;
  ar.b = y;
  datas.i1 = x;
  datas.i2 = y;
  datas.op = ar.sub();
  datas.sample();
```

```
  datas.op = ar.mul();
  datas.sample();

  x = 10;
  y = 9;
  ar.a = x;
  ar.b = y;
  datas.i1 = x;
  datas.i2 = y;
  datas.op = ar.mul();
  datas.sample();

  ar.dump();
}
```

SystemC assertions

- Sc_assert() is an extended version version of assert() from the standard C Library. It a Macro that calls SC_REPORT_FATAL if condition in sc_assert given by the user evaluated to FALSE.
- E.g
- sc_assert(i!=10); //Fails if i==10
- sc_assert(obj != NULL); //checking pointer obj !NULL