Simplified classification in the State Diagram v110

Info (not part of baseline)

This baseline is a replacement for the Type 3/4 PSE single-signature classification state diagram. A number of topics are addressed:

1. The current D2.1 state diagram uses a complex combination of class_num_events and pse_avail_pwr to control the classification flow. A further problem is that in the D2.1 text several permutations that should be possible are not (eg. A Type 4 PSE can only have pse_avail_pwr = 7 or 8) This baseline simplifies this by removing class_num_events and only work with pse_avail_pwr.

2. Two unrelated fixes that prevent the SD from working are fixed in the first subsection

3. The functionality of CLASS_EVAL is to handle the special case that the PSE cannot power at all (pse_avail_pwr < 3). This is merged into the mainline classification diagram.

4. The state POWER_DENIED executes no statements, and has UCT as the exit condition. We can remove it, without this having any affect on state diagram operation.

5. The variables pd_req_power and pd_allocated_power are currently returned by a function. The assignment of those variables is however complex (due to power demotion) and should be done explicitly by the SD, rather than 'magically' get filled out by a function. For this purpose the states CLASS_EVAL_A and CLASS_EVAL_B are introduced.

6. An open ticket on the TDL is to implement improved discovery for the PD's requested Class, even for PSEs that have pse_power_available ≤ 3. Two separate changes are introduced:

   - For PSEs with pse_power_available = 4, these will now always produce 3 class events, to discover the requested Class
   - PSEs with pse_power_available ≤ 3 make an initial 3 events, store the requested Class, reset the PD classification, and proceed back to CLASS_EV1_LCE. To enable this particular behavior, the variable option_classprobe must be set to TRUE prior to classification. The SD will reset this variable to FALSE when it performs a class reset.

7. There are several 'shalls' in the Type 3/4 variable list. These are moved to the appropriate sections and struck from the variable list.

Work items still open:

- PDs that request Class 0 have not been tested
- Harmonize dual-signature with single-signature
- Invalid PD classification currents are not tested

Fixes to the Type 3/4 PSE state diagram

Make changes as follows in Figure 33–15 through 33–22:

FROM state	TO state	Change
DETECT_EVAL	IDLE	(pse_alternative = both) * ((det_temp = only_one) * (sig_pri $\neq$ valid) + (det_temp = both_neither) * (sig_sec $\neq$ valid) + (((CC_DET_SEQ = 0) + (CC_DET_SEQ = 3)) * (det_temp = only_one) * tdet2det_timer_done)) + (pse_alternative $\neq$ both) * (sig_pri $\neq$ valid)
IDLE_INRUSH_PRI	MONITOR_INRUSH_PRI	alt_pwrd_pri ~~* !pwr_app_pri~~
IDLE_INRUSH_SEC	MONITOR_INRUSH_SEC	alt_pwrd_sec ~~* !pwr_app_sec~~

Simplified classification

33.2.5.9 Type 3 and Type 4 variables

Add the following variables:

option_classprobe

This variable indicates if the PSE should determine the requested Class of the PD when pse_avail_pwr is less than 3. When set to TRUE, the PSE will issue 3 class events to determine the requested Class, perform a classification reset by applying V_{Reset} for at least T_{Reset} to the PI (see Table 33–17), 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

pd_allocated_pwr

A variable that indicates the Class that has been assigned to the PD.
Values:

1: Class 1
2: Class 2
3: Class 3
4: Class 4
5: Class 5
6: Class 6
7: Class 7
8: Class 8

Info (not part of baseline)

The pd_req_pwr variable (returned by do_classification) had a shall in it and a may. Both of these have been moved to the classification section. The original text was:

> pd_req_pwr: This variable indicates the power class requested by the PD. When a PD requests a higher class than a PSE can support, the PSE shall assign the PD Class 3, 4, or 6, whichever is the highest that it can support. For Type 3 and Type 4 PSEs, when connected to a single-signature PD, operating over 4-pairs, classification events may appear on one or both pairsets. See 33.2.7.

pd_req_pwr

The variable indicates the power class requested by the PD. When a PD requests a higher Class than a PSE can support, the PSE assigns the PD to Class 3, Class 4, or Class 6, whichever is the highest Class it can support. If pse_avail_pwr is less than 4, this variable does not contain the requested Class by the PSE; see pd_req_pwr_probe.
Values:

1: Class 1
2: Class 2
3: Class 3
4: Class 4
5: Class 5
6: Class 6
7: Class 7
8: Class 8

pd_req_pwr_probe

The variable indicates the power class requested by the PD when option_classprobe is TRUE and pse_avail_pwr is less than 4.
Values:

1: Class 1
2: Class 2
3: Class 3
4: Class 4
5: Class 5
6: Class 6
7: Class 7

8: Class 8

Info (not part of baseline)

We can now remove Table 33–7 (which ties PSE Type to class_num_events) and Table 33–8 (which ties class_num_events to pse_avail_power). This is then replaced by a Table that links PSE Type to the allowed values of pse_avail_power.

Change text on page 82 as follows:

~~Type 3 and Type 4 PSEs shall meet at least one of the allowable variable definition permutations described in Table 33–7 and Table 33–8.~~
Type 3 and Type 4 PSEs shall set pse_avail_pwr, pse_avail_pwr_pri, and pse_avail_pwr_sec from the range in Table 33–7.

Remove Table 33–7 and 33–8.

Insert a new Table as follows:

Table 33–7 — Allowed Type 3 and Type 4 permutations for pse_avail_pwr

PSE Type	pse_avail_pwr	pse_avail_pwr_pri, pse_avail_pwr_sec
Type 3	1 to 6	1 to 4
Type 4	1 to 8	1 to 5

33.2.5.11 Type 3 and Type 4 functions

Remove variables pd_req_pwr and pd_allocated_pwr from the do_classification function.

Rename variables as follows:

Old name	**New name**
pd_class_detected	pd_class_sig
pd_class_detected_pri	pd_class_sig_pri
pd_class_detected_sec	pd_class_sig_sec

33.2.5.12 Type 3 and Type 4 state diagrams

Remove the states CLASS_EVAL and POWER_DENIED (including in and outgoing arcs) from Figure 33–15.

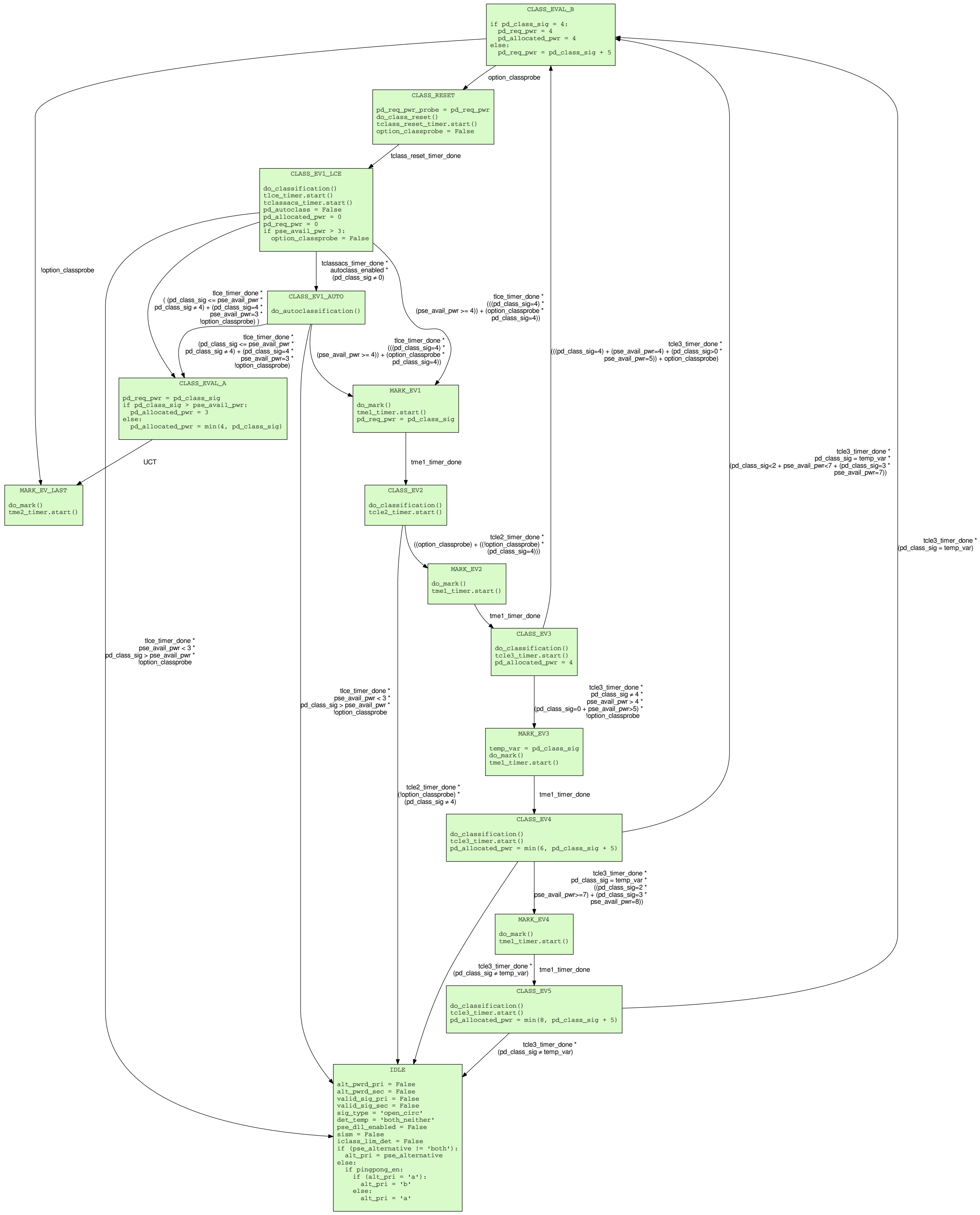
Change Figure 33–15 as follows:

FROM state	TO state	Change
POWER_ON	IDLE	(tmpdo_timer_done + !power_available) * !short_det_pri * !ovld_det_pri * !option_vport_lim_pri * !short_det_sec * !ovld_det_sec * !option_vport_lim_sec ~~* power_available~~

Info (not part of baseline)

Note that the figure wich replaces Fig 33–18 was automatically generated from the state diagram simulator. The layout isn't always optimal, and small changes to the text sometimes cause a complete re-arrangement. When drawn in Frame, a manual, sensible layout will be crafted. Statements are generally to the RIGHT of the arc they belong to.

Replace Figure 33–18 as follows (redraw it in Frame):



```
CLASS_EVAL_B

if pd_class_sig = 4:
  pd_req_pwr = 4
  pd_allocated_pwr = 4
else:
  pd_req_pwr = pd_class_sig + 5
```

```
CLASS_RESET

pd_req_pwr_probe = pd_req_pwr
do_class_reset()
tclass_reset_timer.start()
option_classprobe = False
```

```
CLASS_EV1_LCE

do_classification()
tlce_timer.start()
tclassacs_timer.start()
pd_autoclass = False
pd_allocated_pwr = 0
pd_req_pwr = 0
if pse_avail_pwr > 3:
  option_classprobe = False
```

```
CLASS_EV1_AUTO

do_autoclassification()
```

```
CLASS_EVAL_A

pd_req_pwr = pd_class_sig
if pd_class_sig > pse_avail_pwr:
  pd_allocated_pwr = 3
else:
  pd_allocated_pwr = min(4, pd_class_sig)
```

```
MARK_EV1

do_mark()
tme1_timer.start()
pd_req_pwr = pd_class_sig
```

```
MARK_EV_LAST

do_mark()
tme2_timer.start()
```

```
CLASS_EV2

do_classification()
tcle2_timer.start()
```

```
MARK_EV2

do_mark()
tme1_timer.start()
```

```
CLASS_EV3

do_classification()
tcle3_timer.start()
pd_allocated_pwr = 4
```

```
MARK_EV3

temp_var = pd_class_sig
do_mark()
tme1_timer.start()
```

```
CLASS_EV4

do_classification()
tcle3_timer.start()
pd_allocated_pwr = min(6, pd_class_sig + 5)
```

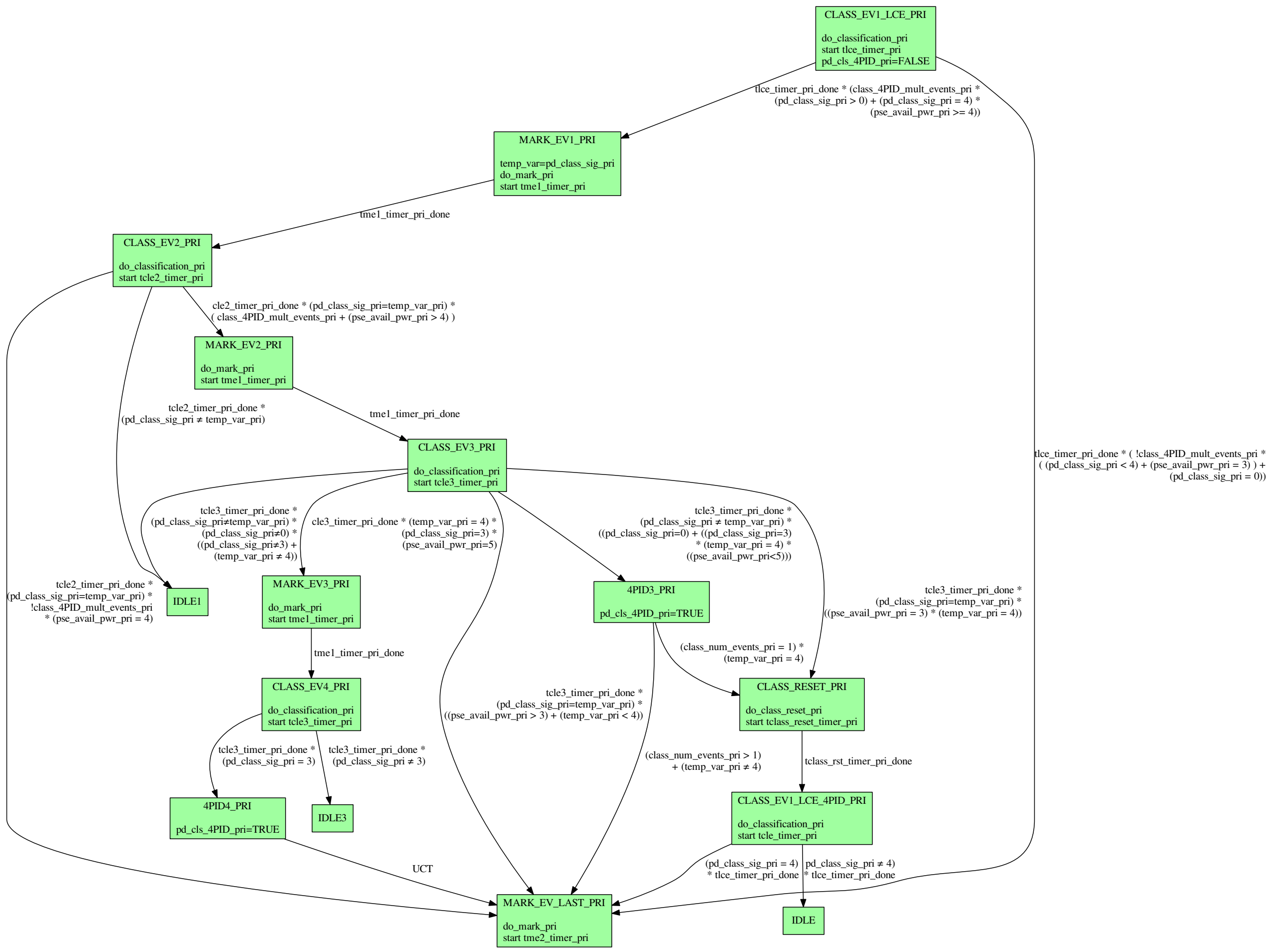
```
MARK_EV4

do_mark()
tme1_timer.start()
```

```
CLASS_EV5

do_classification()
tcle3_timer.start()
pd_allocated_pwr = min(8, pd_class_sig + 5)
```

```
IDLE

alt_pwrd_pri = False
alt_pwrd_sec = False
valid_sig_pri = False
valid_sig_sec = False
sig_type = 'open_circ'
det_temp = 'both_neither'
pse_dll_enabled = False
sism = False
iclass_lim_det = False
if (pse_alternative != 'both'):
  alt_pri = pse_alternative
else:
  if pingpong_en:
    if (alt_pri = 'a'):
      alt_pri = 'b'
    else:
      alt_pri = 'a'
```

option_classprobe
tclass_reset_timer_done
!option_classprobe
tclassacs_timer_done * autoclass_enabled * (pd_class_sig ≠ 0)
tlce_timer_done * ((pd_class_sig <= pse_avail_pwr * pd_class_sig ≠ 4) + (pd_class_sig=4 * pse_avail_pwr=3 * !option_classprobe))
tlce_timer_done * (((pd_class_sig=4) * (pse_avail_pwr >= 4)) + (option_classprobe * pd_class_sig=4))
tlce_timer_done * (pd_class_sig <= pse_avail_pwr * pd_class_sig ≠ 4) + (pd_class_sig=4 * pse_avail_pwr=3 * !option_classprobe)
tlce_timer_done * (((pd_class_sig=4) * (pse_avail_pwr >= 4)) + (option_classprobe * pd_class_sig=4))
tcle3_timer_done * (((pd_class_sig=4) + (pse_avail_pwr=4) + (pd_class_sig>0 * pse_avail_pwr=5)) + option_classprobe)
UCT
tme1_timer_done
tcle3_timer_done * pd_class_sig = temp_var * (pd_class_sig<2 + pse_avail_pwr<7 + (pd_class_sig=3 * pse_avail_pwr=7))
tcle2_timer_done * ((option_classprobe) + ((!option_classprobe) * (pd_class_sig=4)))
tcle3_timer_done * (pd_class_sig = temp_var)
tme1_timer_done
tlce_timer_done * pse_avail_pwr < 3 * pd_class_sig > pse_avail_pwr * !option_classprobe
tlce_timer_done * pse_avail_pwr < 3 * pd_class_sig > pse_avail_pwr * !option_classprobe
tcle3_timer_done * pd_class_sig ≠ 4 * pse_avail_pwr > 4 * (pd_class_sig=0 + pse_avail_pwr>5) * !option_classprobe
tcle2_timer_done * (!option_classprobe) * (pd_class_sig ≠ 4)
tme1_timer_done
tcle3_timer_done * pd_class_sig = temp_var * ((pd_class_sig=2 * pse_avail_pwr>=7) + (pd_class_sig=3 * pse_avail_pwr=8))
tcle3_timer_done * (pd_class_sig ≠ temp_var)
tme1_timer_done
tcle3_timer_done * (pd_class_sig ≠ temp_var)

Replace Figure 33–19 as follows (redraw it in Frame):



Replace Figure 33–20 with that same state diagram, but replace "_pri" with "_sec".

33.2.7 PSE classification of PDs and mutual identification

Change the text on page 106, line 5 as follows:

Physical Layer classification occurs before a PSE supplies power to a PD, when the PSE asserts a voltage in the range of V Class as defined in Table 33–17 onto one or both pairsets. This is called a class event. The PD responds to each class event with a current representing one of a limited number of classification signatures.

The assigned Class is the result of the PDs requested Class and the number of class events produced by the PSE as shown in Table 33–13 and Table 33–14. See 33.3.6 for PD classification behavior. When a single- signature PD requests a higher Class than a Type 3 or Type 4 PSE can support, the PSE shall assign~~s~~ the PD to Class 3, 4, or 6, whichever is the highest that it can support. When a dual-signature PD requests a higher Class than a Type 3 or Type 4 PSE can support, the PSE assigns the PD Class 3 or 4, whichever is the highest that it can support.

33.2.7.2 PSE Multiple-Event Physical Layer classification

Make changes to 33.2.7.2 as follows:

Type 2 PSEs shall provide a maximum of two class events and two mark events. Type 3 PSEs shall provide a maximum of four class events and four mark events for single-signature PDs and a maximum of three class events and three mark events on each pairset for dual-signature PDs unless a class reset event clears the class and mark event counts. Type 4 PSEs shall provide a maximum of five class events and five mark events for single-signature PDs and a maximum of four class events and four mark events on each pairset for dual-sig- nature PDs unless a class reset event clears the class and mark event counts. Type 3 and Type 4 PSEs may issue a class reset event to perform mutual identification or to discover the requested Class of the PD when the PSE's power budget only allows a single classification event.

All class event voltages and mark event voltages shall have the same polarity as defined for V_{Port_PSE-2P} in 33.2.4. Type 3 and Type 4 PSEs may issue classification events on one or both pairsets, when connected to a single-signature PD and operating over 4-pairs.

The PSE shall complete Multiple-Event Physical Layer classification and transition to the POWER_ON, POWER_ON_PRI, or POWER_ON_SEC state without allowing the voltage at the PI or pairset to go below $\mathrm{V}_{\mathrm{Mark}}$ min, unless in the CLASS_RESET, CLASS_RESET_PRI, or CLASS_RESET_SEC states. If the PSE returns to the IDLE state, it shall maintain the PI voltage at $\mathrm{V}_{\mathrm{Reset}}$ for a period of at least $\mathrm{T}_{\mathrm{Reset}}$ min before starting a new detection cycle. If the PSE is in the CLASS_RESET, CLASS_RESET_PRI, or CLASS_RESET_SEC state it shall maintain the PI or pairset voltage at $\mathrm{V}_{\mathrm{Reset}}$ for a period of at least $\mathrm{T}_{\mathrm{Reset}}$ min.

Fixes to the PD state diagram

Info (not part of baseline)

No default value is set for the variable pse_dll_power_type. This causes an invalid comparison on the exit from MDI_POWER1 to MDI_POWER2. Best solution is to set pse_dll_power_type to ‘1’ in the IDLE state.

Append the IDLE state in Figure 33–32 with the statement as follows:

pse_dll_power_type $\Leftarrow$ 1