Modeling Wireless Sensor Network Nodes Using SystemC-AMS

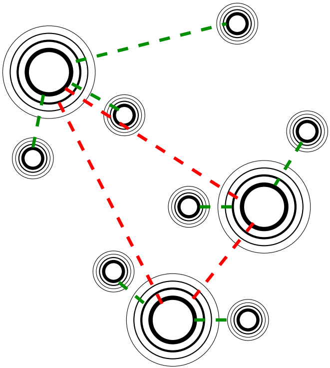
{Michel Vasilevski, Hassan Aboushady,
Francois Pecheux, Laurent de Lamarre}@lip6.fr

Laboratory LIP6
4, place Jussieu
75252 PARIS Cedex 05

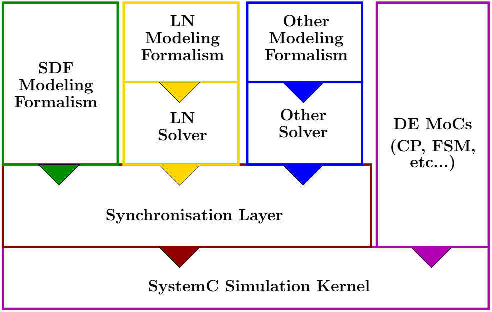
December 2007

Introduction

- Wireless sensor network for environmental and physical monitoring:
  - Temperature, vibration, pressure, motion, poluants
- Analog-digital mixed systems description needed



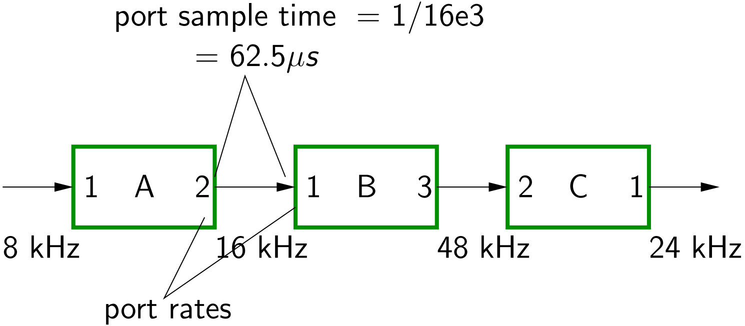
SystemC-AMS introduction



Models of computation (MoC)

- Model : Synchronous Data Flow (SDF)
- Model : Conservative linear network

Synchronous DataFlow : multi-rate synchronisation

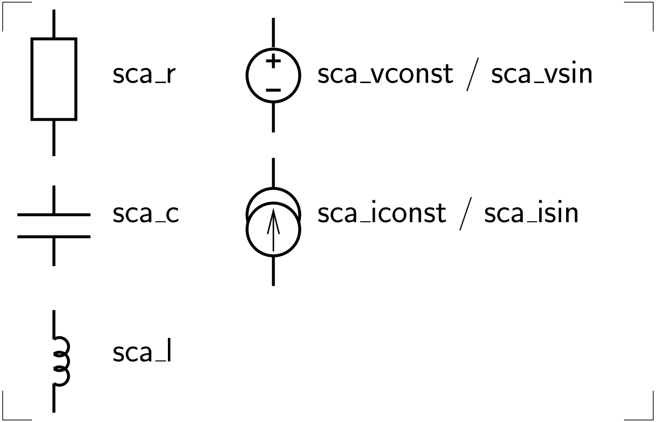


$$\frac{out\_sample\_freq}{out\_sample\_rate} = \frac{in\_sample\_freq}{in\_sample\_rate}$$

Synchronous DataFlow : language elements

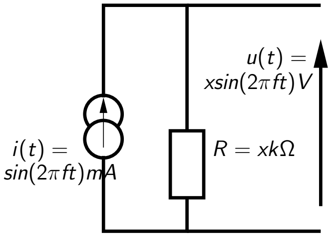
```
SCA_SDF_MODULE (interpolator){
  sca_sdf_in <double> in;          // input port
  sca_sdf_out <double> out;        // output port
  ...
  SCA_CTOR(interpolator){...}
  void init(){
    in.set_T(62.5,SC_US); // simulation sample time setting
                          // freq = 16 kHz
    in.set_rate(1);       // 1 sample consumed
    out.set_rate(3);      // 3 samples produced
  ...}
  void sig_proc(){
    double input = in.read();
    for (int i=0;i<3;i++) out.write(input,i);
  }
};
```

Conservative model

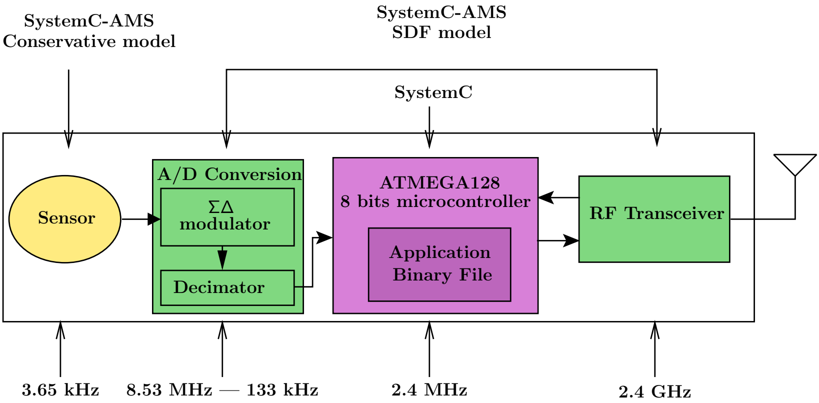


Conservative model : language elements

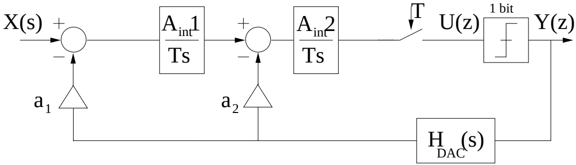
```
SC_MODULE (wave)
{
  sca_elec_port w1;
  sca_elec_ref gnd;

  sca_isin  *i_sin_1;
  sca_r     *i_r_1;

  void init(double a, double f){
    i_r_1->value = a*1000;
    i_sin_1->freq = f;
  }
  SC_CTOR (wave) {
    i_sin_1=new sca_isin("i_sin_1");
    i_sin_1->p(w1);
    i_sin_1->n(gnd);
    i_sin_1->ampl=0.001;

    i_r_1=new sca_r("r1");
    i_r_1->p(w1);
    i_r_1->n(gnd);
  }
};
```

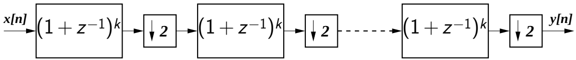


Wireless sensor network node : Architecture and implementation



Analog-to-Digital Converter





Modulator ΣΔ:

- resolution 10 bits
- 2^{nd} order
- OSR=64
- Return-to-Zero feedback

Decimator:

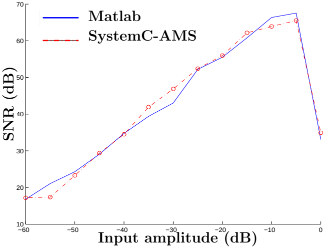
- FIR2 filter
- 3^{rd} order (k=3)

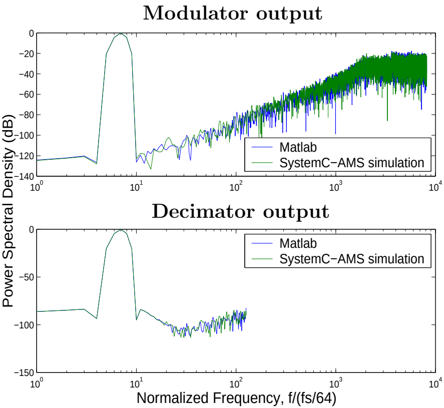
Implementation : SDF model of the ΣΔ integrator

```
SCA_SDF_MODULE (integrator_sd)
{
  sca_sdf_in < double >in;
  sca_sdf_out < double >out;

  double fs, ai;
  sca_vector < double >NUM, DEN, S;
  sca_ltf_nd ltf1;

  void init (double a, double f) {
    DEN (0) = 0.0;
    DEN (1) = 1.0;
    NUM (0) = 1.0/3.0;
    ai=a;
    fs=f;
  }
  void sig_proc () {
    out.write (ltf1(NUM,
                    DEN,
                    S,
                    fs * ai * in.read ()
                   )
              );
  }
  SCA_CTOR (integrator_sd) {}
};
```

Validation : ΣΔ simulation

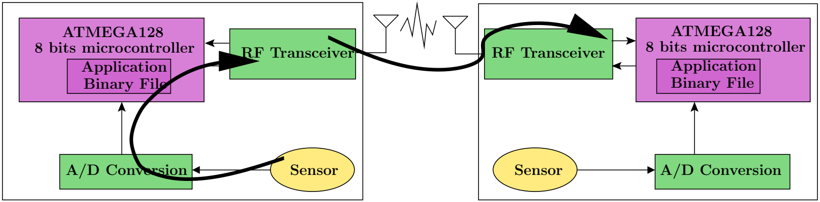
- SNR variation repecting to input amplitude
- Power spectrum of output signals





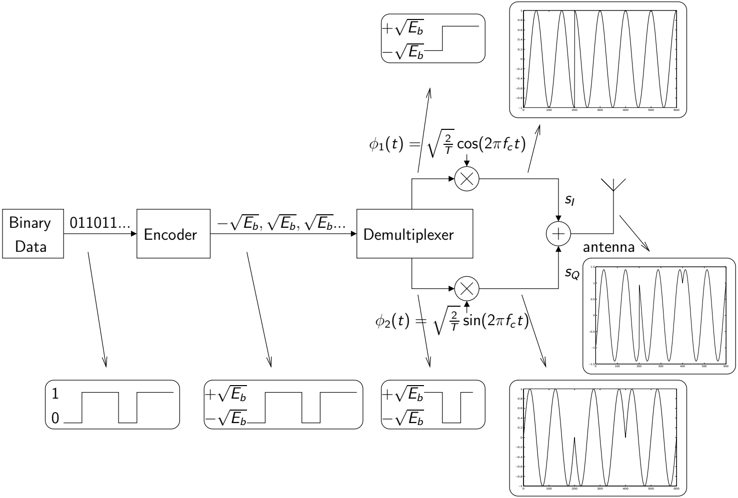
Microcontroller

- ATMEGA128 2.4 MHz
- CABA : Cycle Accurate, Bit Accurate

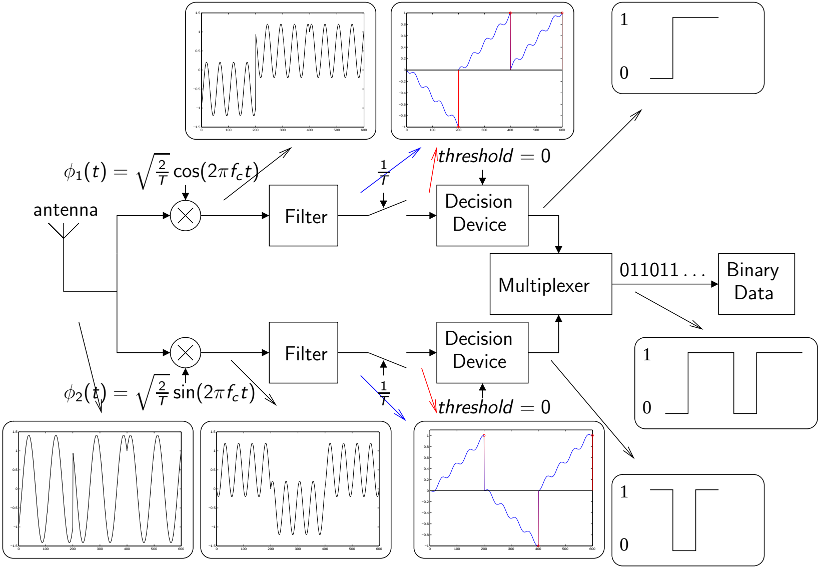


QPSK RF Transmitter

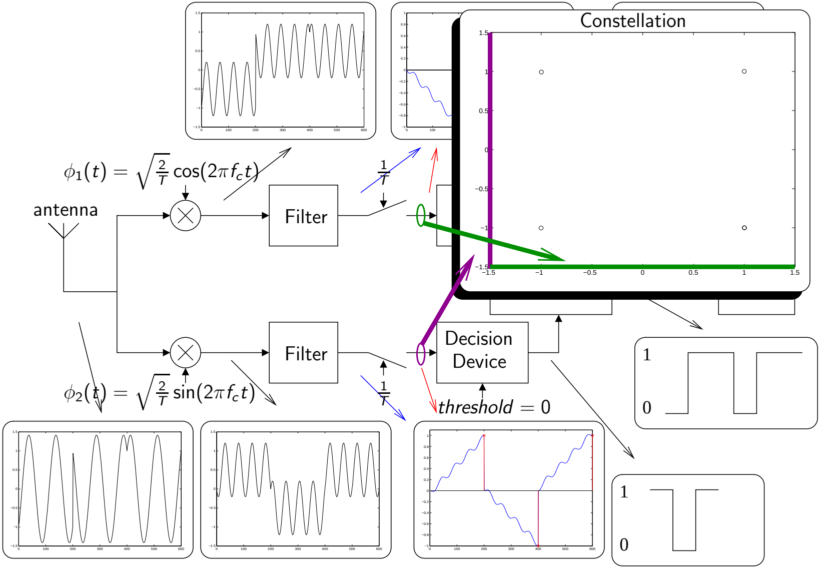
QPSK : quadrature phase shift keying



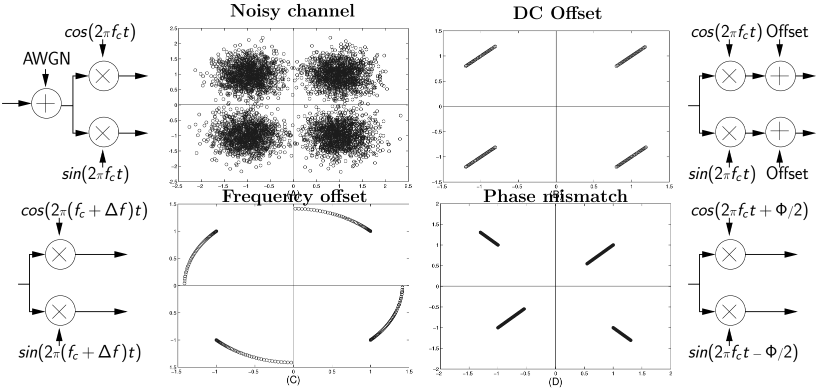
QPSK RF Receiver



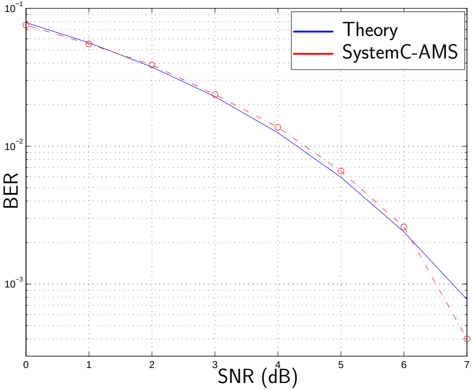
QPSK RF Receiver



RF Validation : Simulation including effects of defaults



RF Validation : BER variation respecting to SNR

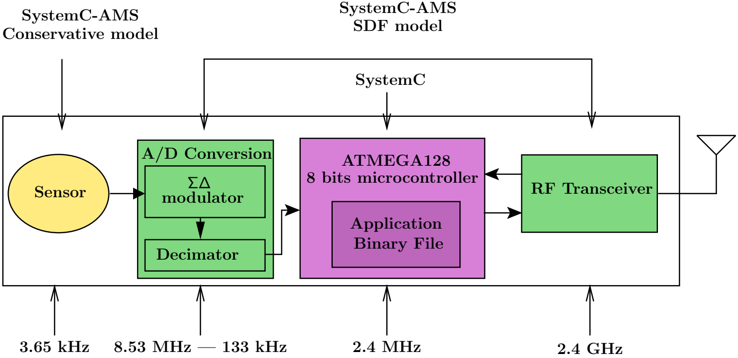


$$BER = \frac{nb\_bits\_incorrect}{nb\_bits\_sent}$$

Simulation results

	Settings	Simulation	Matlab	SystemC-AMS
ADC alone	OSR=64 8 bits output	1 ms 16*1024 pts	1.605s	0.934s
RF alone	2.4 GHz	416.67 μs 10^3 bits 10^7 pts RF	2m30.746s	54.360s
2-mote WSN	Same settings	416.67 μs	–	3m1.654s

Conclusion



Ongoing research :

- Refinements : IIP3, CP1, Noise Figure, I/O Impedances
- Baseband equivalent modeling