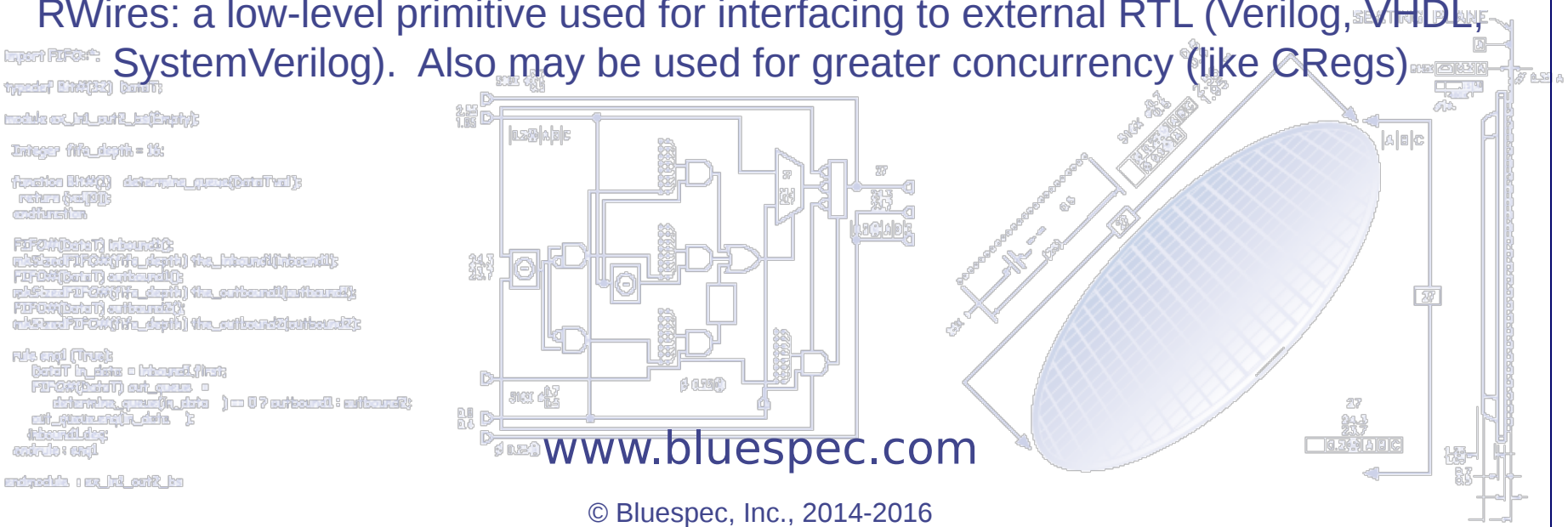
BSV Training

Lec_RWires

RWires: a low-level primitive used for interfacing to external RTL (Verilog, VHDL, SystemVerilog). Also may be used for greater concurrency (like CRegs)



www.bluespec.com

Introduction to RWires

RWires are low-level primitives in the BSV library
- (they can in fact be used to implement CRegs)

An RWire is an abstraction of ordinary wires in terms of BSV concepts of modules, methods, and method orderings.

RWires are frequently used at the boundary of a BSV design to interface with existing Verilog or VHDL for which the exact signal and protocol specifications are already given.

RWires must be used with great care:
- With all other primitives (including CRegs), functional correctness is typically preserved across arbitrary schedules (even one rule per clock). Improved scheduling is primarily concerned with tuning performance without affecting functional correctness.
- This is typically not true with RWires, which are only meaningful for intra-clock communication (and therefore assume certain minimum concurrency in schedules).

The basic primitive is called the “RWire”.
Special cases include PulseWires, Wires, DWires and BypassWires.

RWires

The most general form of "wire" family is the RWire interface and mkRWire primitive module:

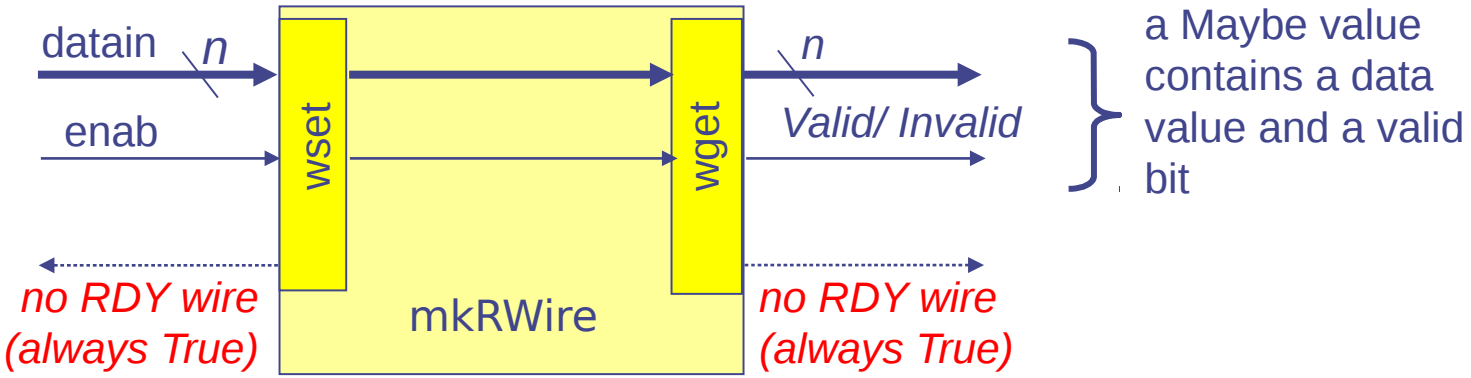
```
interface RWire #(type  t);
   method Action     wset (t datain);
   method Maybe#(t)  wget;
endinterface

module mkRWire (RWire#(t));   // primitive
```

Ordering constraint:
wset < wget

Suppose rule rA invokes rw.wset (x)
Then, in rule rB (logically later in the schedule):

- if (rw.wget matches tagged Valid .x) then rB knows that rA is firing in this clock and communicating the value x
- if (rw.wget matches tagged Invalid) then rB knows that rA is not firing in this clock

Implementation:



PulseWires

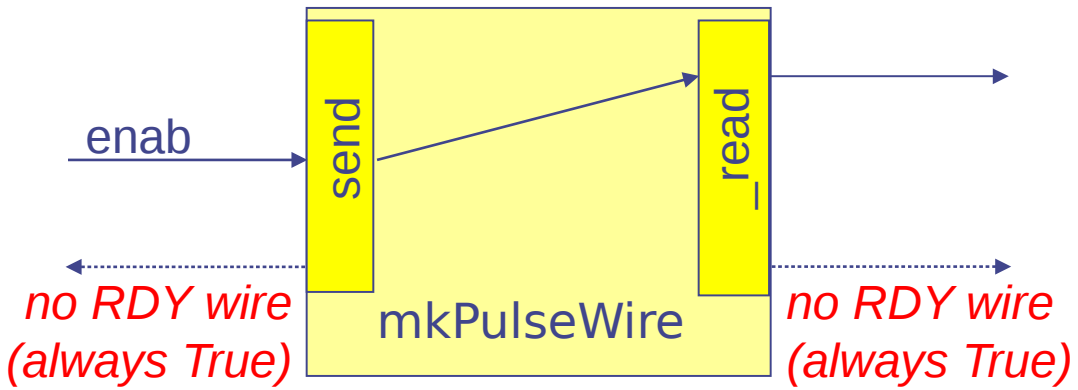
The PulseWire interface and mkPulseWire module is a special case of RWires where there is no value to be communicated

```
interface PulseWire;
    method Action send;
    method Bool   _read;
endinterface

module mkPulseWire (PulseWire);   // primitive
```

Ordering constraint:
send < _read

The _read method returns True if the send method is being invoked, else returns False

Implementation:



The Wire interface and mkDWire module

The Wire interface is just a synonym for the Reg interface.

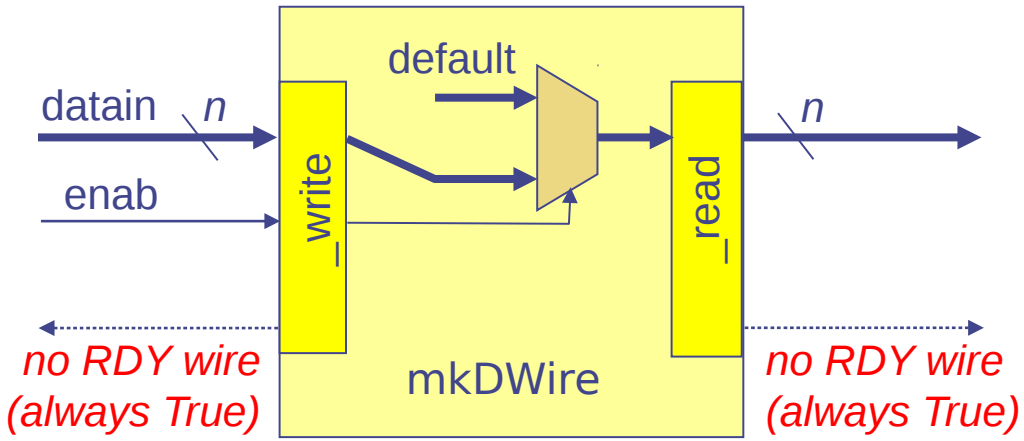
The mkDWire module is a “primitive” (D for default)

```
typedef  Reg#(t)  Wire#(type t);



module mkDWire #(t default) (Wire#(t));
  ... primitive imported Verilog ...
endmodule
```

Ordering constraint:
_write < _read

Note! this is the opposite schedule of mkReg!

Implementation:



Hands-on

- BSV-by-Example book: Examples in Chapter 8

End

Questions?

Join online forums at www.bluespec.com, and ask your question,
or send an e-mail to support@bluespec.com