BMAS 2009

Mixed-Signal Test Development using Open Standard Modeling and Description Language

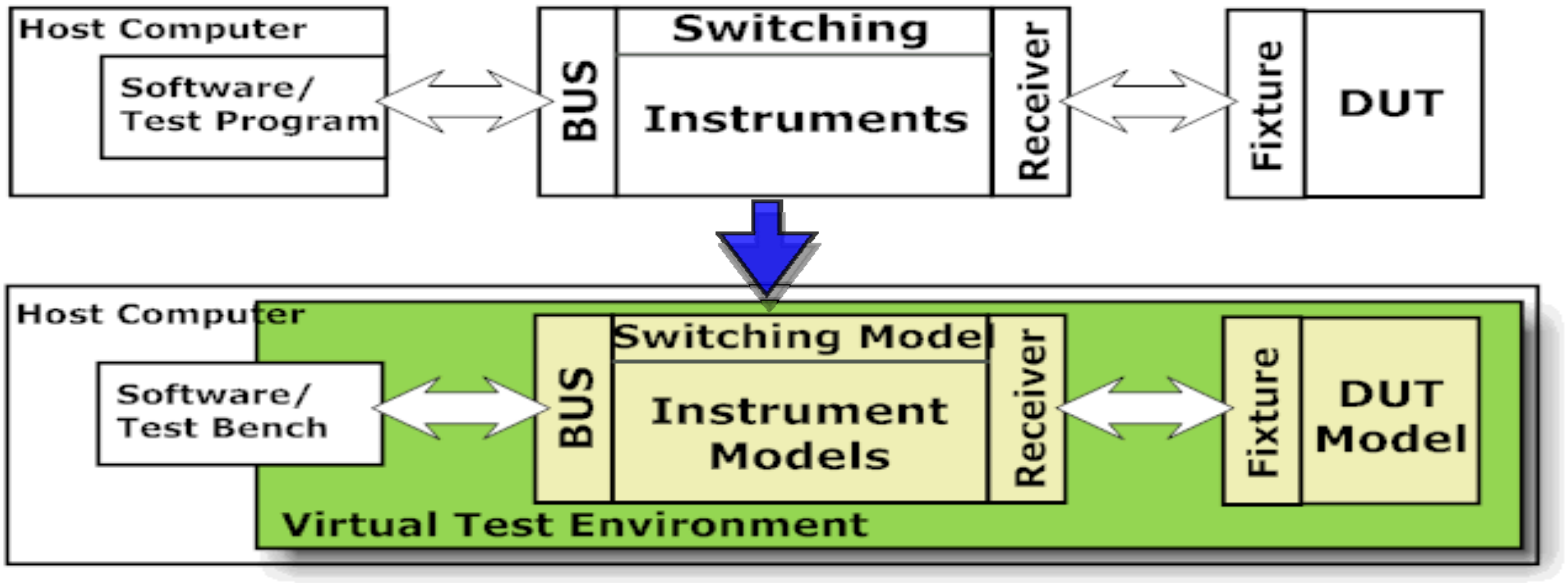
Ping LU, Daniel Glaser, Klaus Helmreich

Chair of Reliable Circuits and Systems
Friedrich-Alexander-University Erlangen-Nuremberg
Tel: +49-9131-85-23114
E-Mail: pinglu@lzs.eei.uni-erlangen.de

Outline

- **Introduction**
  - **Concept of Virtual Test**
  - **Benefits of Virtual Test**
  - **Background**
  - **Critical Requirements**
- Interfacing VT and Test System
- Model Implementation for Virtual Test
- ADC Case Study

Virtual Test ?



General test system architecture and its VT counterpart

- Allow tests being simulated with models of the device under test (DUT) and the target test environment
  - Elements: DUT, Instruments, DI, Software
- Allow test program being developed and debugged within pure simulation environment

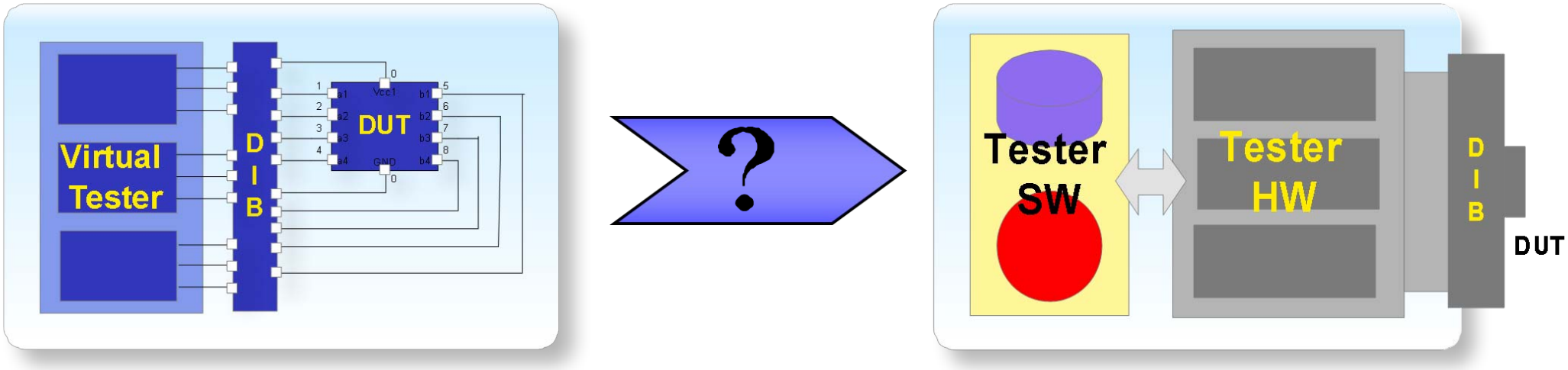
Benefits of Virtual Test (VT)

- Cut time-to-market / time-to-volume
- Reduce test costs
- Enhance test debug efficiency & test quality
- Enhance design verification
- Decrease test development time

Background

- Major research directions
- Test Program Emulation
  - Test program debug
  - **No** support for test simulation
- Separate HDL-based simulation Environment
  - Accuracy test simulation
  - Platform independent
  - **No** support for test program debugging
- Co-simulation between tester's software and simulation environment
  - Test simulation lively interactive with test program execution
  - Platform dependent
  - **Heavy** cost

Critical Requirements

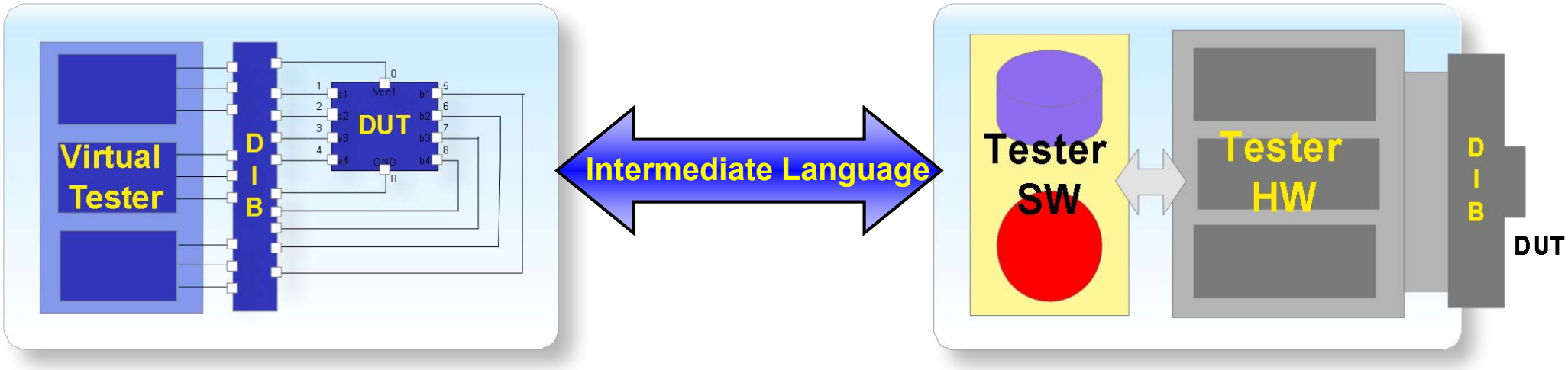


- Interface between VT and test system
- Sophisticated simulation library for test environment
- Simulation efficiency
- Integration

Outline

- Introduction
- **Interfacing VT and Test System**
  - **Intermediate language**
  - **OSA Compliance**
  - **System Overview**
- Model Implementation for Virtual Test
- ADC Case Study

Interfacing VT and test environment

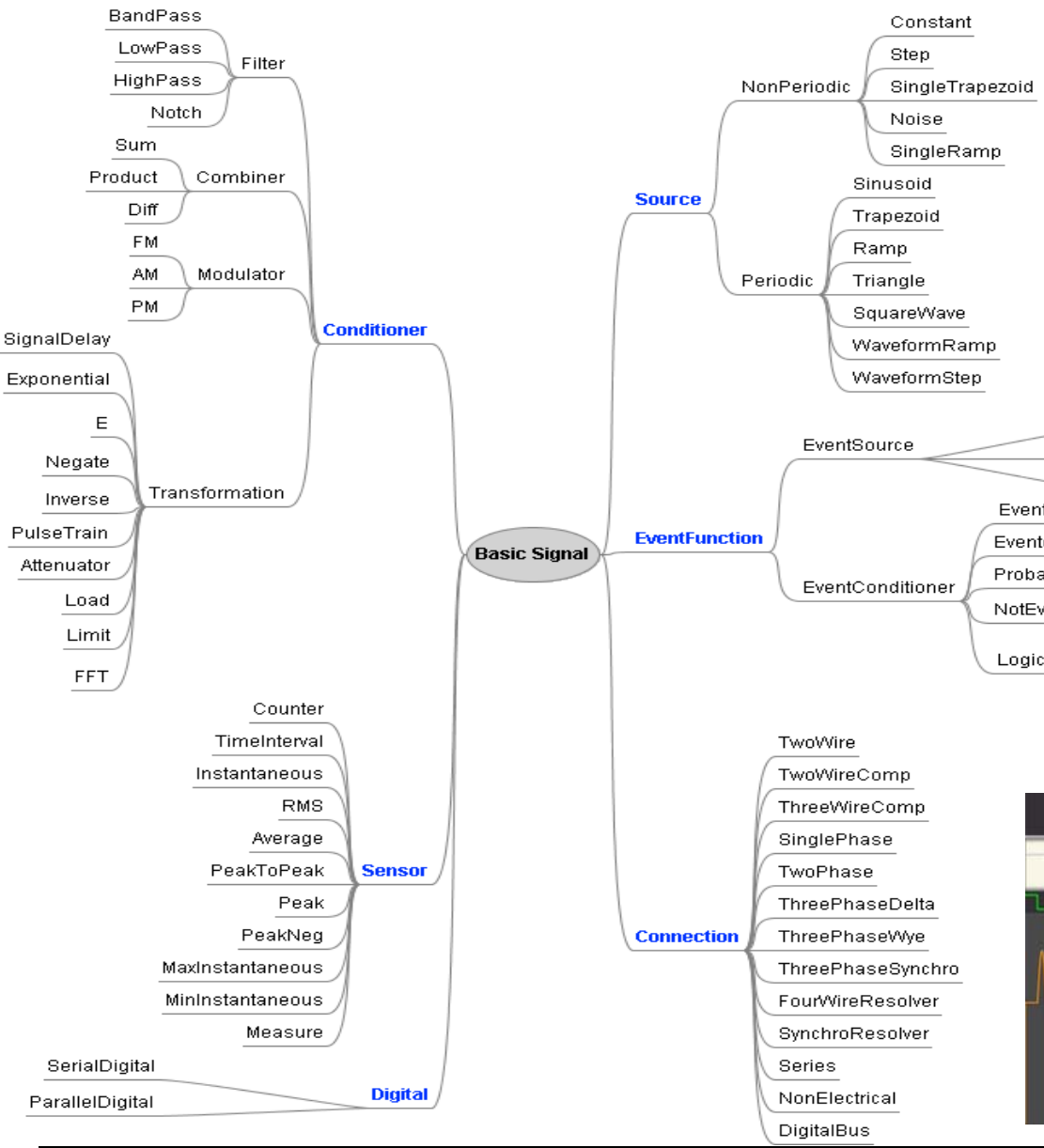


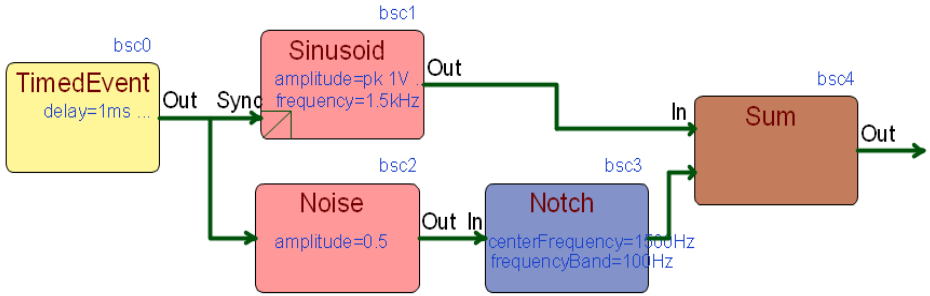
- Test information
  - Circuit setup; waveform (stimulus, measurement); timing; etc.
- Intermediate language
  - EDIF, WAVES, STIL, ATLAS
  - IEEE 1641 Standard for Signal and Test Definition (STD)
- Modeling language

IEEE 1641 Signal and Test Definition (STD)

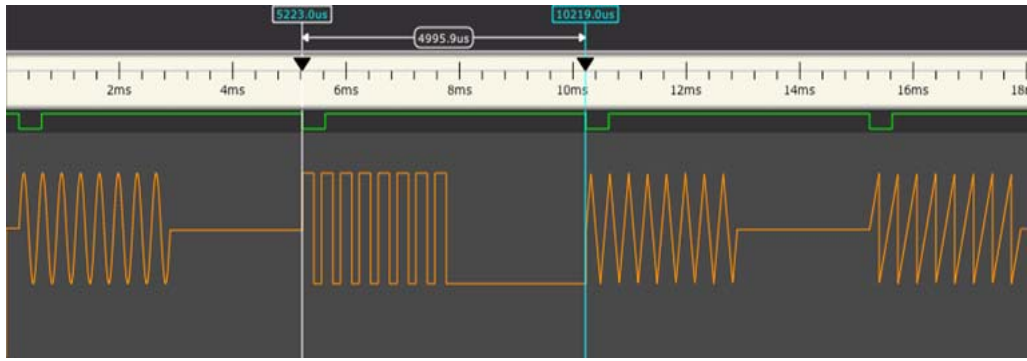
- Accurately define parameters and behavior of test signals
  - Roles: source, conditioner, sensor, event, digital, connection
- A mean to describe the test intent
  - Signal waveform; timing; location
- A mean to describe resource capability
- Language-independent
  - Descriptive language → Information transfer
  - Executive language → simulation

IEEE 1641 Signal and Test Definition (STD)



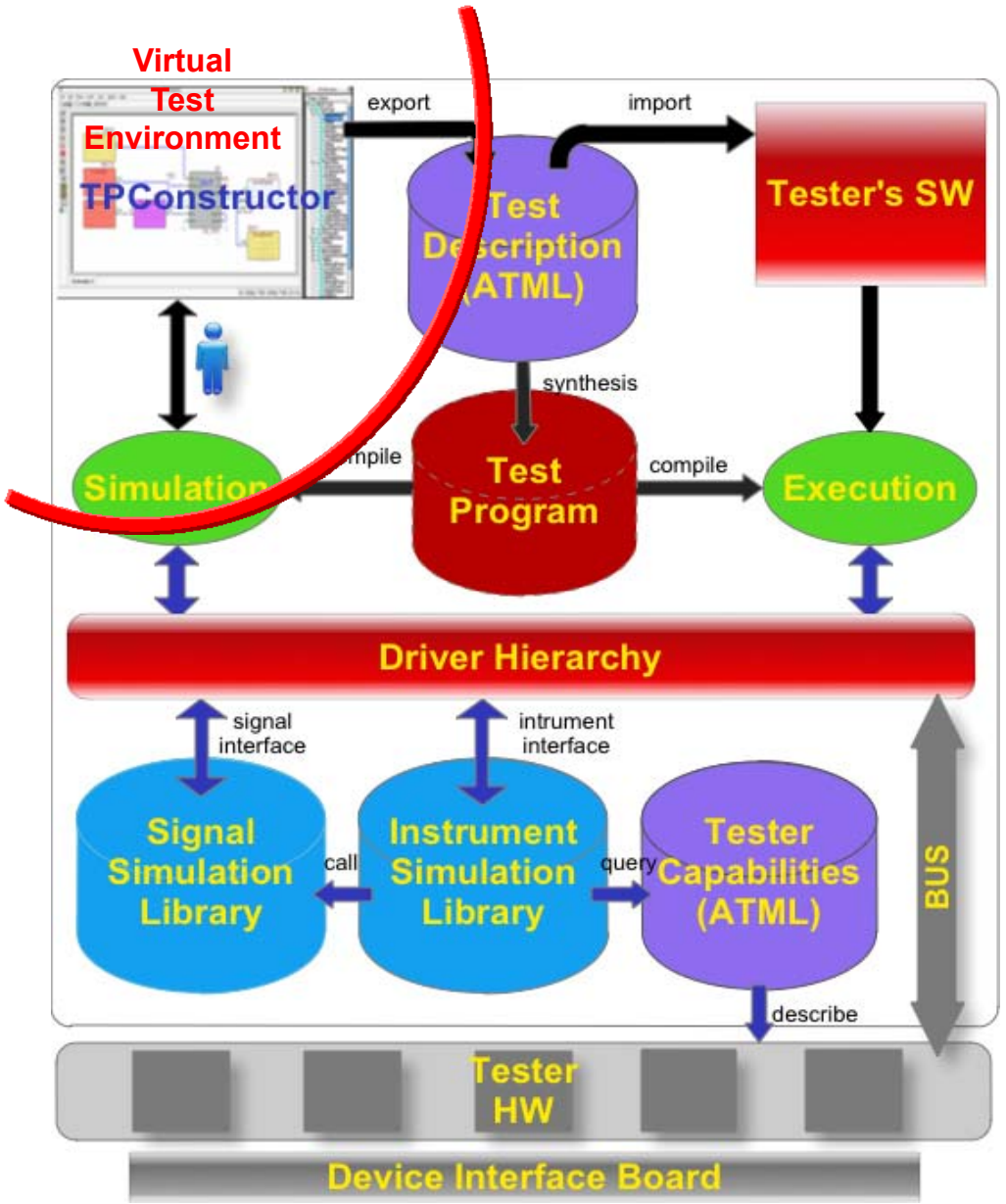



```
/** Construct, Configure */
bsc2 = static_cast<Noise<SYNC> *>(RM.Require("Noise","bsc0"));
bsc2 ->amplitude = "0.1V";
bsc2 ->seed  = static_cast<unsigned int>(std::time(0));

bsc1 = static_cast<Sinusoid<SYNC> *>(RM.Require("Sinusoid", "bsc1"));
bsc1 ->amplitude = "pk 1V errlmt 0.05% resol 1mV";
bsc1 ->frequency = "1.5kHz";

bsc3 = static_cast<Notch<8> *>(RM.Require("Notch","bsc3"));
bsc3 ->centerFrequency = "1500 Hz";
bsc3 ->frequencyBand = "100Hz";

/** Connection */
connect (bsc2->out, bsc3->in,   trace_t);
connect (bsc3->out, bsc4->in[1], trace_t);
connect (bsc1->out, bsc4->in[0], trace_t);
connect (bsc1->sync, bsc0->out, trace_t);   // Synchronize
```



System Overview

- Integration
- Critical Interface
  - Information Interface
  - Driver Interface
- Simulation Environment
  - SystemC/-AMS
- Tester Environment
  - Modular Test System



Outline

- Introduction
- Interfacing VT and Test System
- **Model Implementation for Virtual Test**
  - **Modeling Paradigm**
  - **Modeling of Instruments**
  - **Modeling of DUT & DIB**
  - **Tool chain**
- ADC Case Study

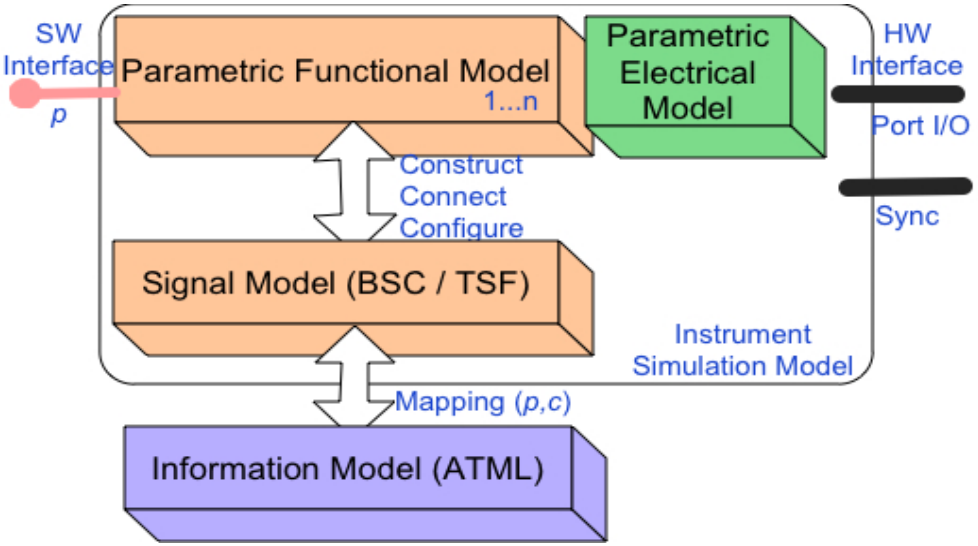
Modeling Paradigm

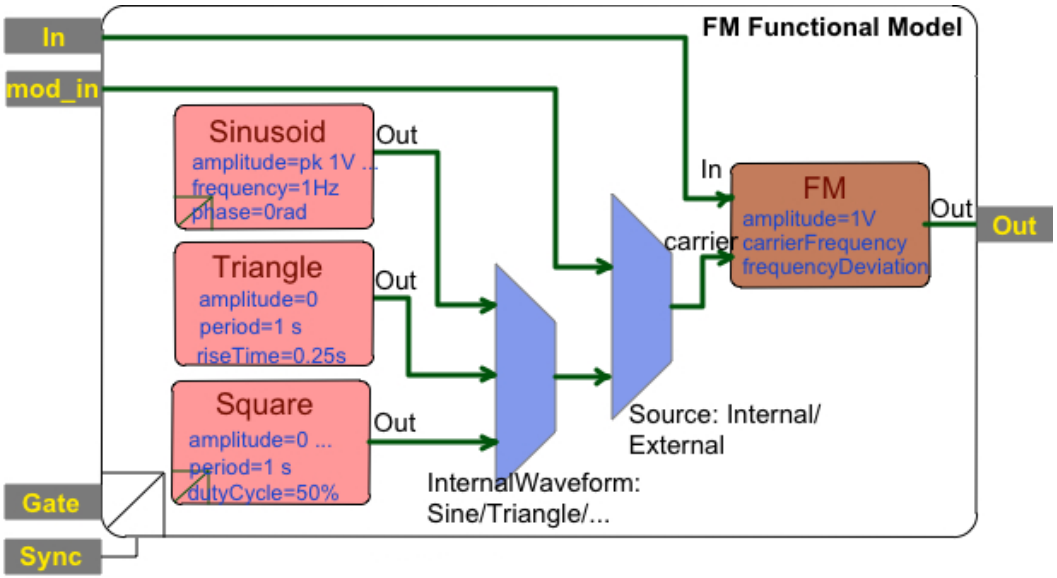
- Motivation
  - Modeling effort vs. resource coverage
  - Simulation efficiency vs. accuracy
- Analysis

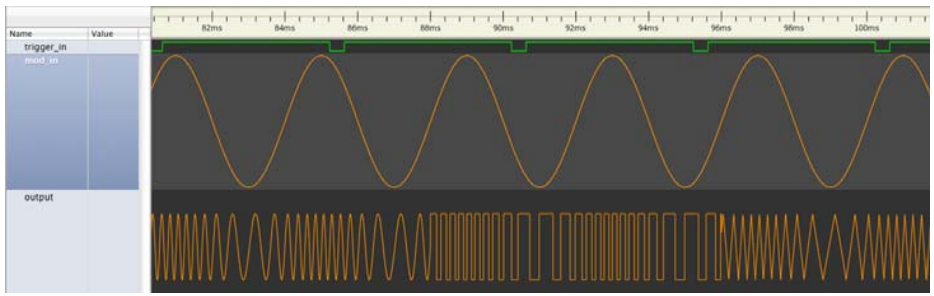
	Tester's Instruments	Device Interface	DUT	Testbench
Interested characteristics & tasks	capabilities, performance, operations	signal distribution, load board design & verification	specification, function behavior, test algorithm	circuit setup, data flow, post-processing
Considered abstraction level	L-1	L-1, L-2, L-3	L-1, L-2	–
Modeling Methodology	datasheet, identification	top-down refinement	from designer, model extract	schematic entry, code generation

Modeling of Instrument

- Parametric Functional Model
- Parametric Electrical Model
- Descriptive Information Model
- Software Interface







Example: Function Generator

```
/** Standard Waveform */
ivifgen->stdFunc->Configure ("ch0",Sinusoid, 3, 0,
3000, 0);

/** Trigger */
ivifgen->trigger->Source = TriggerSourceExternal;
ivifgen->trigger.BurstCount = 8;
```

list.1: test program (partial)

```
HRESULT IIviFgenStdFunc::putAmplitude(double val) {
    HRESULT hr = S_OK;
    if (GetSimulate()) {                          Simulation
        // forward settings to the simulation model
        m_Fgen_sim->FgenStdFunc.putAmplitude (val);
        return hr;
    }

    // forward settings to instrument
    hr = InstrPrintCommand();
    return hr;
}
```

list.2: instrument driver (partial)

```
Capability (FGen):
Source of Sinusoid Signal
amplitude
   Range 0V to 5.0V ;   Resolution 1mV ; Accuracy 0.1%
frequency
   Range 10Hz to 10MHz; Resolution 1Hz ; Accuracy 0.05Hz
Source of AM Signal ...
```

list.3: instrument information (partial)

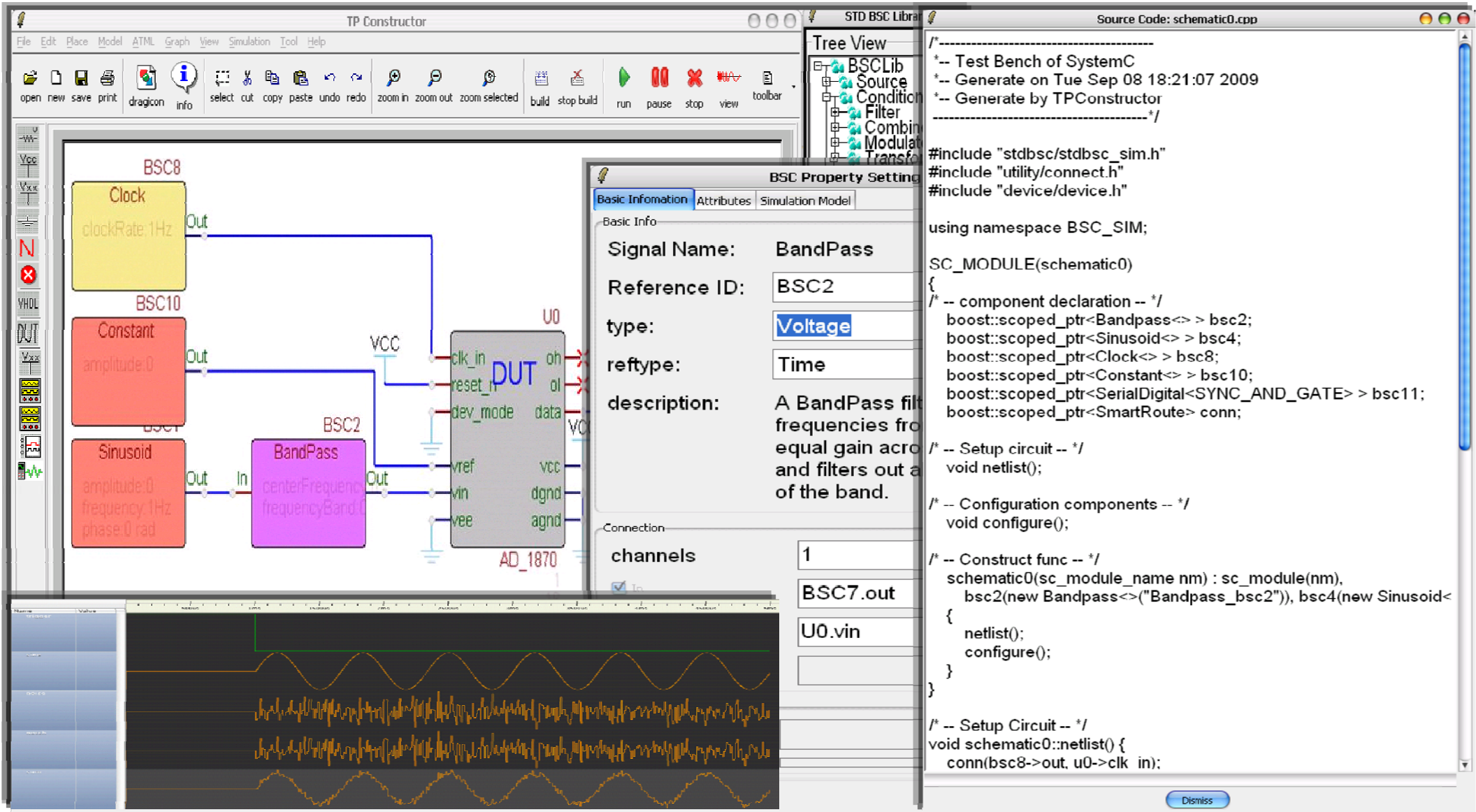
```
                                              SW Control Interface
INSTRUMENT_MODULE(FgenStdFunc), public IIviFgenStdFunc {
public
    /** Interface */                                  Interface
    sc_in<sc_logic> sync;
    sc_in<sc_logic> gate;
    sca_sdf_out<double> out

    /** Member Property */                            Properties
    double DutyCycleHigh;    /* Duty Cycle High : % */
    double Amplitude;        /* Amplitude (pk-pk) : Volt*/
    ...
    IviFgenWaveformEnum Waveform;/* Waveform */

    /** Constructor */                                Constructor
    SC_CTOR(FgenStdFunc);

private:  /** Private Methods */
    /** Pass parameters to std_bsc signal */          Configure
    void configure() ;
    /** Connect Signal Models */                      Connectivity
    void netlist() ;
    /** Map to paticular instrument */                Mapping to resource
    void map (resource_t res) ;

priva  :  /** Private Variables */
    BaseChannel<> *m_channel;
    /** Periodic Waveform */                          Signal components
    boost::scoped_ptr<Sinusoid<SYNC_AND_GATE> > sine;
    boost::scoped_ptr<Triangle<SYNC_AND_GATE> > triangle;
    boost::scoped_ptr<SquareWave<SYNC_AND_GATE> > square;
    /** DC Bias */
    boost::scoped_ptr<Constant<SYNC_AND_GATE> > dc;
    /** Waveform + DC Bias */
    boost::scoped_ptr<Sum<SYNC_AND_GATE,2> > sum;
    /** Multiplexer: select one of the signal:
        sine,square,triangle,rampup,rampdown  */
    boost::scoped_ptr<MUX<SCA_SDF,3> > mux1;
};
```

list.4: instrument model (SystemC / -AMS)

Modeling of DUT & DI

- Modeling of DUT
  - For top-down design flow → direct adoption
  - For bottom-up design flow → function extraction
  - Empirical model → high performance
- Modeling of DI
  - Function module: function block, primitive components
  - Signal path: primitives, transmission line

Tool Chain – example: TPConstructor

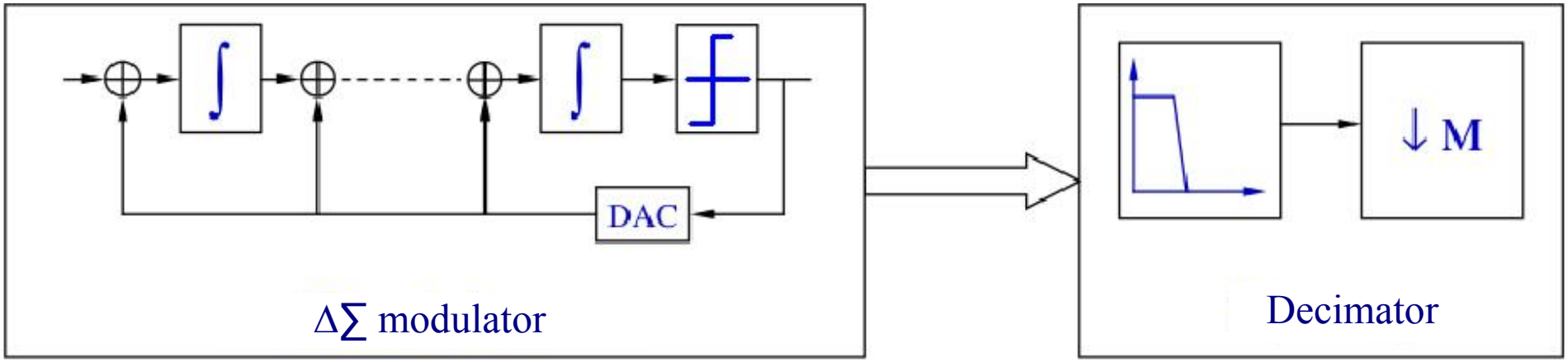


Outline

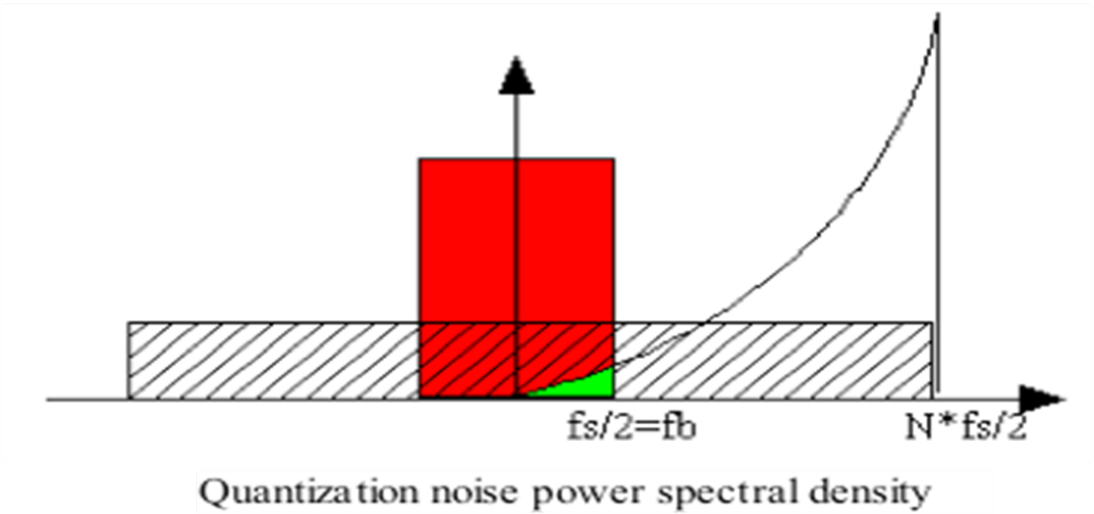
- Introduction
- Interfacing VT and Test System
- Model Implementation for Virtual Test

- **ADC Case Study**
  - **Modeling of ADC**
  - **VT Setup**
  - **Test Results**

Modeling of ADC



Principles : Oversampling + Noise shaping



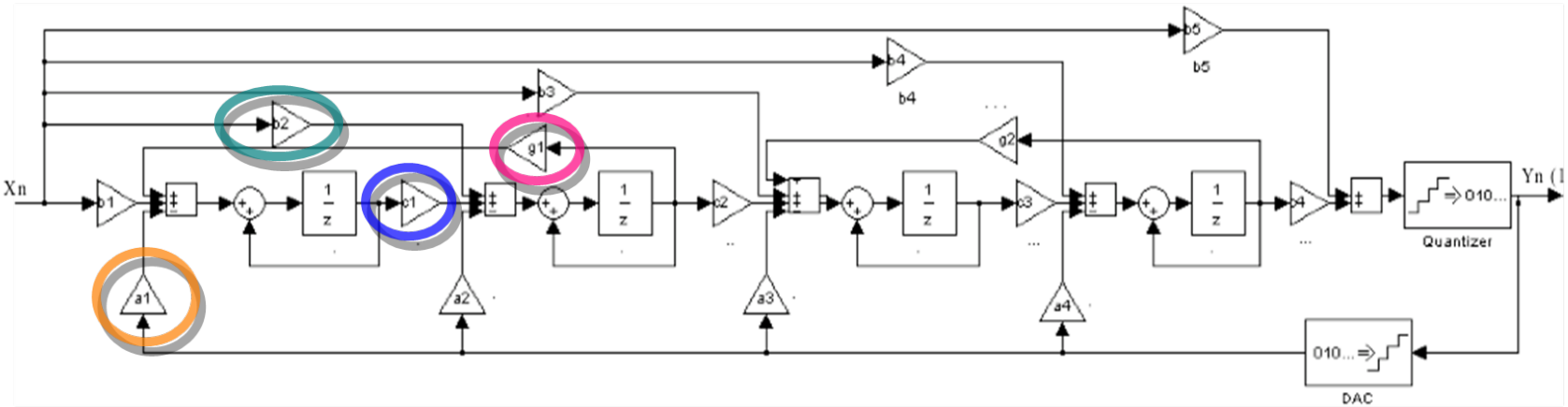
- **Features**
- 16-bits stereo ADC
- 92dB Dynamic Range
- 90dB S/(THD + N)
- 64x Oversampling
- 4th-order integrator
- 1 bit output
- 3-stage, linear phase decimator

Σ-Δ Modulator: Implementation

$$NTF(z) = \frac{(z^2 - 2z + 1)(z^2 - 1.998z + 1)}{(z^2 - 1.492z + 0.5643)(z^2 - 1.701z + 0.7868)}$$

SNR = 97dB

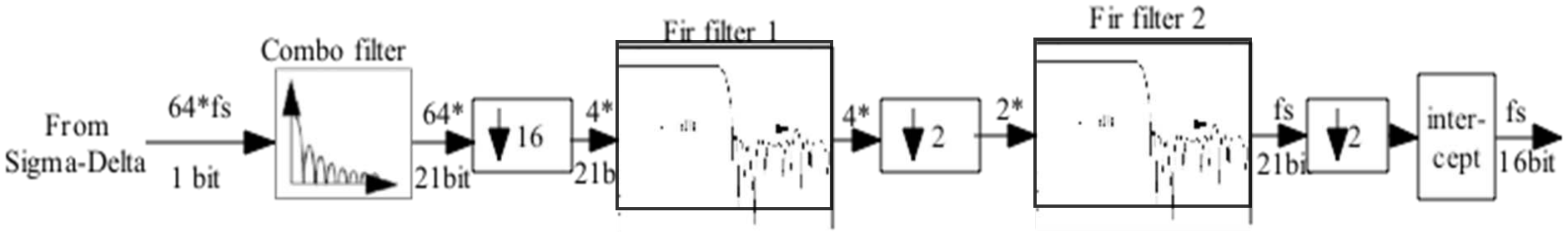
a=[0.0061 0.0524 0.2485 0.5560]
b=[0.0061 0.0524 0.2485 0.5560 1]
c=[1 1 1 1]
g=[0.0003 0.0018]



4-Stage CRFB

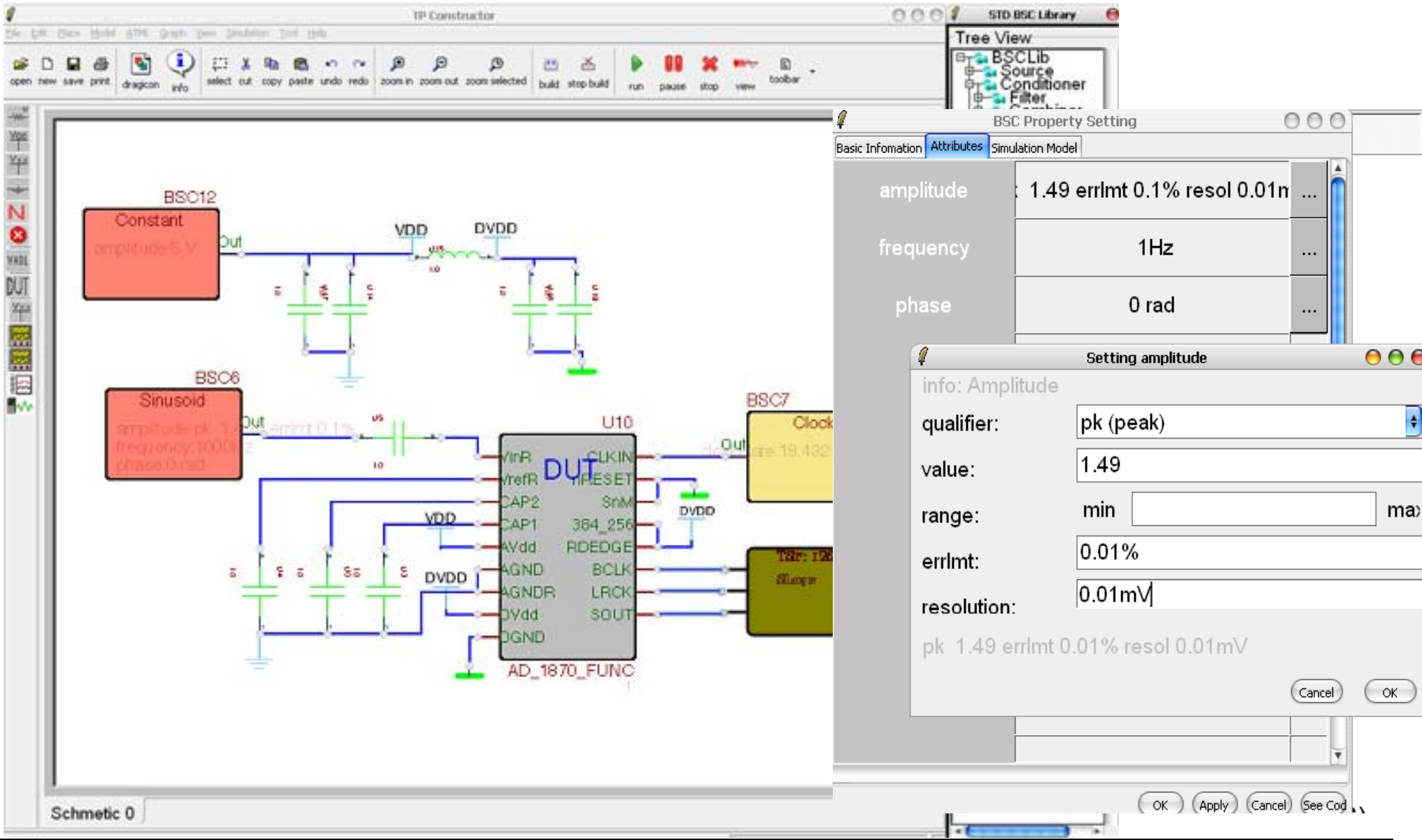
3-stage Decimator Filter

- Purpose
  - Filters the out-of-band noise
  - Reduces the data rate to fs
- Feature :
  - Half Band filter
  - Transfer band: 0.1*fs
  - Stop band attenuation: >90dB
  - Low pass band ripple: <0.006dB

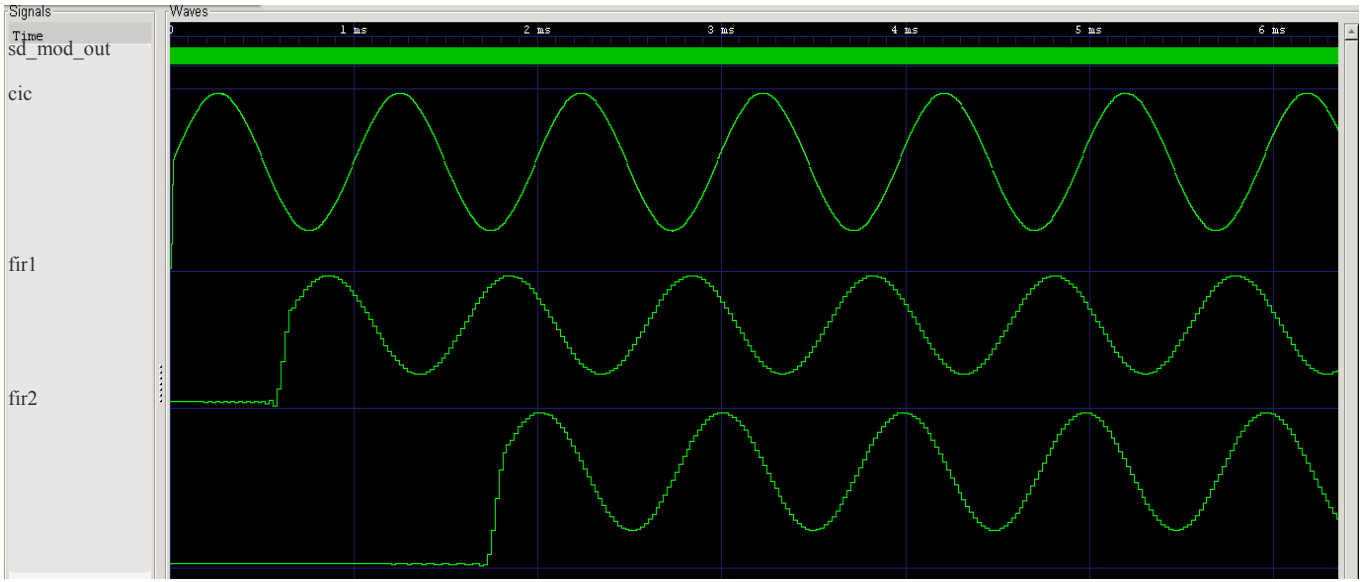


3rd stage Decimation filter

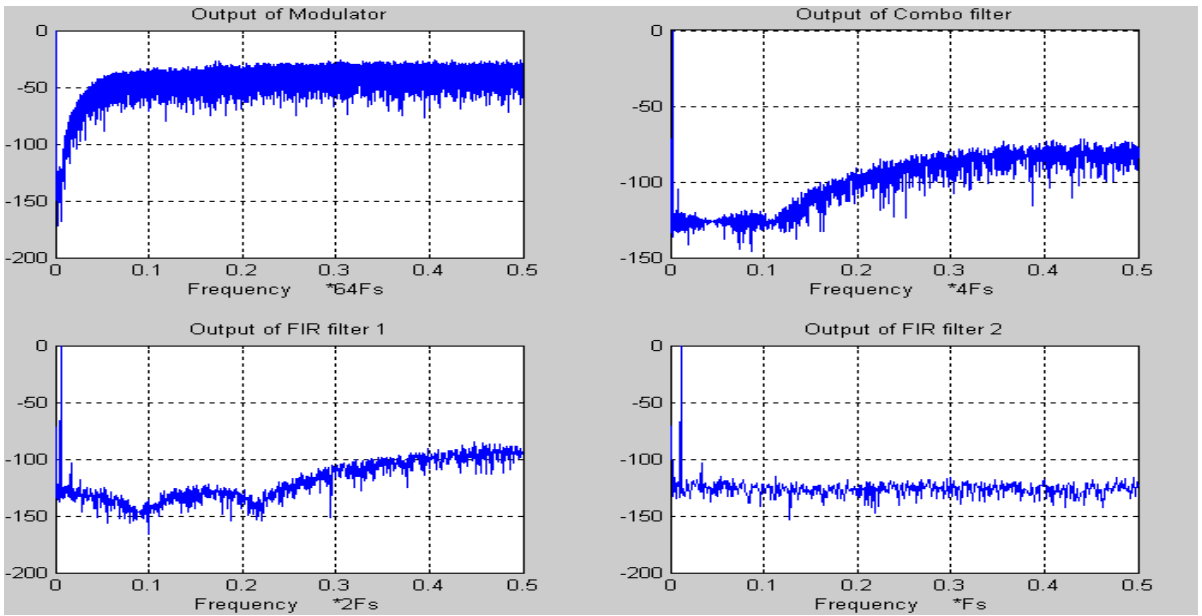
Test Setup


TP Constructor
STD BSC Library
Tree View
BSCLib
Source
Conditioner
Filter
BSC Property Setting
Basic Infomation
Attributes
Simulation Model
amplitude
1.49 errlmt 0.1% resol 0.01n
frequency
1Hz
phase
0 rad
Setting amplitude
info: Amplitude
qualifier:
pk (peak)
value:
1.49
range:
min
errlmt:
0.01%
resolution:
0.01mV
pk 1.49 errlmt 0.01% resol 0.01mV
Cancel
OK
Apply
See Cod
BSC12
Constant
BSC6
Sinusoid
BSC7
U10
DUT
VDD
DVDD
VinR
VrefR
CAP2
CAP1
AVdd
AGND
AGNDR
DVdd
DGND
CLKIN
RESET
SnM
384_256
RDEDGE
BCLK
LRCK
SOUT
AD_1870_FUNC
Schmetic 0

Test Results

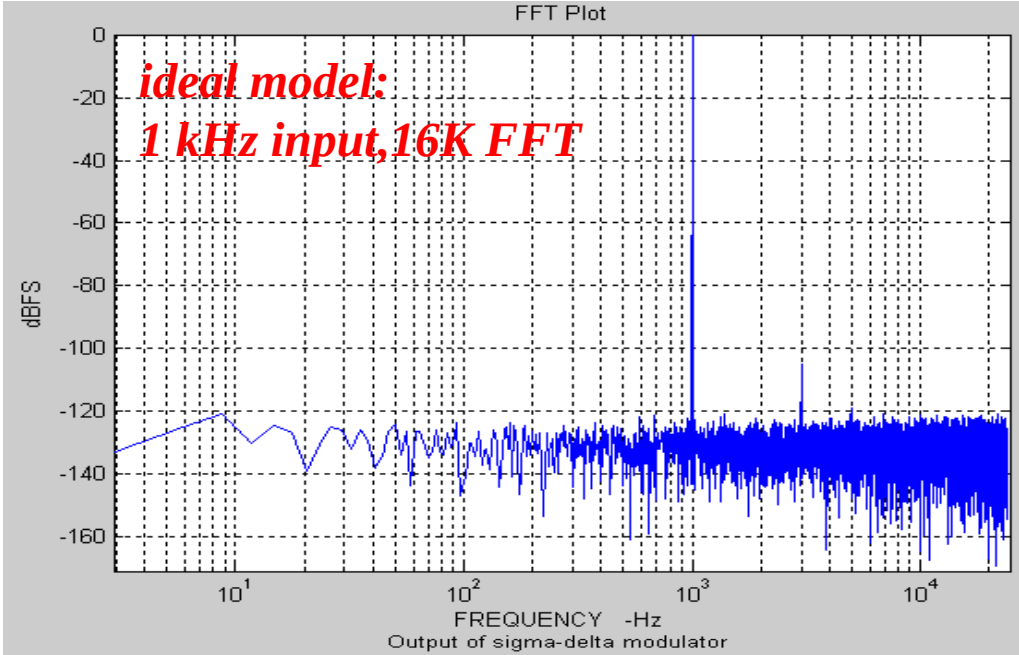


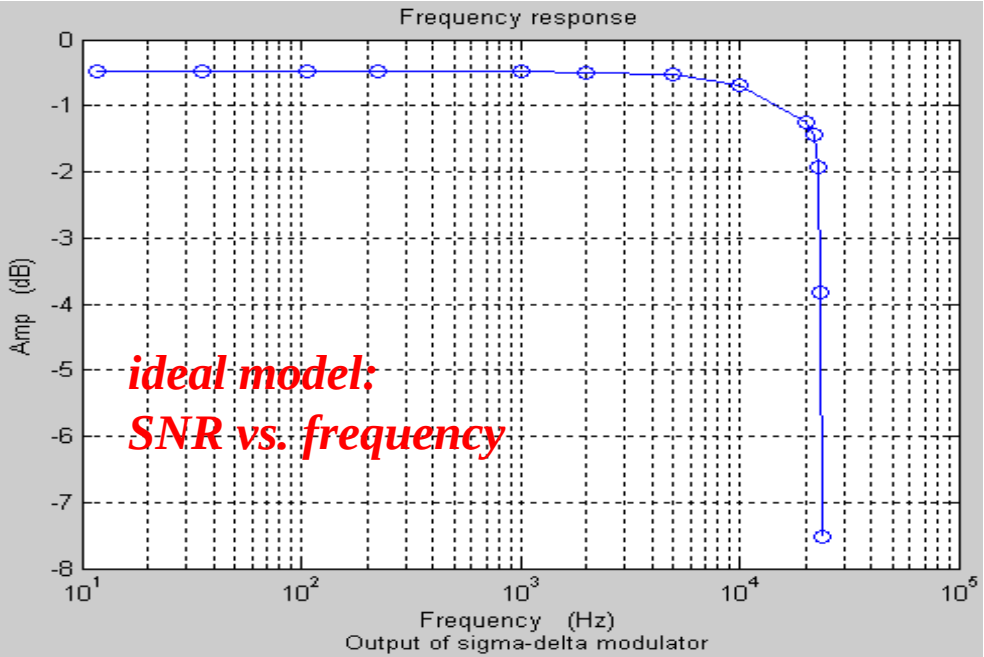
time domain:
internal signal behaviour

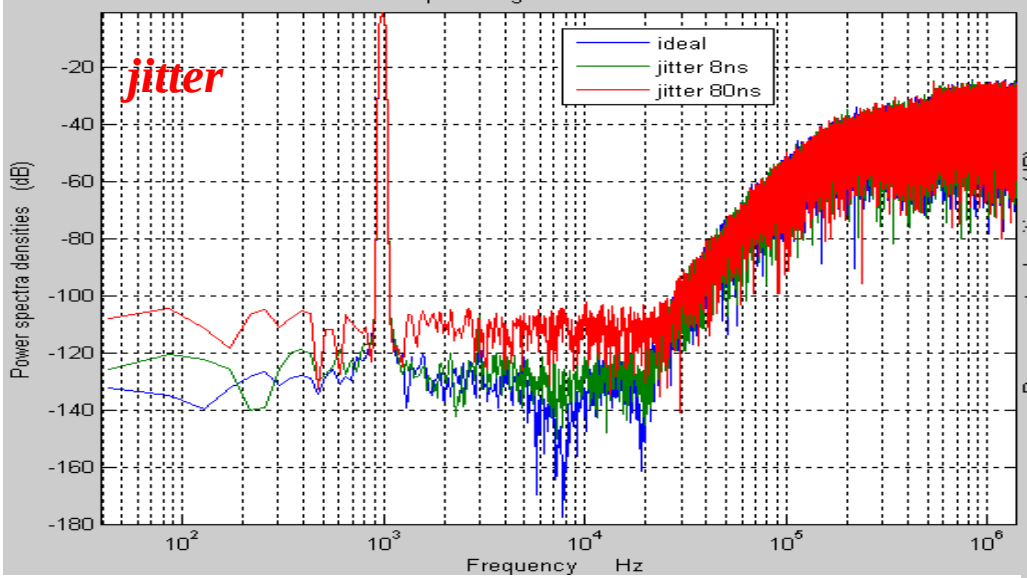


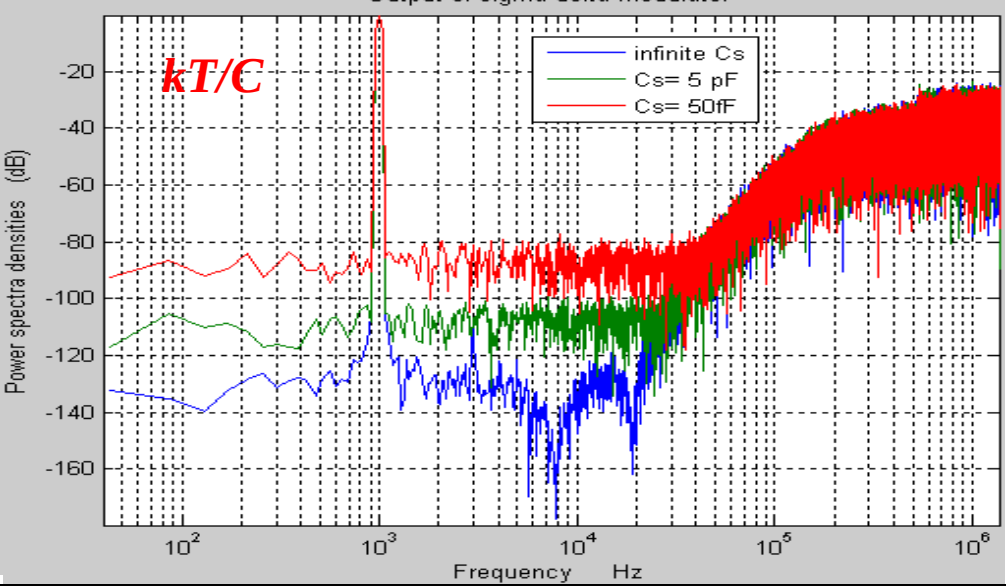
frequency domain:
internal signal behaviour

Test Results

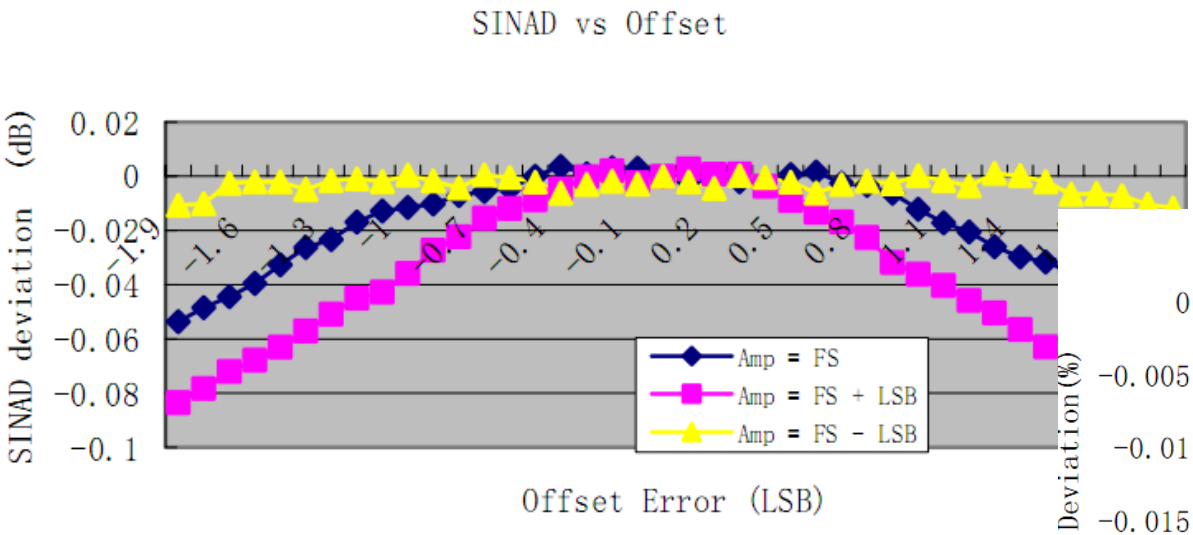


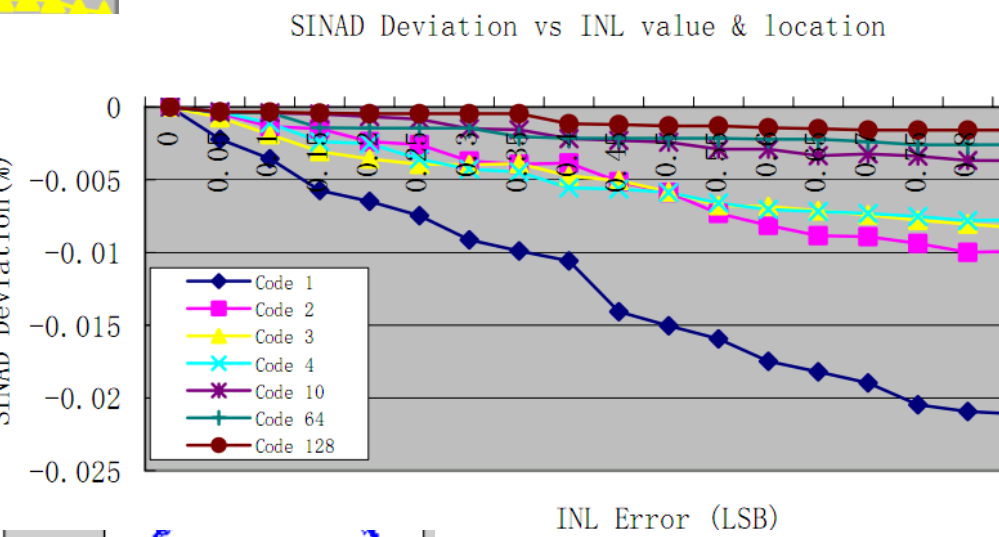


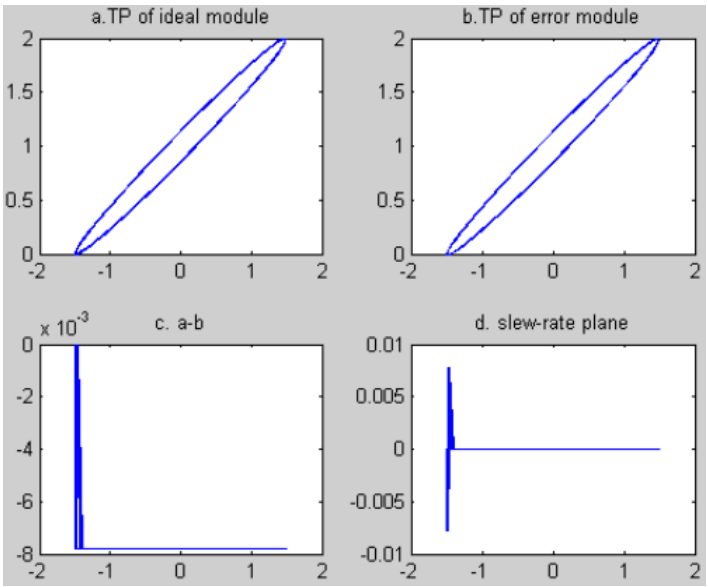


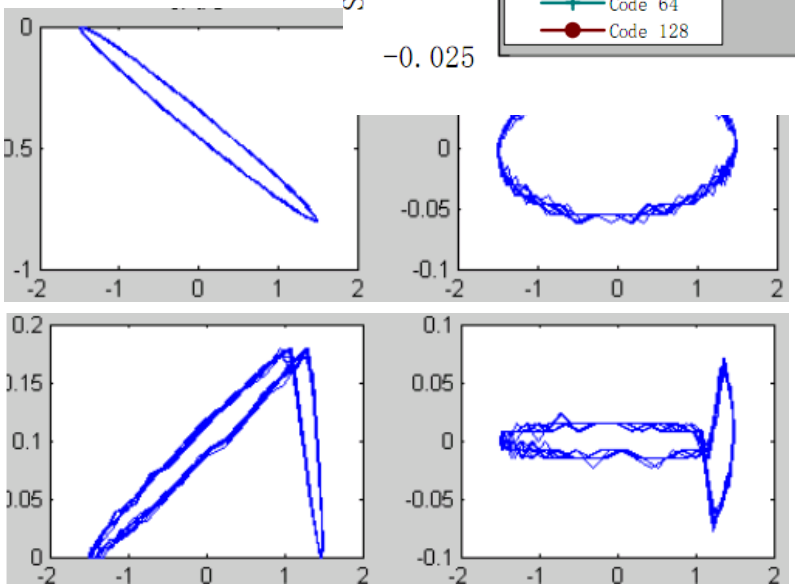


Test Results: examples









Thank you !