

```

timescale 1ns / 1ps
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 01/25/2018 11:19:41 AM
// Design Name:
// Module Name: lab4_clks
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

module lab4_clks(
    input clk_in,
    input greset, //btnR
    output clk,    // system clock for Lab 4
    output digsel, // signal to advance ring counter for display
    output fastclk);

    wire clk_int;
    assign fastclk = clk_int;
    clk_wiz_0 my_clk_inst (.clk_out1(clk_int), .reset(greset), .locked(),
.clk_in1(clk_in));
    clkcntrl4 slowclk (.clk_int(clk_int), .seldig(digsel), .clk_out(clk));

    STARTUPE2 #(.PROG_USR("FALSE"), // Activate program event security feature.
Requires encrypted bitstreams.
                .SIM_CCLK_FREQ(0.0) // Set the Configuration Clock
Frequency(ns) for simulation.
                )
    STARTUPE2_inst (.CFGCLK(), // 1-bit output: Configuration main clock
output
                .CFGMCLK(), // 1-bit output: Configuration internal
oscillator clock output
                .EOS(), // 1-bit output: Active high output signal

```

```

indicating the End Of Startup.
        .PREQ(), // 1-bit output: PROGRAM request to fabric
output
        .CLK(), // 1-bit input: User start-up clock input
        .GSR(greset), // 1-bit input: Global Set/Reset input
(GSR cannot be used for the port name)
        .GTS(), // 1-bit input: Global 3-state input (GTS
cannot be used for the port name)
        .KEYCLEARB(), // 1-bit input: Clear AES Decrypter Key
input from Battery-Backed RAM (BBRAM)
        .PACK(), // 1-bit input: PROGRAM acknowledge input
        .USRCCLKO(), // 1-bit input: User CCLK input
        .USRCCLKTS(), // 1-bit input: User CCLK 3-state enable
input
        .USRDONEO(), // 1-bit input: User DONE pin output
control
        .USRDONETS() // 1-bit input: User DONE 3-state enable
output
    ); // End of STARTUPE2_inst instantiation

endmodule

```

```

module clk_wiz_0

(// Clock in ports
// Clock out ports
output        clk_out1,
// Status and control signals
input         reset,
output        locked,
input         clk_in1
);
// Input buffering
//-----

wire clk_in1_clk_wiz_0;
wire clk_in2_clk_wiz_0;
IBUF clkin1_ibufg
    (.O (clk_in1_clk_wiz_0),
     .I (clk_in1));

// Clocking PRIMITIVE
//-----

// Instantiation of the MMCM PRIMITIVE
//      * Unused inputs are tied off

```

```
//      * Unused outputs are labeled unused
```

```
wire      clk_out1_clk_wiz_0;
wire      clk_out2_clk_wiz_0;
wire      clk_out3_clk_wiz_0;
wire      clk_out4_clk_wiz_0;
wire      clk_out5_clk_wiz_0;
wire      clk_out6_clk_wiz_0;
wire      clk_out7_clk_wiz_0;

wire [15:0] do_unused;
wire      drdy_unused;
wire      psdone_unused;
wire      locked_int;
wire      clkfbout_clk_wiz_0;
wire      clkfbout_buf_clk_wiz_0;
wire      clkfboutb_unused;
    wire clkout0b_unused;
wire      clkout1_unused;
wire      clkout1b_unused;
wire      clkout2_unused;
wire      clkout2b_unused;
wire      clkout3_unused;
wire      clkout3b_unused;
wire      clkout4_unused;
wire      clkout5_unused;
wire      clkout6_unused;
wire      clkfbstopped_unused;
wire      clkinstopped_unused;
wire      reset_high;
```

```
MMCME2_ADV
```

```
#(.BANDWIDTH      ("OPTIMIZED"),
  .CLKOUT4_CASCADE ("FALSE"),
  .COMPENSATION    ("ZHOLD"),
  .STARTUP_WAIT    ("FALSE"),
  .DIVCLK_DIVIDE   (1),
  .CLKFBOUT_MULT_F (9.125),
  .CLKFBOUT_PHASE  (0.000),
  .CLKFBOUT_USE_FINE_PS ("FALSE"),
  .CLKOUT0_DIVIDE_F (36.500),
  .CLKOUT0_PHASE   (0.000),
  .CLKOUT0_DUTY_CYCLE (0.500),
  .CLKOUT0_USE_FINE_PS ("FALSE"),
  .CLKIN1_PERIOD   (10.0))
```

```
mmcm_adv_inst
```

```

// Output clocks
(
    .CLKFBOUT            (clkfbout_clk_wiz_0),
    .CLKFBOUTB           (clkfboutb_unused),
    .CLKOUT0             (clk_out1_clk_wiz_0),
    .CLKOUT0B            (clkout0b_unused),
    .CLKOUT1             (clkout1_unused),
    .CLKOUT1B            (clkout1b_unused),
    .CLKOUT2             (clkout2_unused),
    .CLKOUT2B            (clkout2b_unused),
    .CLKOUT3             (clkout3_unused),
    .CLKOUT3B            (clkout3b_unused),
    .CLKOUT4             (clkout4_unused),
    .CLKOUT5             (clkout5_unused),
    .CLKOUT6             (clkout6_unused),
    // Input clock control
    .CLKFBIN             (clkfbout_buf_clk_wiz_0),
    .CLKIN1              (clk_in1_clk_wiz_0),
    .CLKIN2              (1'b0),
    // Tied to always select the primary input clock
    .CLKINSEL            (1'b1),
    // Ports for dynamic reconfiguration
    .DADDR               (7'h0),
    .DCLK                (1'b0),
    .DEN                 (1'b0),
    .DI                  (16'h0),
    .DO                  (do_unused),
    .DRDY                (drdy_unused),
    .DWE                 (1'b0),
    // Ports for dynamic phase shift
    .PSCLK               (1'b0),
    .PSEN                (1'b0),
    .PSINCDEC            (1'b0),
    .PSDONE              (psdone_unused),
    // Other control and status signals
    .LOCKED              (locked_int),
    .CLKINSTOPPED        (clkinstopped_unused),
    .CLKFBSTOPPED        (clkfbstopped_unused),
    .PWRDWN              (1'b0),
    .RST                 (reset_high));
assign reset_high = reset;

    assign locked = locked_int;
// Clock Monitor clock assigning
//-----
// Output buffering

```

```
//-----
```

```
BUFG clkf_buf  
  (.O (clkfbout_buf_clk_wiz_0),  
    .I (clkfbout_clk_wiz_0));
```

```
BUFG clkout1_buf  
  (.O (clk_out1),  
    .I (clk_out1_clk_wiz_0));
```

```
endmodule
```

```
module clkcntrl4(  
    input clk_int,  
    output seldig,  
    output clk_out);
```

```
    //wire XLXN_38;  
    //wire XLXN_39;  
    wire XLXN_44;  
    wire XLXN_47;  
    wire XLXN_70;  
    wire XLXN_71;  
    wire XLXN_72;  
    wire XLXN_73;  
    //wire XLXN_74;  
    wire XLXN_75;  
    wire XLXN_76;  
    wire XLXN_77;  
    wire clkb2_DUMMY;
```

```
GND XLXI_24 (.G(XLXN_44));
```

```
(* HU_SET = "XLXI_37_73" *)
```

```
CB4CE_MXILINX_clkcntrl4 XLXI_37 (.C(clkb2_DUMMY),  
    .CE(XLXN_73),  
    .CLR(XLXN_76),  
    .CEO(XLXN_72),  
    .Q0(),  
    .Q1(),  
    .Q2(),
```

```

        .Q3() ,
        .TC()) ;

(* HU_SET = "XLXI_38_74" *)
CB4CE_MXILINX_clkcntrl4 XLXI_38 (.C(clkb2_DUMMY) ,
        .CE(XLXN_72) ,
        .CLR(XLXN_76) ,
        .CEO(XLXN_70) ,
        .Q0() ,
        .Q1() ,
        .Q2() ,
        .Q3() ,
        .TC()) ;

(* HU_SET = "XLXI_39_75" *)
CB4CE_MXILINX_clkcntrl4 XLXI_39 (.C(clkb2_DUMMY) ,
        .CE(XLXN_70) ,
        .CLR(XLXN_76) ,
        .CEO(XLXN_71) ,
        .Q0() ,
        .Q1() ,
        .Q2() ,
        .Q3(XLXN_77) ,
        .TC()) ;

//(* HU_SET = "XLXI_40_76" *)
CB4CE_MXILINX_clkcntrl4 XLXI_40 (.C(clk_out) ,
        .CE(XLXN_73) ,
        .CLR(XLXN_76) ,
        .CEO() ,
        .Q0() ,
        .Q1() ,
        .Q2() ,
        .Q3() ,
        .TC(XLXN_75)) ;

VCC XLXI_41 (.P(XLXN_73)) ;
GND XLXI_43 (.G(XLXN_76)) ;
BUF XLXI_328 (.I(clk_int) ,
        .O(clkb2_DUMMY)) ;

`ifdef XILINX_SIMULATOR
    BUF XLXI_336 (.I(XLXN_75) , .O(seldig)) ;
    BUFG XLXI_399 (.I(clk_int) , .O(clk_out)) ;
`else
    BUF XLXI_336 (.I(XLXN_75) , .O(seldig)) ;
    BUFG XLXI_401 (.I(XLXN_77) , .O(clk_out)) ;
`endif

endmodule

```

```
module FTCE_MXILINX_clkcntrl4(C,  
                                CE,  
                                CLR,  
                                T,  
                                Q);
```

```
    parameter INIT = 1'b0;
```

```
    input C;  
    input CE;  
    input CLR;  
    input T;  
    output Q;
```

```
    wire TQ;  
    wire Q_DUMMY;
```

```
    assign Q = Q_DUMMY;  
    XOR2 I_36_32 (.I0(T),  
                  .I1(Q_DUMMY),  
                  .O(TQ));
```

```
    ///(* RLOC = "X0Y0" *)
```

```
    FDCE I_36_35 (.C(C),  
                  .CE(CE),  
                  .CLR(CLR),  
                  .D(TQ),  
                  .Q(Q_DUMMY));
```

```
endmodule
```

```
`timescale 1ns / 1ps
```

```
module CB4CE_MXILINX_clkcntrl4(C,  
                                CE,  
                                CLR,  
                                CEO,  
                                Q0,  
                                Q1,  
                                Q2,  
                                Q3,  
                                TC);
```

```
    input C;  
    input CE;  
    input CLR;  
    output CEO;  
    output Q0;
```

```

output Q1;
output Q2;
output Q3;
output TC;

wire T2;
wire T3;
wire XLXN_1;
wire Q0_DUMMY;
wire Q1_DUMMY;
wire Q2_DUMMY;
wire Q3_DUMMY;
wire TC_DUMMY;

assign Q0 = Q0_DUMMY;
assign Q1 = Q1_DUMMY;
assign Q2 = Q2_DUMMY;
assign Q3 = Q3_DUMMY;
assign TC = TC_DUMMY;
(* HU_SET = "I_Q0_69" *)
FTCE_MXILINX_clkcntrl4 #( .INIT(1'b0) ) I_Q0 (.C(C),
                                                .CE(CE),
                                                .CLR(CLR),
                                                .T(XLXN_1),
                                                .Q(Q0_DUMMY));

(* HU_SET = "I_Q1_70" *)
FTCE_MXILINX_clkcntrl4 #( .INIT(1'b0) ) I_Q1 (.C(C),
                                                .CE(CE),
                                                .CLR(CLR),
                                                .T(Q0_DUMMY),
                                                .Q(Q1_DUMMY));

(* HU_SET = "I_Q2_71" *)
FTCE_MXILINX_clkcntrl4 #( .INIT(1'b0) ) I_Q2 (.C(C),
                                                .CE(CE),
                                                .CLR(CLR),
                                                .T(T2),
                                                .Q(Q2_DUMMY));

(* HU_SET = "I_Q3_72" *)
FTCE_MXILINX_clkcntrl4 #( .INIT(1'b0) ) I_Q3 (.C(C),
                                                .CE(CE),
                                                .CLR(CLR),
                                                .T(T3),
                                                .Q(Q3_DUMMY));

AND4 I_36_31 (.I0(Q3_DUMMY),
              .I1(Q2_DUMMY),
              .I2(Q1_DUMMY),

```



```

        .I3(Q0_DUMMY),
        .O(TC_DUMMY));
AND3    I_36_32 (.I0(Q2_DUMMY),
        .I1(Q1_DUMMY),
        .I2(Q0_DUMMY),
        .O(T3));
AND2    I_36_33 (.I0(Q1_DUMMY),
        .I1(Q0_DUMMY),
        .O(T2));
VCC     I_36_58 (.P(XLXN_1));
AND2    I_36_67 (.I0(CE),
        .I1(TC_DUMMY),
        .O(CEO));
endmodule

```