Comment (**145.2.5.7, P142, L9**)

In D3.1 we add the following functionality, and after simulating some parts and analyzing the changes we did, we found some errors in state machine and variable definitions that need to be corrected.

The intent of adding CLASSIFICATION_PRI, CLASS_PROBE_PRI and the exits to IDLE_PRI and CLASS_RESET_PRI was to allow the following functionality:

a) Allowing detection + class event cycles until host decides to powerup. This seems working.
b) To use the class code detected during do_class_prob_pri for 4PID as well. This seems working too.
c) Allowing doing CLASS_PROBE_PRI with 3 short class events when pse_avail_pwr_pri<4 and then class reset and then single long class event and then power up. This is not working.

Details of the issues:

1) If option_class_probe is set, state machine will either go to IDLE_PRI or to CLASS_RESET_PRI and from there eventually to IDLE_PRI instead of continuing towards powering the port, which is an error (The intent was to allow powering while doing only single long class event and not twice. Now it will not power at all).
   Details (Staring from page 142):
   From CLASSIFICATION_PRI to CLASS_PROBE_PRI option_class_probe_is TRUE.
   From CLASS_PROBE_PRI to CLASS_RESET_PRI pse_avail_pwr_pri<4.
   Moving to CLASS_RESET_PRI on page 143 and then continue to CLASS_EV1_LCE_4PID_PRI.
   Now we have two exits:
   Exit 1: if pd_class_sig_pri=4 we go to MARK_EV_LAST_PRI and then to CLASS_EVAL_PRI on page 144. Now, if we want to POWER_UP_PRI we need to meet the condition pd_req_pwr_pri<=pse_avail_pwr_pri. Since pse_avail_pwr_pri<4 which means pse_avail_pwr_pri=3 which means PSE supports only class 3 or lower and pd_class_sig_pri=4 meaning PD class is 4 or 5 (4,4,0 is class 4. 4,4,3 is class 5), we will fail the exit from CLASS_EVAL_PRI to POWER_UP_PRI due to (pd_req_pwr_pri<pse_avail_pwr_pri) which will end with POWER_DENIED_PRI.
   Exit 2: If pd_class_sig not equal to 4, which means PD class is 3 or lower, we go directly to IDLE_PRI which will also fail to powered which is an error since surely we can power all classes below class 4 starting with pd_class_sig_pri=3 or pd_class_sig_pri=2 etc.
   The end result is no matter what we do when available power<4 we can't power!

2) In the definition of option_class_probe it says that this variable is used only when pse_avail_pwr_pri<4. As a result, it is not possible that we will have exit from CLASS_PROBE_PRI to IDLE_PRI with a condition pse_avail_pwr_pri>=4 since it will never happen in this route….!!!!. As a result, in both exits from CLASS_PROBE_PRI, we need to remove the parameter pse_avail_pwr_pri and replace it with another new parameter.

2-1) As a result of (2) the definition of option_class_probe should not be depending in pse_avail_pwr_pri<4.

2-2) In addition, it will be useful if we could do the detection and classification events cycles regardless if pse_avail_pwr_pri<4.

2-3) It is possible that for the primary alternative the available power will be <4 and for the secondary the available power >4. Therefore, the option_class_probe need to be separate for primary and secondary.

2-4) From (2-1), (2-2) and (2-3), it is proposed to differentiate between option_class_probe_variable used in the single-signature PSE SM part and the dual-signature PSE SM part by using option_class_probe_pri/sec with a bit different definition compared to option_class_probe.

2-5) As a result of not using pse_avail_pwr_pri<4 and pse_avail_pwr_pri>=4 as qualifiers from CLASS_PROB_PRI to IDLE_PRI and CLASS_RESET_PRI, we need to add new variable option_class2idle instead.

3) There are other separate comments that addresses the power demotion logic that is missing and comments shows other routes that fail to power. In order to see the whole picture and discuss it, all the changes are shown with the same state machine.

4) The same comments apply to the secondary part.

Proposed Remedy:

Baseline starts here

Option 1:

1. Add the following variables to the variable list in 145.2.5.4:

option_class_probe_pri

This variable indicates if the PSE should determine the PD requested Class on the Primary Alternative by issuing 3 class events. When set to TRUE, the PSE will issue 3 class events to determine the PD requested Class, perform a classification reset by applying VReset for at least TReset to the PI (see Table 145– 14), followed by a normal classification procedure.

Values:

FALSE: The PSE will not probe for the PD requested Class.

TRUE: The PSE probes for the PD requested Class.

option_class_probe_sec

This variable indicates if the PSE should determine the PD requested Class on the Secondary Alternative by issuing 3 class events. When set to TRUE, the PSE will issue 3 class events to determine the PD requested Class, perform a classification reset by applying VReset for at least TReset to the PI (see Table 145– 14), followed by a normal classification procedure.

Values:

FALSE: The PSE will not probe for the PD requested Class.

TRUE: The PSE probes for the PD requested Class.

~~option_class2idle~~

~~This variable indicates if the PSE should continue from CLASS_PROBE_PRI to IDLE_PRI or from CLASS_PROBE_SEC to IDLE_SEC.~~

~~Values:~~

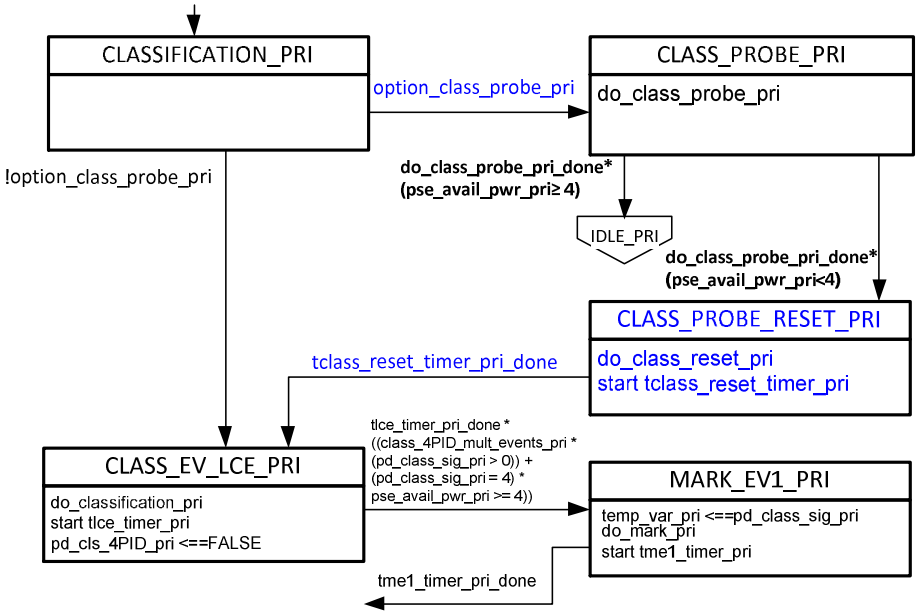
~~FALSE: The PSE will not go to IDLE_PRI or IDLE_SEC.~~

~~TRUE: The PSE will go to IDLE_PRI or IDLE_SEC.~~

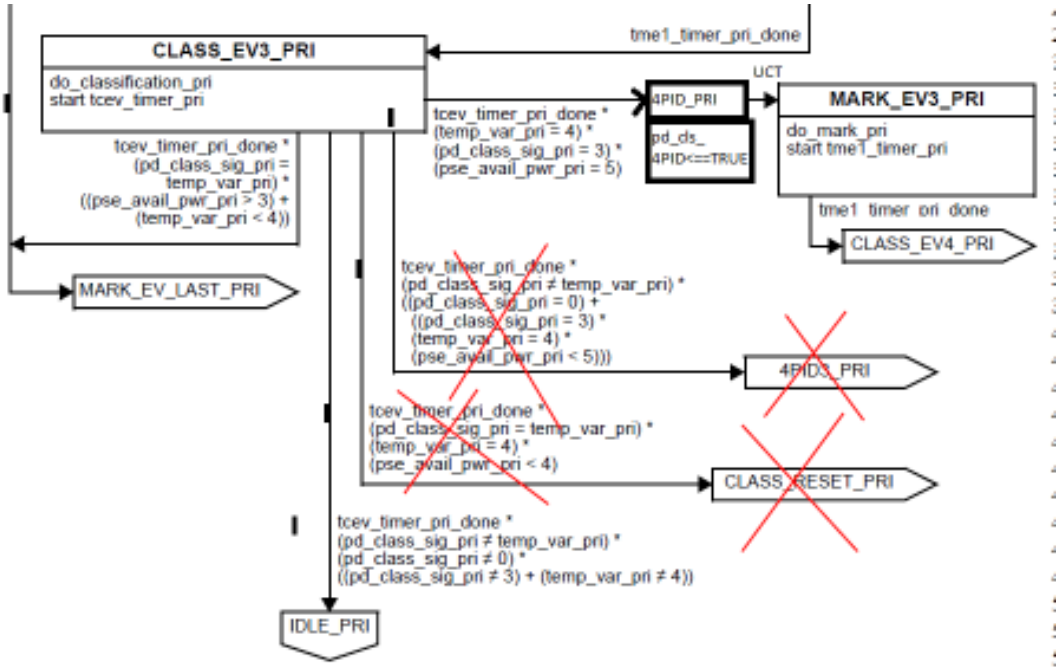
2. change the state machine as follow:

Not part of the baseline
The following solution use the same concept we used for class probe in single-signature PD without affecting the other features we add to D3.1 and fixes problems that were identified and described by other comments.

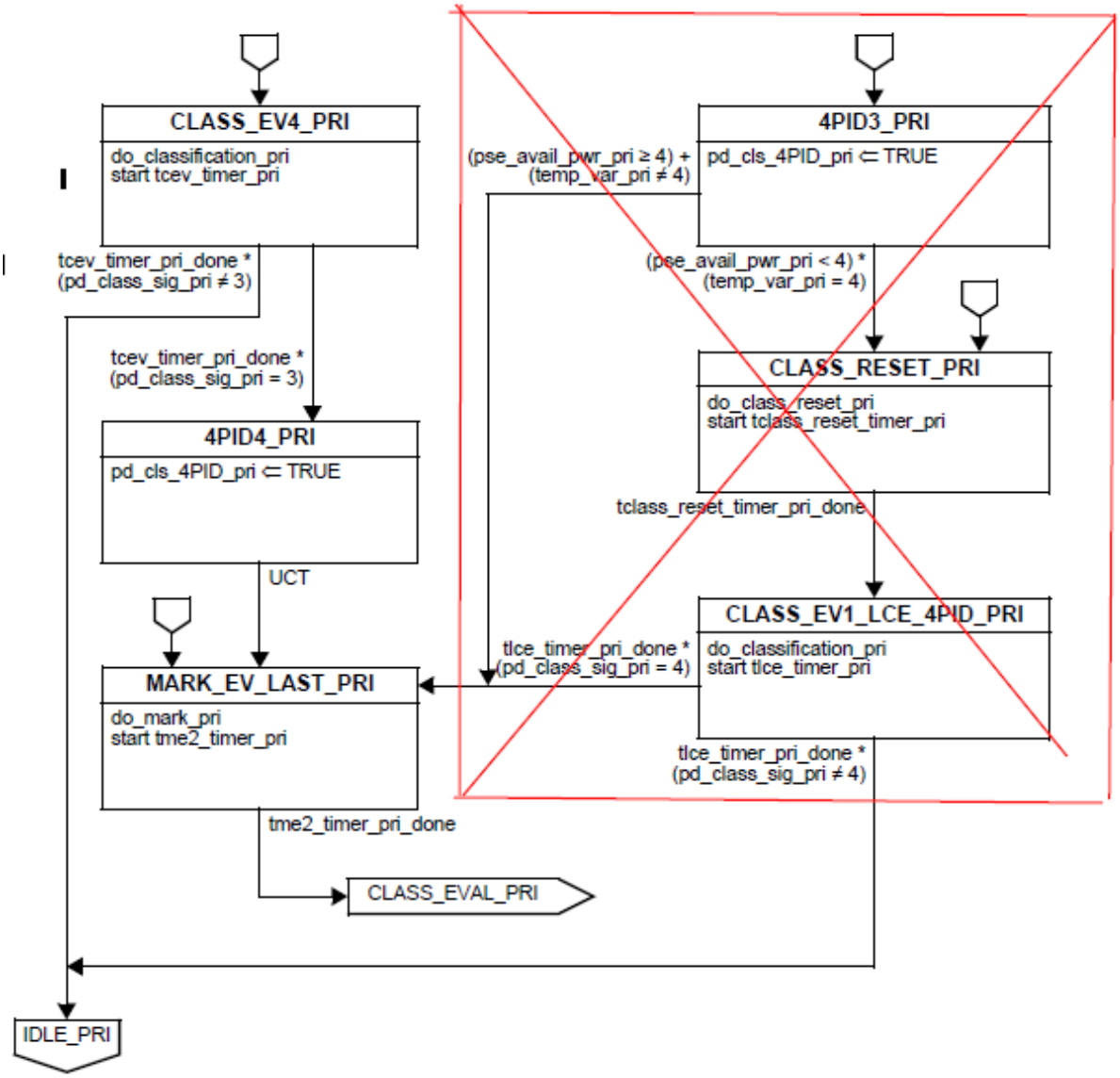
2.1 Make the following changes to the state machine per the following drawing for the primary and secondary per the example done below for the primary.



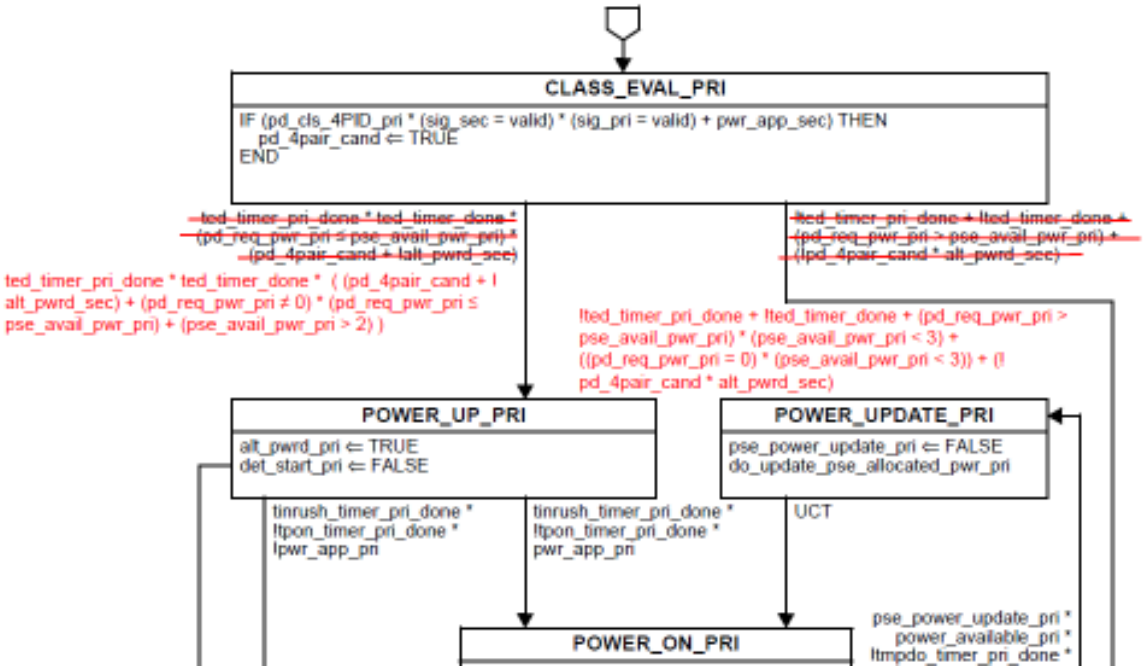
2.2 Make the following changes to the state machine per the following drawing for the primary and secondary per the example done below for the primary.



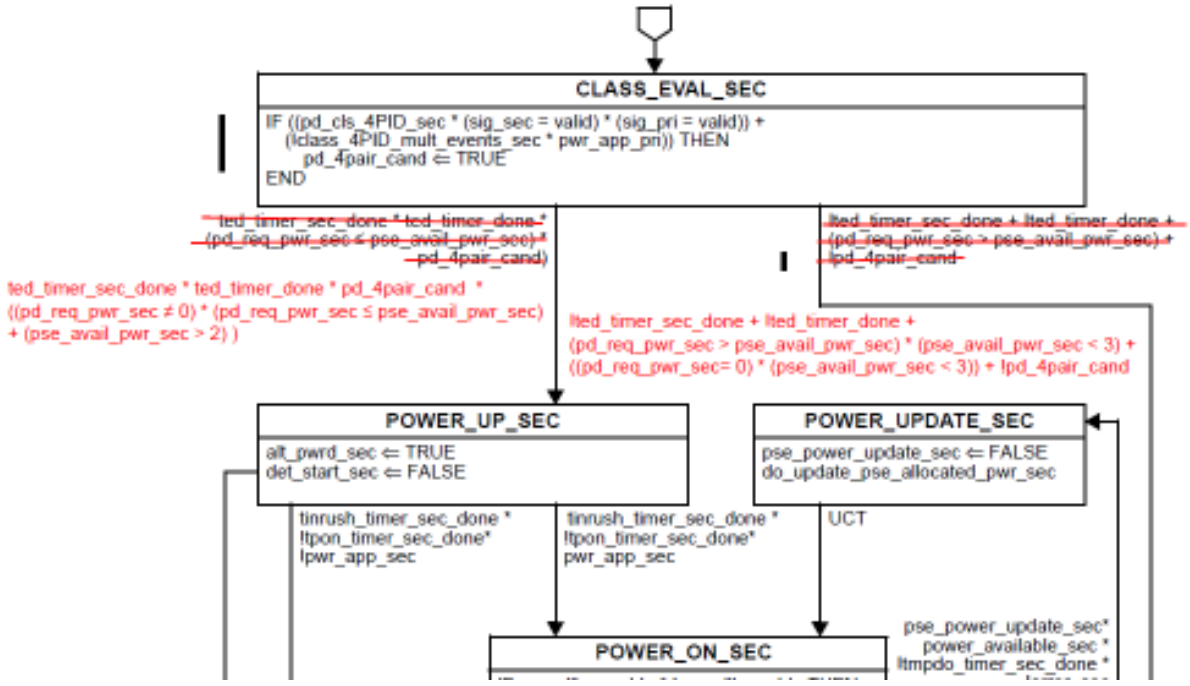
2.3 Make the following changes to the state machine per the following drawing for the primary and secondary per the example done below for the primary.



2.4 *change the state machine for the primary as shown below:*



2.5 *change the state machine for the secondary as shown below:*



Option 2:

3. Add the following variables to the variable list in 145.2.5.4:

option_class_probe_pri

This variable indicates if the PSE should determine the PD requested Class on the Primary Alternative by issuing 3 class events. When set to TRUE, the PSE will issue 3 class events to determine the PD requested Class, perform a classification reset by applying VReset for at least TReset to the PI (see Table 145– 14), followed by a normal classification procedure.

Values:

FALSE: The PSE will not probe for the PD requested Class.

TRUE: The PSE probes for the PD requested Class.

option_class_probe_sec

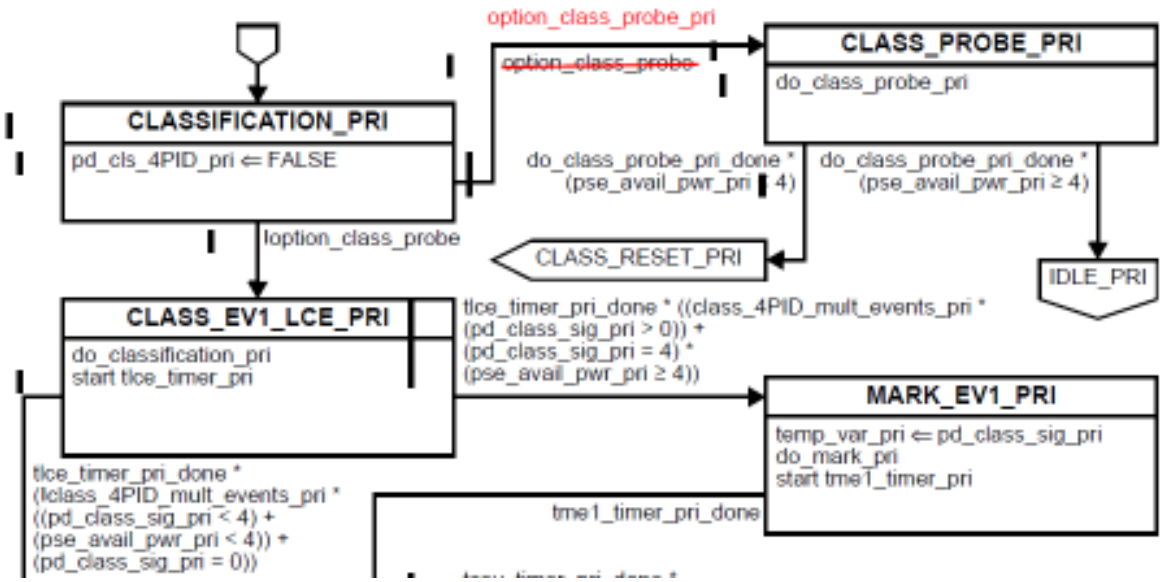
This variable indicates if the PSE should determine the PD requested Class on the Secondary Alternative by issuing 3 class events. When set to TRUE, the PSE will issue 3 class events to determine the PD requested Class, perform a classification reset by applying VReset for at least TReset to the PI (see Table 145– 14), followed by a normal classification procedure.

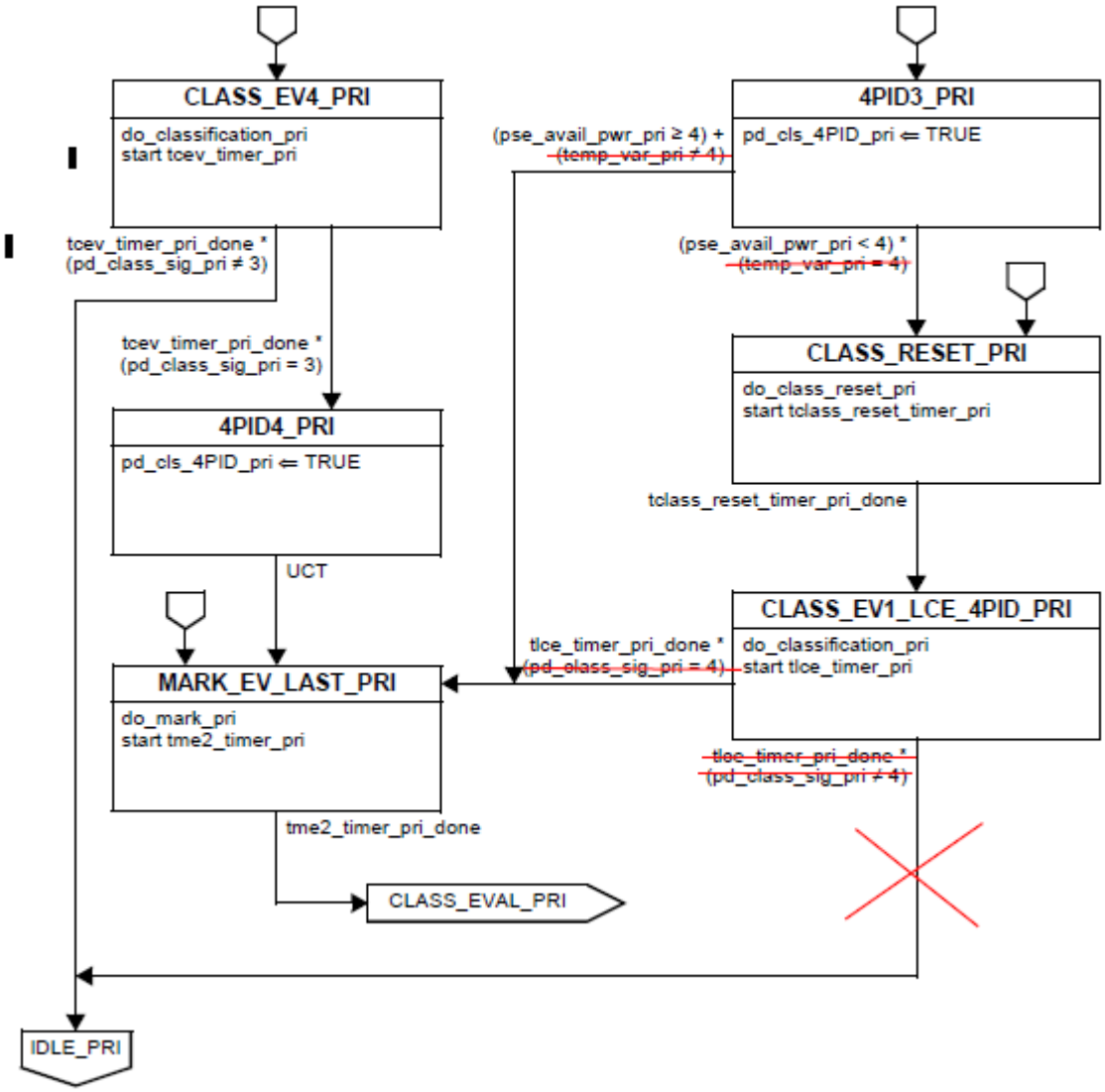
Values:

FALSE: The PSE will not probe for the PD requested Class.

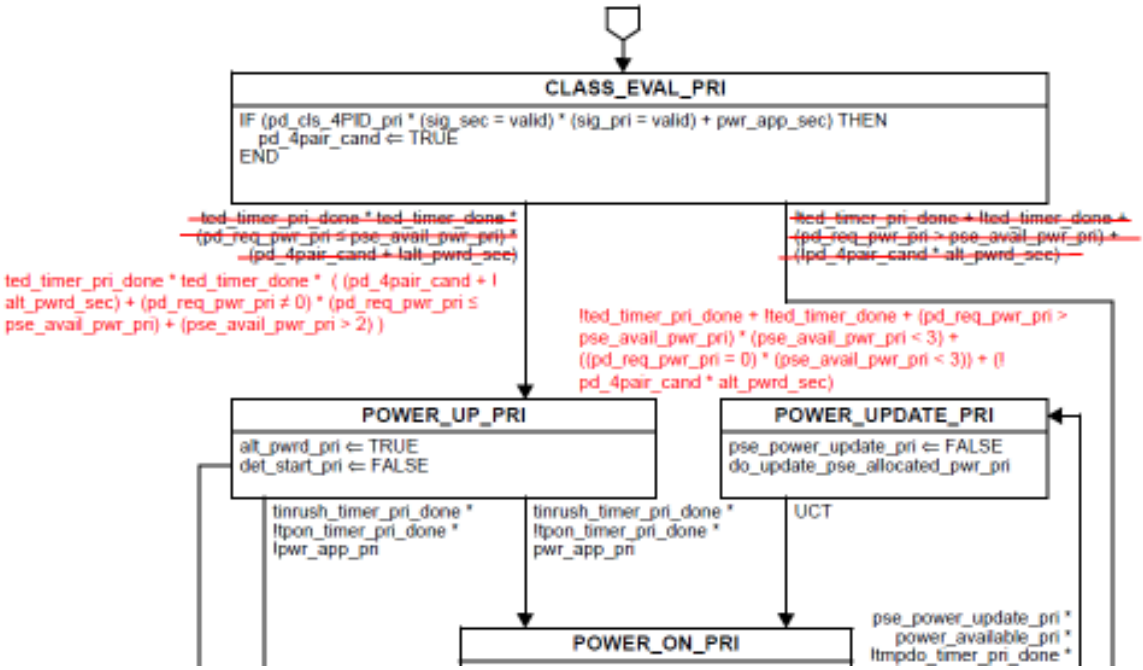
TRUE: The PSE probes for the PD requested Class.

4. *change the state machine for the primary and secondary as shown for the primary.*

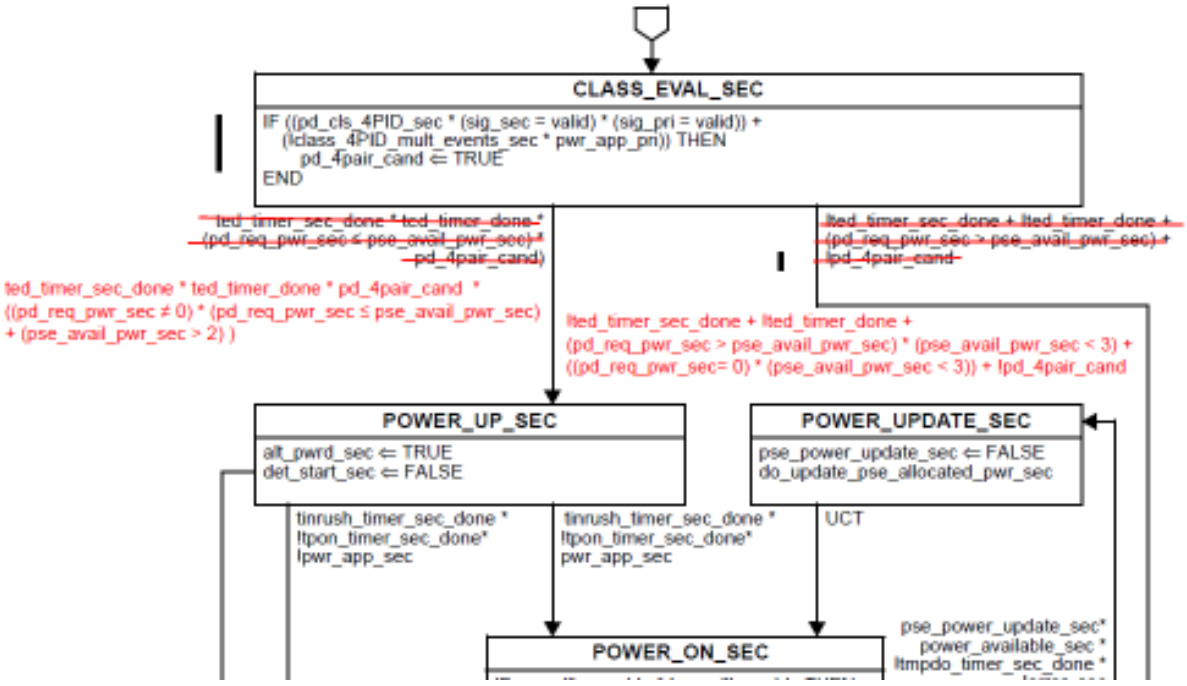



CLASS_EV4_PRI
do_classification_pri
start tcev_timer_pri
tcev_timer_pri_done *
(pd_class_sig_pri ≠ 3)
tcev_timer_pri_done *
(pd_class_sig_pri = 3)
4PID4_PRI
pd_cls_4PID_pri ⇐ TRUE
UCT
MARK_EV_LAST_PRI
do_mark_pri
start tme2_timer_pri
tme2_timer_pri_done
CLASS_EVAL_PRI
IDLE_PRI
4PID3_PRI
pd_cls_4PID_pri ⇐ TRUE
(pse_avail_pwr_pri ≥ 4) +
(temp_var_pri ≠ 4)
(pse_avail_pwr_pri < 4) *
(temp_var_pri = 4)
CLASS_RESET_PRI
do_class_reset_pri
start tclass_reset_timer_pri
tclass_reset_timer_pri_done
CLASS_EV1_LCE_4PID_PRI
do_classification_pri
start tlce_timer_pri
tlce_timer_pri_done *
(pd_class_sig_pri = 4)
tlce_timer_pri_done *
(pd_class_sig_pri ≠ 4)

5. *change the state machine for the primary as shown below:*



6. *change the state machine for the secondary as shown below:*

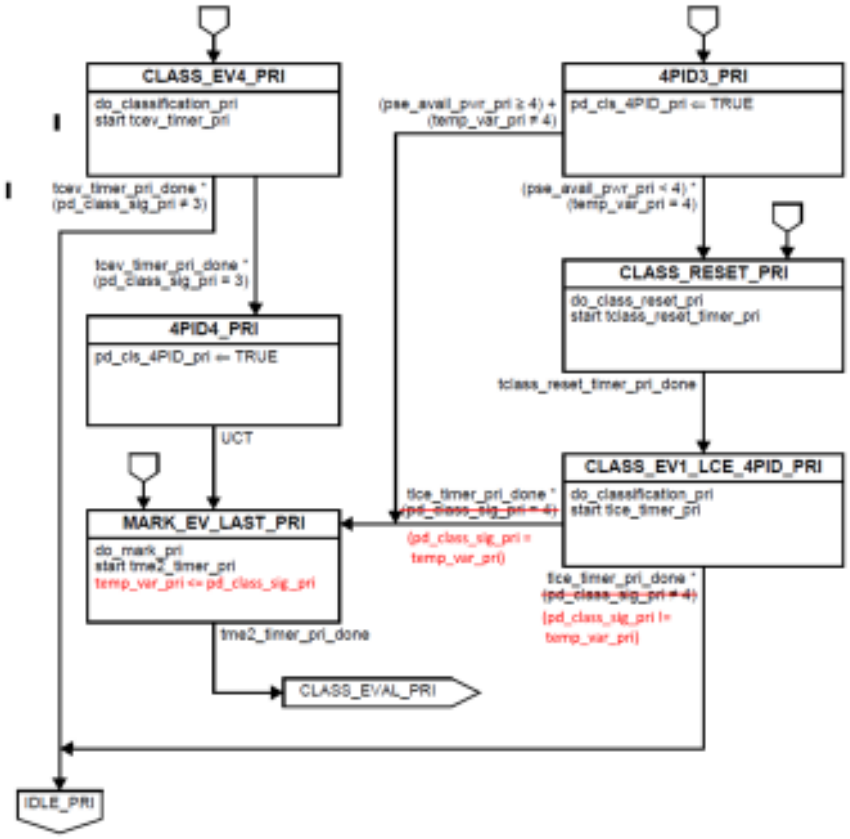


END of Baseline

Annex A - Problems related to option_class probe:

1. We need to different option_class probe variables for primary and secondary since each pair may need different power levels.
2. In option_class probe, the text defines that class_probe is used for pse_avail_pwr<4 but in the state machine there is exit from **CLASS_PROBE_PRI to IDLE_PRI** if pse_avail_pwr>=4 which is impossible.
3. The text for option_class probe for single_signature should be different than the text needed for option_class_probe_pri.

Annex B - Problems related to the use of temp_var_pri per proposal discussed over mails



1. In MARK_EV_LAST_PRI, you add the assignment, temp_var_pri <== pd_class_sig_pri. This assignment will not be used later after MARK_EV_LAST_PRI, therefore it is redundant.
2. The temp_var_pri will get the last value of pd_class_sig_pri.
   In the exit from CLASS_EV1_LCE_4PID_PRI to MARK_EV_LAST_PRI the use of temp_var_pri is not possible since it contains undefined value. Why undefined value?
   We got to CLASS_EV1_LCE_4PID_PRI from CLASS_RESET_PRI and before it we were in CLASS_PROBE_PRI.
   In CLASS_PROBE_PRI we execute the function do_class_probe_pri which doesn't return pd_class_sig_pri. It return only pd_req_pwr_pri and pd_cls_4PID_pri. It mean that temp_var_pri will not get a value.
   When running simulation, we got "use of a variable before initialization" warning.

2.1 Even if we ignore the problem of the usage of temp_var_pri, we will still have a problem since we can arrive to this state of CLASS_EV1_LCE_4PID_PRI from the normal route with class_4PID_mult_events_prI=TRUE.
Details:
with option_class_probe=FALSE and class_4PID_mult_events_pri=TRUE, we can go from CLASSIFICATION_PRI all the way to CLASS_EV3_PRI.
In case of code 4,4,3 for example, we will reach to 4PID3_PRI and then to CLASS_RESET_PRI because temp_var_pri=4. From CLASS_RESET_PRI we go to CLASS_EV1_LCE_4PID_PRI.
Currently temp_var_pri=4 and now we do new class event with CLASS_EV1_LCE_4PID_PRI that result with pd_class_sig=4 while temp_var_pri will get the previous value of pd_class_sig that was 3. As a result we will continue to IDLE_PRI because the propose new condition of pd_class_sig_pri not equal to temp_var_pri.
Meaning we will not power up class 5 PDs even we have pse_avai_pwr_pri=3.

Annex C – Derivation or power demotion logics for the primary

1. Change the exit from CLASS_EVAL_PRI to POWER_DENIGED_PRI from:
!ted_timer_pri_done + !ted_timer_done + (pd_req_pwr_pri > pse_avail_pwr_pri) + (!pd_4pair_cand * alt_pwrd_sec)

[In single signature part, we have:
!ted_timer_sec_done + (!ted_timer_done)+
((pd_req_pwr > pse_avail_pwr) * (pse_avail_pwr < 3)) +((pd_req_pwr = 0) * (pse_avail_pwr < 3))]

To:
!ted_timer_pri_done + !ted_timer_done + (pd_req_pwr_pri > pse_avail_pwr_pri) * (pse_avail_pwr_pri < 3) + ((pd_req_pwr_pri = 0) * (pse_avail_pwr_pri < 3)) + (!pd_4pair_cand * alt_pwrd_sec)

2. Change the exit from CLASS_EVAL_PRI to POWER_UP_PRI from:
ted_timer_pri_done * ted_timer_done * (pd_req_pwr_pri ≤ pse_avail_pwr_pri) * (pd_4pair_cand + !alt_pwrd_sec)

[In single signature part we have:
ted_timer_done * ted_timer_sec_done *
(((pd_req_pwr ≠ 0) * (pd_req_pwr ≤ pse_avail_pwr)) +(pse_avail_pwr > 2))]

To:
ted_timer_pri_done * ted_timer_done * ((pd_4pair_cand + !alt_pwrd_sec) + (pd_req_pwr_pri ≠ 0) * (pd_req_pwr_pri ≤ pse_avail_pwr_pri) + (pse_avail_pwr_pri > 2))

Annex B – Derivation or power demotion logics for the secondary

1. Change the exit from CLASS_EVAL_SEC to POWER_DENIGED_SEC from:
!ted_timer_sec_done + !ted_timer_done + (pd_req_pwr_sec > pse_avail_pwr_sec) + !pd_4pair_cand

[In single signature part we have:
!ted_timer_sec_done + (!ted_timer_done)+
((pd_req_pwr > pse_avail_pwr) * (pse_avail_pwr < 3)) +((pd_req_pwr = 0) * (pse_avail_pwr < 3))]

To:
!ted_timer_sec_done + !ted_timer_done +
(pd_req_pwr_sec > pse_avail_pwr_sec) * (pse_avail_pwr_sec < 3) +
((pd_req_pwr_sec= 0) * (pse_avail_pwr_sec < 3)) + !pd_4pair_cand

2. Change the exit from CLASS_EVAL_SEC to POWER_UP_SEC from:
ted_timer_sec_done * ted_timer_done * (pd_req_pwr_sec ≤ pse_avail_pwr_sec) * pd_4pair_cand)

[In single signature part we have:
ted_timer_done * ted_timer_sec_done *
(((pd_req_pwr ≠ 0) * (pd_req_pwr ≤ pse_avail_pwr)) +(pse_avail_pwr > 2))]

To:
ted_timer_sec_done * ted_timer_done * pd_4pair_cand *
((pd_req_pwr_sec ≠ 0) * (pd_req_pwr_sec ≤ pse_avail_pwr_sec) + (pse_avail_pwr_sec > 2))