802.1 maintenance item 0319: 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)

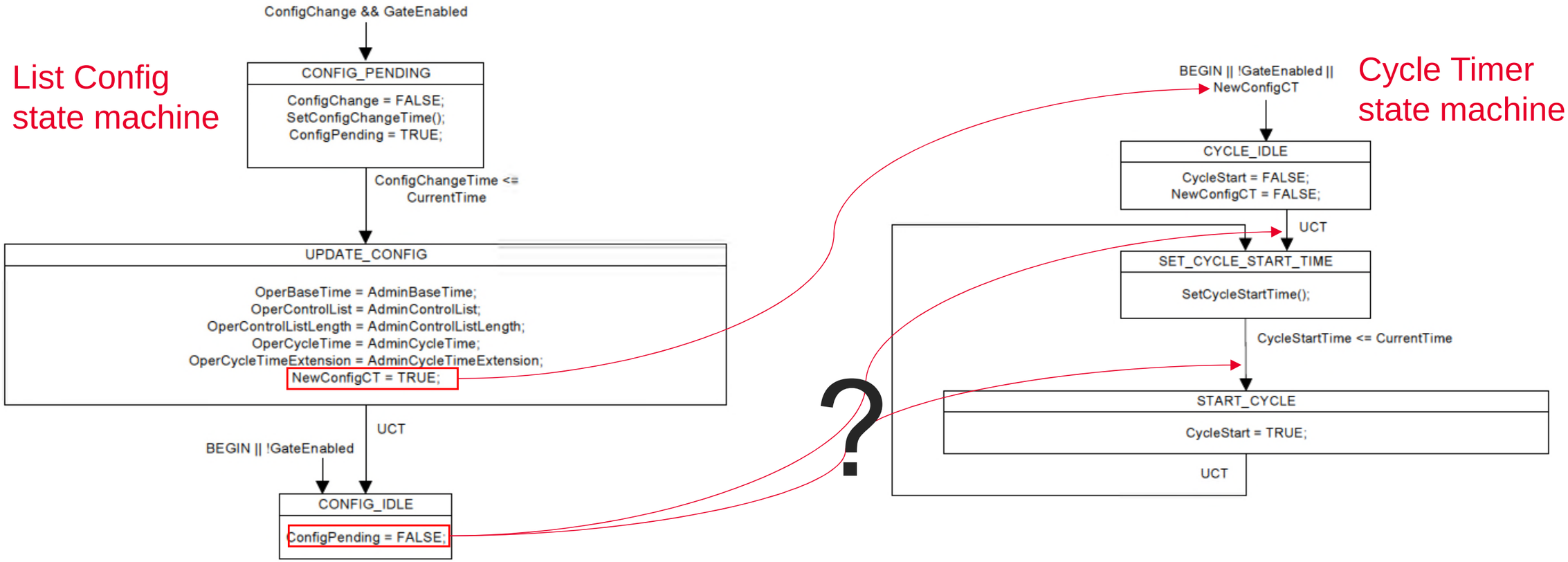
KEYSIGHT TECHNOLOGIES

Alon Regev

Email: alon.regev@keysight.com

AUG 11, 2021

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

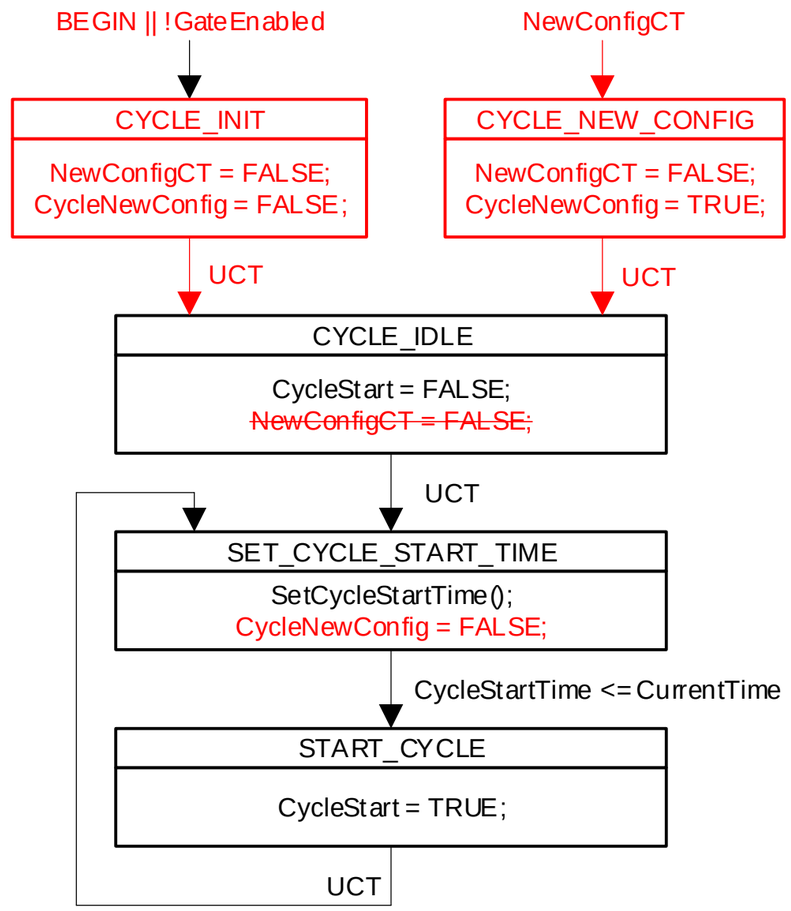
Proposed solution #1 overview

- Currently, configPending is reset to FALSE without knowing if the new config has been applied
- This solution leaves the setting of configPending in the List Config state machine but moves the reset of configPending from the List Config state machine to the Cycle Timer state machine.
- This involves changes to fig 8-18, 8-19, and 8-21 as well as text in clause 8.6.9.1.1.

- Details follow in slides 6-9
  - All references are to https://www.ieee802.org/1/files/private/q-rev-drafts/d1/802-1Q-rev-d1-0.pdf

Proposed solution #2 overview

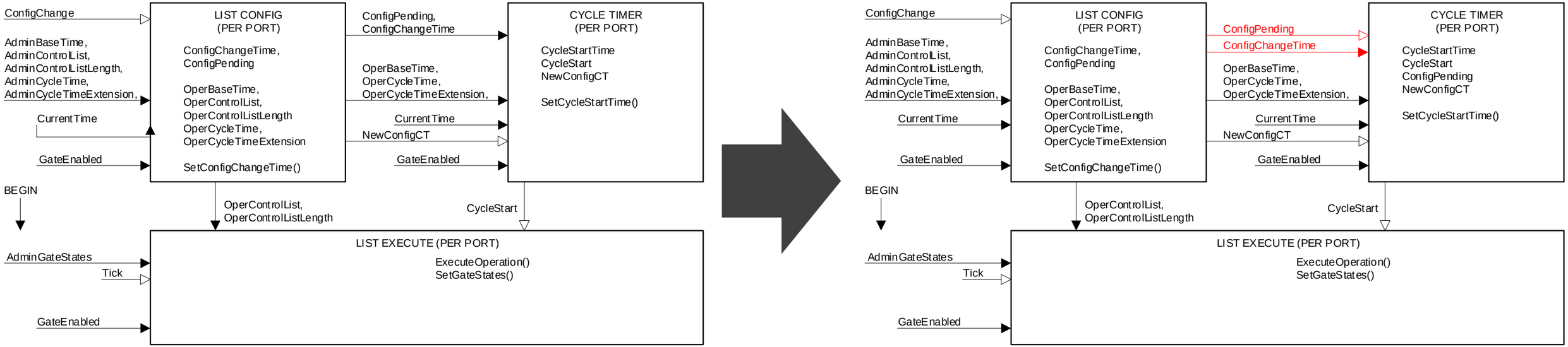
- Currently, configPending is reset to FALSE without knowing if the new config has been applied
  - But the information is available in another variable: NewConfigCT passed by the List Config state machine to the Cycle Timer state machine
- Instead of making changes to multiple state machines, a change to only the Cycle Timer state machine is proposed, where:
  - We track whether the state machine is triggered by NewConfigCT using a new variable (CycleNewConfig)
  - the SetCycleStartTime() procedure is modified to use (configPending || CycleNewConfig) avoiding the race condition



- Details are provided in slides 10-11 below
  - All references are to https://www.ieee802.org/1/files/private/q-rev-drafts/d1/802-1Q-rev-d1-0.pdf

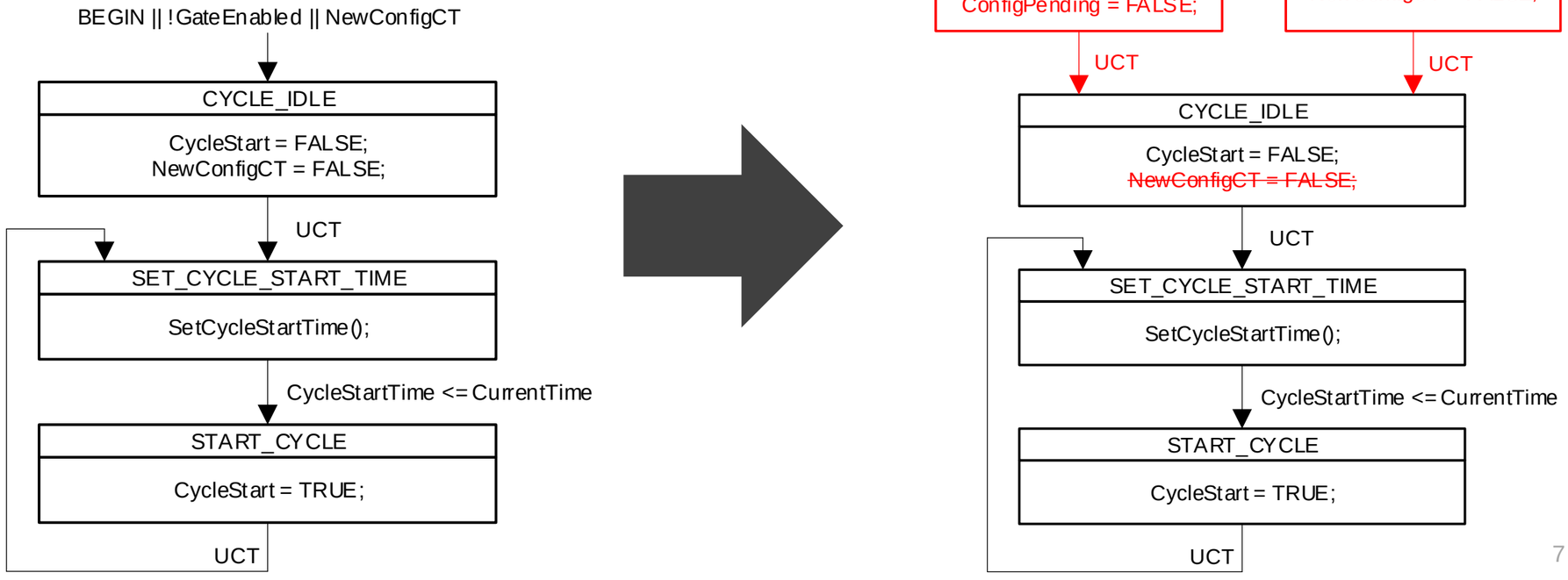
Potential Solution #1 – part 1

- In Clause 8.6.9.3 , Figure 8-18 (Scheduled traffic state machines—overview and relationships):
  - On the arrow with the text “ConfigPending, ConfigChangeTime”, change the text to “ConfigPending” and change the arrow shape to be one with the outline only (no fill).
  - Add an arrow underneath the ConfigPendingArrow starting at the “LIST CONFIG” box and ending at the “CYCLE TIMER” box with the text “ConfigChangeTime”, no arrow at the LIST CONFIG SIDE, and a filled in arrow at the CYCLE TIMER side.



Proposed solution #1 – part 2

- In Clause 8.6.9.1 , Figure 8-19 (Cycle Timer State Machine):
  - Remove the global transition from “BEGIN || !GateEnabled || NewConfigCT” to CYCLE_IDLE
  - Add a new state named “CYCLE_INIT”
    - This state will contain the lines “NewConfigCT = FALSE;” and “ConfigPending = FALSE;”
  - Add a global transition from “BEGIN || !GateEnabled” to the new CYCLE_INIT state
  - Add a new state named “CYCLE_NEW_CONFIG”
    - This state will contain the line “NewConfigCT = FALSE;”
  - Add a global transition from “NewConfigCT” to the new CYCLE_INIT state
  - Add an UCT transition from the CYCLE_INIT state to the CYCLE_IDLE state
  - Add an UCT transition from the CYCLE_NEW_CONFIG state to the CYCLE_IDLE state
  - Remove the line “NewConfigCT = FALSE;” from the CYCLE_IDLE state



Potential Solution #1 – part 3

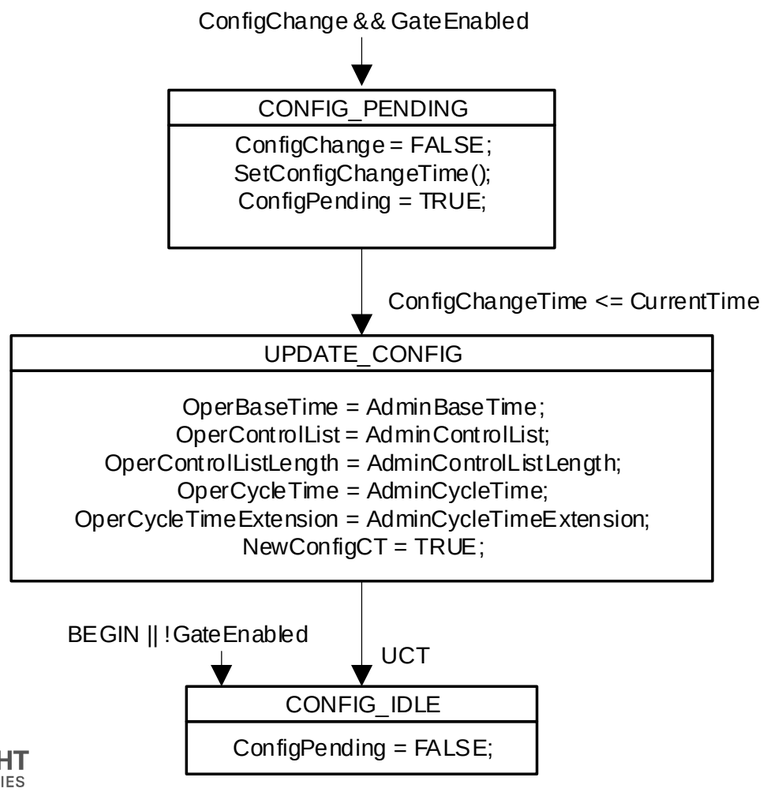
- In Clause 8.6.9.1.1 (SetCycleStartTime()), section “d)”
  - Following “CycleStartTime = ConfigChangeTime”, add a line containing “set ConfigPending = FALSE”

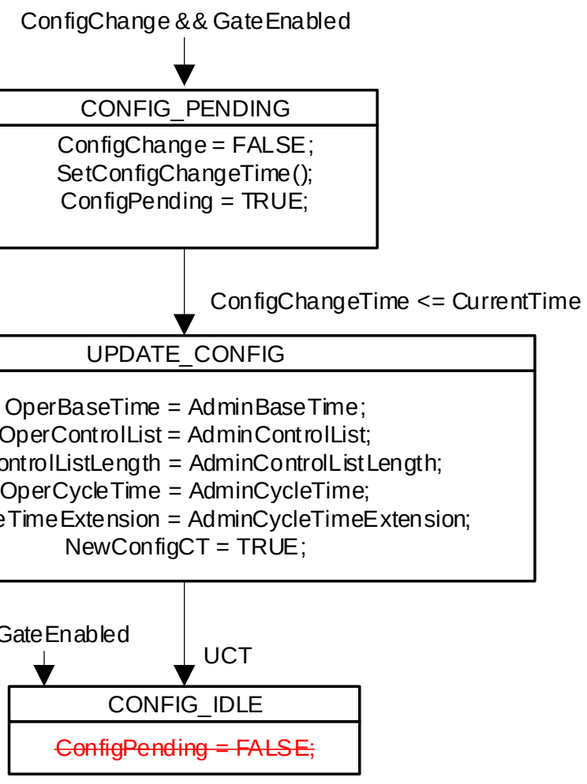
d) If:
ConfigPending = TRUE, and
ConfigChangeTime <= (CurrentTime + OperCycleTime + OperCycleTimeExtension)
Then:
CycleStartTime = ConfigChangeTime
set ConfigPending = FALSE

Potential Solution #1 – part 4 original

- In Clause 8.6.9.3 , Figure 8-21 (List Config State Machine):
  - In the CONFIG_IDLE block, remove the “ConfigPending = FALSE;” line

In the 802.1 maintenance meeting On Aug 11, 2021 this was discussed and recommendation was to get rid of the CONFIG_IDLE state as it is empty. This will be shown in the next slide

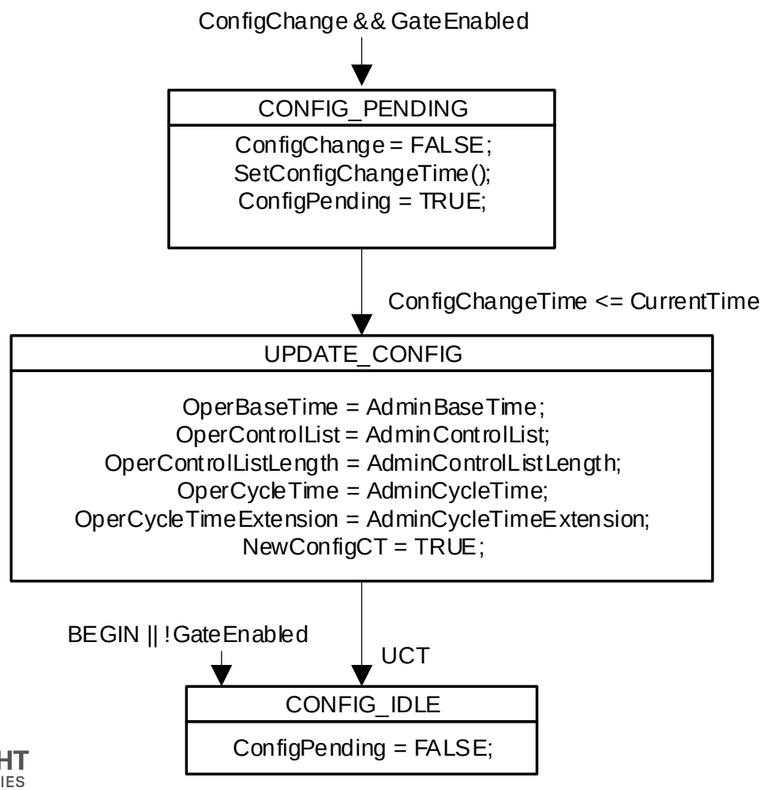


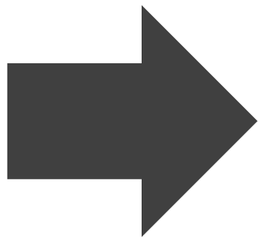


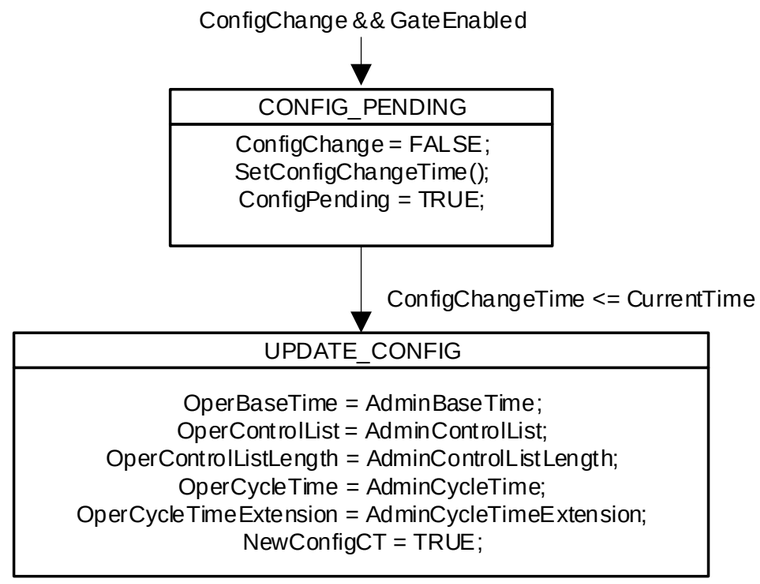
Potential Solution #1 – part 4 revised

- In Clause 8.6.9.3 , Figure 8-21 (List Config State Machine):
  - Remove the CONFIG_IDLE state and all transitions to it.

This is the new proposal as per the discussion in the 802.1 maintenance meeting on August 11, 2021

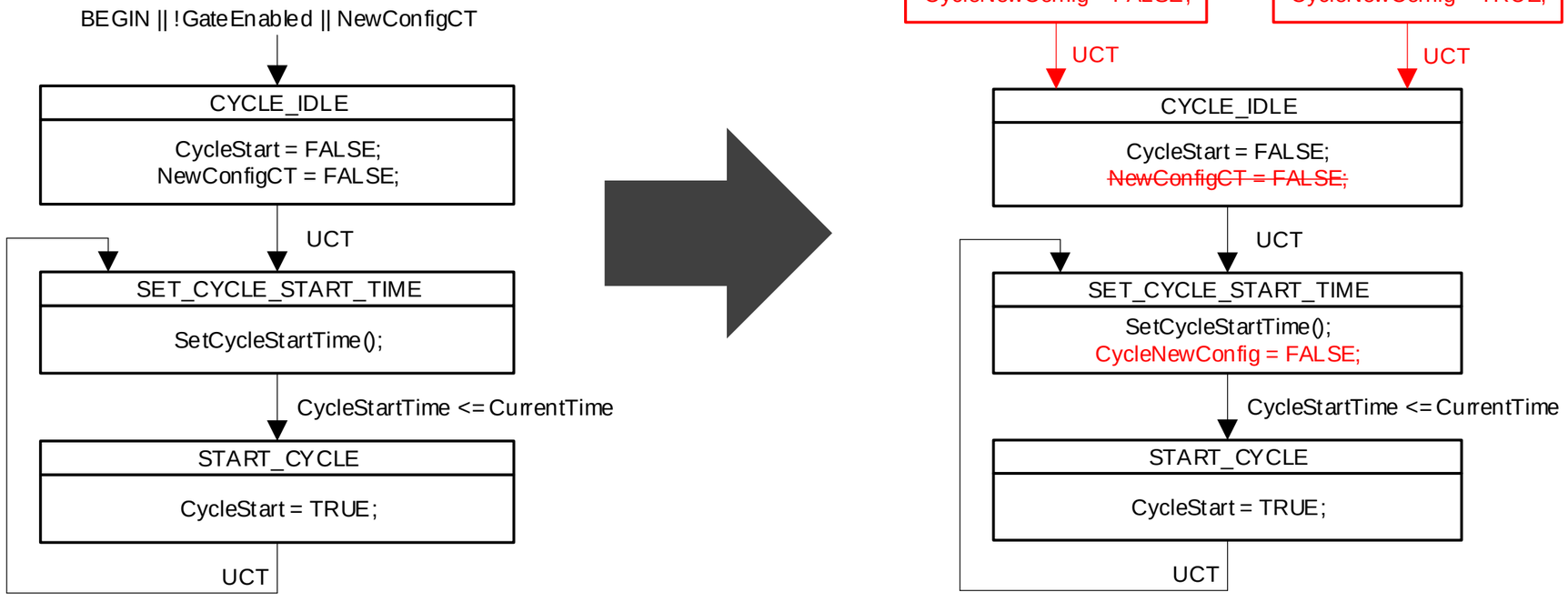






Proposed solution #2 – part 1

- In Clause 8.6.9.1 , Figure 8-19 (Cycle Timer State Machine):
  - Remove the global transition from “BEGIN || !GateEnabled || NewConfigCT” to CYCLE_IDLE
  - Add a new state named “CYCLE_INIT”
    - This state will contain the “NewConfigCT = FALSE;” and “CycleNewConfig = FALSE;”
  - Add a global transition from “BEGIN || !GateEnabled” to the new CYCLE_INIT state
  - Add a new state named “CYCLE_NEW_CONFIG”
    - This state will contain the “NewConfigCT = FALSE;” and “CycleNewConfig = TRUE;”
  - Add a global transition from “NewConfigCT” to the new CYCLE_INIT state
  - Add an UCT transition from the CYCLE_INIT state to the CYCLE_IDLE state
  - Add an UCT transition from the CYCLE_NEW_CONFIG state to the CYCLE_IDLE state
  - Remove the line “NewConfigCT = FALSE;” from the CYCLE_IDLE state
  - In the SET_CYCLE_START_TIME, after “SetCycleStartTime()” add a new line containing “CycleNewConfig = FALSE;”



Proposed solution #2 – part 2

- In Clause 8.6.9.1.1 (SetCycleStartTime() procedure)
  - Replace each occurrence of “ConfigPending = FALSE” with “(ConfigPending = FALSE) and (CycleNewConfig = FALSE)”
  - Replace each occurrence of “ConfigPending = TRUE” with “((ConfigPending = TRUE) or (CycleNewConfig = TRUE))”

a) If:
ConfigPending = FALSE, and
OperBaseTime >= CurrentTime
(i.e., OperBaseTime specifies the current time or a future time)
Then:
CycleStartTime = OperBaseTime.

b) If:
ConfigPending = FALSE, and
OperBaseTime < CurrentTime
(i.e., OperBaseTime specifies a time in the past)
Then:
CycleStartTime = (OperBaseTime + N*OperCycleTime)
where N is the smallest integer for which the relation:
CycleStartTime >= CurrentTime
would be TRUE.

c) If:
ConfigPending = TRUE, and
ConfigChangeTime > (CurrentTime + OperCycleTime + OperCycleTimeExtension)
Then:
CycleStartTime = (OperBaseTime + N*OperCycleTime)
where N is the smallest integer for which the relation:
CycleStartTime >= CurrentTime
would be TRUE.

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

a) If:
(ConfigPending = FALSE) and (CycleNewConfig = FALSE), and
OperBaseTime >= CurrentTime
(i.e., OperBaseTime specifies the current time or a future time)
Then:
CycleStartTime = OperBaseTime.

a) If:
(ConfigPending = FALSE) and (CycleNewConfig = FALSE), and
OperBaseTime < CurrentTime
(i.e., OperBaseTime specifies a time in the past)
Then:
CycleStartTime = (OperBaseTime + N*OperCycleTime)
where N is the smallest integer for which the relation:
CycleStartTime >= CurrentTime
would be TRUE.

a) If:
((ConfigPending = TRUE) or (CycleNewConfig = TRUE)), and
ConfigChangeTime > (CurrentTime + OperCycleTime + OperCycleTimeExtension)
Then:
CycleStartTime = (OperBaseTime + N*OperCycleTime)
where N is the smallest integer for which the relation:
CycleStartTime >= CurrentTime
would be TRUE.

a) If:
((ConfigPending = TRUE) or (CycleNewConfig = TRUE)), and
ConfigChangeTime <= (CurrentTime + OperCycleTime + OperCycleTimeExtension)
Then:
CycleStartTime = ConfigChangeTime