Cryptol aided formal verification of VHDL code
Van der Waerden FPGA SAT solver

- SAT solver
- Complex combinatorial circuit
- Hand optimized
- Highly parameterized (K, L, N, device)
- The proof is computational
- If you are sure that the code is doing what you claim it is doing the proof is more valuable
Combinatorial circuit - simple example (X=A and B and C)

Cryptol code (f.cry)

f:([3][1]) -> [1];
f(x) = x@0 & x@1 & x@2;

VHDL CODE (example.vhdl)

library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

Entity example is
port (clk: in std_logic;
     i : in  std_logic_vector(3-1 downto 0);
     o : out std_logic);
end example;

architecture behavioral of example is
begin
  o <= i(0) and i(1) and i(2);
end behavioral;
Cryptol and equivalence checking

• Using Xilinx and Galois tools compile FPGA example.vhdl into formal model example_ecn.v.fm and using cryptol check equivalence between f.cry and example_ecn.v.fm

  • [mkouril@beowulf bin]$ ./cryptol_fpga f.cry
  • Cryptol FPGA version 0.8, Copyright (c) 2001-2006 Galois Connections Inc. A beta release. www.cryptol.net Type :? for help Loading "f.cry".. Checking types.. Processing.. Done!
  • cd> :equals f "example_ecn.v.fm"
  • EQUIVALENT
  • cd>
Van der Waerden solver

FPGA based SAT solver for Van der Waerden numbers

Parameters - k (number of colors), l (progression length), n (length of the partition), memory size

Total of stages = 12
Memory size = 1024
Combinatorial block
Combinatorial block

• VHDL version
  – thousands of lines of code
  – VHDL is not the most reader friendly language

• Cryptol version
  – around 200 lines
  – nicely structured
Limitations

• Can’t equivalence check the entire circuit because of the backtracking (SAT solver)

• Larger circuits tend to run out of memory or time

• Who verifies that my cryptol specification is correct?!!