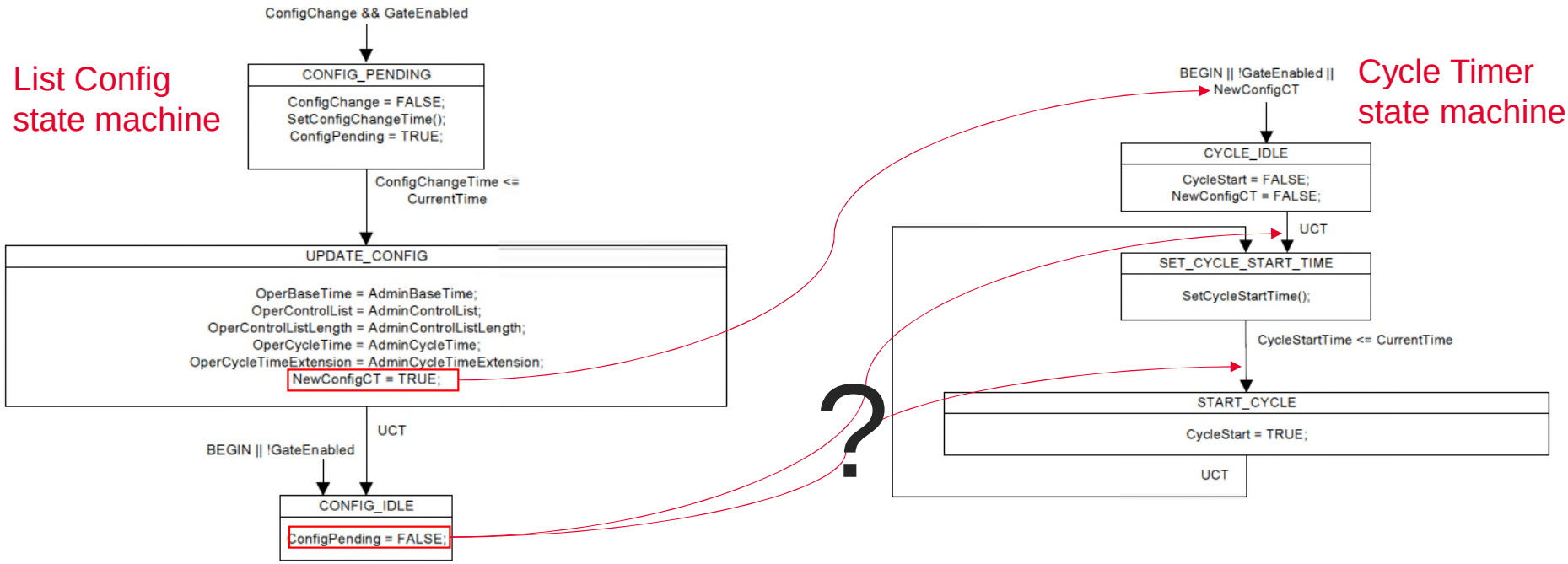
Race condition in 802.1Q-2018 between List Config state machine (clause 8.6.9.3) and Cycle Timer state machine (clause 8.6.9.1)

Alon Regev	APRIL 15, 2021
Email: alon.regev@keysight.com	

Race Condition



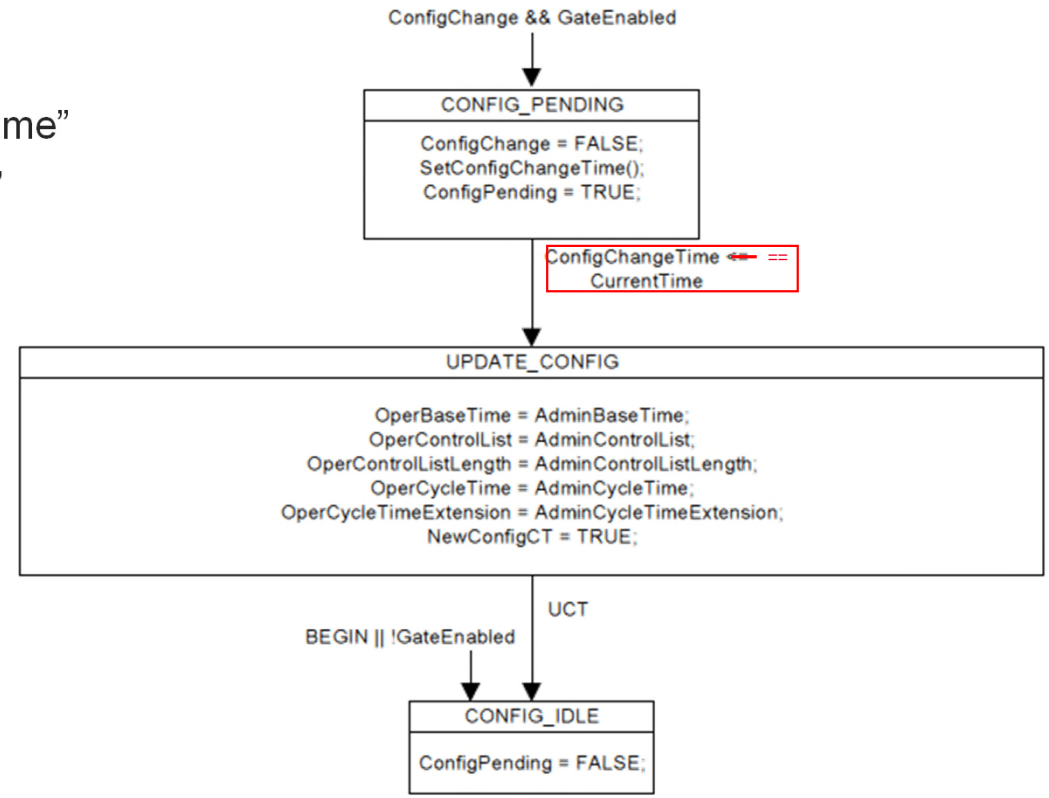
- In the List Config state machine (802.1Q-2018 clause 8.6.9.3), upon a ConfigChange (when GateEnabled is TRUE) ConfigPending is set to TRUE in the CONFIG_PENDING state, remains TRUE in the UPDATE_CONFIG state machine and is then set to FALSE in the CONFIG_IDLE state.
- Also in the List Config stat machine, in the UPDATE_CONFIG state, NewConfigCT is set to TRUE. NewConfigCT being TRUE triggers the Cycle Timer state machine (802.1Q-2018 clause 8.6.9.1) to transition to the CYCLE_IDLE state, which then transitions to the SET_CYCLE_START_TIME (UCT). In the SET_CYCLE_START_TIME state, the SetCycleStartTime() procedure determines which rules should be taken.
- Unfortunately, after the List Config state machine changes to the UPDATE_CONFIG state, it is not clear if ConfigPending will be set to FALSE before or after the Cycle Timer state machine gets to the SET_CYCLE_START_TIME state, hence the race condition.

Affect on SetCycleStartTime() calculation

- This race condition only makes a difference to the outcome of the SetCycleStartTime() calculation when:
  - A dynamic schedule change is done (applying a new gate control list while another one is already running)
  - In the List Config state machine, the transition from CONFIG_PENDING to UPDATE_CONFIG occurs when (ConfigChangeTime < CurrentTime)
    - the problem doesn't occur if the transition occurs when ConfigChangeTime is equal to CurrentTime
- This is the behavior of the SetCycleStart() calculation after the List Config state machine under the two cases:
  - if the Cycle Timer state machine is run before ConfigPending is set to FALSE:
    - ConfigPending is TRUE
    - "ConfigChangeTime <= (CurrentTime + OperCy-cleTime + OperCycleTimeExtension)" must be true as ConfigChangeTime <= CurrentTime
      - this was required in the transition from the CONFIG_PENDING to the UPDATE_CONFIG in the List Config state machine
    - Therefore, the SetCycleStart() will use rule "d)" and set CycleStartTime = ConfigChangeTime
  - if the Cycle Timer state machine is run after ConfigPending is set to FALSE:
    - ConfigPending is FALSE
    - At this point, CurrentTime >= ConfigChangeTime >= OperBaseTime (ConfigChangeTIme is set >= AdminBaseTime in the SetConfigChangeTime() function; OperBaseTime was set AdminBaseTime in the UPDATEC_CONFIG state of the List Config state machine; and CurrentTime >= ConfigChangeTime as this was required in the transition from the CONFIG_PENDING to the UPDATE_CONFIG in the List Config state machine)
    - The question is whether CurrentTime > OperBaseTime or CurrentTime == OperBaseTime:
      - If (ConfigPending = FALSE, and OperBaseTime >= CurrentTime)
        - CycleStartTime = OperBaseTime = AdminBaseTime
      - If (ConfigPending = FALSE, and OperBaseTime < CurrentTime)
        - CycleStartTime = (OperBaseTime + N*OperCycleTime), where N is the smallest integer for which CycleStartTime >= CurrentTime
    - If CurrentTime > OperBaseTime (which will occur if the transition from CONFIG_PENDING to UPDATE_CONFIG in the List Config state machine occurs when ConfigChangeTime < CurrentTime) then the cycle will only start N*OperCycleTime after OperBaseTime essentially not starting a cycle (and not running any gates) for N*OperCycleTime

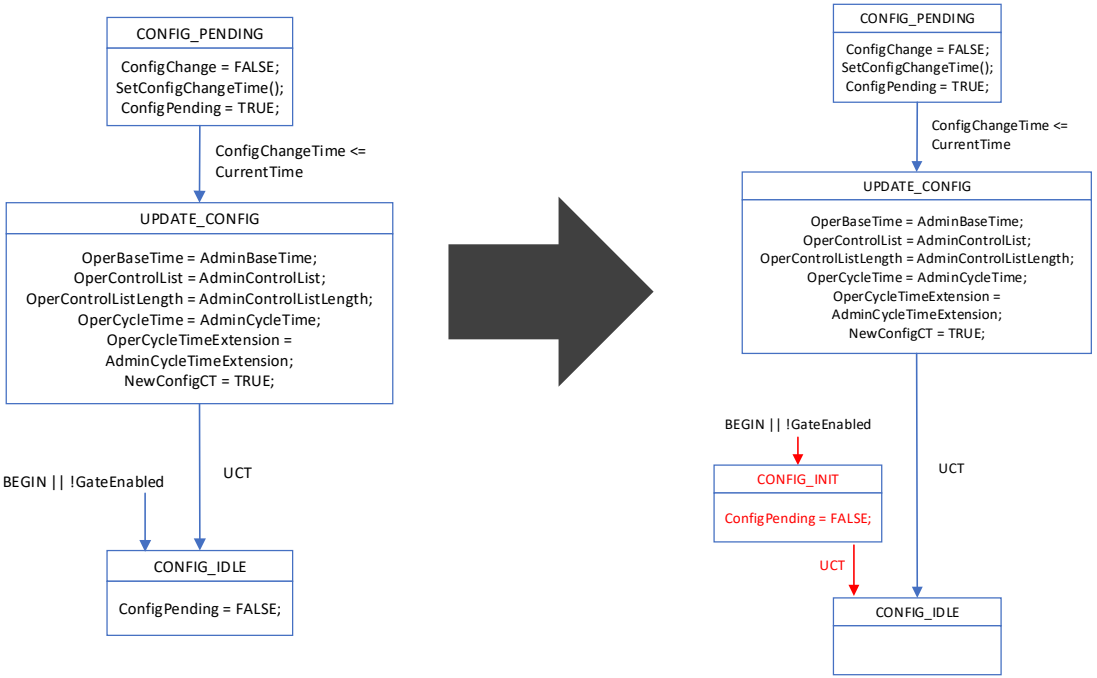
Potential Solution #1

- In the List Config state machine (clause 8.6.9.3, Figure 8-18):
  - Replace “ConfigChangeTime <= CurrentTime”
  - With “ConfigChangeTime == CurrentTime”



Potential Solution #2 – part 1

- In Clause 8.6.9.3 , Figure 8-18 (List Config State Machine):
  - Add a new state named “CONFIG_INIT”
    - This state will contain the “ConfigPending = FALSE;”
  - Add a global transition from “BEGIN || !GateEnabled” to the new CONFIG_INIT state
  - Remove the global transition from “BEGIN || !GateEnabled” to the CONFIG_IDLE state
  - Add an UCT transition from the CONFIG_INIT state to the CONFIG_IDLE state
  - Remove the “ConfigPending = FALSE;” from the CONFIG_IDLE state



Potential Solution #2 – part 2

- In Clause 8.6.9.1.1 (SetCycleStartTime()), section “d)”
  - Following “CycleStartTime = ConfigChangeTime”, add a line containing “set ConfigPending = FALSE”

d) If:
ConfigPending = TRUE, and
ConfigChangeTime <= (CurrentTime + OperCycleTime + OperCycleTimeExtension)
Then:
CycleStartTime = ConfigChangeTime
set ConfigPending = FALSE