P802.1Q-Rev/D0.4 WG recirculation ballot, comment #15

This note discusses ballot comment #15, maintenance request 0286, and includes a proposed replacement Fig 8-20 that differs slightly from the resolution of that request. However the points raised here may have already been taken into account.

1. The state machine conventions (see Annex E of 802.1Q) do not specify behavior when both UCT and another condition are True for exit from a state. Other descriptions of this state machine logic have maintained that the choice of next state is undefined when multiple conditions apply. Annex E does specify the use of ELSE, which would resolve the issue in the present case. However explicitly representing the two cases of a zero and non-zero list would seem to be clearer in this case.

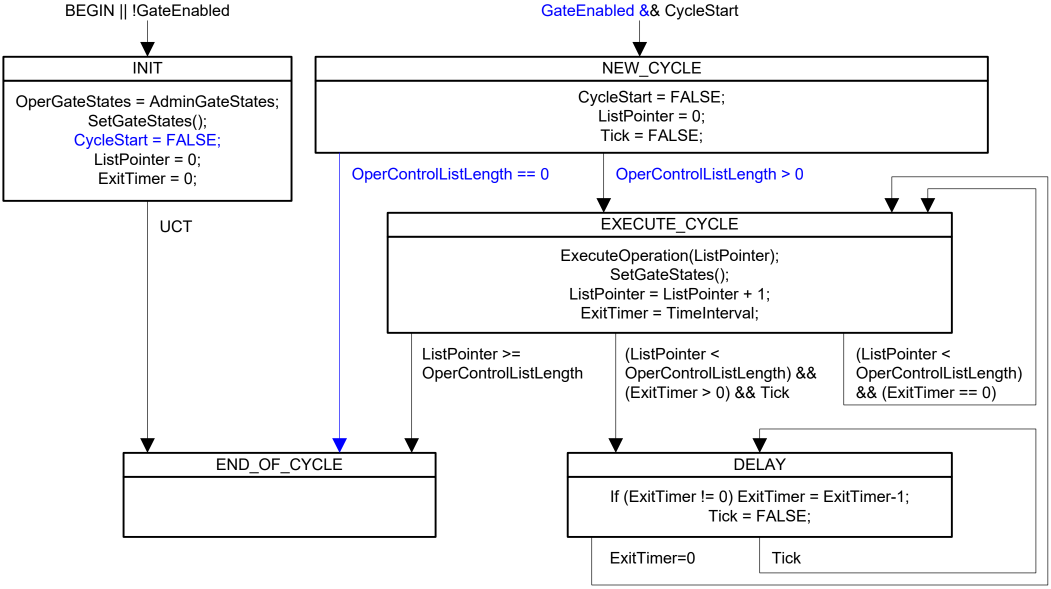
2. Given that clarification it would be possible to simply not leave the NEW_CYCLE state at all if the list length is zero, but this would risk leaving the state at an unaligned time when the list length was changed, and the END_OF_CYCLE state (which has no action, and could have been omitted) was presumably included for clarity of documentation.

3. The state machine behavior in the case of !GateEnabled and CycleStart both being True is undefined [BEGIN is special by convention (and it and open transitions in general should be better described in Annex E, but that should be left for another day) and does not suffer from this problem. In general state machines should follow the rule that “correctness is a local property”, and in this case of this machine the issue is not resolved within the set of related state machines as none of the machines appears to control the relationship between GateEnabled and CycleStart. I believe it is the case that NEW_CYCLE should not be entered if GateEnabled is False. The open entry to NEW_CYCLE is then best qualified “GateEnabled && CycleStart”. To guard against a randomly time start occuring due to a transition to GateEnabled True revealing a previous CycleStart it would also seem wise to reset CycleStart in INIT. Notionally the open transition to INIT continuously executes the INIT actions so that the variables set/reset in that state’s actions will have the assigned values up to the point that BEGIN becomes False and GateEnabled becomes True.

4. Why do we wait for Tick before transitioning from EXECUTE_CYCLE to DELAY?

A suggested replacement Fig 8-20 follows, with areas of technical change in blue (color will be removed when this or a better update is added to Q-Rev). Minor editorial change suggestions for readability: consistency of test conditions in transitions and of variable setting order in states.

Mick Seaman, Project Editor, P802.1Q-Rev 2021-05-02


BEGIN || !GateEnabled
INIT
OperGateStates = AdminGateStates;
SetGateStates();
CycleStart = FALSE;
ListPointer = 0;
ExitTimer = 0;
UCT
GateEnabled && CycleStart
NEW_CYCLE
CycleStart = FALSE;
ListPointer = 0;
Tick = FALSE;
OperControlListLength == 0
OperControlListLength > 0
EXECUTE_CYCLE
ExecuteOperation(ListPointer);
SetGateStates();
ListPointer = ListPointer + 1;
ExitTimer = TimeInterval;
ListPointer >= OperControlListLength
(ListPointer < OperControlListLength) && (ExitTimer > 0) && Tick
(ListPointer < OperControlListLength) && (ExitTimer == 0)
END_OF_CYCLE
DELAY
If (ExitTimer != 0) ExitTimer = ExitTimer-1;
Tick = FALSE;
ExitTimer=0
Tick