1)
* Add a new variable to 33.3.3.7:
  invalid_class_det: A variable indicating if **any IClass measured by the PSE** during do_classification **is invalid or equal to or greater than** IClass_LIM **min** as defined in Table 33-15.
    TRUE: **Measured IClass is invalid or equal to or greater than** IClass_LIM **min** during do_classification.
    FALSE: **Measured IClass is not invalid or is less than** IClass_LIM **min** during do_classification.
* Modify the asynchronous entry arc transition logic into IDLE as follows: "(mr_pse_enable = enable) * (pse_reset + iclass_lim_det + error_condition)"
* Add iclass_lim_det <= FALSE to IDLE

2)
* Add a new variable to 33.3.3.7:
  iclass_lim_det_pri: A variable indicating if **any IClass measured by the PSE** over the primary alternative during do_classification_pri **is invalid or equal to or greater than** IClass_LIM **min** as defined in Table 33-15.
    TRUE: **Measured IClass over primary alternative is invalid or equal to or greater than** IClass_LIM **min** during do_classification_pri.
    FALSE: **Measured IClass over primary alternative is not invalid or is less than** IClass_LIM **min** during do_classification_pri.
* Add an asynchronous entry arc into IDLE_PRI with transition logic "iclass_lim_det_pri"
* Add iclass_lim_det_pri <= FALSE to ENTRY_PRI and IDLE_PRI

3)
* Add a new variable to 33.3.3.7:
  iclass_lim_det_sec: A variable indicating if **any IClass measured by the PSE** over the secondary alternative during do_classification_sec **is invalid or equal to or greater than** IClass_LIM **min** as defined in Table 33-15.
    TRUE: **Measured IClass over secondary alternative is invalid or equal to or greater than** IClass_LIM **min** during do_classification_sec.
    FALSE: **Measured IClass over secondary alternative is not invalid or is less than** IClass_LIM **min** during do_classification_sec.
* Add an asynchronous entry arc into IDLE_SEC with transition logic "iclass_lim_det_sec"
* Add iclass_lim_det_sec <= FALSE to ENTRY_SEC and IDLE_SEC