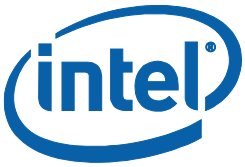


Clock Domain Controller

Integration Guide

Dec 2015

Revision 1.21

Hartej Singh and Yong J. Kim

Intel Restricted Secret

Contents

Figures

Tables

Revision History

Author	Revision Number	Description	Revision Date
J. Wilcox	0.1	Initial release.	12/22/12
J. Wilcox	0.1.1	Modified behavior for IP-Inaccessible entry to ensure clkreq/clkack handshake completes in all cases before POK de-assertion. No interface changes are required for this change.	1/4/13
J. Wilcox	0.2	Incorporated doc clarifications and fixes based on review feedback. Renamed pgcb_force_reset_b to pgcb_force_rst_b to m,atch PGCB. The transition to OFF from FORCE_READY now requires POK to have de-asserted. Added a ON_PENDING and SYNCON state to ensure that clkack has been asserted moving to ON. Several other small changes to clean up FSM corner cases. Updated the P2SB reference code to show config latching.	1/12/13
J. Wilcox	0.2.1	Added a CGATE_PENDING state to explicitly enforce 8 clock delay from gclock ack/active de-assertion to gclock stopping. A few changes to the FSM specifically around exiting RESTORE to handle both clkack properly and cases where all operations complete by the end of the restore phase. Clarified that pok reset value is determined by the DEF_PWRON parameter. Noted that it is permitted to OR together ISM values when multiple IOSF interfaces share a single CDC. Clarified the timing of pwrgate_force de-assertion and removed the recommendation for using greset_b to clear the condition since that is not safe as it is de-asserted asynchronously and the pwrgate_force would be using the ungated clock. Removed the P2SB example.	2/5/13
J. Havican	0.80	Changing Version to 0.80 to match PGCB [HSD: 1274582] CDC now locks ISMs/Boundary in CGATE_PENDING or CGATE if do_force_pgate asserts in these states. • do_force_pgate will stay asserted as long as ism_locked is asserted. • FSM will not return to ON state from CGATE_PENDING if do_force_pgate is asserted. [HSD: 1274578] Added per-reset SCAN overrides, also updated dt_* related signals to Chassis standard names. [HSD: 1274316] Added DFx force enable for clkreq. [HSD: 1274682] FSM resets to ON_PENDING if DEF_PWRON is set. [HSD: 1274683] FSM now resets on pgcb_rst_b instead of reset_b[0]. [HSD: 1274685] CDC now deasserts powergateready and eventually proceeds to ON state if power-gating is disabled while power-gated. [HSD: 1274684] Enforces priority of pmc_<ip>_pg_wake over the ForcePwrgatePOK message (trigger for IP-Inaccessible power-gating). [HSD: 1274581] Added internal signals to cdc_visa output for debug visiblity.	3/11/13

		[HSD: 1274580] Clock now ungates when the next state is ON or SYNCONISM (instead of ungating when present state is ON/SYNCONISM) to reduce latency by 1 clock cycle. [HSD: 1274583] Idle timer now remaines at 0 when expired, instead of rolling over. Updated RTL signal name pgcb_reset_b to match this spec, pgcb_rst_b.	
		Release of Version 1.0 with following changes: 1. Addition of new parameters, ISM_AGT_IS_NS, and RSTR_B4_FORCE 2. Updates to descriptions (clarifications and requirements) for several interface signals including configuration signals 3. Updated notes on ISM Locking (pre-flop locking is now required, and post-flop locking is not allowed) 4. Updated version of CDC state machine diagram, state descriptions and state transitions table. 5. Added new section on CDC behavior details to discuss several important aspects of CDC behavior or exceptions to regular behavior. 6. Enabled metastability and pulse width checks for most of the double-syncs used within the CDC (and defined mechanism for users of the CDC to disable the checks individually for each double-sync). Implemented the following HSDs: [HSD: 1276565] [ASSERTIONS] CDC - Enable Doublesync Checks [HSD: 1276305] [RTL] [FPV] CDC will skip RESTORE if in IP-Accessible PG when ForcePwrgatePOK is received [HSD: 1276306] [RTL] [FPV] CDC can report pwrgate_ready after RESTORE before the main clock is ready [HSD: 1276307] [RTL] [FPV] CDC unlock_ism deassertion will always cross to Main clock domain before unlock_all assertion [HSD: 1276308] [RTL] [FPV] CDC will not assert clkreq during restore [HSD: 1276309] [RTL] [FPV] CDC can enter ON state with clkreq and clkack low [HSD: 1276310] [RTL] [FPV] CDC clock should ungate in CGATE if force_pgate_req [HSD: 1276311] [RTL] [FPV] CDC breaks gclock_req/ack in RESTORE [HSD: 1276548] [RTL] [FPV] CDC Race between unlock_all and force_pgate_req synchronizers [HSD: 1276562] [RTL] CDC Deadlock Possible Going from IP-Accessible to IP-Inaccessible [HSD: 1276564] [RTL] CDC reset value of POK will be 0 regardless of DEF_PWRON [HSD: 1275250] [RTL] 0.8 CDC assertion error [HSD: 1276028] [RTL] CDC to keep clocks ungated during IP-Inaccessible Entry [HSD: 1275787] [ASSERTIONS] Power Gating Assertions needed in CDC/PGCB [HSD: 1275943] [RTL] CDC Force reset mux should use ctech cell for SCAN bypass [HSD: 1275944] [RTL] CDC Pre-Flop ISM Locked Update (Parameterized) [HSD: 1275023] [RTL] CDC should implement DFx ISM clockgate override [HSD: 1275024] [WAIVER] CDC/PGCB lintra waivers for Rev0.8	

		[HSD: 1274883] [RTL] CDC Need To Solidify Config Register Clock-Crossing Recommendation [HSD: 1275328] [RTL] CDC is dropping one clock cycle after reset [HSD: 1274997] [ASSERTION] PGCB/CDC assertions missing on I/Os [HSD: 1275343] [DOC] CDC documentation to call out requirement that clocks be available on IP-Inaccessible exit [HSD: 1275588] [DOC] CDC Documentation Update for clock gate disable	
Hartej Singh	1.1	Release of Version 1.1 with following changes: 1. 2 new input DFx signals (fscan_clkgenctrl*) and other bypass logic added to support usage of the CDC in pre-SCC mode. 1 new output signal (gclock_enable_final) added to support SCC related considerations (refer to Appendix for more information). Added new parameter, PRESCC, for this change. 2. Support for separating out functional clock (on short clock tree) controlled by clock_gate within CDC from functional clock used by other logic within CDC (SD requirement) – for details, refer to section on CDC Behavior Details. Addition of new parameters related to this change - DSYNC_CG_EN, FLOP_CG_EN and CG_LOCK_ISM. 3. Support for Restore of IP blocks that are powered-ON by default (Chassis PG ECN 1570775). 4. Several documentation updates (see change bars) Implemented the following HSDs: **Key changes:** [HSD: 1277154] CDC PV violation with clock-gate combi logic [HSD: 1277155] CDC needs parameter to allow it to be used preSCC [HSD: 1277156] [BXT] PGCB/CDC need ability to default to restore state Corner-case bugs (some are related to above changes): [HSD: 2243684] CDC should treat ism_agent as gclock_req_sync [HSD: 2243686] CDC boundary_locked does not default to locked [HSD: 2243688] CDC could hang in FORCE_READY [HSD: 2243801] CDC to add parameter to support assertion of ISM_LOCK with de-assertion of gclock_active **Assertion fixes:** [HSD: 2243601] [ASSERTION] CDC a_ism_agent_1 is looking at sync version of gclock_req_async [HSD: 1276710] [ASSERTION] CDC Clkgate holdoff assertion fires falsely for IP-Inaccessible Entry [HSD: 1276742] [ASSERTION] CDC a_reset_b_1 assertion needs to be disabled for DEF_PWRON	
Hartej Singh	1.15	Release of Version 1.1 with fix for following issue: [HSD: 2243998] IOSF primary compliance issue seen with CDC v1.1 version (ISMPM 033)	
Jared Havican	1.20	**RTL Changes:** [2281759] Requirement - Provide .sig files for VISA insertion	

		• With this release of the PGCB blocks, we are providing .sig files to be used by the IP to insert VISA within the blocks using the VISA insertion tool. **The *_visa vector outputs are still available but it is recommended that IPs let them dangle and instead use the provided .sig files to enable easier silicon debug.** [2267263] Requirement - Updated to SIP CTECH methodology – requires integrating IPs to include global CTECH libraries in their ACE environment [2244996] Requirement - fism_dfx_clkgate_ovrd now ungates gclock instead of gate (Chassis ECN: 1570764) [2247919] Enhancement - Added SCAN Reset Bypass muxes on reset_b and pok_reset_b inputs to CDC [2392454] Enhancement - fismdfx_force_clkreq now keeps PGD fully awake and unlocked (instead of just asserting clkreq) [2280804] Enhancement – clkreq output is now driven only from pgcb_clk domain (rather than mix between pgcb/func clk) [1275945] Enhancement – Added prescc_clock input for CDCs that are used pre-SCC [2266757] Enhancement – CDC FSM now always waits for clkack to assert before moving out of a parked state [2245023] Enhancement – CDC only looks at clkgate_disabled in ON state to prevent potential hang scenario [2267264] Defect - Moved u_gclockEnAckSync to long clock tree branch [2280824] Defect – CDC always moves from RESTORE to ON to prevent potential hang scenario returning to PGATE [2244989] Defect – ISM will now unlock in SYNCON_ISM when CG_LOCK_ISM is set [2244791] Defect – gclock_active now deassert 8 clocks before gclock is gated **Doc/Assertion/Environment Changes:** [2243993] Documented that it is acceptable for the CDC be in the process of gating clocks while sleep toggles on IP-Inaccessible PG entry [2249492] Called out need to waive Caliber violations regarding logic on reset due to DFx and force_rst_b [2246461] Updated a_pgcb_pwrgate_active_2 assertion [2245424] Assertion updates [2244732] Clarified ITBITS min values is 3 [2245761] Updated to state greset_b is synchronized to clock instead of gclock [2245509] Clarified gclock_active behavior in spec [2392384] Replaced `ifdef ASSERT_ON with `ifndef SVA_OFF so assertions are enabled by default [2266993] Removed ASSERT_ON in HDL files [2246425] Updated CDC Waivers	
Yong J. Kim	1.21	**RTL Changes:** ❑ [1404136840] Defect - CG_LOCK_ISM=1 and ISM_AGT_IS_NS=0: CDC component should assert ism_lock 1 clock earlier to lock agent ISM. ❑ [1503996414] Enhancement – Logic enhancement to avoid	

		potential timing issue with PowerGateReady signal caused by meta-stability during IP-Accessible entry. ❑ [1504001217] Enhancement – Logic enhancement to prevent spurious unlock_all assertion during IP-Inaccessible entry due to meta-stability issue ❑ PGCB CTECH map file update – Meta-Stability Required Parameter and define changes. ❑ [1206405592] Xprop tool compliant coding update. No functional change. ❑ [1207621082] Enhancement - Potential race/deadlock condition due to unlock_ism signal (cdc_restore_pg) assertion could cause issue in CDC S/M **Doc/Assertion/Environment Changes:** ❑ [1504089630] aClkreq hold assertions in CdcMainClock fail if pgcb_reset_b ❑ [1404074824] Assertion improvements for reset awareness: ❑ Tool version upgrades ❑ Updated Lintra Waivers ❑ Updated CDC Waivers

§

1 Architectural Overview

1.1 Overview

The Clock Domain Controller (CDC) is a component that works in conjunction with the Power Gate Control Block (PGCB) to create a power gating solution for IPs that minimizes custom glue logic. Use of the CDC ensure that entry and exit will be done in a consistent and safe manner by all Ips and complex operations like reset and clock management and IOSF ISM management are handled properly.

The PGCB supplies the sequencer for actual power gate entry, and the CDC is responsible for managing a single clock domain within a power gated domain (PGD). Ips are responsible for integrating a PGCB and a CDC for each clock domain in the PGD and handling the minimal glue logic required to create a custom solution. The requirements for this integration and the necessary glue logic are described in this document.

1.2 Roles Performed

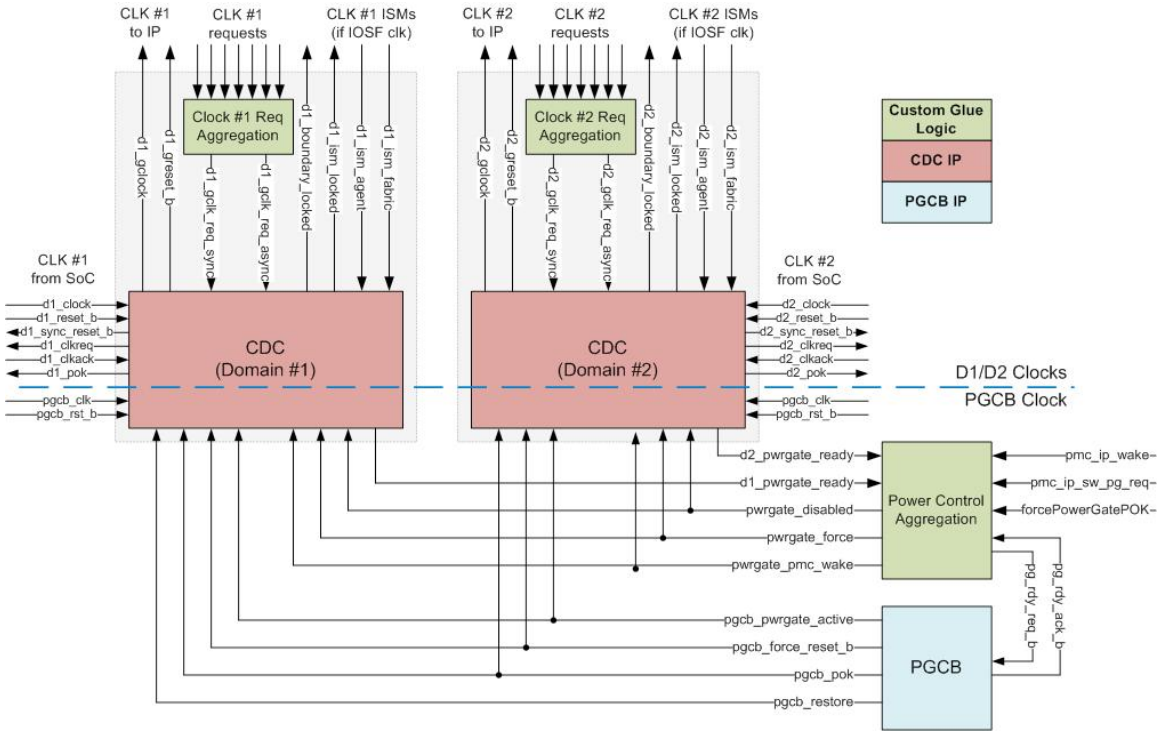
- Provides race & glitch-free control of locking ISMs and controlling clocks during power gating entry and exit.
- One or more CDCs Interface with the PGCB to handle most of the power gate entry/exit management.
- Chassis compliant clkreq/clkack control for clock sources.
- Supports Chassis compliant clkreq/clkack handshakes for both synchronous and asynchronous clock consumers for its domain.
- IOSF 1.1 compliant control of the POK signals.
- Provides synchronized resets,
    - One version is controlled (masked) for supporting the requirements of flops in the power gated domain.
    - Another version that is not masked during power gating but is otherwise coincident with the other for use with any Always On logic.
- Provides local clock gating with configurable idle timers.
- Provide hysteresis timers for managing the entry into clock

Note: for every clock that is controlled through the CDC, designers need to be careful about using the clock in AON/Ungated domain. This is because any time this clock is needed, and the IP is in PG state, it causes a wake event. It is recommended that this clock not be used in the AON/Ungated domain, or else, this clock be used in AON/Ungated domain only when the IP is not in PG state.

1.3 Block Diagram

The following shows how the PGCB, the glue logic and two CDCs would be assembled in an IP that had two clock domains.

Figure 1-1: High Level Connections for 2CDCs



2 CDC Interface

2.1 Parameters

Name	Default	Valid Values	Description
DEF_PWRON	1	0,1	**Default Power On**: Determines whether the initial state after reset_b de-assertion is powered on or powered off. Should be set to match the value of the same parameter in the PGCB. If this is set to one clkreq and pok will be asserted and gclock will be ungated while reset is asserted (gclock_active will be deasserted until the clock is confirmed active by clkack assertion). When set to zero, clkreq, pok and gclock_active will be de-asserted and gclock would be gated during reset. This parameter is SOC-specific (may need to change depending on SOC where SIP is being used).
ITBITS	16	3-16	**Idle Timer Bits:** Determines the size of the idle timer counter. Designs with short max timeout values can reduce gates be reducing the size of their idle timer counter. This parameter is SIP-specific (does not depend on SOC where SIP is being used).
RST	1	≥1	**Reset Count**: Determines the number of reset signals that will be managed for this domain. This parameter is SIP-specific (does not depend on SOC where SIP is being used).
AREQ	1	≥1	**Asynchronous Clock Requests:** Determines the number of asynchronous clock request inputs. Providing dedicated inputs for different asynchronous sources ensures glitch free aggregation of different requests but will require per-request synchronization, increasing design area/cost. This parameter is SIP-specific (does not depend on SOC where SIP is being used).

DRIVE_POK	1	0,1	**Drive Power OK Signal**: This should be set to one for any usage that requires a POK signal, including all IOSF clock domains. For CDCs where this parameter is set to 0, the "pok" output of the CDC always remains '0'. This parameter is SIP-specific (does not depend on SOC where SIP is being used).
ISM_AGT_IS_NS	0	0, 1	**Agent ISM Is Next_State:** If this is set to '1', the *_locked signals are driven as the output of a flop. However, if this is '0' (default value), then the locked signals assert combinatorial manner (in the same cycle that the CDC logic determines that conditions to lock the ISM are satisfied). The de-assertion of the *_locked signals always happens through the output of a flop. This parameter is SIP-specific (does not depend on SOC where SIP is being used).
RSTR_B4_FORCE	0	0, 1	**Restore Before Force:** When this is set to '1', then the CDC ignores any pwrgate_force assertion that happens while pgcb_restore signal is asserted (i.e. Restore phase is in progress). This implies that the CDC requires the restore phase to complete first in order to be able to enter IP-Inaccessible PG (subsequent to the IP-Accessible PG exit). Else, when this parameter is set to '0', the CDC terminates the Restore phase (by masking the pgcb_restore signal internally) if the pwrgate_force signal is asserted during the Restore phase. This parameter is SOC-specific (may need to change depending on SOC where SIP is being used).
PRESCC	0	0, 1	**Pre-SCC Use of CDC:** When set to '1', the logic supports usage of the CDC before the scan clock controller (SCC) instance for that specific clock. For this case, the CDC includes clkgenctrl muxes at the clock-gate enable. These muxes enable the scan logic to control the clock gate during various SCAN modes. These muxes are required for any clock-gate on a clock that is implemented before the clock is passed through a SCAN clock controller (SCC). For the typical cases of SIPs, CDCs are used after the instantiation of the Scan Clock Controller (SCC), hence the default value of this parameter is '0'. NOTE: Refer to "Appendix" section on "Scan

			considerations for CDC Usage" for more details on when this parameter needs to be set to 1. NOTE: Whenever this parameter needs to be set to '1', the instance name of the CDC component is required to have the tag of "prescc". NOTE: The values for the parameters of FLOP_CG_EN and DSYNC_CG_EN are ignored when PRESCC=='1' (hardware behavior in these cases corresponds to the case that FLOP_CG_EN == '1' and DSYNC_CG_EN == '1'). This parameter is SOC-specific (may need to change depending on SOC where SIP is being used).
DSYNC_CG_EN	0	**0, 1**	**Double-sync crossing for Clock Gate Enable:** When set to '1', the clock gate enable as driven from the functional clock domain is considered to be asynchronous to the clock input of the clock gate, therefore the clock gate enable is synchronized using a double-sync component (with separate reset for this component). In many cases, since the CDC clock gate is at the top level of the clock buffer tree for the functional clock, it is not possible to meet setup timing from the leaf-level node of the clock to the clock-gate enable signal. In these cases, the parameter needs to be set to '1'. NOTE: Each SIP needs to work closely with Structural Design (SD) team to understand if this parameter needs to be set to '1' for any of the CDCs used in that SIP. NOTE: Refer to section on "CDC Behavior Details" for more information about this parameter. NOTE: The value for the parameter FLOP_CG_EN is ignored when DSYNC_CG_EN == '1' (hardware behavior in these cases corresponds to the case that FLOP_CG_EN == '1'). This parameter is SOC-specific (may need to change depending on SOC where SIP is being used).
FLOP_CG_EN	1	**0, 1**	**Flopped Clock-gate Enable:** When set to '1' (default value), the enable signal to the clock-gate in the CDC is driven solely by the output of a flop within the CDC. When set to '0', there is a combinatorial path to the

			enable of the clock gate (to allow for faster ungating). However, this combinatorial path may present timing issues because the clock-gate within the CDC is much higher up in the clock tree than the CDC FSM and/or agent fabric interface logic within the SIP. CDC users need to get approval from the SD team before setting this parameter to '0'. When this parameter is set to '0', there is no latency in ungating the clock (assuming the SIP is not power-gated) when any IP wake is detected in the functional clock domain. This parameter does not change the latency associated with wakes detection through the pgcb_clk domain logic of the CDC. This parameter is both SIP and SOC-specific (may need to change depending on SOC where SIP is being used).
CG_LOCK_ISM	0	0, 1	**Clock-gated Lock ISM:** When set to '1', the "ism_locked" signal also asserts whenever the "gclock_active" signal is de-asserted (in addition to asserting for the cases given below when value of this parameter is set to '0'). When set to '0' (default value), the "ism_locked" signal asserts only under following conditions: • For IP-Accessible, when the CDC enters the PGATE_PENDING state • For IP-Inaccessible (or warm reset), when CDC state machine observes assertion of "force_pgate_req" and agent ISM is in idle state. This parameter is intended to be set to '1' in the following case: • The SIP needs to set the parameter DSYNC_CG_EN == '1' (or PRESCC == '1') for a CDC with DRIVE_POK == '1', AND • the fabric interface logic uses the "gclock_active" indication to qualify any activity, but the ISM logic does not use the "gclock_active" qualifier. It is expected that most IOSF primary interfaces in the SIPs have this behavior (ISM activity is not qualified based on "gclock_active"). This parameter is SIP-specific (does not depend on SOC where SIP is being used).

2.2 Macros (`defines)

Name	Description
SVA_OFF	If defined, disables the embedded assertions within the PGCB.
DC	If defined, disables the embedded assertions within the PGCB. Intended to be set during synthesis
SVA_FORMAL	If defined, enables certain assertions/assumptions that are required for Formal Property Validation (FPV). Should not be set during simulation.

2.3 Interface Signals

Table 2-1: PGCB Interface Signals

Name	I/O	Clock	Description
pgcb_clk	In	-	Clock used for the PGCB
pgcb_rst_b	In	pgcb_clk	Reset for the PGCB. This also resets the CDC state machine. De-assertion must be synchronized to the pgcb_clk; assertion may be done asynchronously. This reset must be exposed by the IP to the SOC as per Chassis Reset Arch HAS. Refer to that HAS for details on different options to drive this reset.
pgcb_force_rst_b	In	pgcb_clk	Force greset_b signals to assert. Should be driven directly by the PGCB's signal of the same name.
pgcb_pok	In	pgcb_clk	Power OK indicator. Should be driven directly by the PGCB's signal of the same name.
pgcb_restore	In	pgcb_clk	Restore operation in progress during power gate exit. Should be driven directly by the PGCB's signal of the same name.
pgcb_pwrgate_active	In	pgcb_clk	Power gating active. When set high this indicates that power gating has been started, power is gated or power gating exit has not completed. This indicates to the CDC that it may not unlock. Should be driven directly by the PGCB's signal of the same name.

Table 2-2: Master Clock Interface Signals

Name	I/O	Clock	Description
clock	In	-	Master clock for the domain controlled by this CDC. If using PRESCC==1, this should be the post-SCC clock.
prescc_clock	In	-	If using PRESCC==1, this should be the pre-SCC clock. It is used to drive the final clock-gate for gclock. If PRESCC==0, this should be tied to 0.
reset_b[RST-1:0]	In	-	Asynchronous resets to be used for the clock domain controlled by this CDC. These may be async resets as their de-assertions will be synchronized by the CDC. The CDC inserts SCAN bypass muxes on the input to the CDC and after the reset synchronizer for each of these inputs.
reset_sync_b[RST-1:0]	Out	clock	Synchronized de-assert version of reset_b. *Important*: This should only be used for logic in the Always On domain. Must not be used for any logic that whose power gating is controlled by this CDC.
clkreq	Out	pgcb	Clock request for 'clock'. Is driven completely from the pgcb clock domain and is glitch free. Adheres to Chassis clkreq/clkack protocol. This will be asserted during reset if DEF_PWRON is true. Otherwise it will reset to a de-asserted state. NOTE: When this signal is asserted, the SOC/IP block (As applicable) is required to respond with an active clock on the "clock" input. Otherwise, undefined behavior (including but not limited to prevention of any future PG entry) may result.
clkack	In	*async*	Clock acknowledge for 'clock'. May be asynchronous but must be glitch free. Must adhere to Chassis clkreq/clkack protocol.
pok	out	clock	Power OK indicator compliant with the IOSF 1.1 specification. If the DRIVE_POK parameter is 0, this signal will be a constant 0.

			RESET Value = 'b0 (This is de-asserted during reset (pok_reset_b) regardless of the value of DEF_PWRON). NOTE: It is required that Early Boot devices not use this output signal from the CDC but instead follow the recommendation of the Chassis Reset Arch HAS to tie the *_pok signal output from the IP to the fabric to a value of "1".
pok_reset_b	In	*async*	Reset for the POK signal. Recommendation from Chassis Reset Arch HAS is to tie this to the deepest side_rst_b signal received by the IP block. If an IP has the CDC parameter DRIVE_POK set to 0, then this input signal may be left unconnected (or tied off to 0/1). The CDC inserts SCAN bypass muxes on the input to the CDC and after the reset synchronizer for this input.

Table 2-3: Gated Clock Interface Signals

Name	I/O	Clock	Description
gclock	out	-	Gated version of 'clock'. Should be used as the root clock for this domain in this power gated domain.
greset_b[RST-1:0]	out	clock*	Gated version of reset_b with de-assertion synchronized to clock. *During power gating flows this may be asynchronously asserted and de-asserted, but gclock is guaranteed to not be running at that time.
gclock_req_sync	In	gclock	Glitch free synchronous clock request. Generally used for indication within the IP that the domain needs to remain active, but will also be used to wake the domain. Once gclock_req_sync has been asserted it must not de-assert until gclock_active has been asserted. *However* unlock normal clkreq/clkack handshakes, it does not (and should not) wait for gclock_active to de-assert before re-asserting again.
gclock_req_async[AREQ-1:0]	In	*async*	Asynchronous glitch free clock requests. Each will be individually synchronized locally before used. Should

			use a unique gclock_req_async bit for each source domain that is asynchronously controlling this domain. Once gclock_req_sync has been asserted it must not de-assert until the corresponding gclock_ack_async has been asserted. Though not a functional requirement of the CDC, if a chassis compliant clkack is required for this request, it should not be re-asserted until to the corresponding gclock_ack_async bit has been de-asserted.
gclock_ack_async[AREQ-1:0]	Out	clock	Acknowledge for each gclock_req_async indicating that the gclock_req_async bit has been synchronized and sampled asserted and the clock is active. Can be used for constructing a Chassis compliant clkreq/clkack handshake with an asynchronous clock requester.
gclock_active	Out	clock	Indicates that gclock is active. This signal always deasserts when leaving the ON state. Asserts one cycle after the clock enable for gclock. De-asserts at least 8 clocks before the gclock's enable. Can be used both for assessing the state of the domain and for constructing Chassis compliant clkreq/clkack handshakes with the sources of gclock_req_sync.
gclock_enable_final	Out	clock*	This is the actual signal that drives the clock-gate in the CDC. For SIPs where a functional clock is used to derive various other clocks (through M/N dividers or muxes), this signal may be used to gate other clocks that are derived off the functional gclock. * For cases where the parameters DSYNC_CG_EN =='1' or PRESCC == '1', this signal is the output of a double-sync on the actual clock that is gated by the clock-gate (treated as asynchronous to the clock-treed version of the functional 'clock'). See "Appendix" on various options on avoiding cascaded SCCs (Scan clock controllers) for more information on use of this signal.
ism_fabric	In	clock	IOSF 1.1 Fabric ISM. Used to detect wake conditions from the fabric. Should be connected directly to the input ism_fabric signal for IOSF clock domains. For non-IOSF clock domains this must be tied to 3'd0.

			Note: In cases where multiple IOSF interfaces share the same CDC, their ISMs may be OR'd to create this input. The CDC will only check for zero vs non-zero values on this signal. Also, the CDC assumes that when the ism_fabric changes, then the fabric clock is available for use by the IP. Therefore, under some conditions, the CDC proceeds to an active state and ungates the fabric clock to the PGD even when clkreq to the SOC has not yet been asserted. It is expected that eventually the gclock_req-[sync/async] from the IP will assert and cause the clkreq to the SOC to be asserted. Note that for IOSF-like interfaces (ex PMI), that do not ensure the clock to the endpoint is running when the fabric state changes, this input should be tied to '0 and a gclock_req_* input should be provided to wake the clock instead.
ism_agent	In	gclock	IOSF 1.1 Agent ISM. Used to detect busy conditions from the IOSF agent to ensure a forced power gate entry is done in a way that is compliant with IOSF requirements for de-asserting clkreq and POK. Should be connected to the ism_agent signal (see description of the requirements for this later) for IOSF clock domains. For non-IOSF clock domains this must be tied to 3'd0. Note: In cases where multiple IOSF interfaces share the same CDC, their ISMs may be OR'd to create this input. The CDC will only check for zero vs non-zero values on this signal. Also, the agent_ism moving out of IDLE is not comprehended directly by the CDC as a reason to ungate clocks (or keep clocks ungated). The IP is required to precede this with assertion of gclockreq_[sync/async] to the CDC.
ism_locked	Out	gclock	Indicates that the ISMs for this domain should be locked. See description later in this doc for requirements on how to properly lock ISMs.
boundary_locked	Out	gclock	Indicates that all non-IOSF sources that can access state in this clock domain should be explicitly or implicitly blocked. This signal stays asserted during the restore phase of a power gate exit (if doing restore) to ensure that no agent other than IOSF may

			access state until this restore operation completes.

Table 2-4: Configuration Signals

Name	I/O	Clock	Description
cfg_clkgate_disabled	In	*Async*	Disables any clock gating due to idle conditions. Has no affect on clock gating that is functionally required for safe power gating entry and exit. This control has no effect on power-gating entry or clkreq de-assertion. In other words, this control only disables clock gating until such time as the hysteresis timers for power gating (if enabled) or clock req de-assertion (if enabled) are still not expired. Once any of the above timers expire and corresponding feature (power-gating or clock req de-assertion without power-gating) is enabled, clock gating is then applied independent of this control configuration signals. The CDC assumes this signal is driven asynchronously and therefore synchronizes this signal before it is used.
cfg_clkreq_ctl_disabled	In	*Async*	When set high, this prevents the CDC from de-asserting 'clkreq' due to idle conditions. Has no effect on clkreq de-assertion during a forced power gate entry flow. It is recommended to control this via a configuration register. *Important*: This value must stay valid during power gating, so the value should be latched if configurable and the register is in the PGD. The CDC assumes this signal is driven asynchronously and therefore synchronizes this signal before it is used.
cfg_clkgate_holdoff[3:0]	In	*Static**	Minimum time from the CDC detects all clock reqs de-asserting until gclock is gated. The actual count is 2^value. For example setting this register to 4'd5 would result in a hold-off time of 32 clock cycles. The signal cfg_clkgate_disabled is the corresponding disable signal for this timer value.

cfg_pwrgate_holdoff[3:0]	In	*Static**	Time from gating gclock until the CDC will start the lock entry and allow power gating. Used only when power gating is enabled (pwrgate_disabled=0). The actual count is 2^value. The signal pwrgate_disabled is the corresponding disable signal for this timer value.
cfg_clkreq_off_holdoff[3:0]	In	*Static**	Time from locking the domain and preparing for power gating until the CDC will de-assert clkreq. Used only when power gating is enabled (pwrgate_disabled=0) and cfg_clkreq_ctl_disabled=0. The actual count is 2^value. The signal cfg_clkreq_ctl_disabled is the corresponding disable signal for this timer value. *Important*: This value must stay valid during power gating, so the value should be latched into the AON/ungated domain (if configurable and the register is in the PGD) through an isolation latch or otherwise.
cfg_clkreq_syncoff_holdoff[3:0]	In	*Static**	Time from gating gclock until the CDC will de-assert clkreq (with synchronous ISM wake detection). Used only when power gating is disabled (pwrgate_disabled=1) and cfg_clkreq_ctl_disabled=0. The actual count is 2^value. The signal cfg_clkreq_ctl_disabled is the corresponding disable signal for this timer value.

2.3.1 Note*: Please refer to the sub-section on Detailed gclock_active behavior

The following describes the conditions under which gclock_active can assert/deassert:

- Assertion:
  - Asserted in Reset when DEF_PWRON==1
  - Asserts when entering RESTORE state and clkack==0
  - Asserts when entering ON state
  - Asserts when entering FORCE_PENDING
  - Can assert in CGATE*, PGATE* states if pwrgate_force is asserted (see section 4.1.4)
- Deassertion
  - Deasserted in Reset when DEF_PWRON==0
  - Deasserts in CGATE_PENDING
  - Deasserts in OFF_PENDING if pwrgate_force was asserted

~~2.3.1 Detailed gclock_active behavior~~ Configuration of CDC Timer Values

~~The following describes the conditions under which gclock_active can assert/deassert:~~

- ~~Assertion:~~
  - ~~Asserted in Reset when DEF_PWRON==1~~
  - ~~Asserts when entering RESTORE state and clkack==0~~
  - ~~Asserts when entering ON state~~
  - ~~Asserts when entering FORCE_PENDING~~
  - ~~Can assert in CGATE*, PGATE* states if pwrgate_force is asserted (see section 4.1.4)~~
- ~~Deassertion~~
  - ~~Deasserted in Reset when DEF_PWRON==0~~
  - ~~Deasserts in CGATE_PENDING~~
  - ~~Deasserts in OFF_PENDING if pwrgate_force was asserted~~

~~Configuration of CDC Timer Values~~ to comprehend the requirements around changing these configuration signals.

Table 2-5: Power Control Aggregation Interface Signals

Name	I/O	Clock	Description
pwrgate_disabled	In	pgcb_clk	When set high, idle-based power gating is disabled. Has no effect on forced power gate entry. If this is asserted when the CDC is ready for power gate it will wake up to honor the new policy. This should include all aspects of the enables in the chassis defined Power Control Enable register, including Hardware Autonomous Enable, D3, etc. This must be a glitch free output of a pgcb_clk flop making it safe for clock crossing within the CDC.
pwrgate_force	In	pgcb_clk	Force power gate entry. When asserted, the CDC should prepare for power gating, bypassing all idle timers and waiting only for the agent_ism to go idle. This should be asserted when a ForcePowergatePok message is received by the IP block. *Important*: Once asserted this signal must stay asserted until the IP-Inaccessible or warm reset entry has started and the pgcb_pok signal has de-asserted. It is required for this signal to be de-asserted after the de-assertion of pgcb_pok. This is because the de-assertion of clkreq signal (output from CDC to the SOC) is dependent on the de-assertion of this signal.

			Note that as per Chassis Reset Arch HAS, clkreqs/clkacks have to be de-asserted before the reset to the IP block is asserted by PMC. This must be glitch free making it safe for clock crossing within the CDC.
pwrgate_pmc_wake	In	pgcb_clk	Wake from IP-Inaccessible power gating. This should be driven by the pmc_<ip>_pg_wake signal after it has been synchronized into the pgcb_clk domain.
pwrgate_ready	Out	pgcb_clk	Power gating ready. Indicates that the CDC is locked and can allow power gating. The power gating aggregation logic must use this indication along with the same signal from any other CDCs in the PGD to determine when it is ok to allow power gating. Note: This signal may de-assert this at any time regardless of the state of pgcb_pwrgate_active.

Table 2-6: Test Signals

Name	I/O	Clock	Description
fscan_clkungate	In	*async*	Force gclock to ungate for test modes.
fscan_rstbypen[RST+DRIVE_POK:0]	In	*async*	Scan reset bypass enable. (1 bit per reset synchronizer) Enables the corresponding reset controlled by the CDC to be overridden during scan.
fscan_byprst_b[RST+DRIVE_POK:0]	In	*async*	Scan reset value. (1 bit per reset synchronizer) This drives the value of the resets controlled by the CDC when fscan_rstbypen of the same index is asserted.
fismdfx_force_clkreq	In	*async*	Force assert clkreq. For CDC's that control IOSF Primary or Sideband clock domains, this port should be connected to the top level fism[p|s]dfx_force_clkreq port. For other clock domains, there is no known requirement and may be tied inactive '0'. Note: when this asserts, pgcb_clk must be running for it to have any effect. It will then be synchronized to the pgcb_clk and will cause the domain to power up, unlock, request the clock and ungate the clock and remain in this state

			until it deasserts.
fismdfx_clkgate_ovrd	In	*async*	Forces gclock to ungate. For CDC's that control IOSF Primary or Sideband clock domains, this port should be connected to the top-level fism[p|s]dfx_clkgate_ovrd port. For other clock domains, there is no known requirement and this signal may be tied inactive '0'. Note: when this asserts, the functional clock must be running for it to have any effect as it must be synchronized.
fscan_clkgenctrlen[1:0]	In	*async*	Scan clock bypass enable – this input needs to be propagated to top level of the SIP using the CDC when the CDC is used in pre-SCC configuration within the SIP, i.e. the PRESCC parameter is set to '1'. Refer to section on "CDC Behavior Details" for more information. For CDCs where the PRESCC parameter is set to '0', this input should be tied off to '0'.
fscan_clkgenctrl[1:0]	In	*async*	Scan clock bypass value - this input needs to be propagated to top level of the SIP using the CDC when the CDC is used in pre-SCC configuration within the SIP, i.e. the PRESCC parameter is set to '1'. Refer to section on "CDC Behavior Details" for more information. For CDCs where the PRESCC parameter is set to '0', this input should be tied off to '0'.
cdc_visa[23:0]	Out	pgcb_clk/ gclock	The latest recommendation is that the integrating IP have the VISA tool automatically insert VISA in the CDC and that the cdc_visa output be ignored. See the provided .sig files under $MODEL_ROOT/tools/visa. VISA debug visibility output. VISA signals for the CDC should be connected to the IP's always-on VISA ULM.

3 Integration Notes

3.1 Power Domain

The CDCs along with the PGCB and any glue logic should be in the Always On power domain of the IP.

3.2 Power Control Aggregation Logic

This logic is the "glue" logic between the CDCs, the PGCB and IP-specific controls. This must be done for each IP explicitly since it contains IP-specific interfacing and depends on the number of CDCs present.

This logic handles aggregation of the pwrgate_ready signals from multiple CDCs along with PMC requests to drive the ip_pgcb_pg_rdy_req_b and ip_pgcb_pg_type signals to the PGCB. It also drives the pwrgate_disabled, pwrgate_force and pwrgate_pmc_wake signals to each CDC.

The glue logic operates in the pgcb_clk and includes synchronizing some non-pcgb_clk signals into its clock domain.

The following sections explain how each signal should be driven. Note that if an IP block chooses to implement custom aggregation logic (over and above what is described here), it should be careful about not violating the Chassis requirement (Chassis SIP Powergating spec and Chassis Reset spec) that pmc_<ip>_pg_wake signal has highest priority, even over ForcePwrgatePOK messages. For IP blocks that design the aggregation logic as described below, the CDC handles the prioritization as defined by Chassis.

3.2.1 pwrgate_force

The pwrgate_force signal is a pgcb_clk indication that a ForcePwrGatePOK message was received requiring either a full IP-Inaccessible power gate entry including POK de-assertion, or just POK de-assertion for warm reset.

These two conditions should be clock crossed into the pgcb_clk as independent signals. For descriptions that follow it is assumed that they are crossed into the pgcb_clk as the following signals:

- **force_ip_inaccessible** : pgcb_clk domain signal indicating that a ForcePowerGatePOK message was received with the type specified in the data payload bits 1:0 as 2'b11. This is requesting POK to de-assert and for the IP to enter a full IP-Inaccessible power gated state.
- **force_warm_reset** : pgcb_clk domain signal indicating that a ForcePowerGatePOK message was received with the type specified in the data payload bits 1:0 as 2'b01. This is requesting POK to de-assert in preparation for warm reset but no power gating should be enabled.

Given these signals above, the pwrgate_force signal is simply the OR of the two of these:

```
assign pwrgate_force = force_ip_inaccessible | force_warm_reset;
```

While this signal is required to be glitch free, it doesn't need to be flopped after the OR term since the two conditions will never coincide.

Note that this signal must remain asserted until the CDC starts entry into either the warm reset or IP-Inaccessible states (depending on the type of force requested) and the pgcb_pok is de-asserted. One way of implementing this is ensuring that the pwrgate_force asserts until pgcb_pok is de-asserted by the PGCB. The CDC waits for de-assertion of this signal before de-asserting the "clkreq" output to the SOC when entering IP-Inacc PG or Warm Reset state. Therefore, the IP block is required to de-assert this signal following de-assertion of pgcb_pok (thus enabling rapid de-assertion of clkreq signals to the SOC).

3.2.2 pwrgate_disabled

The pwrgate_disabled signal aggregates several enables and conditions to indicate whether the IP is allowed to power gate. The enables are based on the Power Control Enable (PCE) register as specified in the Chassis Power Management uArch Specification.

This signal must be flopped in the pgcb_clk domain prior to sending to the CDC since it will be crossed over into the CDC's clock domain. The following function describes the expected behavior of this signal (before being flopped). Note that for Ips that have multiple PCI functions, the d3HotActive and d0i3Active terms would need to be vectors, with one bit for each function.

```
/*****************************************************************************************************************\
 *
 *  disablePowerGating
 *
 *  Determines whether power gating should be enabled or disabled based on the values of the Power Control Enable
 *  fields and whether the conditions they control are active.
 *
 *  @param  hardwareAutonomousEnable    : PCE.HAE field value; indicates hardware auto power gate is enabled.
 *  @param  d3HotEnable                 : PCE.D3HE field value; indicates power gating is allowed in D3 hot.
 *  @param  d3HotActive                 : When set to one, this device is in D3 Hot.
 *  @param  d0i3Enable                  : PCE.I3E field value; indicates power gating is allowed in D0i3.
 *  @param  d0i3Active                  : When set to one, this device is in D0i3.
 *  @param  pmcRequestEnable            : PCE.PMCRE field value; indicates power gating is allowed when the
 *                                        pmc_ip_sw_pg_req_b signal is asserted by the PMC.
 *  @param  pmcRequestActive            : Idicates that the PMC is requesting power gating by asserting the
 *                                        pmc_ip_sw_pg_req_b signal low (this is the inverted value of that signal).
 *  @return Asserts when power gating is disabled due to lack of any enabled condition being active as well as
 *          Hardware Autonomous not being enabled.
 *
\*****************************************************************************************************************/
function automatic logic disablePowerGating(logic hardwareAutonomousEnable,
                                            logic d3HotEnable, logic d3HotActive,
                                            logic d0i3Enable, logic d0i3Active,
                                            logic pmcRequestEnable, logic pmcRequestActive);

   if (hardwareAutonomousEnable) return '0;    //Hardwar Autonomous requires no other qualifier
   if (d3HotEnable & d3HotActive) return '0;
   if (d0i3Enable & d0i3Active) return '0;
   if (pmcRequestEnable & pmcRequestActive) return '0;
   return '1;                                  //disabled otherwise

endfunction : disablePowerGating
```

Note that while this shows the disables for the CDC, it is possible that other global disables may exist in a particular IP that could be used to disable power gating at the CDC.

3.2.3 pwrgate_pmc_wake

The pwrgate_pmc_wake signal is simply the pmc_<ip>_pg_wake signal synchronized into the pgcb_clk domain.

3.2.4 ip_pgcb_pg_rdy_req_b

The ip_pgcb_pg_rdy_req_b is the indication to the PGCB that it should enter or stay in a power gated state. This signal may assert or de-assert at any time irrespective of the state of the PGCB. The PGCB will observe and honor any change when it is in an idle state and ignore it the rest of the time.

The CDCs contain all the information to handle all entry or exit conditions including forced entry or exit will control their pwrgate_ready signals appropriately. That makes the ip_pgcb_pg_rdy_req_b simply a NAND of all of the pwrgate_ready signals from each CDC present. For example for a design with 3 CDCs, it would look like this:

```
assign ip_pgcb_pg_rdy_req_b = ~(pwrgate_ready[2] & pwrgate_ready[1] & pwrgate_ready[0]);
```

In some cases there has been an IP specific requirement to allow the CDCs to prepare for power gating but prevent the PGCB itself from entering power gating even when all CDCs are ready for power gating. Those types of conditions would be added to the qualifiers for asserting ip_pgcb_pg_rdy_req_b.

3.2.5 ip_pgcb_pg_type

This signal has an encoded value to indicate to the PGCB what type of power state should be entered. This is only sampled by the PGCB during the cycle when it is powered on and idle and its ip_pgcb_pg_rdy_req_b signal is asserted to indicate that it should start power gate entry. This signal is ignored the rest of the time.

The selection of this is based on force indicators described above in the pwrgate_force description. Using those same signal names, the ip_pgcb_pg_type signal should be driven as follows:

```
assign ip_pgcb_pg_type =    force_ip_inaccessible ?  2'b11 :    //IP-INACC
                                  force_warm_reset         ?    2'b01 : //Warm Reset
                                                                      2'b00 ; //IP-ACC
```

3.3 Clock Requests

3.3.1 Synchronous Requests

Synchronous requests for the clock domain managed by the CDC are made via the gclock_req_sync signal. This signal generally is the logical OR of many 'busy' indications within the IP. It should be glitch free and ideally directly from a flop. In most design this will be used to keep a domain alive rather than to wake it (since it uses the same clock domain that the CDC will gate) but wakes are not precluded.

Clock acknowledges for synchronous requests can be accomplished by ANDing a particular request with the gclock_active signal which indicates that the clock is running.

The clock acknowledges can also generally be done by flopping that request with gclock. If gclock is running and the request is high, then the acknowledge should be valid.

The gclock will remain active for at least 8 clocks after gclock_active de-assertion to ensure consistency with Chassis clkreq/clkack protocols. Please refer to sub-section on "IP-Inaccessible Entry" within the section "CDC Behavior Details" for an exception to this behavior as related to IP-Inaccessible PG and Warm resets (when the signal pwrgate_force is asserted due to IP block receiving a ForcePwrgatePOK message over IOSF-SB).

3.3.2 Asynchronous Requests

The CDC provides a variable number of asynchronous clock request inputs. This allows each async request to be synchronized independently, ensuring the combination of these requests can be kept glitch free.

As a general rule one gclock_req_async signal should be used for each requesting source clock domain. Multiple requesting sources from the same clock domain can flop the OR of all of their requests and send that to the CDC as a single gclock_req_async signal. Each of these inputs requires 2 sets of double flop synchronizers so care should be taken to minimize the number required to minimize the area impact.

The CDC supports a full Chassis-compliant clkreq/clkack handshake for asynchronous sources, providing a gclock_ack_async signal for every corresponding gclock_req_async signal. The acknowledges are driven in the CDC's clock domain and would need to be crossed back into the requesting agents clock domain.

The gclock will remain active for at least 8 clocks after gclock_ack_async de-assertion to ensure consistency with Chassis clkreq/clkack protocols. Please refer to sub-section on "IP-Inaccessible Entry" within the section "CDC Behavior Details" for an exception to this behavior as related to IP-Inaccessible PG and Warm resets (when the signal pwrgate_force is asserted due to IP block receiving a ForcePwrgatePOK message over IOSF-SB).

3.4 IOSF Interfacing

3.4.1 ISM Locking

For the fabric ISM, the ism_locked signal can be used to mask off the value seen by the IOSF interface controller. The fabric ISM should be driven in combinatorial manner to zero to indicate IDLE when the "ism_locked" signal is asserted.

Important: The "ism_fabric" signal to the CDC must be the unmasked version directly from the fabric.

Agent ISM locking can be done using either the current_state of the ISM or the next_state of the ISM. It is required to use the pre-flop locking mechanism for both cases (however, a parameter needs to be set differently for using the next_state of the ISM).

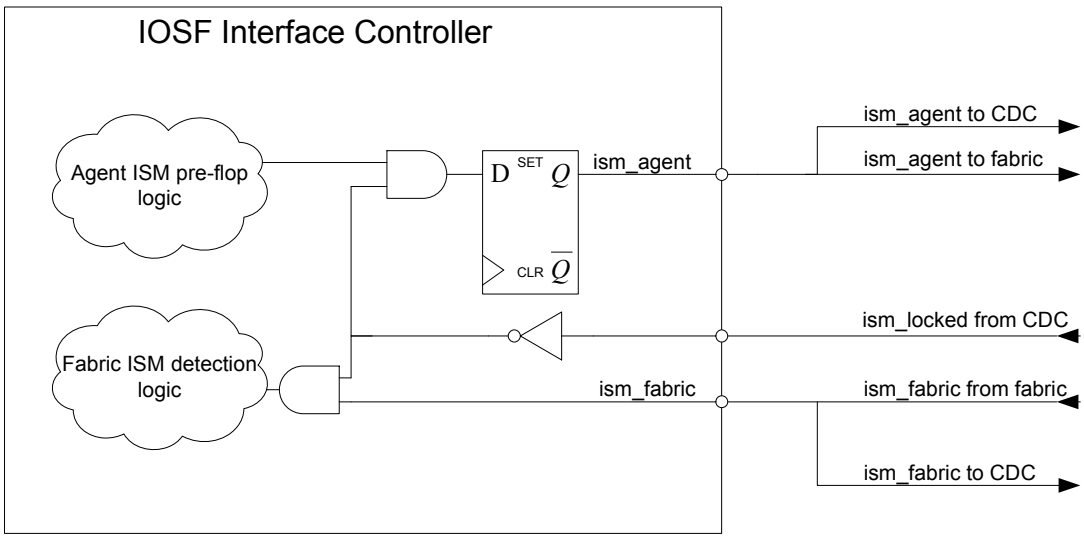
The Post-flop ISM locking approach violates the IOSF spec requirement that outputs be driven directly from a flop and also has the potential to cause a metastability issue in the IOSF fabric – hence this option must *not* be used by IP blocks.

3.4.1.1 Agent ISM Locking Pre-Flop – using current_state

For the agent ISM, the proper way to handle this is to apply the "ism_locked" signal before the flopped output. The input to the final flop that drives the "*_ism_agent" signal should be masked such that it is driven to zero to indicate IDLE when the "ism_locked" signal is asserted.

In the first case, the "ism_agent" signal to the CDC can be the same as the one being provided to the fabric. The CDC supports this by asserting the "ism_locked" signal in combinatorial manner when the ISM is idle (for IP-Inacc entry and Warm Reset).

Figure 3-1: ISM Locking Pre-Flop Example (with combi Lock behavior)



3.4.1.2 Agent ISM Locking Pre-Flop – using next_state

For the second case, the CDC needs to be configured through setting the parameter "ISM_AGT_IS_NS" in order to have the "ism_locked" signal assert from a flop in response to the ISM going to IDLE state (for IP-Inacc entry and Warm Reset).

Important: With this method, the "ism_agent" signal to the CDC must be the ISM value before this mask before the flop.

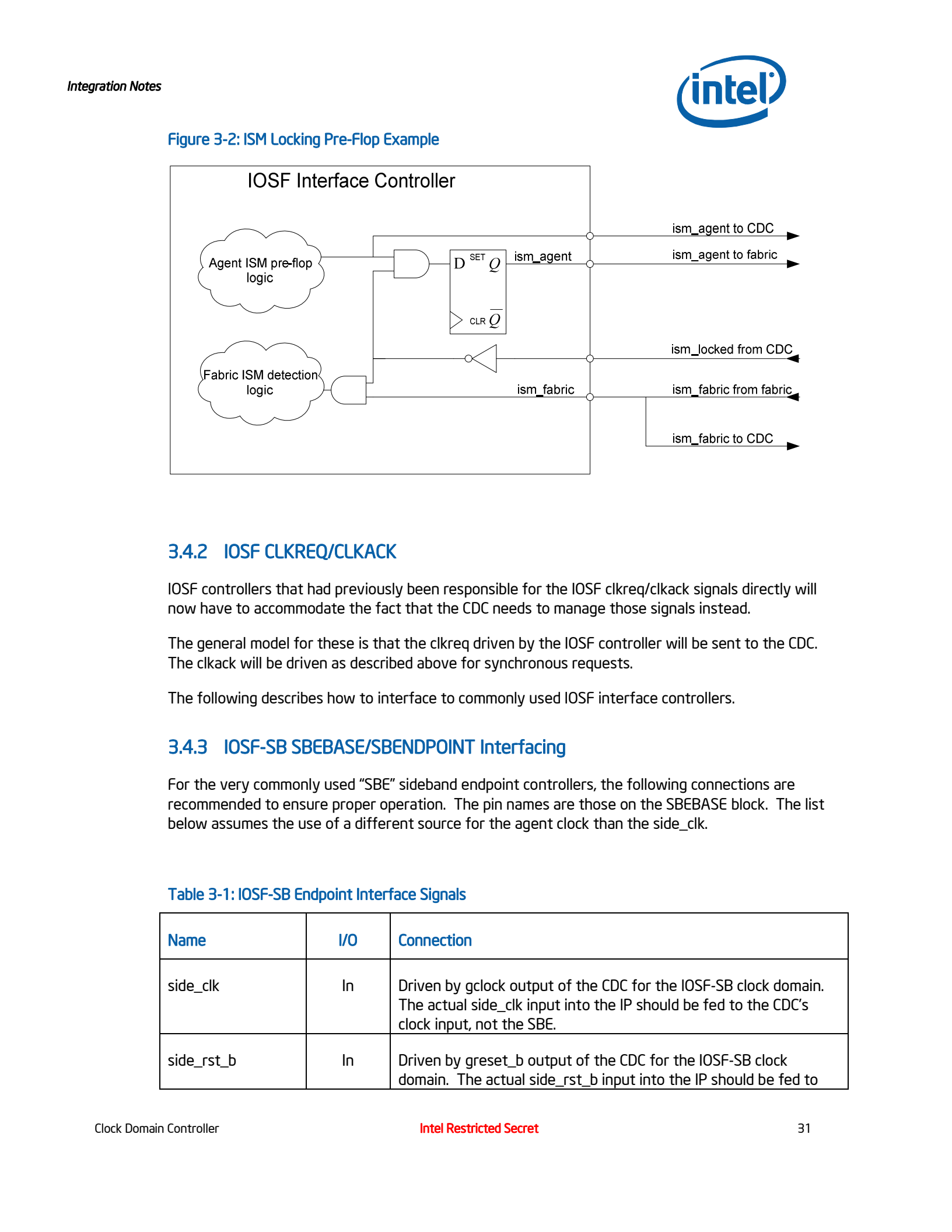


Figure 3-2: ISM Locking Pre-Flop Example

3.4.2 IOSF CLKREQ/CLKACK

IOSF controllers that had previously been responsible for the IOSF clkreq/clkack signals directly will now have to accommodate the fact that the CDC needs to manage those signals instead.

The general model for these is that the clkreq driven by the IOSF controller will be sent to the CDC. The clkack will be driven as described above for synchronous requests.

The following describes how to interface to commonly used IOSF interface controllers.

3.4.3 IOSF-SB SBEBASE/SBENDPOINT Interfacing

For the very commonly used "SBE" sideband endpoint controllers, the following connections are recommended to ensure proper operation. The pin names are those on the SBEBASE block. The list below assumes the use of a different source for the agent clock than the side_clk.

Table 3-1: IOSF-SB Endpoint Interface Signals

Name	I/O	Connection
side_clk	In	Driven by gclock output of the CDC for the IOSF-SB clock domain. The actual side_clk input into the IP should be fed to the CDC's clock input, not the SBE.
side_rst_b	In	Driven by greset_b output of the CDC for the IOSF-SB clock domain. The actual side_rst_b input into the IP should be fed to

		the CDC's reset_b input, not the SBE.
side_clkreq	Out	Should drive a gclock_req_async input on the CDC controlling the side_clk domain.
side_clkack	In	Should be driven by the gclock_ack_async output of the CDC controlling the agent_clk domain. This must be the gclock_ack_async bit that corresponds to the gclock_req_async bit driven by the SBE's side_clkreq signal.
sbi_sbe_clkreq	In	Recommend tying to 0. Clock reqs for running side_clk should go directly to the CDC.
agent_clk	In	Driven by gclock output of the CDC for the Agent clock domain
agent_side_rst_b	In	Driven by greset_b output of the CDC for the Agent clock domain
sbe_sbi_clkreq	Out	Clock request for agent_clk. Should drive a gclock_req_async input on the CDC controlling the agent_clk domain.
sbe_sbi_idle	Out	Clock request for agent_clk. Should be inverted to drive the gclock_req_sync input on the CDC controlling the agent_clk domain.
sbi_sbe_idle	In	No change to this connection. Use what had been done in the past for communicating that the agent is idle.
side_ism_fabric	In	This should be driven by the masked version of the fabric ISM as shown in Section 3.4.1.
side_ism_agent	Out	This should be connected to the "ism_agent" input of the CDC. No masking needs to be done on this signal since the sideband endpoint provides a separate signal ("side_ism_lock_b") for that purpose.
side_ism_lock_b	In	Drive this with inverted version of "ism_locked" signal output of CDC.

3.4.4 IOSF-P Interfacing using IOSFIU and IOSF COMLIB Blocks

IOSF-Primary controllers are less standardized than IOSF-SB endpoints. However there are many Ips using the sub-controller blocks from the ISOF COMLIB and another block called "IOSFIU" which wraps these COMLIB sub-blocks has become somewhat standard as a complete controller solution. This section describes how to interface to the IOSFIU block, with the expectation that this will also be useful for other controllers built on the COMLIB sub-blocks.

Table 3-2: IOSF-Primary Agent Endpoint Interface Signals

Name	I/O	Connection
osc_clk	In	There is no need for a different osc_clk since async detection will be handled elsewhere. So this can be tied to the same source as the prim_clk input. If an osc_clk is being used in an existing design, keeping this as-is will have no negative effects.
osc_rst_b	In	Similar osc_clk, this signal can be connected to the same source as the prim_rst_b input, or it can be left as is in an existing design.
iosf_prim_clk	In	Driven by gclock output of the CDC for the IOSF-P clock domain. The actual prim_clk input into the IP should be fed to the CDC's clock input, not directly to this pin.
iosf_prim_rst_b	In	Driven by greset_b output of the CDC for the IOSF-P clock domain. The actual prim_rst_b input into the IP should be fed to the CDC's reset_b input, not directly to this pin.
prim_clkreq	Out	Asynchronous IOSF-P clock request. Leave unconnected. IOSF-P clock request operations to the CDC can be completely handled by the it_int_biu_actvreq signal.
prim_clkack	In	Use a synchronous clkack using it_int_biu_actvreq as the corresponding clkreq signal. The ways of generating this handshake are described in Section 3.3.1. This signal will be driven by the same source as int_it_biu_actvack.
prim_pok	In	Drive this with inverted version of "ism_locked" signal output of CDC.
it_int_biu_actvreq	Out	Synchronous IOSF-P clock request. Should drive the gclock_req_sync input on the CDC controlling the IOSF-IP clock domain.
int_it_biu_actvack	In	Tie to the same source as the prim_clkack input described above.
int_it_biu_clkreq	In	Should be driven by the it_int_biu_actvreq. The IP should send any requests for the IOSF-P clock directly to the CDC. After doing this, this loopback is required to satisfy an internal dependency in the IOSFIU block.
it_int_biu_clkack	Out	Leave unconnected. No information from this required.
int_clk	In	Driven by gclock output of the CDC for the agent clock domain (or the IOSF-P clock domain if not using a clock crossing with the IOSFIU).
int_rst_b	In	Driven by greset_b output of the CDC for the agent clock domain

		(or the IOSF-P clock domain if not using a clock crossing with the IOSFIU).
prim_ism_fabric	In	This should be driven by the masked version of the fabric ISM as shown in Section 3.4.1.
prim_ism_agent	Out	This should be connected to the "ism_agent" input of the CDC. No masking needs to be done on this signal since the there is a separate signal ("prim_pok") provided by the IOSF primary interface logic for that purpose.
int_it_biu_coreactv	In	Should be driven as before, coalescing all IP requests to use the IOSF-P interface.

3.5 Boundary Locking

The CDC has two types of locking mechanisms: ISM locking, which was described above, and "boundary locking". While ISM locking has a well-defined specific use that is common to all Ips, boundary locking will be used differently in each IP. In many cases it may not be needed at all.

During a power down exit that requires state to be restored, the CDC and PGCB will cause the ISMs to be unlocked to allow the state to be written from the PMC or other agent performing the restore operation. The SoC must guarantee that no agent in the system will access the IP via IOSF during this period for any purpose other than restoring state.

During this restore window, it is important that no other agent access state in the IP. While the SoC has the requirement to ensure IOSF traffic is only used for restore, other paths for accessing state in the IP must be blocked until the restore window completes. The boundary_locked signal stays asserted during this restore window to allow the IP to hold back access from non-IOSF accesses.

Unfortunately there is no clear prescription for exactly how to use boundary locking. Each IP will need to be analyzed to ensure that its state is firewalled until restore completes. Along with the above description of the intent of boundary locking, the following are examples of when it should and should not be used.

Examples of how to use boundary locking:

- Blocking external IO interfaces. For example, stalling upstream activity from a PCIE port until restore has completed in a PCIE root port.
- Preventing access to state values by a microcontroller internal to the IP.
- Preventing autonomous operations within the IP from happening if they rely on state that needs to be restored.

Examples of how not to use boundary locking:

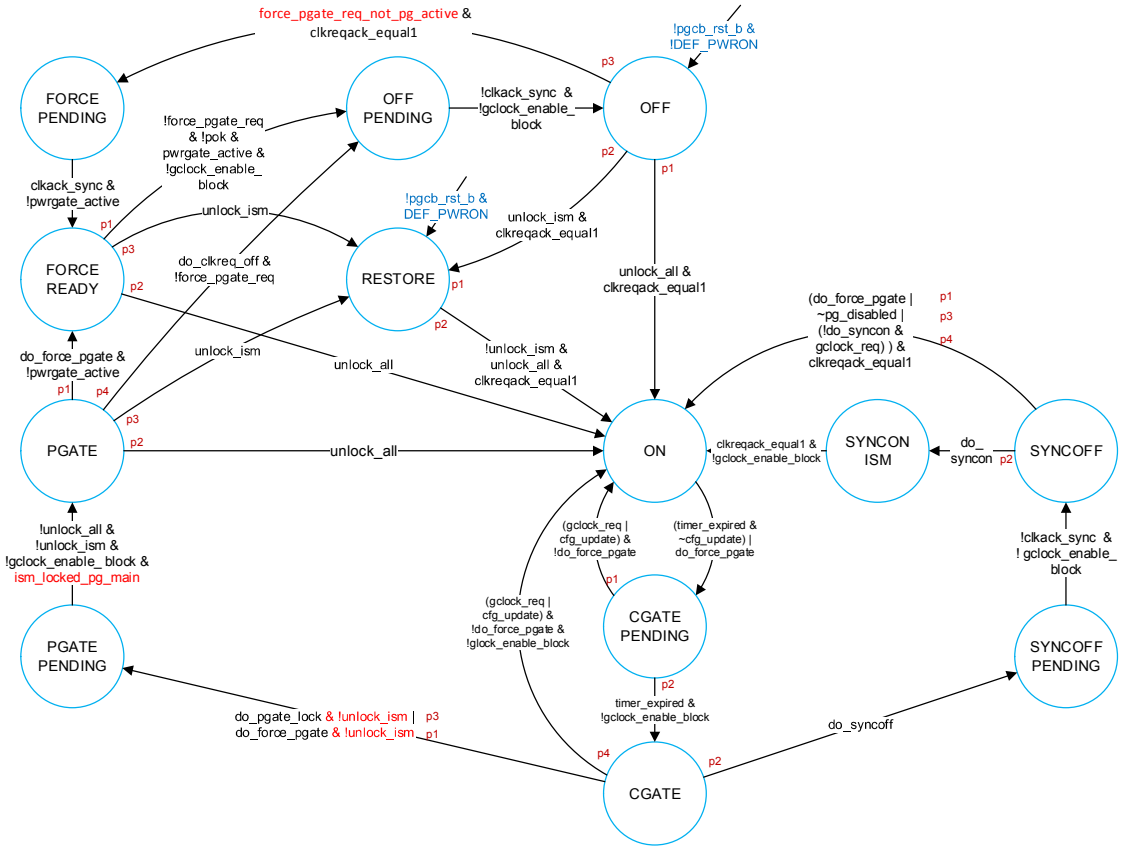
- Preventing access from IO interfaces in IPs that only support power gating in D3. SW would need to re-enable the controller before access is allowed ensuring state will be valid at that time.
- Preventing access in a domain from traffic that originates on IOSF-SB or IOSF-P.

4 Behavioral Overview

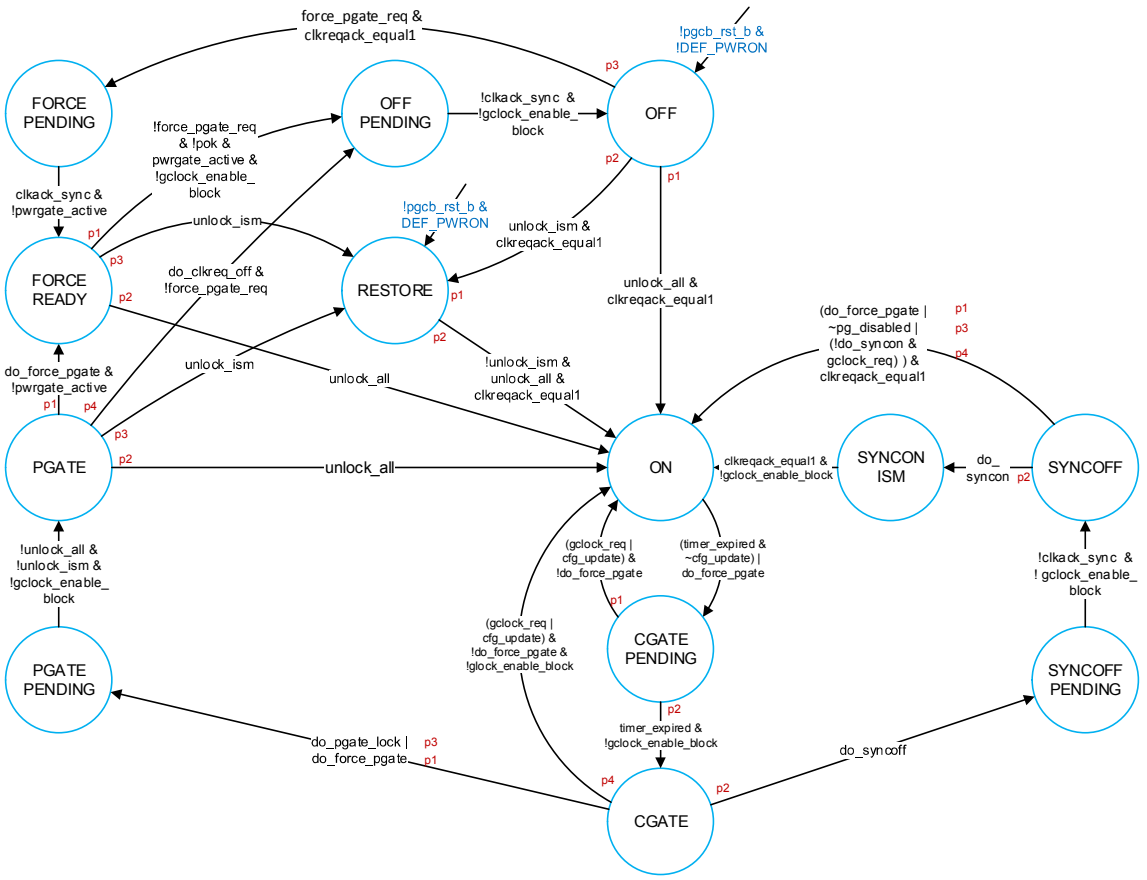
4.1 CDC State Machine

4.1.1 State Diagram

Figure 4-1: CDC State Machine




```
assign do_force_pgate      = force_pgate_req & ( ism_agent==3'b000 | ism_locked );
assign do_syncoff          = (current_state == CDC_CGATE) & timer_expired &  pg_disabled & ~cfg_clkreq_ctl_disabled & ~cfg_update;
assign do_pgate_lock       = (current_state == CDC_CGATE) & timer_expired & ~pg_disabled & ~cfg_update;
assign do_clkreq_off       = (current_state == CDC_PGATE) & timer_expired & ~cfg_clkreq_ctl_disabled;
assign do_syncon           = (current_state == CDC_SYNCOFF) & (ism_fabric != 3'b000);
assign cfg_update          = (pg_disabled ^ pg_disabled_q) | (cfg_clkreq_ctl_disabled ^ cfg_clkreq_ctl_disabled_q) |
                             (cfg_clkgate_disabled ^ cfg_clkgate_disabled_q)
assign clkreqack_equal1    = (clkreq_hold & clkack_sync)
assign gclock_enable_block = (gclock_enable ^ gclock_enable_ack)
```




```
assign do_force_pgate      = force_pgate_req & ( ism_agent==3'b000 | ism_locked );
assign do_syncoff          = (current_state == CDC_CGATE) & timer_expired &  pg_disabled & ~cfg_clkreq_ctl_disabled & ~cfg_update;
assign do_pgate_lock       = (current_state == CDC_CGATE) & timer_expired & ~pg_disabled & ~cfg_update;
assign do_clkreq_off       = (current_state == CDC_PGATE) & timer_expired & ~cfg_clkreq_ctl_disabled;
assign do_syncon           = (current_state == CDC_SYNCOFF) & (ism_fabric != 3'b000);
assign cfg_update          = (pg_disabled ^ pg_disabled_q) | (cfg_clkreq_ctl_disabled ^ cfg_clkreq_ctl_disabled_q) |
                             (cfg_clkgate_disabled ^ cfg_clkgate_disabled_q)
assign clkreqack_equal1    = (clkreq_hold & clkack_sync)
assign gclock_enable_block = (gclock_enable ^ gclock_enable_ack)
```

4.1.2 State Descriptions

Table 4-1: State Descriptions

Name	Description
ON	**Full on state.** Domain is unlocked, clkreq is asserted and gclock is running.
CGATE_PENDING	**Prepare for Clock Gate.** The gclock is running, and the clkreq is asserted but the gclock_active and gclock_ack_async signals are de-asserted (except for case of force_pgate_req being asserted – where gclock_ack_async signals mirror the corresponding gclock_req_async

	signals). The FSM sits in this state for 8 cycles to enforce the requirement that clocks continue to run for 8 cycles after a clkack is de-asserted to the gclock requesters.
CGATE	**Clock gated state.** The gclock signal is not running and gclock_active is de-asserted (except for the case of force_pgate_req being asserted – where the gclock is running and gclock_active is asserted), but clkreq is still asserted and the domain is unlocked.
SYNCOFF_PENDING	**Prepare for Sync Clkreq Off**. The gclock signal is not running (refer to Table 4-3 for exceptions) and clkreq is de-asserted, but the domain is unlocked. CDC waits in this state for clkack de-assertion.
SYNCOFF	**Sync Clkreq Off**. The gclock signal is not running (refer to Table 4-3 for exceptions) and clkreq is de-asserted, but the domain is unlocked.
SYNCON_ISM	**Sync On Due to Fabric ISM**. Domain is unlocked, clkreq is asserted and gclock is running. Though the clock is running, clkack has not yet been sampled asserted. Note that gclock_active and gclock_ack_async will not assert until clkack is asserted and the FSM moves to ON.
PGATE_PENDING	**Prepare for Power Gating Lock**. The gclock signal is not running (except for case of force_pgate_req being asserted – where gclock is running, gclock_active is still asserted, and gclock_ack_async signals mirror the corresponding gclock_req_async signals) and the domain is locked but clkreq remains asserted. This state is required for ensuring previous unlock requests have cleared from the pgcb_clk side of the CDC prior to sending a lock indication back over to the CDC's pgcb_clk logic.
PGATE	**Power Gating Lock**. The gclock signal is not running (except for case of force_pgate_req being asserted – where gclock is running, gclock_active is still asserted, and gclock_ack_async signals mirror the corresponding gclock_req_async signals) and the domain is locked but clkreq remains asserted. Domain is ready for power gate. All gclock_req* assertions and ISM wakes will require sampling in the pgcb_clk and checking for pwrgate_active before unlocking and re-starting the domain.
OFF_PENDING	**Prepare for Clkreq Off.** The gclock signal is not running, the domain is locked and clkreq is de-asserted. Waits in this state for clkack de-assertion. Domain is still for power gate.
OFF	**Clkreq Off**. The gclock signal is not running, the domain is locked and clkreq is de-asserted. Domain is ready for power gate. All gclock_req* assertions and ISM wakes will require sampling in the pgcb_clk and checking for pwrgate_active before unlocking and re-starting the domain. NOTE: This is the default reset state when the DEF_PWRON parameter is 0.

RESTORE	**Power Gate Exit Restore**. Domain ISM are unlocked to allow state restore operations, but the boundary remains locked; clkreq is asserted and gclock is running. gclock_active is asserted, and gclock_ack_async signals mirror the corresponding gclock_req_async signals. NOTE: This is the default reset state when the DEF_PWRON parameter is 1 (based on Chassis PG ECN 1570775).
FORCE_PENDING	**Force Power Gate Ready Pending.** Force power gating has completed all operations to lock the domain, but waits in this state until the clkack has been received to ensure that power gating entry does not start until the clkreq/clkack handshake completes with clock running. This covers a boundary case where a forced power gate or warm reset could complete causing clkreq de-assertion prior to a pending clkack assertion, which would violate the clkreq/clkack handshake protocol. The gclock signal is running, gclock_active is still asserted, gclock_ack_async signals mirror the corresponding gclock_req_async signals and the domain is locked but clkreq remains asserted.
FORCE_READY	**Force Power Gate Ready.** The domain is prepared for forced power gate entry or warm reset. The gclock signal is running, gclock_active is still asserted, gclock_ack_async signals mirror the corresponding gclock_req_async signals and the domain is locked but clkreq remains asserted.

4.1.3 State Transitions

The following are the state transitions (with high level description of relevant conditions) for the CDC's FSM (check the CDC FSM figure for priority order of each transition – and also for full details about the transition conditions).

Table 4-2: State Transitions

Current State	Next State	Condition
OFF	ON	An "unlock all" indication has been received from the CDC's pgcb_clk domain and clkreq/clkack are asserted.
OFF	RESTORE	An "unlock ism" indication has been received from the CDC's pgcb_clk domain and clkreq/clkack are asserted.
OFF	FORCE_PENDING	Force power gate is requested and clkreq/clkack are asserted.
ON	CGATE_PENDING	Force power gate and ism_agent is IDLE -or-

		All gclock requests and the fabric ISM have been idle for 2^cfg_clkgate_holdoff clock cycles AND none of the "cfg_clkgate_disabled, cfg_clkreq_ctl_disabled, pwrgate_disabled" have changed in this clock cycle.
CGATE_PENDING	ON	A gclock request is asserted or fabric ISM transitions out of IDLE or one of the signals for cfg_*_disabled, pwrgate_disabled have changed AND force power gate request is not asserted.
CGATE_PENDING	CGATE	Force power gate request is asserted and ism_agent is IDLE (or locked in IDLE) and 8 cycles have elapsed in this state -or- All gclock requests and the fabric ISM have been idle for 8 clock cycles in this state.
CGATE	PGATE_PENDING	Force power gate request and ism_agent is IDLE (or locked in IDLE) -or- Power gating is not disabled and all gclock requests and the fabric ISM have been idle for 2^cfg_pwrgate_holdoff clock cycles.
CGATE	SYNCOFF_PENDING	Power gating is disabled, CLKREQ control is not disabled and all gclock requests and ISMs have been idle for 2^cfg_clkreq_off_holdoff clock cycles.
CGATE	ON	A gclock request is asserted or fabric ISM transitions out of IDLE or one of the signals for cfg_*_disabled, pwrgate_disabled have changed, and force power-gate request is not asserted.
SYNCOFF_PENDING	SYNCOFF	The synchronized version of clkack is de-asserted.
SYNCOFF	ON	Force power gate and ism_agent is IDLE -or- Power gating is enabled -or- A clock request other than the fabric ISM transition from IDLE is detected. Additionally, clkreq/clkack must be asserted to move to ON.
SYNCOFF	SYNCON_ISM	The fabric ISM transitions out of IDLE.
SYNCON_ISM	ON	The synchronized version of clkack is asserted.
PGATE_PENDING	PGATE	All unlock requests from the CDC's pgcb_clk domain have been cleared. (Waiting in this state

		for this to clear is important to ensuring that the unlock received was not for a previous lock.)
PGATE	FORCE_READY	Force power gate request and ism_agent is IDLE (or locked in IDLE)
PGATE	ON	An "unlock all" indication has been received from the CDC's pgcb_clk domain.
PGATE	RESTORE	An "unlock ism" indication has been received from the CDC's pgcb_clk domain.
RESTORE	ON	An "unlock all" indication has been received from the CDC's pgcb_clk domain AND the restore is complete AND the synchronized version of clkack is asserted.
OFF_PENDING	OFF	The synchronized version of clkack is de-asserted.
FORCE_PENDING	FORCE_READY	The synchronized clkack is asserted.
FORCE_READY	OFF_PENDING	The force power gate request is de-asserted AND POK has been de-asserted AND pwrgate_active indication from PGCB is asserted.
FORCE_READY	RESTORE	When "unlock_ism" is asserted by CDC's pgcb_clk domain.
FORCE_READY	ON	When "unlock_all" indication is received from CDC's pgcb_clk domain.

4.1.4 Clock Gating and FSM States

The following table indicates the relationship of clock-gating of the gclock with respect to the states of the CDC FSM.

Note that for IP-Inaccessible power-gating, if the CDC clock domain is really slow compared to the PGCB clock domain, it is possible that the clock remain running for a short time while the PGCB has initiated the power-gating entry sequence. This will not cause any issues with the state-retention cells as they will be reset cleanly on an exit from PG regardless.

Table 4-3: Clock-gating in Various FSM States

State Name	cfg_clkgate_disabled == '0', pwrgate_force == '0'	cfg_clkgate_disabled == '0', pwrgate_force == '1'	cfg_clkgate_disabled == '1', pwrgate_force == '0'	cfg_clkgate_disabled == '1', pwrgate_force == '1'

ON	UNGATED	UNGATED	UNGATED	UNGATED
CGATE_PENDING	UNGATED	UNGATED	UNGATED	UNGATED
CGATE	GATED	UNGATED	UNGATED	UNGATED
SYNCOFF_PENDING	GATED	GATED	UNGATED	UNGATED
SYNCOFF	GATED	GATED	UNGATED	UNGATED
SYNCON_ISM	UNGATED	UNGATED	UNGATED	UNGATED
PGATE_PENDING	GATED	UNGATED (if pwrgate_active == '0')	GATED	UNGATED (if pwrgate_active == '0')
PGATE	GATED	UNGATED (if pwrgate_active == '0')	GATED	UNGATED (if pwrgate_active == '0')
RESTORE	UNGATED	UNGATED	UNGATED	UNGATED
FORCE_PENDING	n-a	GATED first, then UNGATED	n-a	GATED first, then UNGATED
FORCE_READY	n-a	UNGATED	n-a	UNGATED
OFF_PENDING	GATED	GATED	GATED	GATED
OFF	GATED	GATED	GATED	GATED

4.2 CDC Behavior Details

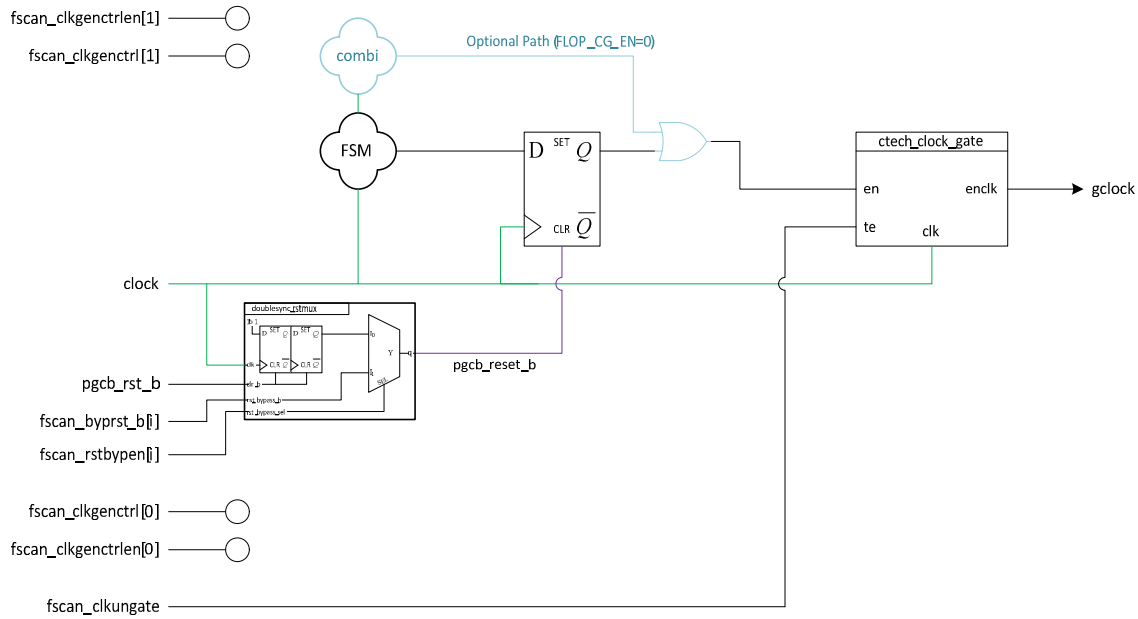
This section provides details and explanations of CDC behavior around certain interesting scenarios.

4.2.1 Configuration of CDC for Asynchronous CG_Enable

The CDC parameter DSYNC_CG_EN defines if the clock-gate enable may be driven synchronously by the CDC FSM or not. The CDC FSM shares the same clock as the functional logic within the SIP using the "*gclock*" (synchronous relationship). When the DSYNC_CG_EN == '0', the logic driving the clock-gate enable within the CDC is represented in the following figure.

Figure 4-2: CDC with Synchronous Clock-gate Enable (DSYNC_CG_EN == '0')

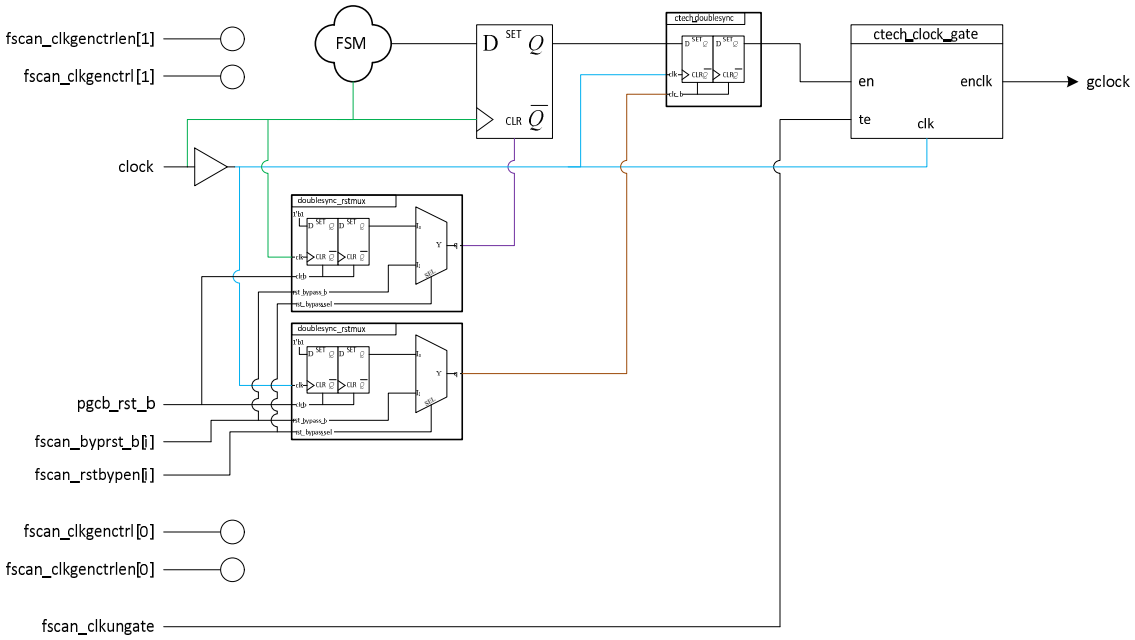
PRESCC=0 && DSYNC_CG_EN=0



However, the clock-gate on the gclock is instantiated higher up in the clock tree for the '*gclock*'. Because of this skew in the clock tree, it is recommended (based on SD feedback) that the clock-gate enable be passed through clock synchronization logic (based on clock branch on which the clock-gate is applied). This asynchronous interface between CDC FSM and the clock-gate is supported by setting the DSYNC_CG_EN parameter to '1'. For this setting, the logic driving the clock-gate enable within the CDC is represented in the following figure.

Figure 4-3: CDC with Asynchronous Clock-gate Enable (DSYNC_CG_EN == '1')

PRESCC=0 && DSYNC_CG_EN=1
NOTE (relationship with other parameters): For this case, HW logic is equivalent to setting FLOP_CG_EN==0 (regardless of actual value provided in the design for this parameter).



NOTE: When the CDC parameter PRESCC is set to 1, the HW implementation is equivalent to having DSYNC_CG_EN == '1' and FLOP_CG_EN == '1' (regardless of the value provided in the design for these two parameters). Refer to description on usage of CDC as pre-SCC logic in the Appendix.

NOTE: For SIPs that need to set DSYNC_CG_EN == '1' (or PRESCC == '1'), refer to parameter definition for CG_LOCK_ISM to check if SIP needs to set that parameter also to '1'.

- Note that when parameter DSYNC_CG_EN is set to '1' or PRESCC is set to '1', in the case, when the clock is being turned OFF, there exists a scenario where the "gclock_active" signal is de-asserted, but the "gclock" is actually still toggling.
- Similarly, when the clock is being turned ON, there exists a scenario where the "gclock" has started toggling but "gclock_active" signal is still de-asserted.
- IMPORTANT: SIP may need to review their logic to ensure that there are no issues with the clocks toggling when "gclock_active" is de-asserted (or else, the "gclock_active" signal may need to be used in more parts of the logic, for example).

IMPORTANT: By nature of the asynchronous interface, whenever DSYNC_CG_EN == '1' or PRESCC == '1', there is additional latency to turn ON or turn OFF the clock through the clock-gate. The latency to gate/ungate the clock is typically 4-6 clock cycles as compared to 1 clock cycle turn-around with PRESCC == '0', DSYNC_CG_EN == '0' and FLOP_CG_EN == '0'. If the CDC user is concerned about the latency impact, there are two possible options to mitigate the latency cost.

4.2.1.1 Option 1: Get SD agreement on using DSYNC_CG_EN == '0'

The clock skew between buffered clock and clock input to clock-gate in the CDC has more impact on PV timing when the 'clock' has a high frequency. In some cases, where the clock is not a high frequency clock, or if the logic in the PGD for that clock has low gate count, the Structural Design (SD) team may be able meet PV timing even assuming a synchronous interface. Therefore, the SIP team should communicate with the SD team for the applicable project to get SD agreement to have the DSYNC_CG_EN parameter remain '0' for the CDCs used in the SIP.

Note that this communication is project-specific, and should include each CDC in the PGD of the SIP.

4.2.1.2 Option 2: Disable CDC clock-gating using 'cfg_clkgate_disabled'

If the SD team requires the SIP to use the value of '1' for the DSYNC_CG_EN parameter, there is another option to avoid the latency impact of clock-gating and ungating (at least for some cases) by disabling the clock-gate within the CDC. This may be done through setting the CDC input signal "cfg_clkgate_disabled".

Even though the asynchronous interface between CDC FSM and CDC clock-gate is still present, but the clock-gate is disabled, therefore the SIP would not see any performance/latency impact due to clock-gating within the CDC.

In order to use this option, the SIP may need to implement lower-level clock gating within the PGD (since the CDC clock-gate is disabled). The CDC can still de-assert the clkreq for the functional 'clock' under the appropriate conditions when power-gating is disabled (i.e. CDC allows support of trunk clock gating even when CDC clock-gating is disabled). Even with this case of clkreq de-assertion, the SIP is able to respond very quickly to fabric wakes on this clock domain (fabric transaction coming downstream to the SIP).

NOTE however that when the SIP enters a power-gated state, the "cfg_clkgate_disable" has no effect, and the clock is still gated to the PGD by the CDC regardless of the value of 'cfg_clkgate_disabled' signal.

4.2.2 IP-Inaccessible Entry: CDC Response to "pwrgate_force" Assertion

Normally, CDC requires all "gclock_req_async/sync signals to be de-asserted, agent ISM state to be in IDLE, fabric ISM state to be IDLE, and hysteresis requirements to have been satisfied before it asserts the "*pwrgate_ready*" signal.

However, when the "*pwrgate_force*" signal is asserted (implying that a ForcePwrgatePOK message has been received by the IP block, and an IP-Inaccessible PG entry or a Warm reset Entry sequence has to be started), the CDC behaves in the following manner:

- First, the CDC de-asserts the "*pwrgate_ready*" signal (in case it is already asserted).

- Further, the CDC attempts to put the clock domain in a locked state –and the only requirement that the CDC satisfies before locking the ISM is to find one clock cycle where the agent ISM is in IDLE state.

- The CDC does not count for any hysteresis delays in this case, nor does it wait for de-assertion of various gclock_req_async/sync signals.

- However, the CDC waits for the signal "*pwrgate_active*" to be de-asserted before it next asserts the "*pwrgate_ready*" signal.

    - This is a requirement from the PGCB – that when entering IP-Inaccessible PG state, the CDC needs to wait for at least one clock cycle where the "*pwrgate_active*" signal is de-asserted, before asserting the "*pwrgate_ready*" signal to the PGCB.

4.2.2.1 IP-Inacc PG (or Warm Reset) Entry request while PGD is in Ip-Acc PG state

It is noteworthy that for the case where an IP block's PGD is in IP-Accessible PG state, and PMC requests entry into IP-Inaccessible PG, the CDC de-asserts the "pwrgate_ready" signal until such time that both of the following conditions are met:

1. CDC functional clock state machine has entered the ForceReady state (implying ISMs and boundaries are locked and clocks are available to de-assert the *_pok signals – as applicable) AND
2. PGCB has de-asserted the "pgcb_pwrgate_active" signal (implying that the PGCB has completed the IP-Acc PG exit flow).

Important: Note that the CDC does not have visibility into any pending downstream completions, or whether all credits have been returned by the agent to the fabric or not (or any other requirements imposed by the IOSF spec with reference to power gating of the interface).

- Therefore, it is required that SIP blocks using the CDC either prevent the ISM from reaching IDLE state until these requirements are all met, OR

- SIP blocks mask the agent ISM input seen by the CDC until such requirements have been satisfied OR

- SIP blocks delay the CDCs within the SIP from seeing '*pwrqate_force*" signal as asserted until such requirements are met.

4.2.2.2 Support for Resolving ISM-Clock Dependencies in SIPs for IP-Inaccessible Entry

The minimum requirement for a SIP block to proceed with IP-Inaccessible Entry is to get its ISMs for the fabrics in IDLE state and meet related IOSF requirements for de-asserting the prim/side_POK signal to the fabric.

- However, it is possible that an SIP block is processing some transaction when the ForcePwrGatePOK message (with relevant opcode) is received on IOSF-SB (this is the trigger for IP-Inaccessible Entry). In order to complete the processing and gracefully get the ISM to IDLE state and de-assert POK, it is possible that the SIP may need functional clocks, even other than the fabric clocks.

- Therefore, when the CDC comprehends that an IP-Inaccessible Entry has been requested, it ungates the clock to the PGD of the SIP block.

- Thus, all clocks to the PGD of the SIP remain ungated until all the CDCs achieve readiness for PG entry (by assertion of "*pwrgate_ready*" signal) and the PGCB initiates the IP-Inaccessible PG flow.

One common example of an ISM-clock dependency is where a SIP block receives a Non-posted register access message on IOSF-SB interface shortly before or after receiving the ForcePwrGatePOK trigger for IP-Inaccessible Entry.

- This access needs to be completed (or an Unsuccessful completion generated) before the "*side_pok*" signal can be de-asserted.

- However, many SIPs need the prim_clk (IOSF primary fabric interface clock) to complete the register access.

- If the various CDCs in the SIP were allowed to gate the functional clocks independently, it might have resulted in a deadlock scenario.

- However, by virtue of the behavior of the CDC, the prim_clk remains available and the SIP can generate completion for the register access in usual manner and then proceed with IP-Inaccessible PG entry.

4.2.2.3 IMPORTANT: Lack of Support for Resolving ISM-ISM Dependencies in SIPs for IP-Inaccessible Entry

It should be noted that while the CDC does not provide any support for resolving ISM-ISM dependencies around IP-Inaccessible PG Entry.

- It is important for each SIP to review its design to see if it is exposed to this issue.

- SIPs that have such a dependency need to handle this within the functional logic.

A common example would be as follows:

- A Non-posted transaction targets the IOSF primary interface of the SIP.

- It is possible that this transaction results in an interrupt message that needs to be sent out on IOSF-SB, before the completion is sent back on IOSf primary fabric.

- However, a ForcePwrGatePOK message is received on IOSF-SB close to the time of the primary transaction.
- Because of the ForcePwrGatePOK message, the IOSF-SB ISM is locked by the CDC (since it may have reached IDLE state and other spec requirements were met).
- Subsequently, the SIP tries to send out the interrupt message on IOSF-SB, but cannot do so since the ISM is already locked.
- This results in a deadlock scenario, since the primary fabric ISM cannot lock until the completion for the incoming transaction has been sent out – and this is prevented because the interrupt message is not being sent out on IOSF-SB.

4.2.2.4 Cases of non-compliance to Chassis clkreq/clkack behavior on internal gclock_req/ack signals at CDC interface

The CDC honors the Chassis clkreq/clkack protocol on the gclock_ack_async signals in most cases. However, there is an exception to this behavior for the case of IP-Inaccessible PG entry and Warm Reset event.

- For these two cases, the CDC enters a power-gate ready state independent of the value of the gclock_req_async/sync signals at its inputs.
- Eventually, it locks the ISMs and boundaries and reports back a power-gate ready state to the IP Aggregation logic.
- During this time, it keeps the gclock_ack_* signals asserted (assuming the gclock_req_async/sync signals are also asserted).
- However, as soon as the PGCB indicates to the CDC that it (the PGCB) has started the PG entry sequence through assertion of "pgcb_pwrgate_active" signal, the CDC drops gclock_ack_* signals without any regard for the value of the gclock_req_* signals.

This CDC behavior has been chosen after careful thought, and there is no known reason to expect that any functional issue should arise from this behavior. However, IP blocks need to review their usage of the CDC and generation of the gclock_req_sync/async signals to the CDC to confirm that the IP block is not impacted by this CDC behavior.

4.2.3 IP-Inaccessible Exit: CDC Response to "*pmc_<ip>_pg_wake*" Assertion

Some of the key features of the CDC behavior on exiting from IP-Inaccessible are listed below. Some of this behavior is also similar during a Warm Reset exit flow.

4.2.3.1 Behavior of CDC When "*pmc_<ip>_pg_wake*" is Asserted

The CDC takes in pgcb_clk version of the "*pmc_<ip>_pg_wake*" signal. This is through the input signal "*pwrgate_pmc_wake*".

- This signal is used by the CDC to de-assert the "*pwrgate_ready*" output signal – which results in the IP exiting a power-gated state (if already power-gated) and remaining power-ungated (as long as the pmc_<ip>_ip_pg_wake signal remains asserted).

- However, the "*pwrgate_pmc_wake"* signal is not used to move the CDC FSM out of PGATE/OFF state. Hence the "*gclock*" remains gated.

- This behavior is to enable the CDC FSM to be active (and ungate "*gclock*" only if the IP logic requests a wake through one of the "*gclock_req_**" signals (input to the CDC).

If the SIP block using the CDC needs to have the "*gclock*" ungated, but none of the gclock_req_* signals to the CDC are to be asserted, one of the alternative options is to connect the "*pmc_<ip>_pg_wake*" signal to one of the "*gclock_req_async*" inputs to the CDC.

- However, this would result in the clock remaining ungated as long as "*pwrgate_pmc_wake*" signal remains asserted, therefore, the SIP team should evaluate this option carefully before choosing to adopt it.

4.2.3.2 Prioritization between "*pmc_<ip>_pg_wake*" and "*pwrgate_force*"

The Chassis PG spec defines the prioritization that a SIP block needs to honor in case of assertion of "*pmc_<ip>_pg_wake*" signal close to the time that a SIP receives the ForcePwrGatePOK message (on IOSF-SB).

The CDC supports Chassis PG HAS mandated prioritization between "*pmc_<ip>_pg_wake*" signal and the "*pwrgate_force*" signal in the following manner:

- When a rising edge is seen on the "*pmc_<ip>_pg_wake*" (as "*pwrgate_pmc_wake*" input signal to CDC) by the CDC, the CDC checks for the CDC's internal latched value of the "*pwrgate_force*" input signal.

- If the internal latched value of "*pwrgate_force*" input signal is already asserted, the "*pwrgate_pmc_wake*" signal is masked from the rest of the CDC logic, until the "*pwrgate_force*" signal de-asserts. Otherwise, the "*pwrgate_pmc_wake*" signal assertion is latched into the CDC in the next clock cycle.

- Similarly, when a rising edge is seen on the "*pwrgate_force*" signal by the CDC, the CDC checks for the current value of the "*pwrgate_pmc_wake*" input signal (thereby ensuring that "*pwrgate_pmc_wake*" gets priority if both input signals happen to be asserted at the same time).
  .

- If the "*pwrgate_pmc_wake*" signal is already asserted, the CDC masks the "*pwrgate_force*" input signal from the rest of the CDC logic, until the "*pwrgate_pmc_wake*" signal de-asserts. Otherwise, the "*pwrgate_force*" signal assertion is latched into the CDC in the next clock cycle.

4.2.3.3 Clkreq behavior on IP-Inaccessible PG exit (and Warm Reset exit)

The CDC has two sets of behavior for asserting *clkreq* to the SOC (for the clock domain controlled by the CDC) – depending on the value of the parameter DRIVE_POK.

- This parameter, which is intended to be set for CDCs on IOSF fabric clocks, when set, causes the *clkreq* to the SOC to be asserted when *pmc_<ip>_pg_wake* signal is asserted while IP is in IP-Inacc PG state.

- The *clkreq* signal remains asserted at least until the PGCB signal "*pgcb_pok*" is asserted or until the *clkack* from the SOC is asserted (whichever happens later).

- Beyond this point, the *clkreq* signal continues to be asserted if the IP asserts *gclock_req_async/sync* to the CDC, or if there is an ism_wake (transaction requested from fabric) or if IP-Accessible PG is disabled.

- Otherwise, the *clkreq* signal is de-asserted.

For CDCs with DRIVE_POK== '0' (typically, non-IOSF clocks), the CDC has the following behavior:

- When IP is in IP-Inaccessible (or Warm Reset state) and PMC asserts *pmc_<ip>_pg_wake* signal, the *clkreq* to the SOC is asserted only after the PGCB signal "*pgcb_pok*" is asserted.

- For this case, the *clkreq* assertion happens only if one of the following hold true:

1. IP has asserted gclock_req_async/sync to the CDC OR
2. IP-Accessible power-gating is disabled OR
3. Fabric is targeting the IP with a transaction (ism_wake)

- If none of the above is true, then *clkreq* is not asserted to the SOC by a CDC with DRIVE_POK == '0' as part of IP-Inaccessible exit or Warm Reset exit.

4.2.3.4 Recommendation on *gclock_req_** input to CDC for IP-Inaccessible PG exit

It is recommended that at least one of the *gclock_req_** inputs to each CDC instantiated by an IP be asserted as the PGD is powered up, and un-isolated during IP-Inaccessible PG exit.

- This is necessary to ensure that the CDC FSM in the functional clock domain stays in the active "ON" state with *clkreq* signal asserted to the SOC.

- Otherwise, it is possible that the *clkreq* signal to the SOC may de-assert (and remain de-asserted) even before the external reset signal from PMC to the SIP block is de-asserted.

This would violate the intent of the Chassis Reset Architecture spec to have SIP clocks active when reset is de-asserted to the SIP block as part of IP-Inaccessible PG exit.

- One important reason to have the clocks active when external resets to the SIP are de-asserted is that the SIP needs to synchronize these resets to various internal clocks.

- Therefore, in order to capture the de-assertion of the resets to all clock domains (as applicable), it highly recommended that the SIP assert at least one of the *gclock_req_** signal to the various CDCs by default out of an IP-Inaccessible PG Exit.

4.2.4 IP-Accessible Entry: Conditions honored by CDC

For the case of IP-Accessible PG entry, the CDC only considers the signals related to gclock_req_async/sync, agent ISM state and fabric ISM state from the IP in order to decide if it is in IDLE state – where the hysteresis counters can start counting.

Therefore, the IP block is required to ensure that it meets all applicable IOSF spec requirements and other IP requirements around power gating before it de-asserts the gclock_req_* signal/s to the CDC.

4.2.5 Detailed gclock_active behavior

The following describes the conditions under which gclock_active can assert/deassert:

- Assertion:
  - Asserted in Reset when DEF_PWRON==1
  - Asserts when entering RESTORE state and clkack==0
  - Asserts when entering ON state
  - Asserts when entering FORCE_PENDING
  - Can assert in CGATE*, PGATE* states if pwrgate_force is asserted (see section 4.1.4)
- Deassertion
  - Deasserted in Reset when DEF_PWRON==0
  - Deasserts in CGATE_PENDING
  - Deasserts in OFF_PENDING if pwrgate_force was asserted

4.2.6 Configuration of CDC Timer Values and Related Recommendations

The CDC supports various timer values as described in Table 2-4.

Important: It is required that an IP block change any of the timer values only when the corresponding cfg_*_disabled signal is set. An IP block is not allowed to change these values when the corresponding cfg_*_disabled signal is cleared.

The corresponding disable signals for each of the cfg_*_holdoff signals is listed in the interface table description.

- The disable signals are synchronized by the CDC into its functional clock domain (clock input), however, the timer values are only qualified with the corresponding disable signal before being used. Hence the requirement above is applicable.

- Note that timer assumes a default value (8 clocks) when the corresponding disable signal is asserted. Therefore, the time expires in a short time, but since the disable is asserted, it has no impact.

- When a disable signal is de-asserted, the timer reloads the applicable timer value and starts counting afresh.

- There is no requirement on sequencing one or more disable signals with respect to other disable signals.

The interface description also lists out which of the timer values need to be passed through isolation latches (because the value is used when power-gating may already have happened).

NOTE: For any given SIP block that uses the CDC, the SIP team may choose to make some of the timer values as fixed (hard-wired) instead of configurable. This is a SIP-specific decision; however, some general considerations apply:

- The decision to hard-wire a timer value should be a conscious decision, with appropriate justification. The start point should be that all timer values are driven from configuration registers.

- The timer value related to hysteresis before entering a power-gate ready state (*cfg_pwrgate_holdoff*) is strongly recommended to be configurable. Also, the default value should typically be chosen such that the CDC does not enter a power-gate ready state at extremely short intervals (since this behavior may have a negative impact on performance).

- The timer values related to de-assertion of clkreq signal (*cfg_clkreq_off_holdoff* and *cfg_clkreq_syncoff_holdoff*) may be chosen to be a low value since the SOC owns the final decision on turning off the clock source. The SOC also takes into account various system idleness parameters before the clock source is turned off.

4.2.7 Waivers related to CDC

A separate folder in the IRR package of the CDC/PGCB contains the waivers related to logic included in the CDC – which may be needed by IP blocks in order waive certain clock-crossing checks and/or timing analysis checks. Please refer to the "Tools" folder for Lintra waivers related to the CDC and to the waiver list file in the "docs/ClockDomainController" folder.

4.2.8 Miscellaneous Behaviors and Issues

4.2.8.1 Dual-space IP blocks with Logic based on Synchronous Resets

Please refer to Reset Architecture HAS for requirements about single <space/group> reset events as related to multi-space (SOC, Early boot, dual-space IP blocks). Note that it is possible that the CDC may have gated the clock to the <space/group_reset_b> domain that is being reset during the reset/power state transition event. It is also possible that this domain has logic based on synchronous resets, and needs the clock to be ungated to actually propagate the reset within this logic.

It is expected that the IP comprehend the above requirement if it includes logic based on synchronous resets and implement a solution for this issue. One possible solution is to have one of the gclock_req_async/sync signals asserted to the CDC (perhaps based on the pmc_<ip>_pg_wake signal) that contains the synchronous reset based logic within the reset domain of interest.

4.2.8.2 Synchronous wakes from Fabric and gclock_ack/gclock_active behavior

For the case that an IP block is in IP-Accessible PG, and receives a request to accept a transaction from the IOSF fabric, the CDC will wait for power to be ungated and for the clock to be requested and clkack to assert before ungating the clock and asserting gclock-active.

For the case of a fabric wake when the CDC is in the SYNCOFF state, with the 'clkreq' signal de-asserted to the SOC, the CDC will ungate the gclock to allow the clock to the IOSF interface to run (per the IOSF spec). However, the CDC only asserts the 'gclock_active' and 'gclock_ack_*' signals when both 'clkreq' and 'clkack' are asserted at the interface of the SIP with the SOC. The CDC assumes that the SIP's fabric interface logic is able to make forward progress on processing the incoming fabric transaction without dependence on the assertion of the 'gclock_active' or 'gclock_ack_*' signals.

4.2.8.3 Using CDCs for Clocks Which Are Not Available in All Active SIP States

There are cases where a specific clock domain used within a SIP's PGD - is not available in some SIP/system states – even though the SIP may be active and functional in those system states.

- For example, a dual-space SIP may use a high frequency PLL clock in S0 system state, but this PLL may be turned off when the system is not in S0 (though the dual-space SIP is still functional in at least some non-S0 system states).
- For such a clock domain, the SIP team needs to figure out what are the appropriate triggers both for requesting the clock at initial entry in to the relevant active state, and then terminating the clock request cleanly when exiting the relevant active state.
- The trigger for entry into active state should be routed into the "*pwrgate_pmc_wake*" input of the relevant CDC. The trigger for exit of the system from the active state should be routed to the "*pwrgate_force*" input of the CDC. Other signals such as pwrgate_active, pgcb_pok, also need to be taken into consideration.

- An example of a simple sequencer and glue logic to interface with a CDC for such a clock is provided in the Appendix.

4.2.8.4 CDC Clock-gating Support for "pgcb_clk"

The CDC supports clock-gating of the "pgcb_clk" only when all the following signals at the CDC interface are de-asserted:

- "gclock_req_async" and "gclock_ack_async"
- "gclock_req_sync" and "gclock_active"
- "clkreq" and "clkack"
- "Ism_fabric" == '0', and "ism_agent" == '0'

As a corollary, any assertions of the above signals should be taken as a wake condition to ungate the "pgcb_clk" (needs to be handled outside of the CDC).

Also, any change in "pwrgate_disabled" signal should be a cause for ungating/waking the "pgcb_clk".

4.2.8.5 Pulse-width check and Metastability checks on double-syncs used within CDC

All double-syncs at the interfaces of the CDC now have pulse-width check and metastability checks enabled (previousl releases had the checks disabled for some of the double-syncs). These checks may now cause violations to be flagged in the IP regressions. The IP team is responsible for reviewing the violations and comprehending if these represent any real issue or not.

If the IP team is confident that a check (pulse-width or metastability) can be waived for a given double-sync, it can be waived using the verilog 'defparam' directive to override the doublesync instance's parameter. This can be done from any hierarchy with the following syntax:

```
defparam <hierarchy>.<PARAMETER> = 0;
```

This directive could be placed within the top level AON/UNgated wrapper for the IP (note that the translate_off directive is required for lintra and synthesis to ignore this). For example:

```
// synopsys translate_off
   defparam
u_sideClkDomainController.u_CdcPgcbClock.u_ismWakePG.PULSE_WIDTH_CHECK
= 0;
// synopsys translate_on
```

Alternately, this could be done in the testbench somewhere (for example):

```
   defparam
lpio_spt_tb.spt_lpio_wrapper1.i_spt_lpio_aon_wrapper.u_sideClkDomainCon
troller.u_CdcPgcbClock.u_ismWakePG.PULSE_WIDTH_CHCEK = 0;
```

Either way should work, though it may be better to place the directive within the RTL itself so that it is portable from IP environment to FC also.

The above example illustrates turning off the pulse width check, but the metastability checks can also be turned off in similar manner.

NOTE: One set of common pulse-width check violations relate to the gclock_req_async/sync input to the CDC.

- When the functional domain of the CDC is active (clock is not gated), these signals may assert, de-assert shortly and then assert again without having enough time for the de-assertion to be synchronized into the pgcb_clk domain (which is typically slower than the functional clock).
- For such cases, if the IP design guarantees that for any assertion of the gclock_req_async signal, the de-assertion happens only after gclock_ack_async has asserted, then it is okay to disable the pulse-width violation check on the corresponding double-sync.

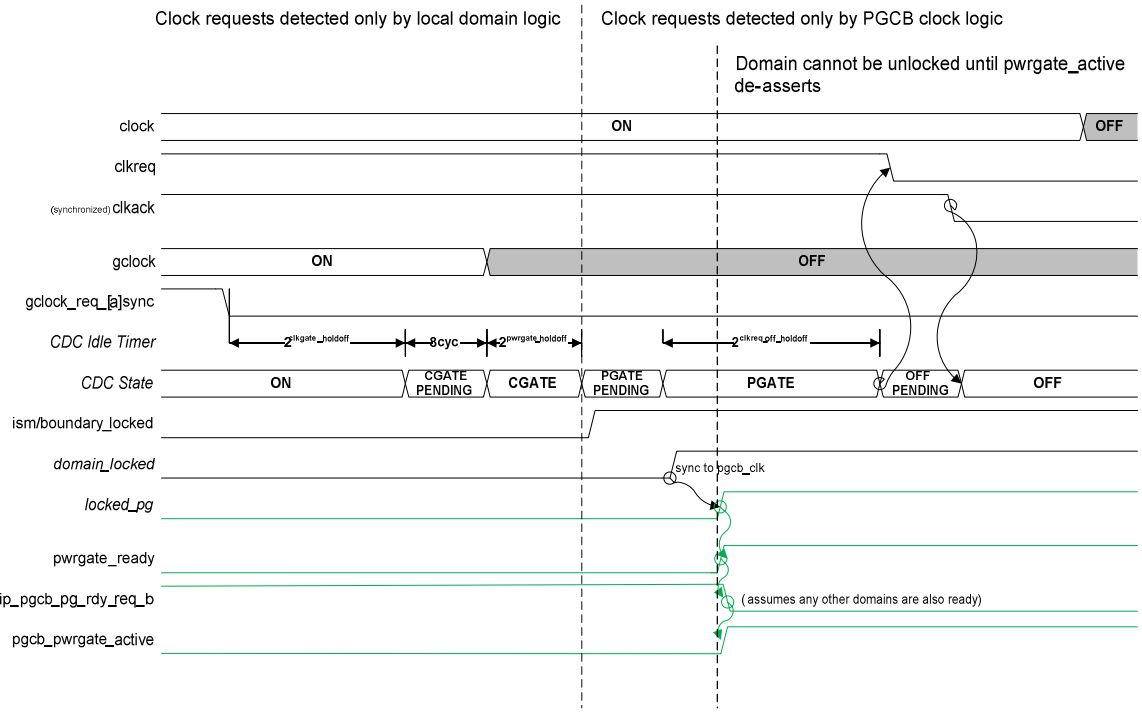
4.2.8.6 SCAN/DFx requirements on reset signals input to the CDC

The CDC now includes SCAN reset bypass muxes on the reset_b and pok_reset_b inputs to the CDC both before and after the reset synchronizer such that the IP does not necessarily need to add these externally (unless they are used in other logic outside of the CDC).

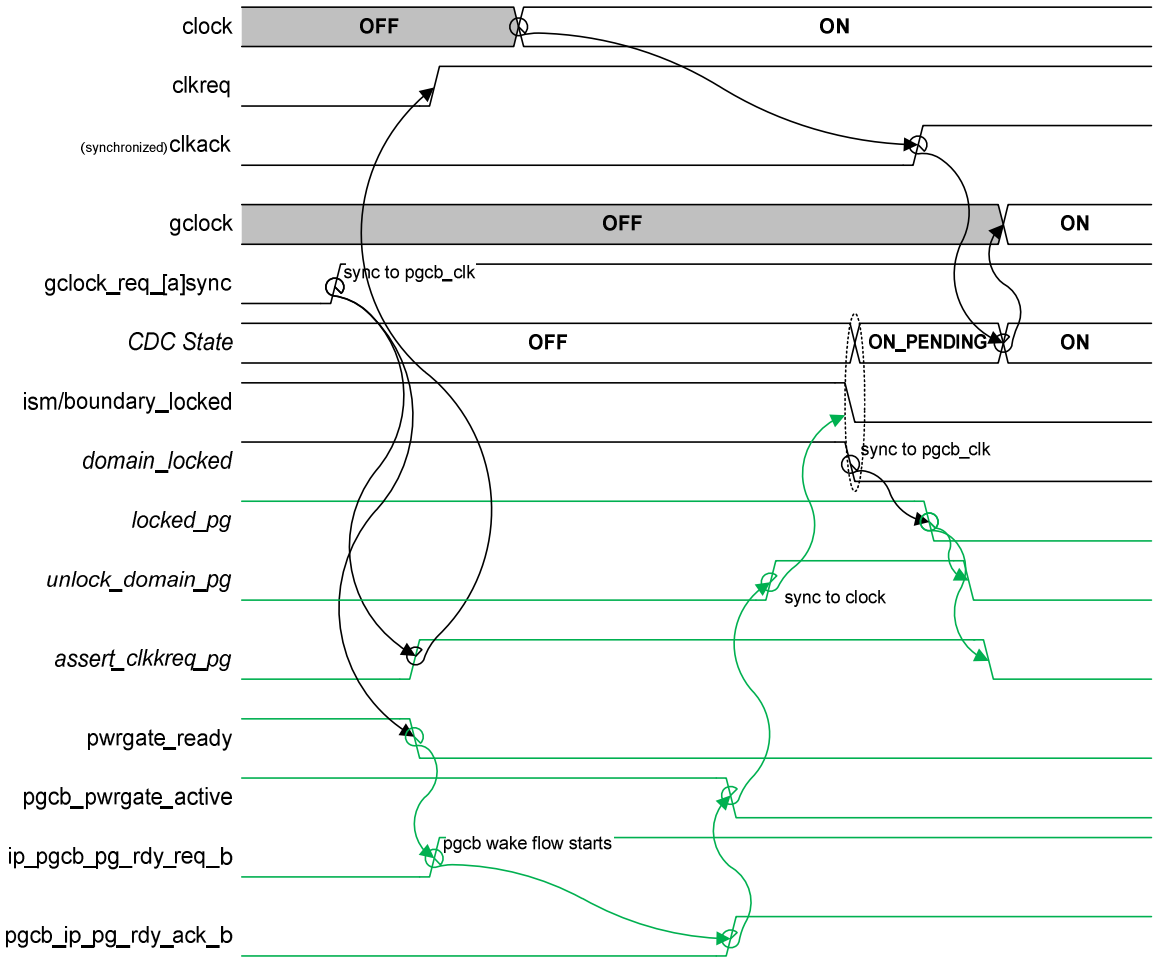
5 Waveforms

In the following waveforms, internal signals are marked with italics and are included only to help where necessary to illustrate the overall flow. Waveforms in green denote pgcb_clk domain signals.

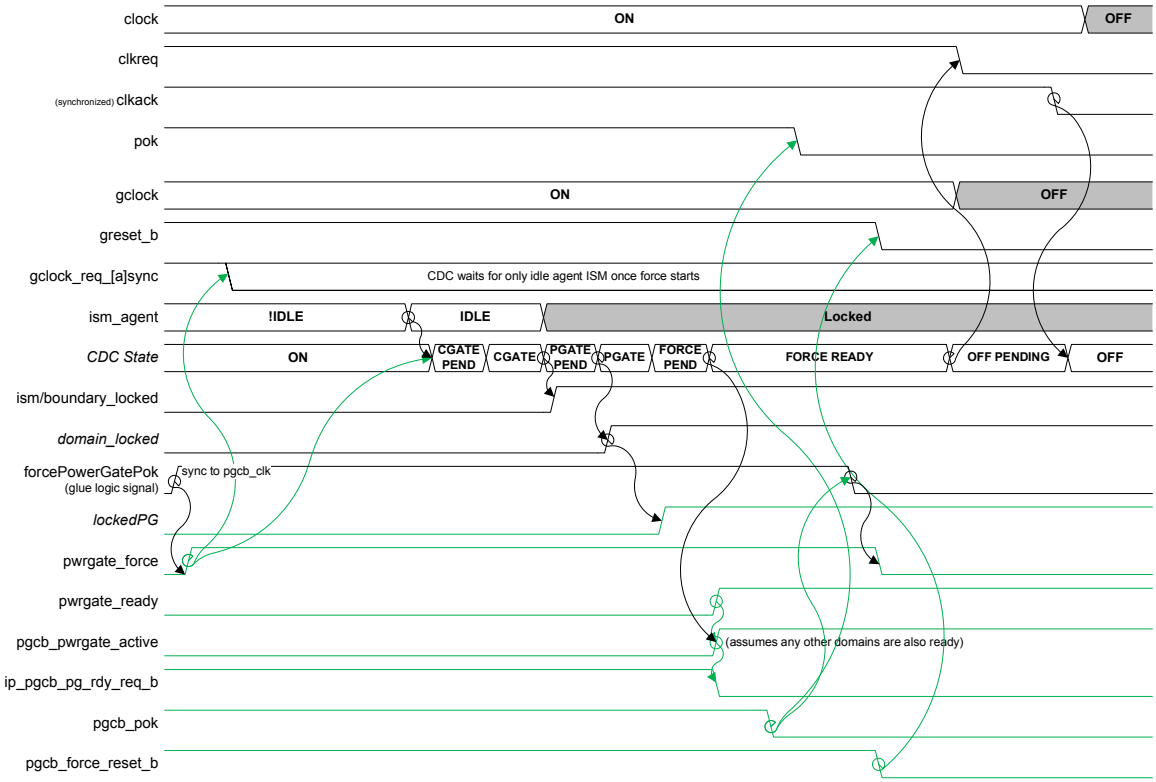
5.1 IP-Accessible Entry



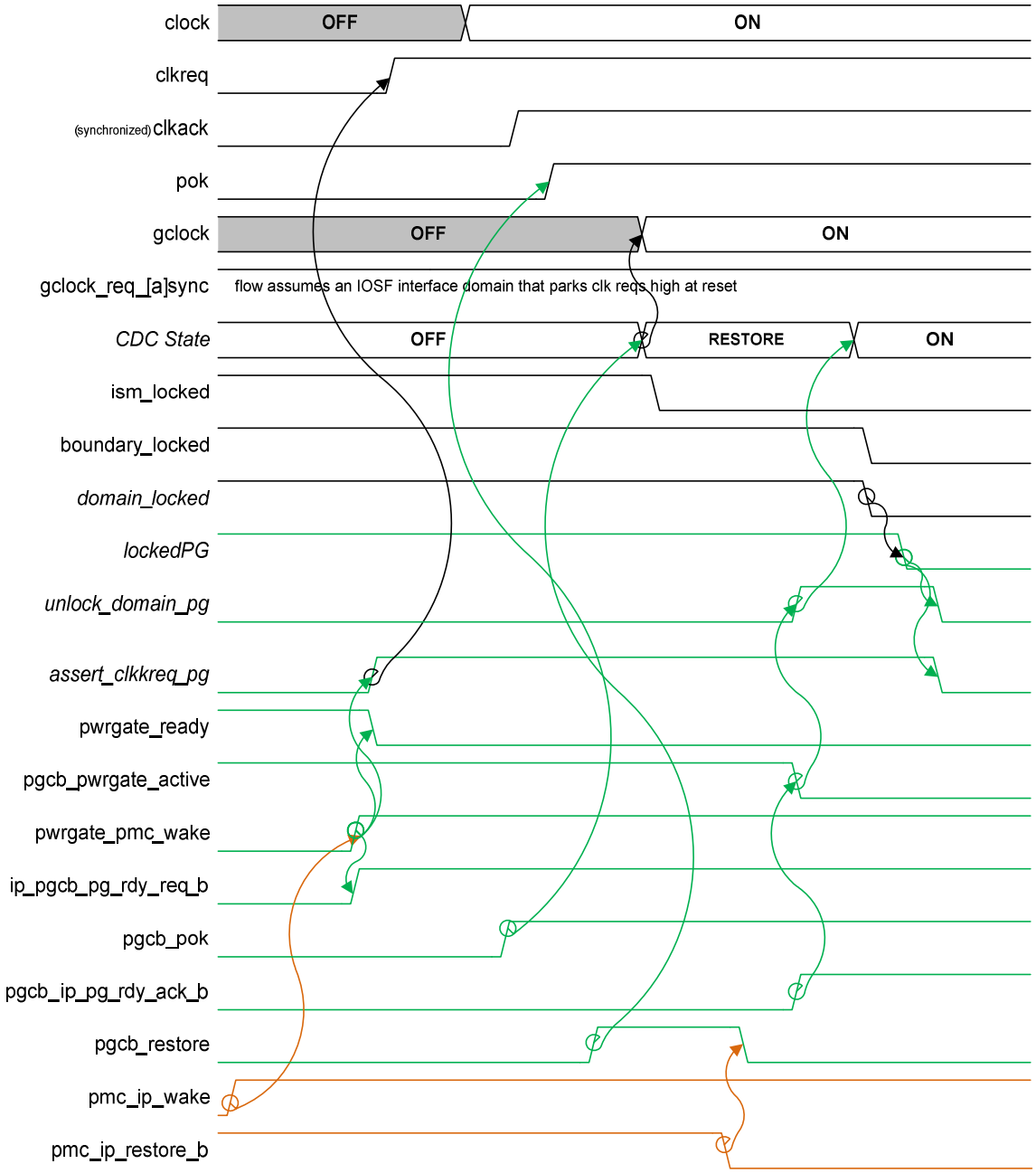
5.2 IP-Accessible Exit



5.3 IP-Inaccessible Entry



5.4 IP-Inaccessible Exit



6 Appendix

6.1 CDC VISA Signals

The latest recommendation is that the integrating IP have the VISA tool automatically insert VISA in the CDC and that the cdc_visa output be ignored. See the provided .sig files under $MODEL_ROOT/tools/visa.

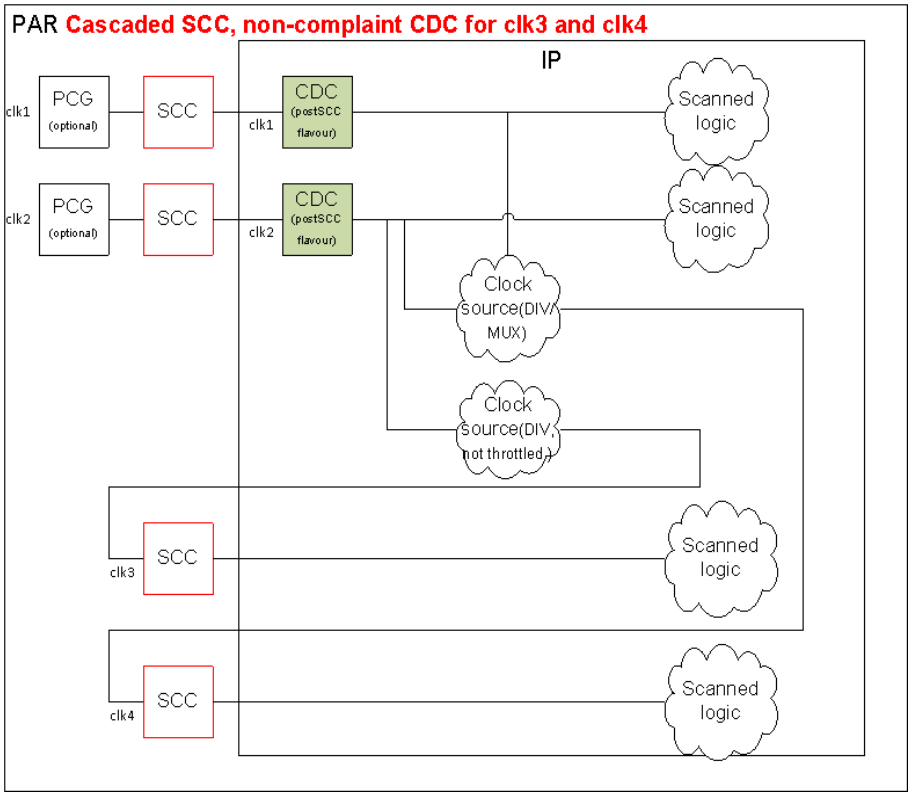
The following table describes the implementation of the 24 bit output vector 'cdc_visa':

Bits	Signal	Clock domain	Description
23:20	Reserved	n-a	Reserved
19	u_CdcPgClock.locked_pg	pgcb_clk	
18	u_CdcPgClock.force_ready_pg	pgcb_clk	
17	u_CdcPgClock.domain_pok_pg	pgcb_clk	
16	u_CdcPgClock.assert_clkreq_pg	pgcb_clk	
15	u_CdcPgClock.unlock_domain_pg	pgcb_clk	
14	u_CdcPgClock.pwrgate_ready	pgcb_clk	
13	u_CdcMainClock.clkreq_start_ok	clock	
12	u_CdcMainClock.gclock_active	clock	
11	u_CdcMainClock.clkreq_hold	clock	
10	u_CdcMainClock.gclock_enable	clock	
9	u_CdcMainClock.gclock_req	clock	
8	u_CdcMainClock.do_force_pgate	clock	
7	u_CdcMainClock.timer_expired	clock	
6:3	u_CdcMainClock.current_state	clock	
2	u_CdcMainClock.ism_wake	clock	
1	u_CdcMainClock.not_idle	clock	
0	u_CdcMainClock.pg_disabled	clock	

6.2 SCAN Considerations for CDC Usage (pre-SCC and post-SCC)

In most cases, the CDC is used in post-SCC (Scan clock controller) within the physical partition in which the SIP is instantiated for any given SOC. However, for the case where a SIP generates local clock(s) from the CDC clock, it results in the issue of cascaded SCCs'. It is a primary requirement of the Scan flow that SCCs are not allowed to be cascaded. An example of this violation case is provided below.

Figure 6-1: Example of Violation of Non-Cascaded SCC Requirement with CDCs



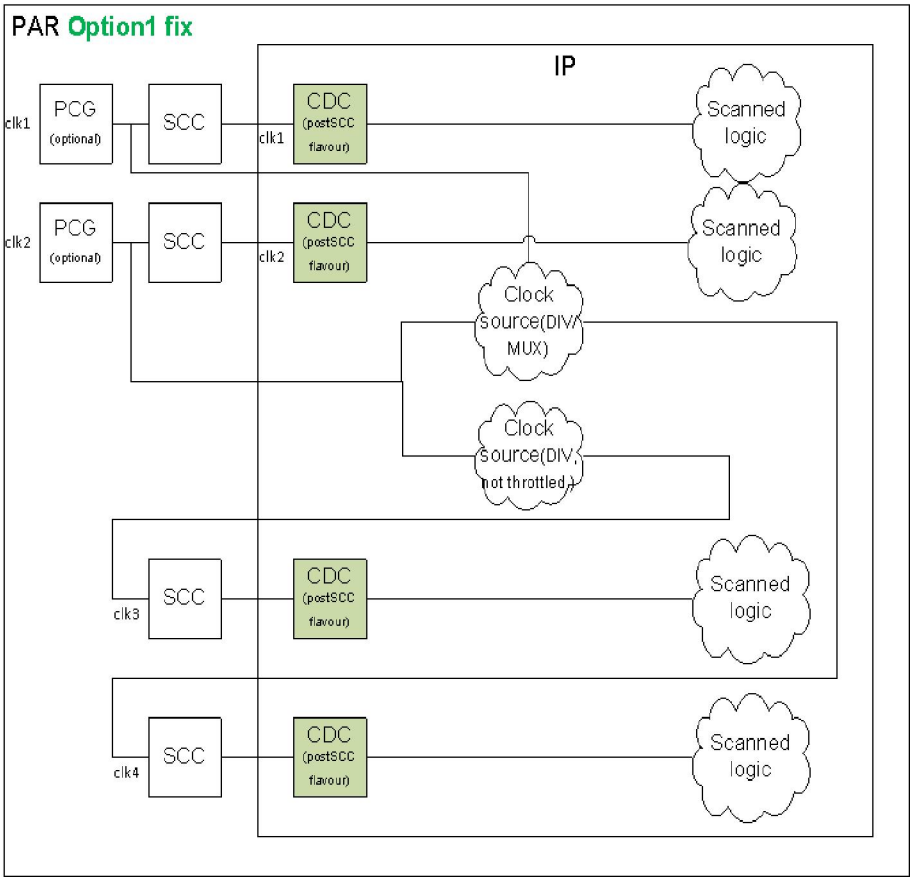
For the above case, clk1 and clk2 are in compliance with Scan requirements, but clk3 and clk4 have the issue of cascaded SCCs'.

There are at least three different options to resolve this issue. Any SIP team that has a clock micro-architecture that may result in cascaded SCCs is requested to work closely with the SCAN/DFX team to pick one of these options.

6.2.1 Option 1: Changes within SIP clocking structure

For this option, the SIP is required to source the clock used in the clock divider/mux logic separately from original clock. Furthermore, the SIP needs to have separate CDC component on this clock, after it has been divided/muxed and passed through an SCC component. This approach is shown below.

Figure 6-2: Changes to Fix Cascaded SCC Issue with SIP Changes



Advantages of this approach:

1. Can use CDC without any pre-SCC support
2. No scan coverage loss
3. No assumption on balancing of pre-SCC and post-SCC clock

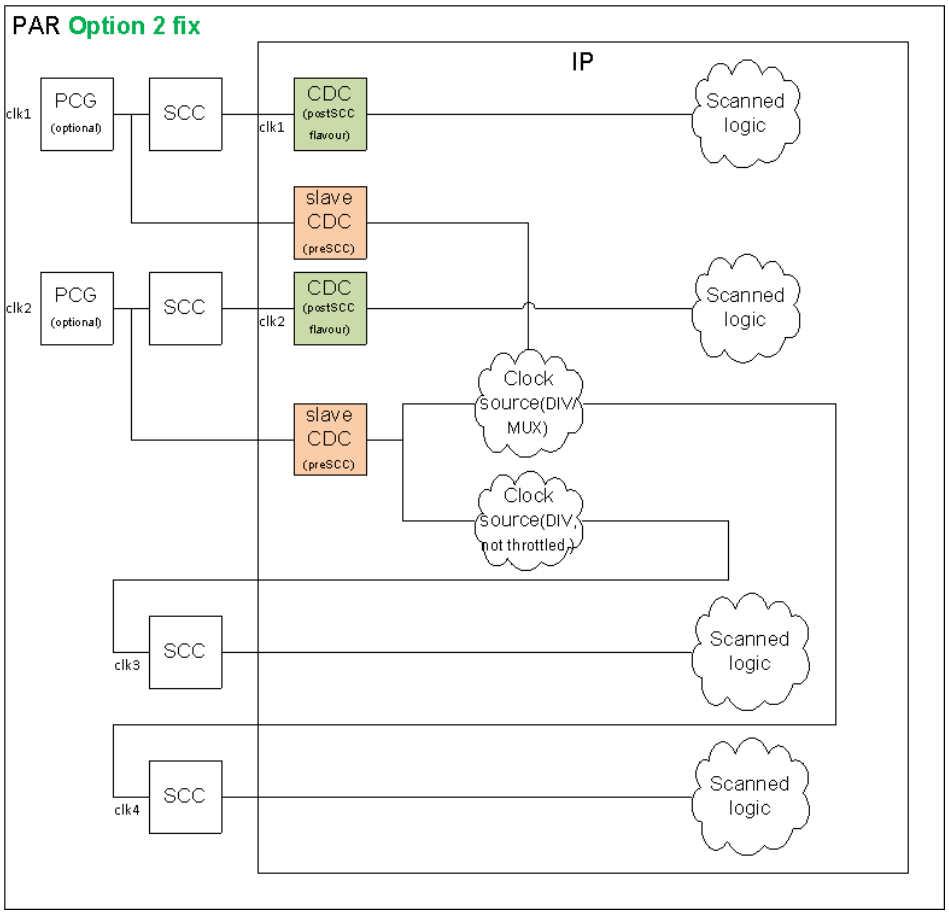
Disadvantages of this approach:

1. SIP needs to have separate CDC for each generated clock
2. SIP may need to add multiple and separate clock ports for source clock
3. All clock source logic moves to pre-SCC domain.

6.2.2 Option 2: Using separate clock-gates for branch of source clock used for generating derivative clocks

For this option, the SIP is required to use separate clock-gate component(s) on the branch(es) of the source clock that is (are) used to generate derivative clock(s). The clock-gate component needs to include support for special SCAN bypass controls (that are applicable to all pre-SCC logic). This component is referred as the "slave CDC" in the figure below.

Figure 6-3: Solution for Cascaded SCCs using "slave CDC"



Advantages of this approach:

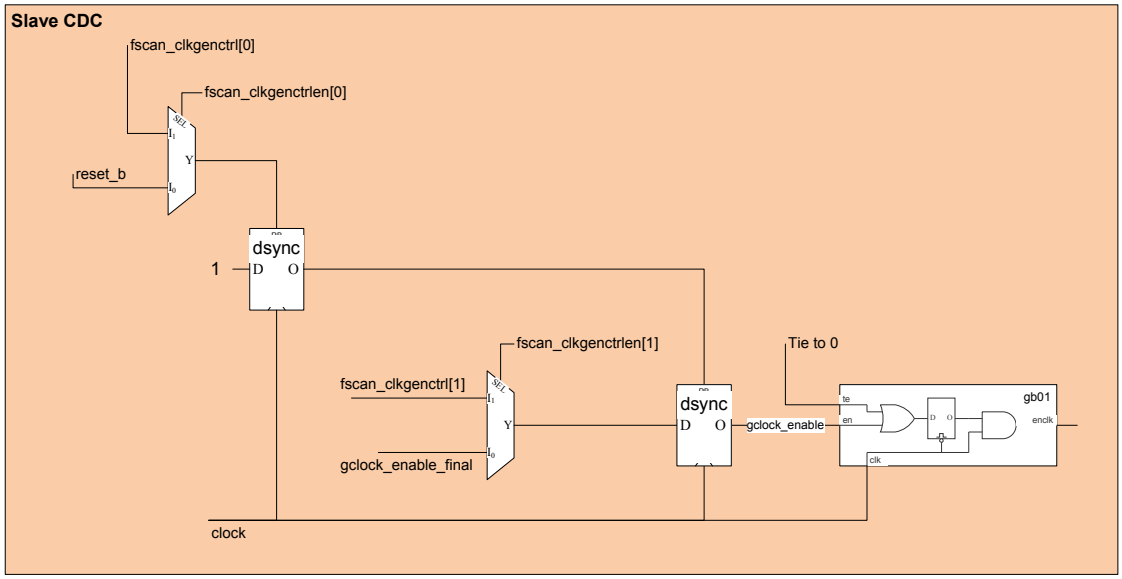
1. Can use CDC without any pre-SCC support
2. No scan coverage loss

Disadvantages of this approach:

1. Requires assumption of balancing of pre-SCC and post-SCC clock
2. SIP may need to add multiple and separate clock ports for source clock
3. Requires separate clock-gate component with special SCAN bypass controls (needed for pre-SCC logic)
4. All clock source logic moves to pre-SCC domain.

NOTE: The CDC provides the output signal "*gclock_enable_final*" to control the "slave CDC" component in the figure above. Any SIP team intending to use this approach should consult the SCAN/DFx team to ensure that all requirements for bypass control for pre-SCC logic are included within the design of the "slave CDC". Reference design of the "slave CDC" is included below (SIP teams should still consult with respective SCAN/DFX teams to ensure this design can be used as-is). Note that the CDC providing the '*gclock_enable_final*' signal may need to be configured with either DSYNC_CG_EN == '0' or '1' and the DSYNC shown in the figure below may also need to be replaced by double-syncs at the mux inputs for the fscan_clkgenctrl* signals.

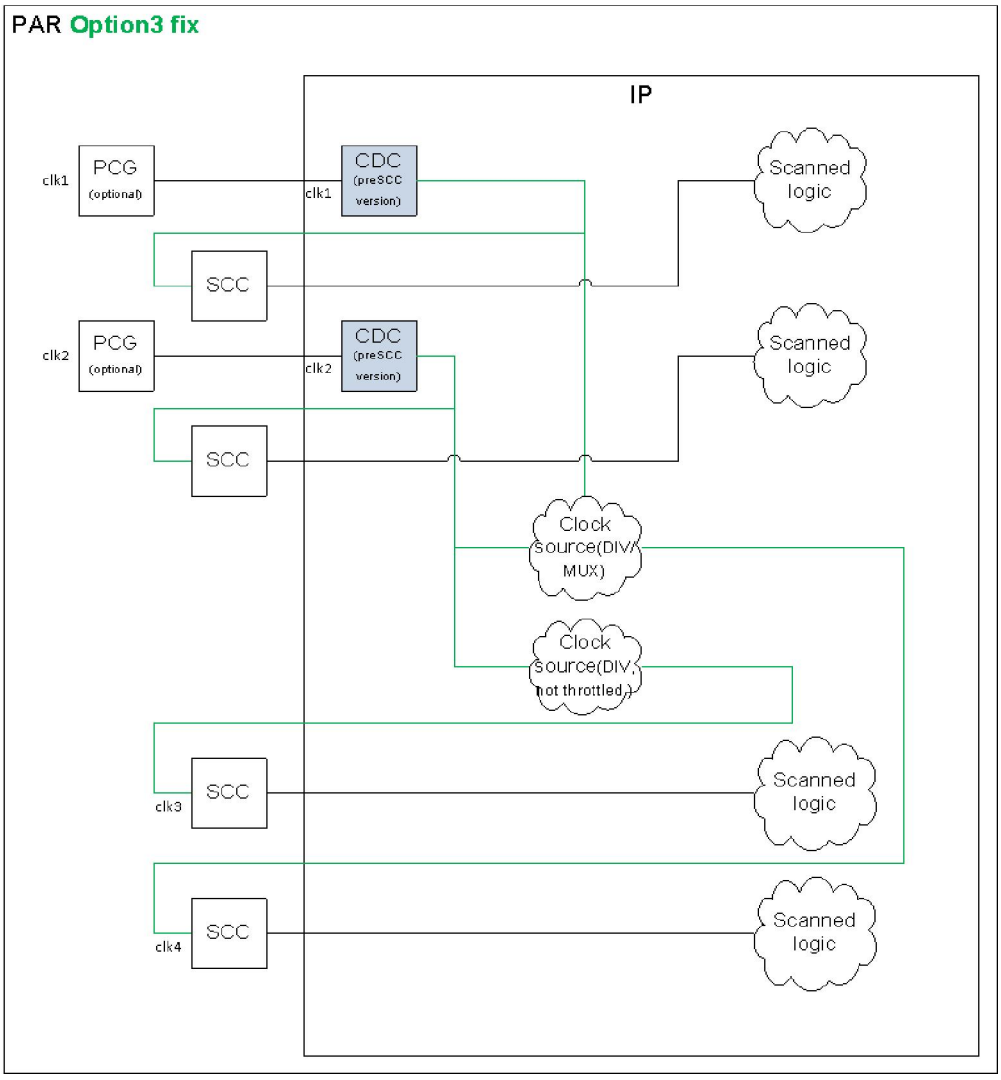
Figure 6-4: Reference Design for "Slave CDC"



6.2.3 Option 3: Changes within CDC to support Pre-SCC usage

For this option, the CDC is used in the SIP in a pre-SCC manner. Therefore, the CDC is required to meet the special SCAN bypass control requirements for pre-SCC logic. However, the SIP team has minimal impact from this approach.

Figure 6-5: Option to Solve Cascaded SCC Issue with CDC parameter PRESCC=='1'



Advantages of this approach:

1. Minimal changes to the SIP
2. No scan coverage loss (assuming CDC FSM is using post-SCC clock – see note below)
3. No need for separate clock-gate component with special bypass support
4. No assumption of balancing of pre-SCC and post-SCC clocks.

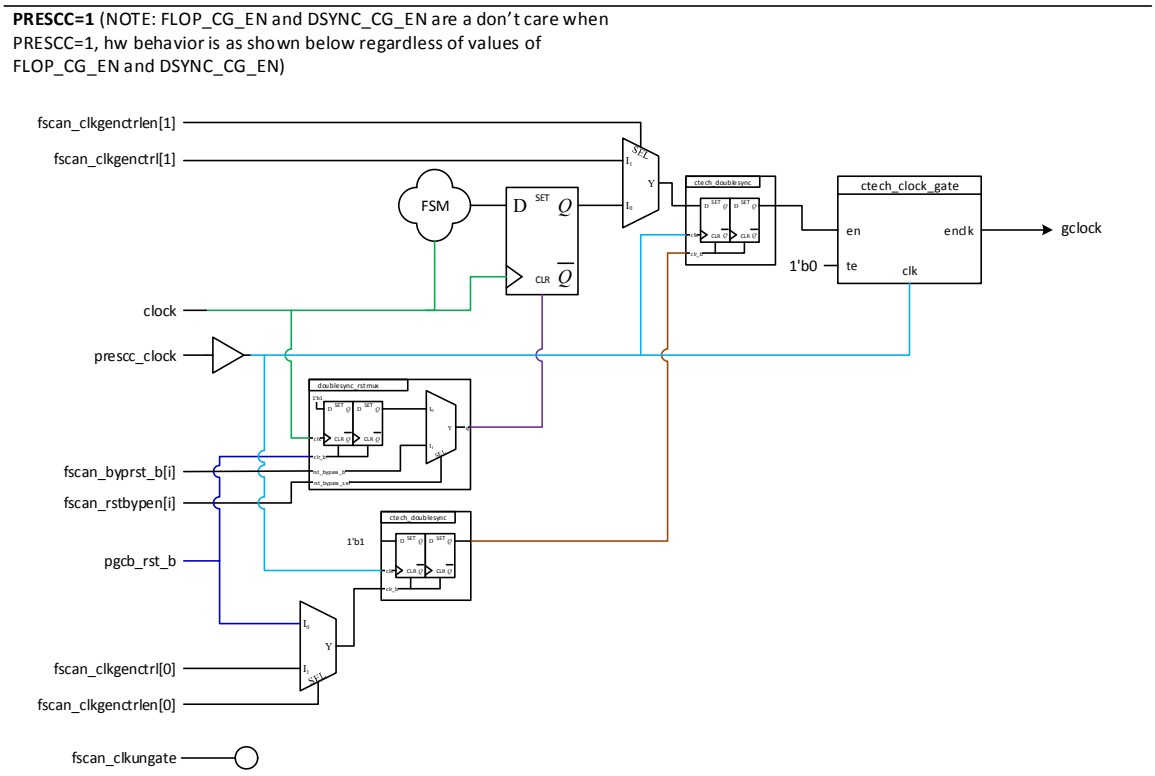
Disadvantages of this approach:

1. All clock source logic moves to pre-SCC domain.

NOTE: The CDC now has separate clock ports for pre-SCC and post-SCC clocks. The prescc_clock input needs to be connected to the pre-SCC clock and the clock input should be connected to the post-SCC clock.

The internal changes to the CDC to support pre-SCC usage are shown in the figure below. Note that the user needs to set the parameter PRESCC to '1'. Also, the design assumes asynchronous relationship between the CDC FSM clock and clock used in the clock-gate within the CDC. This configuration also suffers from potential performance impact because of the increased latency for gating/ungating of the clock (refer to section on "CDC Behavior Details" for further information on options for mitigation of the latency impact).

Figure 6-6: Internal CDC Implementation of Clock-gate Enable When PRESCC == '1'

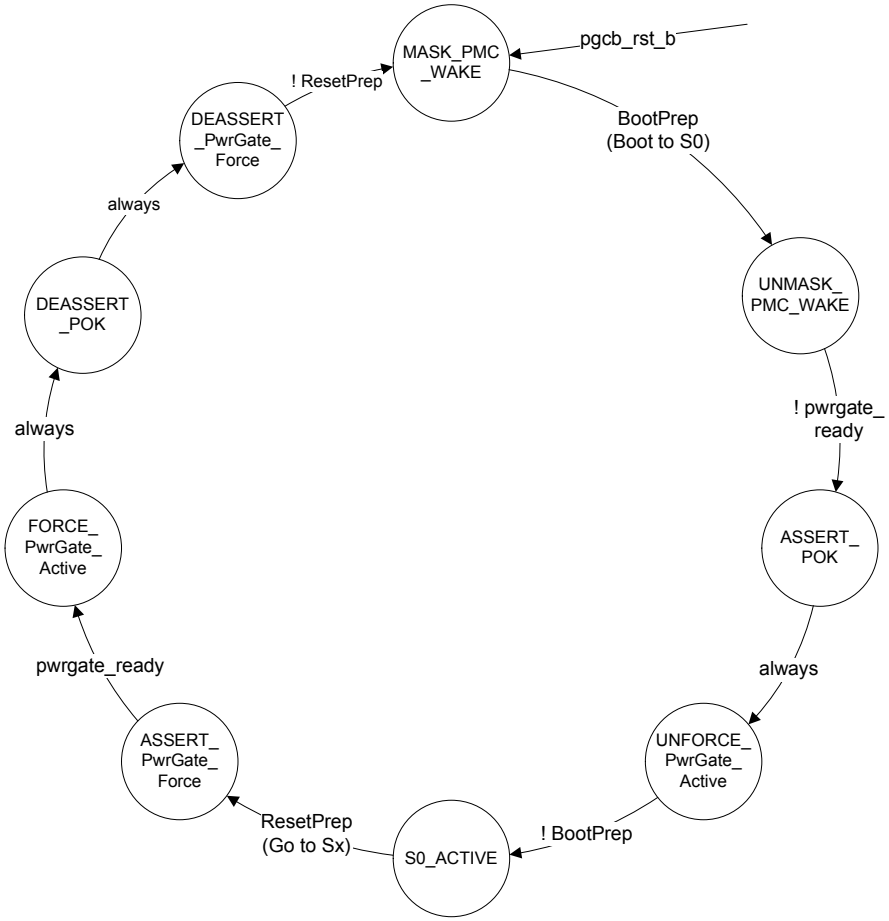


6.3 Example of Control Sequencer for CDC that Operates in S0 Only

NOTE: The sequencer example and other logic depicted here is only for reference purposes, it is not validated and the users are fully responsible for reviewing and validating any design based on this.

The following figure refers to the example of sequencer that may be used to control a CDC within a dual-space device, where the CDC's functional clock domain is operational only in S0 (the clock is not even available in Sx system states). For this case, the trigger used for entry into active state (S0) is a BootPrep message from PMC to the SIP and the trigger for exit from the active state isa ResetPrep message from PMC. However, a SIP could as well use some other forms of triggers.

Figure 6-7: Example Sequencer for CDCs for Clocks Not Available in all IP Active States



STATE/outputs	pwrgate_force	FRC_pwrgate_active	pgcb_pok	MASK_pmc_pg_wake	ResetPrepAck	BootPrepAck
MASK_PMC_WAKE	0	1	0	1	0	0
UNMASK_PMC_WAKE	0	1	0	0	0	0
ASSERT_POK	0	1	1	0	0	0
UNFORCE_PwrGate_Active	0	0	1	0	0	1
STATE_ACTIVE	0	0	1	0	0	0
ASSERT_PwrGate_Force	1	0	1	0	0	0
FORCE_PwrGate_Active	1	1	1	0	0	0
DEASSERT_POK	1	1	0	0	0	0
DEASSERT_PwrGate_Force	0	1	0	0	1	0

Using the sequencer above, some additional glue logic may be used as suggested below.

Figure 6-8: Example Aggregation Glue Logic for Use with CDC Sequencer