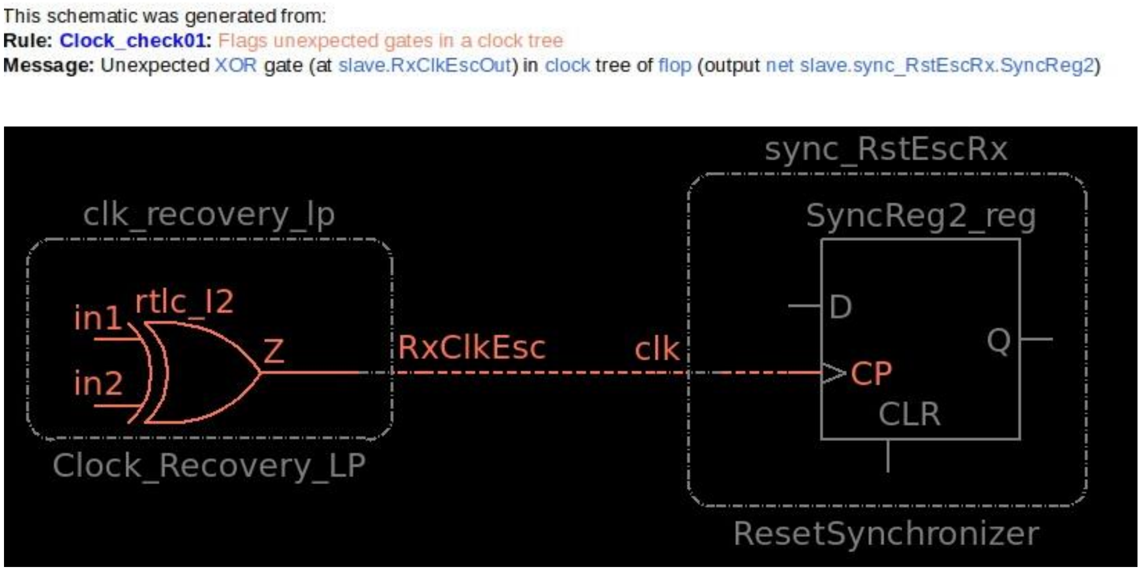
Spyglass CDC

1- Clock_reset_integrity Goal

- **Clock_check01**

  Potential glitch in clock tree due to unexpected gates in clock tree



This schematic was generated from:
Rule: Clock_check01: Flags unexpected gates in a clock tree
Message: Unexpected XOR gate (at slave.RxClkEscOut) in clock tree of flop (output net slave.sync_RstEscRx.SyncReg2)

  ➢ Solution:

  We need this gate to recover clock from the data. Waive this warning.

- **Clock_check04**

  Both positive and negative edges of clocks used in the same deign.

  ➢ Solution:

  We need to use both edges in the low power blocks to achieve target functionality.
  Waive this warning.

2- Cdc_verify_struct Goal

- **Ar_unsync01**

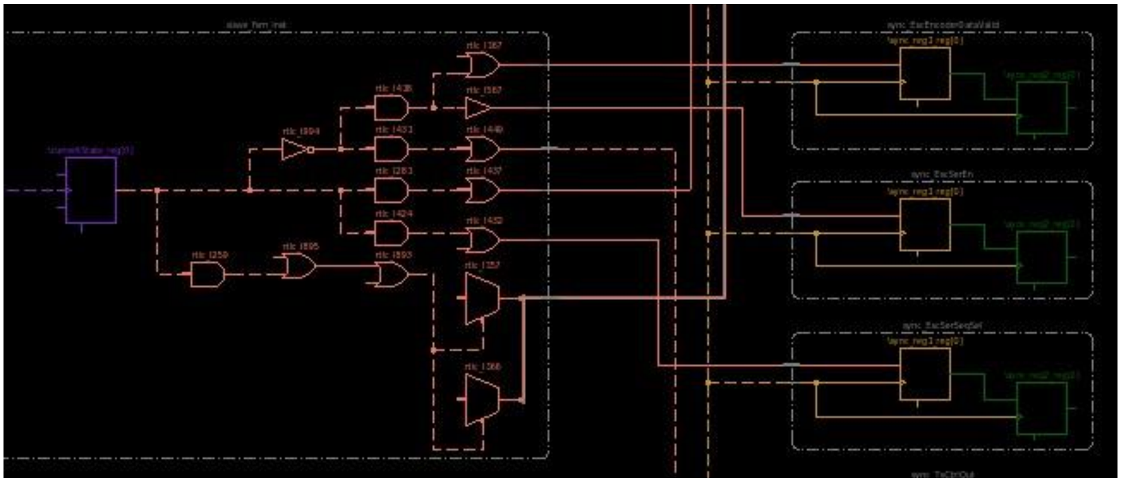
  Reports unsynchronized reset signals in the design.

  - Solution:

    Adding reset synchronization to the different domains in the design.

- **Ac_glitch03**

  Reports clock domain crossings subject to glitches.



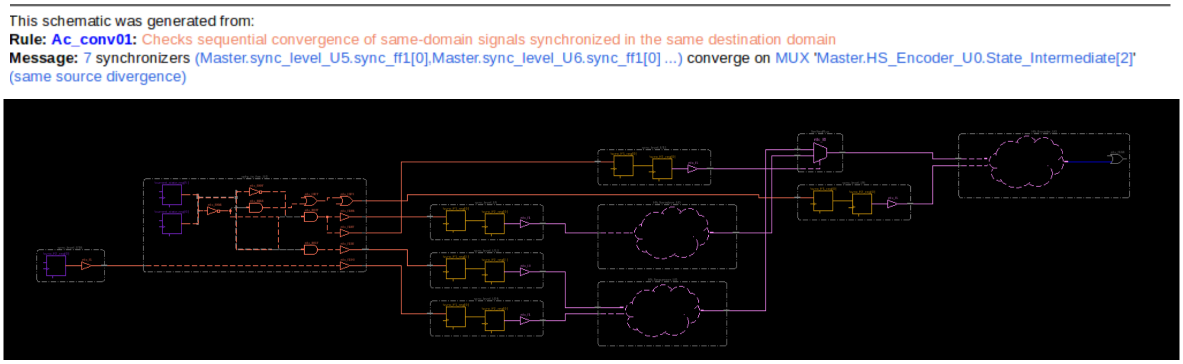
  Outputs from combinational circuit input to synchronizers which cause glitches.

  - Solution:

    Adding FFs to each output of the block before the synchronizers.

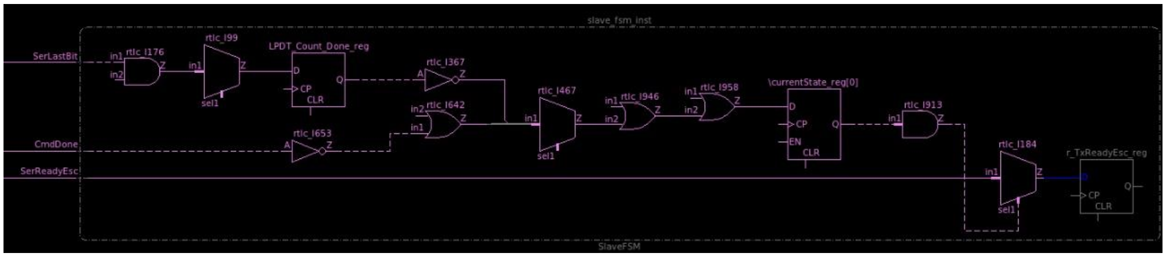
- **Ac_conv01**

  Checks for sequential convergence of properly synchronized control crossings.

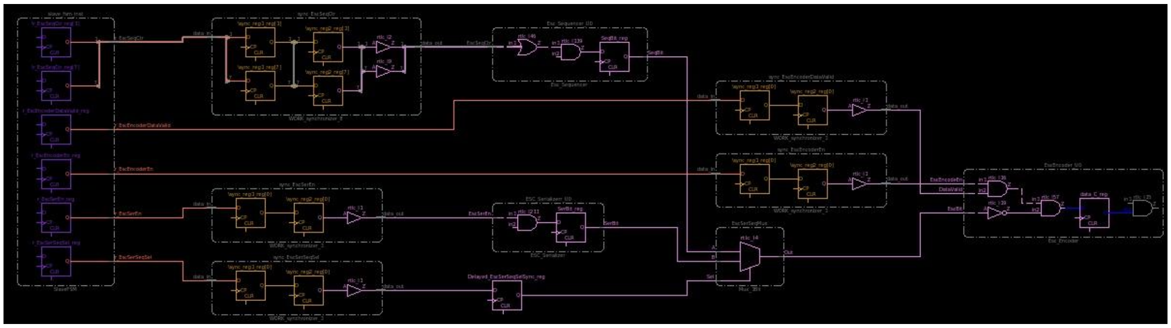


This schematic was generated from:
Rule: **Ac_conv01:** Checks sequential convergence of same-domain signals synchronized in the same destination domain
Message: 7 synchronizers (Master.sync_level_U5.sync_ff1[0],Master.sync_level_U6.sync_ff1[0] ...) converge on MUX 'Master.HS_Encoder_U0.State_Intermediate[2]' (same source divergence)

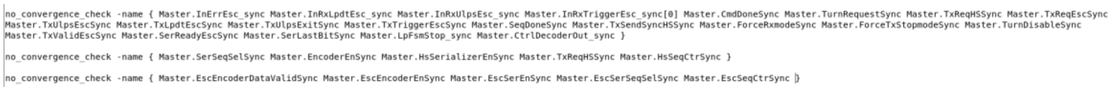
This schematic was generated from:
Rule: Ac_conv01: Checks sequential convergence of same-domain signals synchronized in the same destination domain
Message: 3 synchronizers (slave.sync_CmdDone.sync_reg1[0],slave.sync_SerReadyEsc.sync_reg1[0] ...) converge on MUX 'slave.slave_fsm_inst.TxReadyEsc'



This schematic was generated from:
Rule: Ac_conv01: Checks sequential convergence of same-domain signals synchronized in the same destination domain
Message: 10 synchronizers (slave.sync_EscSerEn.sync_reg1[0],slave.sync_EscEncoderEn.sync_reg1[0] ...) converge on flop 'slave.EscEncoder_U0.data_C'



- Solution:
  Since these signals are not functionally related, convergence checks are not required

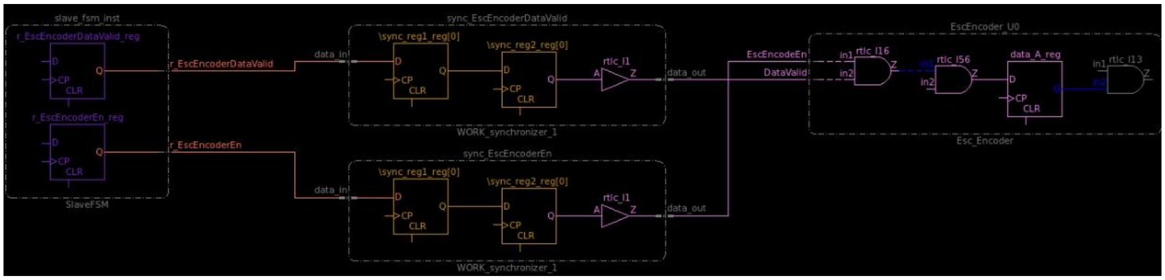


```
no_convergence_check -name { Master.InErrEsc_sync Master.InRxLpdtEsc_sync Master.InRxUlpsEsc_sync Master.InRxTriggerEsc_sync[0] Master.CmdDoneSync Master.TurnRequestSync Master.TxReqHSSync Master.TxReqEscSync
Master.TxUlpsEscSync Master.TxLpdtEscSync Master.TxUlpsExitSync Master.TxTriggerEscSync Master.SeqDoneSync Master.TxSendSyncHSSync Master.ForceRxmodeSync Master.ForceTxStopmodeSync Master.TurnDisableSync
Master.TxValidEscSync Master.SerReadyEscSync Master.SerLastBitSync Master.LpFsmStop_sync Master.CtrlDecoderOut_sync }

no_convergence_check -name { Master.SerSeqSelSync Master.EncoderEnSync Master.HsSerializerEnSync Master.TxReqHSSync Master.HsSeqCtrSync }

no_convergence_check -name { Master.EscEncoderDataValidSync Master.EscEncoderEnSync Master.EscSerEnSync Master.EscSerSeqSelSync Master.EscSeqCtrSync }
```

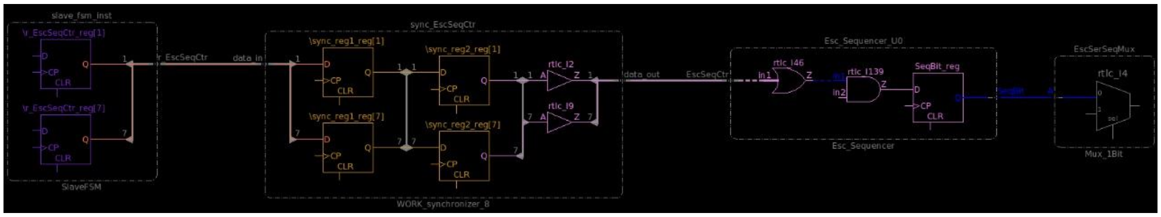
- **Ac_conv02**
  Checks for combinational convergence of properly synchronized control crossings.

  Output signals from the same block synchronized to a block in another domain.

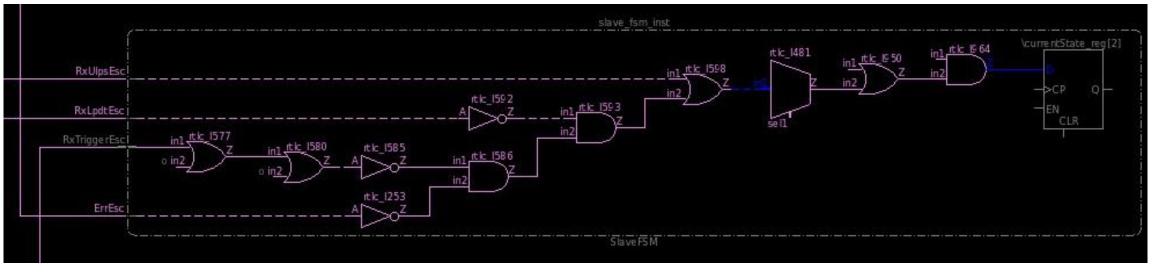
This schematic was generated from:
Rule: Ac_conv02: Checks combinational convergence of same-domain signals synchronized in the same destination domain
Message: 2 synchronizers (slave.sync_EscEncoderEn.sync_reg1[0],slave.sync_EscEncoderDataValid.sync_reg1[0]) converge on flop 'slave.EscEncoder_U0.data_A'. Gray encoding check: 'DISABLED'



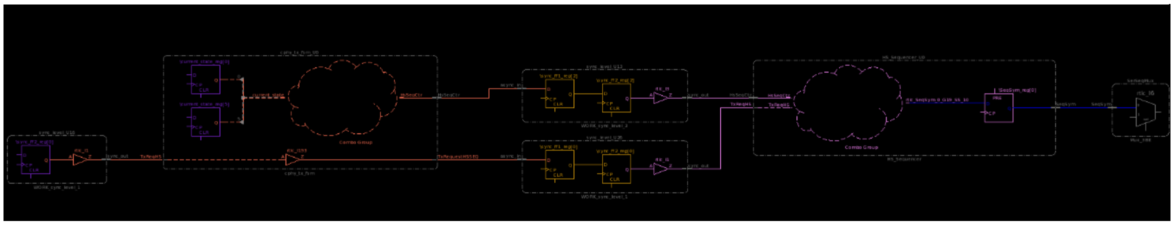
This schematic was generated from:
Rule: Ac_conv02: Checks combinational convergence of same-domain signals synchronized in the same destination domain
Message: 6 synchronizers (slave.sync_EscSeqCtr.sync_reg1[1:0],slave.sync_EscSeqCtr.sync_reg1[7:4]) converge on flop 'slave.SeqBit'. Gray encoding check: 'DISABLED'



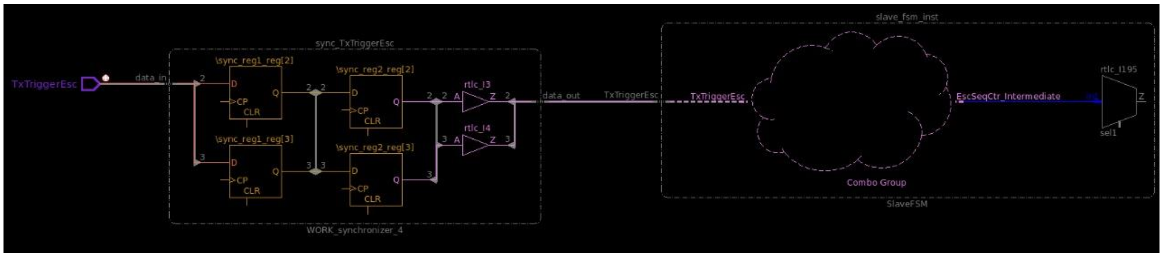
This schematic was generated from:
Rule: Ac_conv02: Checks combinational convergence of same-domain signals synchronized in the same destination domain
Message: 4 synchronizers (slave.sync_InRxTriggerEsc.sync_reg1[0],slave.sync_InRxUlpsEsc.sync_reg1[0] ...) converge on combinational gate 'slave.slave_fsm_inst.nextState[2]'. Gray encoding check: 'DISABLED'



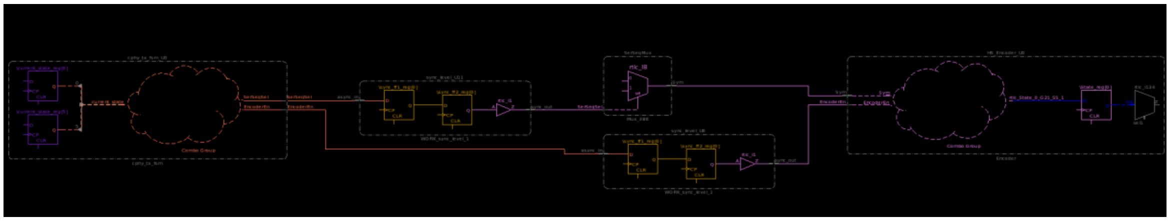
This schematic was generated from:
Rule: Ac_conv02: Checks combinational convergence of same-domain signals synchronized in the same destination domain
Message: 4 synchronizers (Master.sync_level_U13.sync_ff1[2:0],Master.sync_level_U26.sync_ff1[0]) converge on flop 'Master.SeqSym[0]' (same source divergence). Gray encoding check: 'DISABLED'



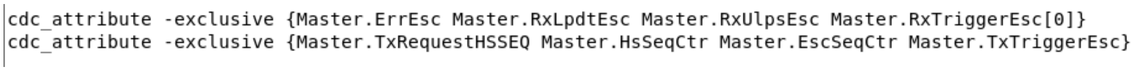
This schematic was generated from:
Rule: Ac_conv02: Checks combinational convergence of same-domain signals synchronized in the same destination domain
Message: 4 synchronizers (slave.sync_TxTriggerEsc.sync_reg1[3:0]) converge on combinational gate 'slave.slave_fsm_inst.EscSeqCtr_Intermediate[0]'. Gray encoding check: 'DISABLED'



This schematic was generated from:
Rule: **Ac_conv02:** Checks combinational convergence of same-domain signals synchronized in the same destination domain
Message: 2 synchronizers (Master.sync_level_U6.sync_ff1[0],Master.sync_level_U11.sync_ff1[0]) converge on flop 'Master.HS_Encoder_U0.State[0]' (same source divergence). Gray encoding check: 'DISABLED'

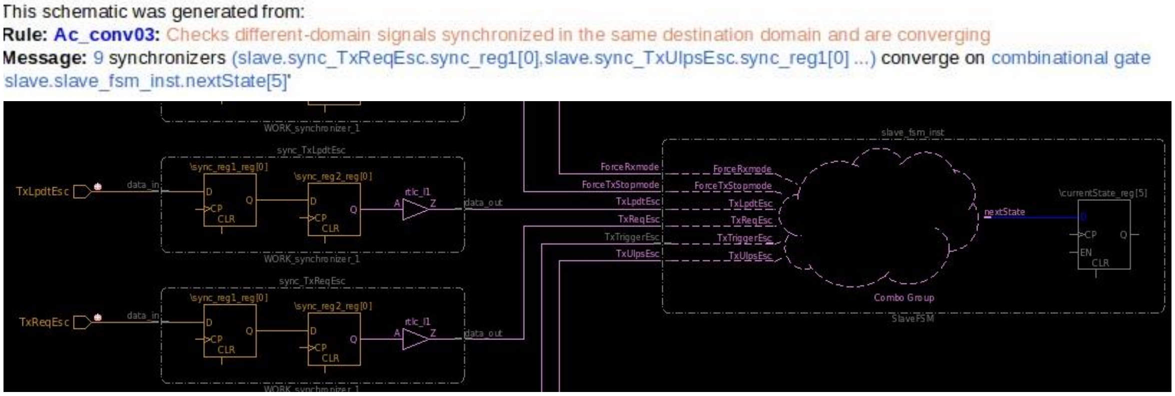


- Solution:
  For each warning, we make sure in the design that these signals are grey encoded.

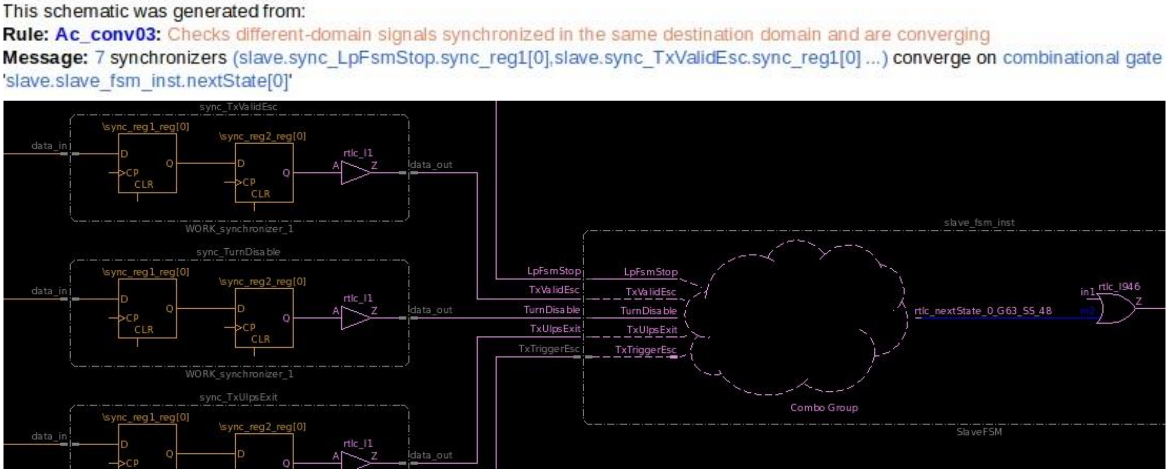


```
cdc_attribute -exclusive {Master.ErrEsc Master.RxLpdtEsc Master.RxUlpsEsc Master.RxTriggerEsc[0]}
cdc_attribute -exclusive {Master.TxRequestHSSEQ Master.HsSeqCtr Master.EscSeqCtr Master.TxTriggerEsc}
```

- **Ac_conv03**
  Checks different-domain signals synchronized in the same destination domain and are converging.

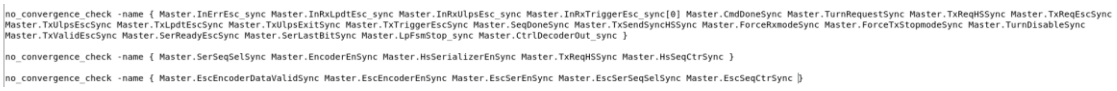


This schematic was generated from:
Rule: **Ac_conv03:** Checks different-domain signals synchronized in the same destination domain and are converging
Message: 9 synchronizers (slave.sync_TxReqEsc.sync_reg1[0],slave.sync_TxUlpsEsc.sync_reg1[0] ...) converge on combinational gate slave.slave_fsm_inst.nextState[5]'



This schematic was generated from:
Rule: **Ac_conv03:** Checks different-domain signals synchronized in the same destination domain and are converging
Message: 7 synchronizers (slave.sync_LpFsmStop.sync_reg1[0],slave.sync_TxValidEsc.sync_reg1[0] ...) converge on combinational gate 'slave.slave_fsm_inst.nextState[0]'

  - Solution:
    For each warning, the signals involved are unrelated and do not change simultaneously; therefore, a convergence check is not required

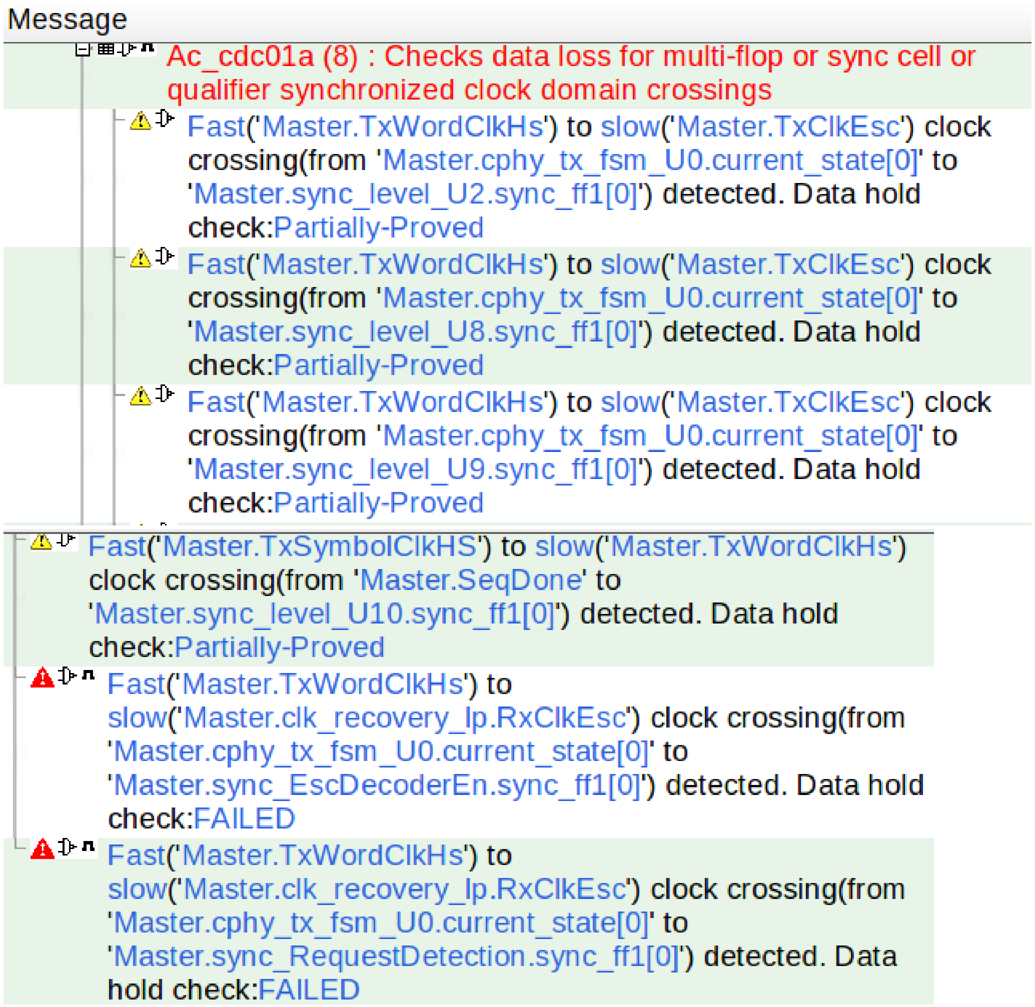


```
no_convergence_check -name { Master.InErrEsc_sync Master.InRxLpdtEsc_sync Master.InRxUlpsEsc_sync Master.InRxTriggerEsc_sync[0] Master.CmdDoneSync Master.TurnRequestSync Master.TxReqHSSync Master.TxReqEscSync
Master.TxUlpsEscSync Master.TxLpdtEscSync Master.TxUlpsExitSync Master.TxTriggerEscSync Master.SeqDoneSync Master.TxSendSyncHSSync Master.ForceRxmodeSync Master.ForceTxStopmodeSync Master.TurnDisableSync
Master.TxValidEscSync Master.SerReadyEscSync Master.SerLastBitSync Master.LpFsmStop_sync Master.CtrlDecoderOut_sync }

no_convergence_check -name { Master.SerSeqSelSync Master.EncoderEnSync Master.HsSerializerEnSync Master.TxReqHSSync Master.HsSeqCtrSync }

no_convergence_check -name { Master.EscEncoderDataValidSync Master.EscEncoderEnSync Master.EscSerEnSync Master.EscSerSeqSelSync Master.EscSeqCtrSync }
```

3- Cdc_verify Goal

- **Ac_ccd01a**

  Checks data-loss for multi-flop or sync cell or qualifier synchronized clock domain crossings



```
Message
Ac_cdc01a (8) : Checks data loss for multi-flop or sync cell or
qualifier synchronized clock domain crossings
  ⚠ Fast('Master.TxWordClkHs') to slow('Master.TxClkEsc') clock
    crossing(from 'Master.cphy_tx_fsm_U0.current_state[0]' to
    'Master.sync_level_U2.sync_ff1[0]') detected. Data hold
    check:Partially-Proved
  ⚠ Fast('Master.TxWordClkHs') to slow('Master.TxClkEsc') clock
    crossing(from 'Master.cphy_tx_fsm_U0.current_state[0]' to
    'Master.sync_level_U8.sync_ff1[0]') detected. Data hold
    check:Partially-Proved
  ⚠ Fast('Master.TxWordClkHs') to slow('Master.TxClkEsc') clock
    crossing(from 'Master.cphy_tx_fsm_U0.current_state[0]' to
    'Master.sync_level_U9.sync_ff1[0]') detected. Data hold
    check:Partially-Proved
⚠ Fast('Master.TxSymbolClkHS') to slow('Master.TxWordClkHs')
  clock crossing(from 'Master.SeqDone' to
  'Master.sync_level_U10.sync_ff1[0]') detected. Data hold
  check:Partially-Proved
⚠ Fast('Master.TxWordClkHs') to
  slow('Master.clk_recovery_lp.RxClkEsc') clock crossing(from
  'Master.cphy_tx_fsm_U0.current_state[0]' to
  'Master.sync_EscDecoderEn.sync_ff1[0]') detected. Data hold
  check:FAILED
⚠ Fast('Master.TxWordClkHs') to
  slow('Master.clk_recovery_lp.RxClkEsc') clock crossing(from
  'Master.cphy_tx_fsm_U0.current_state[0]' to
  'Master.sync_RequestDetection.sync_ff1[0]') detected. Data
  hold check:FAILED
```

- Solution:

  For all these signals, we ensure in the design that each signal is held high or low long enough to be reliably captured by the slow clock domain without data loss.

  So, waive this rule.