



## All these need common capabilities and infrastructure

Although emulation, modeling and acceleration are quite different reasons for using an FPGA, and the communities involved are quite different (almost disjoint), they all require the same kind of capabilities and infrastructure:

- High-level design language for FPGA
- Co-execution of host software with FPGA model/IP
- High-level host  $\Leftrightarrow$  FPA communication facilities
- FPGA debugging support
- Portability across FPGA platforms
- Low cost FPGA platforms

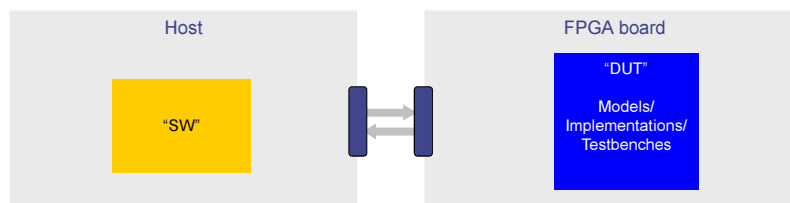
*BSV obviously addresses the first requirement.  
In this lecture we provide a brief overview of what Bluespec  
has to offer for the remaining requirements.*

3

© Bluespec, Inc., 2012

bluespec

## What is needed to execute your DUT on an FPGA?



Of course, your DUT needs to be synthesizable to an FPGA, but that is only part of the story ...

- High-level control of your DUT from the host (start, stop, reset, single step, ...)
- High-level communication of data between host and DUT
  - Abstraction above particular physical communication medium between host and DUT (e.g., PCIe, Ethernet, USB, ...)
  - Mapping between different bit-representations of data on host and DUT
  - TLM ("transaction level") transactors, instead of signal-level communication
- Synchronization of host with DUT
- Visibility, from the host, of data inside the DUT
- Libraries of pre-built components usable inside and beside the DUT

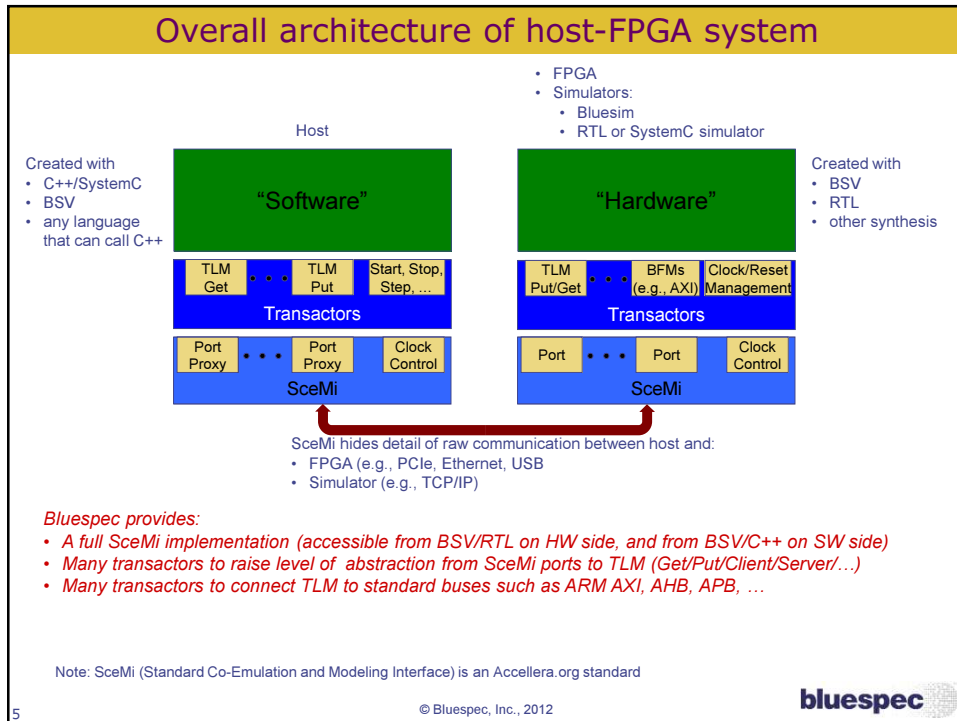
### *Analogy:*

- *We rarely write software for a "raw" computer---an OS (operating system) provides more comfortable, higher-level abstractions of the raw hardware and devices*
- *Similarly, Bluespec provides tools and infrastructure that make FPGA execution and debug easy, allowing you to focus on your DUT rather than wrestling with raw FPGA board hardware*

4

© Bluespec, Inc., 2012

bluespec

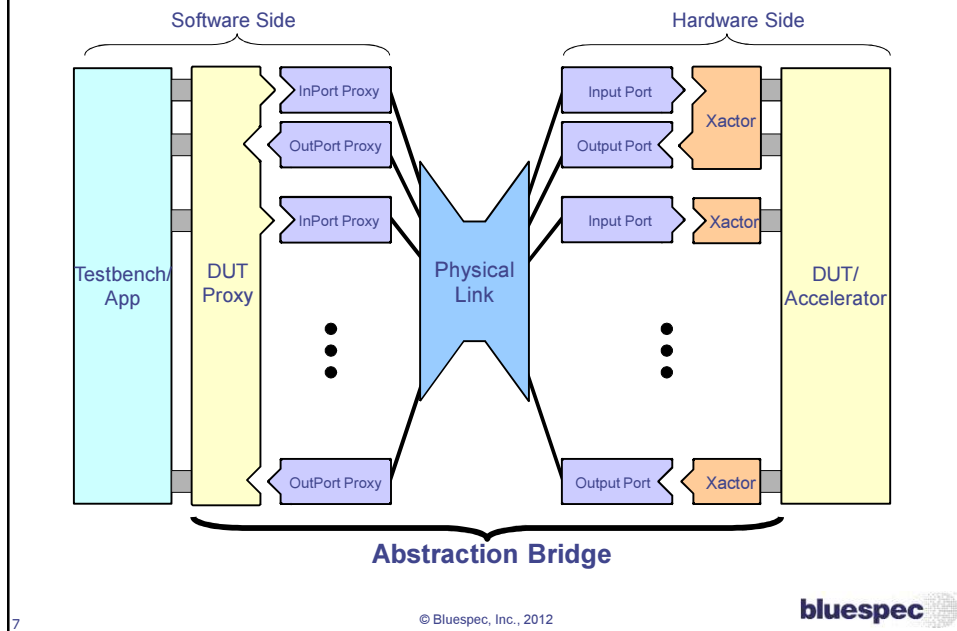


In the next few slides, we'll go bottom-up:

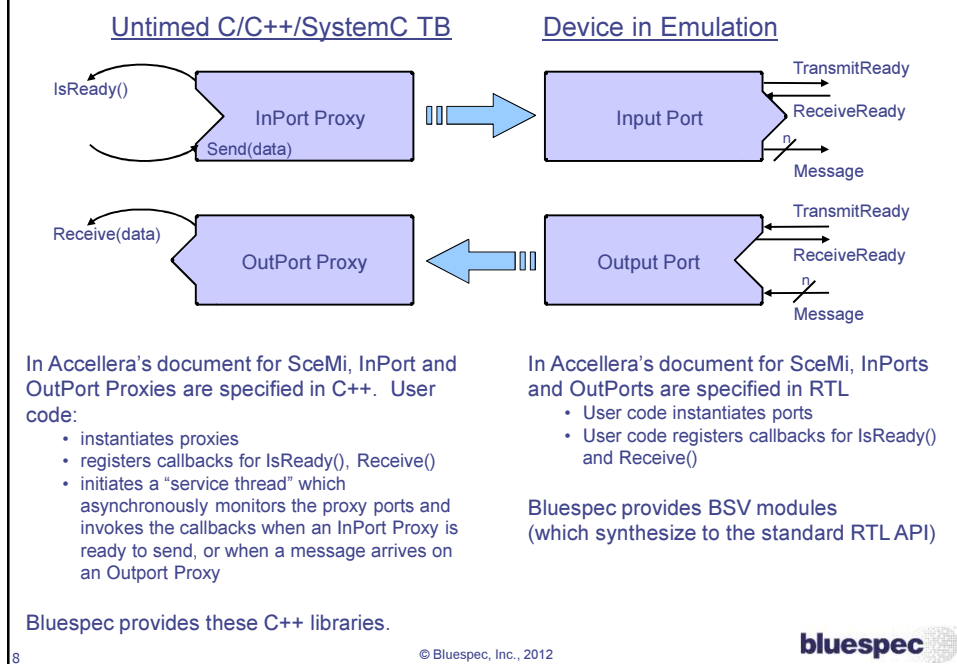
- SceMi
- Bluespec's transactors
- Overall flow

6
© Bluespec, Inc., 2012

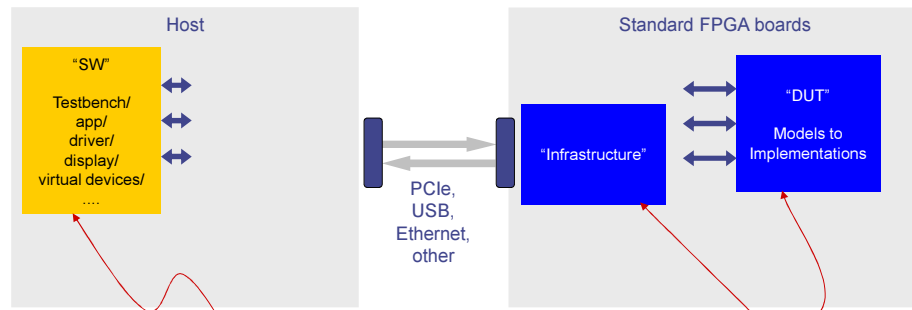
## SceMi communication: message ports and port-proxies



## SceMi message port APIs



## The need for DUT clock control



SW is usually "untimed".  
DUT usually has some notion of time (RTL clocks or "model clocks").  
Need mechanism to regulate (stall, release) DUT clock to keep it "in sync" with SW  
(*"time dilation"*).

E.g., burst output from DUT delivered as single TLM object to Testbench.  
E.g., request from DUT to Testbench, with response expected in "10 cycles".

Note: other parts of the FPGA must continue to be clocked while the DUT is stalled.  
E.g., request-response to Testbench while DUT is stalled.

9

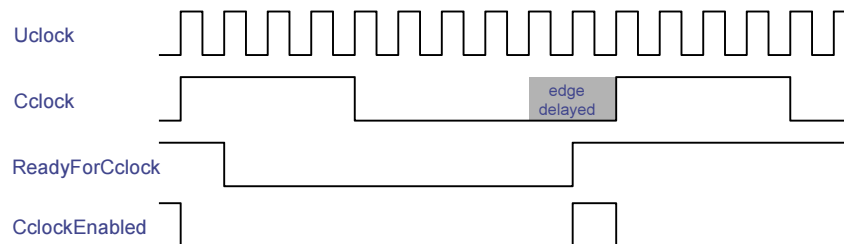
© Bluespec, Inc., 2012

bluespec

## SceMi clock controls: basic concepts

SceMi terminology:

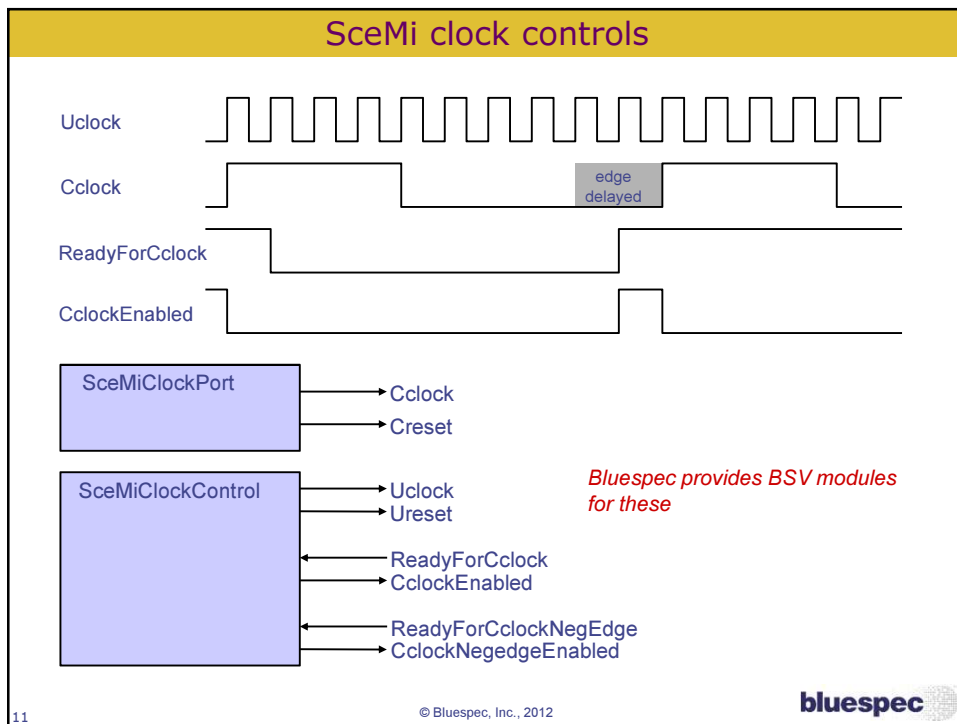
- "cclock" = *controlled* clock for the DUT on the FPGA
- "uclock" = *uncontrolled* clock on the FPGA (for SceMi infrastructure etc.)
- SceMi APIs provide for deriving and controlling cclocks from a uclock



10

© Bluespec, Inc., 2012

bluespec



### Beyond basic SceMi: TLM abstractions

Basic SceMi has rather low-level communication APIs:

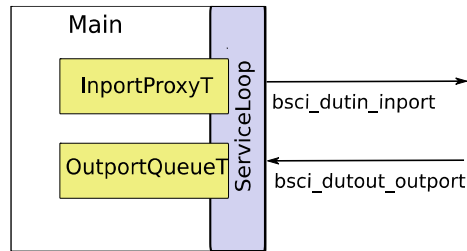
- Callbacks on the SW side
- InPorts and OutPorts on the HW side

Bluespec enables higher-level communications on top of SceMi:

- TLM (Transaction Level Modeling) operations Get/Put on both the SW and the HW side
- Automatic representation-conversion for data types

12 © Bluespec, Inc., 2012 **bluespec**

## TLM abstractions on the SW side



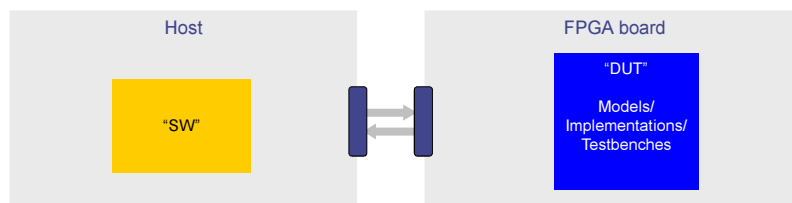
- Callbacks are messy because they are called “asynchronously” and need to be careful about consistent access to state shared with the main computation
- Bluespec provides simple queues:
  - Blocking/non-blocking put/get calls to send/receive messages
  - Callbacks are internal implementation details

13

© Bluespec, Inc., 2012

bluespec

## Data Type conversion



Consider the following similar type definitions in C++ and BSV:

// C++

```
enum Baz { Red, Green };
```

```
struct Foo {
    int x;
    bool y;
    Baz z;
};
```

// BSV

```
typedef enum { Red, Green } Baz;
```

```
typedef struct {
    int x;
    Bool y;
    Baz z;
} Foo;
```

Although logically the same, their bit-representations may be quite different.

Bluespec provides automatic converters that can be embedded in the SW-side transactors.

14

© Bluespec, Inc., 2012

bluespec

## TLM abstractions on the HW side



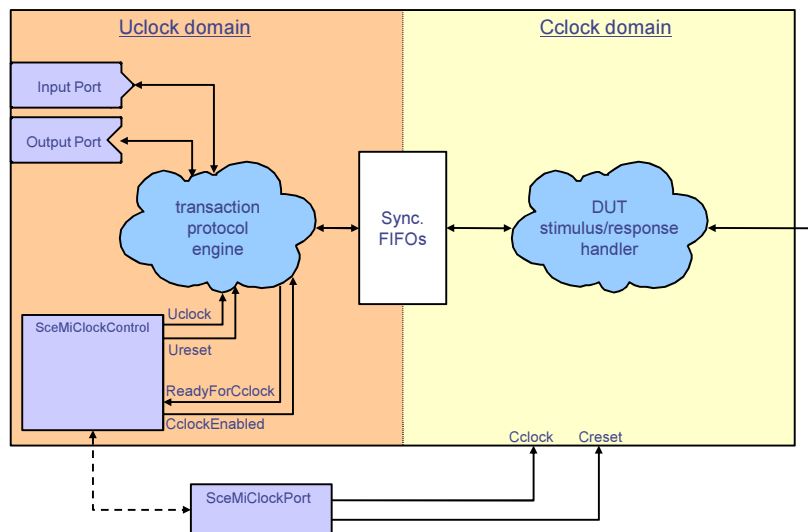
Bluespec provides standard BSV Get, Put, Client and Server transactors

15

© Bluespec, Inc., 2012

bluespec

## BSV HW-side transactors package much SceMi detail



16

© Bluespec, Inc., 2012

bluespec



## Bluespec's SceMi tool flow

A host-FPGA system is a classical “distributed system”:

- Hosts and FPGAs are multiple (and dissimilar) computing agents that communicate with each other

It is therefore not surprising that building and deploying such a system involves similar tasks as in classical distributed computing

- Building multiple executables with different tools
- Protocols for agents to establish communications with each other
- etc.

Bluespec's emulation toolbox has a 'build' script to automate this flow.

References:

`$(BLUESPEC_HOME)/training/BSV/tutorials/emVM/tutorial-emVM.pdf`

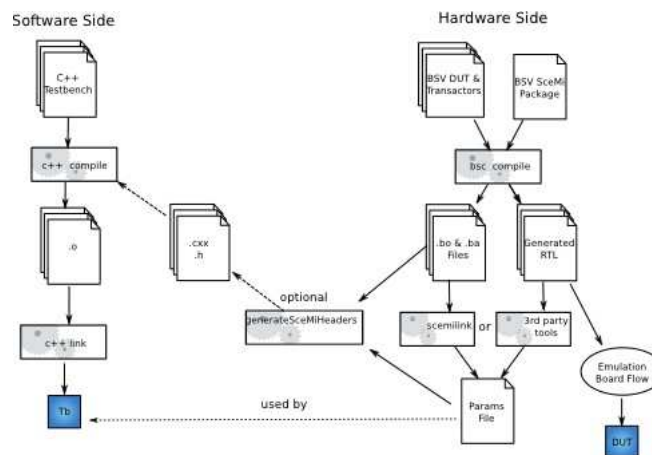
`$(BLUESPEC_HOME)/doc/BSV/emVM.pdf`

17

© Bluespec, Inc., 2012

bluespec

## Bluespec's SceMi tool flow



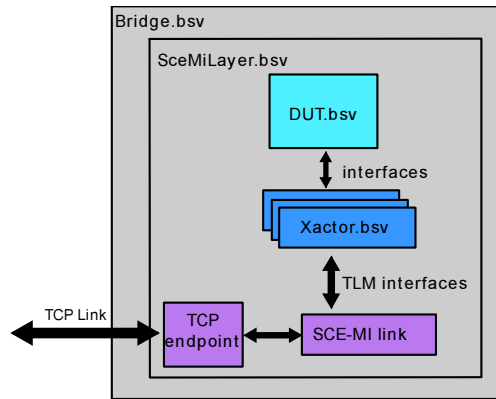
- The Params file is used by the SW side to discover features of the DUT (e.g., number and type of SceMi ports, communication protocol with HW side, etc.)
- The generateSceMiHeaders tool is used to generate type conversions, etc.

18

© Bluespec, Inc., 2012

bluespec

## Structure of BSV HW side



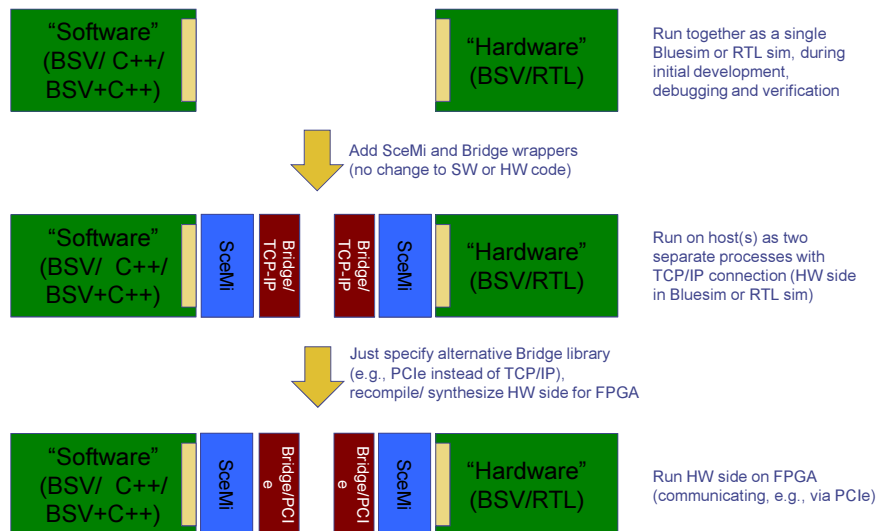
- SceMiLayer.bsv is written for each DUT
  - DUT-specific, Link-independent
  - Instantiates DUT
  - Instantiates SceMi ports/transactors and clock controls, connects them to DUT
    - Completely boiler-plate, if the DUT just has a standard interface (like Server)
- Bridge.bsv is a standard library file, chosen for the particular link type (e.g., TCP, PCIe, ...)
  - DUT-independent, Link-specific

19

© Bluespec, Inc., 2012

bluespec

## Typical BSV "refinement" methodology



*Note: because there is no change to the SW or HW side, and the SceMi and Bridge layers are just wrappers, changes in the SW or HW side can be repeatedly tested in all three scenarios.*

20

© Bluespec, Inc., 2012

bluespec

## Bluespec's emulation suite

The following slide mentions more advanced features of Bluespec's emulation suite, including:

- Probes
  - Ability to view waveforms from signals inside the FPGA DUT
    - (As usual, in BSV source terms, if the DUT was written in BSV)
- Hot-swap Co-simulation
  - Ability to co-simulate a Verilog module in a Verilog simulator in exact lock-step with the *same* module that is running in the FPGA DUT, thereby providing the full visibility of a Verilog simulator
- Automatic "control console" generation by which you can
  - control your FPGA DUT's execution
  - switch on/off probes, hot-swap co-simulation, etc.

These are described in detail in the references:

`$(BLUESPEC_HOME)/training/BSV/tutorials/emVM/tutorial-emVM.pdf`

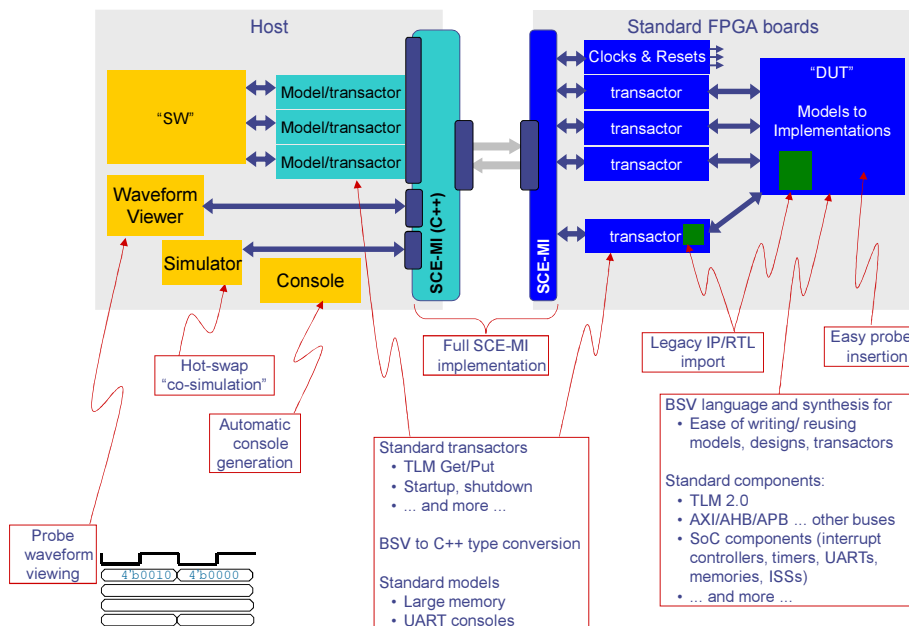
`$(BLUESPEC_HOME)/doc/BSV/emVM.pdf`

21

© Bluespec, Inc., 2012

bluespec

## Bluespec technology for host-FPGA execution and debug



22

© Bluespec, Inc., 2012

bluespec

## BClib (Bluespec Convey library)

Bluespec's library for developing applications on the Convey HPC (High Performance Computing) platform

(also portable to other host-FPGA HPC platforms)

23

© Bluespec, Inc., 2012

bluespec

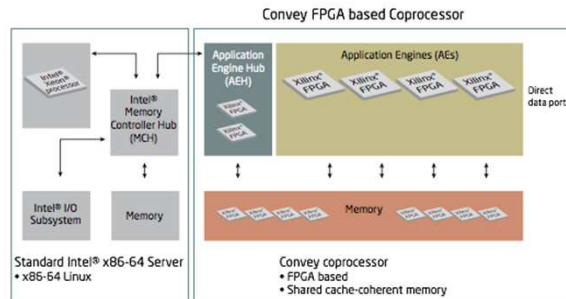
## Context: Convey "Hybrid Computing" (HC) platforms



(illustrations from Convey literature, [www.conveycomputer.com](http://www.conveycomputer.com))

Rack-mount box with two boards

- A standard Intel x86 server board
- A custom board with 4 user-programmable FPGAs (Virtex 5/6) and large, high-performance memory



x86 and FPGAs have shared view of all memory (coherent, same virtual address space)

Intel x86 "coprocessor instruction" protocol to hand-off from x86 to FPGA and back

Since coprocessor instruction semantics are implemented on FPGAs, each such setup is called a "personality"

FPGA memory subsystem:

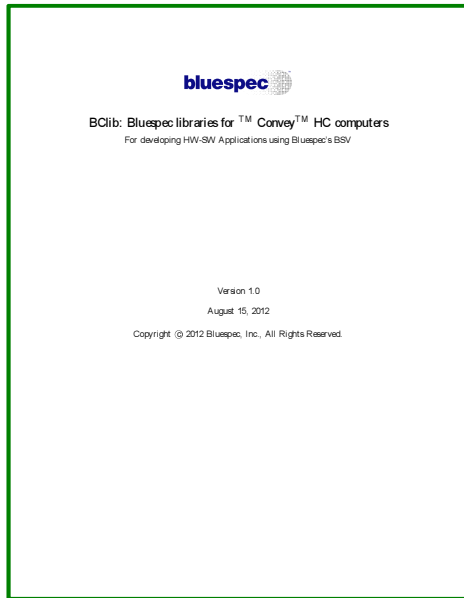
- 8 memory banks with parallel access from each FPGA (80 GB/s total)
- Pipelined (up to ~256 requests in flight)
- Out-of-order read responses. Requests/responses carry user-supplied 32b tag
- Fire-and-forget writes; weak ordering; memory 'fence' op to wait for write-completions

24

© Bluespec, Inc., 2012

bluespec

## BClib: complete Kit and Reference Manual



BClib kit includes:

- C and BSV libraries to dramatically simplify SW-HW linkage and communication
- Integrated Bluesim simulation of full SW-HW app
- BSV facilities to deal with high-performance memory pipelining, memory ordering, memory routing
- Reference Manual with detailed worked-out examples

*It is possible to build sophisticated apps now in days/weeks/few months, compared to a year or more*

*We expect BClib libraries to be easily portable to other host-FPGA platforms.*

25

© Bluespec, Inc., 2012

bluespec

bluespec

End

```

open FPGAs;
include BSW32;
module export_test_320000;
  type t = BitVec[32];
  type t2 = BitVec[64];
  type t3 = BitVec[128];
  type t4 = BitVec[256];
  type t5 = BitVec[512];
  type t6 = BitVec[1024];
  type t7 = BitVec[2048];
  type t8 = BitVec[4096];
  type t9 = BitVec[8192];
  type t10 = BitVec[16384];
  type t11 = BitVec[32768];
  type t12 = BitVec[65536];
  type t13 = BitVec[131072];
  type t14 = BitVec[262144];
  type t15 = BitVec[524288];
  type t16 = BitVec[1048576];
  type t17 = BitVec[2097152];
  type t18 = BitVec[4194304];
  type t19 = BitVec[8388608];
  type t20 = BitVec[16777216];
  type t21 = BitVec[33554432];
  type t22 = BitVec[67108864];
  type t23 = BitVec[134217728];
  type t24 = BitVec[268435456];
  type t25 = BitVec[536870912];
  type t26 = BitVec[1073741824];
  type t27 = BitVec[2147483648];
  type t28 = BitVec[4294967296];
  type t29 = BitVec[8589934592];
  type t30 = BitVec[17179869184];
  type t31 = BitVec[34359738368];
  type t32 = BitVec[68719476736];
  type t33 = BitVec[137438953472];
  type t34 = BitVec[274877906944];
  type t35 = BitVec[549755813888];
  type t36 = BitVec[1099511627776];
  type t37 = BitVec[2199023255552];
  type t38 = BitVec[4398046511104];
  type t39 = BitVec[8796093022208];
  type t40 = BitVec[17592186044416];
  type t41 = BitVec[35184372088832];
  type t42 = BitVec[70368744177664];
  type t43 = BitVec[140737488355328];
  type t44 = BitVec[281474976710656];
  type t45 = BitVec[562949953421312];
  type t46 = BitVec[1125899906842624];
  type t47 = BitVec[2251799813685248];
  type t48 = BitVec[4503599627370496];
  type t49 = BitVec[9007199254740992];
  type t50 = BitVec[18014398509481984];
  type t51 = BitVec[36028797018963968];
  type t52 = BitVec[72057594037927936];
  type t53 = BitVec[144115188075855872];
  type t54 = BitVec[288230376151711744];
  type t55 = BitVec[576460752303423488];
  type t56 = BitVec[1152921504606846976];
  type t57 = BitVec[2305843009213693952];
  type t58 = BitVec[4611686018427387904];
  type t59 = BitVec[9223372036854775808];
  type t60 = BitVec[18446744073709551616];
  type t61 = BitVec[36893488147419103232];
  type t62 = BitVec[73786976294838206464];
  type t63 = BitVec[147573952589676412928];
  type t64 = BitVec[295147905179352825856];
  type t65 = BitVec[590295810358705651712];
  type t66 = BitVec[1180591620717411303424];
  type t67 = BitVec[2361183241434822606848];
  type t68 = BitVec[4722366482869645213696];
  type t69 = BitVec[9444732965739290427392];
  type t70 = BitVec[18889465931478580854784];
  type t71 = BitVec[37778931862957161709568];
  type t72 = BitVec[75557863725914323419136];
  type t73 = BitVec[151115727451828646838272];
  type t74 = BitVec[302231454903657293676544];
  type t75 = BitVec[604462909807314587353088];
  type t76 = BitVec[1208925819614629174706176];
  type t77 = BitVec[2417851639229258349412352];
  type t78 = BitVec[4835703278458516698824704];
  type t79 = BitVec[9671406556917033397649408];
  type t80 = BitVec[19342813113834066795298816];
  type t81 = BitVec[38685626227668133590597632];
  type t82 = BitVec[77371252455336267181195264];
  type t83 = BitVec[154742504910672534362390528];
  type t84 = BitVec[309485009821345068724781056];
  type t85 = BitVec[618970019642690137449562112];
  type t86 = BitVec[1237940039285380274899124224];
  type t87 = BitVec[2475880078570760549798248448];
  type t88 = BitVec[4951760157141521099596496896];
  type t89 = BitVec[9903520314283042199192993792];
  type t90 = BitVec[19807040628566084398385987584];
  type t91 = BitVec[39614081257132168796771975168];
  type t92 = BitVec[79228162514264337593543950336];
  type t93 = BitVec[158456325028528675187087900672];
  type t94 = BitVec[316912650057057350374175801344];
  type t95 = BitVec[633825300114114700748351602688];
  type t96 = BitVec[1267650600228229401496703205376];
  type t97 = BitVec[2535301200456458802993406410752];
  type t98 = BitVec[5070602400912917605986812821504];
  type t99 = BitVec[10141204801825835211973625643008];
  type t100 = BitVec[20282409603651670423947251286016];
  type t101 = BitVec[40564819207303340847894502572032];
  type t102 = BitVec[81129638414606681695789005144064];
  type t103 = BitVec[162259276829213363391578010288128];
  type t104 = BitVec[324518553658426726783156020576256];
  type t105 = BitVec[649037107316853453566312041152512];
  type t106 = BitVec[1298074214633706907132624082305024];
  type t107 = BitVec[2596148429267413814265248164610048];
  type t108 = BitVec[5192296858534827628530496329220096];
  type t109 = BitVec[10384593717069655257060992658440192];
  type t110 = BitVec[20769187434139310514121985316880384];
  type t111 = BitVec[41538374868278621028243970633760768];
  type t112 = BitVec[83076749736557242056487941267521536];
  type t113 = BitVec[166153499473114484112975882535043072];
  type t114 = BitVec[332306998946228968225951765070086144];
  type t115 = BitVec[664613997892457936451903530140172288];
  type t116 = BitVec[1329227995784915872903807060280344576];
  type t117 = BitVec[2658455991569831745807614120560689152];
  type t118 = BitVec[5316911983139663491615228241121378304];
  type t119 = BitVec[10633823966279326983230456482242756608];
  type t120 = BitVec[21267647932558653966460912964485513216];
  type t121 = BitVec[42535295865117307932921825928971026432];
  type t122 = BitVec[85070591730234615865843651857942052864];
  type t123 = BitVec[170141183460469231731687303715884105728];
  type t124 = BitVec[340282366920938463463374607431768211456];
  type t125 = BitVec[680564733841876926926749214863536422912];
  type t126 = BitVec[1361129467683753853853498429727072845824];
  type t127 = BitVec[2722258935367507707706996859454145691648];
  type t128 = BitVec[5444517870735015415413993718908291383296];
  type t129 = BitVec[10889035741470030830827987437816582766592];
  type t130 = BitVec[21778071482940061661655974875633165533184];
  type t131 = BitVec[43556142965880123323311949751266331066368];
  type t132 = BitVec[87112285931760246646623899502532662132736];
  type t133 = BitVec[174224571863520493293247799005065324265472];
  type t134 = BitVec[348449143727040986586495598010130648530944];
  type t135 = BitVec[696898287454081973172991196020261297061888];
  type t136 = BitVec[1393796574908163946345982392040522594123776];
  type t137 = BitVec[2787593149816327892691964784081045188247552];
  type t138 = BitVec[5575186299632655785383929568162090376495104];
  type t139 = BitVec[11150372599265311570767859136324180752990208];
  type t140 = BitVec[22300745198530623141535718272648361505980416];
  type t141 = BitVec[44601490397061246283071436545296723011960832];
  type t142 = BitVec[89202980794122492566142873090593446023921664];
  type t143 = BitVec[178405961588244985132285746181186892047843328];
  type t144 = BitVec[356811923176489970264571492362373784095686656];
  type t145 = BitVec[713623846352979940529142984724747568191373312];
  type t146 = BitVec[1427247692705959881058285969449495136382746624];
  type t147 = BitVec[2854495385411919762116571938898990272765493248];
  type t148 = BitVec[5708990770823839524233143877797980545530986496];
  type t149 = BitVec[11417981541647679048466287755595961091061972992];
  type t150 = BitVec[22835963083295358096932575511191922182123945984];
  type t151 = BitVec[45671926166590716193865151022383844364247891968];
  type t152 = BitVec[91343852333181432387730302044767688728495783936];
  type t153 = BitVec[182687704666362864775460604089535377456991567872];
  type t154 = BitVec[365375409332725729550921208179070754913983135744];
  type t155 = BitVec[730750818665451459101842416358141509827966271488];
  type t156 = BitVec[1461501637330902918203684832716283019655932542976];
  type t157 = BitVec[2923003274661805836407369665432566039311865085952];
  type t158 = BitVec[5846006549323611672814739330865132078623730171904];
  type t159 = BitVec[11692013098647223345629478661730264157247460343808];
  type t160 = BitVec[23384026197294446691258957323460528314494920687616];
  type t161 = BitVec[46768052394588893382517914646921056628989841375232];
  type t162 = BitVec[93536104789177786765035829293842113257979682750464];
  type t163 = BitVec[187072209578355573530071658587684226515959365500928];
  type t164 = BitVec[374144419156711147060143317175368453031918731001856];
  type t165 = BitVec[748288838313422294120286634350736906063837462003712];
  type t166 = BitVec[1496577676626844588240573268701473812127674924007424];
  type t167 = BitVec[2993155353253689176481146537402947624255349848014848];
  type t168 = BitVec[5986310706507378352962293074805895248510699696029696];
  type t169 = BitVec[11972621413014756705924586149611790497021399392059392];
  type t170 = BitVec[23945242826029513411849172299223580994042798784118784];
  type t171 = BitVec[47890485652059026823698344598447161988085597568237568];
  type t172 = BitVec[95780971304118053647396689196894323976171195136475136];
  type t173 = BitVec[191561942608236107294793378393788647952342390272950272];
  type t174 = BitVec[383123885216472214589586756787577295904684780545900544];
  type t175 = BitVec[766247770432944429179173513575154591809369561091801088];
  type t176 = BitVec[1532495540865888858358347027150309183618739122183602176];
  type t177 = BitVec[3064991081731777716716694054300618367237478244367204352];
  type t178 = BitVec[6129982163463555433433388108601236734474956488734408704];
  type t179 = BitVec[12259964326927110866866776217202473468949912977468817408];
  type t180 = BitVec[24519928653854221733733552434404946937899825954937634816];
  type t181 = BitVec[49039857307708443467467104868809893875799651909875269632];
  type t182 = BitVec[98079714615416886934934209737619787751599303819750539264];
  type t183 = BitVec[196159429230833773869868419475239575503198607639501078528];
  type t184 = BitVec[392318858461667547739736838950479151006397215279002157056];
  type t185 = BitVec[784637716923335095479473677900958302012794430558004314112];
  type t186 = BitVec[1569275433846670190958947355801916604025588861116008628224];
  type t187 = BitVec[3138550867693340381917894711603833208051177722232017256448];
  type t188 = BitVec[6277101735386680763835789423207666416102355444464034512896];
  type t189 = BitVec[12554203470773361527671578846415332832204710888928069025792];
  type t190 = BitVec[25108406941546723055343157692830665664409421777856138051584];
  type t191 = BitVec[50216813883093446110686315385661331328818843555712276103168];
  type t192 = BitVec[100433627766186892221372630771322662657637687111424552206336];
  type t193 = BitVec[200867255532373784442745261542645325315275374222849104412672];
  type t194 = BitVec[401734511064747568885490523085290650630550748445698208825344];
  type t195 = BitVec[803469022129495137770981046170581301261101496891396417650688];
  type t196 = BitVec[1606938044258990275541962092341162602522202993782792835301376];
  type t197 = BitVec[3213876088517980551083924184682325205044405987565585670602752];
  type t198 = BitVec[6427752177035961102167848369364650410088811975131171341205504];
  type t199 = BitVec[12855504354071922204335696738729300820177623950262342682411008];
  type t200 = BitVec[25711008708143844408671393477458601640355247900524685364822016];
  type t201 = BitVec[51422017416287688817342786954917203280710495801049370729644032];
  type t202 = BitVec[102844034832575377634685573909834406561420991602098741459288064];
  type t203 = BitVec[205688069665150755269371147819668813122841983204197482918576128];
  type t204 = BitVec[411376139330301510538742295639337626245683966408394965837152256];
  type t205 = BitVec[822752278660603021077484591278675252491367932816789931674304512];
  type t206 = BitVec[1645504557321206042154969182557350504982735865633579863348609024];
  type t207 = BitVec[3291009114642412084309938365114701009965471731267159726697218048];
  type t208 = BitVec[6582018229284824168619876730229402019930943462534319453394436096];
  type t209 = BitVec[13164036458569648337239753460458804039861886925068638906788872192];
  type t210 = BitVec[26328072917139296674479506920917608079723773850137277813577744384];
  type t211 = BitVec[52656145834278593348959013841835216159447547700274555627155488768];
  type t212 = BitVec[105312291668557186697918027683670432318895095400549111254310977536];
  type t213 = BitVec[210624583337114373395836055367340864637790190801098222508621955072];
  type t214 = BitVec[421249166674228746791672110734681729275580381602196445017243910144];
  type t215 = BitVec[842498333348457493583344221469363458551160763204392890034487820288];
  type t216 = BitVec[1684996666696914987166688442938726917102321526408785780068975640576];
  type t217 = BitVec[3369993333393829974333376885877453834204643052817571560137951281152];
  type t218 = BitVec[6739986666787659948666753771754907668409286105635143120275902562304];
  type t219 = BitVec[13479973333575319897333507543509815336818572211270286240551805124608];
  type t220 = BitVec[26959946667150639794667015087019630673637144422540572481103610249216];
  type t221 = BitVec[53919893334301279589334030174039261347274288845081144962207220498432];
  type t222 = BitVec[107839786668602559178668060348078522694548577690162289924414440996864];
  type t223 = BitVec[215679573337205118357336120696157045389097155380324579848828881993728];
  type t224 = BitVec[431359146674410236714672241392314090778194310760649159697657763987456];
  type t225 = BitVec[862718293348820473429344482784628181556388621521298319395315527974912];
  type t226 = BitVec[1725436586697640946858688965569256363112777243042596638790631055949824];
  type t227 = BitVec[3450873173395281893717377931138512726225554486085193277581262111899648];
  type t228 = BitVec[6901746346790563787434755862277025452451108972170386555162524223799296];
  type t229 = BitVec[13803492693581127574869511724554050904902217944340773110325048447598592];
  type t230 = BitVec[27606985387162255149739023449108101809804435888681546220650096895197184];
  type t231 = BitVec[55213970774324510299478046898216203619608871777363092441300193790394368];
  type t232 = BitVec[110427941548649020598956093796432407239217743554726184882600387580788736];
  type t233 = BitVec[220855883097298041197912187592864814478435487109452369765200775161577472];
  type t234 = BitVec[441711766194596082395824375185729628956870974218904739530401550323154944];
  type t235 = BitVec[883423532389192164791648750371459257913741948437809479060803100646309888];
  type t236 = BitVec[1766847064778384329583297500742918515827483896875618958121606201292619776];
  type t237 = BitVec[3533694129556768659166595001485837031654967793751237916243212402585239552];
  type t238 = BitVec[70673882591
```