PSE State Diagram – Sequence 3

Considerations for Staggered Detection and Power-on of Dual-Signature PDs.

Dan Dove, DNS for LTC
Chris Bullock, Cisco
Dylan Walker, Cisco

IEEE 802.3bt Task Force
Oct 2015

Goal(s):

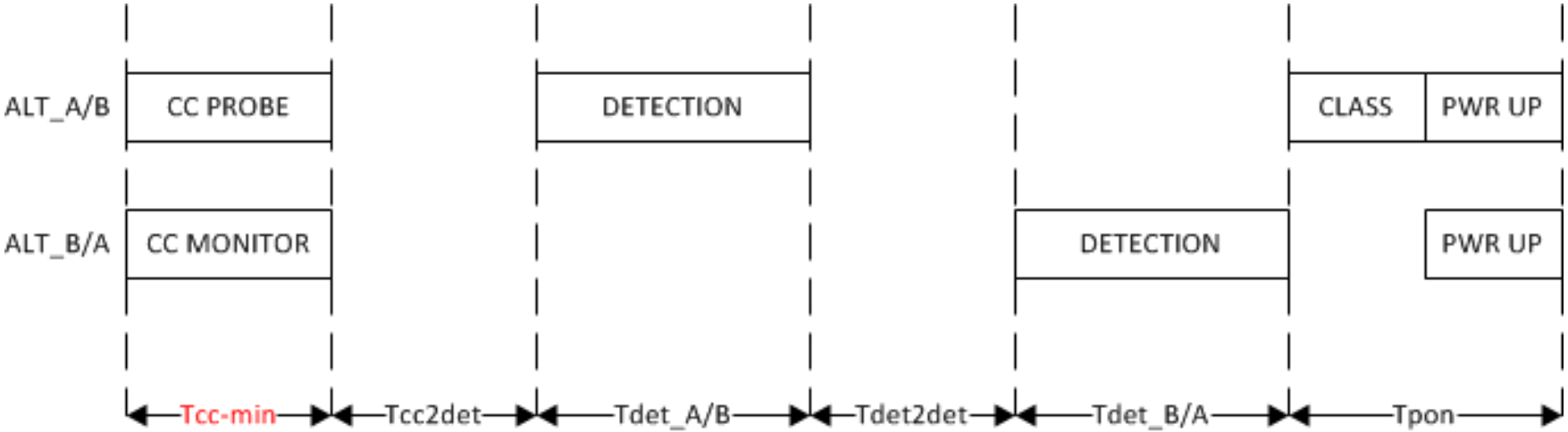
1. Add a sequence to the Type3 / Type 4 State Diagram that allows for a more cost-effective implementation in some architectures.
2. Add text to define the interchangeable nature of the treatment of Alternative-A and Alternative-B in the State Diagram.

Sequence 3:

Staggered Detection and Power-up of DS PDs.

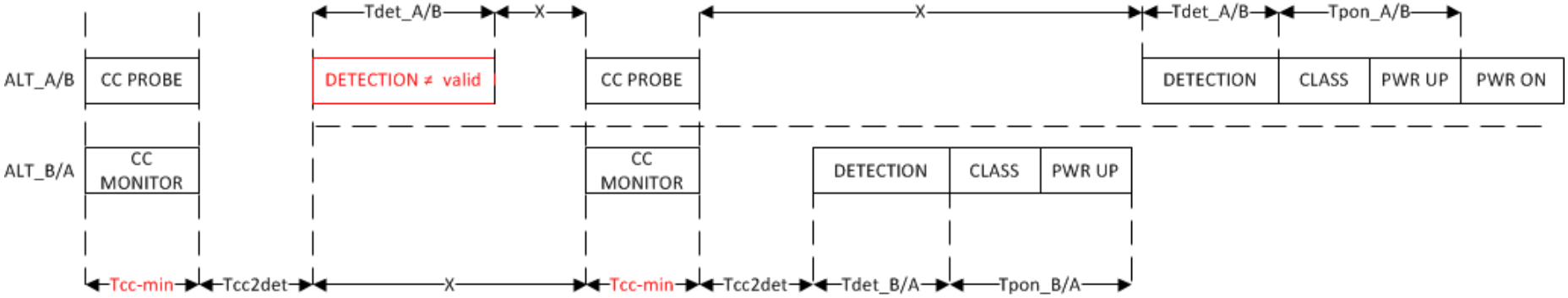
CC → Det/Cls/Pwr ALT_A/B → Det/Cls/Pwr ALT_B/A

CC → Det (SS PD)



For SS PDs, Sequence 3 is identical to Sequence 0

CC → Det/Cls/Pwr Alt_A/B (DS PD)



Tcc2det is enforced for the first pairset to successfully power.

Tdet and Tpon are enforced for each pairset.

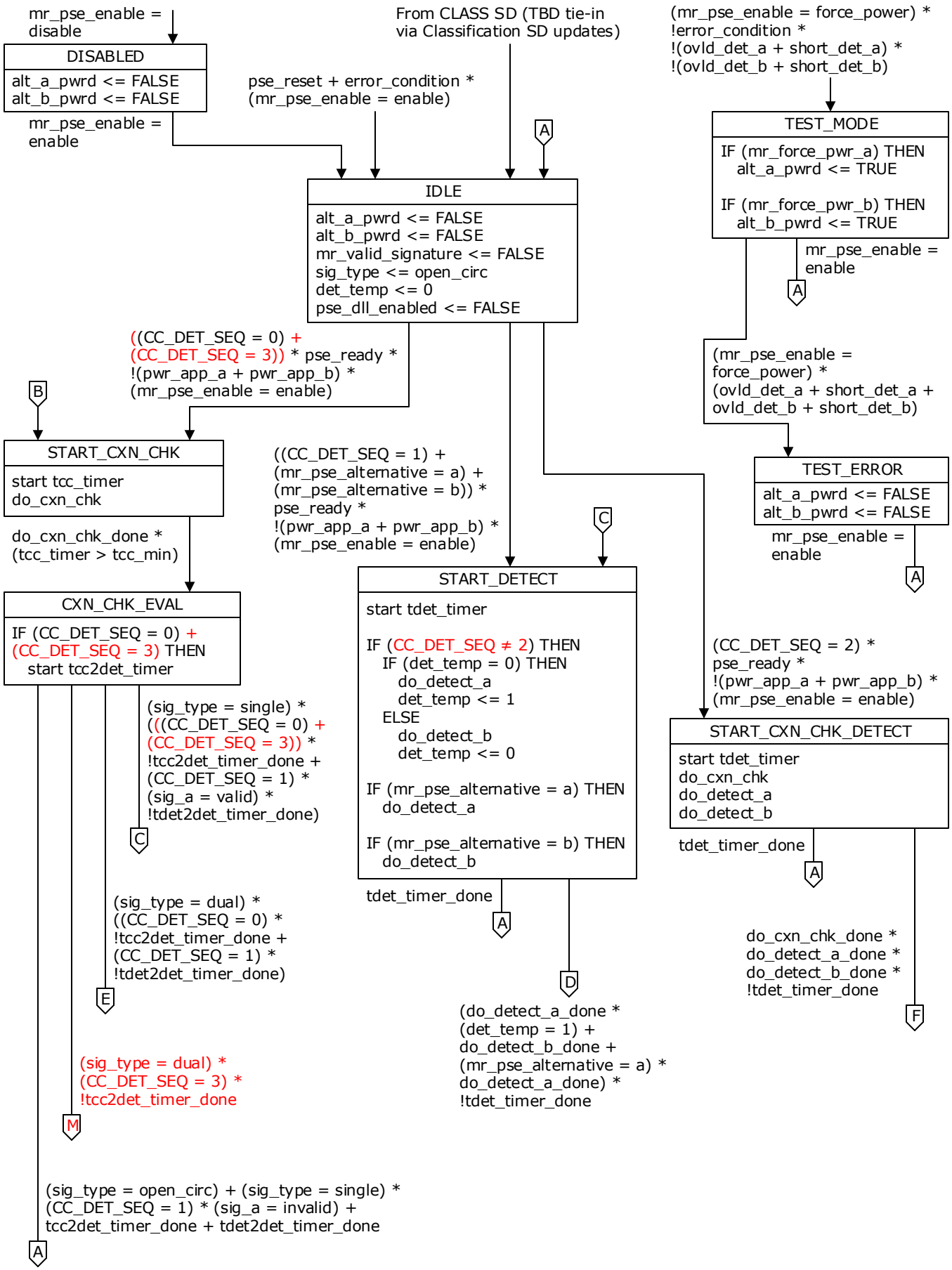
Alternative A/B Parity:

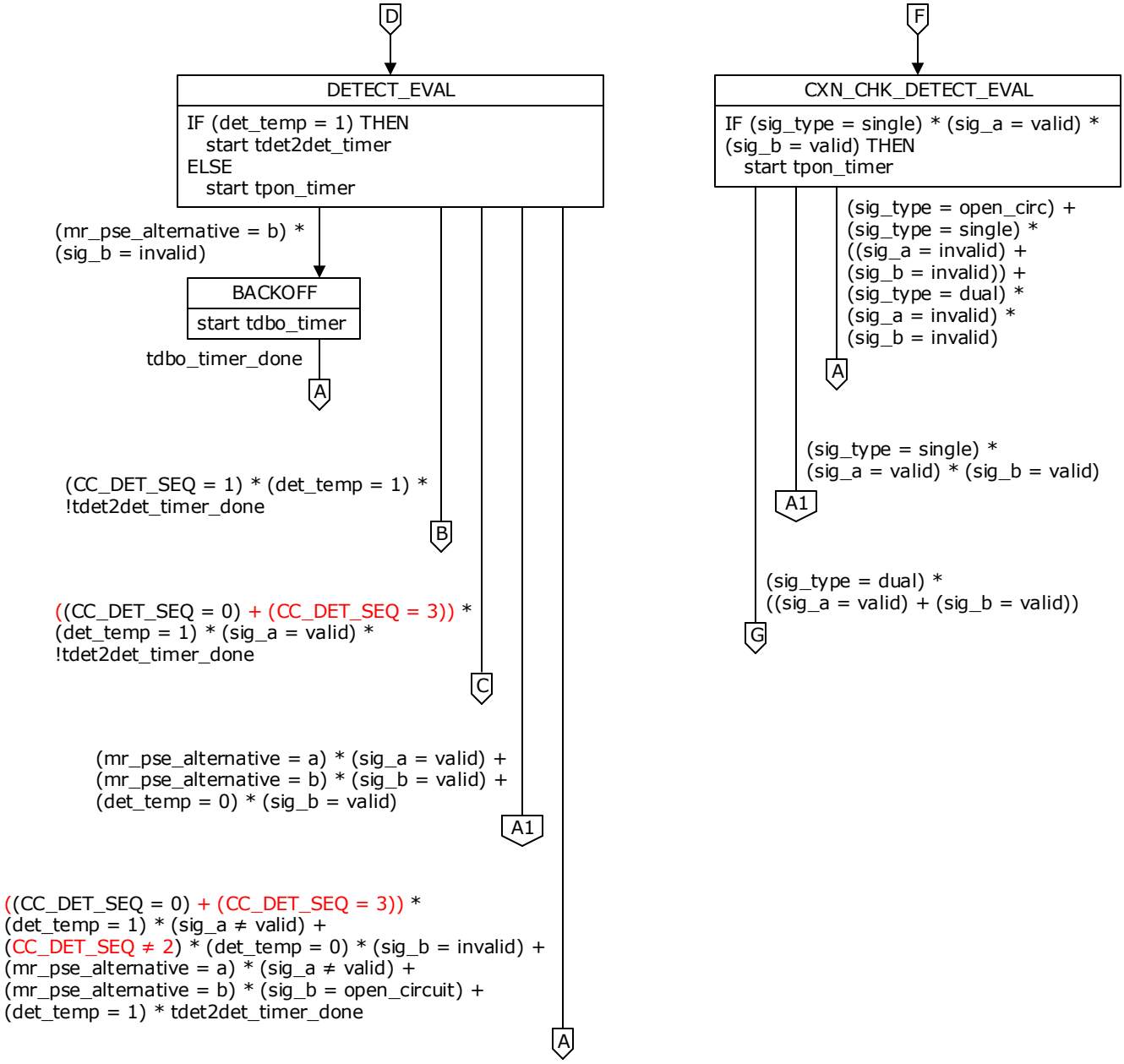
Treatment of Alt-A and Alt-B are interchangeable for the purposes of the State Diagram

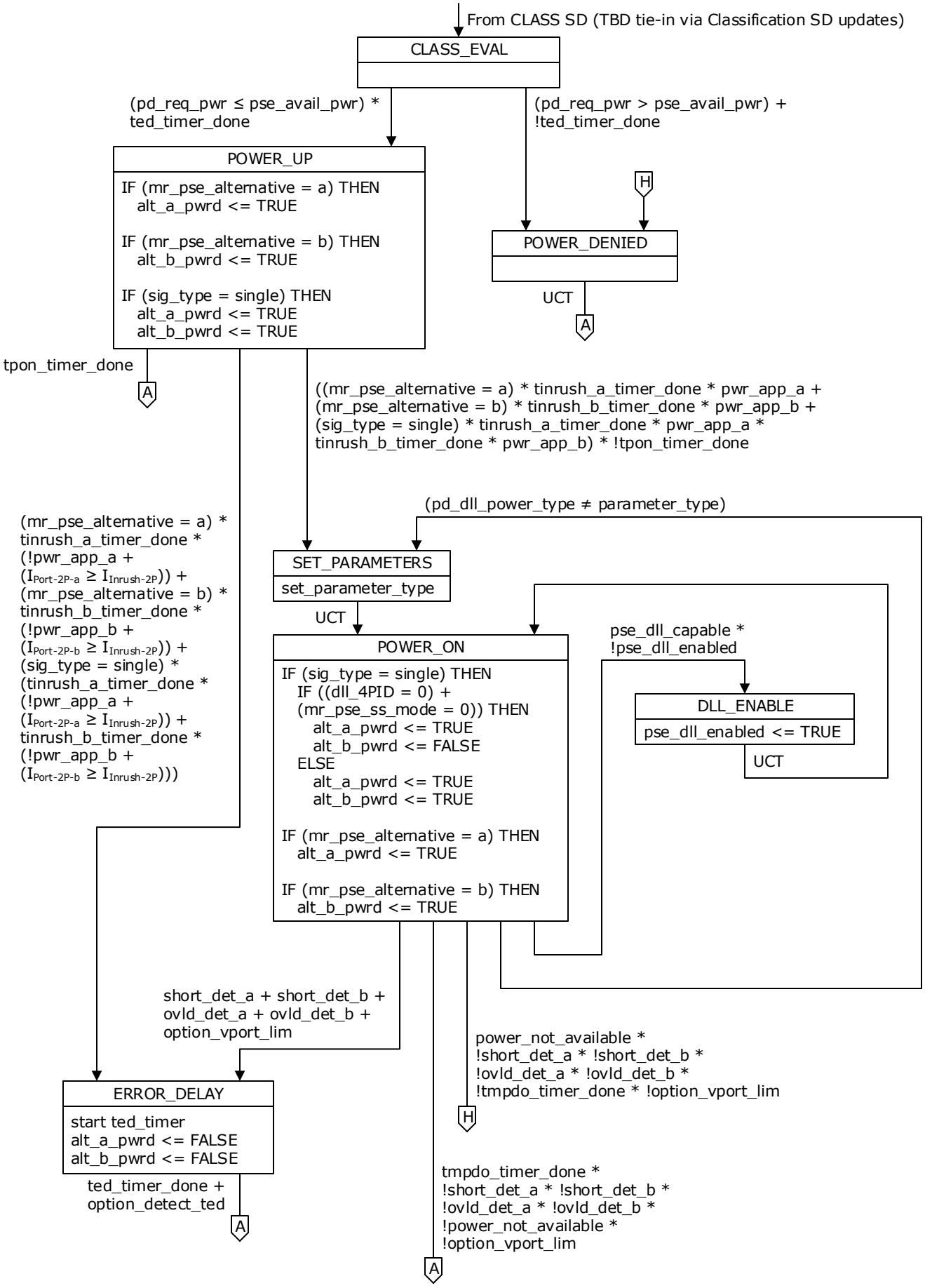
Alternative A/B Parity text:

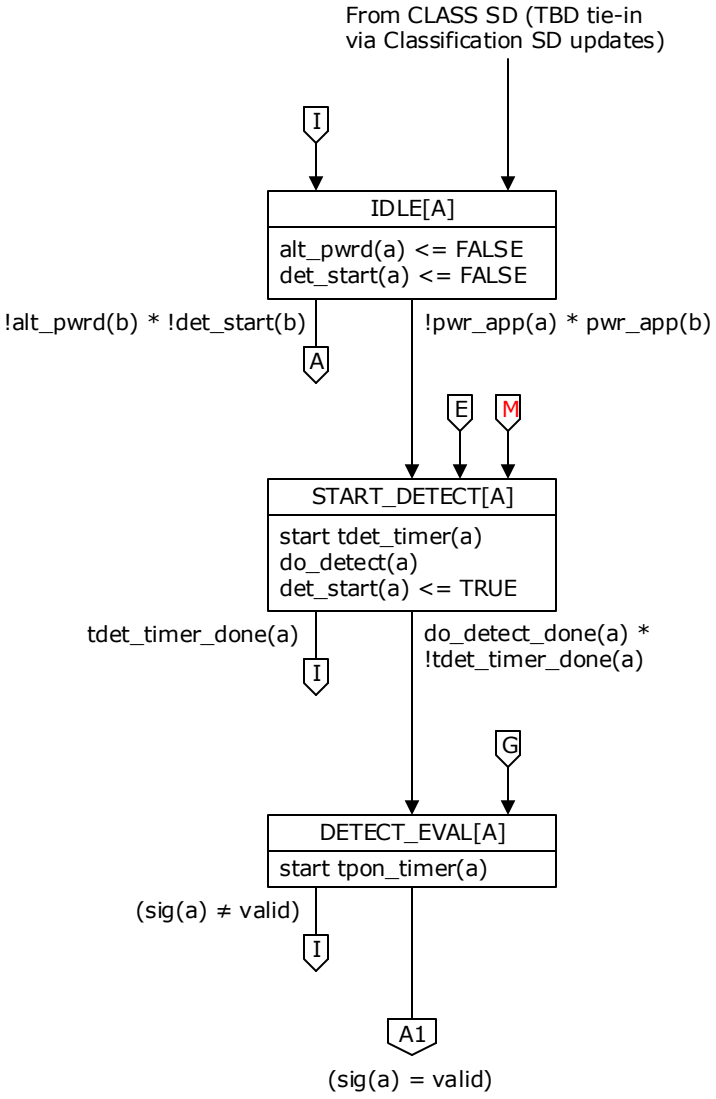
Add the following text in section 33.2.4.1:

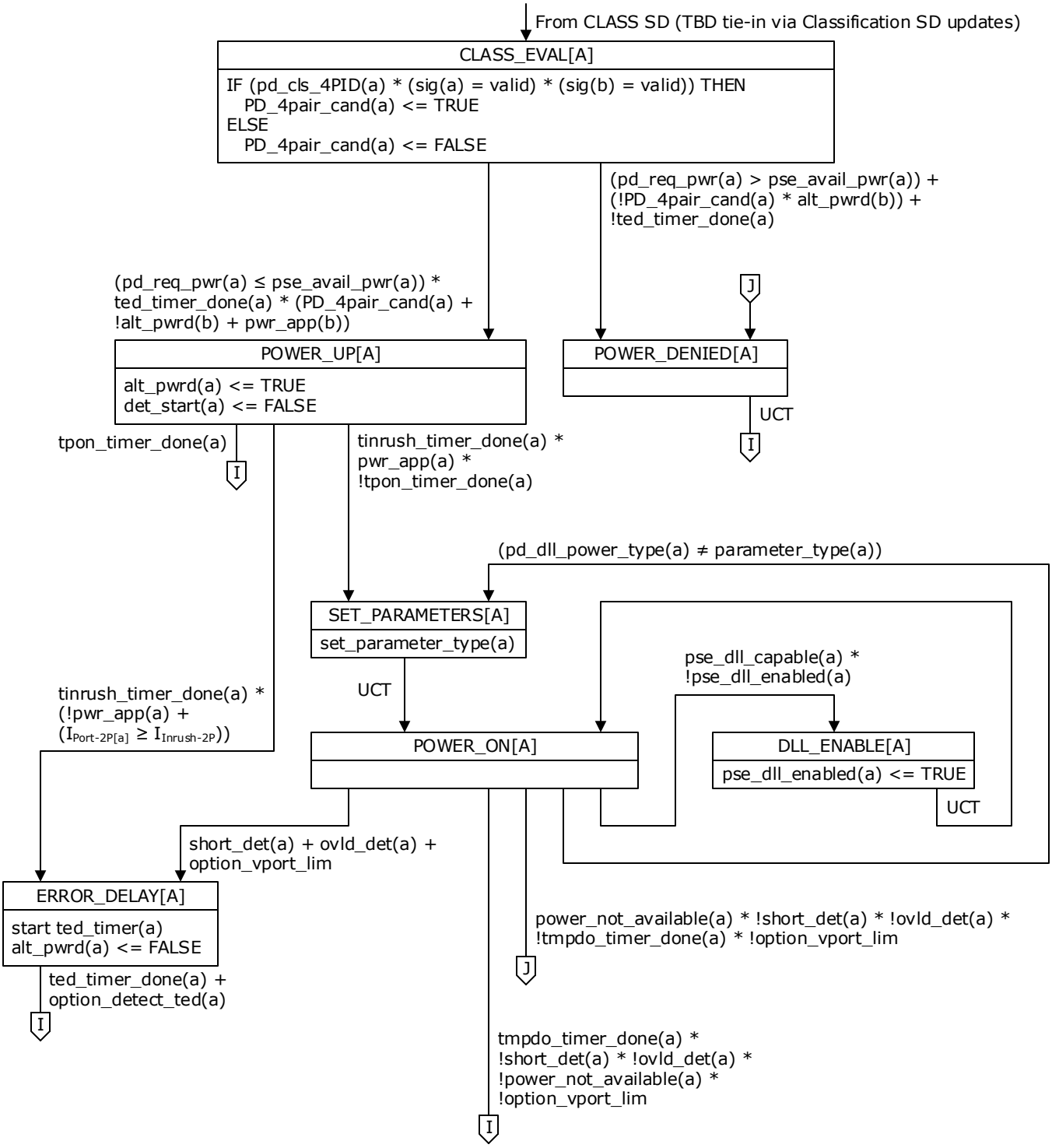
“In the Type 3 and Type 4 state diagram, Alternative A and Alternative B are depicted as serving distinct roles during 4-pair operation. In any implementation, the behaviors of the Alternatives may be reversed as long as the roles are established in IDLE and shall be maintained in every other state.”

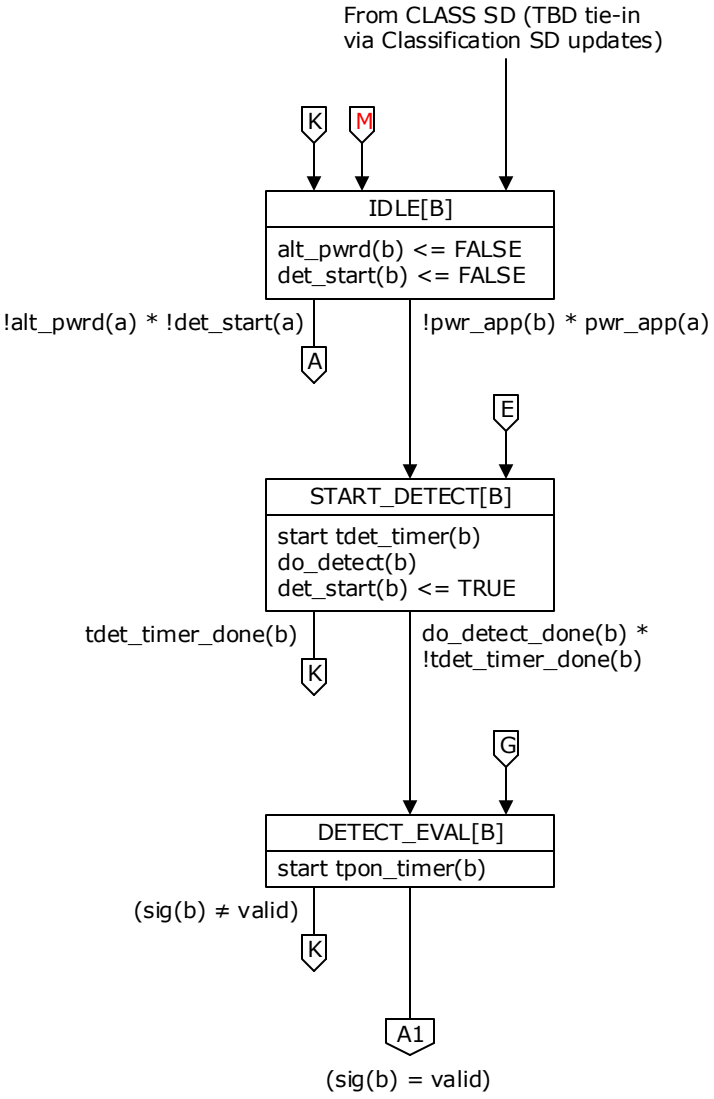

mr_pse_enable = disable
DISABLED
alt_a_pwrd <= FALSE
alt_b_pwrd <= FALSE
mr_pse_enable = enable
From CLASS SD (TBD tie-in via Classification SD updates)
pse_reset + error_condition * (mr_pse_enable = enable)
A
IDLE
alt_a_pwrd <= FALSE
alt_b_pwrd <= FALSE
mr_valid_signature <= FALSE
sig_type <= open_circ
det_temp <= 0
pse_dll_enabled <= FALSE
(mr_pse_enable = force_power) * !error_condition * !(ovld_det_a + short_det_a) * !(ovld_det_b + short_det_b)
TEST_MODE
IF (mr_force_pwr_a) THEN
alt_a_pwrd <= TRUE
IF (mr_force_pwr_b) THEN
alt_b_pwrd <= TRUE
mr_pse_enable = enable
A
((CC_DET_SEQ = 0) + (CC_DET_SEQ = 3)) * pse_ready * !(pwr_app_a + pwr_app_b) * (mr_pse_enable = enable)
(mr_pse_enable = force_power) * (ovld_det_a + short_det_a + ovld_det_b + short_det_b)
B
START_CXN_CHK
start tcc_timer
do_cxn_chk
TEST_ERROR
alt_a_pwrd <= FALSE
alt_b_pwrd <= FALSE
mr_pse_enable = enable
A
((CC_DET_SEQ = 1) + (mr_pse_alternative = a) + (mr_pse_alternative = b)) * pse_ready * !(pwr_app_a + pwr_app_b) * (mr_pse_enable = enable)
C
do_cxn_chk_done * (tcc_timer > tcc_min)
CXN_CHK_EVAL
IF (CC_DET_SEQ = 0) + (CC_DET_SEQ = 3) THEN
start tcc2det_timer
START_DETECT
start tdet_timer
IF (CC_DET_SEQ ≠ 2) THEN
IF (det_temp = 0) THEN
do_detect_a
det_temp <= 1
ELSE
do_detect_b
det_temp <= 0
IF (mr_pse_alternative = a) THEN
do_detect_a
IF (mr_pse_alternative = b) THEN
do_detect_b
(CC_DET_SEQ = 2) * pse_ready * !(pwr_app_a + pwr_app_b) * (mr_pse_enable = enable)
START_CXN_CHK_DETECT
start tdet_timer
do_cxn_chk
do_detect_a
do_detect_b
(sig_type = single) * (((CC_DET_SEQ = 0) + (CC_DET_SEQ = 3)) * !tcc2det_timer_done + (CC_DET_SEQ = 1) * (sig_a = valid) * !tdet2det_timer_done)
C
tdet_timer_done
A
tdet_timer_done
A
(sig_type = dual) * ((CC_DET_SEQ = 0) * !tcc2det_timer_done + (CC_DET_SEQ = 1) * !tdet2det_timer_done)
E
D
(do_detect_a_done * (det_temp = 1) + do_detect_b_done + (mr_pse_alternative = a) * do_detect_a_done) * !tdet_timer_done
do_cxn_chk_done * do_detect_a_done * do_detect_b_done * !tdet_timer_done
F
(sig_type = dual) * (CC_DET_SEQ = 3) * !tcc2det_timer_done
M
(sig_type = open_circ) + (sig_type = single) * (CC_DET_SEQ = 1) * (sig_a = invalid) + tcc2det_timer_done + tdet2det_timer_done
A

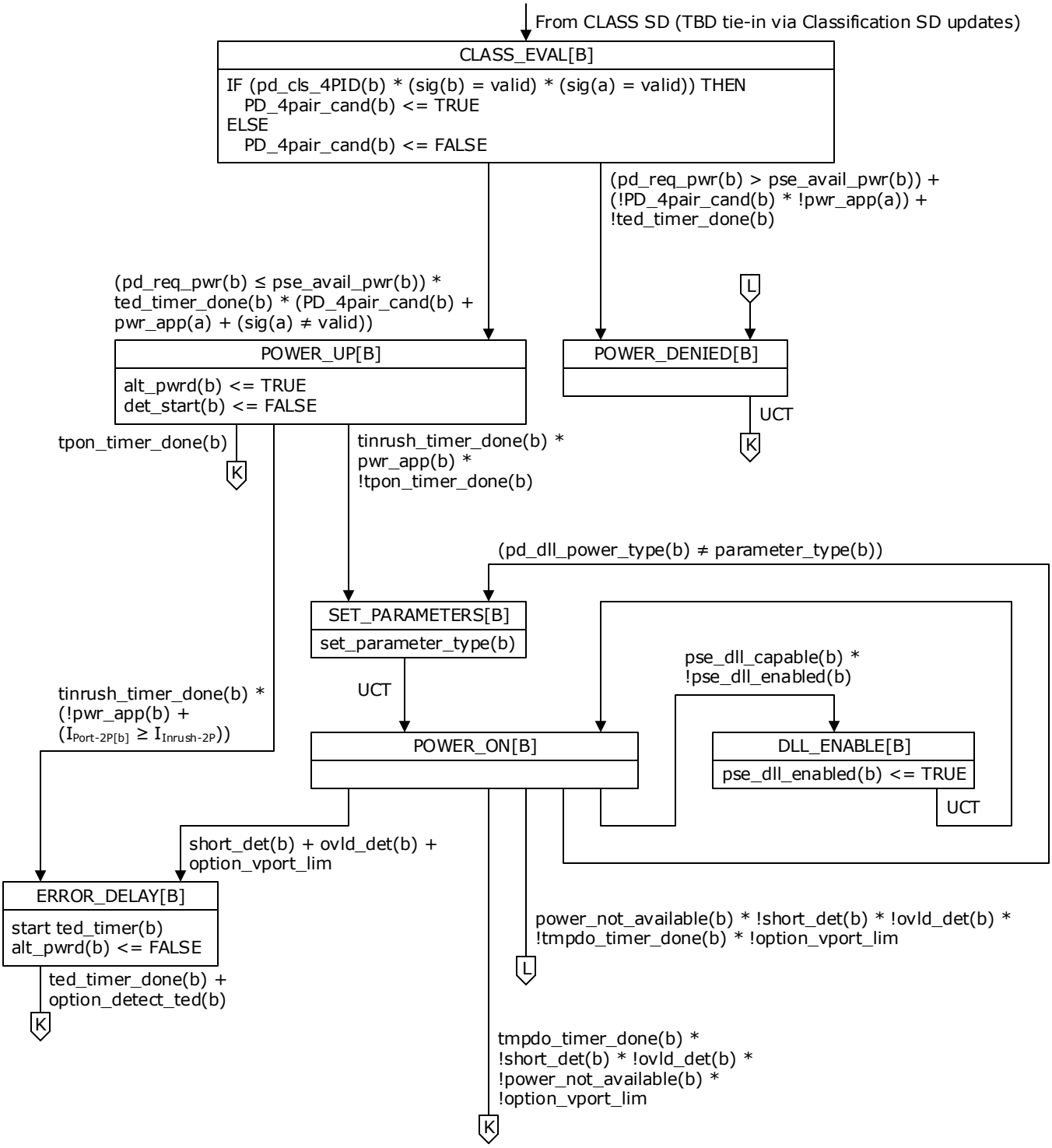

D
DETECT_EVAL
IF (det_temp = 1) THEN
start tdet2det_timer
ELSE
start tpon_timer
(mr_pse_alternative = b) *
(sig_b = invalid)
BACKOFF
start tdbo_timer
tdbo_timer_done
A
(CC_DET_SEQ = 1) * (det_temp = 1) *
!tdet2det_timer_done
B
((CC_DET_SEQ = 0) + (CC_DET_SEQ = 3)) *
(det_temp = 1) * (sig_a = valid) *
!tdet2det_timer_done
C
(mr_pse_alternative = a) * (sig_a = valid) +
(mr_pse_alternative = b) * (sig_b = valid) +
(det_temp = 0) * (sig_b = valid)
A1
((CC_DET_SEQ = 0) + (CC_DET_SEQ = 3)) *
(det_temp = 1) * (sig_a ≠ valid) +
(CC_DET_SEQ ≠ 2) * (det_temp = 0) * (sig_b = invalid) +
(mr_pse_alternative = a) * (sig_a ≠ valid) +
(mr_pse_alternative = b) * (sig_b = open_circuit) +
(det_temp = 1) * tdet2det_timer_done
A
F
CXN_CHK_DETECT_EVAL
IF (sig_type = single) * (sig_a = valid) *
(sig_b = valid) THEN
start tpon_timer
(sig_type = open_circ) +
(sig_type = single) *
((sig_a = invalid) +
(sig_b = invalid)) +
(sig_type = dual) *
(sig_a = invalid) *
(sig_b = invalid)
A
(sig_type = single) *
(sig_a = valid) * (sig_b = valid)
A1
(sig_type = dual) *
((sig_a = valid) + (sig_b = valid))
G


From CLASS SD (TBD tie-in via Classification SD updates)
CLASS_EVAL
(pd_req_pwr ≤ pse_avail_pwr) *
ted_timer_done
(pd_req_pwr > pse_avail_pwr) +
!ted_timer_done
POWER_UP
IF (mr_pse_alternative = a) THEN
alt_a_pwrd <= TRUE
IF (mr_pse_alternative = b) THEN
alt_b_pwrd <= TRUE
IF (sig_type = single) THEN
alt_a_pwrd <= TRUE
alt_b_pwrd <= TRUE
H
POWER_DENIED
UCT
A
tpon_timer_done
A
((mr_pse_alternative = a) * tinrush_a_timer_done * pwr_app_a +
(mr_pse_alternative = b) * tinrush_b_timer_done * pwr_app_b +
(sig_type = single) * tinrush_a_timer_done * pwr_app_a *
tinrush_b_timer_done * pwr_app_b) * !tpon_timer_done
(pd_dll_power_type ≠ parameter_type)
(mr_pse_alternative = a) *
tinrush_a_timer_done *
(!pwr_app_a +
(I_Port-2P-a ≥ I_Inrush-2P)) +
(mr_pse_alternative = b) *
tinrush_b_timer_done *
(!pwr_app_b +
(I_Port-2P-b ≥ I_Inrush-2P)) +
(sig_type = single) *
(tinrush_a_timer_done *
(!pwr_app_a +
(I_Port-2P-a ≥ I_Inrush-2P)) +
tinrush_b_timer_done *
(!pwr_app_b +
(I_Port-2P-b ≥ I_Inrush-2P)))
SET_PARAMETERS
set_parameter_type
UCT
POWER_ON
IF (sig_type = single) THEN
IF ((dll_4PID = 0) +
(mr_pse_ss_mode = 0)) THEN
alt_a_pwrd <= TRUE
alt_b_pwrd <= FALSE
ELSE
alt_a_pwrd <= TRUE
alt_b_pwrd <= TRUE
IF (mr_pse_alternative = a) THEN
alt_a_pwrd <= TRUE
IF (mr_pse_alternative = b) THEN
alt_b_pwrd <= TRUE
pse_dll_capable *
!pse_dll_enabled
DLL_ENABLE
pse_dll_enabled <= TRUE
UCT
short_det_a + short_det_b +
ovld_det_a + ovld_det_b +
option_vport_lim
power_not_available *
!short_det_a * !short_det_b *
!ovld_det_a * !ovld_det_b *
!tmpdo_timer_done * !option_vport_lim
H
ERROR_DELAY
start ted_timer
alt_a_pwrd <= FALSE
alt_b_pwrd <= FALSE
ted_timer_done +
option_detect_ted
A
tmpdo_timer_done *
!short_det_a * !short_det_b *
!ovld_det_a * !ovld_det_b *
!power_not_available *
!option_vport_lim
A


From CLASS SD (TBD tie-in
via Classification SD updates)
I
IDLE[A]
alt_pwrd(a) <= FALSE
det_start(a) <= FALSE
!alt_pwrd(b) * !det_start(b)
A
!pwr_app(a) * pwr_app(b)
E
M
START_DETECT[A]
start tdet_timer(a)
do_detect(a)
det_start(a) <= TRUE
tdet_timer_done(a)
I
do_detect_done(a) *
!tdet_timer_done(a)
G
DETECT_EVAL[A]
start tpon_timer(a)
(sig(a) ≠ valid)
I
A1
(sig(a) = valid)


From CLASS SD (TBD tie-in via Classification SD updates)
CLASS_EVAL[A]
IF (pd_cls_4PID(a) * (sig(a) = valid) * (sig(b) = valid)) THEN
PD_4pair_cand(a) <= TRUE
ELSE
PD_4pair_cand(a) <= FALSE
(pd_req_pwr(a) > pse_avail_pwr(a)) +
(!PD_4pair_cand(a) * alt_pwrd(b)) +
!ted_timer_done(a)
(pd_req_pwr(a) ≤ pse_avail_pwr(a)) *
ted_timer_done(a) * (PD_4pair_cand(a) +
!alt_pwrd(b) + pwr_app(b))
J
POWER_UP[A]
alt_pwrd(a) <= TRUE
det_start(a) <= FALSE
POWER_DENIED[A]
UCT
I
tpon_timer_done(a)
I
tinrush_timer_done(a) *
pwr_app(a) *
!tpon_timer_done(a)
(pd_dll_power_type(a) ≠ parameter_type(a))
SET_PARAMETERS[A]
set_parameter_type(a)
UCT
pse_dll_capable(a) *
!pse_dll_enabled(a)
tinrush_timer_done(a) *
(!pwr_app(a) +
(I_Port-2P[a] ≥ I_Inrush-2P))
POWER_ON[A]
DLL_ENABLE[A]
pse_dll_enabled(a) <= TRUE
UCT
short_det(a) + ovld_det(a) +
option_vport_lim
ERROR_DELAY[A]
start ted_timer(a)
alt_pwrd(a) <= FALSE
power_not_available(a) * !short_det(a) * !ovld_det(a) *
!tmpdo_timer_done(a) * !option_vport_lim
J
ted_timer_done(a) +
option_detect_ted(a)
I
tmpdo_timer_done(a) *
!short_det(a) * !ovld_det(a) *
!power_not_available(a) *
!option_vport_lim
I


From CLASS SD (TBD tie-in
via Classification SD updates)
K
M
IDLE[B]
alt_pwrd(b) <= FALSE
det_start(b) <= FALSE
!alt_pwrd(a) * !det_start(a)
A
!pwr_app(b) * pwr_app(a)
E
START_DETECT[B]
start tdet_timer(b)
do_detect(b)
det_start(b) <= TRUE
tdet_timer_done(b)
K
do_detect_done(b) *
!tdet_timer_done(b)
G
DETECT_EVAL[B]
start tpon_timer(b)
(sig(b) ≠ valid)
K
A1
(sig(b) = valid)


From CLASS SD (TBD tie-in via Classification SD updates)
CLASS_EVAL[B]
IF (pd_cls_4PID(b) * (sig(b) = valid) * (sig(a) = valid)) THEN
PD_4pair_cand(b) <= TRUE
ELSE
PD_4pair_cand(b) <= FALSE
(pd_req_pwr(b) > pse_avail_pwr(b)) +
(!PD_4pair_cand(b) * !pwr_app(a)) +
!ted_timer_done(b)
(pd_req_pwr(b) ≤ pse_avail_pwr(b)) *
ted_timer_done(b) * (PD_4pair_cand(b) +
pwr_app(a) + (sig(a) ≠ valid))
L
POWER_UP[B]
alt_pwrd(b) <= TRUE
det_start(b) <= FALSE
POWER_DENIED[B]
UCT
K
tpon_timer_done(b)
K
tinrush_timer_done(b) *
pwr_app(b) *
!tpon_timer_done(b)
(pd_dll_power_type(b) ≠ parameter_type(b))
SET_PARAMETERS[B]
set_parameter_type(b)
UCT
pse_dll_capable(b) *
!pse_dll_enabled(b)
tinrush_timer_done(b) *
(!pwr_app(b) +
(I_Port-2P[b] ≥ I_Inrush-2P))
POWER_ON[B]
DLL_ENABLE[B]
pse_dll_enabled(b) <= TRUE
UCT
short_det(b) + ovld_det(b) +
option_vport_lim
ERROR_DELAY[B]
start ted_timer(b)
alt_pwrd(b) <= FALSE
power_not_available(b) * !short_det(b) * !ovld_det(b) *
!tmpdo_timer_done(b) * !option_vport_lim
L
ted_timer_done(b) +
option_detect_ted(b)
K
tmpdo_timer_done(b) *
!short_det(b) * !ovld_det(b) *
!power_not_available(b) *
!option_vport_lim
K