AdcClockAlignment

Marc Defossez
Sr. Staff Applications Engineer

Created: August 22, 2007
Modified: September 27, 2010

DISCLAIMER:

Disclaimer:

This disclaimer is not a license and does not grant any rights to the materials distributed herewith. Except as otherwise provided in a valid license issued to you by Xilinx, and to the maximum extent permitted by applicable law: (1) THESE MATERIALS ARE MADE AVAILABLE "AS IS" AND WITH ALL FAULTS, AND XILINX HEREBY DISCLAIMS ALL WARRANTIES AND CONDITIONS, EXPRESS, IMPLIED, OR STATUTORY, INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, OR FITNESS FOR ANY PARTICULAR PURPOSE; and (2) Xilinx shall not be liable (whether in contract or tort, including negligence, or under any other theory of liability) for any loss or damage of any kind or nature related to, arising under or in connection with these materials, including for any direct, or any indirect, special, incidental, or consequential loss or damage (including loss of data, profits, goodwill, or any type of loss or damage suffered as a result of any action brought by a third party) even if such damage or loss was reasonably foreseeable or Xilinx had been advised of the possibility of the same.

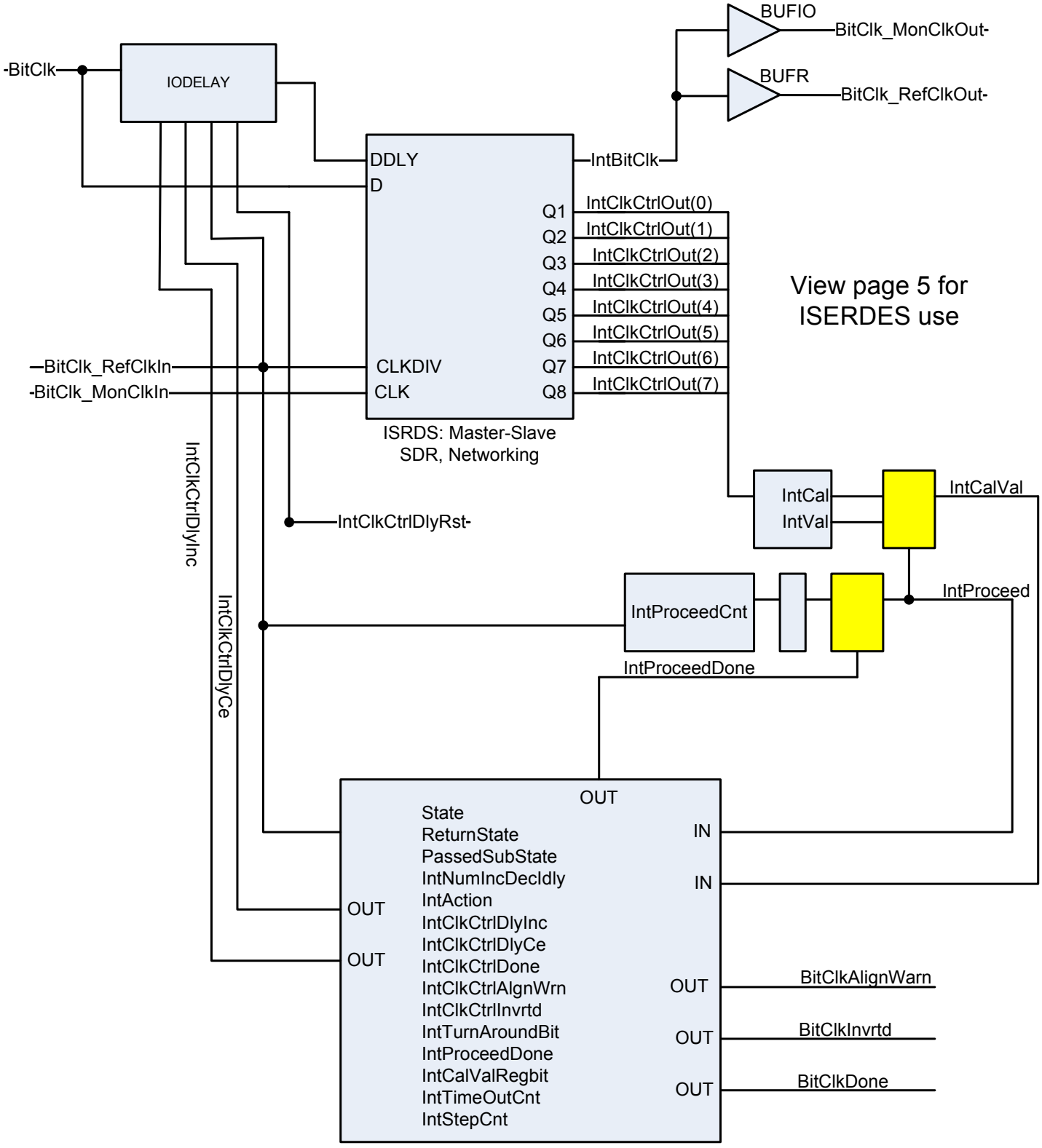
CRITICAL APPLICATIONS

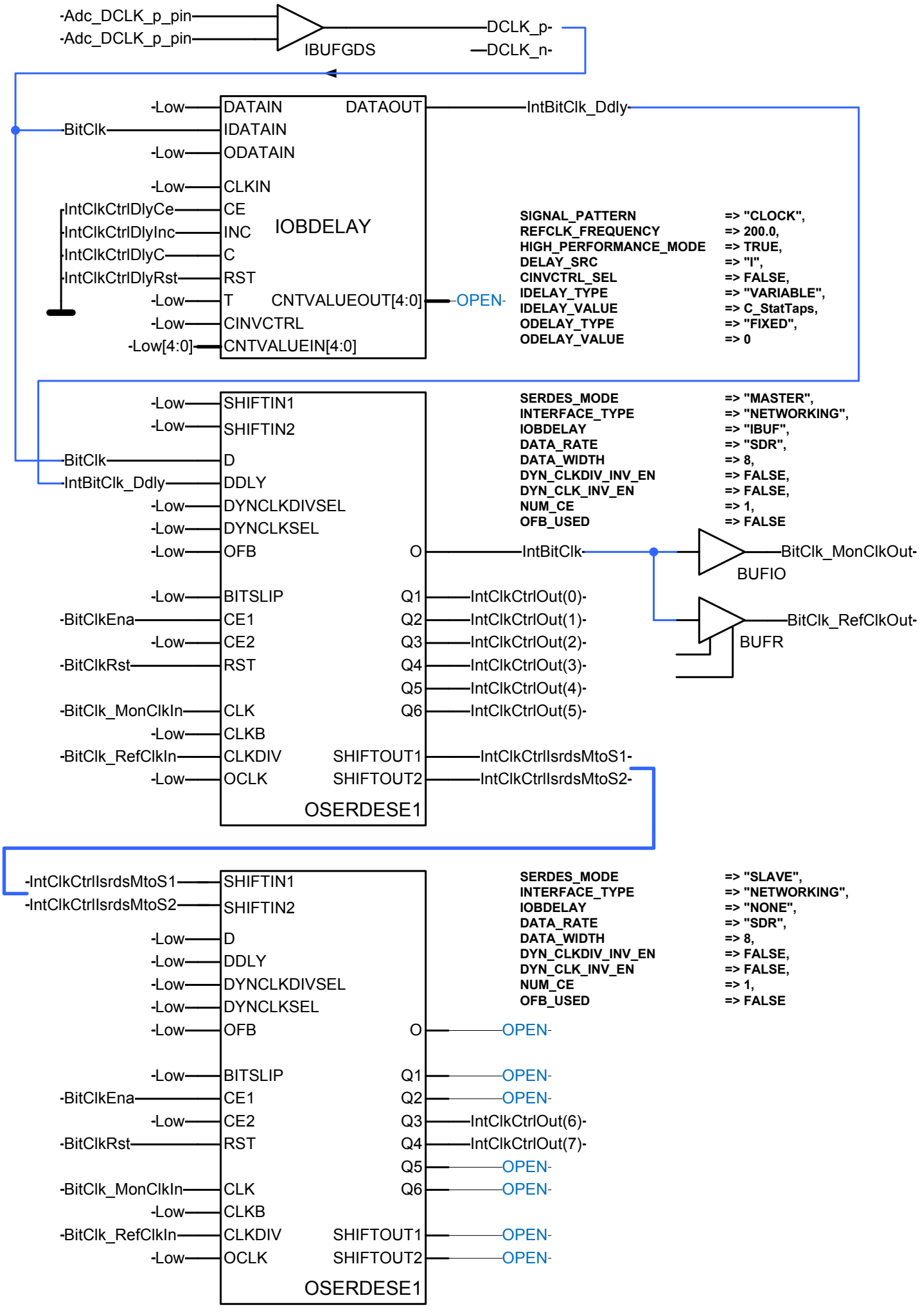
Xilinx products are not designed or intended to be fail-safe, or for use in any application requiring fail-safe performance, such as life-support or safety devices or systems, Class III medical devices, nuclear facilities, applications related to the deployment of airbags, or any other applications that could lead to death, personal injury, or severe property or environmental damage (individually and collectively, "Critical Applications"). Customer assumes the sole risk and liability of any use of Xilinx products in Critical Applications, subject only to applicable laws and regulations governing limitations on product liability.

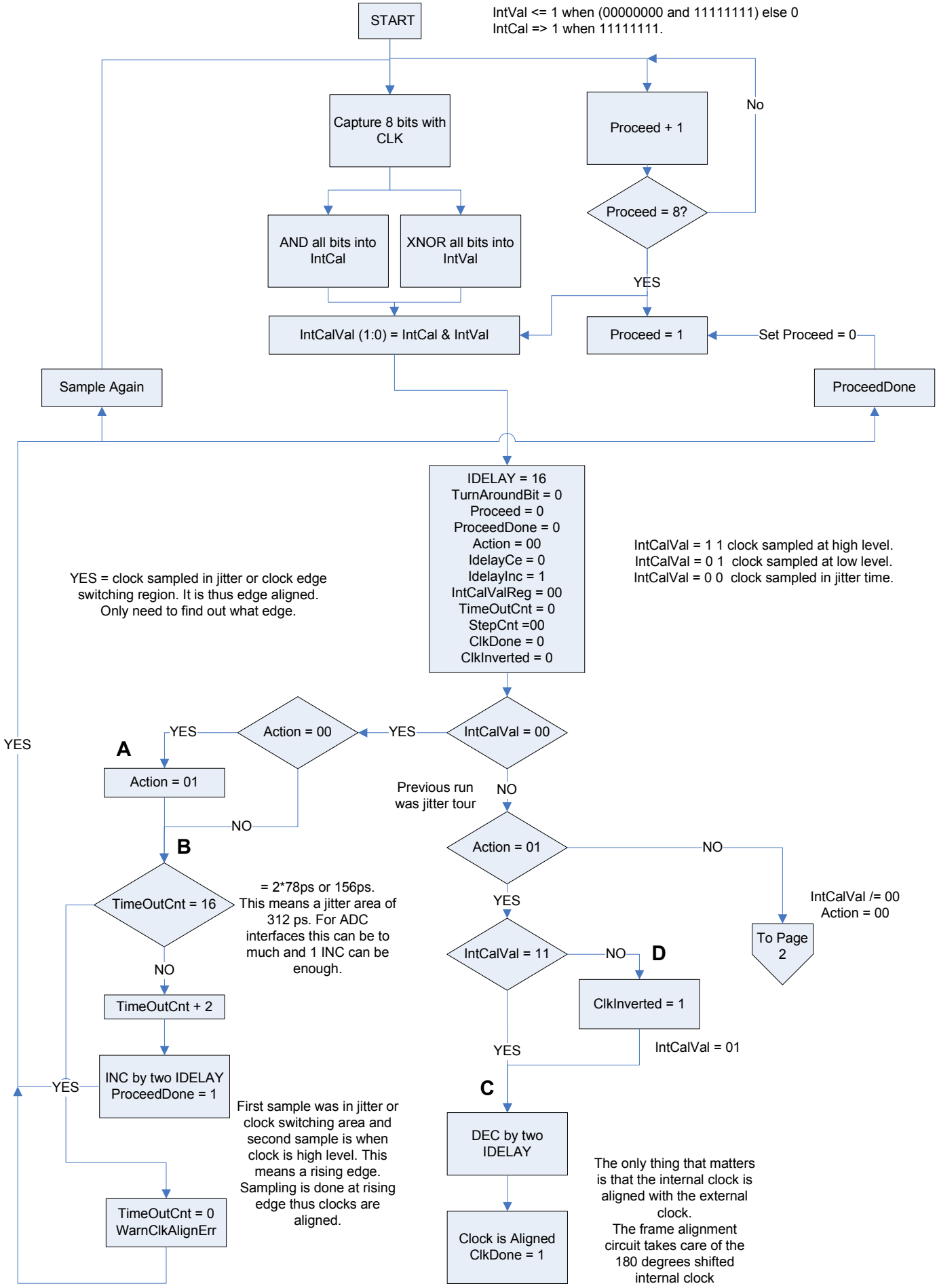
THIS COPYRIGHT NOTICE AND DISCLAIMER MUST BE RETAINED AS PART OF THIS FILE AT ALL TIMES.

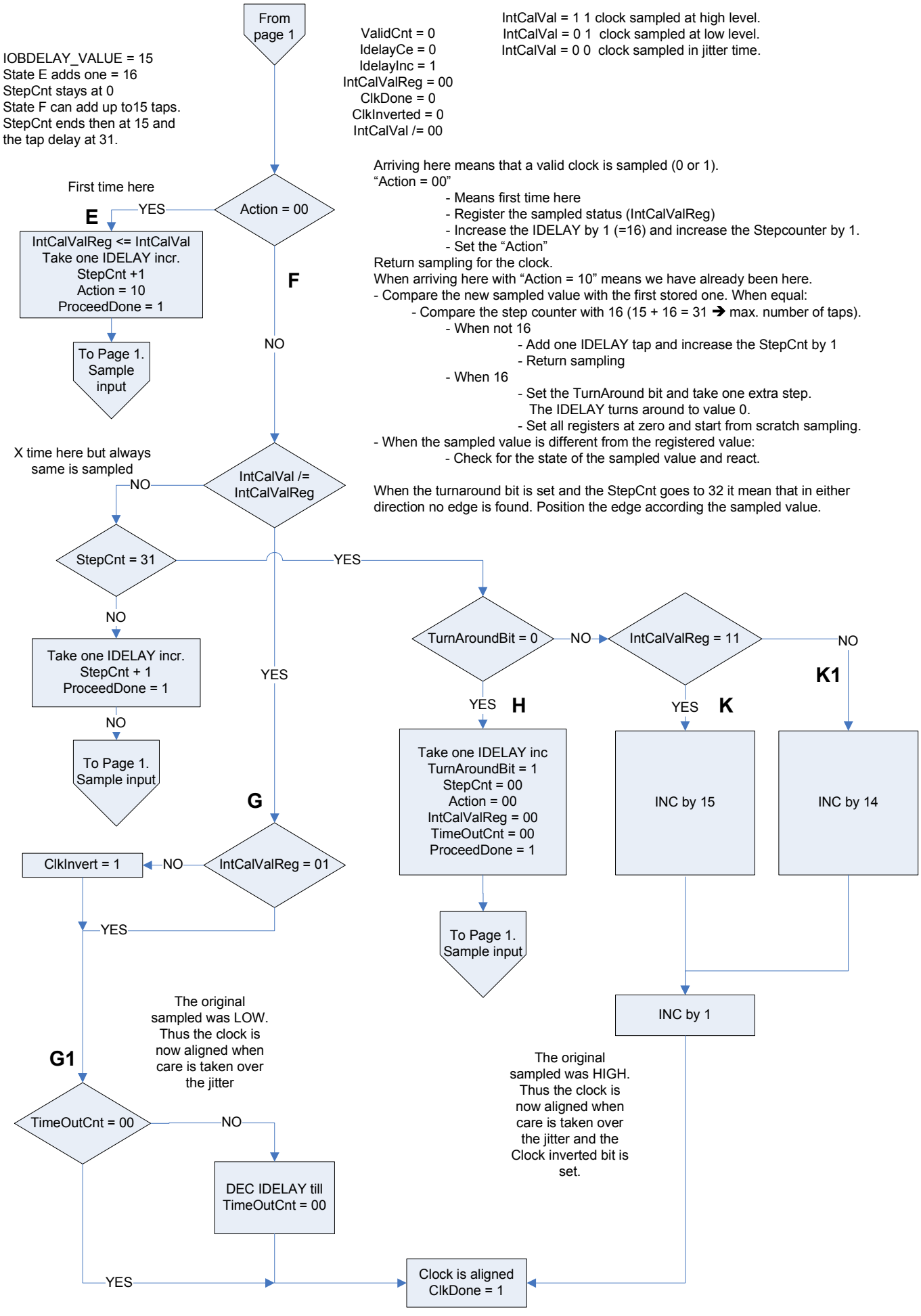
Contact: e-mail hotline@xilinx.com phone + 1 800 255 7778

This page is intentionally left blank.

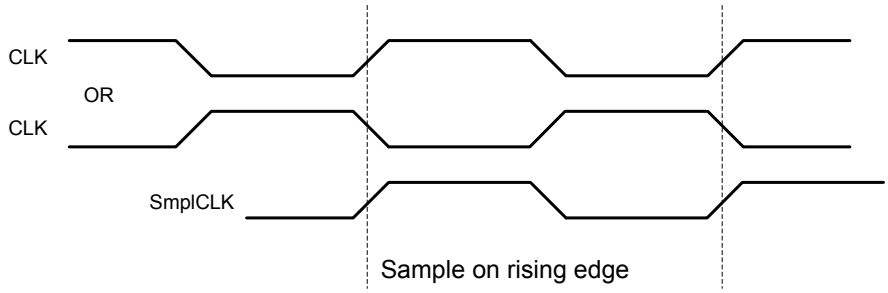

-BitClk
IODELAY
BUFIO
BitClk_MonClkOut-
BUFR
BitClk_RefClkOut-
DDLY
D
IntBitClk
Q1
Q2
Q3
Q4
Q5
Q6
Q7
Q8
IntClkCtrlOut(0)
IntClkCtrlOut(1)
IntClkCtrlOut(2)
IntClkCtrlOut(3)
IntClkCtrlOut(4)
IntClkCtrlOut(5)
IntClkCtrlOut(6)
IntClkCtrlOut(7)
View page 5 for ISERDES use
BitClk_RefClkIn
-BitClk_MonClkIn
CLKDIV
CLK
ISRDS: Master-Slave SDR, Networking
IntClkCtrlDlyInc
IntClkCtrlDlyRst-
IntCal
IntVal
IntCalVal
IntProceed
IntProceedCnt
IntProceedDone
IntClkCtrlDlyCe
OUT
State
ReturnState
PassedSubState
IntNumIncDecIdly
IntAction
IntClkCtrlDlyInc
IntClkCtrlDlyCe
IntClkCtrlDone
IntClkCtrlAlgnWrn
IntClkCtrlInvrtd
IntTurnAroundBit
IntProceedDone
IntCalValRegbit
IntTimeOutCnt
IntStepCnt
IN
IN
OUT
OUT
OUT
OUT
BitClkAlignWarn
BitClkInvrtd
BitClkDone


-Adc_DCLK_p_pin
-Adc_DCLK_p_pin
IBUFGDS
DCLK_p-
DCLK_n-
-Low
DATAIN
DATAOUT
IntBitClk_Ddly
BitClk
IDATAIN
-Low
ODATAIN
-Low
CLKIN
IntClkCtrlDlyCe
CE
IntClkCtrlDlyInc
INC
IOBDELAY
IntClkCtrlDlyC
C
IntClkCtrlDlyRst
RST
-Low
T
CNTVALUEOUT[4:0]
OPEN-
-Low
CINVCTRL
-Low[4:0]
CNTVALUEIN[4:0]
SIGNAL_PATTERN => "CLOCK",
REFCLK_FREQUENCY => 200.0,
HIGH_PERFORMANCE_MODE => TRUE,
DELAY_SRC => "I",
CINVCTRL_SEL => FALSE,
IDELAY_TYPE => "VARIABLE",
IDELAY_VALUE => C_StatTaps,
ODELAY_TYPE => "FIXED",
ODELAY_VALUE => 0
-Low
SHIFTIN1
-Low
SHIFTIN2
BitClk
D
IntBitClk_Ddly
DDLY
-Low
DYNCLKDIVSEL
-Low
DYNCLKSEL
-Low
OFB
O
IntBitClk
SERDES_MODE => "MASTER",
INTERFACE_TYPE => "NETWORKING",
IOBDELAY => "IBUF",
DATA_RATE => "SDR",
DATA_WIDTH => 8,
DYN_CLKDIV_INV_EN => FALSE,
DYN_CLK_INV_EN => FALSE,
NUM_CE => 1,
OFB_USED => FALSE
BUFIO
BitClk_MonClkOut-
BUFR
BitClk_RefClkOut-
-Low
BITSLIP
Q1
IntClkCtrlOut(0)-
-BitClkEna
CE1
Q2
IntClkCtrlOut(1)-
-Low
CE2
Q3
IntClkCtrlOut(2)-
-BitClkRst
RST
Q4
IntClkCtrlOut(3)-
Q5
IntClkCtrlOut(4)-
-BitClk_MonClkIn
CLK
Q6
IntClkCtrlOut(5)-
-Low
CLKB
-BitClk_RefClkIn
CLKDIV
SHIFTOUT1
IntClkCtrlIsrdsMtoS1-
-Low
OCLK
SHIFTOUT2
IntClkCtrlIsrdsMtoS2-
OSERDESE1
-IntClkCtrlIsrdsMtoS1
SHIFTIN1
-IntClkCtrlIsrdsMtoS2
SHIFTIN2
-Low
D
-Low
DDLY
-Low
DYNCLKDIVSEL
-Low
DYNCLKSEL
-Low
OFB
O
OPEN-
SERDES_MODE => "SLAVE",
INTERFACE_TYPE => "NETWORKING",
IOBDELAY => "NONE",
DATA_RATE => "SDR",
DATA_WIDTH => 8,
DYN_CLKDIV_INV_EN => FALSE,
DYN_CLK_INV_EN => FALSE,
NUM_CE => 1,
OFB_USED => FALSE
-Low
BITSLIP
Q1
OPEN-
-BitClkEna
CE1
Q2
OPEN-
-Low
CE2
Q3
IntClkCtrlOut(6)-
-BitClkRst
RST
Q4
IntClkCtrlOut(7)-
Q5
OPEN-
-BitClk_MonClkIn
CLK
Q6
OPEN-
-Low
CLKB
-BitClk_RefClkIn
CLKDIV
SHIFTOUT1
OPEN-
-Low
OCLK
SHIFTOUT2
OPEN-
OSERDESE1


START
IntVal <= 1 when (00000000 and 11111111) else 0
IntCal => 1 when 11111111.
Capture 8 bits with CLK
AND all bits into IntCal
XNOR all bits into IntVal
IntCalVal (1:0) = IntCal & IntVal
Proceed + 1
Proceed = 8?
No
YES
Proceed = 1
Set Proceed = 0
ProceedDone
Sample Again
IDELAY = 16
TurnAroundBit = 0
Proceed = 0
ProceedDone = 0
Action = 00
IdelayCe = 0
IdelayInc = 1
IntCalValReg = 00
TimeOutCnt = 0
StepCnt =00
ClkDone = 0
ClkInverted = 0
IntCalVal = 1 1 clock sampled at high level.
IntCalVal = 0 1 clock sampled at low level.
IntCalVal = 0 0 clock sampled in jitter time.
YES = clock sampled in jitter or clock edge switching region. It is thus edge aligned. Only need to find out what edge.
IntCalVal = 00
Action = 00
Action = 01
A
B
Previous run was jitter tour
NO
YES
TimeOutCnt = 16
= 2*78ps or 156ps. This means a jitter area of 312 ps. For ADC interfaces this can be to much and 1 INC can be enough.
TimeOutCnt + 2
INC by two IDELAY
ProceedDone = 1
TimeOutCnt = 0
WarnClkAlignErr
First sample was in jitter or clock switching area and second sample is when clock is high level. This means a rising edge. Sampling is done at rising edge thus clocks are aligned.
Action = 01
IntCalVal = 11
D
ClkInverted = 1
IntCalVal = 01
C
DEC by two IDELAY
Clock is Aligned
ClkDone = 1
The only thing that matters is that the internal clock is aligned with the external clock.
The frame alignment circuit takes care of the 180 degrees shifted internal clock
IntCalVal /= 00
Action = 00
To Page 2


From page 1
IOBDELAY_VALUE = 15
State E adds one = 16
StepCnt stays at 0
State F can add up to15 taps.
StepCnt ends then at 15 and the tap delay at 31.
ValidCnt = 0
IdelayCe = 0
IdelayInc = 1
IntCalValReg = 00
ClkDone = 0
ClkInverted = 0
IntCalVal /= 00
IntCalVal = 1 1 clock sampled at high level.
IntCalVal = 0 1 clock sampled at low level.
IntCalVal = 0 0 clock sampled in jitter time.
First time here
Action = 00
YES
E
IntCalValReg <= IntCalVal
Take one IDELAY incr.
StepCnt +1
Action = 10
ProceedDone = 1
To Page 1. Sample input
NO
F
Arriving here means that a valid clock is sampled (0 or 1).
"Action = 00"
- Means first time here
- Register the sampled status (IntCalValReg)
- Increase the IDELAY by 1 (=16) and increase the Stepcounter by 1.
- Set the "Action"
Return sampling for the clock.
When arriving here with "Action = 10" means we have already been here.
- Compare the new sampled value with the first stored one. When equal:
- Compare the step counter with 16 (15 + 16 = 31 ➔ max. number of taps).
- When not 16
- Add one IDELAY tap and increase the StepCnt by 1
- Return sampling
- When 16
- Set the TurnAround bit and take one extra step. The IDELAY turns around to value 0.
- Set all registers at zero and start from scratch sampling.
- When the sampled value is different from the registered value:
- Check for the state of the sampled value and react.
When the turnaround bit is set and the StepCnt goes to 32 it mean that in either direction no edge is found. Position the edge according the sampled value.
X time here but always same is sampled
IntCalVal /= IntCalValReg
NO
StepCnt = 31
YES
NO
Take one IDELAY incr.
StepCnt + 1
ProceedDone = 1
NO
To Page 1. Sample input
YES
TurnAroundBit = 0
NO
IntCalValReg = 11
NO
K1
YES
H
YES
K
Take one IDELAY inc
TurnAroundBit = 1
StepCnt = 00
Action = 00
IntCalValReg = 00
TimeOutCnt = 00
ProceedDone = 1
INC by 15
INC by 14
To Page 1. Sample input
G
IntCalValReg = 01
NO
ClkInvert = 1
YES
INC by 1
The original sampled was LOW. Thus the clock is now aligned when care is taken over the jitter
The original sampled was HIGH. Thus the clock is now aligned when care is taken over the jitter and the Clock inverted bit is set.
G1
TimeOutCnt = 00
NO
DEC IDELAY till TimeOutCnt = 00
YES
Clock is aligned
ClkDone = 1

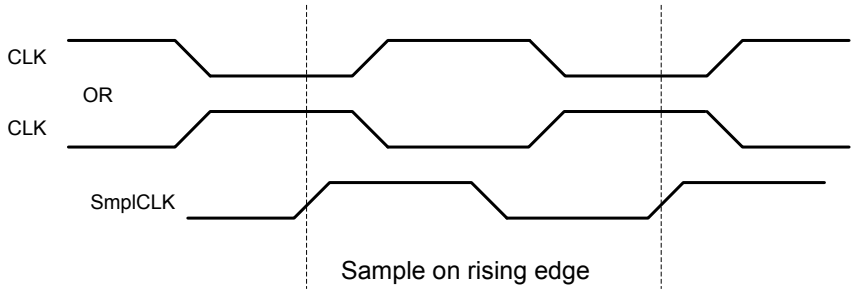
CASE 1



Sampling happens in jitter or cross over area.

1: Increase the IDELAY by 1 or 2 taps.
2: Detect the state of the sampled clock (High or Low)
3: When change of clock state set the ClkInvert bit
   Go to step 4.
   Else repeat steps 1 to 3
        (means lot of jitter or slow edge clock)
4: Step 1 or 2 IDELAY taps back.

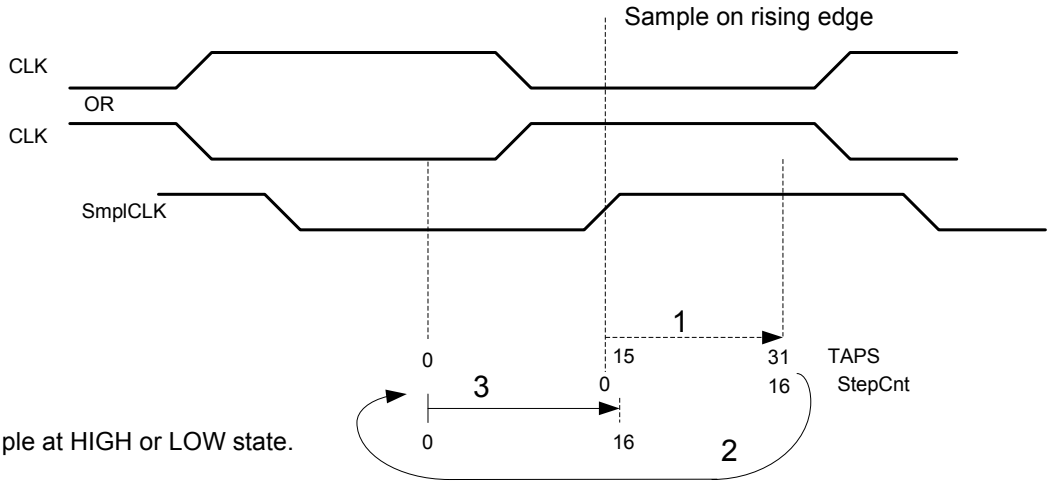
CASE 2



Sampling happens at HIGH or LOW state.

1: Save the first sampled state (IntCalValReg).
2: Increase the IDELAY by 1 tap and sample again.
3: When new and old sample are equal return to 2.
4: When new and original sample are different detect the state of the sampled clock (High or Low)
5: When LOW set the ClkInvert bit
7: Step 1 or 2 IDELAY taps back depending last sampling happened in the jitter or clock cross over area.

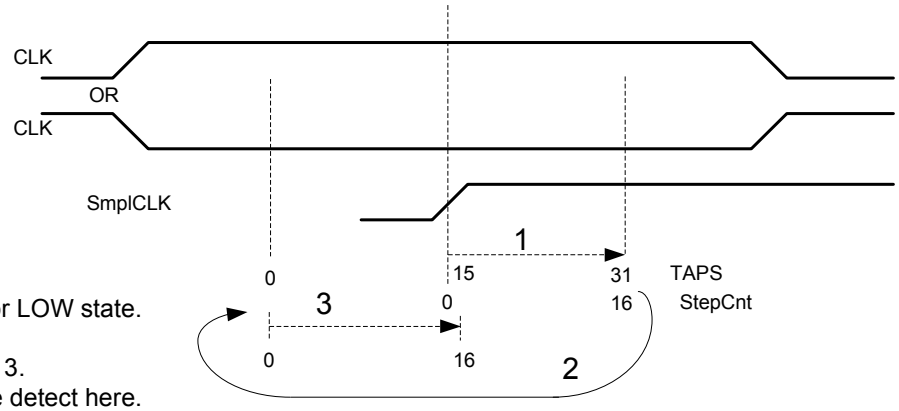
CASE 3



Sample at HIGH or LOW state.

1: Act as in CASE 2.
2: Increase with each time the TAP and a StepCnt.
3: When the StepCnt reach 16 this means that we are at the max value of the IDELAY (31) and no edge is seen register this event set all value to zero and take one extra step.
4: The IDELAY turns around and ends at 0.
5: Start from scratch searching for an edge.

CASE 4



Sample at HIGH or LOW state.

1: Act as in CASE 3.
2: No edge can be detect here.
3: Check the state of the first sample (High or Low)
4: Position the clock accordingly.
When High: position the clock to the left (TAP 0)
When Low: position the clock to the right (TAP 15)
Doing this alignes the clock as well as possible to the original incoming clock.