Bluespec

電気通信大学
大学院情報システム学研究科
三好健文

2010.08.31

bluespec

THE SYNTHESIZABLE MODELING COMPANY™

UNIFYING ARCHITECTURE, DESIGN AND VERIFICATION

Can you use emulation/FPGA-based prototypes before RTL today?

bluespec 2.0

A customer's complex IP block went from paper specification to **50 MHz model** running in emulation in **6 weeks**.

1 2 3 4 5

WHITEPAPER:

Synthesizable Models Enable Early Emulation for Complex IP
Run complex models at MHz speeds in weeks

WHITEPAPER:

Synthesizable Test Benches for ASIC Emulation/Prototyping or FPGAs
Ethernet MAC test bench case study: SystemVerilog VMM versus BSV

BLUESPEC

blue·spec (blu'spek) **1** The Synthesizable Modeling Company™ **2** Eliminates the modeling-to-RTL gap **3** Makes emulation feasible pre-RTL (not to mention more affordable and easier-to-use) for high-speed modeling, early software development & verification **4** Unifies architecture, design and verification with the only general purpose high-level synthesis

RESOURCES

On-Demand Webinar:
Designing Synthesizable Transactors and BFMs
Featuring Bluespec CTO Rishiyur S. Nikhil
Click here to view the webinar >

On-Demand Webinar:
10-minute technical overview of Bluespec
Featuring Bluespec CTO Rishiyur S. Nikhil
Click here to view the webinar >

NEWS

June 10, 2010
Bluespec High-Level Synthesis Toolset is Selected by Fujitsu
Read more >

March 25, 2010
Bluespec High-Level Synthesis Toolset is Selected by Panasonic
Read more >

February 9, 2010
Bluespec Delivers Plug-and-Play Library for Algorithm and Datapath Design
Read more >

Bluespecとは

- RTLから検証まで一貫した記述
  - シミュレーション
  - Verilogコードの生成

- 型によるコンパイル時チェック
- 多数のライブラリ群
  - 有限ステートマシン
  - サーバレスポンス/リクエスト

なぜ、Bluespec??

	生産性	速度	合成可能	教育コスト
Bluespec	◎	◎	○	？？
C/C++/SystemC	○	○	×	○
Verilog HDL	△	×/◎	○	△

目次

- とりあえずコードをみてみよう
- Bluespecツール群

- Bluespecのいろいろ
  - クロックはどこへ消えた
  - 強力な型システム
  - ライブラリ探訪

- 並列プログラミングの今後

Bluespecデザインコンテストのサンプル

http://www.cybernet.co.jp/bluespec/documents/Sort_Sample.pdf

```
interface BubSort_IFC;
  method Action start(Vector#(5, int) a);
  method Vector#(5, int) result();
endinterface

(* execution_order ="disp, fin" *)
(* preempts =
      "(swap_3, swap_2, swap_1, swap), fin" *)
(* synthesize *)
module mkBubSort (BubSort_IFC);
  Vector#(5, Reg#(int)) x
                          <- replicateM(mkReg(0));
  Reg#(Bool) sorted <- mkDReg(False);

  rule disp ;
   $write("%2d : ", $time);
   for (Integer i=0; i<5; i=i+1)
     $write("x[%0d]=%2d, ", i, x[i]);
     $display("");
  endrule

  rule fin (x[0] != 0);
    sorted <= True;
  endrule

  for (Integer i=0; i<4; i=i+1) begin
    rule swap ((x[i] > x[i+1]));
      x[i] <= x[i+1];
      x[i+1] <= x[i];
    endrule
  end

  method Action start(Vector#(5, int) a);
    writeVReg(x, a);
  endmethod

  method Vector#(5, int) result() if (sorted);
    return readVReg(x);
  endmethod

endmodule
```

Bluespecデザインコンテストのサンプル

http://www.cybernet.co.jp/bluespec/documents/Sort_Sample.pdf

```
interface BubSort_IFC;
  method Action start(Vector#(5, int) a);
  method Vector#(5, int) result();
endinterface

(* execution_order ="disp, fin" *)
(* preempts =
      "(swap_3, swap_2, swap_1, swap), fin" *)
(* synthesize *)
module mkBubSort (BubSort_IFC);
  Vector#(5, Reg#(int)) x
                          <- replicateM(mkReg(0));
  Reg#(Bool) sorted <- mkDReg(False);

  rule disp ;
   $write("%2d : ", $time);
   for (Integer i=0; i<5; i=i+1)
     $write("x[%0d]=%2d, ", i, x[i]);
     $display("");
  endrule
```

```
  rule fin (x[0] != 0);
   sorted <= True;
  endrule
  rule swap ((x[i] > x[i+1]));
    x[i] <= x[i+1];
    x[i+1] <= x[i];
  endrule
endmodule
```

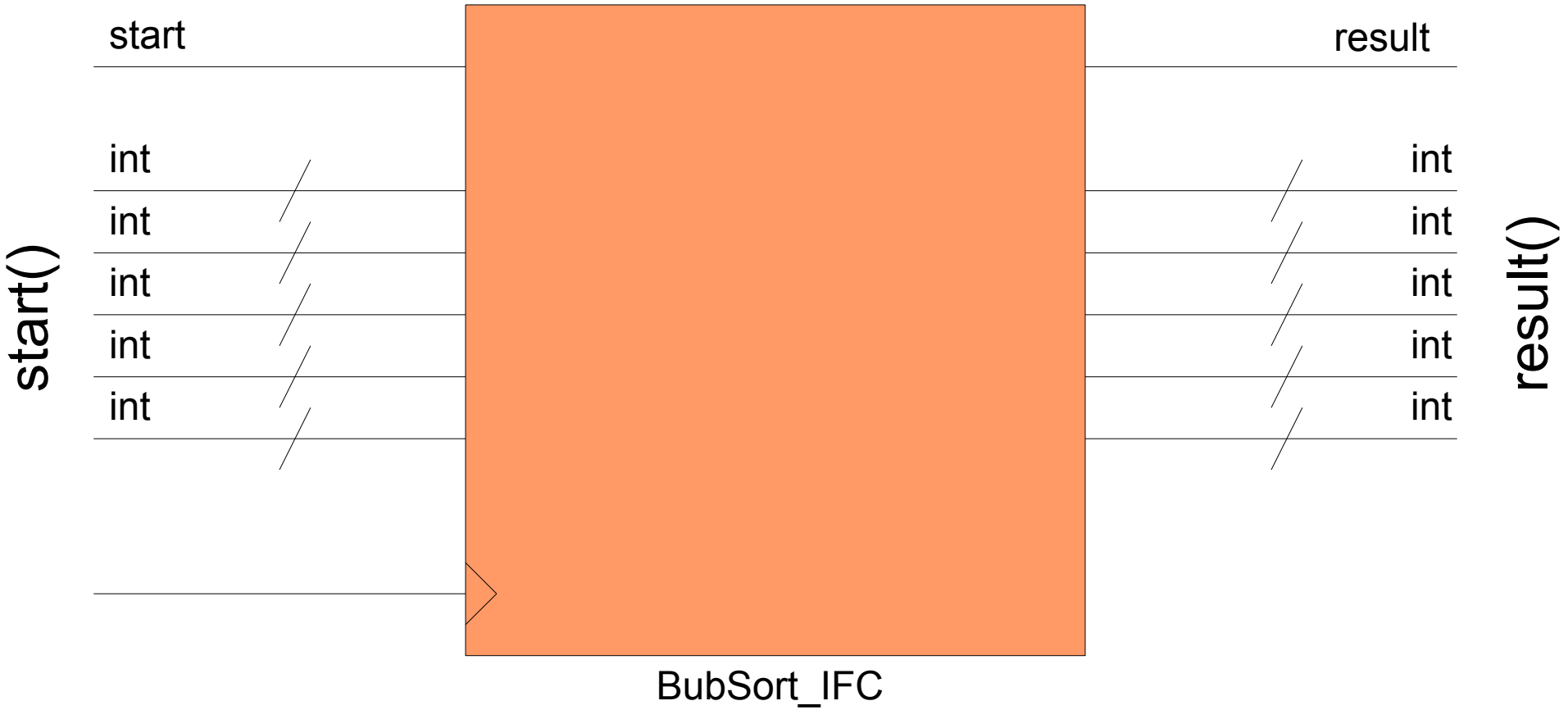
外部へのエクスポート
と実装

```
method Action start(Vector#(5, int) a);
  writeVReg(x, a);
endmethod
```

```
method Vector#(5, int) result() if (sorted);
  return readVReg(x);
endmethod
```

Bluespecデザインコンテストのサンプル

```
interface BubSort_IFC;
  method Action start(Vector#(5, int) a);
  method Vector#(5, int) result();
endinterface
```



Bluespecデザインコンテストのサンプル

http://www.cybernet.co.jp/bluespec/documents/Sort_Sample.pdf

```
interface BubSort_IFC;
  method Action start(Vector#(5, int) a);
  method Vector#(5, int) result();
endinterface

(* execution_order ="disp, fin" *)
(* preempts =
      "(swap_3, swap_2, swap_1, swap), fin" *)
(* synthesize *)
module mkBubSort (BubSort_IFC);
  Vector#(5, Reg#(int)) x
                          <- replicateM(mkReg(0));
  Reg#(Bool) sorted <- mkDReg(False);

  rule disp ;
   $write("%2d : ", $time);
   for (Integer i=0; i<5; i=i+1)
     $write("x[%0d]=%2d, ", i, x[i]);
     $display("");
  endrule
```

```
rule fin (x[0] != 0);
  sorted <= True;
endrule
```

```
for (Integer i=0; i<4; i=i+1) begin
  rule swap ((x[i] > x[i+1]));
    x[i] <= x[i+1];
    x[i+1] <= x[i];
  endrule
end
```

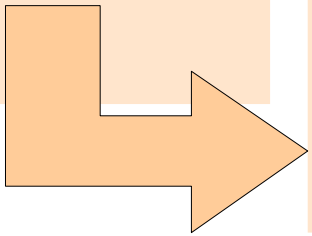
```
method Action start(Vector#(5, int) a);
  writeVReg(x, a);
endmethod

method Vector#(5, int) result() if (sorted);
  return readVReg(x);
endmethod
```

条件を満たすときに実行される
組み合わせ回路

Bluespecデザインコンテストのサンプル

forは繰り返しではなくて生成文

```
for (Integer i=0; i<4; i=i+1) begin
  rule swap ((x[i] > x[i+1]));
    x[i] <= x[i+1];
    x[i+1] <= x[i];
  endrule
end
```



```
rule swap ((x[0] > x[1]));
    x[0] <= x[1];
    x[1] <= x[0];
endrule
```

```
rule swap ((x[1] > x[2]));
    x[1] <= x[2];
    x[2] <= x[1];
endrule
```

…

```
rule swap ((x[3] > x[4]));
    x[3] <= x[4];
    x[4] <= x[3];
endrule
```

Bluespecデザインコンテストのサンプル

http://www.cybernet.co.jp/bluespec/documents/Sort_Sample.pdf

```
interface BubSort_IFC;
  method Action start(Vector#(5, int) a);
  method Vector#(5, int) result();
endinterface

(* execution_order ="disp, fin" *)
(* preempts =
      "(swap_3, swap_2, swap_1, swap), fin" *)
(* synthesize *)
module mkBubSort (BubSort_IFC);
  Vector#(5, Reg#(int)) x
                          <- replicateM(mkReg(0));
  Reg#(Bool) sorted <- mkDReg(False);

  rule disp ;
   $write("%2d : ", $time);
   for (Integer i=0; i<5; i=i+1)
     $write("x[%0d]=%2d, ", i,
     $display("");
  endrule
```

```
  rule fin (x[0] != 0);
    sorted <= True;
  endrule

  for (Integer i=0; i<4; i=i+1) begin
    rule swap ((x[i] > x[i+1]));
      x[i] <= x[i+1];
      x[i+1] <= x[i];
    endrule
  end

  method Action start(Vector#(5, int) a);
    writeVReg(x, a);
  endmethod
                                    if (sorted);
```

finが呼ばれる前に
swap_3～swap_1が実行され
安定するまでfinの実行が待たされる

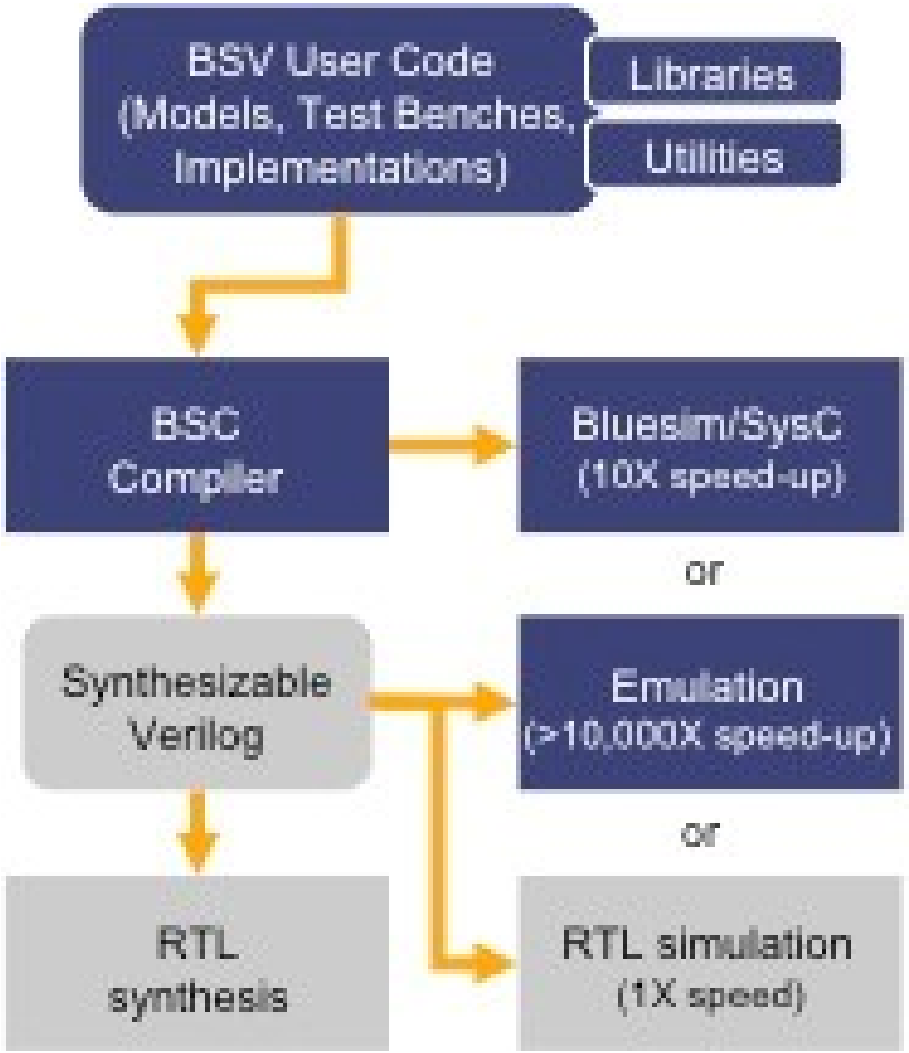
目次

- とりあえずコードをみてみよう
- Bluespecツール群

- Bluespecのいろいろ
  - クロックはどこへ消えた
  - 強力な型システム
  - ライブラリ探訪

- 並列プログラミングの今後

Bluespecシステムツール群




```
> bsc -u -sim ram.bsv
> bsc -sim -o sim.out -e tbBram \
tbBram.ba
```

```
> bsc -verilog -g mkBram_Test \
ram.bsv
```

Bluespecシステムツール群

他言語との連携

- C/C++
- RTL(Verilog HDL/VHDL)

Bluespecシステムツール群

他言語との連携(C)

- BluespecでCの関数を呼ぶための準備

```
import "BDPI" function Bool my_and (Bool, Bool);
```

- 実際に呼び出される関数

```
unsigned char my_and (unsigned char x,
                      unsigned char y);
```

Bluespecシステムツール群

他言語との連携(Verilog)

```
module mkVerilog_SRAM_model (clk, v_in_address, v_in_data,
                             v_in_write_not_read, v_in_enable, v_out_data);
    parameter FILENAME = "Verilog_SRAM_model.data";
    parameter ADDRESS_WIDTH = 10;
    parameter DATA_WIDTH = 8;
    parameter NWORDS = (1 << ADDRESS_WIDTH);

    input clk;
    input [ADDRESS_WIDTH-1:0] v_in_address;
    input [DATA_WIDTH-1:0] v_in_data;
    input v_in_write_not_read;
    input v_in_enable;
    output [DATA_WIDTH-1:0] v_out_data;
    ...
endmodule
```

呼び出されるモジュール

Bluespecシステムツール群

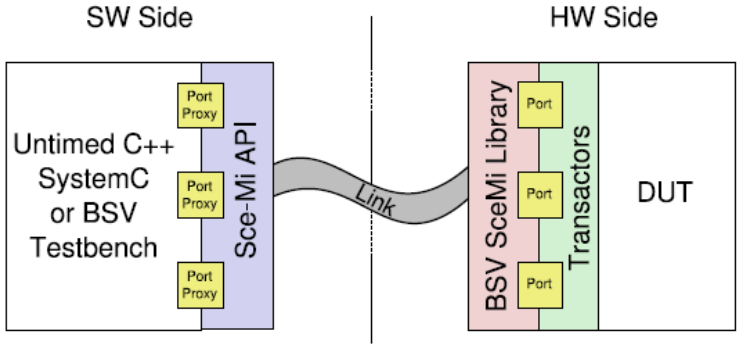
他言語との連携(Verilog)

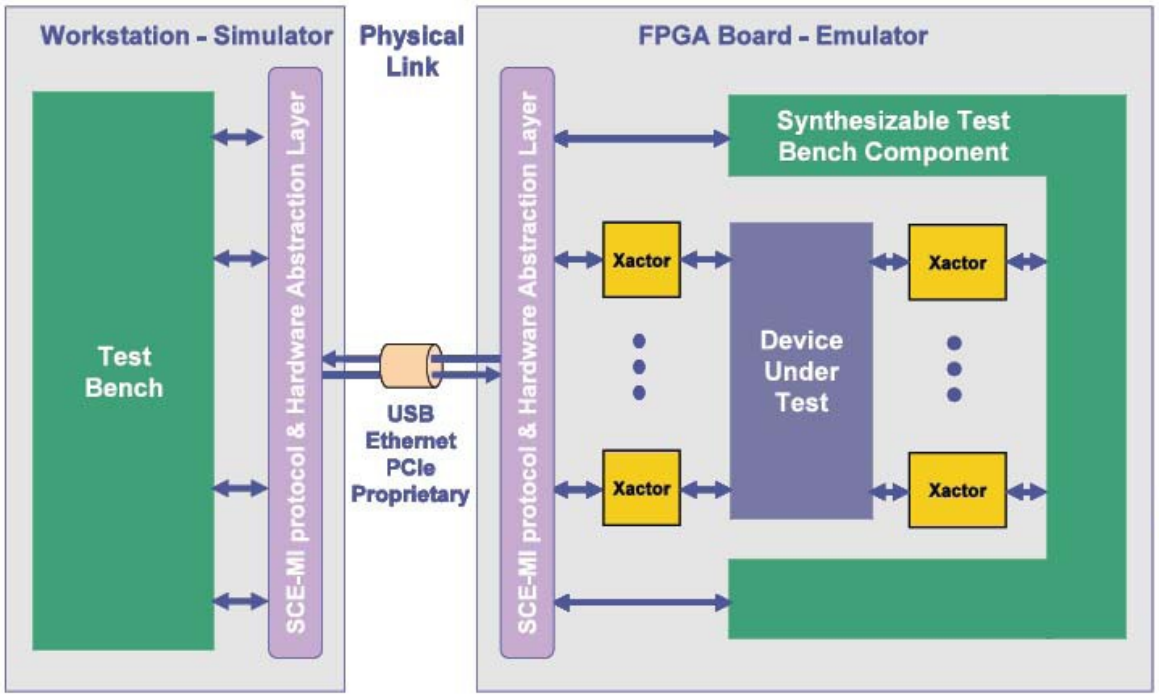
```
import "BVI" mkVerilog_SRAM_model =
  module mkSRAM #(String filename) (SRAM_Ifc #(addr_t, data_t))
    provisos(Bits#(addr_t, addr_width), Bits#(data_t, data_width));
      parameter FILENAME = filename;
      parameter ADDRESS_WIDTH = valueOf(addr_width);
      parameter DATA_WIDTH = valueof(data_width);
      method request (v_in_address, v_in_data, v_in_write_not_read)
                      enable (v_in_enable);
      method v_out_data read_response;
      default_clock clk(clk, (*unused*) clk_gate);
      default_reset no_reset;
      schedule (read_response) SB (request);
endmodule
```

呼び出すための準備

Bluespecシステムツール群

FPGAによるシミュレーション(SCE-MI)





(脱線)FAME分類

[0bit目] 0:Direct, 1:Decoupled
[1bit目] 0:Full RTL, 1:Abstracted Machine
[2bit目] 0:Single-Threade, 1:Multi-Threaded

Level 000: Direct FAME (ex. Quickturn)
Level 001: Decoupled FAME
(ex. Green Flash memory system)
Level 011: Abstract FAME (ex. HAsim)
Level 111: Multithreaded FAME (ex. RAMP Gold)

目次

- とりあえずコードをみてみよう
- Bluespecツール群

- Bluespecのいろいろ
  - クロックはどこへ消えた
  - 強力な型システム
  - ライブラリ探訪

- 並列プログラミングの今後

クロックはどこへ消えた

```
interface Test_ifc;
   method Bit#(26) result();
endinterface

module mkTest(Test_ifc);
   Reg#(Bit#(26)) counter <- mkReg(0);
   rule r0;
      counter <= counter + 1;
   endrule
   method Bit#(26) result();
      return readReg(counter);
   endmethod
endmodule
```

クロックはどこへ消えた

```
module mkTest(CLK, RST_N, result, RDY_result);
  input  CLK;
  input  RST_N;

  // value method result
  output [25 : 0] result;
  output RDY_result;
/* snip */
  always@(posedge CLK)
  begin
    if (!RST_N)
      begin
        counter <= `BSV_ASSIGNMENT_DELAY 26'd0;
      end
    else
      begin
        if (counter$EN) counter <= `BSV_ASSIGNMENT_DELAY counter$D_IN;
      end
  end
/* snip */
endmodule  // mkTest
```

Bluespecの型システム

型って何？

- HaskellとかOcamlとかのあれ
- 型チェックで安全プログラミング？？
- なんか格好いい

で、Bluespecでは？

- かなり厳格な型システム
- データの種類ではなくてオブジェクトの種類

Bluespecの型システム

型って何？

- HaskellとかOcamlとかのあれ
- 型チェックで安全プログラミング？？
- なんか格好いい

で、Bluespecでは？

- かなり厳格な型システム
- データの種類ではなくてオブジェクトの種類
- 型クラス

Bluespecの型システム

- Bit型
  - Bit#(n)　(= bit[n-1:0]), Bool (= Bit#(1))
  - Int#(n), int (= Int#(32))
  - Uint#(n)
- Non Bit型
  - Integer
  - String
  - Interfaces, Modules, Rules
  - Action, ActionValue
  - functions

Bluespecの型システム

- Bit型
  - Bit#(n)　(= bit[n-1:0]), Bool (= Bit#(1))
  - Int#(n), int (= Int#(32))
  - Uint#(n)
- Non Bit型
  - Integer
  - String
  - Interfaces, Modules, Rules
  - Action, ActionValue
  - functions

この”n”もnumeric typeという型をもつ

Bluespecの型システム

明示的な型変換

- pack/unpack (= to Bit#(n)/from Bit#(n))
- zeroExtend/signedExtend
- truncate
- fromInteger
- valueOf

```
...
   Bit#(1) aB1;
   Bool aBool = unpack( aB1 ) ;  // unpack from Bit
...
   Bit#(16) aB16 ;
   Int#(16) aI16 = unpack( aB16 );       // size matters
...
   Bit#(16) bB16 ;
   bB16 = pack( aI16 ) ;         // convert back to bits
...
   // Truncates and Extends do not require any bit width information
   UInt#(16) aU16;
   UInt#(14) aU14 = truncate ( aU16 ) ;
...
   Bit#(16) aB16;
   Bit#(14) aB1  = truncate ( aB16 ) ;
```

Bluespecの型システム

Bluespec Library型

- Reg#(a)
- Wire#(a)
- FIFO#(a)
- PulseWriter#(a)/PulseReader#(a)

ライブラリ探訪

StmtFSM

ライブラリ探訪

Vector

ライブラリ探訪

ClientServer

目次

- とりあえずコードをみてみよう
- Bluespecツール群

- Bluespecのいろいろ
  - クロックはどこへ消えた
  - 強力な型システム
  - ライブラリ探訪

- 並列プログラミングの今後

並列プログラミングの今後

Bluespecに学ぶ

- rule/methodはいいなあ
  - 細粒度並列性はHDL系が一番書きやすいかも
  - ruleの切り出しが鍵か？
- Bluespecシミュレータはマルチプロセッサ非対応

並列プログラミングの今後

- そもそも並列プログラミングは存在しえるか？
  - Bamboo
  - Molatomium
- 並列化コンパイラ？
  - OSCARコンパイラ
  -
- ディレクティブ？
  - *Ss
  - XcalableMP