Vivado从此开始（To Learn Vivado From Here）

本书围绕Vivado四大主题

- 设计流程
- 时序约束
- 时序分析
- Tcl脚本的使用

作者：高亚军（Xilinx战略应用高级工程师）

- 2012年2月，出版《基于FPGA的数字信号处理（第1版）》
- 2012年9月，发布网络视频课程《Vivado入门与提高》
- 2015年7月，出版《基于FPGA的数字信号处理（第2版）》
- 2016年7月，发布网络视频课程《跟Xilinx SAE学HLS》

- 内容翔实全面：涵盖Vivado所有基本功能
- 讲解深入浅出：结合大量案例，帮助读者加强对基本概念的理解
- 描述图文并茂：给出具体操作步骤，易于快速动手实践

UltraFast Design
Clocking

Lauren Gao

Clocking

- Use MMCM or PLL Properly
- Create an Output Clock
- Clock Resource Selection Summary
- Source Synchronous Interface

Use MMCM or PLL Properly

Ug949 > Ch4 > Clocking > Controlling the Phase...

- **While using MMCM or PLL, pay attention to the following**
  - Do not leave any inputs floating
  - RST should be connected to the user logic
    - Grounding of RST can cause problems if the clock is interrupted
  - LOCKED output should be used in the implementation of reset
    - Synchronous logic clocked by the clock coming out of the PLL should be held in reset till LOCKED is asserted
    - The LOCKED signal would need to be synchronized before getting used in a synchronous portion of the design
  - The need for BUFG in the feedback path is important only if the PLL/MMCM output clock needs to be phase aligned with the input reference clock
  - Confirm the connectivity between CLKFBIN and CLKFBOUT

Safe Clock Startup and Sequencing

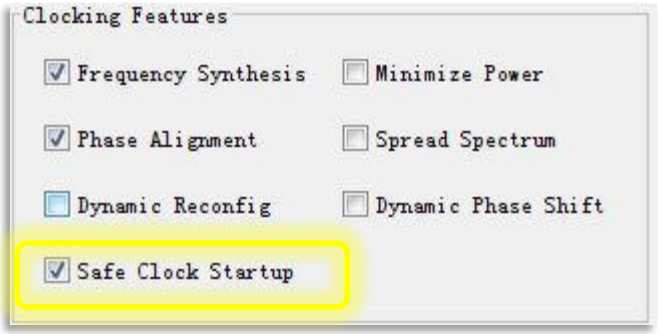
Pg065 > Ch4 > Customizing and Generating the Core

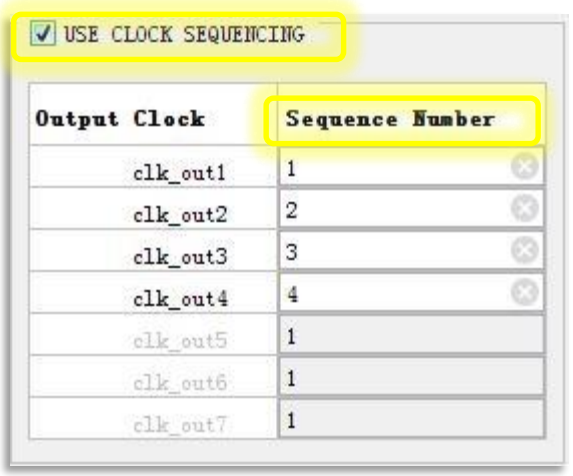
Safe Clock Startup

- Enable stable and valid clock at the output using BUFGCE after Locked is sampled High for 8 input clocks

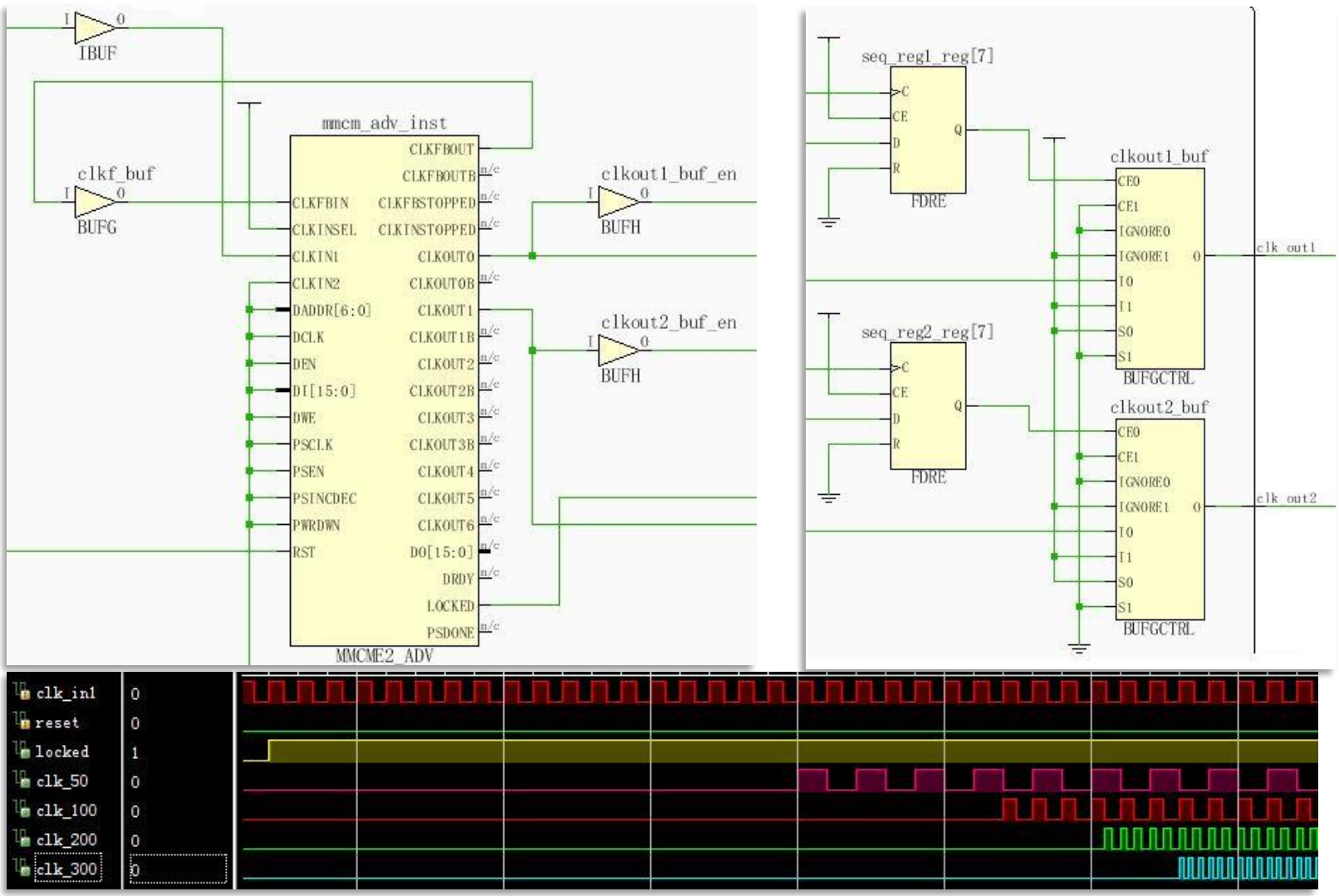
Sequencing

- Enable Clocks in a sequence according to the number entered through GUI
- Delay between two enabled output clocks in sequence is 8 cycle of second clock in the sequence clock
- It is useful for a system where modules need to be start operating one after the other





Safe Clock Startup and Sequencing Demo



Settings on MMCM or PLL

Ug949 > Ch4 > Clocking > Controlling the Phase...

- **Incorrect settings on the MMCM or PLL may**
  - Increase clock uncertainty due to increased jitter
  - Build incorrect phase relationships
  - Make timing more difficult

Name	Path 1
Slack	0.647ns
Source	cmd_parse_i0/send_resp_data_reg[10]/C
Destination	resp_gen_i0/to_bcd_i0/bcd_out_reg[5]/D
Path Group	clk_rx_clk_core
Path Type	Setup (Max at Slow Process Corner)
Requirement	10.000ns (clk_rx_clk_core rise@10.000ns -
Data Path Delay	9.326ns (logic 2.345ns (25.145%) route 6.
Logic Levels	18 (CARRY4=5 LUT3=3 LUT5=6 LUT6=4)
Clock Path Skew	-0.023ns
Clock Uncertainty	0.060ns

Clock uncertainty
in timing analysis

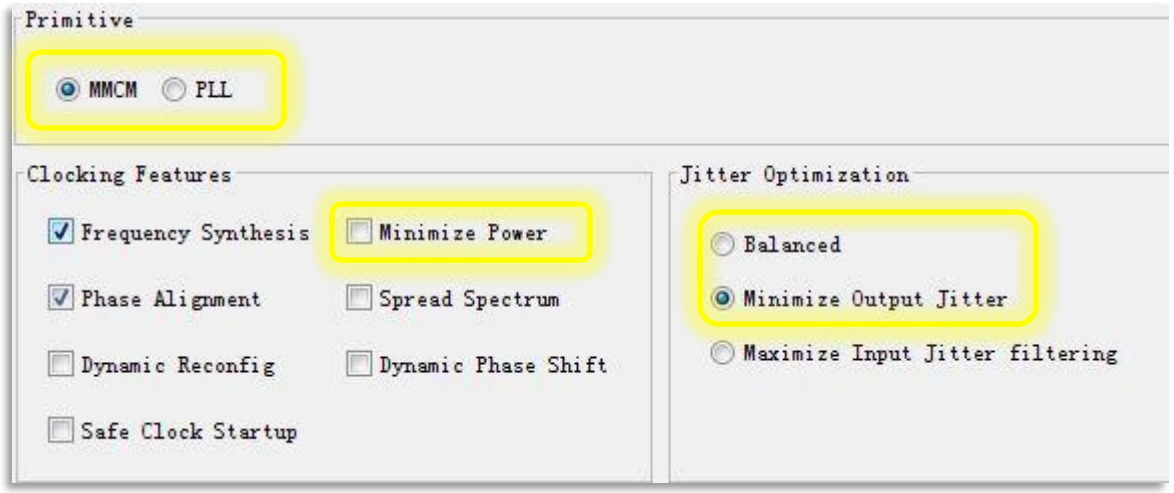
Destination Clock Path

Delay Type	Delay	Cumulative
(clock clk_rx_clk_core rise edge)	(r) 10.000	10.000
	(r) 0.000	10.000
net (fo=0)	0.000	10.000
IBUFDS (Prop ibufds I O)	(r) 0.803	10.803
net (fo=1, unplaced)	0.442	11.246
MMCME2_ADV (Prop mmcme2_adv CLKIN1 CLKOUT0)	(r) -4.291	6.955
net (fo=1, unplaced)	0.442	7.397
BUFG (Prop bufg I O)	(r) 0.083	7.480
net (fo=533, unplaced)	0.442	7.923
clock pessimism	-0.648	7.275
clock uncertainty	-0.060	7.215
FDRE (Setup fdre C D)	0.057	7.272
Required Time		7.272

MMCM/PLL Settings, Your Goals: Power or Jitter

Ug949 > Ch4 > Clocking > Controlling the Phase...

- **Depending on your goals,the settings in the Clocking Wizard may be changed to**
  - Further minimize jitter, and thus improve timing at the cost of higher power
  - Go the other way to reduce power but increase output jitter



- **MMCM**
  - **Balanced**
  - **Minimize Output Jitter**
- **PLL**
  - **Balanced**
  - **Minimize Output Jitter**

If you select 'Minimize Power', 'Minimize Output Jitter' is removed!

Jitter Comparison Between Different Settings

MMCM: Balanced

Output Clock

VCO Freq = 1020.000 MHz

Output Clock	Port Name	Output Freq (MHz)	Phase (degrees)	Duty Cycle (%)	Pk-to-Pk Jitter (ps)	Phase Error (ps)
clk_out1	clk_out1	340.000	0.000	50.0	200.090	300.046
clk_out2	clk_out2	85.000	0.000	50.0	242.325	300.046

MMCM: Minimize Output Jitter

Output Clock

VCO Freq = 1275.000 MHz

Output Clock	Port Name	Output Freq (MHz)	Phase (degrees)	Duty Cycle (%)	Pk-to-Pk Jitter (ps)	Phase Error (ps)
clk_out1	clk_out1	340.000	0.000	50.0	88.242	83.270
clk_out2	clk_out2	85.000	0.000	50.0	113.465	83.270

PLL: Balanced

Output Clock

VCO Freq = 1020.000 MHz

Output Clock	Port Name	Output Freq (MHz)	Phase (degrees)	Duty Cycle (%)	Pk-to-Pk Jitter (ps)	Phase Error (ps)
clk_out1	clk_out1	340.000	0.000	50.0	200.090	300.046
clk_out2	clk_out2	85.000	0.000	50.0	242.325	300.046

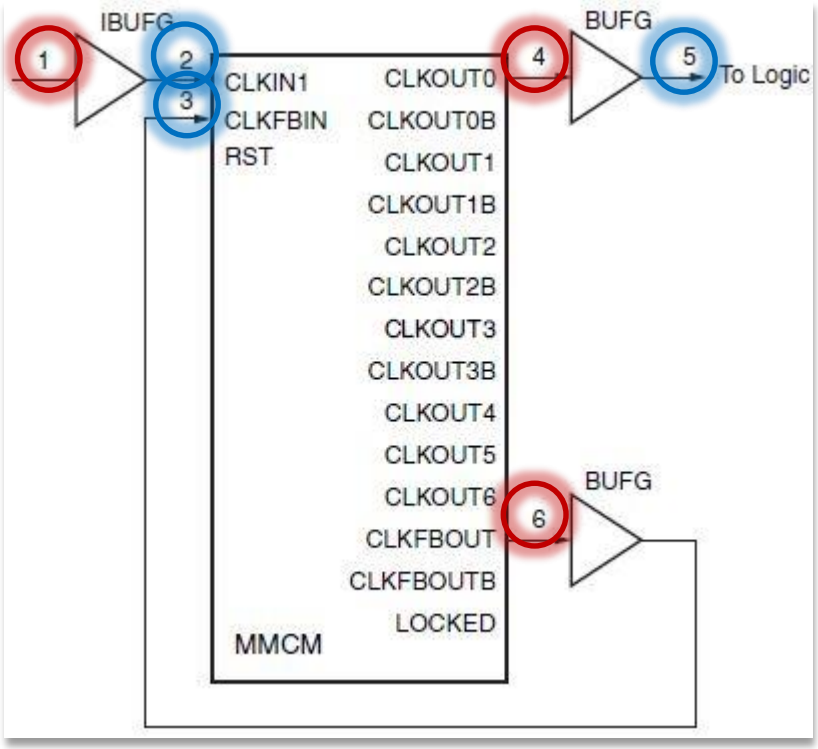
PLL: Minimize Output Jitter

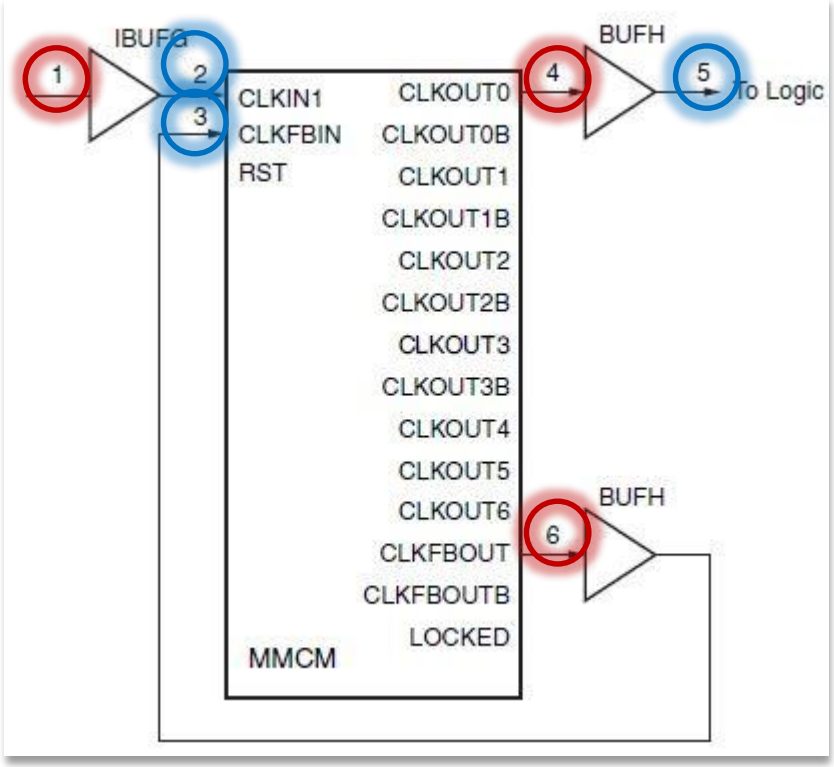
Output Clock

VCO Freq = 1700.000 MHz

Output Clock	Port Name	Output Freq (MHz)	Phase (degrees)	Duty Cycle (%)	Pk-to-Pk Jitter (ps)	Phase Error (ps)
clk_out1	clk_out1	340.000	0.000	50.0	69.055	82.376
clk_out2	clk_out2	85.000	0.000	50.0	87.746	82.376

Phase Between Output Clock





1 4 6 — Same phase

2 3 5 — Same phase

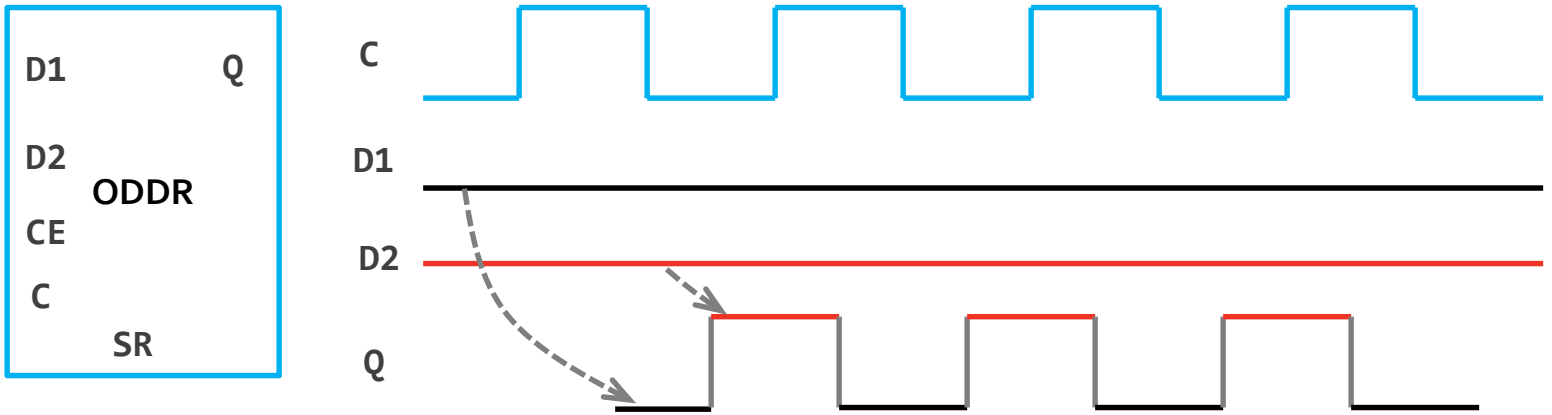
1 4 6

2 3 5

Creating an Output Clock

Ug949 > Ch4 > Clocking > Creating an Output Clock

- **An effective way: ODDR can forward a copy of the clock to the output**
- **This is useful for propagating a clock and DDR data with identical delays**
  - Tying the D1 input of the ODDR primitive High, and the D2 input Low



Clock Resource Selection Summary 1

Ug949 > Ch4 > Clocking > Clock Resource Selection Summary

- **BUFG**
  - Use when a high-fanout clock must be provided to several clock regions throughout the device
  - Use for very high fanout non-clock nets such as a global reset
- **BUFGCE**
  - Use to stop a large-fanout several-region clock domain
- **BUFGMUX/BUFGCTRL**
  - Use to change clock frequencies or clock sources during the operation of your design

Clock Resource Selection Summary 2

Ug949 > Ch4 > Clocking > Clock Resource Selection Summary

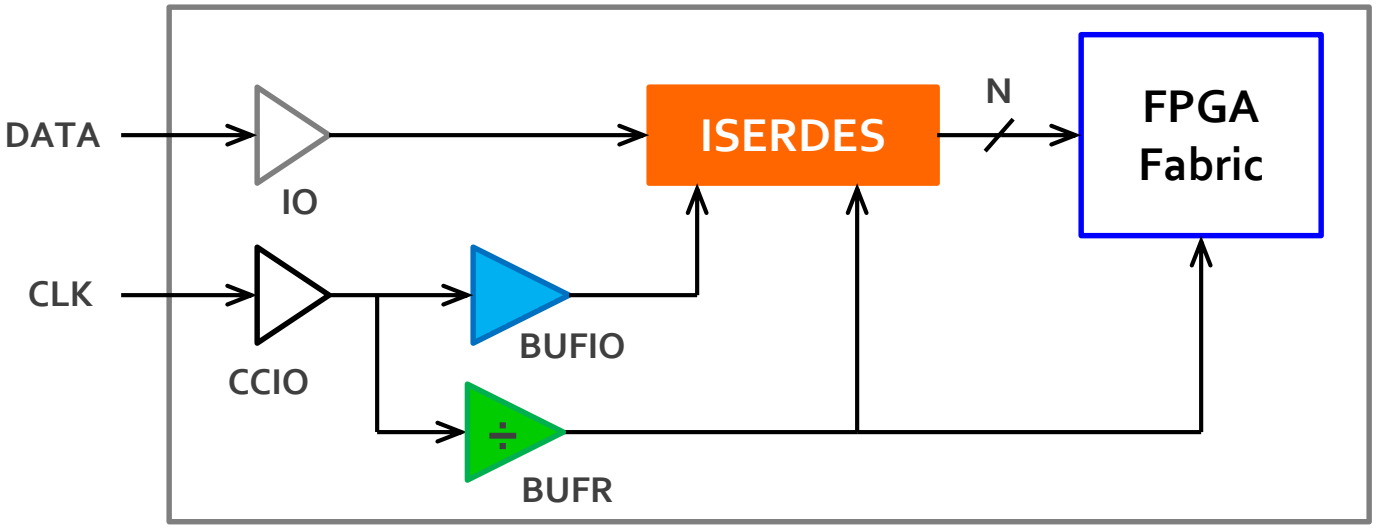
- **BUFH**
  - Use for smaller clock domains of logic that can be contained within a single clock region
- **BUFR**
  - Use for small to medium sized clock networks that do not require performance higher than 450 MHz
- **BUFIO**
  - Use for externally provided high-speed I/O clocking generally in source synchronous data capture
- **BUFMR**
  - Use when you need to use BUFRs or BUFIOs in more than one vertically adjacent clock regions for a single clock source

Clock Resource Selection Summary 3

Ug949 > Ch4 > Clocking > Clock Resource Selection Summary

- **PLL and MMCM**
  - PLL provides a better control of jitter
  - MMCM can provide a wider range of output frequencies.
  - For tighter timing requirement, PLLs might be best, provided they can provide the frequency of interest
- **IDELAY / IODELAY**
  - Use on an input clock to add small amounts of additional phase offset (delay)
  - Use on input data to add additional delay to data thus effectively reducing clock phase offset in relation to the data
- **ODDR**
  - Use to create an external forwarded clock from the device

Source-Synchronous Interface

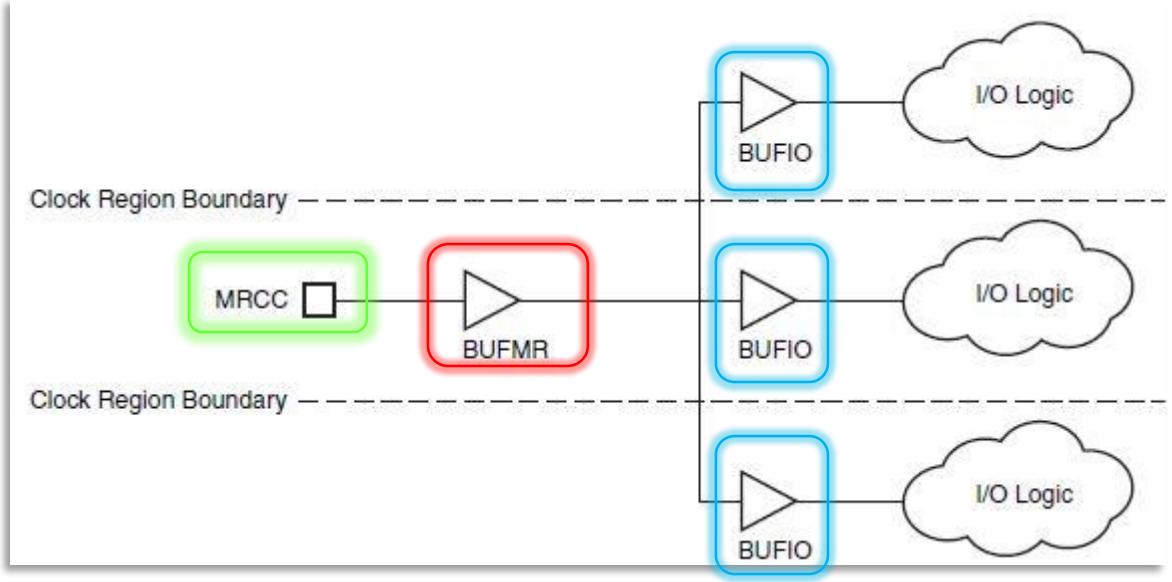


Symbol	Description	Speed Grade					Units
		1.0V				0.9V	
		-3	-2/-2L	-1	-1M	-2L	
T_{BIOCKO_O}	Clock to out delay from I to O	1.04	1.14	1.32	1.32	1.48	ns
Maximum Frequency							
F_{MAX_BUFIO}	I/O clock tree (BUFIO)	800.00	800.00	710.00	710.00	710.00	MHz

Symbol	Description	Speed Grade					Units
		1.0V				0.9V	
		-3	-2/-2L	-1	-1M	-2L	
Maximum Frequency							
F_{MAX_BUFR} (1)	Regional clock tree (BUFR)	600.00	540.00	450.00	450.00	450.00	MHz

Driving Multiple BUFIOs

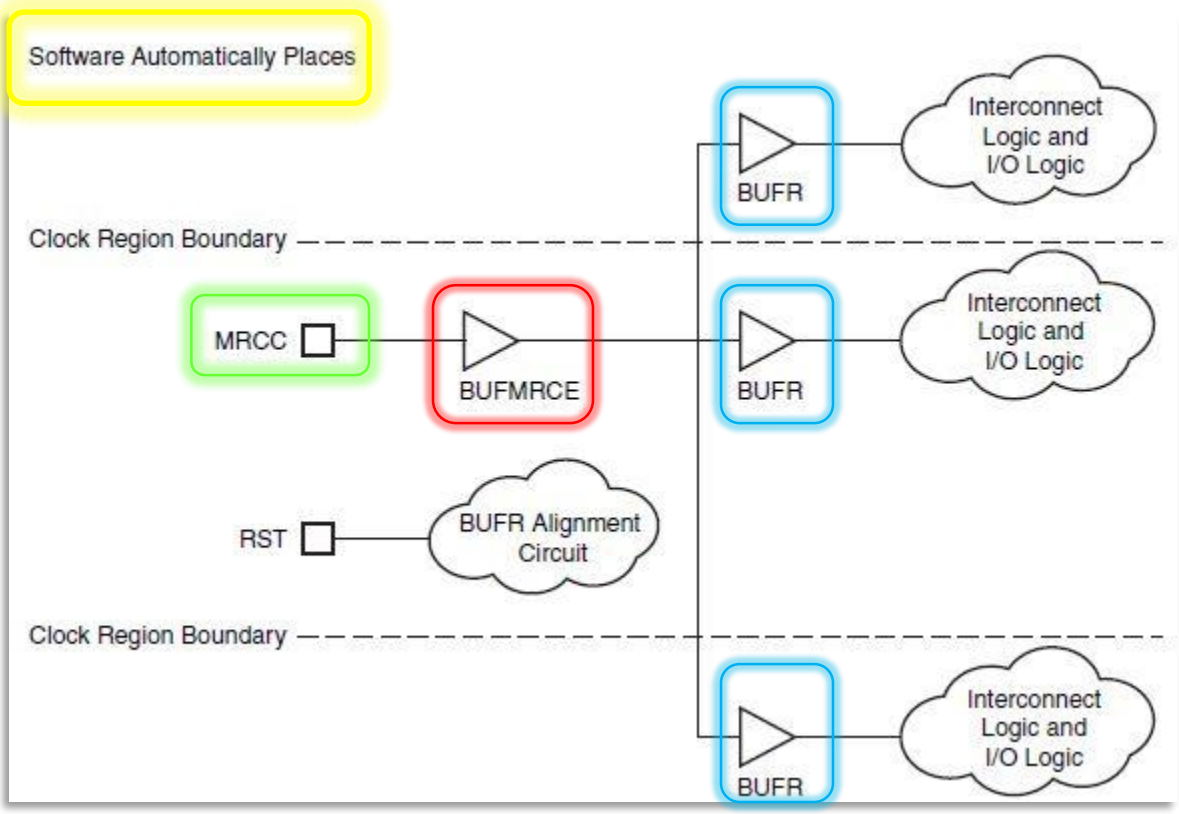
Ug107 > Appx. A: Multi-Region Clocking



- Although BUFRs can perform this function, BUFIOs supply the highest performance operation and drive dedicated clock nets within the I/O column
- The placer software automatically places the buffers in the appropriate location

Driving Multiple BUFRs

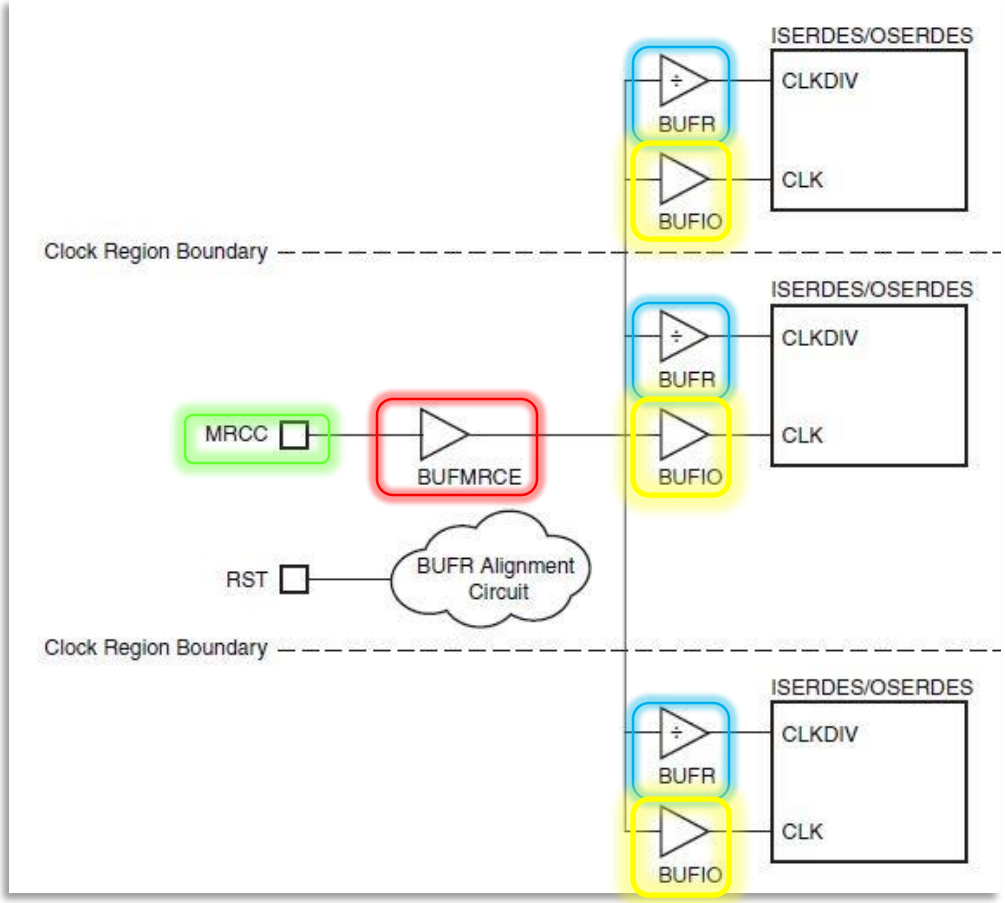
Ug107 > Appx. A: Multi-Region Clocking



- If the divide value in the BUFR is being used, then all BUFR instances must be reset while the BUFMRCE is disabled
- The placer software automatically places the buffers in the appropriate location

Driving Multiple BUFRs (with Divide) and BUFIO

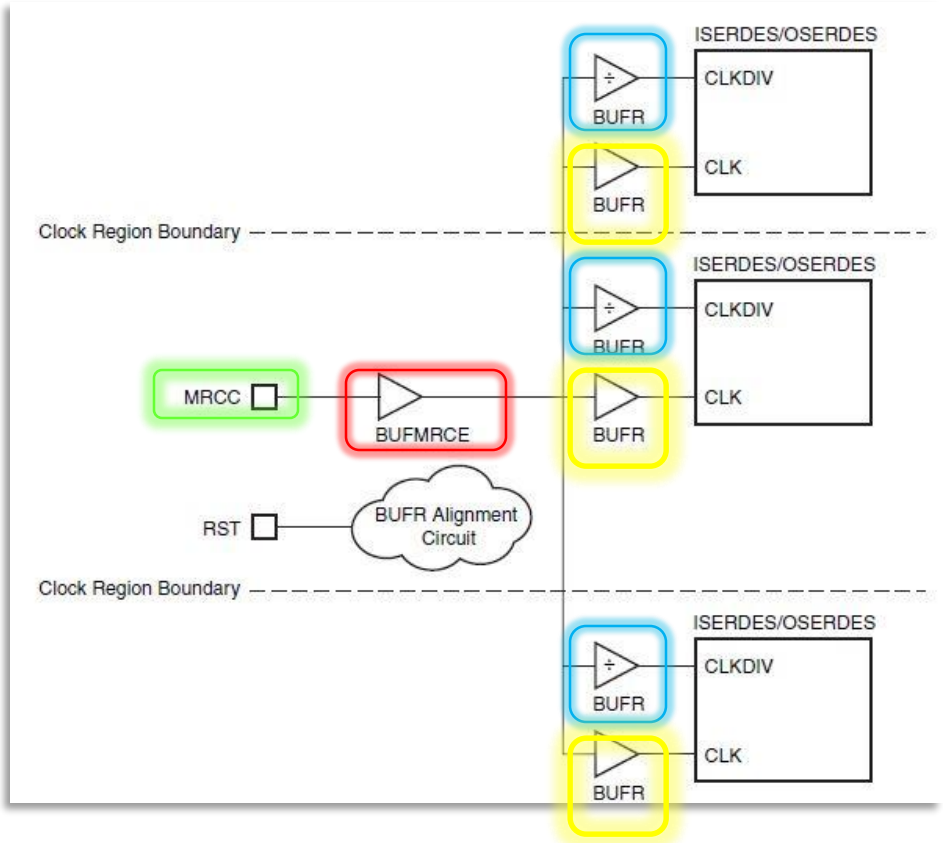
Ug107 > Appx. A: Multi-Region Clocking



- Manually place the buffers with a LOC Constraint
- The logic driven by the buffers is automatically placed in the appropriate location

Driving Multiple BUFRs (With and Without Divide)

Ug107 > Appx. A: Multi-Region Clocking



- Manually place the buffers with a LOC Constraint
- The logic driven by the buffers is automatically placed in the appropriate location

Synchronizing BUFRs Driven by a BUFMR

Ug107 > Appx. A: Multi-Region Clocking

- In order to clock a single interface that spans multiple banks, a BUFMR must be used to drive the BUFIO and BUFR in the different regions
- The dividers on each BUFR are independent; they must be synchronized in order to ensure proper operation of the interface
  - Use a BUFMRCE to disable the clock feeding the BUFRs
  - Assert the CLR on all the BUFRs
    - This resets the dividers in the BUFRs
  - Deassert the CLR on all the BUFRs
    - This allows the dividers to start on the next rising edge the input clock (currently gated)
  - Assert the CE on the BUFMR
    - Starts the clocks to all BUFRs
    - BUFRs are now in sync