***************Draft***************

Comparing SpinalHDL to Autofpga When a PLL is required Lesson Learned 03/20/21

***************Draft***************

The catzip project was created for the CatBoard using software packages (Autofpga, Icestorm, Yosys, Nextpnr, Verilator and ZipCPU).
The main clock on the Catboard is 100MHz. As the logic in FPGA is increased the speed needs to be reduced to obtain a working system. In the catzip.bin require a PLL to reduce the input freqency to 40Mhz.

The first steps in creating the catzip.bin was using Autofpga to generate a toplevel.v and main.v in the folder ~/testbuilds/catzip/rtl/catzip.

Autofpga requires a file ~/testbuilds/catzip/auto-data/clockpll40.txt with the following. Autofpga uses the tags

```
@$CLKFREQHZ=40000000
@PREFIX=clk
@CLOCK.NAME=clk
@TOP.DEFNS=
	wire		s_clk, s_reset;
@TOP.INSERT=
	assign  s_reset = 1'b0; // This design requires local, not global resets
@TOP.PORTLIST=i_clk
@TOP.IODECL= input  i_clk;
@REGDEFS.H.DEFNS=
#define		CLKFREQHZ@$CLKFREQHZ
@BDEF.DEFN=
#define		CLKFREQHZ@$CLKFREQHZ
@CLOCK.NAME=clk
@CLOCK.FREQUENCY= @$CLKFREQHZ
```

```
##
##
@$CLKFREQHZ=40000000
@PREFIX=clk
@CLOCK.NAME=clk
@TOP.DEFNS=
	wire		s_clk, s_reset;
@TOP.INSERT=
	assign		s_reset = 1'b0; // This design requires local, not global resets

`ifdef VERILATOR
	assign		s_clk = i_clk;
`else
	wire  clk_40mhz, pll_locked;
	SB_PLL40_CORE #(
		.FEEDBACK_PATH("SIMPLE"),
		.DELAY_ADJUSTMENT_MODE_FEEDBACK("FIXED"),
		.DELAY_ADJUSTMENT_MODE_RELATIVE("FIXED"),
		.PLLOUT_SELECT("GENCLK"),
		.FDA_FEEDBACK(4'b1111),
		.FDA_RELATIVE(4'b1111),
		.DIVR(4'b0100),		// DIVR =  4
		.DIVQ(7'b0011111),		// DIVQ =  31
		.DIVF(3'b100),		// DIVF =  4
		.FILTER_RANGE(3'b010)	// FILTER_RANGE = 2
	) plli (
		.REFERENCECLK	(i_clk	),
		.PLLOUTCORE	(clk_40mhz	),
		.LOCK		(pll_locked  ),
		.BYPASS		(1'b0		),
		.RESETB		(1'b1		)
	);
	//SB_GB global_buffer(clk_40mhz, s_clk);
	assign	s_clk = clk_40mhz;
`endif
@TOP.PORTLIST=i_clk
@TOP.IODECL= input  i_clk;
@REGDEFS.H.DEFNS=
#define	CLKFREQHZ	@$CLKFREQHZ
@BDEF.DEFN=
#define	CLKFREQHZ	@$CLKFREQHZ
@CLOCK.NAME=clk
@CLOCK.FREQUENCY= @$CLKFREQHZ
```

The file /home/devel/testbuilds/catzip/auto-data/Makefile requires the following:

```
DATA := global.txt bkram.txt buserr.txt clockpll40.txt	\
	pic.txt pwrcount.txt version.txt			\
	hbconsole.txt gpio.txt dlyarbiter.txt zipbones.txt sdramdev.txt \
	mem_sdram_bkram.txt sdramscope.txt

```

/home/devel/icestorm/icefuzz/tests/sb_pll40_core.v

```
module top(
	input   REFERENCECLK,
	output  PLLOUTCORE,
	output  PLLOUTGLOBAL,
	input   EXTFEEDBACK,
	input   [7:0] DYNAMICDELAY,
	output  LOCK,
	input   BYPASS,
	input   RESETB,
	input   LATCHINPUTVALUE,

	//Test Pins
	output  SDO,
	input   SDI,
	input   SCLK
);
	SB_PLL40_CORE #(
		.FEEDBACK_PATH("DELAY"),
		// .FEEDBACK_PATH("SIMPLE"),
		// .FEEDBACK_PATH("PHASE_AND_DELAY"),
		// .FEEDBACK_PATH("EXTERNAL"),

		.DELAY_ADJUSTMENT_MODE_FEEDBACK("FIXED"),
		// .DELAY_ADJUSTMENT_MODE_FEEDBACK("DYNAMIC"),

		.DELAY_ADJUSTMENT_MODE_RELATIVE("FIXED"),
		// .DELAY_ADJUSTMENT_MODE_RELATIVE("DYNAMIC"),

		.PLLOUT_SELECT("GENCLK"),
		// .PLLOUT_SELECT("GENCLK_HALF"),
		// .PLLOUT_SELECT("SHIFTREG_90deg"),
		// .PLLOUT_SELECT("SHIFTREG_0deg"),

		.SHIFTREG_DIV_MODE(1'b0),
		.FDA_FEEDBACK(4'b1111),
		.FDA_RELATIVE(4'b1111),
		.DIVR(4'b0000),
		.DIVF(7'b0000000),
		.DIVQ(3'b001),
		.FILTER_RANGE(3'b000),
		.ENABLE_ICEGATE(1'b0),
		.TEST_MODE(1'b0)
	) uut (
		.REFERENCECLK   (REFERENCECLK   ),
		.PLLOUTCORE     (PLLOUTCORE     ),
		.PLLOUTGLOBAL   (PLLOUTGLOBAL   ),
		.EXTFEEDBACK    (EXTFEEDBACK    ),
		.DYNAMICDELAY   (DYNAMICDELAY   ),
		.LOCK           (LOCK           ),
		.BYPASS         (BYPASS         ),
            .RESETB        (RESETB        ),
            .LATCHINPUTVALUE(LATCHINPUTVALUE),
            .SDO           (SDO           ),
            .SDI           (SDI           ),
            .SCLK          (SCLK          )
      );
endmodule
```

Using SpinalHDL to create pll Verilog.

The file /home/devel/SpinalKeyboardtest/src/main/scala/testcode/toplevel.scala is what determines the what is placed in the file /home/devel/SpinalKeyboardtest/toplevel_pll.v The first is used to separate the code into a package. The green highlted area are just comments.

```
package testcode

import spinal.core._
/*
`ifdef VERILATOR
      assign      s_clk = i_clk;
`else
      wire   clk_40mhz, pll_locked;
      SB_PLL40_CORE #(
            .FEEDBACK_PATH("SIMPLE"),
            .DELAY_ADJUSTMENT_MODE_FEEDBACK("FIXED"),
            .DELAY_ADJUSTMENT_MODE_RELATIVE("FIXED"),
            .PLLOUT_SELECT("GENCLK"),
            .FDA_FEEDBACK(4'b1111),
            .FDA_RELATIVE(4'b1111),
            .DIVR(4'b0100),          // DIVR =  4
            .DIVQ(7'b0011111),             // DIVQ =  31
            .DIVF(3'b100),           // DIVF =  4
            .FILTER_RANGE(3'b010)       // FILTER_RANGE = 2
      ) plli (
            .REFERENCECLK     (i_clk        ),
            .PLLOUTCORE     (clk_40mhz    ),
            .LOCK          (pll_locked  ),
            .BYPASS        (1'b0        ),
            .RESETB        (1'b1        )
      );
      //SB_GB global_buffer(clk_40mhz, s_clk);
      assign      s_clk = clk_40mhz;
`endif
```

```
  SB_PLL40_CORE #(
    .FEEDBACK_PATH("SIMPLE"),
    .DIVR(4'b0010),
    .DIVF(7'h16),
    .DIVQ(3'b110),
    .FILTER_RANGE(3'b010)
  ) plli (
    .REFERENCECLK    (io_CLK_100       ), //i
    .PLLOUTCORE      (plli_PLLOUTCORE  ), //o
    .LOCK            (plli_LOCK        ), //o
    .BYPASS          (_zz_1            ), //i
    .RESETB          (_zz_2            )  //i
  );
  assign _zz_1 = 1'b0;
  assign _zz_2 = 1'b1;
*/

case class SB_PLL40_CORE() extends BlackBox {
  val REFERENCECLK: Bool = in Bool
  val PLLOUTCORE, LOCK = out Bool
  val BYPASS = in Bits()
  val RESETB = in Bits()
  addGeneric("FEEDBACK_PATH", "SIMPLE")
  addGeneric("DIVR", B(2,4 bits))
  addGeneric("DIVF", B(22,7 bits))
  addGeneric("DIVQ", B(6,3 bits))
  addGeneric("FILTER_RANGE", B(2,3 bits))
  //addGeneric("BYPASS", B(0,1 bits))
  //addGeneric("RESETB", "RESET")
  //addGeneric("EXTFEEDBACK","()")
}
class toplevel_pll() extends Component {
  val io = new Bundle() {
    val CLK_100 = in Bool
  }
  val plli: SB_PLL40_CORE = new SB_PLL40_CORE()
  plli.REFERENCECLK <> io.CLK_100
  plli.BYPASS <> B(0,1 bit)
  plli.RESETB <> B(1,1 bit)
}
//Generate the toplevel_pll Verilog
object toplevel_pllVerilog {
  def main(args: Array[String]) {
    SpinalVerilog(new toplevel_pll)
  }
}
```

The blue highted area above is the scala code used to generate the file toplevel_pll.v When you execute the command below SpinalHDL generates the file /home/devel/SpinalKeyboardtest/toplevel_pll.v.

```
sbt run
[info] welcome to sbt 1.3.13 (Raspbian Java 11.0.9.1)
[info] loading settings for project spinalkeyboardtest-build from plugins.sbt ...
[info] loading project definition from /home/devel/SpinalKeyboardtest/project
[info] loading settings for project spinalkeyboardtest from build.sbt ...
[info] set current project to SpinalTemplateSbt (in build file:/home/devel/SpinalKeyboardtest/)
[info] Compiling 1 Scala source to /home/devel/SpinalKeyboardtest/target/scala-2.11/classes ...
[warn] Multiple main classes detected.  Run 'show discoveredMainClasses' to see the list

Multiple main classes detected, select one to run:

 [1] CatGenerateTop
 [2] GenerateIP
 [3] GenerateTop
 [4] Testinalkeyboardtest / Compile / compile 0s
 [5] catboard.Main
 [6] mylib.MyTopLevelSim
 [7] mylib.MyTopLevelVerilog
 [8] mylib.MyTopLevelVerilogWithCustomConfig
 [9] mylib.MyTopLevelVhdl
 [10] spinal.lib.com.axi.MasterTopLevelVerilog
 [11] testcode.toplevel_pllVerilog



[info] running (fork) testcode.toplevel_pllVerilog
[info] [Runtime] SpinalHDL v1.4.3    git head : adf552d8f500e7419fff395b7049228e4bc5de26
[info] [Runtime] JVM max memory : 944.0MiB
[info] [Runtime] Current date : 2021.03.20 09:44:43
[info] [Progress] at 0.000 : Elaborate components
[info] [Progress] at 0.357 : Checks and transforms
[info] [Progress] at 0.593 : Generate Verilog
[info] [Warning] 2 signals were pruned. You can call printPruned on the backend report to get more
informations.
[info] [Done] at 0.882
[success] Total time: 45 s, completed Mar 20, 2021, 9:44:44 AM
```

```
// Generator : SpinalHDL v1.4.3    git head : adf552d8f500e7419fff395b7049228e4bc5de26
// Component : toplevel_pll
// Git hash  : c55b26fadf47bc93e2e90c58db49fa4d197839fc
```

/home/devel/SpinalKeyboardtest/toplevel_pll.v

```
module toplevel_pll (
  input            io_CLK_100
);
```

```
  wire       [0:0]    _zz_1;
  wire       [0:0]    _zz_2;
  wire                plli_PLLOUTCORE;
  wire                plli_LOCK;

  SB_PLL40_CORE #(
    .FEEDBACK_PATH("SIMPLE"),
    .DIVR(4'b0010),
    .DIVF(7'h16),
    .DIVQ(3'b110),
    .FILTER_RANGE(3'b010)
  ) plli (
    .REFERENCECLK    (io_CLK_100        ), //i
    .PLLOUTCORE      (plli_PLLOUTCORE  ), //o
    .LOCK            (plli_LOCK        ), //o
    .BYPASS          (_zz_1            ), //i
    .RESETB          (_zz_2            )  //i
  );
  assign _zz_1 = 1'b0;
  assign _zz_2 = 1'b1;

endmodule
```